

# DPL™ 9

## User Guide



[www.syncopation.com](http://www.syncopation.com)

Copyright © 2017 Syncopation Software, Inc. All rights reserved.

Printed in the United States of America.

Revised April 2017.

# Table of Contents

<b>1. Introduction .....</b>	<b>7</b>
1.1 DPL™ 9 User Guide Overview.....	7
1.2 Installing and Activating Your DPL™ License .....	12
1.3 DPL™ Help and Resources .....	17
1.4 A Brief Tour of DPL™ .....	22
1.5 What's New in DPL™ 9 .....	41
<b>2. Building a Model in Decision Tree-focused Mode.....</b>	<b>43</b>
2.1 Decision Tree Overview .....	43
2.2 Symmetric vs. Asymmetric Trees .....	45
2.3 Adding a Decision Node .....	50
2.4 Adding a Chance Node.....	54
2.5 Adding Probabilistic Conditioning .....	68
2.6 Linking Decision Tree Events to Excel .....	71
2.7 Creating and Linking the Output Metric.....	73
2.8 Running a Preliminary Analysis.....	76
2.9 Adding a Downstream Decision .....	81
2.10 Ignore Conditioning in Unknown State.....	84
<b>3. Analyzing a Decision Analysis Model .....</b>	<b>87</b>
3.1 Policy Tree™ .....	92
3.2 Risk Profile.....	105
3.3 Policy Summary™.....	111
3.4 Saving Outputs within a Workspace .....	113
3.5 Expected Value of Perfect Information / Control .....	115
3.6 Value Correlations .....	119
3.7 Copying/Pasting to a Presentation .....	121
<b>4. Building a Model in Influence Diagram-focused Mode.....</b>	<b>124</b>
4.1 Value Nodes and Spreadsheet Links.....	124
4.2 Influence Arcs .....	133
4.3 Running an Initial Analysis .....	136
4.4 Decision Nodes.....	137
<b>5. Conducting Sensitivity Analyses.....</b>	<b>153</b>
5.1 Value Tornado Diagrams.....	153
5.2 Formatting Tornado Diagrams and Other Charts.....	158
5.3 Discrete Chance Nodes .....	161
5.4 Base Case Tornado Diagrams.....	167
5.5 Initial Decision Alternatives Tornado Diagrams .....	171
5.6 Adding and Analyzing a Downstream Decision .....	173
5.7 Rainbow Diagram on a Value .....	189
5.8 Rainbow Diagram on a Probability .....	193
5.9 Probabilistic Base Case Tornado Diagram .....	196
<b>6. Building and Analyzing Monte Carlo Simulation Models.....</b>	<b>199</b>
6.1 The Excel Cash Flow Model .....	200
6.2 Creating Linked Values.....	203

6.3	Running a Value Tornado Diagram.....	208
6.4	Using the Workspace Manager.....	211
6.5	Continuous Chance Nodes.....	212
6.6	Comparing Risk Profiles.....	217
6.7	Modeling an Up-Front Decision.....	221
6.8	Performing a Sensitivity Analysis.....	229
6.9	Modeling a Downstream Decision.....	231
<b>7.</b>	<b>Conditioning and Learning in Decision Models.....</b>	<b>236</b>
7.1	Incorporating Imperfect Information in a Model.....	236
7.2	Perform Subtree and Continue.....	250
7.3	Summary of DPL™ Influence Arcs.....	254
<b>8.</b>	<b>Incorporating Multiple Attributes and Objective Functions.....</b>	<b>256</b>
8.1	Incorporating Multiple Attributes.....	256
8.2	Using a Constraint Function.....	273
8.3	Modeling Multiple Objective Functions (Enterprise only).....	276
<b>9.</b>	<b>Multidimensional Value Nodes.....</b>	<b>291</b>
9.1	Creating and Linking a One-Dimensional Value Node.....	291
9.2	Creating and Linking a Two-Dimensional Value Node.....	307
<b>10.</b>	<b>Time Series Percentiles and Multiple Metrics.....</b>	<b>313</b>
10.1	Time Series Percentiles.....	313
10.2	Risk Profiles and Policy Trees™ for Multiple Attributes.....	333
10.3	Initial Decision Alternatives Time Series Percentiles.....	340
10.4	Using Multidimensional Value Nodes in Get/Pay Expressions.....	348
<b>11.</b>	<b>Using the Endpoint Database™.....</b>	<b>353</b>
11.1	Recording and Viewing Endpoints.....	354
11.2	Playing Endpoints.....	363
11.3	Reconnecting Endpoints to a Model.....	374
11.4	Exporting Endpoints.....	378
11.5	Importing Endpoints.....	382
11.6	Recording Endpoint in Parallel using Multiple Servers (Enterprise only).....	388
<b>12.</b>	<b>Advanced Decision Analysis Results.....</b>	<b>393</b>
12.1	Policy Tree™ Features.....	393
12.2	Subtree Risk Profile™.....	400
12.3	Policy Summary™ Features.....	404
12.4	Option Value Charts™.....	413
<b>13.</b>	<b>Risk Tolerance.....</b>	<b>417</b>
13.1	Incorporating a Risk Tolerance.....	417
13.2	Advanced Utility Functions.....	424
<b>14.</b>	<b>Advanced Sensitivity Analyses.....</b>	<b>426</b>
14.1	Two-Way Rainbow Diagrams.....	426
14.2	Event Tornadoes.....	434
14.3	When to Use Which DPL™ Sensitivity Output.....	438

<b>15. Pruned Sequential Asymmetric Trees (Enterprise only)</b> .....	<b>441</b>
15.1 Tutorial: Modeling Sequential Drill Decisions.....	443
<b>16. Converting Spreadsheets to DPL™ Code</b> .....	<b>452</b>
16.1 How to Convert Spreadsheets .....	452
16.2 Spreadsheet Practices for Easier Conversion.....	461
<b>17. DPL™ Programs and Code</b> .....	<b>463</b>
17.1 What is a DPL™ Program? .....	463
17.2 Overview of DPL Program Components .....	464
17.3 Converting a Model to a Program .....	466
17.4 Defining Components in the Definition Section .....	471
17.5 Defining Event Sequences in the Sequence Section .....	484
17.6 Advanced Techniques for Programs .....	493
<b>18. Running Excel Macros from DPL™ (Enterprise only)</b> .....	<b>497</b>
18.1 When to Use Excel Macros .....	497
18.2 Tutorial: Building a DPL™ Model for a Spreadsheet Updated by a Macro .....	497
<b>19. Multiple Experts (Enterprise only)</b> .....	<b>512</b>
19.1 Why Use Multiple Experts? .....	512
19.2 Overview of DPL's™ Multiple Experts Feature .....	513
19.3 Tutorial: Using Multiple Experts to Assess Early Product Approval .....	516
<b>20. Estimating Probabilities from Data (Enterprise only)</b> .....	<b>522</b>
20.1 Tutorial: Moving from Analyzing Datasets to Decisions.....	522
<b>21. Database Linking in DPL™ (Enterprise only)</b> .....	<b>543</b>
21.1 Overview .....	543
21.2 ODBC Data Sources .....	543
21.3 Database Tables in DPL™ .....	547
21.4 Configuring Database Access within DPL™ .....	555
21.5 Loading Database Schema .....	559
21.6 Creating Database-Linked Models .....	560
21.7 Databases Configured for Revision Tracking .....	579
<b>22. DPL™ Developer API (Enterprise only)</b> .....	<b>580</b>
22.1 Overview .....	580
22.2 Controlling DPL™ from Visual Basic .....	581
22.3 API Objects and Types.....	588
22.4 API Reference .....	592
<b>23. DPL™ User Function Libraries (Enterprise only)</b> .....	<b>614</b>
23.1 Overview .....	614
23.2 Technical Considerations.....	614
23.3 Implicit Functions .....	615
23.4 Explicit Functions.....	619
23.5 DPL™ Callback Functions .....	622
23.6 Code Examples.....	627

---

<b>A Overview of Spreadsheet Linking .....</b>	<b>632</b>
A.1 Types of Spreadsheet Links in a DPL™ Model .....	632
A.2 Calculation Links .....	633
A.3 Initialization Links .....	641
A.4 Managing Spreadsheet Links.....	650
<b>B Using Strategy Tables .....</b>	<b>652</b>
<b>C System Requirements and Compatibility with Older Releases.....</b>	<b>668</b>
<b>D Keyboard Shortcuts .....</b>	<b>669</b>
<b>E Glossary of DPL™ and Decision Analysis Terms.....</b>	<b>670</b>
<b>Index.....</b>	<b>689</b>

# 1. Introduction

Congratulations on your purchase of DPL 9!

We're confident that the game-changing analytics, polished modeling interface, and high level of performance offered by the DPL 9 Release will exceed your expectations. If you've used DPL in the past, you'll find that the ease-of-use improvements, performance enhancements, and improved interoperability in this release will serve as a major boost to your productivity.

The DPL modeling environment was designed to give you the capabilities you need to maintain focus on the problem at hand -- the decision and not the tools. This guide is an extension of that perspective, teaching you essential DPL skills in the context of simple but realistic decision analytic models.

The first section provides specific information about the tutorial contained within this user guide. The subsequent sections of this chapter describe how to get help and access resources, show you how to install and activate your DPL license, and provide a brief tour of the DPL user interface. The final section lists the major changes and new features included in the DPL 9 Release.

## 1.1 DPL™ 9 User Guide Overview

---

This DPL 9 User Guide covers the features of both the DPL Professional and DPL Enterprise versions and is divided into 22 tutorial chapters plus appendices.

Chapters 2 through 5 cover the basic modeling features and primary outputs of DPL. These chapters were written specifically with users who are new to DPL in mind. Various sections of these chapters do, however, assume a basic knowledge of decision analysis, Excel, and valuation techniques. If you are new to DPL, you should find that these chapters provide a thorough introduction to the skills you will need to use the basic functionality of the software effectively and work your way through the advanced tutorials contained in later chapters of the guide.

Chapters 6 through 17 cover more advanced modeling features offered by the DPL Professional version. Some of these chapters include features that are offered by the DPL Enterprise version only. They further assume you

have some basic experience using DPL, as well as familiarity with decision and risk analysis concepts, Excel, Monte Carlo simulation and valuation techniques. If you are new to DPL, you should complete the tutorials in Chapters 2 through 5 of the guide before covering the material in the remaining chapters. In general, Chapters 6 through 17 are intended to be "stand-alone", although new users may find that Chapters 8, 9, and 10 are best read sequentially, as discussed below.

Chapters 18 through 23 are also intended to be "stand-alone" with each covering features offered by the DPL Enterprise version. If you are working through the guide with the DPL Professional version you will not be able to complete the tutorials contained within these chapters (the same goes for earlier sections designated "Enterprise only"). Contact the Syncopation Sales Team if you have the DPL Professional version but find that you have a need for one of the features only offered by the DPL Enterprise version.

### 1.1.1 DPL™ 9 User Guide Chapter Descriptions

This guide focuses on DPL's graphical model development environment and is intended to be read while working with DPL. The chapters contain tutorials that will give you the essential tools needed to develop real world decision models in DPL.

Chapters 2 and 3 contain a tutorial for building a model in Decision Tree-focused mode starting from a spreadsheet that analyzes a drug development opportunity. These two chapters together cover a broad range of basic Decision Tree modeling skills and cover five of DPL's primary analytic outputs: Policy Tree, Risk Profile, Policy Summary, VOIC, and Value Correlations.

Chapters 4 and 5 contain a tutorial in which a model is built in Influence Diagram-focused mode from a spreadsheet representing the business plan for a new product. Similar to the two previous chapters, Chapters 4 and 5 taken together cover all basic Influence Diagram modeling techniques and introduces a variety of DPL sensitivity analyses.

*Note: You'll find that most of the later chapters assume you have already been through the tutorials contained in Chapters 2 through 5 of this manual, or that you are already familiar with the DPL modeling environment.*

Chapter 6 is a standalone chapter that contains a tutorial focused on building an Excel-linked DPL model for a financial risk analysis application. Continuous chance nodes are introduced to the model, Monte Carlo simulations are initiated and the analytic results are analyzed. This tutorial



assumes familiarity with cash flow spreadsheets and with the analysis of uncertainty by Monte Carlo simulation.

Chapter 7 leverages DPL's modeling tools and features for more complex decision models and situations, including probabilistic conditioning, Bayesian updating, and learning models. The last section includes a summary of DPL Influence Arc types.

Chapter 8 demonstrates how to create models with multiple attributes and combine these attributes into a single objective function. Multiple attributes can also be employed to track various metrics even if they aren't incorporated into the objective function. This use of multiple attributes is discussed further in Chapters 9 and 10. The use of constraint functions are also covered in the chapter. The last section contains a tutorial in which a model is modified to optimize multiple objective functions for an attacker/defender scenario.

Chapter 9 demonstrates how to use multidimensional value nodes in the Influence Diagram for driver and metric nodes. The tutorial within the first section will look at an example of a one-dimensional array metric node, a series metric node and a series driver node. The tutorial within the second section defines a two-dimensional array driver node that is linked to Excel.

Chapter 10 contains a tutorial that uses a multidimensional value node in the generation of a Time Series Percentiles chart. Because Time Series Percentiles use multiple attributes, and can be used with multidimensional value nodes for greater convenience and efficiency, you may wish to review Chapters 8 and 9 before completing the tutorial in Chapter 10. Within the last section an Initial Decision Alternatives Time Series Percentiles chart is generated.

Chapter 11 covers the Endpoint Database feature in DPL. The Endpoint Database can be recorded (saved) and viewed. A recorded Endpoint Database can be re-played to quickly produce decision analysis results even if the model has been modified in certain ways. With large models, the Endpoint Database capability can significantly reduce the time required to do multiple runs in order to generate outputs. The last section includes a tutorial in which endpoints are recorded in parallel on multiple machines, resulting in quicker runs.

Chapter 12 covers the more advanced features of the DPL Policy Tree and Policy Summary. It also introduces two additional outputs: Subtree Risk Profiles and Option Value Charts.

Chapter 13 shows you how to use DPL to model attitudes toward risk and make decisions based on certain equivalent rather than expected value. Utility functions are introduced in this chapter.

Chapter 14 covers advanced sensitivity analysis features in DPL, including two-way Rainbow Diagrams and Event Tornado Diagrams. Chapter 14 also contains guidance about when to use each type of sensitivity analysis offered by DPL.

Chapters 16 and 17 describe and demonstrate how DPL programs and DPL code are created, and how spreadsheets linked to DPL can be converted to DPL code for significant improvements in model runtime.

*Note: Chapters 18 through 23 cover additional features of the DPL Enterprise version. They assume you have significant experience using DPL, and parts of it also assume familiarity with database and programming concepts.*

Chapter 18 illustrates how to link DPL Enterprise to a spreadsheet that contains a calculation macro.

Chapter 19 covers DPL Enterprise's expert aggregation interface and contains a tutorial on how to create models with expert aggregation nodes in them.

Chapter 20 covers DPL Enterprise's ability to estimate probabilities from an external dataset (CSV). It further demonstrates how users can progress from data analysis to decision making with a few modifications to the model.

Chapter 21 documents the DPL Enterprise version's database linking capabilities. You will complete a tutorial in which you set up and run a database-linked model.

Chapter 22 covers the Application Programming Interface (API). This feature allows you to control and run DPL from other applications, such as Visual Basic for Applications (VBA), C# and VB.NET.

Chapter 23 covers DPL's user function library interface.

Appendix A summarizes DPL's spreadsheet linking features and gives suggestions on when and how to use each feature.

Appendix B contains a tutorial on strategy tables, which allow you to "aggregate" several up-front (initial) decisions into a logical, manageable set of strategies to be analyzed.

Appendix C lists DPL's system requirements and compatibility with older releases.

Appendix D provides a comprehensive list of DPL's keyboard shortcut commands.

Appendix E provides a glossary of common DPL and Decision Analysis terms.

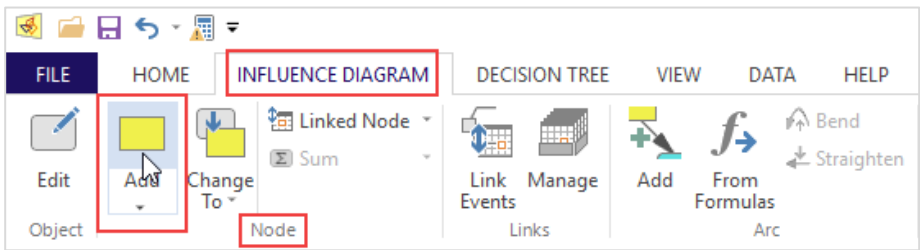
### 1.1.2 DPL™ 9 User Guide Conventions

A few conventions have been used in the text that follows. An instruction to you in a tutorial is contained in a bulleted paragraph with a block arrow:

⇒ Do this step now.

Information to be entered in edit boxes, Excel cells, etc. is contained within double-quotes. Do not include the double-quotes when entering the information.

DPL ribbon tab names, groups, and commands are separated by vertical bars (|). For example, "Influence Diagram | Node | Add" refers to the default command indicated by the icon in the Add split button, located in the Node group on the Influence Diagram tab of the command ribbon as show in Figure 1-1.



**Figure 1-1. Help Convention: Influence Diagram (Tab) | Node (Group) | Add (Command)**

For a split button, when a command other than the default indicated in the icon is desired you will be prompted to drop-down the split button and select the command from the drop-down list.

The word "Settings" following a vertical bar refers to the dialog box launcher (☰), which is located at the bottom right corner of the given tab group. For example, "Home | Run | Settings" refers to the dialog box launcher for the Run group on the Home tab of the ribbon and will launch the Run Settings Dialog.

If there are multiple tabs within a dialog, then the item following the last | refers to the desired tab. For example, File | Options | Outputs refers the Outputs tab of the File Options dialog.

## 1.2 Installing and Activating Your DPL™ License

---

### 1.2.1 Installing the DPL™ Application on Windows

If you have not already installed DPL on your computer, do so now.

For the DPL trial:

- ⇒ Click the link provided on the DPL 9 Professional Trial Download page or within the email regarding your trial code.


For purchased DPL licenses:

- ⇒ You should have received your DPL 9 software electronically via e-mail from sales@syncopation.com. Follow the instructions contained in the e-mail.
- ⇒ Once the DPL9.msi file has been downloaded and opened, follow the on-screen instructions.

Note: different instructions apply to installing DPL on a Mac. Refer to your license email or contact sales@syncopation.com for a Mac download link.

### 1.2.2 Activating Your DPL™ License

Once you've completed the download and installation, you'll need to activate your DPL license (trial or purchased) using the license codes or license file contained in the DPL license email. If you have been provided a license file, the email will contain instructions on where to place the file on your hard drive for activation. To activate with license codes:

- ⇒ Double-click the DPL 9 icon on your desktop (). The DPL License Setup dialog will appear (Figure 1-2).

DPL License Setup (Release 9.00.00) (Built Jan 11 2017)

License information

Please enter your DPL serial number and installation key below.

Serial number:  (15 digits)

Installation key:      (5\*4 digits)

Check License Info

License file

If you have a DPL license file, browse for it now.

Browse for License File

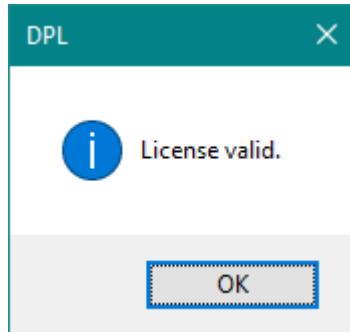
Cancel

**Figure 1-2. DPL License Setup dialog**

- ⇒ Carefully enter your Serial Number and Installation key within their respective fields and press the Check License Info button.

Note: if you are activating with a license file and you have placed the file in the correct location on your hard disk, this dialog will not appear as this step is bypassed when DPL finds the file. If you've been provided with a license file and this dialog appears, do not attempt to enter a serial number and install key. Instead, click on the Browse for License File under the *License File* section and navigate to the folder where you placed the license file.

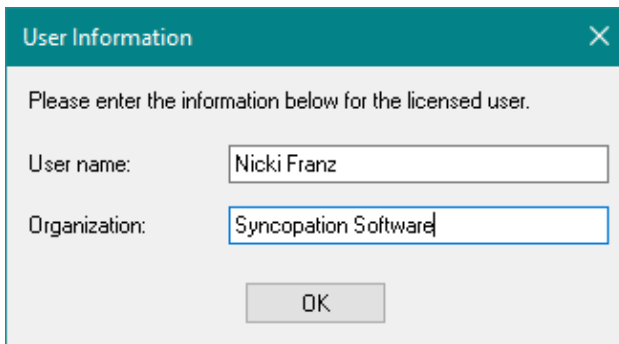
- ⇒ If the license information you've entered is valid, you will receive the following message:



**Figure 1-3. DPL License Valid dialog**

*Note: if the DPL License and/or Maintenance and Support coverage are due to expire in 30 days or less, you will receive a prompt notifying you of this each time you activate or open DPL.*

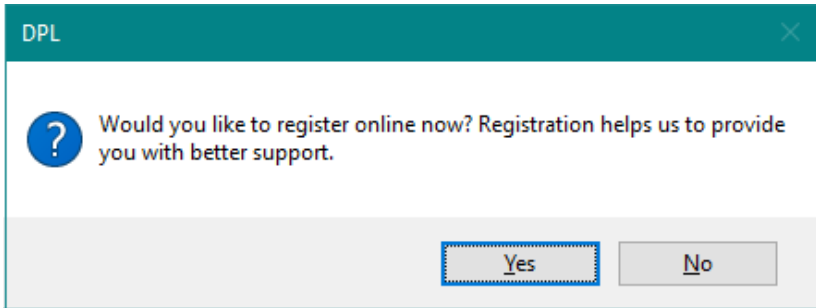
After clicking OK within the License Valid dialog, you are presented with the User Information dialog (Figure 1-4) in which you should enter the named user within the *User name* field and the licensing organization within the *Organization* field. The information entered will be displayed alongside other DPL license information within the Session Log (Figure 1-12) and About DPL dialog (Figure 1-10).



**Figure 1-4. DPL User Information dialog**

### 1.2.3 Registering Your DPL™ License

After clicking OK, DPL will ask if you'd like to register your license online with Syncopation Software (Figure 1-5).



**Figure 1-5. DPL License Registration prompt**

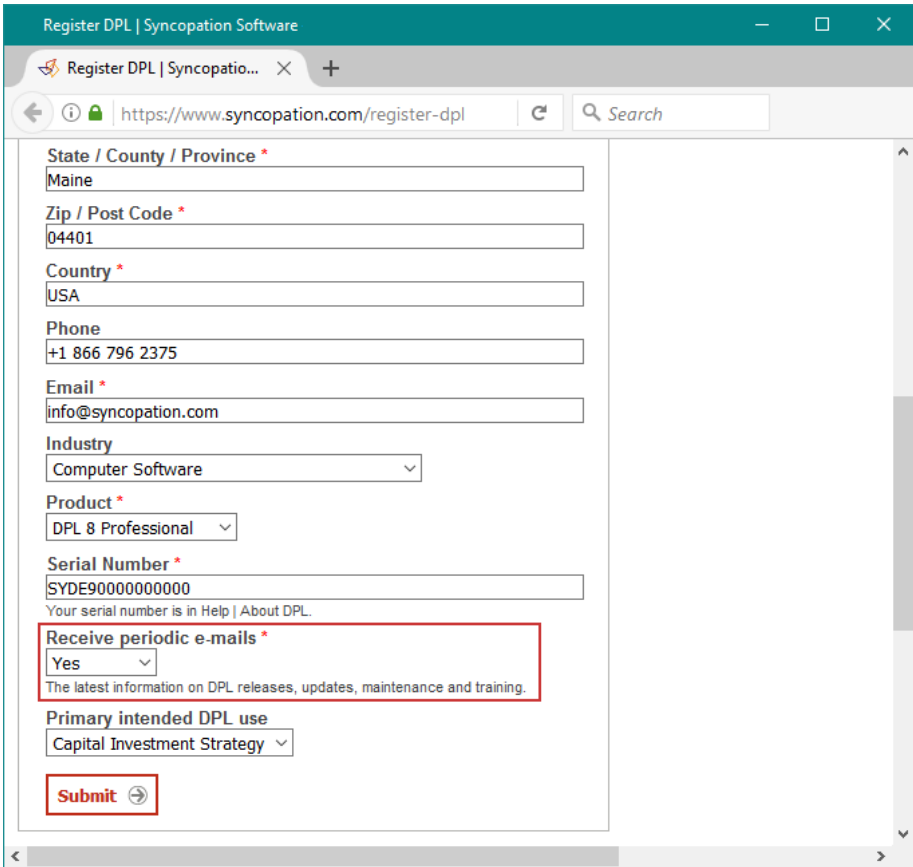
If you click No to the prompt DPL will let you know that you can register your license through Syncopation's website any time by selecting the Help | Resources | Register command (Figure 1-8) or by navigating to the following link directly and submitting the webform:

<https://www.syncopation.com/register-dpl>

If you haven't done so (or, if it's been a while) we recommend you register your DPL license to ensure Syncopation has your most current license and contact information. This helps us to provide you with the best user experience possible.

If you were to click Yes to the registration prompt a web browser will open to the Register DPL web form (Figure 1-6). Here you should enter your current contact and license information. One important required field near the bottom of the form asks you to specify your subscription preference, e.g., whether or not you'd like to receive periodic emails from Syncopation regarding the latest information on DPL releases, promotions, updates, maintenance fixes and training.

If you set your preference to Yes your contact information will be entered into Syncopation's secure CMS system. Once entered you can expect to receive no more than 3 emails per month from Syncopation. If you set your preference to No thanks, you will not be entered into our CMS system and therefore will not receive regular emails from us.



**Figure 1-6. Register DPL License Webform**

After the Register prompt the DPL application will open. If yours is a new installation, DPL will display the Example Navigator, which is covered in the next section on DPL Help and Resources.



## 1.3 DPL™ Help and Resources

### 1.3.1 Example Navigator

When you launch DPL for the first time, the Examples Navigator window is displayed (Figure 1-7). The Navigator contains a sampling of the example models installed with your license. The Navigator provides the industry vertical, techniques employed, and a brief description of the given DPL example model. Many users find it helpful to browse through already built models to learn new techniques and as a source of ideas.

You can find all of the example models and their linked Excel spreadsheets within the Examples folder within your DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples.

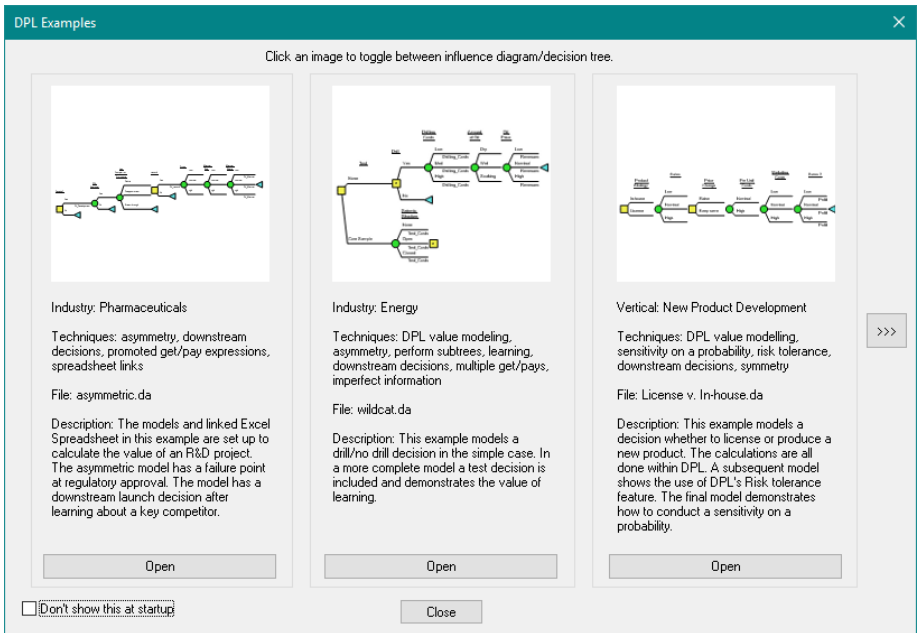


Figure 1-7. DPL™ Examples Navigator

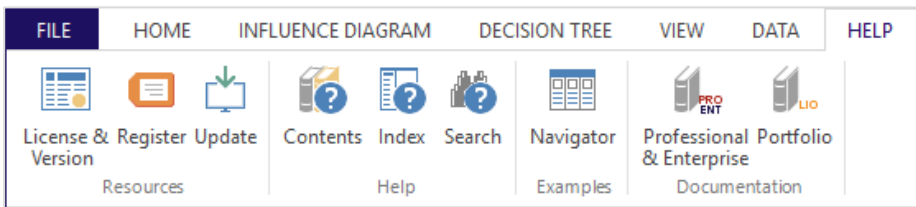
If you prefer not to be shown the Examples Navigator upon start up, check the box labeled *Don't show this as startup* in the bottom left corner of the dialog. You can re-open the Navigator at any time by clicking the Navigator button on the Help tab (Figure 1-8).

### 1.3.2 Online Help

DPL's online help contains detailed descriptions of:

- DPL features and dialogs
- How DPL performs various calculations
- How to interpret each of DPL's outputs
- Common error and warning messages that may be generated by DPL

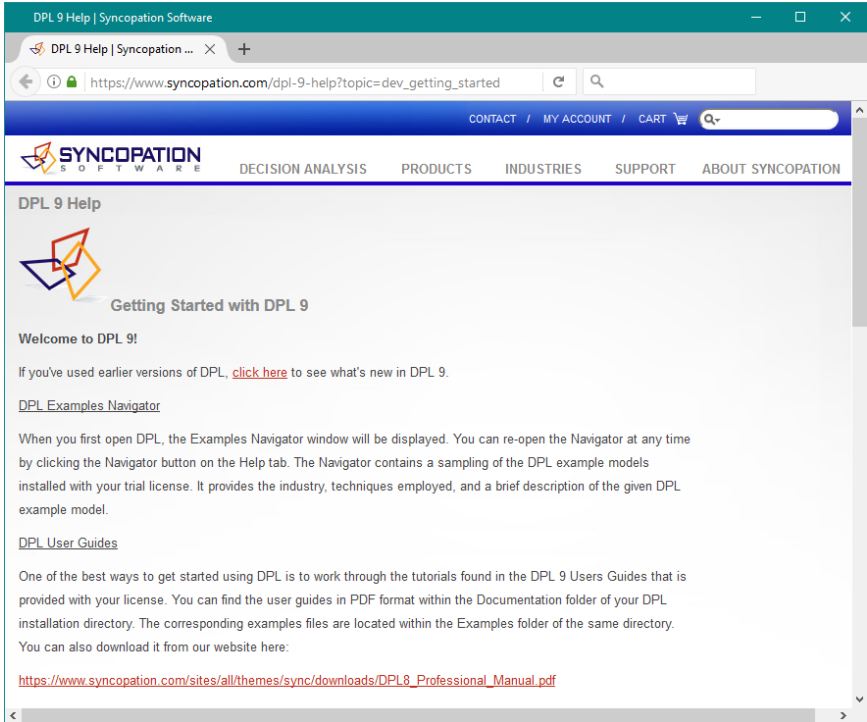
To access online help you can use the commands within the Help group of the Help tab on the ribbon (Figure 1-8) or by pressing F1. Either method will launch a web browser that contains the appropriate DPL help topic.



**Figure 1-8. Help Tab Commands**

The Help | Contents button will launch the *Getting Started with DPL 9* help topic page (Figure 1-9). The Help | Index command will launch the DPL 9 Help Index page, which provides a list of keywords and help topic title links. The Help | Search command will launch the DPL 9 Help topic search page in which you can search for specific terms within the body, keywords, and titles of the help topics.

When you press F1 while modeling in DPL a web browser will open to a contextual online help topic that corresponds to what you're doing or viewing in DPL.



**Figure 1-9. Launched DPL 9 Help Topic in Web Browser (Help | Help | Contents)**

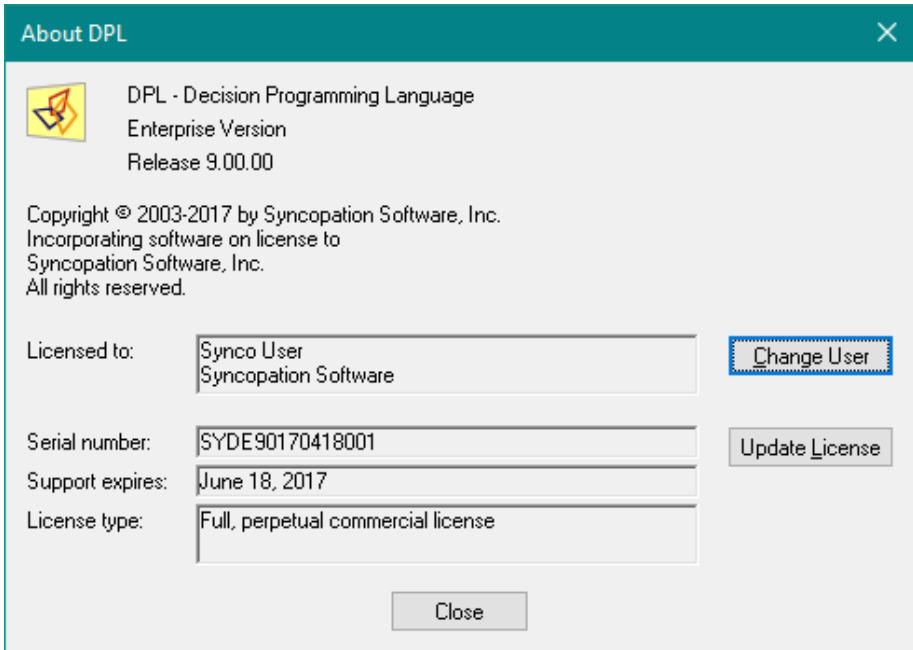
### 1.3.3 Support Team

Perpetual licenses of DPL include 60 days of Maintenance and Support to help you get started and should be renewed annually thereafter. DPL licenses covered by a Maintenance and Support agreements are eligible to receive expert technical and modeling support through phone and email via the following contact points:

**Email: support@syncopation.com**  
**Phone: 1 866 796 2375, Ext. 2.**

Maintenance and Support coverage also includes free software maintenance releases and major upgrades to the product. If you'd like more information about Syncopation's Maintenance and Support coverage, send an inquiry to support@syncopation.com. We pride ourselves on the quality of our support, and we highly recommend you not take any unnecessary risks and keep your DPL Maintenance and Support coverage up-to-date!

When you submit a support request, we ask that you provide us with the information necessary to assist you most efficiently. It is helpful when the support request includes your DPL license version (e.g., Pro), maintenance release (e.g., 9.01.02), license type and serial number – all of which can be obtained via the Session Log (Figure 1-12) or the About DPL dialog shown in Figure 1-10 (Help | License).



**Figure 1-10. About DPL dialog (Help | Resources | License & Version)**

It is also helpful if you include a detailed description of the steps leading up to the issue and the specific behavior you've observed. Screenshots of the application and any error messages received are also helpful. If you can send the DPL model and linked spreadsheets to us, that is usually the best way to bottom out an issue in a timely manner. Any materials sent are treated as strictly confidential.

### 1.3.4 Website Resources

- **FAQs:** Your most frequently asked DPL support and ordering questions answered.  
(<https://www.syncopation.com/faq-page>)
- **Imperfect Information Blog:** From DPL updates, modeling tips and best practices to commentary on the latest topics within decision science.  
(<https://www.syncopation.com/blog>)
- **DPL Industry Examples:** Explore real world DPL examples models by Industry.  
(<https://www.syncopation.com/industry-examples>)
- **DPL Instructional Videos:** Learn about basic decision analytic tools and concepts via short, simple tutorial videos using DPL.  
(<https://www.syncopation.com/instructional-decision-analysis-videos>)

### 1.3.5 DPL™ Training

The quickest, most efficient way to become proficient in DPL modeling is to participate in a DPL training course. Syncopation offers DPL instructor-led, interactive web training sessions that are designed to efficiently transfer essential DPL skills to you without taking you away from your desk. The web-based sessions save on travel costs and are an easy fit into your busy schedule.


For several users we offer customized DPL in-house training courses which also minimize travel costs while also allowing participants to focus on topics specific to their organization. In an in-house setting, we can additionally offer an application workshop focused on a particular issue, model or approach. An application workshop is a great way to "kick-start" a new analytical initiative by giving participants immediate, directly-relevant experience applying DPL in a new setting. For more information on DPL Training see our training overview page:

<https://www.syncopation.com/training>

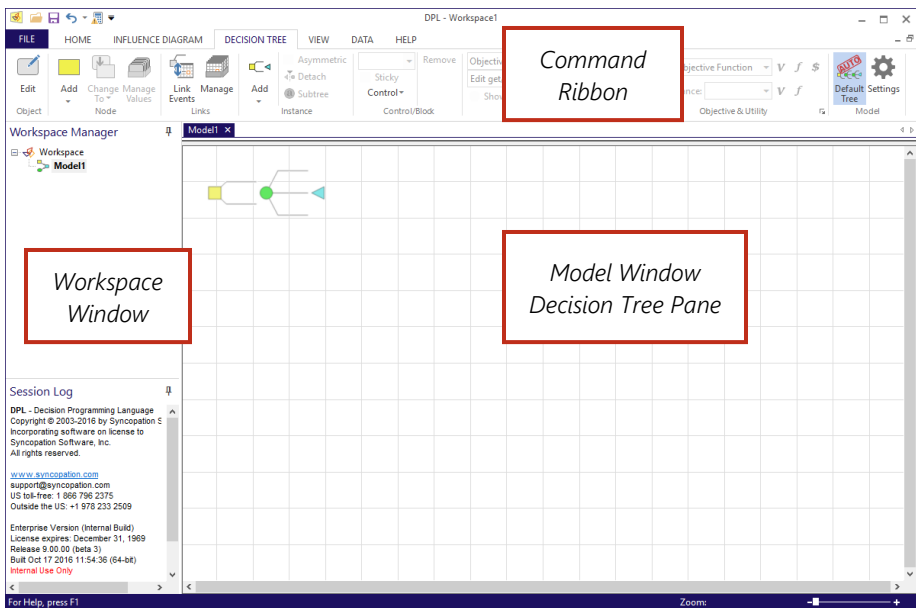
Please send your training inquiries to [sales@syncopation.com](mailto:sales@syncopation.com).

## 1.4 A Brief Tour of DPL™

### 1.4.1 The DPL™ Workspace

⇒ Start DPL by double-clicking the icon on your desktop (  ) or double-clicking the application in Windows Explorer.

DPL will load with an empty, maximized Decision Tree pane within the Model Window on the right-hand side of the screen. On the left-hand side is the Workspace Window. See Figure 1-11.



**Figure 1-11. New Workspace with Empty Model Window in Decision Tree-focused Modeling Mode**

### 1.4.2 The Workspace Window

The Workspace Window contains two different panes: 1) the Workspace Manager pane at the top and 2) the Session Log pane at the bottom. The Workspace Manager pane displays all the documents in the open Workspace in a Windows Explorer-style tree structure. The top level is the Workspace. Below the Workspace are displayed all the models in the Workspace. Below each model are displayed the outputs and other items

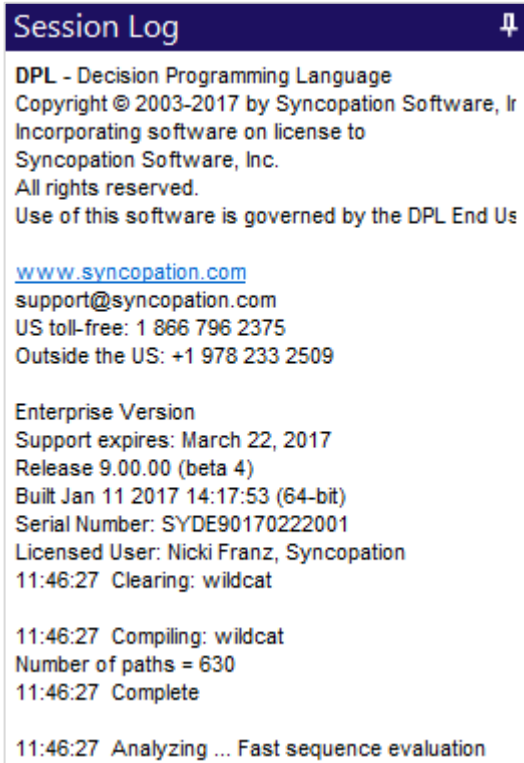
associated with each model. Currently there is only one model (Model1) in the Workspace.

- ⇒ Right-click on the icon for Model1 to see the context menu for a model. Notice among other things that you can delete, rename or duplicate a model.

Within a DPL Workspace, you can develop, run, and store results and outputs from multiple models. The model most recently ran, compiled or viewed is bolded in the Workspace Manager. In general, when you run a Decision Analysis or use any other run or compile command, DPL runs or compiles the active (visible) model. Once you've run or compiled a model, its item in the Workspace Manager will have a compiled indicator, double chevrons ("»»...««"), around the model name.

The Session Log displays information about the current session. Presently, it displays information about the version of DPL you are running and to whom the software is licensed. See Figure 1-12. You can also access DPL license information by clicking Help | Resources | License & Version.

As you build models and run analyses, further information such as compilation messages, expected values, and percentiles are written to the Session Log. Certain error messages are also written in the Session Log. The Session Log is cleared each time you close DPL and each time you open a new Workspace. The Session Log can be edited like any other text document. The contents of it can be copied and pasted to other applications for support assistance or for further analyses.



**Figure 1-12. The Session Log**

Both the Workspace Manager and the Session Log can be undocked by clicking the top of the pane and dragging away from the side of the window or by right-clicking within the desired pane and selecting "Dock". The pane can be left floating or moved to one of the edges of the application window and re-docked there. The window border changes as you drag it to indicate whether and where it will be docked. If you dock one pane on top of the other, the window will show only the "top" pane and create two tabs at the bottom of the Workspace Window, one for each pane view.

The Workspace Manager and Session Log can be hidden using the Auto-hide function (the thumb tack button on the top right of the pane). When the thumb tack is in Pin Down mode, the pane stays visible. Pressing the thumb tack button will toggle to Pin Up mode and the pane will collapse to a tab on the edge of the screen if docked or to a window bar if un-docked. When you mouse-over the tab or window bar, the pane will expand to its visible state. When your mouse leaves the tab/pane area, the pane will



collapse to a tab/window bar again. To keep the pane at its visible state, click the thumb tack to toggle to Pin Down mode.

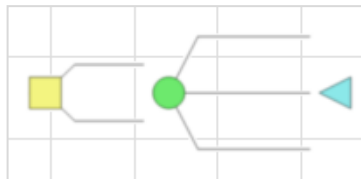
**1.4.3 The Model Window**

The Model Window contains two panes: 1) the Influence Diagram pane and 2) the Decision Tree pane – though both are not necessarily visible at once. Upon launching DPL for the first time, the Decision Tree pane is maximized within the Model Window by default. The Influence Diagram pane is present but minimized and, consequently, hidden from view. This means you are starting in Decision Tree-focused modeling mode.

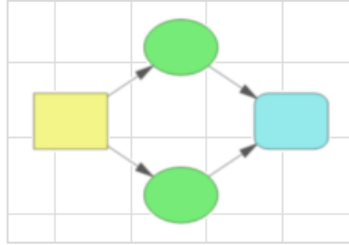
You can change the preferred modeling mode (i.e., the pane maximized) for new models via the "For new models, maximize pane" setting within File | Options | General (Figure 1-16). Upon start up and for newly created models you can choose to have the Decision Tree pane maximized, the Influence Diagram pane maximized, or a halfway split with both panes visible. When the Model Window is in halfway split mode, the Influence Diagram is at the top with the Decision Tree pane on bottom.

When a new, blank workspace is open you'll notice a semi-transparent watermark located in the top left corner of the pane(s), indicating whether it is the Influence Diagram or Decision Tree pane.

If you are in Decision Tree-focused mode the Decision Tree pane is maximized, the Decision Tree tab is active and you will see the following indicator:



If you are in Influence Diagram-focused mode the Influence Diagram pane is maximized, the Influence Diagram tab is active and you will see the following indicator:



If you choose Neither within File | Options | General then the Model Window is split halfway with the two panes divided horizontally by a splitter and both watermarks are visible within their respective panes. You can click and drag the splitter bar between the panes to change the proportion of each. Dragging the splitter bar to the top or bottom edge of the Model Window will make only one of the panes visible, as will double-clicking the splitter bar. Double-clicking the splitter again will set the splitter to the mid-way point between the panes. Lastly, you can quickly switch between maximized panes by using the Tab key.

When you switch between maximized Decision Tree/Influence Diagram panes, the active tab may switch as well. For example, if the Decision Tree pane is maximized and the Decision Tree tab is active and you switch over to the Influence Diagram pane, the Influence Diagram tab will become active, and vice versa.

Lastly, you can change the splitter to split the screen vertically by selecting the radio button View | Layout | Vertical Split (which results in the Influence Diagram being on the left and Decision Tree on the right).

#### 1.4.4 Document Navigator

In addition to the Workspace Manager, all the documents are also displayed as tabs in the Document Navigator (see Figure 1-13) just above the Model Window. You can click the tab for the item once to activate it or press the (X) to delete the item. If the desired tab is not visible in the Document Navigator use the arrows at the far right to scroll through the document tabs.

You can also use Ctrl+Tab to make quick switches between windows in the Workspace.

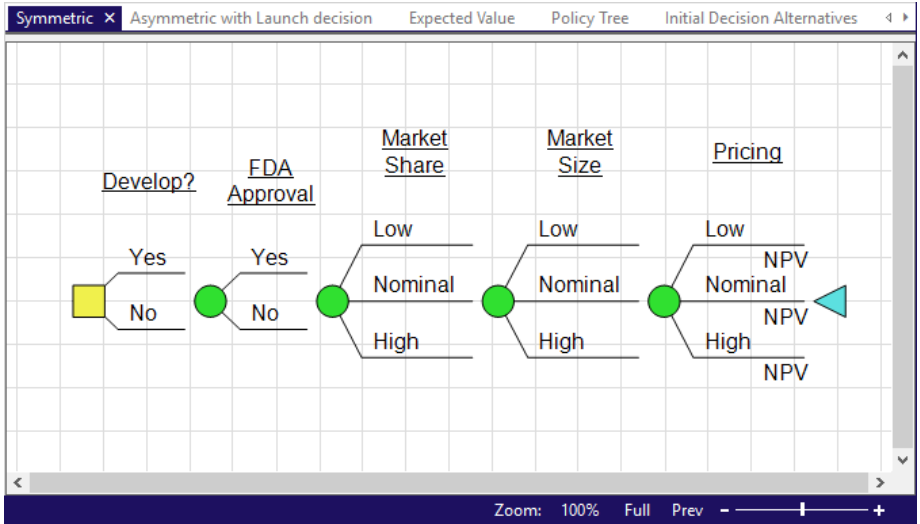
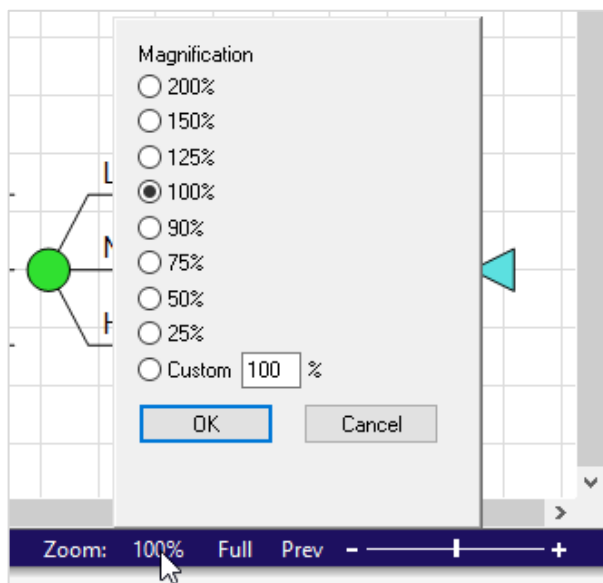






Figure 1-13. Document tabs in the Document Navigator

### 1.4.5 Zoom Controls

You can zoom the Model, Policy Tree and Policy Summary Windows by using the Zoom controls located on the status bar at the bottom right of the application window (see Figure 1-14). For a description of the Zoom controls see Table 1-1.



**Figure 1-14. Zoom Controls on the Status Bar with Zoom dialog active**

<b>Button/Control</b>	<b>Name</b>	<b>Action</b>
	Zoom Percentage	This button launches the Zoom dialog (as seen in Figure 1-14) within which you can select a preset level of zoom or specify a custom zoom level.
	Zoom Full	Zoom the active pane so that all of the Influence Diagram/Decision Tree is visible in the pane.
	Zoom Previous	Return to the previous zoom.
	Zoom Slider Control	Click the -/+ to increase/decrease zoom level or manually click and drag the Zoom slider control to the desired zoom level.

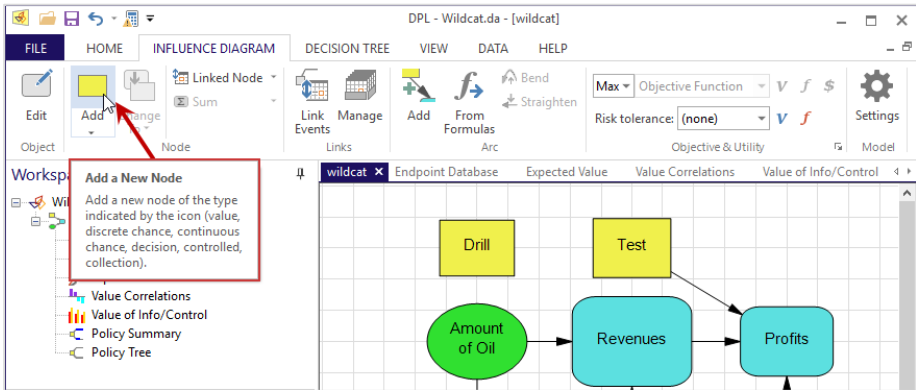
**Table 1-1. Zoom Controls Descriptions**

You can also zoom these windows via the context menu or zoom by selection. To zoom by selection, click the left mouse button on the top left corner of the region you'd like to view, press Ctrl+Shift and then drag the mouse down and to the right. A rectangle indicates what will be displayed when you release the mouse button. Release the mouse button when the rectangle surrounds the portion of the diagram you wish to view.

**1.4.6 Introduction to the DPL Command Ribbon**

Above the Workspace and Model Windows, you will find a Microsoft Office-style command ribbon consisting of multiple tabs. Each tab is split up into contextual groups that contain one or more commands.

You can view a short description of a particular command on the ribbon by holding your cursor over the item until the tooltip note appears.



**Figure 1-15. DPL Ribbon Command Tooltip**

### File Tab


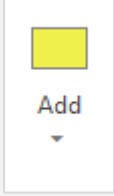
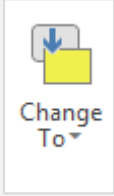

The File tab contains standard commands for creating a new Workspace, opening and saving Workspaces, and printing. You can embed/unembed a DPL workspace in a Excel spreadsheet file. You can also access DPL and Workspace-level options via File | Options. These options will be discussed in more detail later in the chapter.

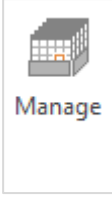
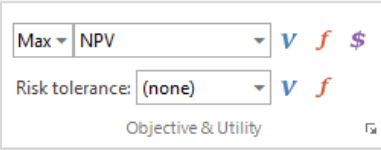

### Home Tab

The Home tab contains a Workspace group within which you can add, delete, rename, copy, cut, paste, select and find/replace items in the Workspace. The Run group and Sensitivity group each display the analyses that can be run on models and the options associated with those analyses.

### Influence Diagram/Decision Tree Tabs

The Influence Diagram (ID) and Decision Tree (DT) tabs (formerly a single Model Tab) contain a number commands for editing DPL models. Some ribbon command groups are available from both tabs, such as the Links group. Others are specific to one or the other. For example, the Get/Pay tab is only available from the Decision Tree tab. The commands, their names and the action associated with each are described in the Tables Table 1-2 through Table 1-5.

Control	Action
 <p style="text-align: center;">Edit</p> <p style="text-align: center;"><b>ID/DT   Object   Edit</b></p>	<p><b>Object Group</b></p> <p><u>Edit (Object) button</u></p> <p>Edit the selected object. This can be used to edit a selected node, branch(es), arc, text box, or get/pay expression. If nothing is selected, the Select Node dialog will open displaying a list of all the nodes in the model.</p>
 <p style="text-align: center;">Add</p> <p style="text-align: center;"><b>ID/DT   Node   Add</b></p>	<p><b>Node Group</b></p> <p><u>Add (Node) split button*</u></p> <p>Add a new, unlinked node to the Influence Diagram and/or Decision Tree of the default type indicated by the icon.</p> <p>Note: Value nodes do not appear independently in the Decision Tree but can be added to get/pay expression on an event's branch(es).</p>
 <p style="text-align: center;">Change To</p> <p style="text-align: center;"><b>ID/DT   Node   Change To</b></p>	<p><u>Change to (Node type) split button</u></p> <p>Change type of selected node to the default type indicated by icon.</p> <p>You can select multiple value nodes at once (Ctrl+Click) in order to change them to chance nodes with a single click.</p>
 <p style="text-align: center;">Link Events</p> <p style="text-align: center;"><b>ID/DT   Links   Link Events</b></p>	<p><b>Links Group</b></p> <p><u>Link Events button</u></p> <p>Launches the Link Model Events dialog which serves as an interface for linking the model's decision, chance, and controlled nodes to spreadsheet driver cells.</p>

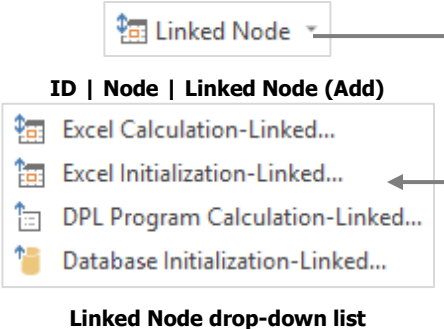
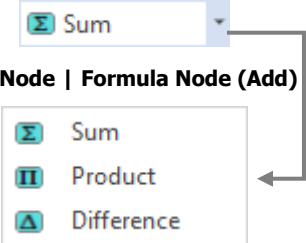
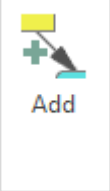
 <p style="text-align: center;"><b>Manage</b></p> <p style="text-align: center;"><b>ID/DT   Links   Manage</b></p>	<p><u>Manage Links button</u></p> <p>Launches the Manage Links dialog which provides an overview of all sources to which the model is linked and also a summary of node links by source. Link sources can be edited/updated from the dialog. Spreadsheet can be converted to a DPL Program from the dialog as well.</p>
 <p style="text-align: center;"><b>ID/DT   Objective &amp; Utility</b></p>	<p><b>Objective &amp; Utility Group</b></p> <p><u>Objective &amp; Utility combo box</u></p> <p><i>Only enabled for multiple attribute models.</i> Use the drop-down list to select a single attribute to use for the objective function or type in an expression involving more than one attribute and/or value from the model. Use Max/Min drop-down list to either maximize or minimize the objective function. Use the ( <b>V</b> ), ( <b>f</b> ) or ( <b>\$</b> ) buttons to insert a variable, function, or attribute.</p> <p><i>For single attribute models this is set to Objective Function, and grayed out.</i></p> <p><u>Built-In Risk Tolerance combo box</u></p> <p>Specify the coefficient for the built-in risk tolerance function in order to model attitudes towards risk.</p> <p>Use the ( <b>V</b> ) or ( <b>f</b> ) buttons to insert a variable or function.</p>
 <p style="text-align: center;"><b>Settings</b></p> <p style="text-align: center;"><b>ID/DT   Model   Settings</b></p>	<p><b>Model Group</b></p> <p><u>Model Settings button</u></p> <p>Launches the Model Settings dialog within which you can change a variety of settings for the active model. The Influence Diagram tab command will launch the dialog with the Links tab active. The Decision Tree tab command will launch the dialog with the Decision Tree tab active.</p>


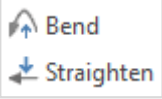
**Table 1-2. Shared Ribbon Commands for Decision Tree and Influence Diagram Tabs**

\* *Note: DPL has several split buttons on the command ribbon. The icon displayed within the top half of the split button indicates the current default command. Click the top half of the button to choose the default command. If you desire a command other than the default, drop-down the split button to choose the command from the list.*

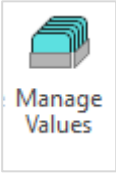
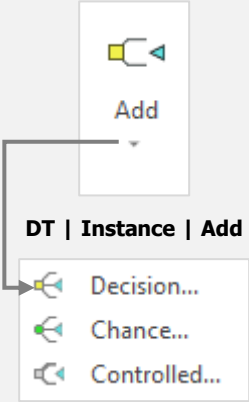
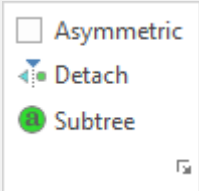


If you select a command from the split button's drop-down list, the default command and resulting icon changes to that. The split button allows you to execute whichever item you last executed again without having to drop down the list.

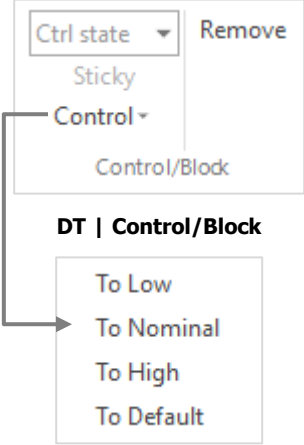
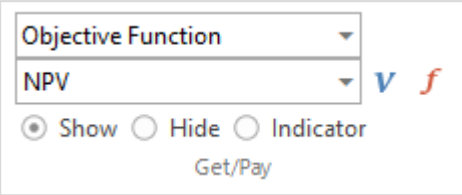
Control	Action
 <p><b>ID   Node   Linked Node (Add)</b></p> <p><b>Linked Node drop-down list</b></p>	<p><b>Node Group (ID only)</b></p> <p><u>Linked Node (Add) split button</u></p> <p>Add a linked value node of the type indicated by the icon to the Influence Diagram.</p> <p>Nodes can be linked to an Excel spreadsheet (Calculation or Initialization-linked), a DPL Program, or a database.</p>
 <p><b>ID   Node   Formula Node (Add)</b></p> <p><b>Formula Node drop-down list</b></p>	<p><u>Formula Node (Add) split button</u></p> <p>Add a summation, product, or difference node to the model, as calculated from the value nodes selected.</p>
 <p><b>ID   Arc   Add</b></p>	<p><b>Arc Group</b></p> <p><u>Add (New Arc) button</u></p> <p>Add new influence arc between nodes to indicate timing, formulaic dependence, or conditioning.</p>


 <p><b>ID   Arc   From Formulas</b></p>	<p><u>From Formulas button</u></p> <p>Add new influence arc(s) to Influence Diagram that are based on the formulas of the data definitions of each node.</p>
 <p><b>ID   Arc   Bend/Straighten</b></p>	<p><u>Bend/Straighten buttons</u></p> <p>Bend or straighten the selected influence arc(s).</p>

**Table 1-3. Influence Diagram Tab-specific Ribbon Commands**

Control	Action
 <p><b>DT   Node   Manage Values</b></p>	<p><b>Node Group (DT only)</b></p> <p><u>Manage Values</u></p> <p>Launches the Select Value dialog which lists all the value nodes in the model. Values in the list can be deleted or edited (i.e., the Node Definition dialog for the selected value node will open).</p>
 <p><b>DT   Instance   Add</b></p> <p><b>Instance drop-down list</b></p>	<p><b>Instance Group</b></p> <p><u>(Add) Instance split button</u></p> <p>Add a new node instance to the Decision Tree of the type indicated by the icon.</p> <p>Choices in the drop-down list include adding a decision instance, adding a chance instance, or adding a controlled instance.</p> <p>Within the Select (Node type) dialog, you can choose to add an instance of an existing node or create a new node.</p>
 <p><b>Right side of DT   Instance</b></p>	<p><u>Asymmetric check box</u></p> <p>Check this box to modify the outcome grouping of selected node(s) or branch(es) to be asymmetric. Uncheck the checkbox to make the node symmetric.</p> <p><u>Detach button</u></p> <p>Detach selected node and create a new detached subtree starting with the selected node.</p> <p><u>Perform Subtree button</u></p> <p>Performs a subtree that begins with the node selected and is performed at the connection point (blue triangle) selected.</p>

**Table 1-4. Decision Tree Tab-specific Ribbon Commands**

Control	Actions
 <p><b>Control Selection drop-down list</b></p>	<p><b>Control/Block Group</b></p> <p><u>Control combo box</u> <i>(for a single node)</i></p> <p>Control the selected node to the branch selected in the combo. Control is removed if (none) is selected.</p> <p><u>Sticky checkbox</u></p> <p>Check box to make the control setting sticky for selected node(s). Once designated sticky, control settings for these node(s) must be changed individually.</p> <p><u>Control Selection drop-down</u> <i>(for multiple nodes at once)</i></p> <p>Control branches of the selected nodes to Low, Nominal, High, or Default (as defined in the node's definition) state.</p> <p><u>Remove</u></p> <p>Remove control from selected node(s).</p>
 <p><b>DT   Get/Pay</b></p>	<p><b>Get/Pay Group</b></p> <p><u>Select Attributes drop-down</u></p> <p>For multiple attribute models, choose the attribute from the drop-down list for which you'd like to edit the get/pay expression for the selected branch(es).</p> <p>For single attribute models this box is set to Objective Function.</p> <p><u>Edit Get/Pay combo box</u></p> <p>Use the drop-down list to select a single quantity or type in an expression for the get/pay for the selected branch(es). For multiple attribute models, the get/pay is for the attribute selected in the Attributes drop-down list otherwise it is for the objective function. Use the ( <b>V</b> ) or ( <b>f</b> ) buttons to insert a variable or function.</p>

	<p><u>Get/Pay Display radio buttons</u></p> <p>Display, hide or show an indicator (\$) for the get/pay expression for the selected branch(es).</p>
 <p><b>DT   Model   Default Tree</b></p>	<p><b>Model Group (DT only)</b></p> <p><u>Re-build Default Tree</u></p> <p>Build or re-build the default tree. Your existing tree is deleted and replaced with a symmetric, default tree.</p>

**Table 1-5. Decision Tree Tab-specific Ribbon Commands Continued**

**View Tab**

You can access several useful display commands within the View tab. Once you have multiple items within your workspace, you can arrange windows and icons and switch between windows via commands within the View | Window group.

Within the View | Layout group there are commands to change the model window split from horizontal to vertical, snap nodes to the grid and align or arrange nodes. You can also add a block of text, typically used for titles or explanatory notes, within the Influence Diagram or Decision Tree. Note that text blocks can be edited independent of the text associated with nodes (node names and branch names).

You can toggle back and forth between the Certain Equivalents and Expected Value view for many outputs by clicking View | Results | CE. The Certain Equivalents view is available only if a risk tolerance has been specified.

You can change the font, font style, and font size for various items within the Influence Diagram, Decision Tree and output charts using the controls in the View | Font group.

Lastly, you can choose to show or hide the tip boxes that are displayed when the mouse is hovered over a node or graphical object. The Show Tips button is shaded blue upon launching DPL, indicating Show Tips is on by default.

**Data Tab**

Within the data tab you can export the data or statistics of a particular output via the Data | Export icon. Within Data | Distributions you can view

the Statistics for a distribution or reduce the distribution to a single chance node. Data | Update includes commands for updating reports or charts using the latest data. DPL Enterprise users can set up an ODBC Datasource for a Workspace and then load the database schema information within the Data | Database group. And lastly, the Data | Learning group offers commands for estimating and updating probabilities from external outcome data in a .CSV file.

### **Help Tab**

You can view or update your DPL version, license and license information or register your DPL license via the commands within the Help | Resources group. You can launch the DPL Online Help within a web browser by clicking one of the three icons within the Help | Help group. You can launch the Examples Navigator by clicking the Help | Examples | Navigator button. And lastly, you can open the DPL 9 User Guide in PDF format by clicking the Help | Documentation | Professional & Enterprise button.

### **Contextual Tabs and Commands**

Some ribbon tabs are contextual and will appear only when a particular item is active. For example, the Policy tab appears only when a Policy Tree or Policy Summary is active. As you work your way through the following tutorials, in addition to the Policy Tab, you will find the following contextual tabs: the Grid Tab (when an Endpoint Database is active) and the Chart | Series and Chart | Format tabs (when a chart, such as a Risk Profile Chart or Tornado Diagram, is active).

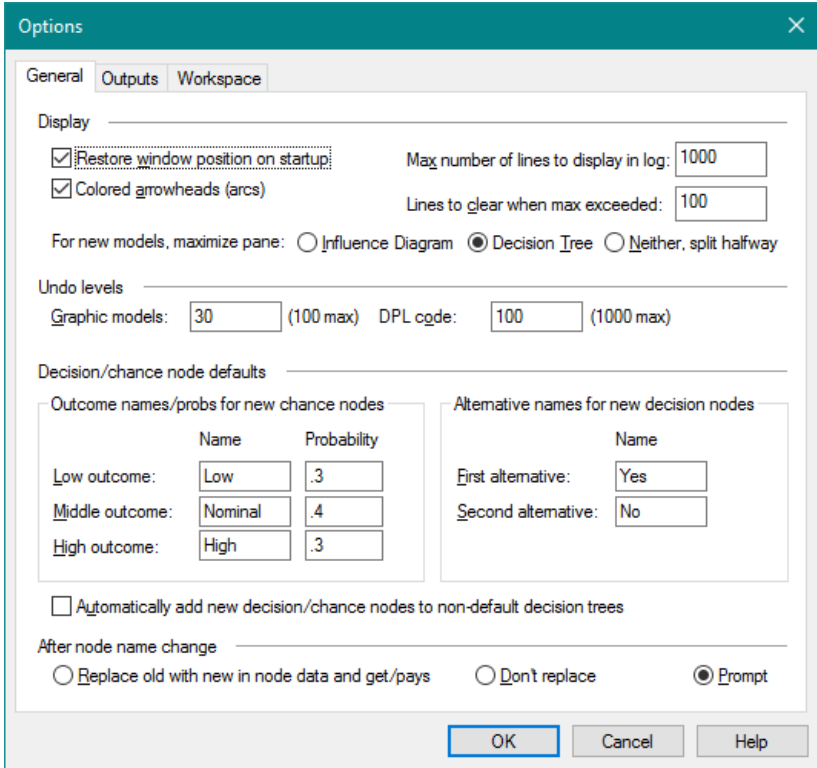
Furthermore, not all commands within a particular tab will always be enabled since some of them are context specific. For example, the options within the Decision Tree | Get/Pay group are only enabled when you have branch(es) selected in the Decision Tree.

## **1.4.7 DPL™ and Workspace-Level and Model Options**

You will now check that workspace options and model settings are set in a way so that as you work with DPL, what you see on your screen will be similar to the images in this guide. In DPL 9, there are a few places to set options. You can access DPL and Workspace-level options via File | Options. Spreadsheet linking and settings that are specific to the models within a Workspace are set within the Modeling Settings dialog (Influence Diagram/Decision Tree | Model | Settings). Lastly, there are a few settings that are specific to models within a Workspace that are set via dialog box launchers. For example, Home | Run | Settings will open the Run Settings dialog where settings relating to a decision analysis run are set. The various model options are discussed throughout the manual.

⇒ Select File | Options to access the DPL and Workspace-level options.

The Options dialog appears with the General tab active as shown in Figure 1-16.



**Figure 1-16. General Tab of File Options dialog**

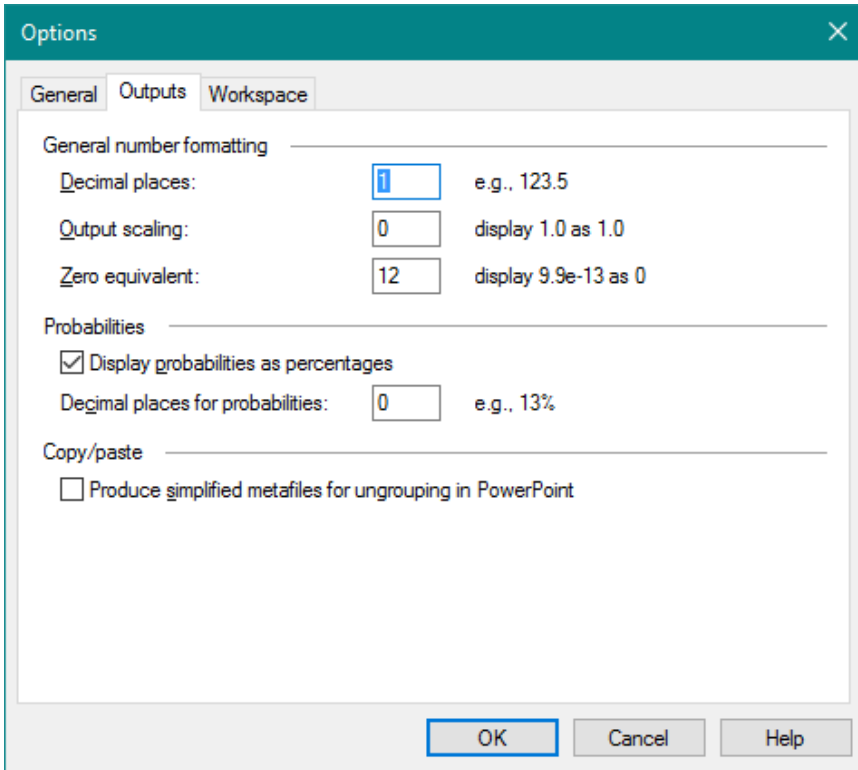
⇒ Select the Outputs tab.

⇒ Make sure that Decimal places is set to 1.

⇒ Make sure that Display probabilities as percentages is checked.

⇒ Make sure that Decimal places for probabilities is set to 0. The Outputs tab should look like Figure 1-17.

⇒ Click OK to close the File Options dialog.



**Figure 1-17. Outputs Tab of Options Dialog**

All of the options on the General tab are at the installation level, i.e., they are not Workspace specific. The options on the Outputs and Workspace tabs are Workspace-level options and can vary from Workspace to Workspace.



## 1.5 What's New in DPL™ 9

---

This section is intended as a reference for users of previous versions of DPL. A list of some of the major changes and new features in DPL 9 by version are provided below.

### **DPL 9 Professional**

#### **More flexible modeling, Decision Tree or Influence Diagram-focused**

Select which modeling mode you prefer: Decision Tree-focused, Influence Diagram-focused, or no preference (i.e., split halfway between) and build your model entirely in the view you prefer.

#### **Modeling Ease-of-use Improvements**

Multi-select any object in the Influence Diagram or Decision Tree to make multiple changes at once. Manipulate the Decision Tree via a streamlined and more intuitive use of the mouse.

#### **Graphical Output Improvements**

Exercise full control over the marker size in all charts and the percentiles and labels displayed in Risk Profiles.

#### **New Time Series Percentile Chart for Initial Decision Alternatives**

*(Covered in Section 10.3)*

Visualize cash flow over time for all your initial decision alternative to gain a complete picture of how uncertainty evolves over time.

#### **Selective Attributes** *(Covered in Section 10.2.1)*

Got a lot of attributes? With this feature you can slim down your Policy Tree by selecting exactly which attributes are displayed.

#### **Perform Subtree and Continue** *(Covered in Section 7.2)*

Just like it says – perform a subtree and then continue descending the tree with further events so you can define your decision problem with precision.

### **DPL 9 Enterprise**

#### **Multiple-objective Functions** *(Covered in Section 8.3)*

Who's making that decision? Optimize multiple objective functions simultaneously within a single model to get a handle on the actions of partners or adversaries.

#### **Parallel Endpoint Recording** *(Covered in Section 11.6)*

Divide and conquer by running a large model on several different machines.

**Allow Event Already Active** *(Covered in Chapter 15)*

Need to model a decision where you need to encounter the same event more than once on a single path in the tree? Model it with ease by enabling the new "Allow event already active and halt with penalty value" feature.

**Enhanced API and Improved Database Performance**

Expanded functionality for embedding DPL in an automated decision support system and better database connectivity under the covers.

**DPL Takes you from Data to Decisions** *(Covered in Chapter 20)*

Estimate probabilities and discover conditional relationships so you can swiftly leverage your data to make better decisions.

**All Versions****64-bit Executable**

Exploit the scalability of the 64-bit Windows architecture.

**.XLSX Conversion**

DPL now does Excel XML: convert an .xlsx/xlsm spreadsheet to a DPL Program for a huge performance boost.

## 2. Building a Model in Decision Tree-focused Mode

The next few chapters of this guide will focus on building simple spreadsheet-linked DPL models from scratch for real-world decision-problems. In this chapter a spreadsheet-linked DPL model is built in Decision Tree-focused mode. In DPL, Influence Diagrams and Decision Trees work together to provide a complete definition of the decision problem – but at times, certain decision-problems lend themselves to being modeled primarily within one medium or the other.

A Decision Tree provides an intuitive, map-like representation of the sequence and structure of a decision problem. Throughout this tutorial you'll be introduced to many key Decision Tree building features, such as, creating local Decision Tree events, modifying the Decision Tree structure, adding conditioning, linking tree events and the output metric to Excel and editing get/pay expressions. If you are new to DPL and would like to familiarize yourself with the skills needed to build Decision Tree models, this is a great place to get started. To round out your DPL modeling skills, Chapter 4 contains a tutorial in which you build a model in Influence Diagram-focused mode from an Excel spreadsheet.

In this tutorial you will be developing a Decision Tree model for a drug development decision. The final model will include technical, regulatory, and commercial uncertainties along with a downstream decision, or exit option. At the start of Chapter 3 you will run an analysis on the completed model and examine the resulting DPL outputs.

### 2.1 Decision Tree Overview

---

A DPL Decision Tree is made up of decision and chance nodes and one or more output metrics, which can also be referred to as objectives. A decision node has one or more alternatives. During a decision analysis, one alternative is determined to be optimal. Decision nodes appear as yellow squares with its number of alternatives as branches in the Decision Tree.

A discrete chance node has a discrete set of outcomes that are uncertain. Discrete chance nodes appear as green circles with its number of outcomes as branches in the Decision Tree. Since both decision nodes and chance

nodes represent things that happen at a particular point in time, they are referred to as events.

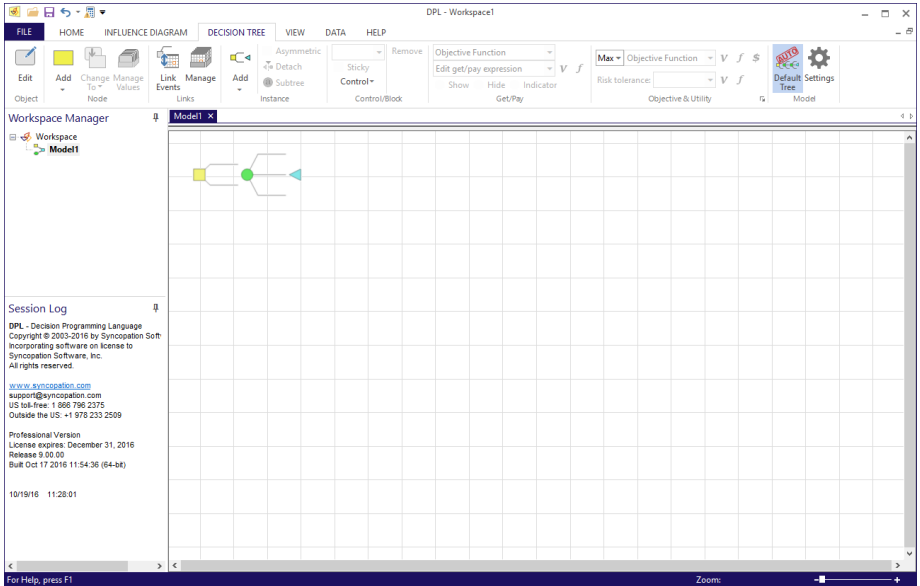
The output metric is the calculated end result of the model/spreadsheet to be maximized (or minimized, if appropriate).

You will be developing a Decision Tree model for a drug development decision by adding several local, non-spreadsheet linked events to the Decision Tree. The events and the output metric for the model will then be linked to the Drug Development.xlsx Excel spreadsheet, which is an example file that has been installed with the DPL software.

The resulting DPL model and its associated spreadsheet are set up to calculate the value of a pharmaceutical R&D project. The project is in an early stage, so there is lot of uncertainty about whether it will succeed through the various technical hurdles. If it does succeed, its revenues are influenced by the following market uncertainties: Key Competitor Outcome, Pricing, Market Size and Market Share.

⇒ Open DPL if it isn't already.

DPL will load with an empty Model Window on the right-hand side of the screen as shown in Figure 2-1. The pane(s) displayed within the Model Window will depend on your current modeling mode setting: Decision Tree-focused, Influence Diagram-focused, or a halfway split. If you've never changed the setting, the DPL Workspace will open in Decision Tree-focused modeling mode by default, as shown in Figure 2-1.



**Figure 2-1. Blank DPL Workspace with Decision Tree Pane Maximized**

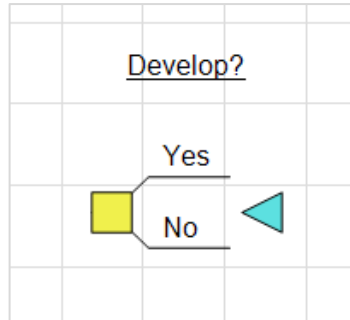
Before you start building the model, you will change some of the default decision and chance node settings for the model in order to better fit the decision structure.

## 2.2 Symmetric vs. Asymmetric Trees

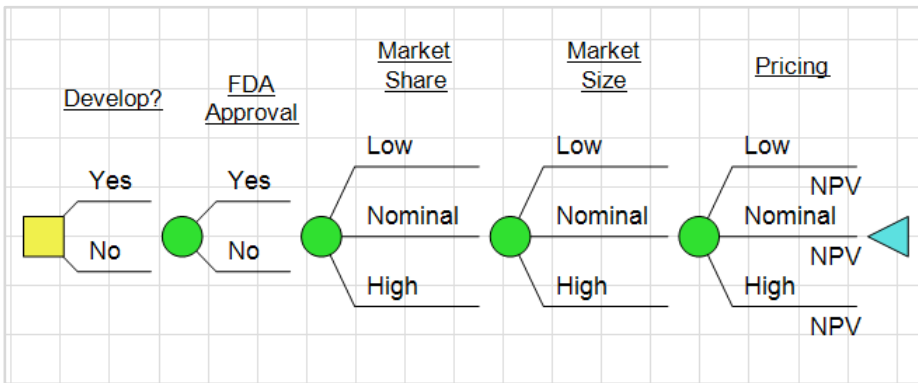
Prior to modifying node settings you'll first need to understand the difference between a "symmetric" and an "asymmetric" node in DPL. A symmetric node is defined as a node whose branches (e.g., outcomes or alternatives) all lead to the same next event. Once placed, you'll see that a symmetric node will have only blue endpoint node or connection point (the blue triangle) for all of its branches. A symmetric Decision Tree is one that contains only symmetric nodes.

With the current default model settings new events added to the Decision Tree will have a symmetric outcome grouping. Furthermore, if you were to start building your model within the Influence Diagram DPL will automatically build a symmetric, default Decision Tree for you. Default trees will be discussed in detail within Section 4.4.

See Figure 2-3 and Figure 2-6 for an example of a single symmetric decision node and a symmetric Decision Tree.

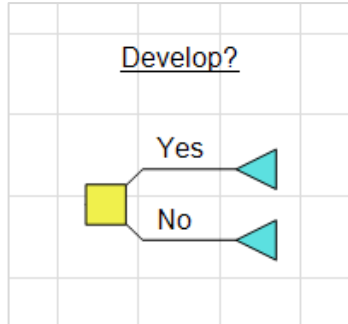


**Figure 2-2. A Symmetric Decision Node**

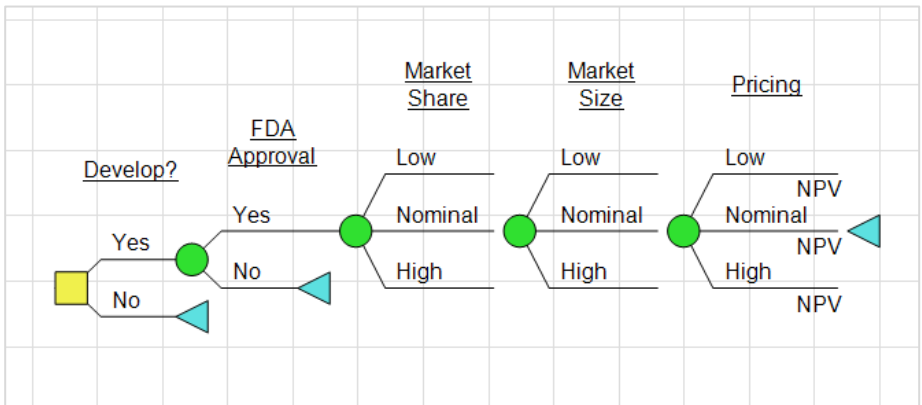


**Figure 2-3. A Symmetric Decision Tree with 5 Symmetric Nodes**

In contrast, an asymmetric Decision Tree contains one or more asymmetric nodes. With asymmetric nodes, each branch may lead to a different next event. When you add an asymmetric node to the Decision Tree, you'll find that each branch has its own endpoint node or connection point. See Figure 2-4 and Figure 2-5 for an example of a single asymmetric decision node and an asymmetric Decision Tree.



**Figure 2-4. An Asymmetric Decision Node**



**Figure 2-5. An Asymmetric Decision Tree with 2 Asymmetric Nodes**

The concept of asymmetry in Decision Tree modeling is important because many business decisions are asymmetric in nature: different strategic alternatives are often influenced by different sources of uncertainty. Modeling that asymmetry directly in the tree results in a model which reveals the true structure of the problem, can improve computational efficiency, and is useful for communication purposes.

The drug development model you're about to build includes several uncertain, 2-outcome failure points, which are modeled with asymmetric nodes. In order to build the model most efficiently, you will change the applicable default model and workspace-level settings for newly added Decision Tree events to match this structure.

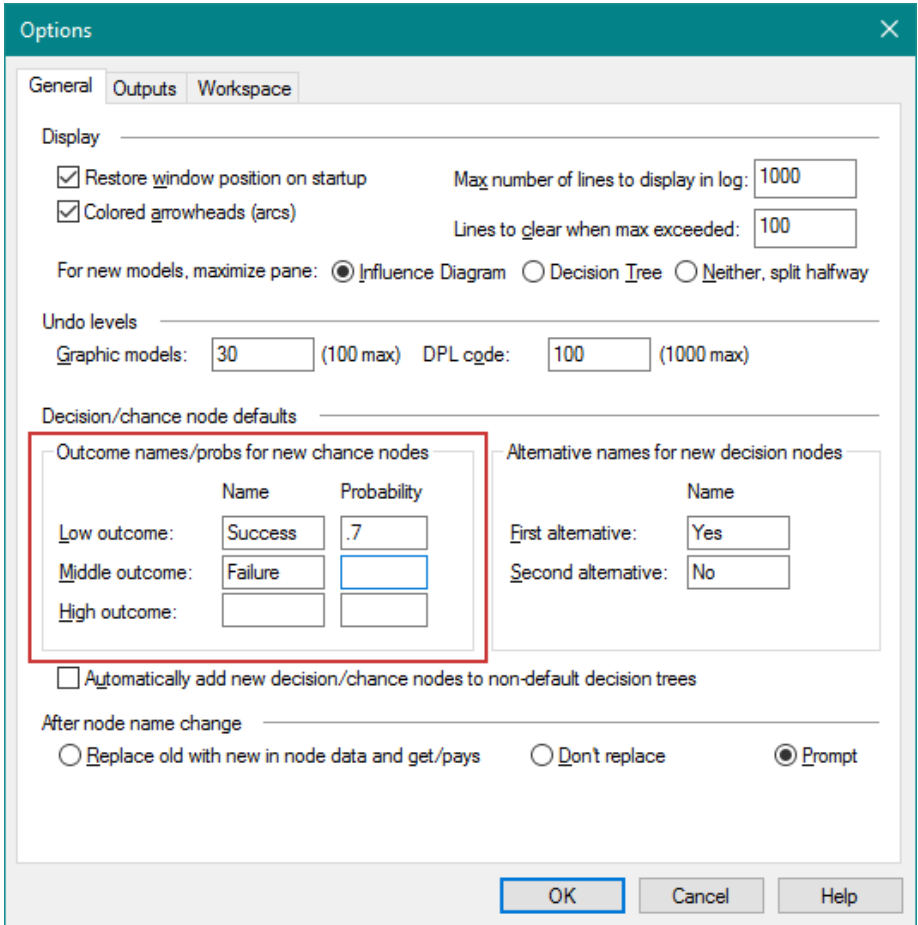
- ⇒ Click File | Options, which will launch the File Options dialog with the General tab active.

Notice the default settings for *Outcome names/probs for new chance nodes* under the *Decision/chance node defaults* section. The outcome names for new chance nodes are set to be Low, Nominal and High with probabilities of .3, .4, .3 assigned, respectively. You will change these defaults to better match the structure of the events that define the drug development case. This will result in a more efficient and compact model build out.

- ⇒ Delete the name and probability assigned to the High outcome.
- ⇒ Select the name for the Low outcome and rename it to be "Success".
- ⇒ Change the probability for the Low outcome to be ".7".
- ⇒ Edit the name of the Nominal outcome to be "Failure" and delete its probability.

To the right, beneath the *Alternative names for new decision nodes* heading, you can see that the default alternatives for new decisions added to the tree will be set to "Yes" and "No". You'll leave the defaults for decision events as they are. The outcome name and probabilities for new chance nodes should now look like Figure 2-2.





**Figure 2-6. Updated Outcome Name and Probabilities for New Chance Nodes**

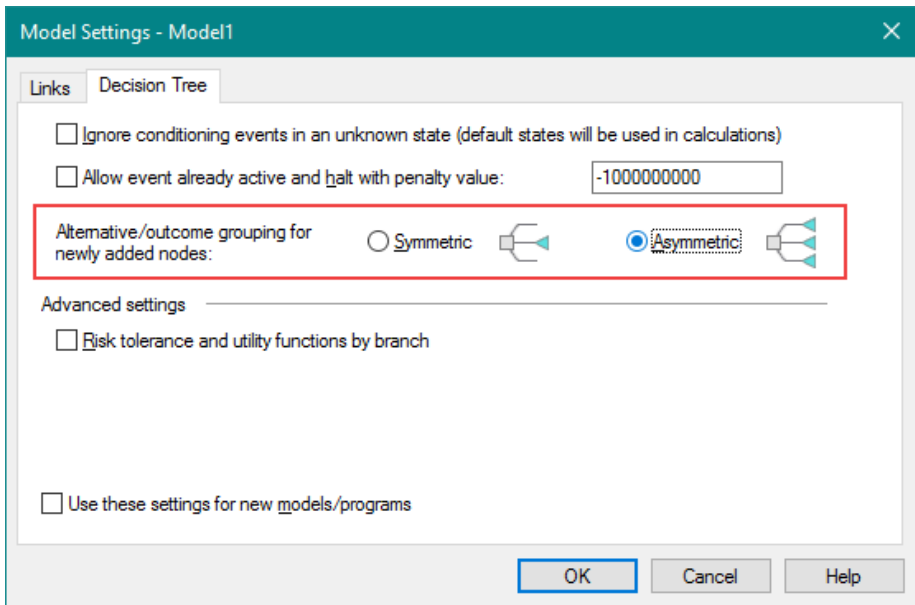
⇒ Click OK to close the File Options dialog.

Now you will change the default outcome grouping for newly added chance events. More specifically, you are going to change the default outcome grouping setting so that new events added to the tree are asymmetric.

⇒ Click Decision Tree | Model | Settings. The Decision Tree tab of the Model Settings dialog is displayed.

⇒ Set the *Alternative/outcome grouping for newly added nodes* radio button to Asymmetric.

The Decision Tree tab should now look like Figure 2-7.



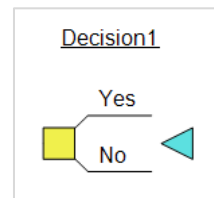
**Figure 2-7. Updated Alternative/Outcome Grouping Settings**

⇒ Click OK to close the Model Settings dialog.

## 2.3 Adding a Decision Node

You will start to develop the model by adding several local events to the Decision Tree. The first event added is the upfront or initial decision as to whether to develop the drug or not.

A decision node has two or more alternatives. DPL will choose the decision alternative that maximizes the output of the model (referred to as the objective function). A decision node is represented in the Decision Tree by a yellow square. Each branch coming from the node represents a decision alternative. The image to the right displays a symmetric decision node named Decision1 with two alternatives, Yes and No.



*Symmetric  
Decision Node*

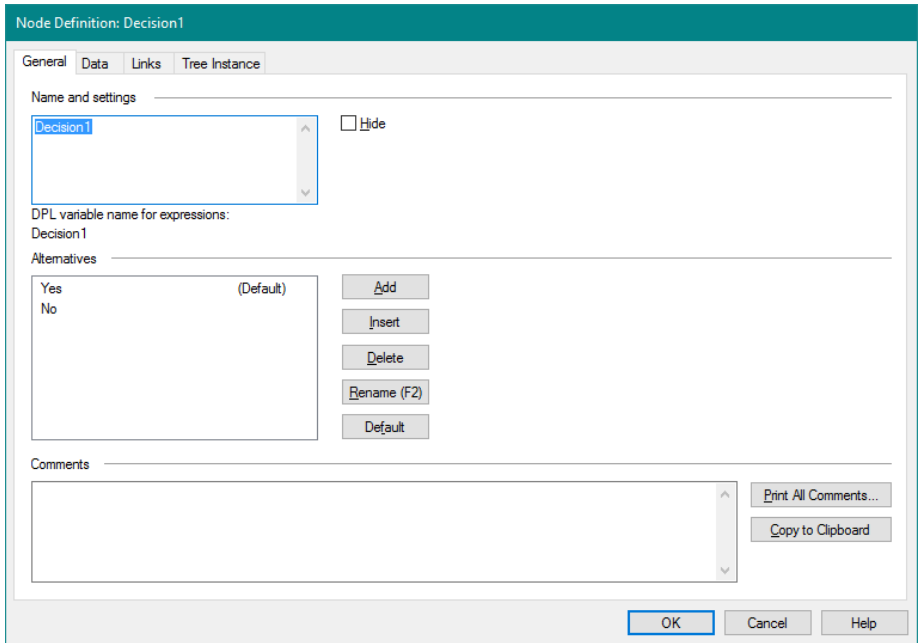
⇒ Click top half of the Decision Tree | Node | Add split button to add a new decision node to the tree.

Decision node should be the default command indicated by the icon. If it isn't, drop down the split button list and select it.

Once you've clicked Add, your cursor will move down to the Decision Tree pane and have a semi-transparent decision node beneath it.

⇒ Click somewhere in the left side of the Decision Tree pane to place the decision node.

Once placed the Node Definition is opened with the General tab active as shown in Figure 2-8. This is where the node name and its states are defined.



**Figure 2-8. General tab of Node Definition dialog for Decision Node**

Note: There is an Alternatives list box on the General tab of the Node Definition dialog for decision nodes. You can add, delete, insert or rename decision alternatives. As you saw previously, the default alternative names can be changed in File | Options | General. You will the alternative names as is.

Notice that the first decision alternative ("Yes") is designated as the "default" alternative. You don't need to worry about this right now. Default alternatives will be discussed in more detail in Sections 2.10 and 5.4.

⇒ Type "Develop?" for the node name.

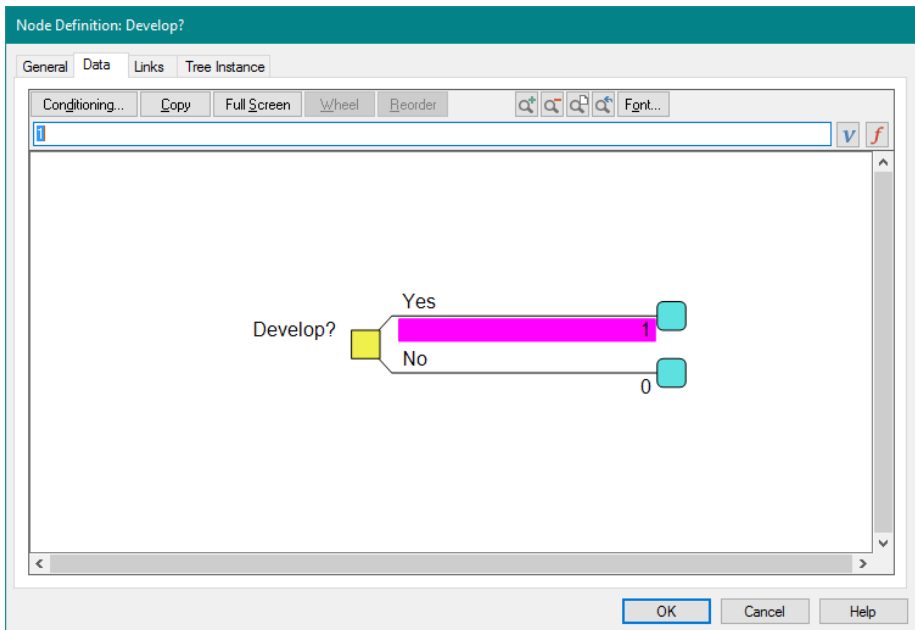
- ⇒ Leave the alternatives as the defaults, Yes and No.
- ⇒ Switch to the Data tab.

The Data tab displays a graphical data input tree that allows you to enter a value expression for each decision alternative. The first alternative is shaded magenta, DPL's selection color, which means the value edit box for the "Yes" alternative is selected for editing. You can use the down arrow key or the Enter key to select the next alternative.

There are several buttons on the Data tab which allow you to edit a node's conditioning; zoom the input data tree; maximize the dialog to full screen; and specify variables and functions via a dialog to help input the data. These are discussed in more detail later.

- ⇒ Enter a value of "1" within the value text edit box for the Yes alternative and press the down arrow key to move to the No alternative.
- ⇒ Enter a value of "0" for the No alternative and press Enter.

Your data input tree should now match Figure 2-9.



**Figure 2-9. Data Input Tree for Develop? Decision Node**

Note: All of the events included in this Decision Tree model will have "flag" values associated with their outcomes/alternatives (e.g., 1 and 0), as opposed to the event's actual numeric values (e.g., \$10,000,000, 50,000 units, etc.). These flags correspond to logic in the spreadsheet. All Yes/No and Success/Fail events are assigned values of 1 and 0, respectively. Chance events with Low/Nominal/High outcomes are assigned flag values of 1, 2, and 3, respectively.

⇒ Click OK to close the Node Definition dialog.

Notice that the newly added decision node is asymmetric with each decision alternative having its own blue endpoint node. This is due to the fact that the default outcome grouping settings have been changed. In the Decision Tree, blue endpoint nodes serve as the connection point when adding events to the tree.

⇒ Press ESC to deselect Develop?. Your model should look like Figure 2-10.

Note in a number of the images that follow nothing is selected. The images will differ slightly from what you see in DPL after e.g., editing a node. By default, items remain selected in DPL after you edit them. The images show the model with no selection.

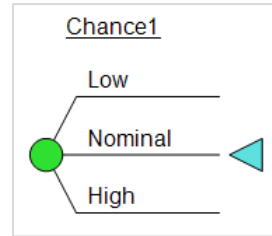


**Figure 2-10. Asymmetric Develop? Decision Node**

## 2.4 Adding a Chance Node

Now you'll add the first uncertain event to the tree: a discrete chance node. In DPL, a discrete chance node has two or more outcomes which occur with specified probabilities. During an analysis, DPL will evaluate each chance outcome to calculate the expected value of the decision model when rolling back the Decision Tree.

A discrete chance node is represented in the Decision Tree by a bright green circle. Each branch coming from the node represents a chance outcome. The image to the right displays a symmetric, discrete chance node named *Chance1* that has three outcomes: Low, Nominal, and High.



*Symmetric Discrete  
Chance Node*

- ⇒ Create a new discrete chance node by dropping down the Decision Tree | Node | Add split button and selecting Discrete Chance from the list.
- ⇒ Place the node on the endpoint (blue triangle) for the Yes alternative of the Develop? decision. The General tab of the Node Definition dialog appears.

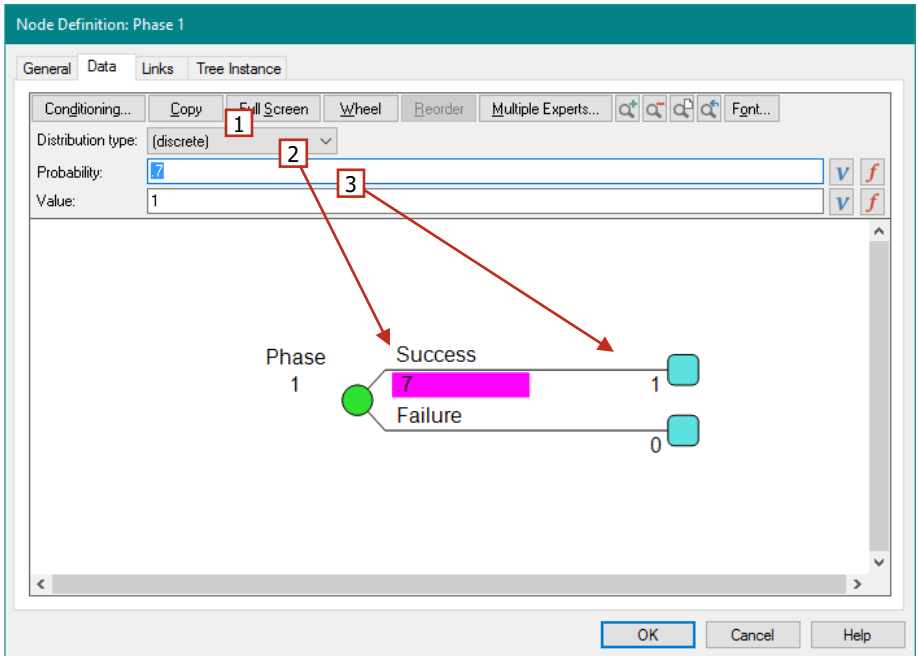
Similar to the decision node in Section 2.3, the General tab of the Node Definition dialog contains an Outcomes list box within which you can add, delete, insert or rename chance outcomes. The outcomes reflect the names specified earlier within File | Options | General (Figure 2-6). Furthermore, it designated the "Success" outcome as the default outcome. You don't need to change this setting for the purposes of this tutorial.

- ⇒ Type "Phase 1" for the node name. Leave the outcomes as they are.
- ⇒ Switch to the Data tab.

The data input tree for discrete chance nodes includes the following input options for each outcome:

1. Distribution type drop-down list,
2. a Probability edit box,
3. and a Value edit box

These input options are labeled in Figure 2-11.



**Figure 2-11. Data Inputs for Discrete Chance Node Outcomes**

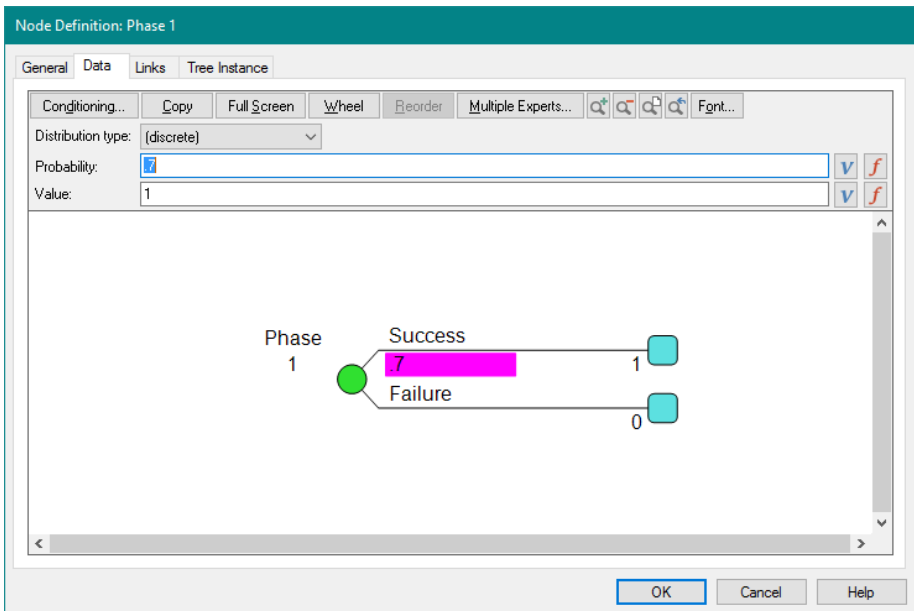
The Distribution type drop-down is currently set to "(discrete)" which is what you want – as you will be entering explicit probability and value data throughout this tutorial. For certain chance events, you may find that it's more appropriate to select a named distribution from the list for a node's probability and/or value data. For more information on named distributions, refer to the Online Help (pressing F1 launches a web browser with contextual Online Help).

Presently, the probability input for the Success outcome is ".7" and is selected for editing. Recall that ".7" is the default probability specified within File | Options | General for the Success outcome for new chance nodes. Note that you can specify numbers or formulas in the probability

edit box but you'll be leaving the probability as it is. There is a 70% chance of success in Phase 1 of the project and that it continues to Phase 2.

- ⇒ Press the down arrow key to select the value edit box for the Success outcome and enter a "1".
- ⇒ Press the down arrow twice (skipping over the probability input for the Failure outcome) and enter a "0" for the value input for the Failure outcome. Press the Enter key.

The Data tab of the Node Definition dialog should now look like Figure 2-12

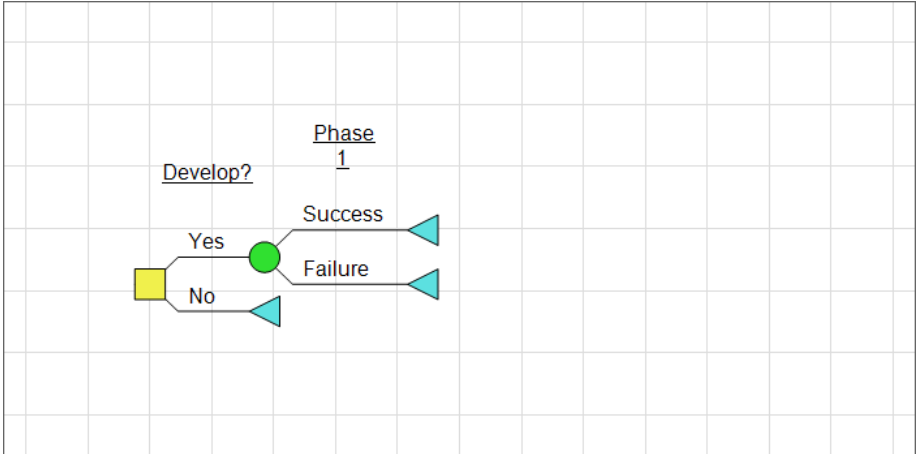


**Figure 2-12. Data Input Tree for Phase 1 Discrete Chance Node**

Note that you entered a probability for only one of the two uncertain outcomes, DPL will automatically calculate the other (1-p). This is a small time saver allowed by DPL.

- ⇒ Click OK to close the Node Definition dialog. Your model should now look like Figure 2-13.



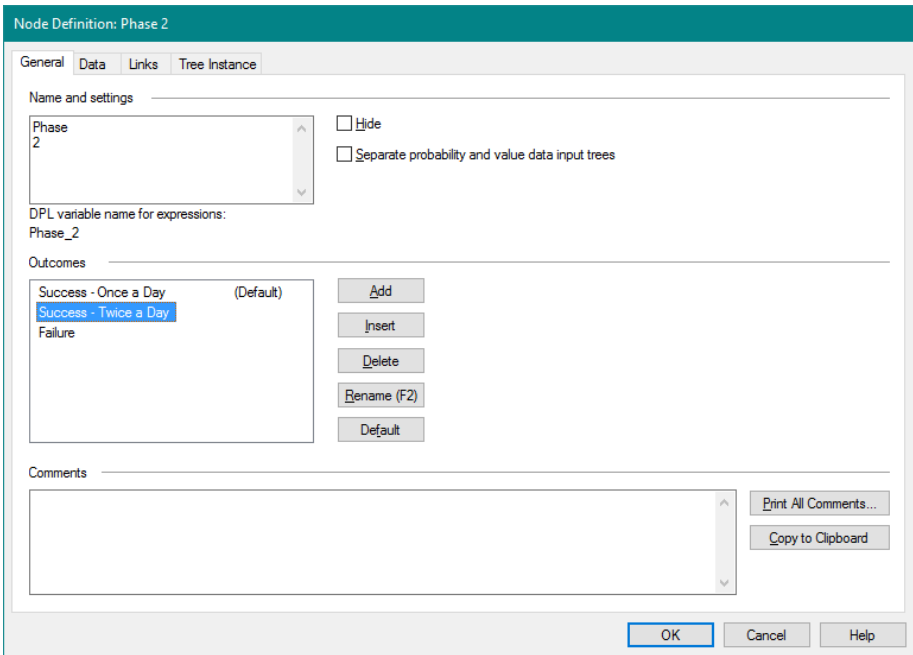


**Figure 2-13. Decision Tree with Newly Added Chance Node**

Again, you should note that each of the Phase 1 outcomes has its own unique blue endpoint node. Notice as well, that the Phase 1 node is connected only to the Yes alternative, and not the No alternative. The No alternative ends in a blue endpoint indicating that it is a terminal alternative (DPL will stop calculating the path at that point). Now you'll add the next uncertain event, which has a slightly different structure than the first.

- ⇒ Click Decision Tree | Node | Add to add another discrete chance node to the tree.
- ⇒ Place the node on the endpoint for the Success outcome of the Phase 1 uncertainty. The General tab of the Node Definition dialog appears.
- ⇒ Type "Phase 2" for the node name.
- ⇒ Within the *Outcomes* box, double-click the first outcome (Success) to enter text edit mode.
- ⇒ Edit the outcome name to be "Success - Once a Day" and press the Enter key.
- ⇒ Arrow down once to select the next outcome. Click the Insert button to insert a new outcome. Name this new outcome "Success - Twice a Day" and press Enter.

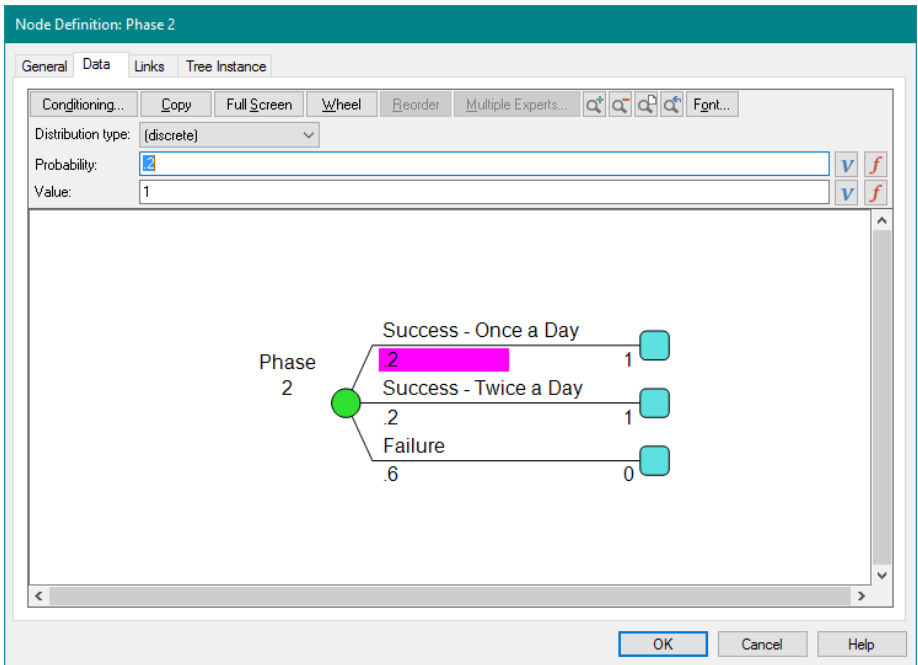
- ⇒ Leave the Failure outcome as is. The General Tab should look like Figure 2-14.



**Figure 2-14. General Tab of Node Definition Dialog for Phase 2 Chance Node**

- ⇒ Switch to the Data tab.
- ⇒ Enter .2 for the probability of Success – Once a day.
- ⇒ Arrow down once, enter 1 for the value of Success – Once a day.
- ⇒ Repeat the process to enter .2 and 1 for the probability and value of Success – Twice a Day, respectively.
- ⇒ Repeat the process again to enter .6 and 0 for the probability and value of Failure, respectively.

The Data tab of the Node Definition dialog for Phase 2 should now look like Figure 2-15.

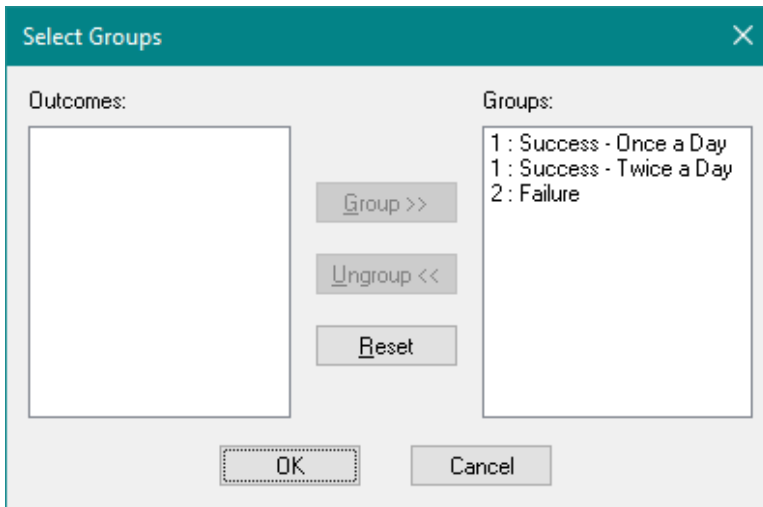


**Figure 2-15. Data tab of Node Definition dialog for Phase 2 Chance Node**

The outcome grouping for this 3-outcome uncertainty is mixed, meaning some subset of the states leads to the same next event, while another subset leads to a different next event. More specifically, the two success outcomes will share a single blue endpoint node and the failure outcome will have its own endpoint. You will modify the node's outcome grouping setting via the Tree Instance tab.

- ⇒ Switch to the Tree Instance tab.
- ⇒ Within the *Outcome grouping* section select the Mixed radio button. This will launch the Select Groups dialog.
- ⇒ Select both the "Success - Once a Day" and "Success - Twice a Day" outcomes by pressing the Ctrl key and clicking each one.
- ⇒ Click the Group >> button.
- ⇒ Select the Failure outcome and click the Group >> button.

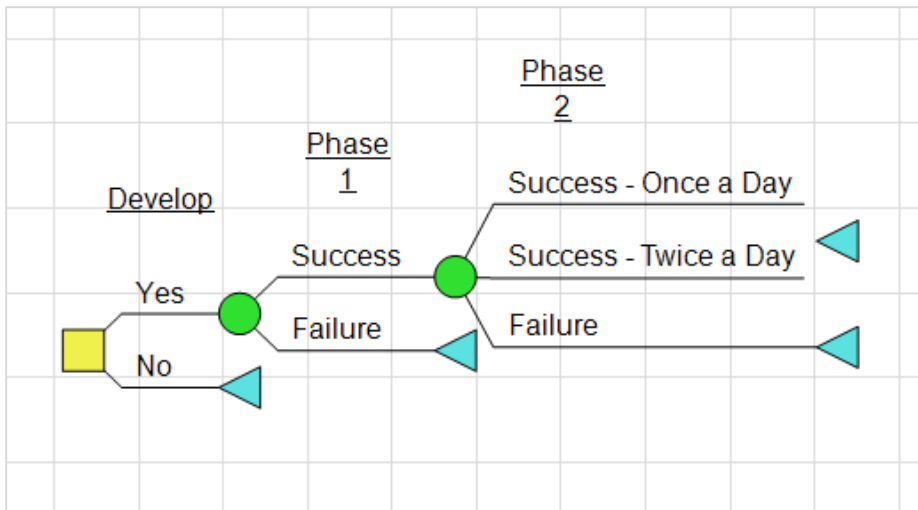
If you make a mistake, click the reset button to reset the outcomes and try again. The Select Group dialog should look like Figure 2-16.



**Figure 2-16. Select Groups dialog for Phase 2 Chance Node**

⇒ Click OK twice to close both dialogs.

Your Decision Tree should now look like Figure 2-17.

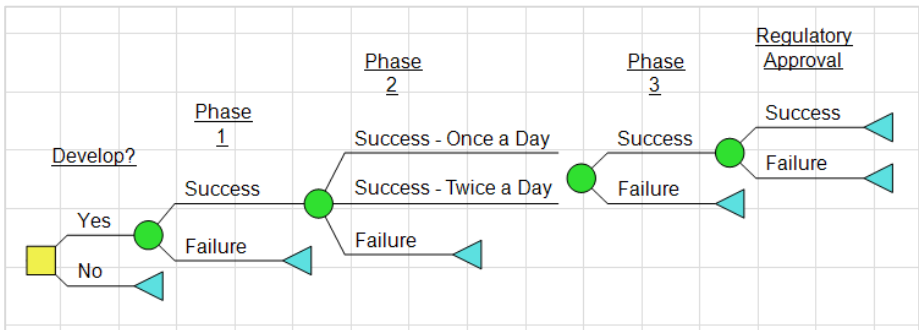


**Figure 2-17. Decision Tree with Newly Added Mixed Outcome Chance Node**

Next there are two additional 2-outcome, Success/Fail Discrete Chance nodes to incorporate to round out the technical and regulatory hurdles.

- ⇒ Create a new discrete chance node and place it on the endpoint for the Success outcomes of the Phase 2 node.
- ⇒ Type "Phase 3" for the node name.
- ⇒ Switch to the Data tab and enter .8 for probability of success. As before, you can leave the probability blank for Failure.
- ⇒ Enter values of "1" for Success and "0" for Failure. Click OK.
- ⇒ Create another new discrete chance node, place it on the endpoint for the Success outcome of Phase 3 and name it "Regulatory Approval".
- ⇒ Switch to the Data tab and enter .9 for probability of success.
- ⇒ Enter values of "1" for Success and "0" for Failure. Click OK.

Your Decision Tree should now look like Figure 2-18.



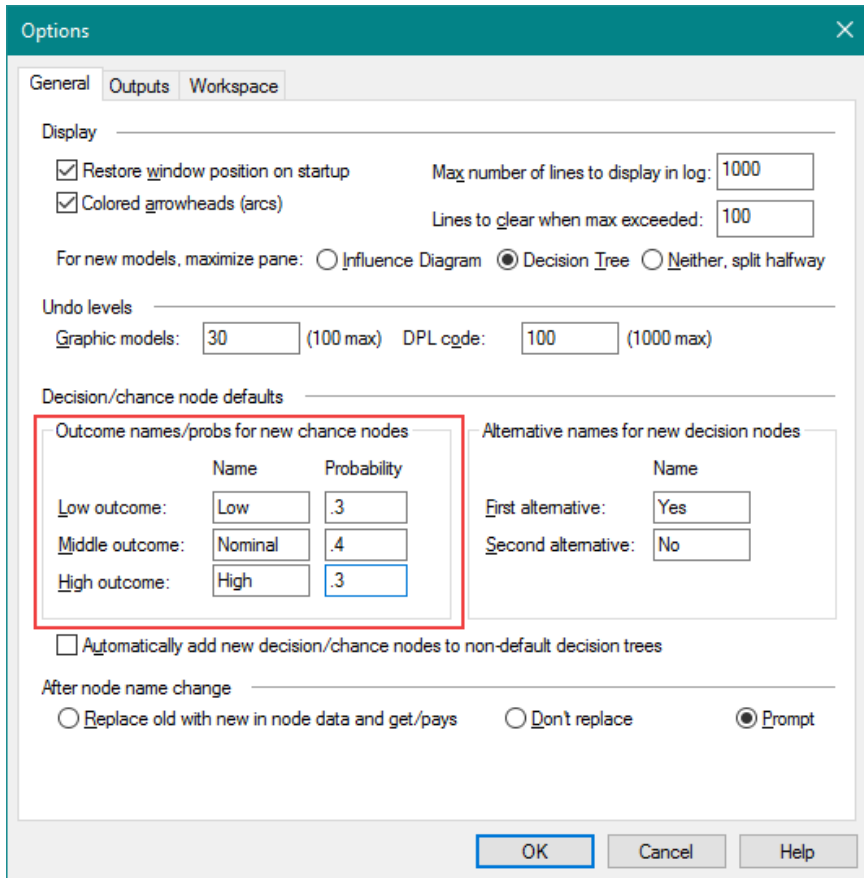
**Figure 2-18. Decision Tree with Technical and Regulatory Uncertainties**

At this point, you may want to name your DPL model and save your Workspace file.

- ⇒ Select the item for the model in the Workspace Manager (DPL gave it the default name of "Model1").
- ⇒ Press F2 to edit the name.
- ⇒ Type "Drug Development" or any other name you'd like.
- ⇒ Press Enter to save the edit.
- ⇒ Save the Workspace to a convenient location and give it a name of your choice using File | Save.

Now you will add a few market uncertainties to the model. The structure of these uncertainties are more similar to the DPL's original defaults (e.g., 3-outcome and symmetric), so you'll revert back to those settings.

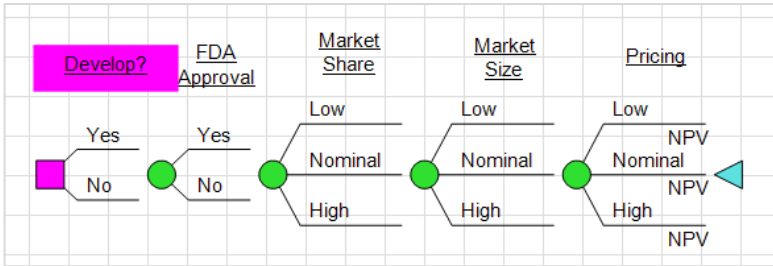
- ⇒ Go to File | Options | General.
- ⇒ Under the *Outcome names/probs for new chance nodes* heading, edit the name and probabilities to match Figure 2-19.



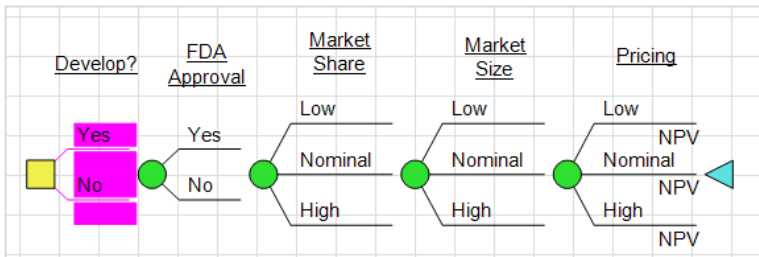
**Figure 2-19. Outcome Names and Probabilities for New Chance Nodes Reverted to Original Defaults**

- ⇒ Click OK to close the File Options dialog.
- ⇒ Click Decision Tree | Model | Settings and set the *Alternative/outcome grouping for newly added nodes* radio button to Symmetric.
- ⇒ Click OK to close the Model Settings dialog.

Before editing the tree further, it should be noted that in DPL you can have either a node or branch(es) selected within the Decision Tree. Note: DPL indicates what is selected by coloring it magenta. In Figure 2-20, the Develop? **node is selected**. In Figure 2-21, the **branches are selected** of the Develop? node.



**Figure 2-20. Develop? Node Selected in Decision Tree**



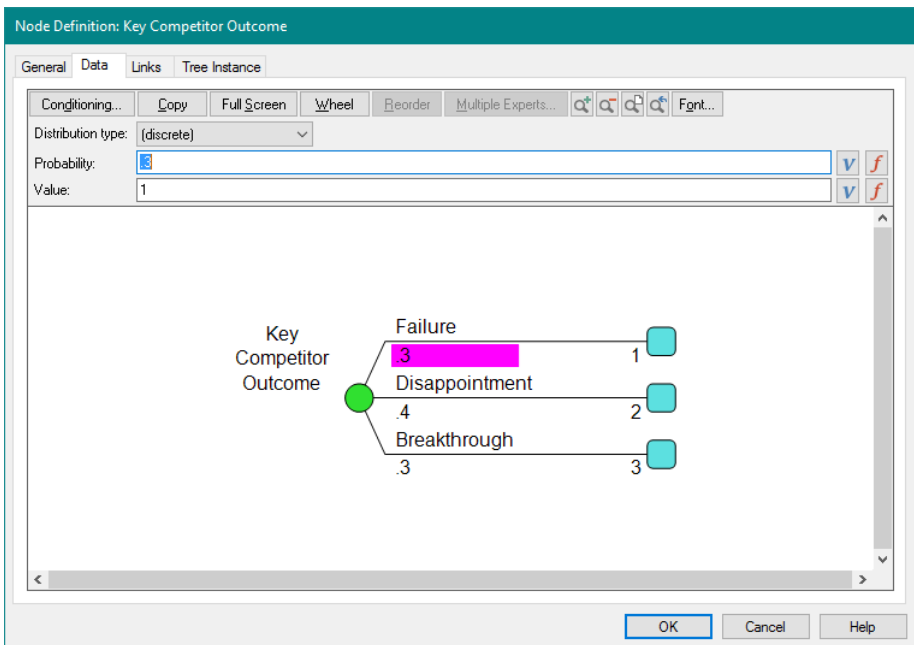
**Figure 2-21. Develop? Branches Selected in Decision Tree**

Double-clicking a node in the Decision Tree launches the Tree Instance tab of the Node Definition dialog. Double-clicking the branches of a node in the Decision Tree brings up the Get/Pay tab of the Branch Definition dialog (more on this later). Further to this, certain button and edit boxes (for example, the Edit Get/Pay box) on the command ribbon are active/inactive depending on what is selected in the Decision tree (node or branch(es)). Lastly, you can have multiple items and a mixture of nodes and branches selected in the Decision Tree. Many of the commands on the Decision Tree tab can be executed when you have multiple items selected, regardless of whether you have a mixture of nodes and branches selected.

You will now continue building out the model by adding the market uncertainties to the Decision Tree.

- ⇒ Add a new discrete chance node to the model via Decision Tree | Node | Add.
- ⇒ Place the node underneath the Develop decision in the tree. It should not be connected to any other nodes yet.
- ⇒ Type in "Key Competitor Outcome" for the node name and rename the outcomes to be "Failure", "Disappointment", and "Breakthrough".
- ⇒ Switch to the Data tab and leave the default probabilities.
- ⇒ Select the right end of the Failure branch in the data input tree.
- ⇒ Enter 1.
- ⇒ Arrow down twice and enter 2 for the value of Disappointment.
- ⇒ Repeat to enter 3 for the value of Breakthrough.

The data input tree for Key Competitor Outcome should look like Figure 2-22.

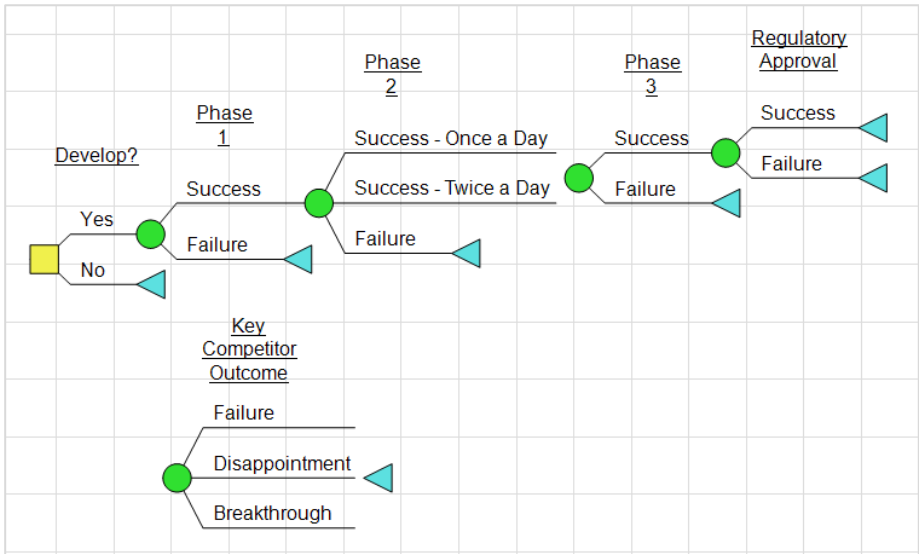


**Figure 2-22. Data Input Tree for Key Competitor Outcome**

- ⇒ Click OK to close the dialog.



Your Decision Tree should now look like Figure 2-23.



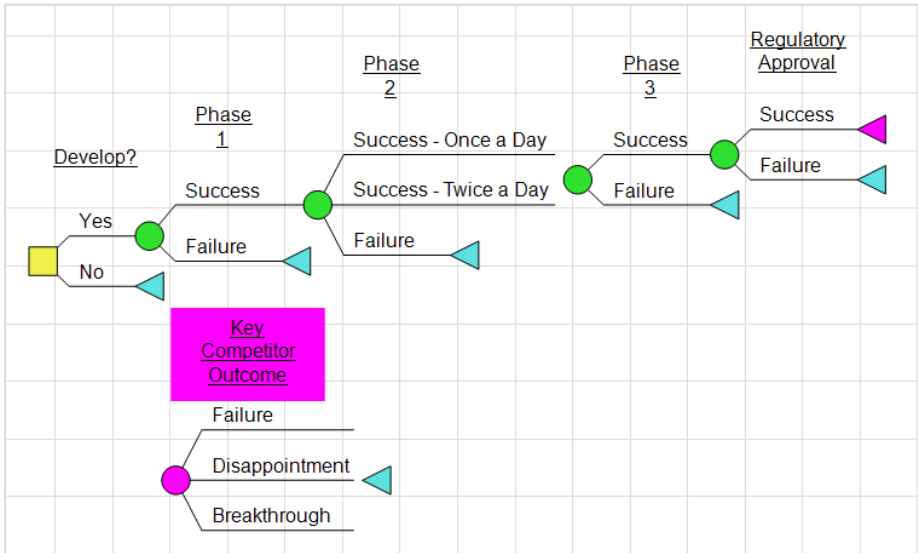
**Figure 2-23. Decision Tree with Key Competitor Outcome Added**

You may be wondering why the Key Competitor Outcome isn't connected to the rest of the tree. For large Decision Trees, you can improve readability (i.e., increase the zoom level) by breaking the tree into two or more sections and placing each section in a more easily viewable space. But you'll need to link the subtree sections together via DPL's Perform Subtree command.

Sections of a Decision Tree are referred to as subtrees. When a subtree is needed in more than one place in the Decision Tree, you can use the Perform Subtree feature to avoid having redundant instances of the same set of nodes. In this case, you're only using the Perform Subtree feature for cosmetic purposes, so it is performed just once. You'll do this in order to link the Key Competitor Outcome event (and those that will eventually follow) to the Success endpoint of the Regulatory Approval node.

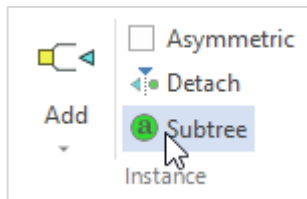
⇒ Select both the endpoint for the Success outcome of the Regulatory Approval node and the Key Competitor Outcome node by pressing the Ctrl key and clicking both items.

The selections should look as they do in Figure 2-24.



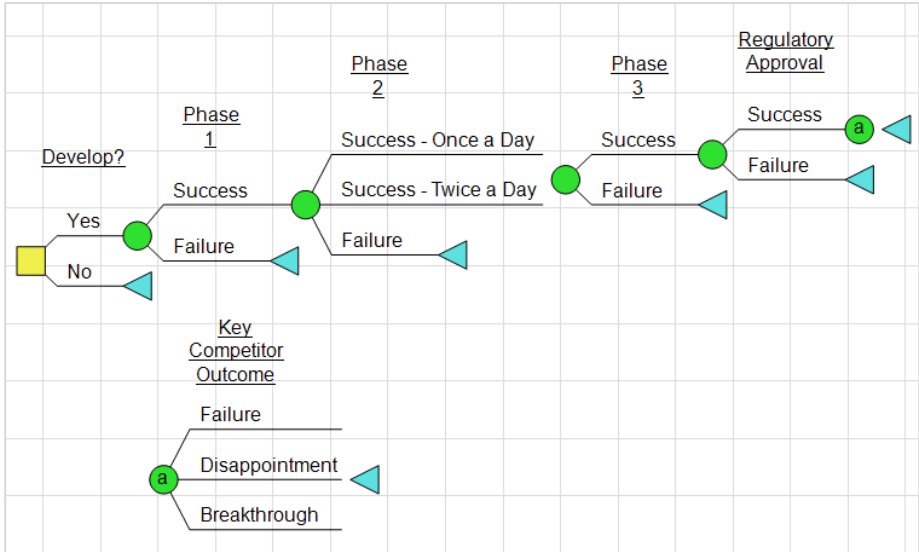
**Figure 2-24. Decision Tree with Items Selected for Performing a Subtree**

⇒ With the items selected, click the Decision Tree | Instance | Subtree button as shown in Figure 2-25.



**Figure 2-25. Decision Tree | Instance | Subtree Command**

DPL assigns a label consisting of a single letter on the node to be performed (the perform target) at the head of the subtree and also adds a node of the same type and with the same label as the perform target just before a new connection endpoint (the perform reference) (Figure 2-26). In effect, you have copied the node and its subtree to the point from where it is to be performed.

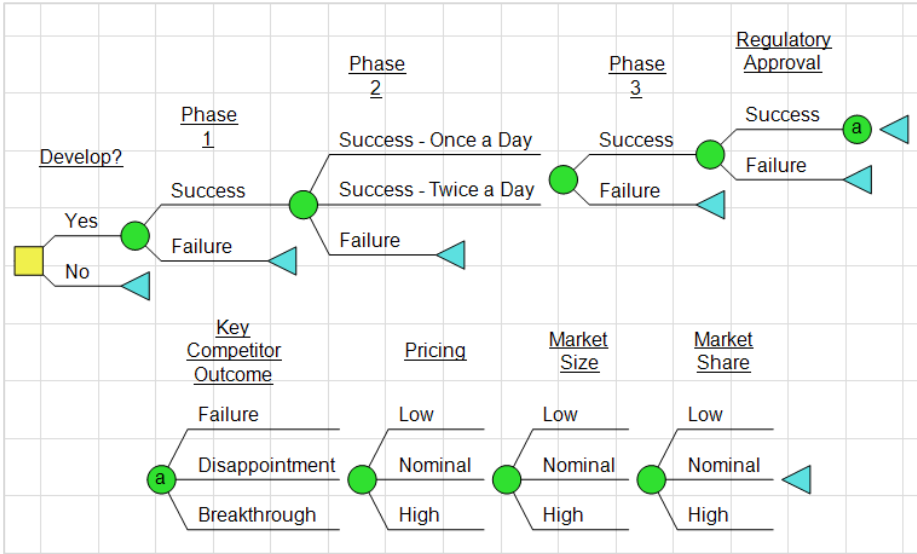


**Figure 2-26. Decision Tree with Perform Subtree Link**

Now you'll add the remainder of the market uncertainties: Pricing, Market Share and Market Size.

- ⇒ Click Decision Tree | Add | Discrete Chance and place it on the endpoint for Key Competitor Outcome.
- ⇒ Type "Pricing" for the node name and leave the default outcomes.
- ⇒ Enter values of "1", "2", and "3" for the Low, Nominal, and High outcomes. Be sure to enter in the values position and leave the default probabilities as is.
- ⇒ Click OK to close the Node Definition dialog.
- ⇒ Just as you did for Pricing, add two more chance nodes to the tree for Market Size and Market Share. Each will have the default probabilities and flag values of "1", "2", and "3" for the Low, Nominal and High outcomes, respectively.

Your Decision Tree should now match Figure 2-27.



**Figure 2-27. Decision Tree with Technical, Regulatory, and Market Uncertainties**

## 2.5 Adding Probabilistic Conditioning

The Marketing Team believes that the Market Share outcome will depend heavily on the dosing outcome of Phase 2. If a once-a-day formulation is achieved, a significant share of the market will be captured. On the flip-side, if only a twice-a-day formulation is achieved a much lower market share will be captured.

In DPL, you can model this conditioning explicitly within the Market Share's node data. You can specify that the dosing outcome of the Phase 2 chance node influences the **probabilities** of the Market share chance node – but not the values. To model this most efficiently you'll separate the Data input tree for Market Share into two: one for probabilities and one for values.

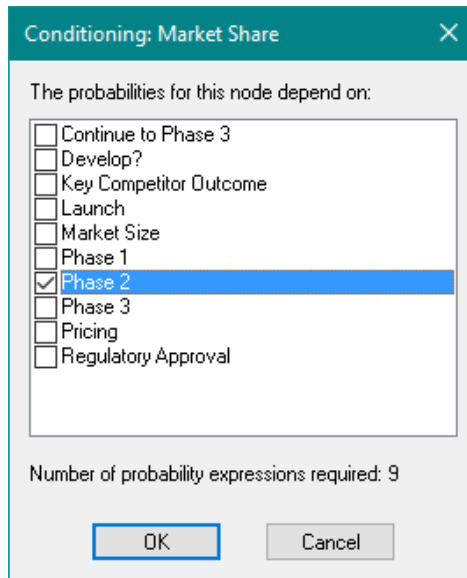
- ⇒ Double-click the Market Share node (not the branches) to edit its data.
- ⇒ Switch to the General tab and check the *Separate probability and value data input trees* box.

Note that the Data tab has been split into tabs: one for entering value data (*Values*) and the other for entering probability data (*Probabilities*). You'll be editing just the Probabilities.

⇒ Switch to the Probabilities tab and click the Conditioning button.

The Conditioning dialog will appear with a list of all the events in the tree.

⇒ Select Phase 2 as shown in Figure 2-28. Click OK to close the Conditioning dialog.



**Figure 2-28. Selecting the Phase 2 Uncertainty to Condition the Probabilities of Market Share**

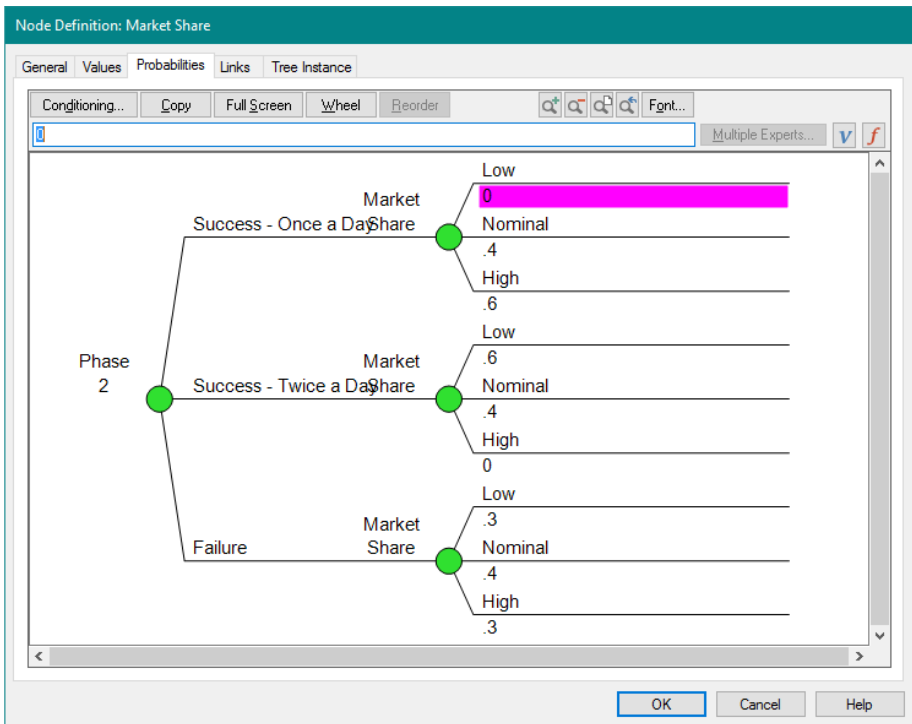
As indicated near the bottom of the dialog, you will now have 9 probability inputs for Market Share: a Low, Nominal, and High outcome for each of the three outcomes of the Phase 2 node.

⇒ Enter the probabilities provided in Table 2-1(left to right, then down to the next row) for Market Share:

	Low	Nominal	High
<b>Success – Once a Day</b>	0	0.4	0.6
<b>Success – Twice a Day</b>	0.6	0.4	0
<b>Failure</b> (not used but required)	0.3	0.4	0.3

**Table 2-1. Probability Inputs for Market Share given Phase 2**

Your data input tree for the probabilities of Market Share should look like Figure 2-29.



**Figure 2-29. Probability Data Input Tree for Market Share with Conditioning**

As you can see from Market Share's probability data input tree, if you succeed in producing a once-a-day formulation you are much more likely to capture significant market share (High outcome is 60%). Whereas for the twice-a-day formulation you're likely to capture a lower portion of market share (60%).

⇒ Click OK to close the Node Definition dialog.

## 2.6 Linking Decision Tree Events to Excel

It is possible to put all value model calculations directly in DPL but most often a DPL Decision Tree/Influence Diagram model is linked to an existing cash flow spreadsheet that does the value model calculations and calculates metrics such as NPV (net present value). You'll proceed by linking the Decision Tree events to a typical cash flow spreadsheet. You can quickly link each of these chance and decision events in the tree to a named, driver cell in the Drug Development.xlsx spreadsheet via the Link Model Events dialog.

⇒ Click Decision Tree | Links | Link Events.

You'll be prompted to browse for the location of your Excel spreadsheet. The spreadsheet is located within the Examples folder of the DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples. The directory may vary slightly depending on your license type (trial or full version) and operating system.

⇒ Select Drug Development.xlsx from the list and click Open.

This will open the spreadsheet and launch the Link Model Events dialog. The first column (*Name*) lists the all events in the model by node type, with decisions coming first, and then in creation order within type. The edit boxes within the *Cell* column are used to link events to a particular driver cell. The *Cell* drop-down lists all the named, driver cells suitable for linking alphabetically by SheetName!Cell\_Name. The last two columns indicate the *Link Type* (Unlinked, Driver, or Metric) and *Node Type* (Decision, Discrete Chance, or Continuous Chance).

⇒ For the Develop decision, click in the Cell edit box. It will become active (i.e., magenta border) and a drop-down arrow will appear.

⇒ Click the drop-down arrow to view the list of driver cells and select "Assumptions!Develop" from the list.

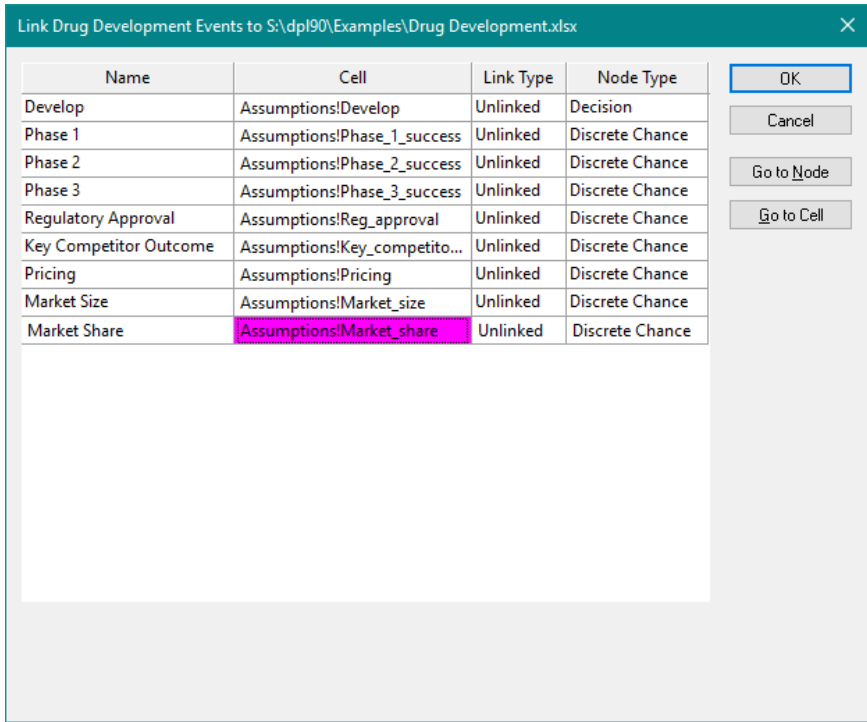
⇒ Continue linking the rest of the events using Table 2-2

<b>Name</b>	<b>Cell</b>
Phase 1	Assumptions!Phase_1_success
Phase 2	Assumptions!Phase_2_success
Phase 3	Assumptions!Phase_3_success
Regulatory Approval	Assumptions!Reg_approval
Key Competitor Outcome	Assumptions!Key_competitor_outcome
Pricing	Assumptions!Pricing
Market Size	Assumptions!Market_size
Market Share	Assumptions!Market_share

**Table 2-2. Node and Cell Names for Linking Model Events**



Your Link Model Events dialog should look like Figure 2-30.



**Figure 2-30. Link Model Events dialog with Driver Cells specified**

⇒ Click OK to close the dialog.

When you build a spreadsheet-linked DPL model as you are now, you will typically have DPL send different sets of drivers to Excel and then ask Excel to send back one or more metrics. Nodes in DPL that send data to Excel are called driver nodes. All of the nodes you have created so far are driver nodes. Nodes that receive data back from Excel are called metric nodes. You will now create a metric node.

## 2.7 Creating and Linking the Output Metric

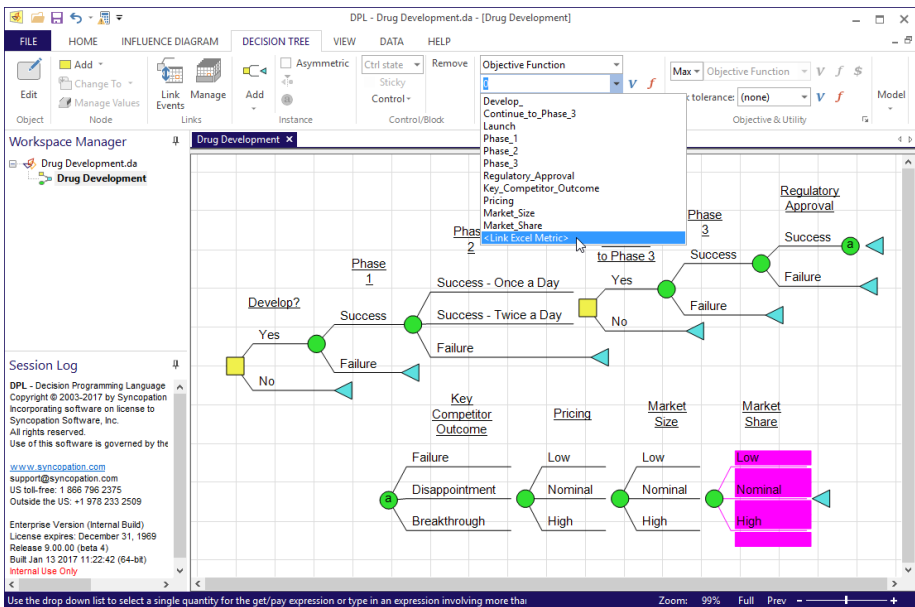
The model is nearly complete. The last step before running a decision analysis is to create and link the output metric to the named, metric cell in Excel and add it to the Decision Tree as a get/pay expression. A get/pay expression tells DPL what value to get (i.e., you receive) or pay (i.e., you

pay out) at each point in the tree. The get/pay expression is displayed below the branch(es) on which it occurs in the Decision Tree.

It should be noted that metric nodes are a type of value node. Value nodes are a number or calculated quantity in DPL. Value nodes do not appear directly in the Decision Tree; rather, the name of value node(s) may appear in get/pay expressions on branches of events in the Decision Tree. Unlike a decision or chance node, a value node is not an "event" and as such does not appear as a node in the sequence of the tree.

With the desired set of branches selected, you'll use the Link Excel Metric command to create and link the output metric and place it in the Decision Tree as a get/pay expression all at once.

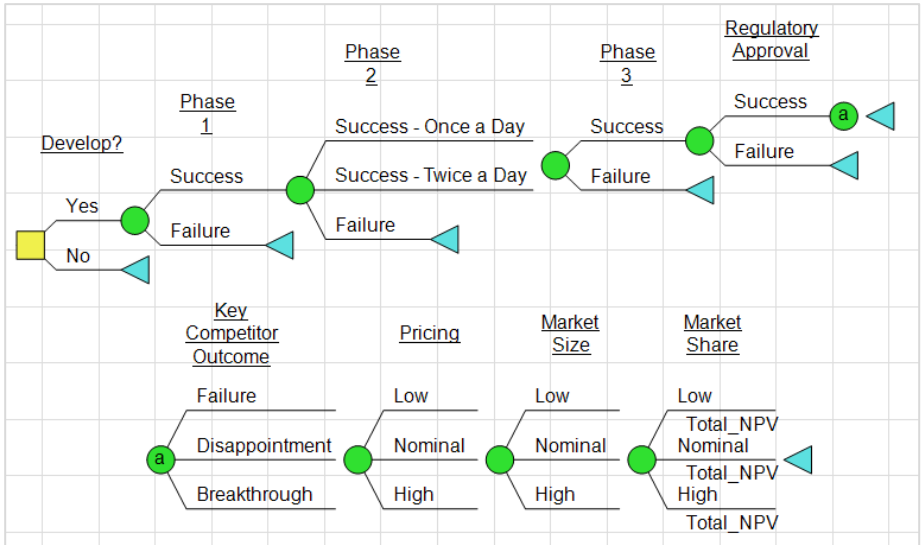
- ⇒ Select the branches of the Market Share node.
- ⇒ Within the Decision Tree | Get/Pay group drop-down the Get/Pay combo box (bottom combo box in the group; see Figure 2-31).
- ⇒ Choose "<Link Excel Metric>" at the bottom of the list as shown in the Figure 2-31.



**Figure 2-31. Using Link Excel Metric Command to Add Get/Pay Expression to Decision Tree**

The Range Names dialog will appear displaying a list of named ranges from the Drug Development.xlsx spreadsheet that contain formulas -- as DPL correctly assumes you are looking for a calculated output metric.

- ⇒ Scroll to the bottom of the list and select "Total\_NPV".
- ⇒ Click Select to close the dialog. The output metric is created and the metric is placed in the Get/Pay expression on the branches of the Market Share node as shown in Figure 2-32.

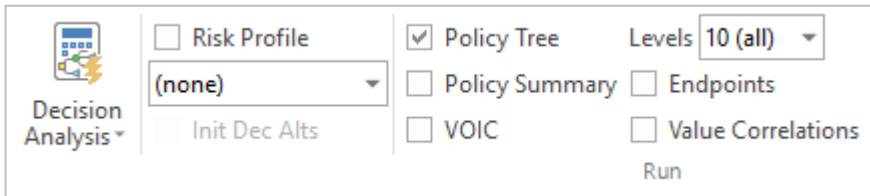


**Figure 2-32. Decision Tree with Linked Output Metric for Get/Pay Expression**

## 2.8 Running a Preliminary Analysis

Now that that all of the events and an output metric are defined and linked to the spreadsheet you are ready to run a preliminary decision analysis on the model to produce results. It's usually a good idea to run the model periodically and check results as it's built to ensure proper set up and logical results.

- ⇒ Navigate to the Home tab. Check Policy Tree in the Home | Run group.
- ⇒ Ensure all other outputs are unchecked. The Home | Run group should match Figure 2-33.

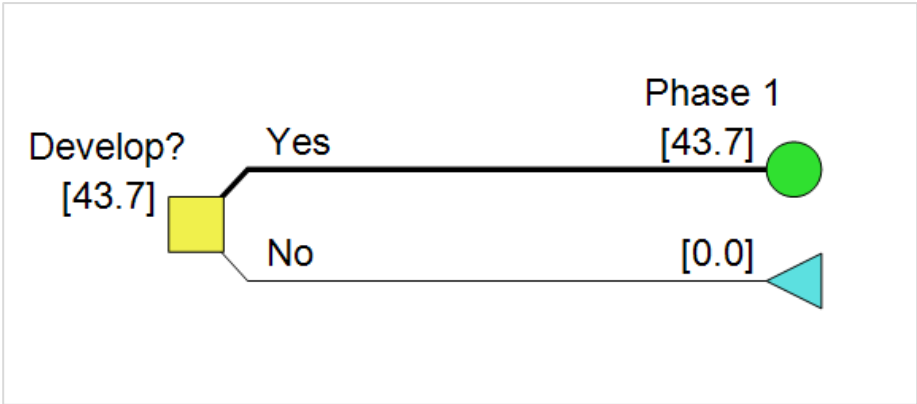


**Figure 2-33. Home | Run Group Selection for Preliminary Analysis**

The default Evaluation Method indicated by the Decision Analysis split button is Fast Sequence Evaluation, which is what you want.

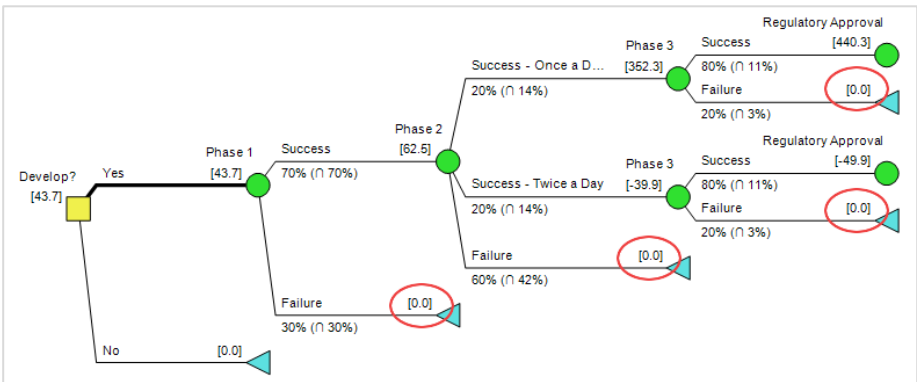
- ⇒ Click top part of the Home | Run | Decision Analysis button. DPL will run the analysis and generate the outputs requested.

When the run is complete the Policy Tree is displayed (Figure 2-34). The Policy Tree output is described in-depth a bit later within Section 3.1. For now it is key to understand that the optimal decision alternative (indicated by a bolded line) is to proceed with developing the drug. If you do so the expected NPV is 43.7 whereas if you decide not to develop the value is zero. You will now expand the Policy Tree in order to inspect some of the scenarios further down the tree.



**Figure 2-34. Policy Tree™ Output for Drug Development Model**

- ⇒ Right-click on the chance node at the end of the Yes branch of the Develop? decision. This will launch the context menu.
- ⇒ Select Expand to Level... from the list. This launches the Expand Policy Tree dialog.
- ⇒ Within the *Expand to level* box, enter "5". Click OK to close the dialog and expand the Policy Tree (Figure 2-35).



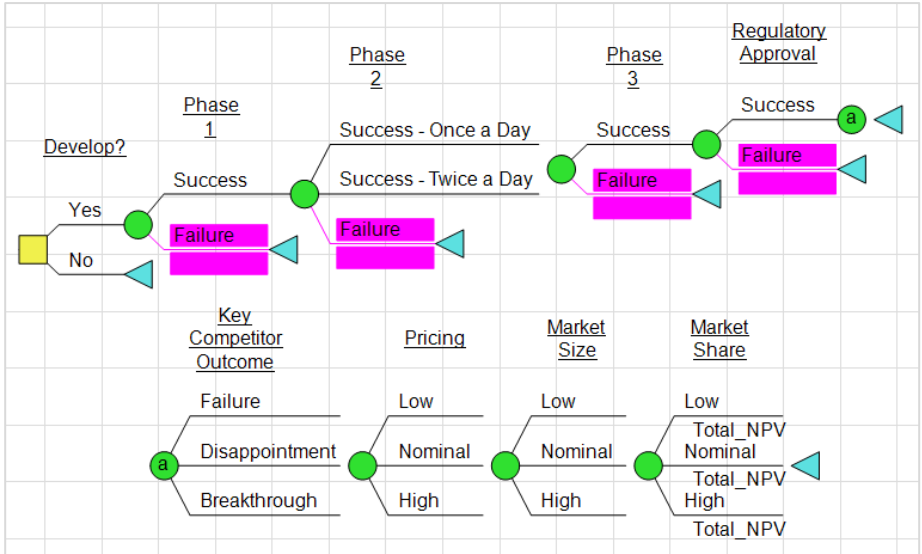
**Figure 2-35. Intial Policy Tree™ Expanded 5 Levels**

Looking further down the tree you'll find that if the project were to fail at any of the technical hurdles the expected value (the value at the end of branches contained in brackets) is zero. DPL assumes a value of zero for the No/Failure points because no get/pays were specified on those branches.

In reality if you were to develop the project through to Regulatory Approval, for example, and fail to be approved, you would incur the cost of R&D up to that point. To correctly model the situation at hand, you will add a Get/Pay expression to each of the Failure branches of the Decision Tree so that costs up to those points are included in the results.

The Drug Development spreadsheet is set up so that a single output metric, Total NPV, is calculated correctly for each scenario in the tree (or path through the tree). I.e., the spreadsheet is set up to calculate this value correctly at each failure point, in addition to all the scenarios if you get regulatory approval the end of the tree. Note that instead of using one metric for all cash flows associated with the project you could separate cash flow into multiple metrics, such as development costs, launch costs and revenues/marketing costs once the product is on market. It would be your choice as to how you'd like to set up your spreadsheet logic and resulting Get/Pay expressions. Importing data from several places in the spreadsheet can result in a model that is more flexible and efficient, and can also help communicate the model's logic.

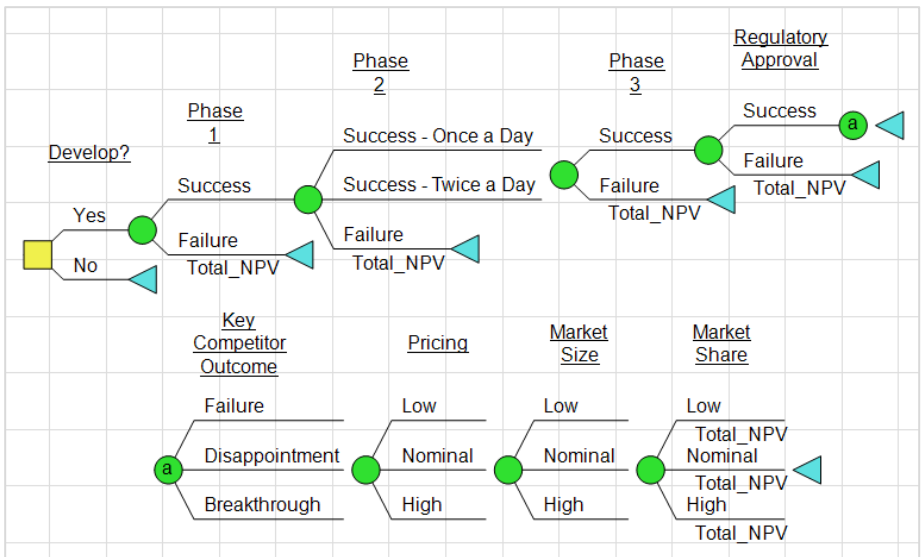
- ⇒ Double-click on the model within the Workspace Manager to activate it (or press Ctrl+F12).
- ⇒ Select the branches of the Market Share node and press Ctrl+C to copy the get/pay expression to the clipboard.
- ⇒ Select the Failure branch of Phase 1 and then hold down the Ctrl key to select each of the remaining Failure branches. Your selections should look as they do in Figure 2-36.



**Figure 2-36. Decision Tree with All No/Failure Points Selected**

⇒ Press Ctrl+V to paste the get/pay expression to the selected branches.

You can click in whitespace to de-select the branches. In addition to the last node, Total\_NPV has been added as a get/pay expression on all of the Failure branches in the Decision Tree (Figure 2-37).



**Figure 2-37. Decision Tree with Get/Pay Expression on All Failure Points**

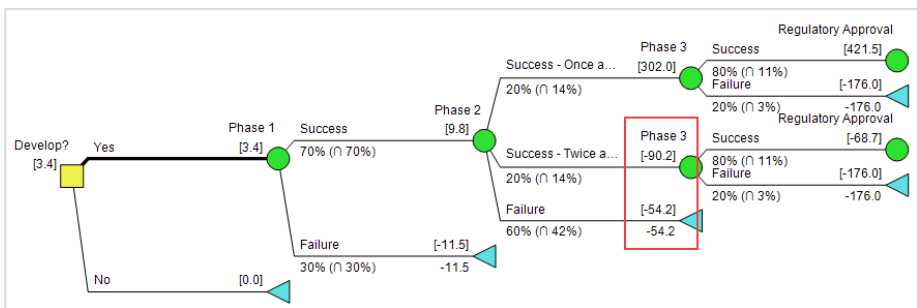
You'll run another analysis to see how the Policy Tree results have changed. On the Home tab, ensure that Policy Tree is the only output requested.

⇒ Click Home | Run | Decision Analysis.

DPL gives a warning that any previously generated outputs will be overwritten by the run, which is OK for the purposes of this tutorial. Click Yes to continue with the analysis. When the run is complete the new Policy Tree is displayed

⇒ As before, right-click on the chance node at the end of the Yes branch for the Develop? decision and select Expand to Level... within the context menu.

⇒ Enter "5" within the Expand Policy Tree dialog. Your Policy Tree should look like Figure 2-38.



**Figure 2-38. Policy Tree™ output for Model with Multiple Get/Pays**

The Expected Value of the model now stands at 3.4, a significant drop when compared to the results of the previous run (43.7). This decrease makes sense as the model now accounts for the R&D costs incurred at various failure points within the tree. The optimal decision alternative remains to develop the drug but the expected value has dropped significantly.

After a review, management has asked you to consider options that will mitigate some of the project risk. One particular scenario in the Policy Tree has caught their attention. It's circled in red above in Figure 2-38. If you achieve the Success – Once a Day outcome in Phase 2 the Expected Value heading into Phase 3 is 302. But if you have to settle for the Success –



Twice a Day outcome, the Expected Value is -90.2. Continuing at this point is actually worse than failing at Phase 2 altogether (EV: -54.2).

This tells you that the viability of the development project depends largely on whether or not a once-a-day formula can be developed. Consequently, you will add an explicit downstream decision to the tree as to whether to continue to develop the drug through phase 3, given the potential outcomes of the Phase 2 uncertainty.


## 2.9 Adding a Downstream Decision

---

Asymmetric trees often involve downstream decisions that represent risk mitigation options or future opportunities. These downstream decisions are sometimes referred to as real options. A real option is analogous to a financial option. With a financial option, you buy the right but not the obligation to purchase (or sell) an underlying financial asset (e.g., a share of stock) at some future date. With a real option, you invest an initial sum of money in a project or venture which gives you the ability but does not obligate you to invest more money in the project or venture at a future date after learning something about the project/venture. E.g., you could pay for some market research on a product and subsequently decide whether to launch the product or not. In this market research example, the real option is the launch decision. You get to decide whether or not launch after learning about the market research results.

In the model you are developing, the development and launch costs are significant so one might ask whether there are situations in which you suspect the product will perform poorly and you would like to know whether you are better off continuing with development in those scenarios. The spreadsheet is set up for this new decision so you'll create the node, link it to Excel, and add it as a downstream decision to the tree.

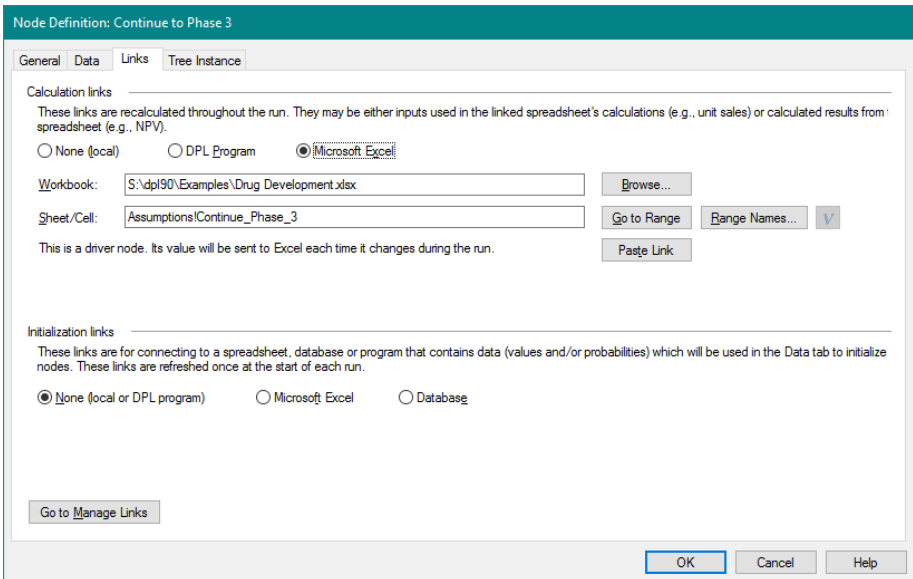
- ⇒ Activate the model by clicking its item in the Workspace Manager.
- ⇒ Switch to the Decision Tree tab of the ribbon.
- ⇒ Create a new decision node by dropping down the Decision Tree | Node | Add split button and selecting Decision from the list.
- ⇒ Place the node on top of the Phase 3 chance node.

You many have noticed that when you hovered the mouse cursor over the node, it changed to a reorder () cursor, indicating that the node will be added to the tree just before the node it's dropped on.

- ⇒ On the General tab of the Node Definition dialog type "Continue to Phase 3" for the node name.
- ⇒ Switch to the Data tab and enter a value of "1" for Yes and "0" for No.
- ⇒ Switch to the Links tab. Within the *Calculation links* section at the top, select Microsoft Excel.

DPL will automatically assume that you'd like to link to the same spreadsheet as the rest of the nodes in the model, which is correct. You'll just need to specify the specific named range. To do so:

- ⇒ Click the Range Names... button.
- ⇒ Within the Range Names dialog double-click on Continue\_Phase\_3. Your links tab should match Figure 2-39.



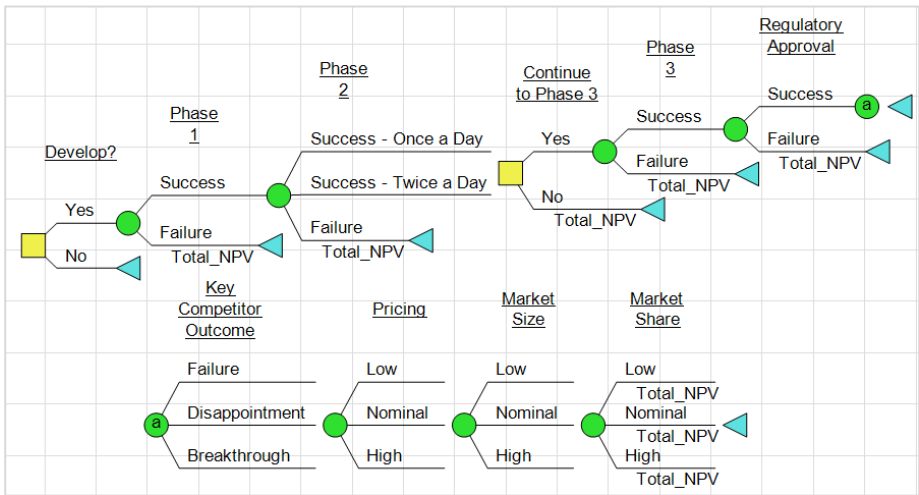
**Figure 2-39. Links tab of Node Definition dialog for Continue to Phase 3 Decision node**

- ⇒ Switch to the Tree Instance tab and change the Alternative grouping to Asymmetric.
- ⇒ Click OK to close the Node Definition dialog.
- ⇒ Re-connect the tree by dragging the Phase 3 chance node (and its subtree) and dropping it on the endpoint for the Yes branch of the Continue to Phase 3 decision.

Lastly, you need to add a Get/pay expression to the No branch of the new decision, indicating that development costs need to be accounted for at that point if you do not continue to Phase 3.

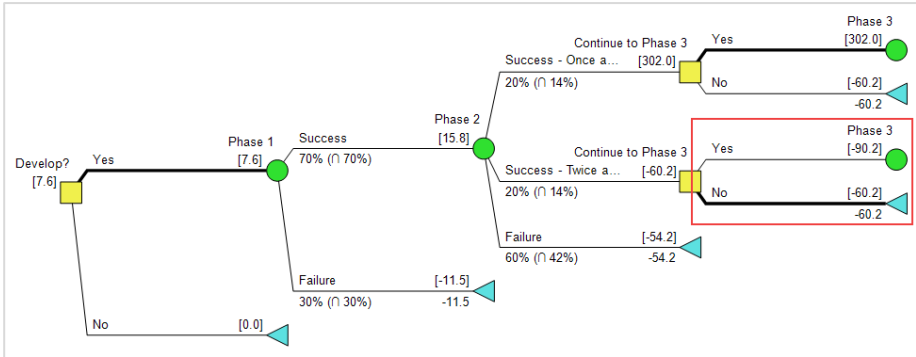
- ⇒ Select the No branch of the Continue to Phase 3 decision.
- ⇒ Drop-down the lower Edit Get/Pay combo box within the Decision Tree | Get/Pay group and select Total NPV from the list.

Your Decision Tree should now look like Figure 2-40. You'll run the model one last time to see if the downstream decision increases the value of the project.



**Figure 2-40. Decision Tree with Downstream Decision Added**

- ⇒ Navigate to the Home tab and click the Decision Analysis button to generate a new Policy Tree (Figure 2-41).



**Figure 2-41. Policy Tree™ output for Model with Downstream Decision**

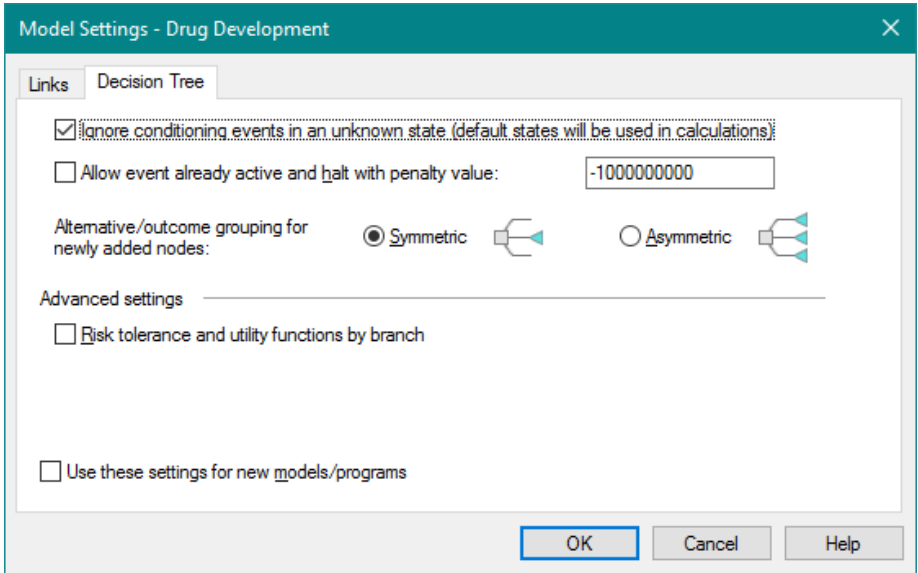
The Expected Value of the model has increased from 3.4 to 7.6. Where is this value coming from? Refer to the section of the tree outlined in red above (Figure 2-41). If you end up with a twice-a-day formulation in Phase 2 the best decision alternative is to not continue to Phase 3. By adding the downstream decision, you have modeled the flexibility to stop the project and cut your losses for this unfavorable scenario. Modeling this flexibility adds value to the project.

Be sure to save your workspace before proceeding. At the start of Chapter 3 you will use the model saved at the end of this section to generate DPL's primary decision analysis outputs.

## 2.10 Ignore Conditioning in Unknown State

Before continuing to the next chapter you will examine an option in DPL that has some bearing on the current model.

- ⇒ Activate the model.
- ⇒ Select Decision Tree | Model | Settings. The Decision Tree tab of the Model Settings dialog will open. Note that the *Ignore conditioning events in an unknown state (default states will be used in calculations)* is checked by default. See Figure 2-42.



**Figure 2-42. Decision Tree tab of Model Settings dialog showing Ignore conditioning events in unknown state Setting**

Without this setting checked, DPL assumes that the output metric depends on every driver in a spreadsheet linked model such as the one you are building. In the failure scenarios in the Decision Tree, several of the drivers are not known when Total NPV is calculated, e.g., when you fail at Phase 1, none of the remaining technical uncertainties nor the commercial uncertainties are in a known state. However, the spreadsheet is set up to be able to calculate Total NPV correctly in each of the failure scenarios. E.g., it knows to ignore Phase 2 costs if you fail at Phase 1 and it knows to include any revenues if you fail at any phase or decided not to continue. By having *Ignore conditioning events in an unknown state* on, DPL assumes that the tree is valid and that the output metric can be correctly calculated in every scenario regardless of unknown states.

⇒ Click Cancel to close the dialog without making any changes.

**Warning:** To safely use this option, you must make sure that the nodes that are skipped do not affect spreadsheet calculations in the scenarios where they are not known. In those scenarios, the values for the "default" state (decision alternative or chance outcome) are sent to the spreadsheet. For example, when Regulatory Approval is Failure, DPL sends the default (in this case the middle or Nominal) values for Key Competitor Outcome, Pricing, Market Share, and Market Size to Excel before telling Excel to

recalculate NPV. The Excel model must be coded to ignore these values when Regulatory Approval is Failure.

You will now look at how to set default states.

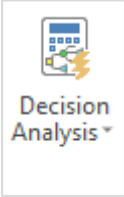


- ⇒ Double-click the Pricing chance node in the Decision Tree.
- ⇒ Click on the General tab. The Nominal outcome is designated as the default state for Pricing.
- ⇒ Select another state and press the Default button. The default changes.
- ⇒ Click Cancel to discard the change.


When you create new chance and decision nodes, DPL assigns a default state. If you delete the default state, be sure to assign a new state as default. Each node should have one of its states assigned as the default within the General tab of the Node Definition dialog. This is particularly important for asymmetric Decision Trees and when using the *Ignore conditioning events in an unknown state* option.

### 3. Analyzing a Decision Analysis Model

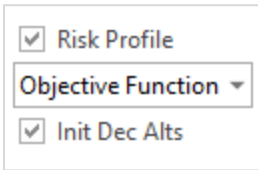
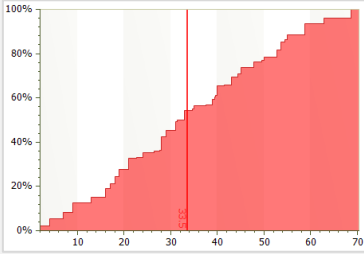
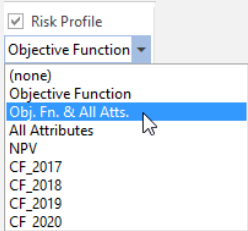
Up to this point, you've only requested a Policy Tree output but it should be noted that within the Home | Run ribbon group there are several output options available when running a Decision Analysis. These options allow you to generate probability distributions, policy outputs, value of information and record endpoints during a run. The model you saved at the end of Section 2.9 is now ready to produce a variety of these results available from the Home | Run group with a decision analysis run.

The majority of the analysis and output options included within the Home | Run group are discussed in detail within Table 3-1, Table 3-2, and Table 3-3.

<b>Control</b>	<b>Description</b>
	<p><i>The Home   Run   Decision Analysis split button tells DPL by what method to analyze the Decision Tree. The methods and their corresponding icons are outlined below.</i></p>
<p><b>Fast sequence evaluation</b></p> 	<p>Provides the exact expected values but optimizes run time by using the structure of the tree to avoid visiting every endpoint of the tree. Risk Profiles (and hence percentiles) may be approximate.</p>
<p><b>Full tree enumeration</b></p> 	<p>Provides exact expected values and visits every endpoint of the tree. Risk Profiles (and percentiles) are exact and optionally the Endpoint Database can be recorded for re-use.</p>

<p><b>Discrete tree simulation</b></p> 	<p>Provides an estimate of expected values by sending a specified number of samples down paths in the tree. Useful to reduce runtime when an exact expected value is not required.</p>
--	--

**Table 3-1. Evaluation Method Types**

<u>Control</u>	<u>Description</u>
	<p><i>This section of the Home   Run group includes controls for probability distribution related outputs that DPL can generate with a model run.</i></p>
<p><b>Risk Profile</b></p> 	<p>Creates a graph showing the full range and likelihood of possible values for the item selected in the Risk Profile Quantity drop-down box (typically the Objective Function). The format of the graph can be changed after the model run.</p>
<p><b>Select Risk Profile Quantity box</b></p> 	<p>Select the quantity for which you would like a Risk Profile. For single attribute models this can only be the Objective Function. For models with multiple attributes or objective functions, you can choose to generate a Risk Profile chart for any single attribute/objective function or all.</p>



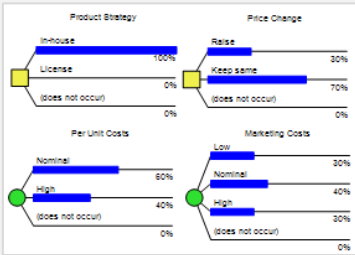
### Init Dec Alts (Initial Decision Alternatives)

Creates risk profile graphs for each of the alternatives of the decision at the head of the tree. Only available when the Risk Profile Quantity box is set to Objective Function, and the model includes only one objective function.

**Table 3-2. Probability Distribution Output Options**

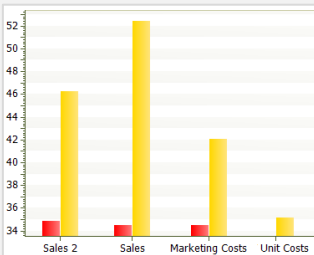
Control	Description
<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <input checked="" type="checkbox"/> Policy Tree      Levels <span style="border: 1px solid #ccc; padding: 2px;">12 (all)</span> ▾  <input type="checkbox"/> Policy Summary    <input type="checkbox"/> Endpoints  <input type="checkbox"/> VOIC                    <input type="checkbox"/> Value Correlations  <div style="text-align: right; margin-top: 5px;">Run</div> </div>	<p><i>This section of the Home / Run group includes controls for decision policy related outputs that DPL can generate with a model run.</i></p>
<h3>Policy Tree™</h3>	<p>Creates a calculated output tree that displays the optimal alternative for each decision, the branch and joint probabilities, get/pay values and rolled-back expected values for the objective function(s) and attributes at each point in the tree.</p>

**Policy Summary™**



Creates a view that summarizes the information in the Policy Tree™. More specifically it displays the policy conditional probabilities for each decision and chance event in the tree.

**VOIC (Expected Value of Perfect Information/Control)**



Creates a chart showing both the expected value of perfect information and the expected value of control for each discrete chance node in the model.

**Number of levels**

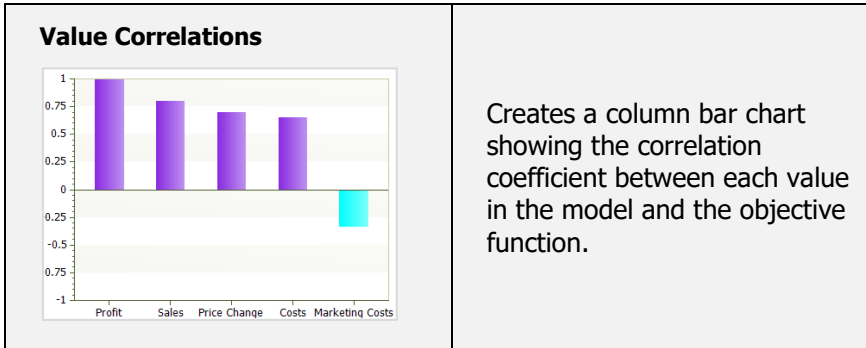
A screenshot of a software interface showing a dropdown menu for 'Levels'. The current selection is '7 (all)'. Other visible options include '(none)', '2', '3', '4', '5', and '6'.

Tells DPL how many levels down the tree to gather policy information. A tree with N nodes in its longest path will have N+1 levels. Must be set to all for generating VOIC charts.

**Endpoints**

	0	1	2	3	4	5
0	Endpoint	Product...	Sales	Per_Unit...	Marketin...	Price_Ch...
1	4	In_house	Low	Nominal	Low	Raise
2	1	In_house	Low	Nominal	Low	Raise
3	2	In_house	Low	Nominal	Low	Raise
4	3	In_house	Low	Nominal	Low	Keep_sa...
5	4	In_house	Low	Nominal	Low	Keep_sa...
6	5	In_house	Low	Nominal	Low	Keep_sa...
7	6	In_house	Low	Nominal	Nominal	Raise
8	7	In_house	Low	Nominal	Nominal	Raise
9	8	In_house	Low	Nominal	Nominal	Raise
10	9	In_house	Low	Nominal	Nominal	Keep_sa...

Tells DPL to record an endpoint database for the run. The endpoint database allows you to "replay" the endpoints to quickly perform further analyses without value model recalculation. If Endpoints is checked, the evaluation method is set to full tree enumeration.



**Table 3-3. Policy Output Options**

The last section of the Home | Run group contains two buttons, Clear Mem (✘) and Compile (🔧). Clear Mem tells DPL to clear the compile structures and any unsaved results from a Decision Analysis run for the active model. The Compile command tells DPL to compile the active model to test for missing data or other errors.

Note that there are some additional evaluation methods in the Decision Analysis drop-down list not included in Table 3-1: Monte Carlo simulation, Full Tree Enumeration from Endpoints, Record Endpoints using Servers and Act as Endpoint Server. Monte Carlo simulation and the options associated with this evaluation method will be covered in Chapter 6. Full Tree Enumeration from Endpoints and Parallel Endpoint Recording will be covered in Chapter 11.

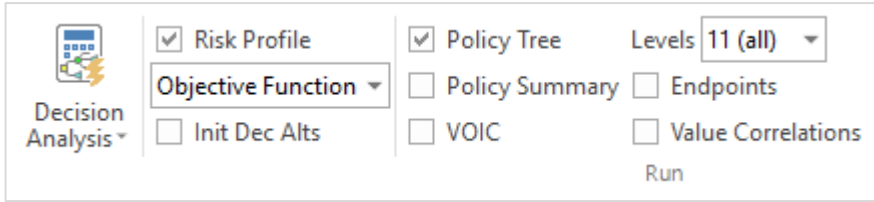
The dialog box launcher within the Home | Run group launches the Run Settings dialog. This dialog contains a number of run and compilation settings -- most of which are self-explanatory. A few of the options will be discussed in detail later in the manual.

You will continue working with the model saved at the end of Section 2.9. If you've closed DPL do the following:

- ⇒ Start DPL and open the workspace you saved at the end of Section 2.9 by using File | Open.

Note: if you're starting here and haven't worked through the tutorial in Chapter 2 or are not confident that the model built in Chapter 2 is correct, open the workspace Drug Development\_DT.da from the Examples folder in your DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples.

- ⇒ The selections within the Home | Run group should match what is shown in Figure 3-1.



**Figure 3-1. Outputs Requested within the Home | Run Group**

- ⇒ Make sure Risk Profile is checked. For this single attribute model, the Select Risk Profile Quantity box just below it is set to Objective Function – the only option.
- ⇒ Make sure Init Dec Alts (i.e., Initial Decision Alternatives) is not checked.
- ⇒ Make sure Policy Tree™ is checked.
- ⇒ Make sure Number of levels is set to "11 (all)".
- ⇒ Make sure Policy Summary™, Expected Value of Perfect Information/Control (VOIC), Endpoints, and Value Correlations are not checked.

The default evaluation method indicated by the Decision Analysis button icon is Fast Sequence Evaluation. This is fine for the purposes of this tutorial.

- ⇒ Click the Home | Run | Decision Analysis icon or press F10.

DPL runs the analysis and produces the requested outputs.

## 3.1 Policy Tree™

DPL creates an item beneath the model in the Workspace Manager for the Policy Tree output and activates it once the run is complete as shown in Figure 3-3. Before discussing the Policy Tree, take another look at the Workspace Manager. Note that DPL has put double chevrons ("»»...««") around the model you just ran. This indicates the model is compiled (i.e., checked for errors and data structures loaded in memory) and is ready to run. Once you modify the model, the compiled indicator is removed.

Take a look at the Session Log as well (Figure 3-2). Note that the number of paths in the model is 177, the Expected Value of the model is 7.6, and it took just over a second for DPL to run the model (the runtime will vary by machine).

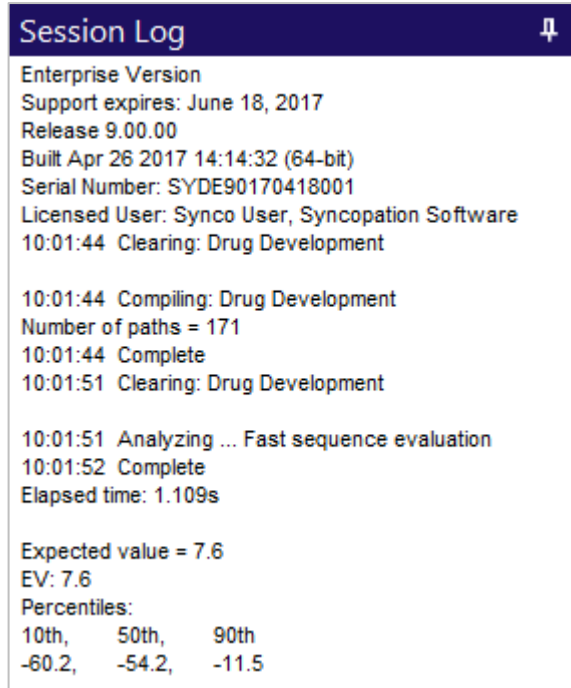


Figure 3-2. Session Log a Analysis Completes

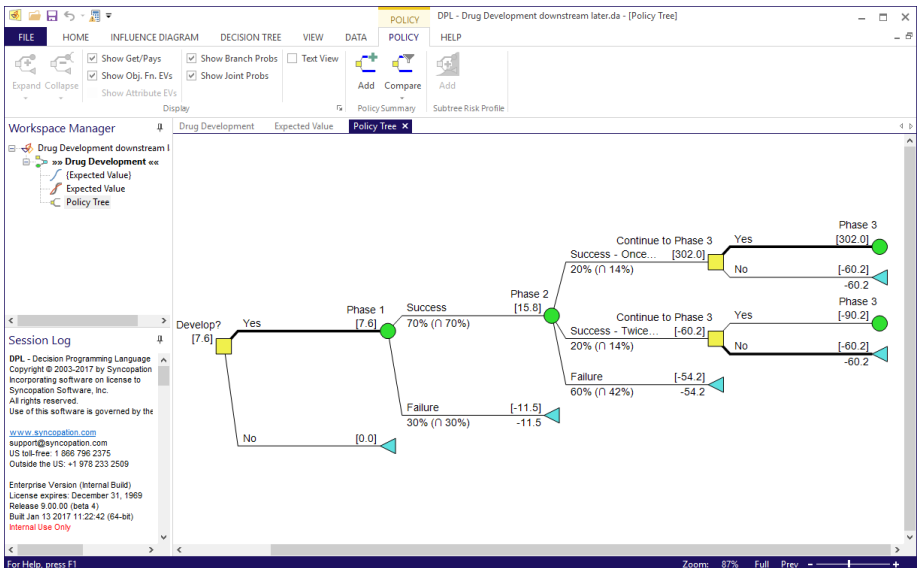
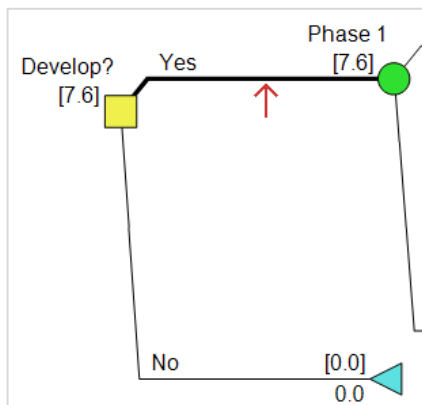


Figure 3-3. Policy Tree™ output for Model

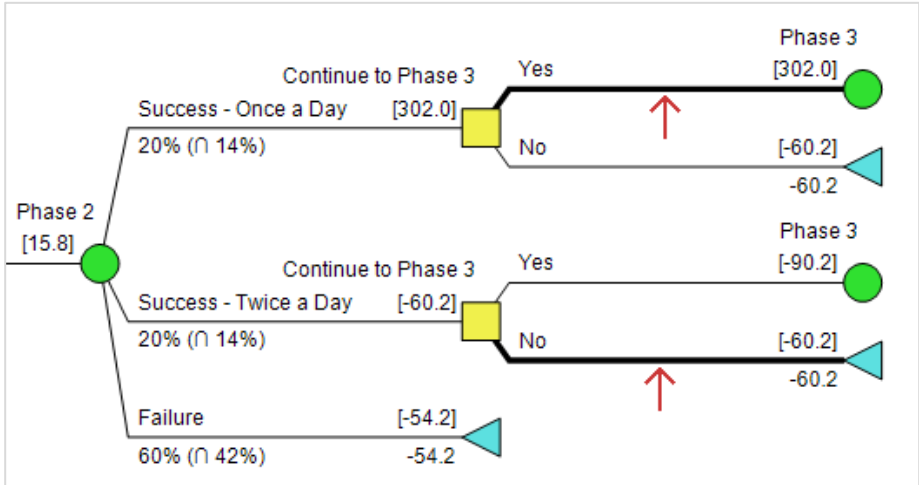
The Policy Tree displays information in a Decision Tree format about the results of the decision analysis. Nodes in the Policy Tree are displayed using the same symbols as in the Decision Tree. Decision nodes are yellow squares. Discrete chance nodes are bright green circles. Endpoints are blue triangles. The first thing to note is the structure of the Policy Tree. To the left it displays the initial Develop? decision with each of its alternative branches (Yes and No). The Yes alternative branch leads into the next event (Phase 1) while the No alternative branch has a blue triangle at the end of its branch. As mentioned previously, the blue triangle is DPL's representation for an endpoint.

The Policy Tree also displays the expected value (or rolled back expected value) at each point in the tree in square brackets ([ ]). The [7.6] to the left of the Develop? decision is the expected value of the objective function for the complete model or tree. Look further down the tree to the two rolled back expected values that appear next and you see that the expected value beneath Phase 1 is 7.6 and the expected value for the endpoint at No is 0. As mentioned in Section 2.3, in this model DPL maximizes the objective function. Therefore when confronted with a decision with two alternatives, one of which has a rolled back expected value of 7.6 and the other a rolled back expected value of 0, DPL chooses the alternative with the rolled back expected value of 7.6. This can be seen in the Policy Tree, not only by the [7.6] next to Develop?, but also by the Yes branch of Develop? being displayed with a thick branch line. DPL indicates the optimal alternative of a decision at each point in the Policy Tree by displaying the optimal alternative with a thick branch.



**Figure 3-4. Optimal Decision Alternative indicated with a Thick Branch Line**

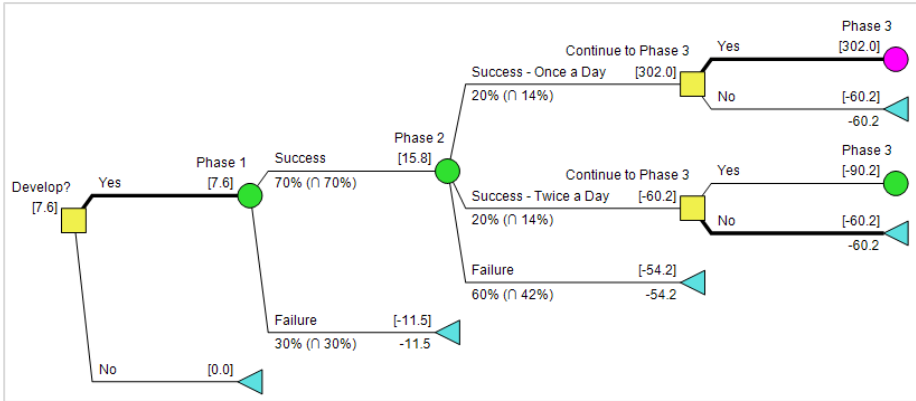
Looking further down the tree (Figure 3-5) you'll see that the optimal decision policy is to Continue to Phase 3 if you succeed in attaining a once per day formulation in Phase 2, but you should cut your losses and not continue if a twice a day formulation is achieved.



**Figure 3-5. Optimal Decision Policy Indicated for Downstream Decision**

To explore even further down the Policy Tree you will expand from the Phase 3 chance node at the end of the tree.






- ⇒ Select the Phase 3 chance node at the end of the Yes alternative of the Continue to Phase 3 decision node as shown in Figure 3-6. Once selected the Expand split button within Policy | Display group becomes active.



**Figure 3-6. Phase 3 Chance Node Selected for Expanding**

The Expand split drop-down lists five menu commands for changing the state of expansion of the Policy Tree. If you were to right-click on the same node in the Policy Tree to bring up the context menu you will find four of the same expand commands, with the exception of the Expand All command. All are summarized in Table 3-4.



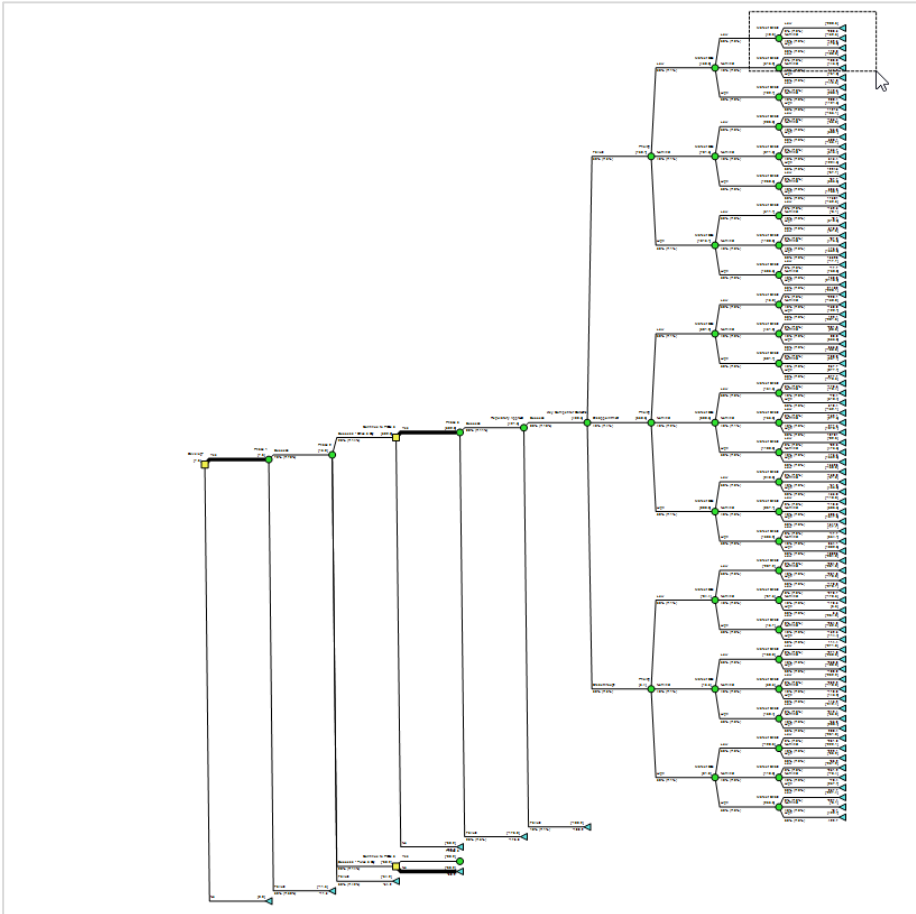
<u>Control</u>	<u>Action</u>
<p><b>Expand</b></p> 	<p>Expands the currently selected node one level if the node has never been expanded before or to the previously expanded state. This same action can also be accomplished by double-clicking the node.</p>
<p><b>Expand and Zoom</b></p> 	<p>Does the same thing as Expand but then zooms the Policy Tree so that the whole tree can be seen. This action can also be accomplished by Shift+double-clicking the node.</p>
<p><b>Expand to Level...</b></p> 	<p>Brings up the Expand Policy Tree dialog in which you can specify the level to which you want the subtree expanded. The Policy Tree is expanded from the node selected.</p>
<p><b>Expand Subtree</b></p> 	<p>Expands the entire subtree in the Policy Tree from the selected node.</p>
<p><b>Expand All</b></p> 	<p>Expands the entire Policy Tree so that all nodes are fully expanded.</p>

**Table 3-4. Policy Tree™ Expand Commands**

- ⇒ With the node still selected, drop-down the Expand split button and select Expand Subtree.
- ⇒ Use the keyboard shortcut Ctrl+L to Zoom Full on the Policy Tree. Your Policy Tree should look similar to Figure 3-7

You should find (if you haven't clicked anywhere) that the chance node you've expanded from is still selected but the Expand button will no longer be enabled on the command ribbon. The Collapse button is now enabled on the ribbon. There are two collapse options, Collapse and Collapse and Zoom. Analogous to Expand, the Collapse command will collapse the tree to the previous level. This action can also be accomplished by double-clicking the node.

Now that the Policy Tree is expanded, you can see several endpoints (displayed as blue triangles) at the right end of the tree. This model contains 177 endpoints (or paths), so the tree would be difficult to read if all the endpoints are displayed at once. Practical applications often involve thousands or millions of endpoints, so it's important to be able to manipulate the Policy Tree to show the desired information at a level that is readable.

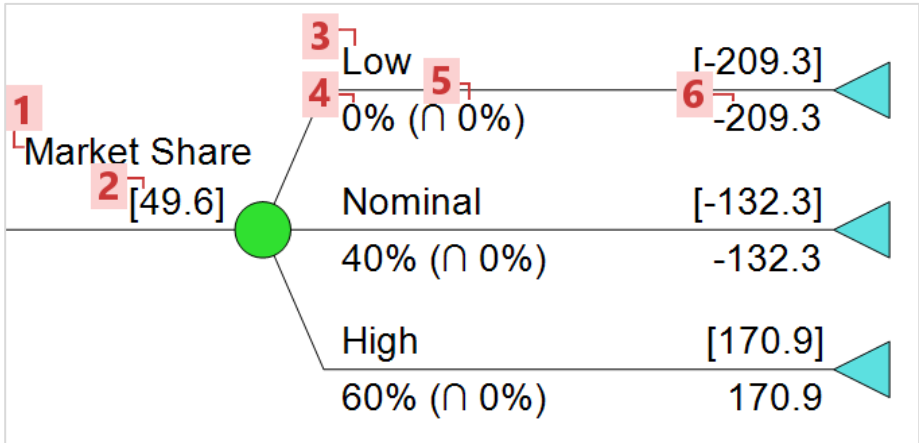


**Figure 3-7. A Policy Tree™ Subtree Expanded to Endpoints**

You'll zoom by selecting in order to get a closer look at a section of the tree that includes endpoints.

- ⇒ At the top right most corner of the output put your cursor just above and to the left of the top chance node as shown in Figure 3-7.

⇒ Press Ctrl+Shift while you click and drag the mouse down and to the right. A box will show you what will be displayed when you release the mouse button. Release the mouse button and keys when the box surrounds the portion of the diagram selected in Figure 3-7. See the zoomed section with the data elements labeled in Figure 3-8.



**Figure 3-8. Data Elements labeled in Zoomed Policy Tree™**

For a description of these data elements labeled above see Table 3-5 and Table 3-6.

<b>Data element (location key)</b>	<b>Definition</b>
Node Name (1)	
Rolled-back expected values (2) Value is displayed in square brackets, e.g., [49.6].	The expected value of the objective function for the subtree headed by the node adjacent to the value

**Table 3-5. Node Information Displayed in the Policy Tree™**

<b>Data element (location key)</b>	<b>Definition</b>
State name (3)	
Branch probability (4) The probability can either be displayed as a percent between 0%-100% or as a number between 0-1. This data is displayed for Discrete Chance nodes only.	The conditional probability of this discrete chance outcome occurring on this path. If the discrete chance is conditioned by other nodes, this is the probability of the chance outcome occurring given the outcomes on this path of the nodes that condition it.
Joint probability (5) Displayed as above. Plus, joint probabilities are indicated by a set intersection symbol and are contained within parentheses, e.g., ( $\cap$ 0.11%).	The joint probability (also referred to as the "path probability") is the probability of the particular path occurring. It is the product of the chance outcome probabilities along the path defining the endpoint.
Get/pay values (6) If there is no get/pay value at this point in the tree, then nothing is displayed.	The value of the get/pay expression (if any) that occurs at this point in the tree.

**Table 3-6. Branch Information Displayed in the Policy Tree™**

Notice that the joint probabilities are all 0%. The joint probability is in fact not zero but a very small percentage. With the current model settings percentages are set to display zero decimal places. If you'd like to update the number formatting to see the path probabilities, do the following:

- ⇒ Go to File | Options | Outputs
- ⇒ Within the *Probabilities* section, enter a "2" within the *Decimal places for probabilities* box.
- ⇒ Click OK to close the dialog.

The Policy Tree display will update to reflect the new number formatting. If you'd like your outputs to exactly match what you see in the tutorials for the rest of the chapter, you should go back into File | Options | Outputs

and change the decimal places for probabilities back to 0 before continuing on.

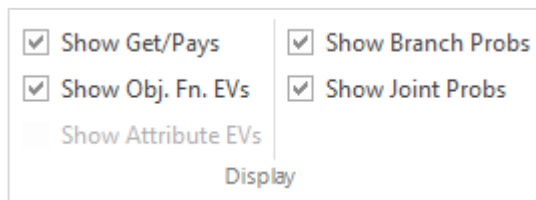
In the model there is a get/pay expression (Total\_NPV) at each Failure outcome and No alternative of the downstream decision in the tree. Therefore, you will see the get/pay value for Total\_NPV at each of these points in the Policy Tree. At each point in the tree, the Total\_NPV value is for a single, specific scenario. You could verify Total\_NPV for that specific scenario in the spreadsheet by setting the inputs in the spreadsheet to match the inputs for that path in the Policy Tree.

At all of the endpoints of the tree (Failure outcomes, No alternative for the downstream decision and all the endpoint at Market Share), the value of the get/pay expression (Total\_NPV) is the same as the rolled-back expected value, which is why the same number appears above and below the branch at these points.

You can use the Policy Tree to find out a number of things about your model. For example, the expected value of the model is -60.2 given that you develop the drug when a twice a day formulation is achieved at Phase 2 (-60.2 is the rolled-back expected for the Yes alternative of the Continue to Phase 3 decision at the Success – Twice a Day outcome for the Phase 2 chance node).

⇒ Try manipulating the Policy Tree by expanding / collapsing different parts of it. You can do this by double-clicking a node, using the context menu or using the Expand/Collapse split buttons within the Policy | Display group on the ribbon.

You can also directly control the data elements that are displayed in the Policy Tree via the Policy | Display group as shown in Figure 3-9.



**Figure 3-9. Data Element Display options within the Policy | Display group**

⇒ Experiment with using the checkboxes to turn on and off the different data elements within the tree display.

There are more formatting options within the Format Policy Tree dialog.

- ⇒ Select Policy | Display | Options (the dialog box launcher) to launch the Format Policy Tree dialog (Figure 3-10).

**Format Policy Tree**

Display

Branch length: 100%      Branch offset: 120%

Show

Node names       State names

Branch probabilities       Joint probabilities

Objective function expected values       Attribute expected values (Eat Policy)

Get/pay values

Select objective function EVs to display:

Select attributes to display (EVs and get/pays):

Branch filter

Based on probabilities       Based on rolled-back objective function EVs

Min: 0      Min: 0

Max: 1      Max: 0

OK      Cancel

**Figure 3-10. Format Policy Tree Dialog**

You can use the two spin buttons at the top of the *Display* section to increase or decrease the Branch length and Branch offset percentages in order to alter the shape and appearance of the tree. The same display elements options that are on the ribbon and included beneath the *Show* heading.

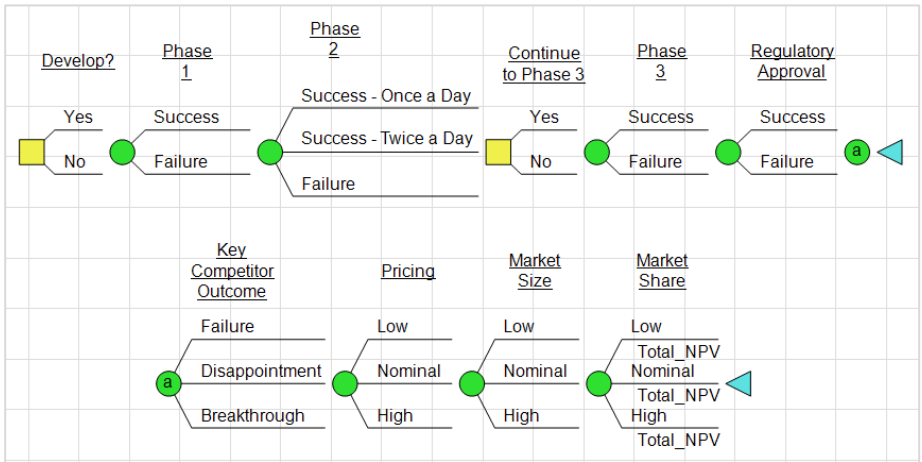
If your model includes multiple objective functions or attributes, you can select which ones to display in the Policy Tree on an individual basis within their respective selection boxes. This will be covered later in Sections 10.2.1 (multiple attributes) and 8.3 (multiple objective functions).

Lastly, beneath the *Branch filter* section you can apply branch filters based on branch probabilities or rolled back expected values.

### 3.1.1 Comparison to Symmetric Decision Tree Version

You might wonder how the model, analysis and results would have differed had you built a completely symmetric Decision Tree for this drug development decision. Note that this is for example purposes only and you will not be making any edits to your current model.

Displayed in this section are some screenshots that show how a symmetric version of the Decision Tree would compare to your current model. Figure 3-11 shows a symmetric version of the Decision Tree with a single get/pay at the end.



**Figure 3-11. A Screenshot of a Symmetric Version of the Drug Development Model**

Figure 3-12 shows the Session Log information at the completion of a Decision Analysis run on the symmetric model.

```

Session Log
10:06:06 Clearing: Drug Development
10:06:06 Compiling: Drug Development
Number of paths = 7776
10:06:06 Complete

10:06:06 Analyzing ... Fast sequence evaluation
10:06:51 Complete
Elapsed time: 44.093s

Expected value = 7.6
EV: 7.6
Percentiles:
10th,    50th,    90th
-60.2,  -54.2,  -11.5

```

**Figure 3-12. Session Log Information for Symmetric Model Analysis**

Note that the number of paths has increased from 177 to 7,776! Furthermore the symmetric model took much longer to complete the run (nearly 50 seconds on the same machine that took just over a second for the asymmetric model). And lastly, you can see that the Expected Value for the model matches that of the current model, 7.6. For this example the asymmetry in the Decision Tree proves to have a significant impact on runtime, results in a model which is much more efficient and generates results that are much more compact and easy to read.

Figure 3-13 shows what the Policy Tree output looks like for the symmetric version of the model. Notice that the values at all the points in the tree for the Phase 1 Failure outcome are the same. The asymmetric version of the tree removes this redundancy.



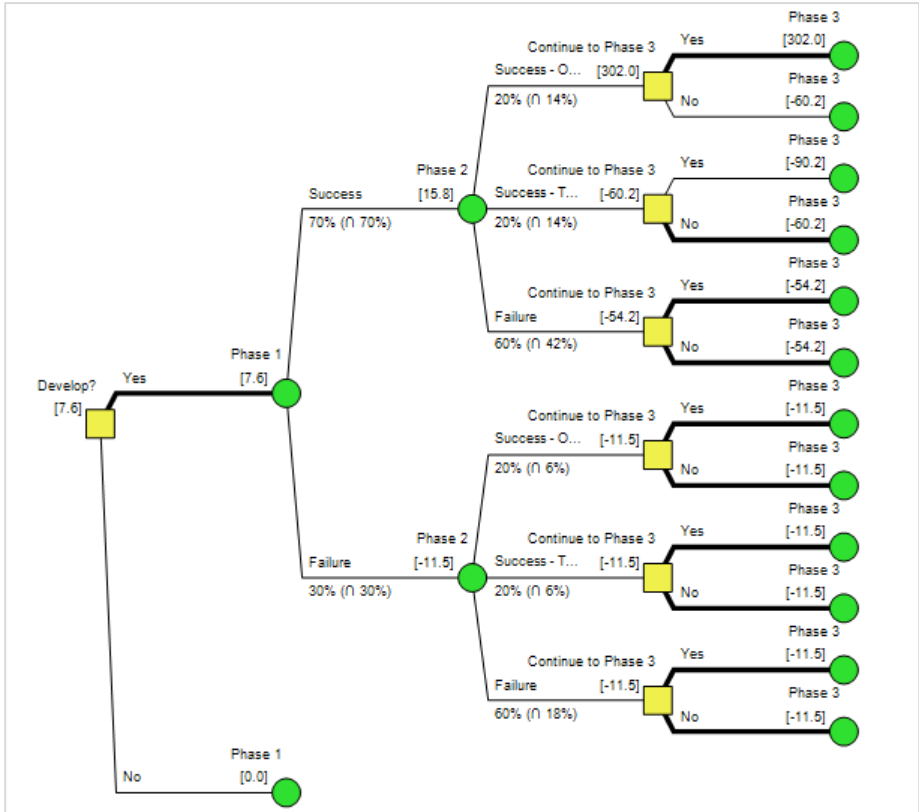


Figure 3-13. Policy Tree™ for Symmetric Model

## 3.2 Risk Profile

As shown in Figure 3-14, DPL creates one or more items in the Workspace Manager for each of the outputs it created. For Risk Profiles, DPL creates at least two items: one item for the Risk Profile chart and one item for each Risk Profile dataset in the chart. In the decision analysis run that you requested previously, DPL created a Risk Profile dataset and Risk Profile chart for the objective function under the optimal decision policy.

If you had requested Initial Decision Alternatives, DPL would have created two Risk Profile datasets: one for the Yes alternative of Develop? and one the No alternative. A third item (a Risk Profile chart) would have displayed both risk profiles within a chart for comparison. Note that the Risk Profile generated for the No alternative would have been uninteresting, as it is a

terminal alternative that would be displayed as a vertical line at 0. For more information on displaying and comparing multiple Risk Profiles within a single Risk Profile chart, see Section 6.6.

In the Workspace Manager, a Risk Profile chart is indicated by the icon (📊) and a Risk Profile dataset by the icon (📄). When you generate a Risk Profile for the optimal alternative policy, DPL names the Risk Profile chart and dataset "Expected Value". DPL encloses items in the Workspace Manager that do not correspond to windows (such as the Risk Profile datasets) with curly braces i.e., {Expected Value}.

- ⇒ Double-click on the item for Risk Profile chart called Expected Value (📊) in the Workspace Manager to activate the window. The Expected Value Risk Profile chart becomes active as shown in Figure 3-14.

Note: you can also double-click on the Risk Profile dataset called Expected Value. DPL activates the Risk Profile chart that displays it. If you double-click on a Risk Profile dataset that is not displayed in a Risk Profile chart, DPL creates a chart for the dataset and activates the chart.

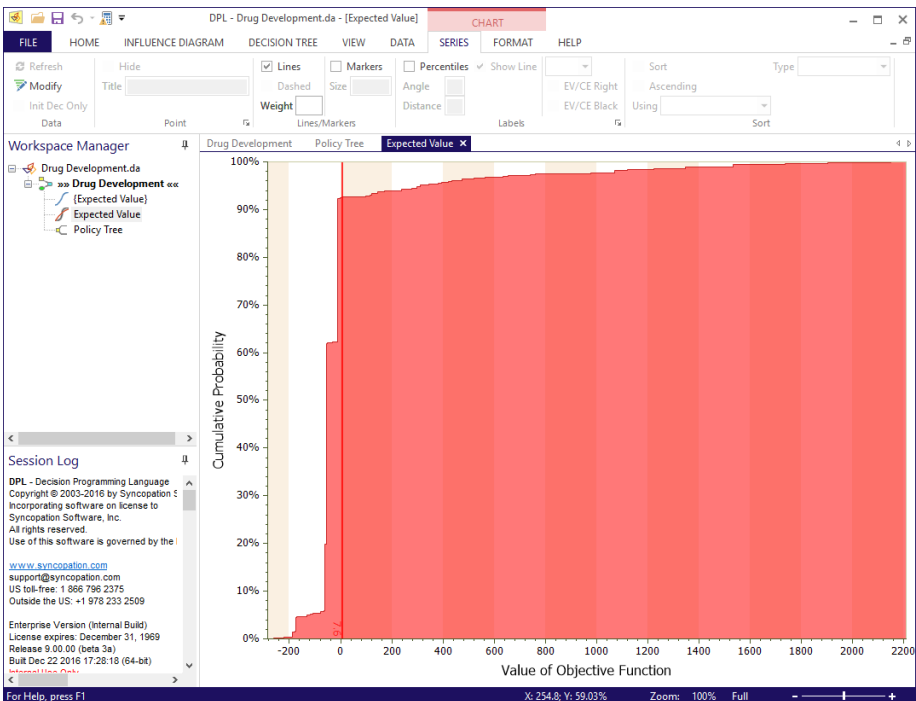


Figure 3-14. Risk Profile Chart for Optimal Decision Policy

By default, DPL creates a cumulative Risk Profile chart. A cumulative Risk Profile chart displays the value of the objective function (or attribute in multiple attribute models) on the x-axis (horizontal axis) and cumulative probability on the y-axis (vertical axis). A cumulative Risk Profile chart can be read by choosing a value on the x-axis, determining where the Risk Profile intersects with the vertical line "drawn" from the value, and then "drawing" a horizontal line over to the y-axis. The value where the horizontal line meets the y-axis is the probability that the objective function is less than or equal to the chosen value on the x-axis. For example, if you choose the value 300, you see that the vertical line drawn from 300 intersects the Risk Profile at a probability value of approximately 95%. Therefore, the probability that the objective function is 300 or less is 95%.

To create a Risk Profile chart, DPL calculates all the endpoint values (objective function values and probabilities) associated with the model, and then aggregates the endpoints into a user-determined number of intervals for display. The number of intervals used can significantly affect the appearance of the Risk Profile chart. The number of intervals has no impact on the expected value of the Risk Profile.

The precision of the percentiles from a Risk Profile chart is affected by the number of intervals. In the example in the previous paragraph, 300 is the 95<sup>th</sup> percentile. The 10<sup>th</sup> percentile is the x-axis value found by "drawing" a horizontal line from 10% on the y-axis to where it intersects the curve and then "drawing" a vertical line down from the intersection to the x-axis. If you are interested in percentiles and require more precision, you should set the number of intervals to be larger. A larger number of intervals requires more memory and has a slight performance impact.

The number of intervals can be changed within the Run Settings dialog (Home | Run | Settings).

- ⇒ Double-click on the model within the Workspace Manager to activate it (or press Ctrl+F12).
- ⇒ Click Home | Run | Settings to launch the Run Settings dialog. Under the *Risk Profile* section you should see that the *Number of intervals* is 500.
- ⇒ Click Cancel.

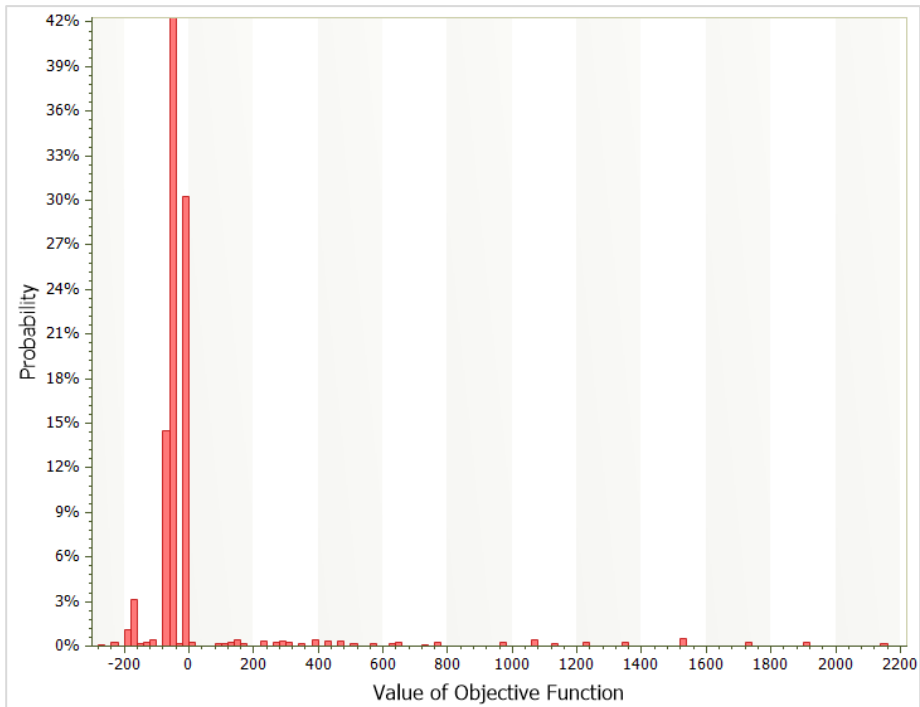
In general, 250 to 500 is usually a reasonable setting for most models. (Note: in the model as it currently stands there are only 177 paths in the tree – a small portion of which are in the optimal policy and displayed in the Risk Profile. Setting the number of intervals to greater than 500 will not affect how the Risk Profile charts looks since there are many fewer

scenarios than intervals.) If you have distributions with very long tails, you may wish to increase the number of intervals.

DPL displays the expected value (EV) of the objective function on the Risk Profile chart by default. Displaying the expected value of the distribution is particularly useful when comparing multiple Risk Profiles. It is also helpful to display the expected value if the Risk Profile is less symmetric because it may be difficult to estimate the expected value from the risk profile.

If you'd like to view your chart as a Histogram, do the following:

- ⇒ Activate the Risk Profile Chart and check the checkbox next to Histogram in the Format | Display group.
- ⇒ Uncheck the EV / CE Lines and EV / CE Values checkboxes in Format | Display. DPL updates the Risk Profile chart as shown in Figure 3-15.



**Figure 3-15. Frequency Histogram Type Risk Profile Chart**

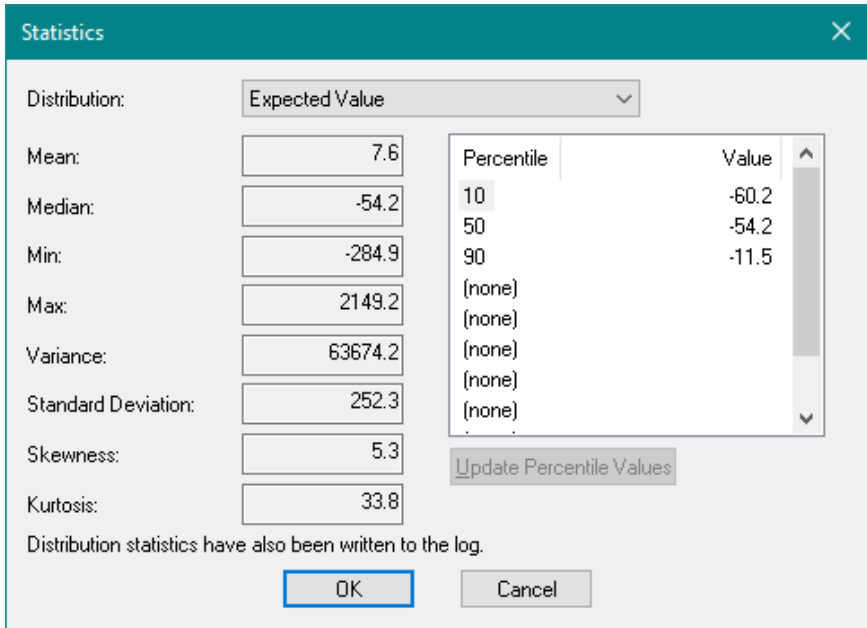
A frequency histogram displays the Risk Profile information in a different format. The x-axis still displays the objective function and the y-axis still displays probability values. The height of each bar in the frequency histogram Risk Profile chart indicates the probability that the objective

function falls within the range determined by the horizontal position and width of the bar. For example, in the frequency histogram Risk Profile chart in Figure 3-15 there is approximately a 30% chance that the objective function falls in the range -60 to 0 (the height of the second highest bar is 30% on the y-axis; it starts at -60 on the x-axis and ends at 0).

⇒ Revert back to a Cumulative risk profile chart type by un-checking the Histogram checkbox in Format | Display.

Now let's examine the statistics and percentiles for the distribution.

⇒ Click Data | Distributions | Statistics. You can also right-click on the Risk Profile and select Statistics from the context menu. The Statistics dialog appears as shown in Figure 3-16.



**Figure 3-16. Risk Profile Statistics**

The Statistics dialog gives you a standard statistical description of the distribution as selected by the drop down box at the top of the dialog. If the Risk Profile chart contained more than one Risk Profile dataset, you can use the drop down to select which distribution's statistic the dialog displays (See Section 6.6 for more on multiple Risk Profiles within a single chart). The dialog provides the minimum and maximum values, the variance and standard deviation, the higher moments (skewness and kurtosis), and the 10<sup>th</sup>, 50<sup>th</sup> and 90<sup>th</sup> percentiles of the distribution.

For example, the percentiles table shows that the 10<sup>th</sup> percentile of the distribution is -60.2; the objective function is less than or equal to -60.2 with probability 10%. Similarly, with probability 90%, the objective function is less than or equal to -11.5 (or conversely there is a 10% probability that the objective function exceeds -11.5). As you can see, this is a highly risky venture. You can use the table to determine the values for other percentiles, which you'll do now.

- ⇒ Double-click the first (none) item within the Percentile column to put it into edit mode and enter "95".
- ⇒ Click the Update Percentile Values button.

The 95<sup>th</sup> percentile of the distribution is 310.4.

To display the percentiles specified within the table on the chart:

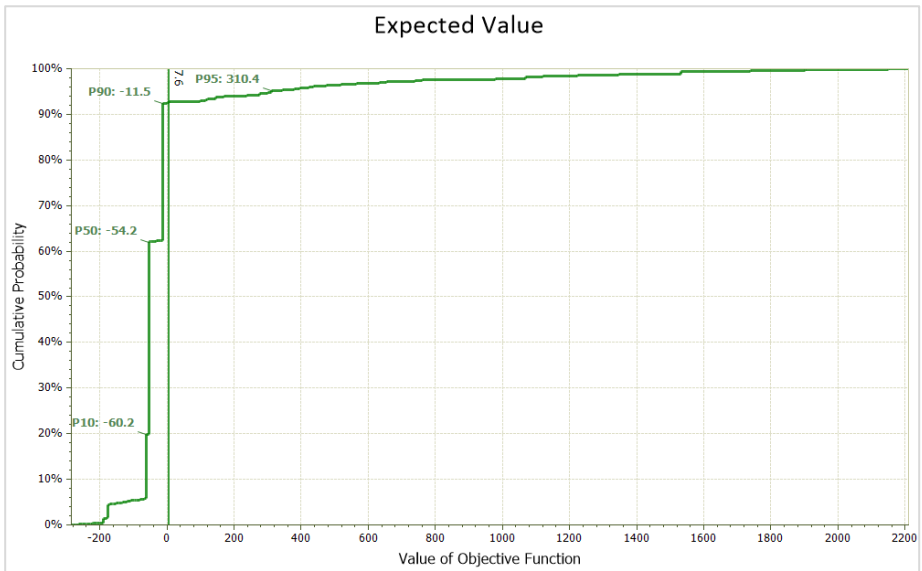
- ⇒ First, click OK to close the Statistics dialog.
- ⇒ To display the 10<sup>th</sup>, 50<sup>th</sup>, 90<sup>th</sup>, and 95<sup>th</sup> percentile values directly on the Risk Profile, check the Percentiles box within Series | Labels.

You can also view/update the Percentiles table by clicking Chart | Series | Labels | Settings.

A Risk Profile chart can be formatted in a number of ways. If you'd like, experiment with some of the other formatting options available on the Format tab. Figure 3-17 demonstrates some of the ways in which a chart can be formatted. If you'd like your chart to match, you should do the following:

- ⇒ Create a title for the chart by checking the checkbox next to Chart in the Format | Titles group. To rename the title, double-click the default title "Chart 1" and type in "Expected Value".
- ⇒ Select the Chart title text and change the font within View | Font to one of your choice. You can do the same with the values along both the x and y-axis and the axis label ("Cumulative Probability").
- ⇒ Within the Format | Display group uncheck Shading. This will cause the background shading of the chart to go away.
- ⇒ Uncheck the checkbox next to Color Fill within the Format | Color group. This will leave a single line with no color fill beneath it.
- ⇒ Select the line (it will turn magenta). Go to Chart | Series | Lines/Markers. In the Weight edit box, change the number from 2 to 3. This will thicken the weight of the line.

- ⇒ Go back to the Format | Display group. Check the checkboxes next to X Grid Lines, Y Grid Lines, and Dashed Grids to format the grid lines in the background of the chart.
- ⇒ Select the line within the chart and use the Color fill bucket split-button to select a new color for the line and percentiles text.
- ⇒ Turn on the EV/CE Lines and EV/CE Values. Select the EV line to edit the location and size of the EV Value via the Series | Labels group.
- ⇒ Select one of the percentiles and use the settings within the Chart | Series | Labels group to re-position percentiles, if needed.



**Figure 3-17. Formatted Risk Profile Chart**

See Section 5.2 for more on formatting charts.

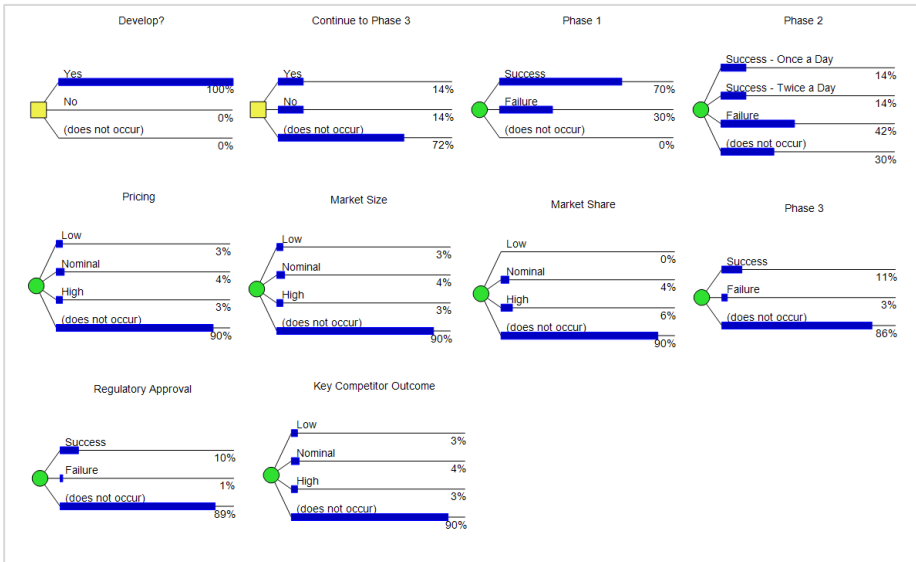
### 3.3 Policy Summary™

You will now examine DPL's Policy Summary. A Policy Summary is often used to understand the decision policy for a downstream decision (i.e., a decision that occurs after one or more chance nodes). You can request to have a Policy Summary generated as a part of a decision analysis run (check its box within the Home | Run group) or create one from the Policy

Tree output window (Policy | Policy Summary). Since there is already a Policy Tree in the workspace, you'll do the latter.

- ⇒ If it isn't already, activate the Policy Tree.
- ⇒ Click the Policy | Policy Summary | Add button to create and activate the Policy Summary output (Figure 3-18).

With downstream decisions there is not necessarily a single alternative that is always optimal. Note that in the Policy Tree a different alternative is optimal for the Continue to Phase 3 decision depending upon the outcome of Phase 2. The Policy Summary provides a useful view of the information in the Policy Tree, particularly for models with larger and/or more complex Policy Trees.



**Figure 3-18. Policy Summary™**

The Policy Summary displays summary information about the optimal policy. For each node, it displays the policy dependent probabilities for each state of the node at the end of the state branch. A policy dependent probability is the probability weighted fraction of optimal scenarios in which the state (either decision alternative or chance outcome) occurs.

You will note that DPL also displays an additional state for each node called "(does not occur)" and displays the probability of this happening as well. The "(does not occur)" state will have a non-zero policy dependent probability if there are scenarios in which the node does not occur. This



common for asymmetric trees such as the one you've created in this example.

For initial decisions (those that appear before any chance nodes in the Decision Tree), the policy dependent probability of the optimal branch is always equal to one (or 100%) and is always zero (0%) for the rest. So in the Policy Summary, the optimal decision alternative for the initial decision is the one with 100% probability. As you know, Yes is the optimal alternative for the Develop? Decision and Figure 3-18 displays the Yes alternative with a 100% policy dependent probability.

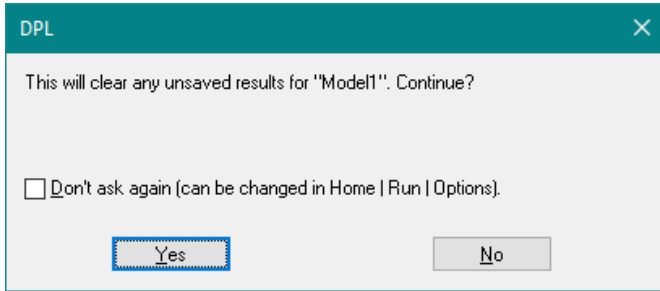
For downstream decisions, the Policy Summary displays the probability weighted fraction of scenarios for which each downstream decision alternative is optimal. Continue to Phase 3 is a downstream decision because it occurs after a discrete chance event. As indicated in Figure 3-18, the Yes alternative for Continue to Phase 3 has a policy dependent probability of 14% while No has a policy dependent probability of 14% as well. In 72% of scenarios, this decision event doesn't occur.

The Policy Summary also displays the policy dependent probabilities of the outcomes for discrete chance nodes. For discrete chance nodes that are not conditioned by others and that do not depend on the optimal policy, the policy dependent probabilities are equal to the input probabilities. In the current model, this is only true of Phase 1 event. You can see that the policy dependent probabilities for the remaining uncertainties do not match the input probabilities for the reasons described above.

## 3.4 Saving Outputs within a Workspace

---

All outputs described in Chapter 3 up to this point (Risk Profile Charts and Datasets, Policy Trees, Policy Summary) along with some that will be described later in the guide (Value of Information/Control, Value Correlations, and Endpoint Databases) will remain in the Workspace until the next run of the model. When you have one or more of these outputs associated with your model and you run a Decision Analysis, you will be given a warning that any unsaved results from the previous run will be cleared or overwritten by the new run. See Figure 3-19.



**Figure 3-19. Unsaved Results Warning**

An unsaved output or dataset will be cleared, even if you do not select the output to be created in the subsequent run. For example, if you have an unsaved Risk Profile associated with your model and run a Decision Analysis in which you do not request a Risk Profile, the Risk Profile from the previous run is still deleted.

To save outputs from a particular run so that they will not be overwritten in subsequent runs, you must re-name the output in the Workspace Manager. Note that there is no limit to the number of saved sets of results you can have in DPL workspace.

When you rename a Risk Profile dataset the corresponding Risk Profile chart will also be renamed, and therefore, saved as well.

Outputs generated from a sensitivity analyses (tornado diagrams, rainbow diagrams, Time Series Percentiles, and Option Value charts) and Policy Summaries that are generated from the Policy tab are not overwritten in subsequent Decision Analysis runs or analyses. They will remain in the workspace until you explicitly delete them.

When you save a DPL workspace (File | Save) you save all outputs (renamed or otherwise) included in your workspace, along with any formatting done.

Further if you format certain outputs such as a Risk Profile chart but do not rename it, DPL will not delete the chart but will re-use it for the next run if a Risk Profile is requested. This allows you to make formatting changes once, make edits to the data in the model or structure of the model and re-run to produce a new chart with the same formatting. This may be convenient if you need to do a series of runs for a presentation. See Section 3.7 for information on copying outputs to presentations.

## 3.5 Expected Value of Perfect Information / Control

---

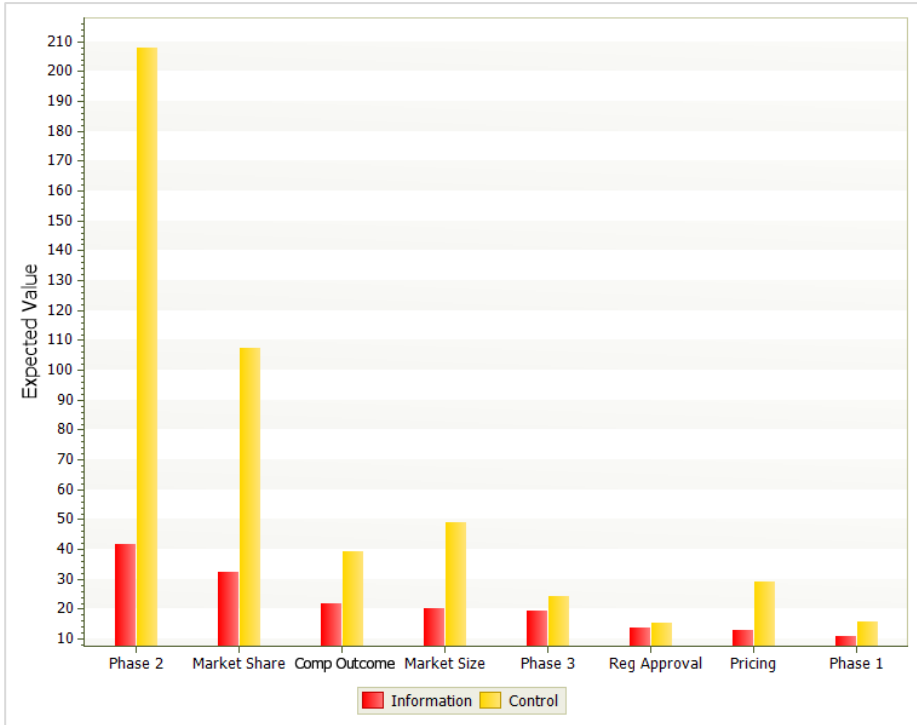
The expected value of perfect information / control is actually two outputs which both indicate how important it is to reduce the uncertainty associated with each discrete chance node in a model. The expected value of perfect information indicates how valuable it would be to have perfect information for a chance node before making a decision. The expected value of control indicates how valuable it would be to have complete control over the uncertainty.

- ⇒ In the Home | Run group, make sure the following are checked: Risk Profile, Policy Tree, VOIC and Value Correlations.
- ⇒ Click Home | Run | Decision Analysis or press F10. Click Yes to the overwrite prompt. DPL clears the model of all Policy Tree and Risk Profile outputs (the Policy Summary is maintained), produces the requested outputs, and activates the Policy Tree.

Note if you formatted the Risk Profile chart as described in Section 3.2 then as discussed at the end of Section 3.4 the Risk Profile chart will be re-used in this run preserving the formats you made.

- ⇒ Double-click the Value of Info/Control item in the Workspace Manager to activate the output.

You will now examine the expected value of perfect information / control results.



**Figure 3-20. Expected Value of Perfect Information / Control**

DPL displays the value of perfect information for each chance node as a red vertical bar and the value of control as a yellow vertical bar. By default, both sets of bars are displayed and the nodes are sorted in descending order by their value of perfect information.

DPL calculates the expected value of perfect information for each chance node, in essence, by moving it to the front of the Decision Tree (or as far to the front as it can) while keeping all other nodes fixed in position and "re-running" the model. (Note that DPL's solution algorithms allow it to calculate the expected value of perfect information without actually having to re-run the model.) DPL calculates the expected value of the objective function for the modified model.

The expected value of perfect information for a chance node is the difference in the expected value of the objective function of the modified model and the expected value of the objective function of the base model.

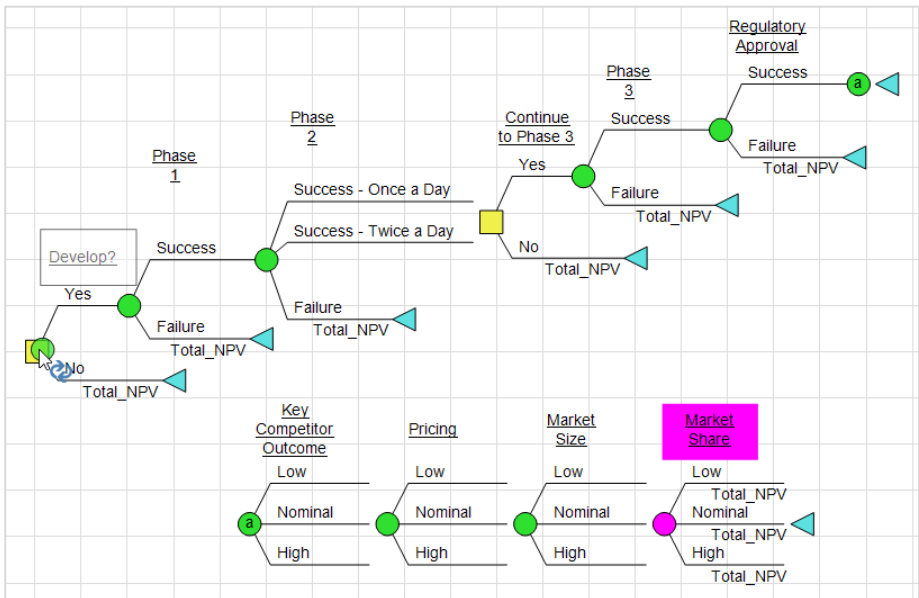
It is called the expected value of perfect information because the modified model simulates the situation in which you have perfect information about the uncertainty before you need to make a decision. For example, in the

base model you must make the decision to develop the drug before you know what market share will be. The difference in value between the modified model and the base model is due to the information gained.

For a given chance node, if the value of perfect information is greater than zero, then the optimal alternative must be different given the outcome of the chance node in the reordered tree (otherwise the expected value of the objective function of the model with the reordered tree would be the same).

You will modify the model and generate a new Policy Tree for comparison to the original Policy Tree in order to see how DPL calculates the value of perfection information for Market Share. To do so:

- ⇒ Press Ctrl+F12 to activate the model.
- ⇒ Select and drag the Market Share node to the head of the Decision Tree. Notice when you hover the node cursor over another node, a re-order icon appears under your cursor as shown in Figure 3-21.



**Figure 3-21. Dragging Market Share node to Re-order Decision Tree**

- ⇒ Place the Market Share node on top of the Develop? decision node. It should now be at the head of the tree.
- ⇒ Select the branches of the Market Share node and Ctrl+X to cut the get/pay expression from the branches.

⇒ Select the branches of the last node in the tree (Market Size), and press Ctrl+V to paste the get/pay expression to its branches.

You will save some of the outputs generated for future use. Most importantly you'll rename and save the original Policy Tree so you can compare it to the Policy Tree generated for the modified tree.

To rename items in the Workspace Manager you can right-click the item and choose Rename from the context menu or select the item and press F2.

⇒ Rename the outputs as follows:

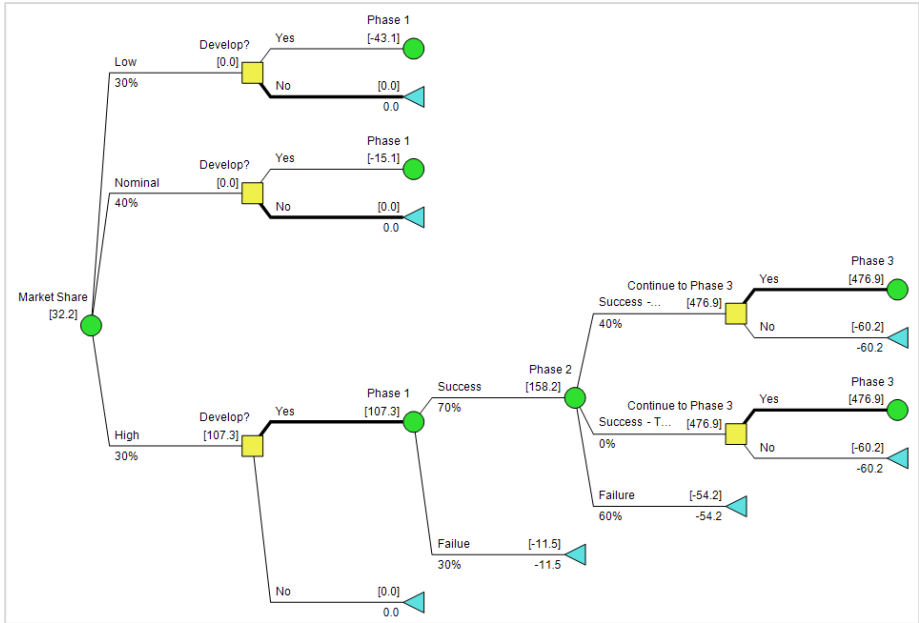
Original Output Name	Renamed Output
Value Correlations	Saved Value Correlations
Value of Info/Control	Saved VOIC
Policy Tree	Original Policy Tree

⇒ Leave the names for the Risk Profile dataset and chart as is. You will allow these to be deleted with the next run as they aren't needed.

⇒ In the Home | Run group make sure Policy Tree is checked, all other outputs should be un-checked.

⇒ Run a Decision Analysis and say Yes to the overwrite prompt.

First off, notice that the Risk Profile dataset and chart have been deleted – as you did not rename, and therefore, did not save them. Even though you may have formatted the Risk Profile chart and it got re-used in the previous run, the chart is deleted in this run since you did not ask for a Risk Profile this time. The three outputs that were re-named have been maintained through the run, along with the Policy Summary which was created from the Policy tab and not as a part of an analysis. There are now two Policy Tree outputs associated with the model ("Original Policy Tree" and "Policy Tree"). The "Policy Tree" for the modified model is active as shown in Figure 3-22.



**Figure 3-22. Policy Tree to illustrate Value of Perfect Information**

The expected value for the modified model is 32.2. Recall (or look back at the original Policy Tree) that the EV for the base model is 7.6. Therefore, the value of perfect information for Market Share is  $32.2 - 7.6 = 24.6$  (which is the height of the Information bar for Market Share in the VOIC chart). This indicates that the optimal alternative must have changed for one of the outcomes of Market Share. If you had the luxury of knowing which state Market Share was in before deciding to develop the drug, you would choose not to proceed in certain scenarios. As shown in the Policy Tree above the optimal decision alternative is not to develop when Market Share is Low or Nominal.

### 3.6 Value Correlations

In this section you will examine a simple example of a Value Correlations chart. Value Correlation charts are bar charts that display the correlation coefficients between each value and event in the model and the objective function.

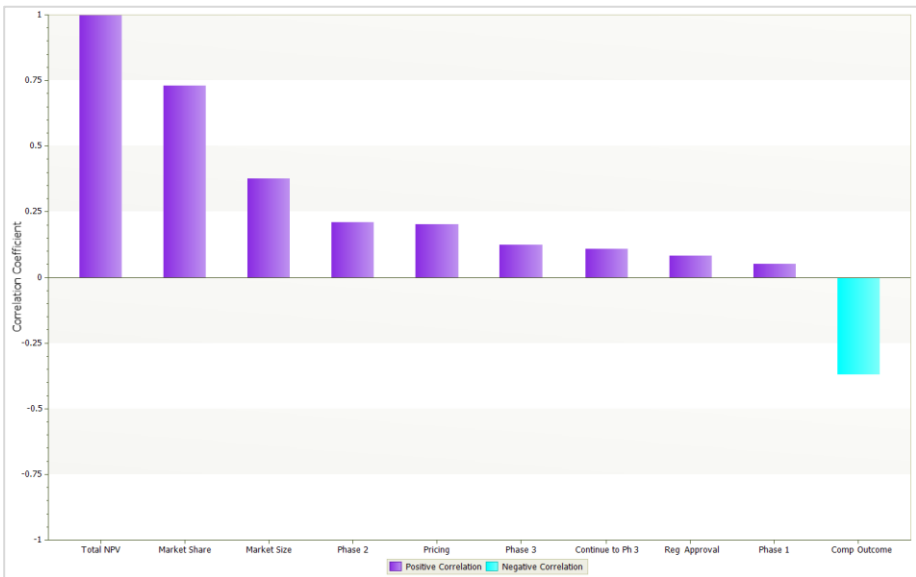
Correlation coefficients are measures of statistical association between two variables. Correlation coefficients can vary between 1.0 (perfect positive

correlation) and -1.0 (perfect negative correlation). Perfect positive correlation (1.0) means that as one of the two variables increases, the other variable always increases proportionately. Perfect negative correlation (-1.0) means that as one variable increases, the other variable always decreases proportionately.

Correlation does not always imply causation; that is, even if two variables are correlated, it does not necessarily mean that an increase or decrease in one variable is causing the change in the value of the other variable. Lack of causation may be due to, for example, both variables being influenced by some other variable which is not in the model.

Note when you request value correlations DPL must evaluate the model using full tree enumeration and does this automatically. You may have noticed previously that this changes the default action of the Home | Run | Decision Analysis button. For this simple example, it is unlikely you will notice the difference. For larger models, you may wish to return to fast sequence evaluation after running value correlations. Use the Home | Run | Decision Analysis drop-down list to do this the next time you run the model.

⇒ Double-click on the Saved Value Correlations item to activate the window. The Value Correlations chart is displayed as in Figure 3-23.



**Figure 3-23. Value Correlations Chart**



The Value Correlations chart assumes that the optimal decision policy is followed. It tells you which values in your model tend to vary in the same direction (positive correlation) or in the opposite direction (negative correlation) as the objective function, which in this case is NPV. Positive correlations are shown as purple bars, while negative correlations are aqua. A correlation (bar) is shown for each value in the model for which the correlation coefficient can be calculated. Each value corresponds to either a decision node, a chance node, or a value node in the model. If a chance node or decision node has no values associated with it, DPL correlates the state numbers with the objective function. E.g., for a decision node with two alternatives, the first alternative has a state number of 1 and the second alternative has a state number of 2.

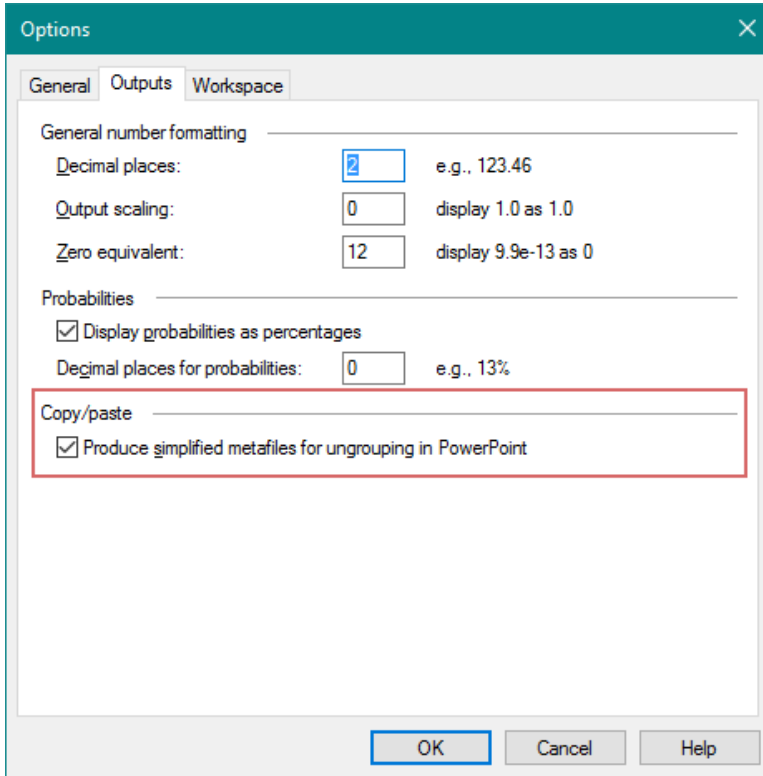
As Figure 3-23 shows, only one of the chance events is highly correlated with the objective function. The first bar (Total\_NPV) simply indicates that the objective function is perfectly correlated with itself. Market Share is highly positively correlated with the objective function. This makes sense because the higher the market share, the higher the revenues and NPV are. On the other hand, Key Competitor Outcome is negatively correlated with the objective function (Total\_NPV).

In decision analyses with complex value models, Value Correlations charts are useful for understanding how the variability in the objective function is related to the variability in individual values such as chance nodes and downstream decisions.

## 3.7 Copying/Pasting to a Presentation

---

All of the output charts in DPL can be copied and pasted to other applications, e.g., presentation software or word processing software such as Microsoft PowerPoint or Word. In addition, you can copy and paste DPL models. When you copy an image, it is copied onto the clipboard both in metafile format and bitmap format. Depending on whether and to what extent you intend to edit the copied image, one format might be preferable to another. In particular, if you wish to edit the chart image in PowerPoint other than just resizing it subsequent to pasting it, you should turn on the *Produce simplified metafiles for ungrouping in Powerpoint* setting within File | Options | Outputs (Figure 3-24). If you are using an application other than PowerPoint, you may need to experiment with the Paste options to find the best one.



**Figure 3-24. Produce Simplified Metafile Setting within File Options dialog**

If you will not be editing the image itself (other than resizing it) subsequent to pasting it, paste it as a bitmap. The image will look exactly as it does on screen. If you maintain the aspect ratio (height to width) when you resize, you can resize the image and still maintain how it looks from when you copied it. If you intend to make more substantial edits to the image, paste it as a metafile. In Powerpoint, you can then un-group it and make various edits. Note: the simplified metafile setting above results in better looking images when they are un-grouped in Powerpoint.

- ⇒ Open a presentation or word processing software application that you use (e.g., Microsoft Powerpoint). Use a blank document where you can paste an example chart.
- ⇒ Navigate to File | Options | Outputs and check the box for Produce simplified metafiles for ungrouping in PowerPoint as shown in Figure 3-24.

- ⇒ Activate the VOIC chart you examined earlier.
- ⇒ Click Home | Edit | Copy (or Press Ctrl+C) to copy the chart to the clipboard.
- ⇒ Switch to the document you opened. Use Edit | Paste (this pastes as a bitmap by default) or Edit | Paste Special to paste the chart.
- ⇒ If you're using PowerPoint or Word and have copy/pasted the image as a metafile, you can right-click on the image and select Group | Ungroup from the context menu to facilitate edits.
- ⇒ Close the application (save the new document with the chart image only if you want to; you don't need it later in this guide.)

For this particular bar chart you may not see a difference between the bitmap and simplified metafile image once ungrouped in PowerPoint. But it should be noted that some charts (e.g., Risk Profiles) will need to be in simplified metafile format for PowerPoint to ungroup the image properly for editing.

The above steps can be used for copying an image of your DPL model or any of DPL's output charts to other applications. Note: because the chart is copied as a picture, the underlying data is not copied to the other application.

## 4. Building a Model in Influence Diagram-focused Mode

This tutorial focuses on building a spreadsheet-linked model in Influence Diagram-focused mode for a simple decision analysis application. The main purpose of an Influence Diagram is to show the relationships among the different factors relevant to a decision. DPL's Influence Diagram modeling environment provides a rich set of features to give you flexibility in defining these relationships.

In this chapter, you'll familiarize yourself with key Influence Diagram modeling features like adding Excel-linked value nodes, running sensitivity analyses, introducing uncertainty and conditioning, and generating results. If you are new to DPL and would like to familiarize yourself with the modeling features of the Influence Diagram, this tutorial is a good place to start. If you would prefer to model in Decision Tree-focused mode, you should go through the tutorial contained in Chapters 2 and 3.

A DPL model is always a combination of an Influence Diagram and Decision Tree. When modeling in Influence Diagram-focused mode, DPL continues to update the default Decision Tree behind-the-scenes. You can press the Tab key or drag the splitter to the middle of the model window to view the Decision Tree. Keep in mind that once you've made edits to the default tree, you've taken over the controls and DPL will no longer make edits to the tree as you model in the Influence Diagram.

### 4.1 Value Nodes and Spreadsheet Links

---

You will develop a decision model by adding spreadsheet-linked value nodes to the Influence Diagram. Each of these nodes will be linked to a range in the Excel spreadsheet called *New Product Strategy.xlsx*, which is an example file installed with the software.

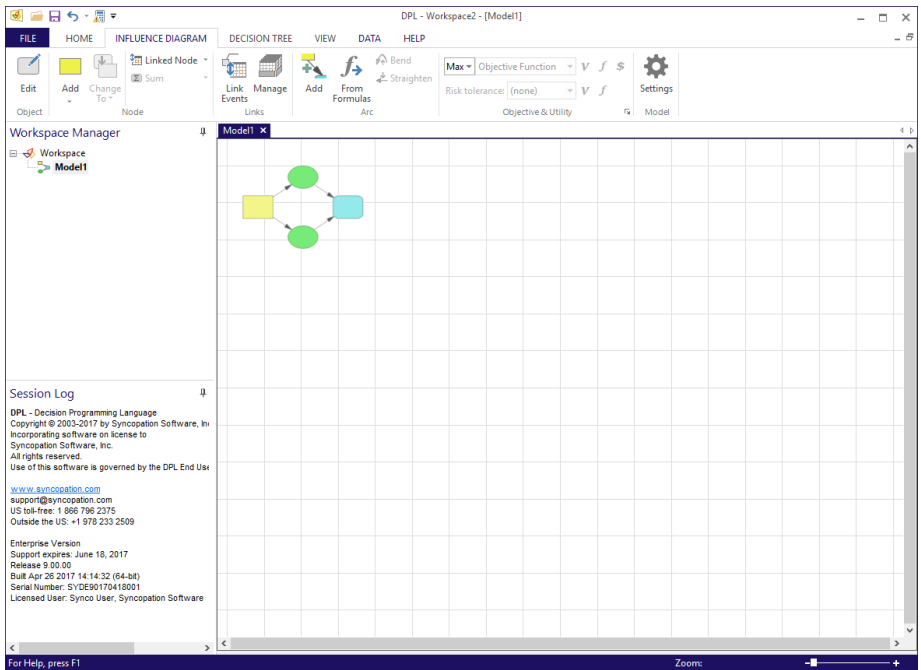
A value node is a constant (i.e., a number) or calculated quantity (i.e., a formula) in a DPL decision model. They can be linked to a cell in a spreadsheet or be local (i.e., not linked and contain a number or formula in DPL). Value nodes are represented by rounded blue rectangles in the Influence Diagram.

⇒ Open DPL.

By default, DPL will start with an empty Decision Tree pane maximized on the right-hand side of the screen. You'll be building the model in Influence Diagram-focused mode so you'll switch to Influence Diagram-focused mode.

⇒ Press Tab to switch to the Influence Diagram pane of the model window. See Figure 4-1.

The Influence Diagram pane will now be maximized within the Model Window and the Influence Diagram ribbon tab active.




**Figure 4-1. New Workspace with Decision Tree pane maximized**

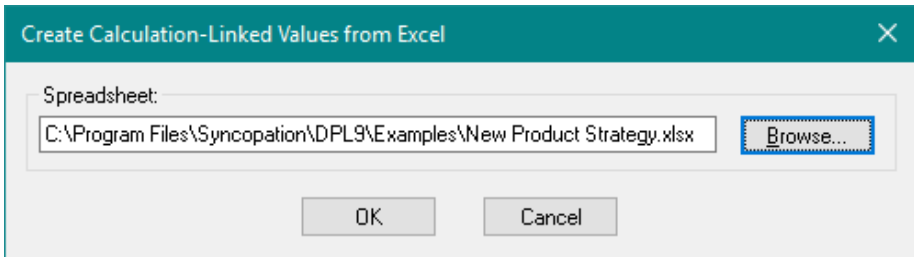
If you wish to have all new models start with the Influence Diagram pane active and maximized, do the following:

- ⇒ Click File | Options. The General tab of the File Options dialog is displayed.
- ⇒ Select the Influence Diagram radio button for the *For new models, maximize pane* setting. Click OK to close the dialog.

You're going to start building out the model by adding linked value nodes to the Influence Diagram. To do so:

- ⇒ Click the Influence Diagram | Node | Linked Node split button. Note initially the default action of this button should be to add Excel Calculation-Linked nodes (  ) which is what you want. If it isn't, drop down the list and select it.
- ⇒ In the Create Calculation Linked Values from Excel dialog (Figure 4-2), click Browse and select the file New Product Strategy.xlsx. The spreadsheet is located within the Examples folder of the DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples.

The directory may vary slightly depending on your license type and operating system.



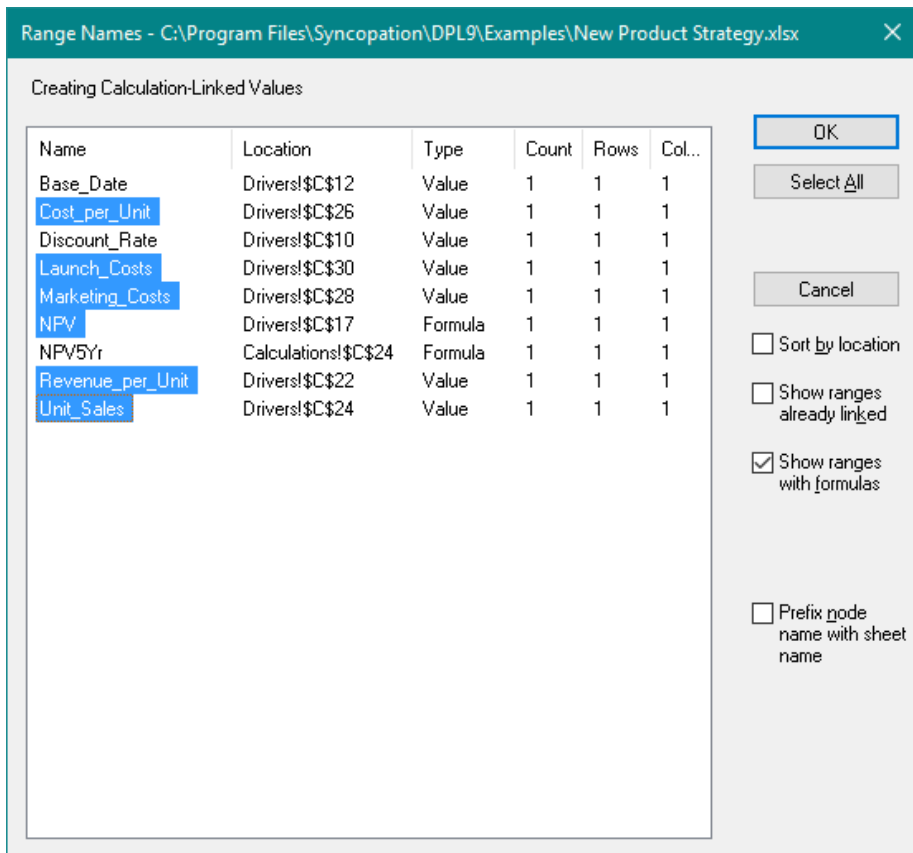
**Figure 4-2. Create Linked Values Dialog**

- ⇒ Click OK. The Range Names dialog appears.

The Range Names dialog displays all named ranges in the spreadsheet that are suitable for linking to value nodes.

Important note: DPL only displays named ranges or cells in the Range Names dialog. Before linking a spreadsheet to DPL, you should ensure that ranges you wish to link are named in Excel. Please consult your Excel documentation if you are not familiar with named cells/ranges, as they are essential for using DPL with Excel.

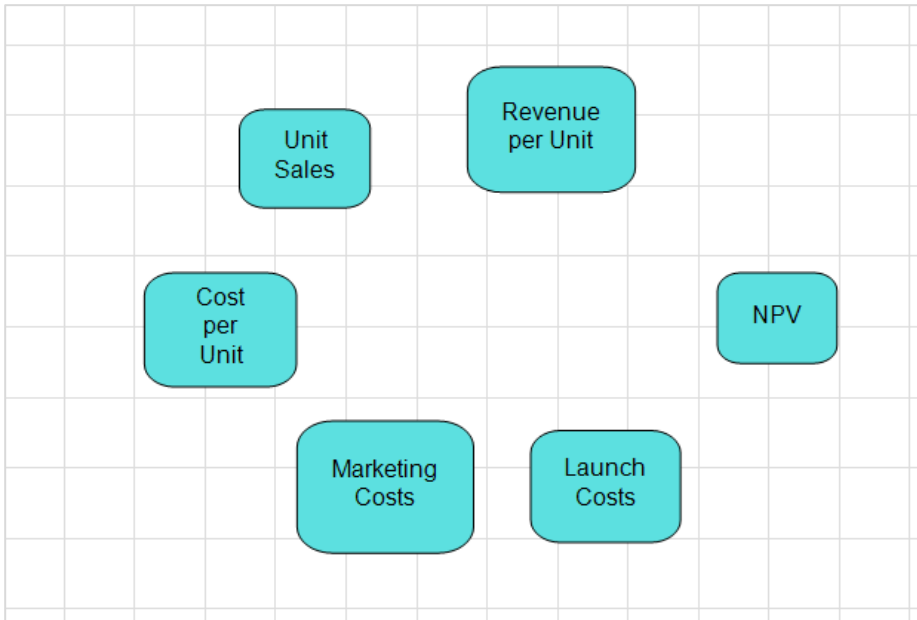
- ⇒ Select the ranges named Cost\_per\_Unit, Launch\_Costs, Marketing\_Costs, NPV, Revenue\_per\_Unit and Unit\_Sales as shown in Figure 4-3. (Hint: to select multiple names, hold down the Ctrl key as you click the names.)



**Figure 4-3. Range Names Dialog with Selection**

- ⇒ Click OK. DPL creates the linked value nodes and adds them to the Influence Diagram.

⇒ Arrange the nodes so they look like Figure 4-4.

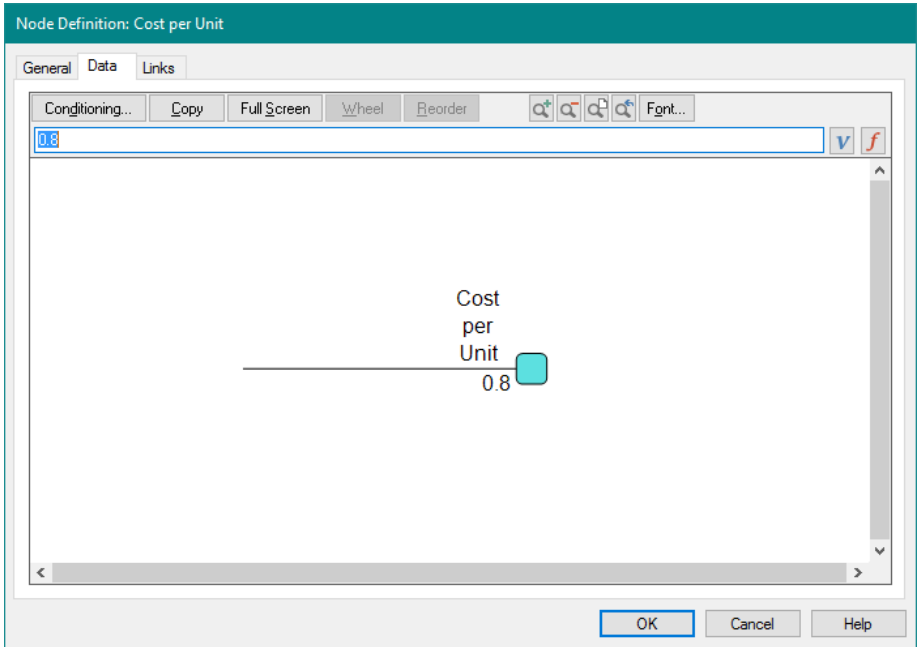


**Figure 4-4. Model with Six Value Nodes Arranged**

When you build a spreadsheet-linked DPL model, you will typically have DPL send different sets of drivers to Excel and then ask Excel to send back one or more metrics. Nodes in DPL that send data to Excel are called driver nodes. Nodes that receive data back from Excel are called metric nodes. Driver nodes are typically linked to ranges in Excel that contain constant numbers (not formulas). While metric nodes are typically linked to ranges in Excel that contain formulas. In the current model, NPV is a metric node and all the rest are driver nodes.

⇒ Double-click on the Cost per Unit value node. The Data tab of the Node Definition dialog is displayed. See Figure 4-5.





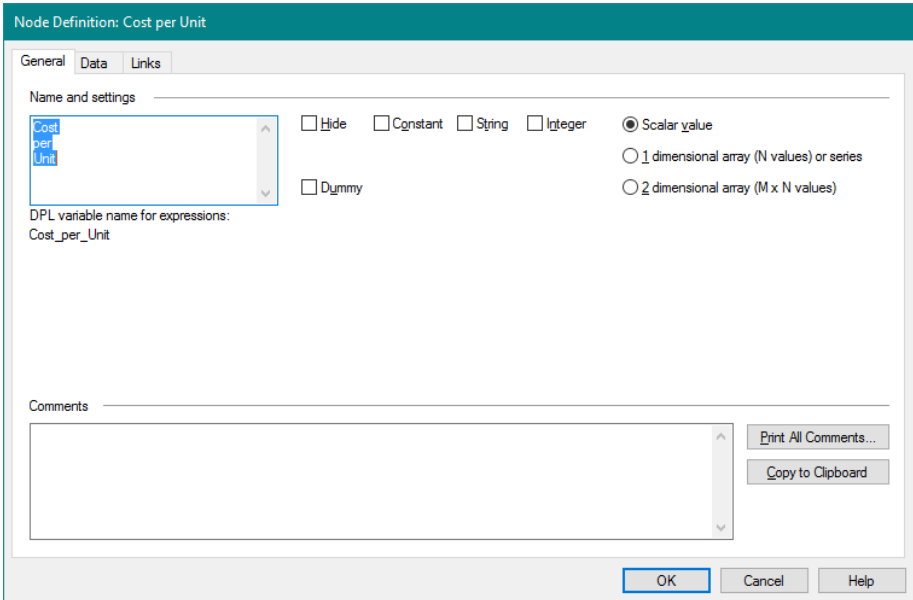
**Figure 4-5. Data Tab of the Node Definition Dialog for Cost per Unit**

The Data tab displays a data input tree for the value node and allows you to enter the data for this node. Because this node was created as a linked value, DPL has filled in the value of 0.8, which comes from the cell named Cost\_per\_Unit in the spreadsheet.

⇒ Switch to the General tab of the Node Definition dialog. See Figure 4-6.

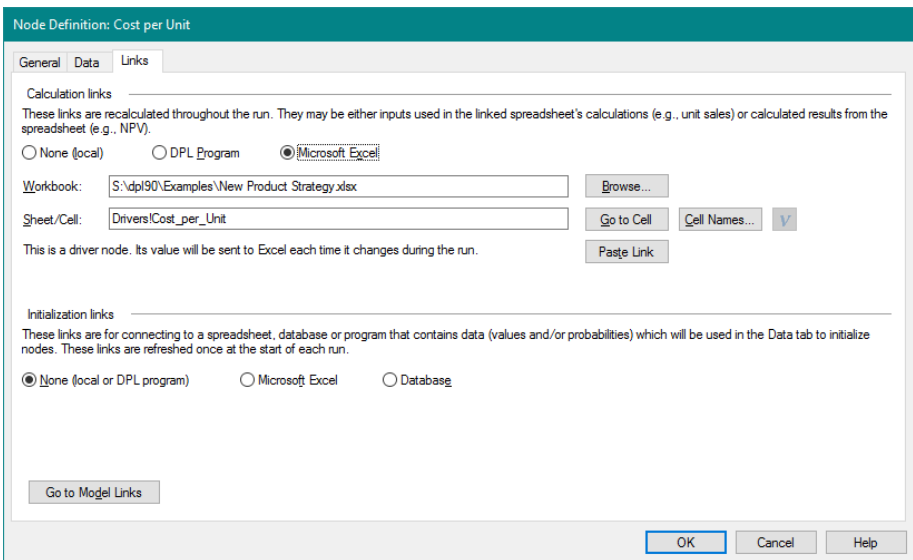
To the right of the *Name and settings* section are several options specific to value nodes. For instance, via the checkboxes you can choose to hide the value node or specify that it is a dummy node (i.e., a node that doesn't include data and is ignored by DPL during an analysis), which can be helpful in indicating calculations more clearly. You can also specify that the value is a constant, string, or integer.

Further to the right are a set of radio buttons that allows you to specify the dimensionality of the value node. Value nodes can be a single value (i.e., a scalar) like Cost per Unit is currently, a one dimensional array, a series, or a two-dimensional array. If you specify anything other than a Scalar value, the number of rows, columns or both should be specified on the Data tab.



**Figure 4-6. General Tab of the Node Definition Dialog for Cost per Unit**

⇒ Switch to the Links tab of the Node Definition dialog. See Figure 4-7.



**Figure 4-7. Links Tab of Node Definition dialog for Cost per Unit**

The Links tab contains information about the links between the node and Excel. There are two categories of spreadsheet links: calculation links and initialization links. Calculation links are used to send or receive data from Excel during the course of a run. Initialization links are used to set the values (or probabilities) for a node at the beginning of each run. Initialization links are discussed in Appendix A.3.

The radio buttons near the top of the Calculation links section of the dialog indicate whether the node is linked and if so, to what (Excel or a DPL Program). The workbook as well as the sheet and cell that the node is linked to are displayed in edit boxes. The sheet and cell syntax is the same as Excel's syntax (i.e., Sheet1!Cell\_name).

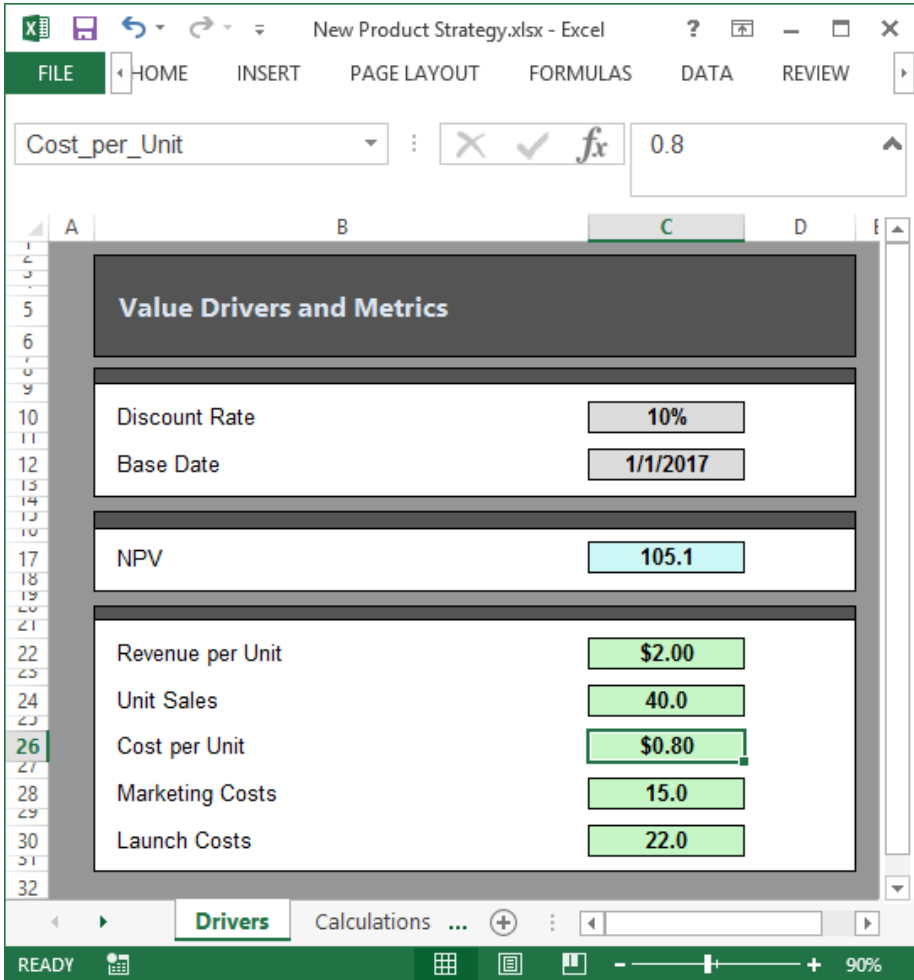
Table 4-1 summarizes the functions of the buttons in the Calculation links section.

<b><u>Button</u></b>	<b><u>Action</u></b>
Browse...	Finds a spreadsheet or allows you to change to which spreadsheet this node is linked.
Range Names...	Brings up the Range Names dialog which can be used to change the sheet/cell to which this node is linked.
Go to Range	Takes you to the cell/range in Excel to which this node is linked.
Paste Link	Pastes a link to a cell from Excel. The cell must be copied in Excel first.

**Table 4-1. Calculation Links Buttons**

⇒ Click the Go to Range button.

DPL activates Excel and selects the sheet and range to which the node is linked. As you can see in Figure 4-8, the range named Cost\_per\_Unit contains the value \$0.80, which is what DPL copied for the node's data.



**Figure 4-8. Excel with Linked Range Selected**

- ⇒ Switch back to DPL.
- ⇒ Click OK to close the Node Definition dialog without making any changes.

At this point, you may want to name your DPL model and save your Workspace file.

- ⇒ Select the item for the model in the Workspace Manager (DPL gave it the default name of "Model1").
- ⇒ Press F2 to edit the name.



- ⇒ Type "License vs. In-house" or any other name you'd like.
- ⇒ Press Enter to save the edit.
- ⇒ Save the Workspace and give it a name of your choice using File | Save.

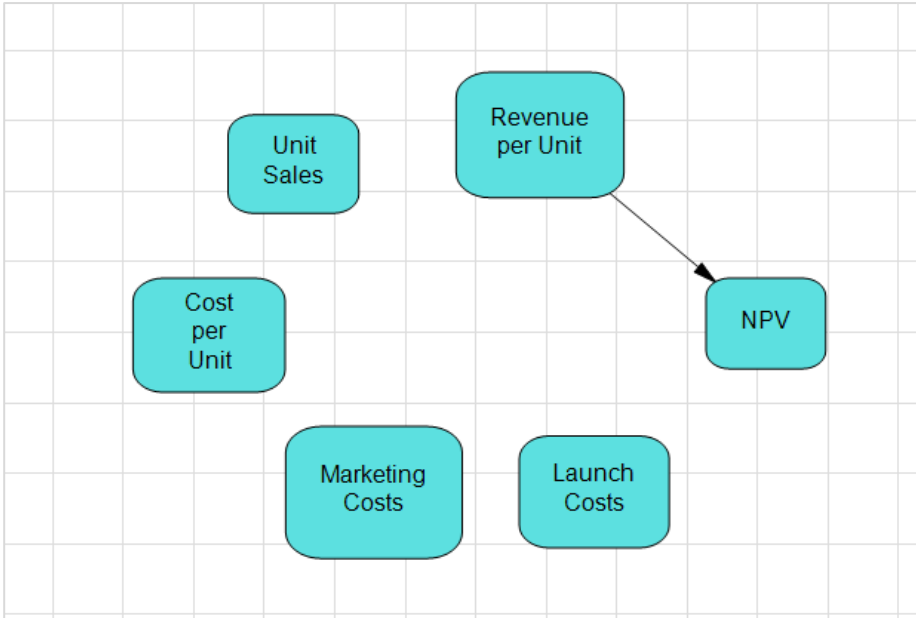
## 4.2 Influence Arcs

---

You will now add some influence arcs to the model. Influence arcs indicate one or more of three things to DPL: 1) dependency 2) conditioning and/or 3) timing. (Note: see the Glossary of DPL and DA Terms for definitions of dependency and conditioning.)

Influence arcs are useful because they visually indicate which nodes depend on or are conditioned by other nodes. You will start by creating an influence arc between Revenue per Unit and NPV.

- ⇒ Click in whitespace or hit Esc to deselect Cost per Unit.
- ⇒ To create an influence arc, click the Influence Diagram | Influence/Arc | Add button on the ribbon (see Table 1-3). The mouse cursor changes to the begin arc cursor (  ).
- ⇒ Place the cursor over the Revenue per Unit node and click. Now the cursor changes to the end arc cursor (  ) and an arc is anchored at the Revenue per Unit node. Revenue per Unit is called the predecessor node for this arc.
- ⇒ Place the cursor over the NPV node and click. An influence arc is drawn from Revenue per Unit to NPV. Click in whitespace to deselect the nodes and arc. See Figure 4-9. NPV is called the successor node for this arc.



**Figure 4-9. Model with Influence Arc**

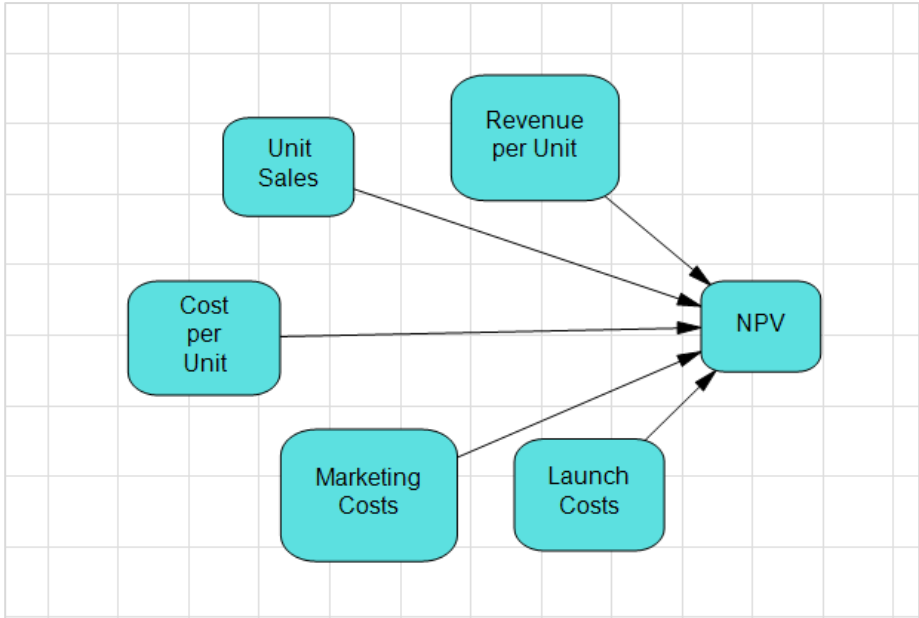
An influence arc that points from node A to node B indicates that node B depends on, is conditioned by and/or occurs after node A. In Figure 4-9, the influence arc you just added indicates that NPV depends on Revenue per Unit.

Note: DPL indicates which arc is selected by coloring it magenta. In general, DPL indicates selected items in the Influence Diagram or Decision Tree by coloring them magenta.

There is an easy way to have DPL automatically generate influence arcs based on your formulas or spreadsheet links.

⇒ Click Influence Diagram | Influence/Arc | From Formulas on the ribbon (see Table 1-3).

DPL generates the influence arcs from the formulas found in the node data. Your model should now have all the arcs shown in Figure 4-10.



**Figure 4-10. Model with All Influence Arcs**

Note: the above process provided you with some familiarity on how to create individual Excel-linked nodes and how to create individual influence arcs. This information is useful when building and modifying DPL models. However, DPL also provides a command which will create the model shown in Figure 4-10 in a single step. Before continuing, you will try this now.

- ⇒ Select Home | Workspace | Add to WS | Excel Linked Model.
- ⇒ In the Create Model from Excel dialog, click Browse and select the file New Product Strategy.xlsx. It is in the folder with DPL examples, usually C:\Program Files\Syncopation\DPL9\Examples.
- ⇒ Within the Range Names dialog select the same ranges as shown Figure 4-3 (Cost\_per\_Unit, Launch\_Costs, Marketing\_Costs, NPV, Revenue\_per\_Unit, and Unit\_Sales).
- ⇒ Click OK.
- ⇒ You will receive a prompt that linked nodes have been created in the Influence Diagram. Click OK.
- ⇒ Press Tab to activate the Influence Diagram.

DPL has created a new model in the Workspace called New Product Strategy (i.e., the same name as the spreadsheet). Other than node arrangement this model is identical to the model in Figure 4-10. This is a much quicker way to build a model from an Excel spreadsheet. However, it is also important to know the steps outlined in Sections 4.1 and 4.2.

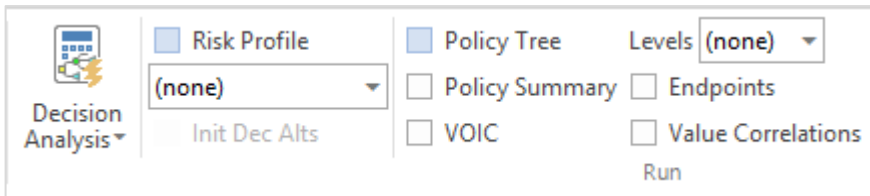
- ⇒ Select the New Product Strategy model in the Workspace Manager.
- ⇒ Press Delete.
- ⇒ Click Delete Model in the Confirm Delete dialog.

You will now continue with the initial model you built.

## 4.3 Running an Initial Analysis

You are at the point where you can run the model to check the model for correctness. You may wish to save your Workspace before running the model.

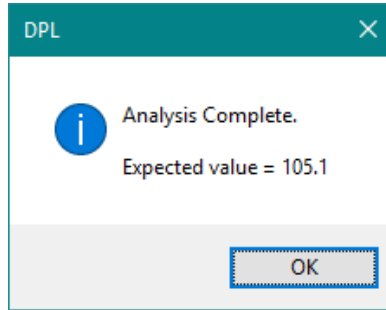
- ⇒ Click on the Home tab. Within the Home | Run group there are a number of different output and run options available in a Decision Analysis run which were covered in Chapter 3.
- ⇒ Uncheck Risk Profile and Policy Tree, as these outputs would be uninteresting (see Figure 4-11).



**Figure 4-11. Decision Analysis Options in Home | Run Group**

- ⇒ Click the Home | Run | Decision Analysis icon to run the default analysis (fast sequence evaluation).
- ⇒ DPL runs the model and brings up the Analysis Complete dialog. See Figure 4-12.





**Figure 4-12. Analysis Complete Dialog**

You will not normally see the Analysis Complete dialog since it is only displayed if no outputs are requested for a Decision Analysis run within the Home | Run group. In this instance, the model is deterministic; it does not yet include any uncertain events so the expected value is just the calculated value of NPV.

⇒ Click OK to dismiss the Analysis Complete dialog.

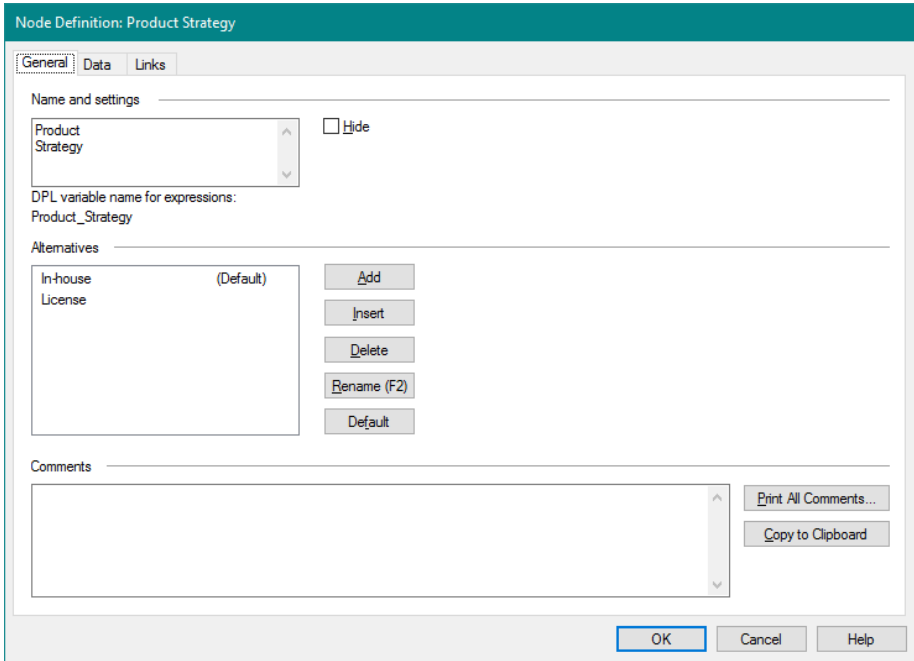
## 4.4 Decision Nodes

---

You will now build out your DPL model by adding a local decision node.

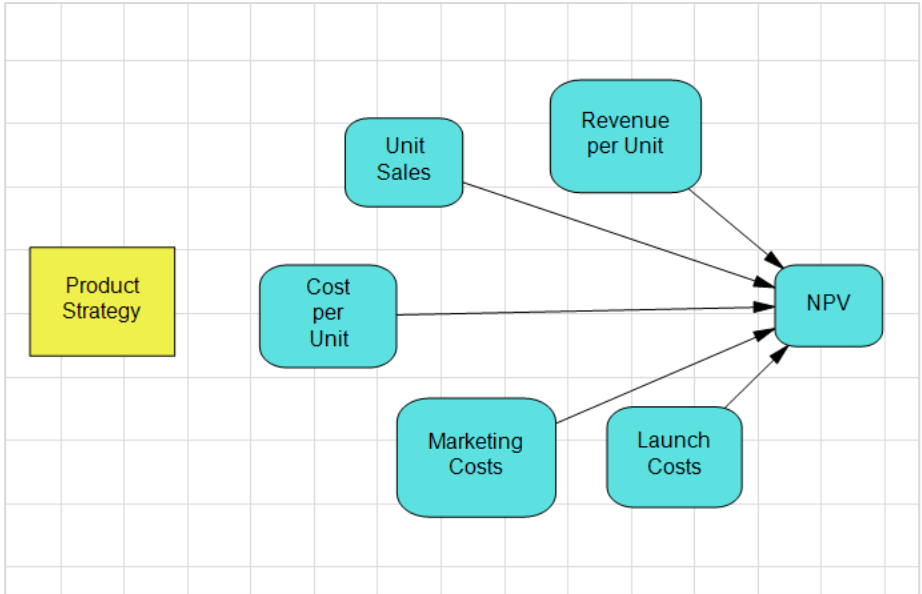
- ⇒ Click the Influence Diagram | Node | Add split button to create a decision node (it should be the default indicated).
- ⇒ Click in a region to the left of the value nodes in the Influence Diagram. The Node Definition dialog appears.
- ⇒ Type "Product Strategy" for the Name of the node. DPL will automatically split the node name into multiple lines.
- ⇒ Select the Yes alternative, type "In-house" to rename it. Note: you may also click the Rename button or press F2 to edit an alternative name.
- ⇒ Rename the No alternative to "License".

Your node definition dialog should now look like Figure 4-13.



**Figure 4-13. Node Definition Dialog for a Decision Node**

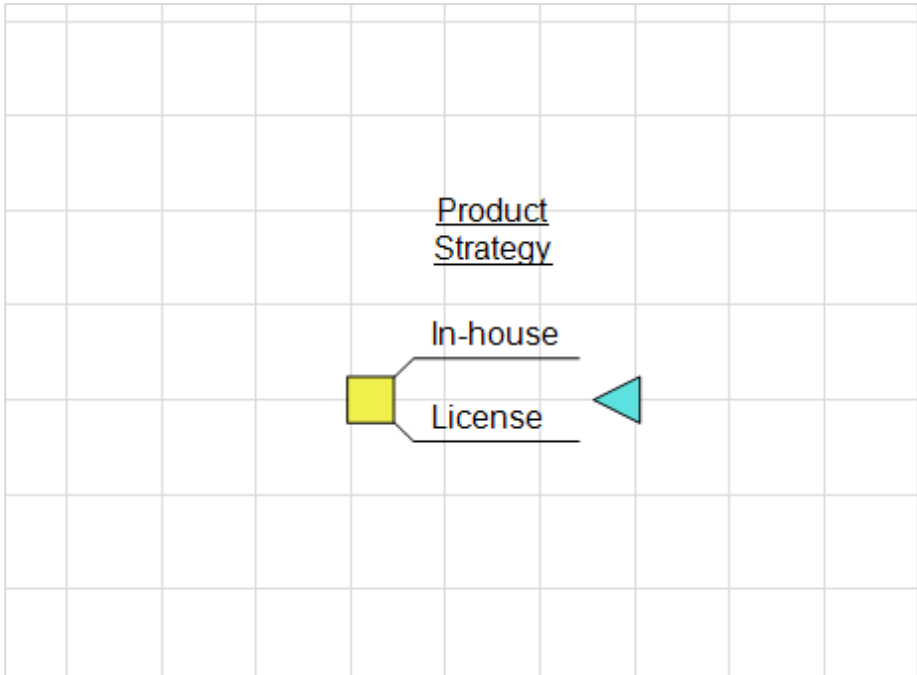
- ⇒ Click OK to close the Node Definition dialog. Your model should now look something like Figure 4-14.



**Figure 4-14. Model with Product Strategy Decision**

Now that there is an event in the model, DPL has begun to build a symmetric, Decision Tree for you within the Decision Tree pane – which currently isn't visible.

- ⇒ Press the Tab key to switch your view to the Decision Tree pane. It should match what you see in Figure 4-15.



**Figure 4-15. Decision Tree pane displaying Default Tree**

This Decision Tree is referred to as the Default Tree. If you do not modify the Decision Tree yourself, DPL will continue to build a symmetric Default Tree for you – even when the Decision Tree pane isn't visible (see Section 2.2 for a discussion of symmetric trees). If you build your model entirely from the Influence Diagram and find once it is completed that the Default Tree has the structure you want, then there is no need to change the Default Tree.

Once you modify the Default Tree, you have taken over control of the tree. DPL will no longer build a Default Tree for you. You can always ask DPL to create the Default Tree from the current Influence Diagram by clicking Decision Tree | Model | Default Tree (see Table 1-4).

⇒ Press the Tab key again to switch back to the Influence Diagram.

In order to incorporate the new decision into your model, you will need to add two additional nodes.

⇒ Create a new value node by dropping down the Influence Diagram | Node | Add split button and selecting Value from the list.

- ⇒ Click near the top of the Influence Diagram to place the new node. The Node Definition dialog appears.
- ⇒ Type "License Fee" for the node name.
- ⇒ Switch to the Data tab.
- ⇒ Type "0.75" for the node data.
- ⇒ Click OK to close the Node Definition dialog.
- ⇒ Click Influence Diagram | Influence/Arc | Add (see Table 1-3). DPL uses License Fee as the predecessor node since it is selected and changes the cursor to the end arc cursor.
- ⇒ Draw an influence arc from License Fee to Revenue per Unit by clicking on Revenue per Unit.

Note: Another way to draw an influence arc is to hold down the Shift key, click on the first node (such as License Fee), and then click on the second node (such as Revenue per Unit). You can also right-click in the Influence diagram and select Add Arc from the context menu.

- ⇒ Repeat the process above, adding a new value node called "Price" with data of "2" that has an arc leading to Revenue per Unit.

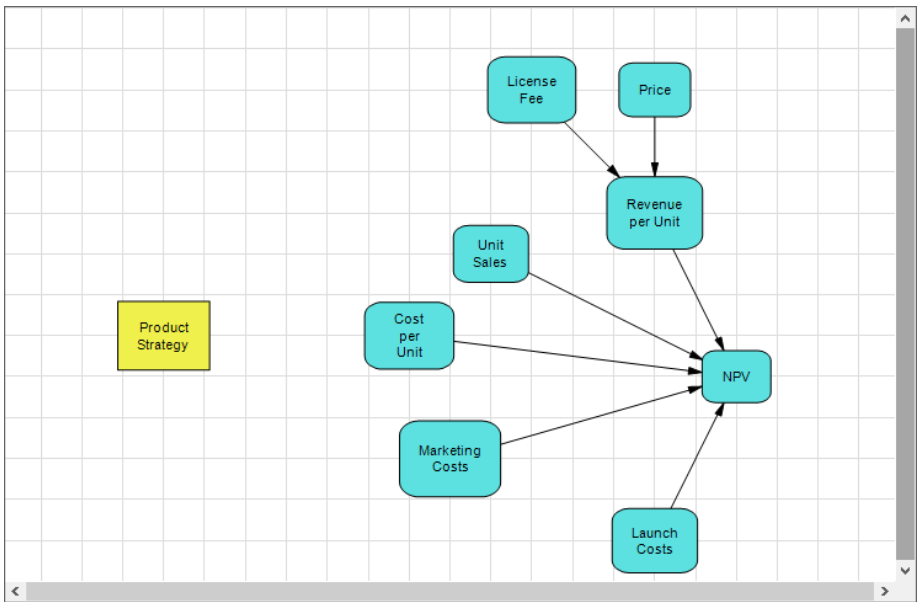
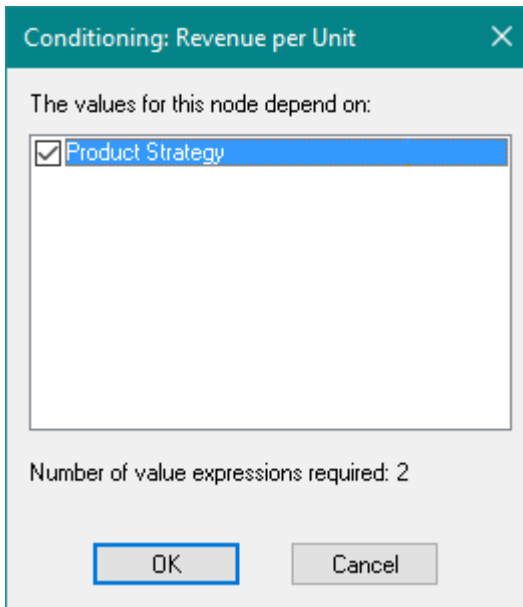


Figure 4-16. Model with Price and License Fee Nodes

Your model should look like Figure 4-16 above. You will now edit the definition of Revenue per Unit to incorporate the Product Strategy decision.


- ⇒ Double-click the Revenue per Unit node to edit it. The Node Definition dialog appears with the Data tab selected.
- ⇒ Click the Conditioning button. The Conditioning dialog appears.
- ⇒ Click the checkbox next to Product Strategy as shown in Figure 4-17.

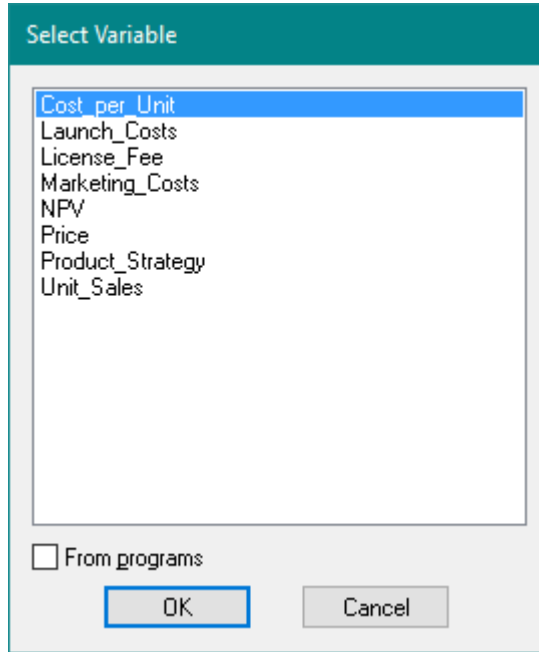


**Figure 4-17. Conditioning Dialog**

- ⇒ Click OK to close the Conditioning dialog.

The Node Definition dialog Data tab now indicates that two values are required for Revenue per Unit: one for the In-house alternative of the Product Strategy decision and a second for the License alternative. DPL has copied the value that was specified when the Revenue per Unit node was not conditioned on anything to both entries. DPL indicates the currently selected entry by coloring it magenta in the data input tree.


- ⇒ With the In-house alternative selected, click the select Variable button (  ) to display the Select Variable dialog. See Figure 4-18.

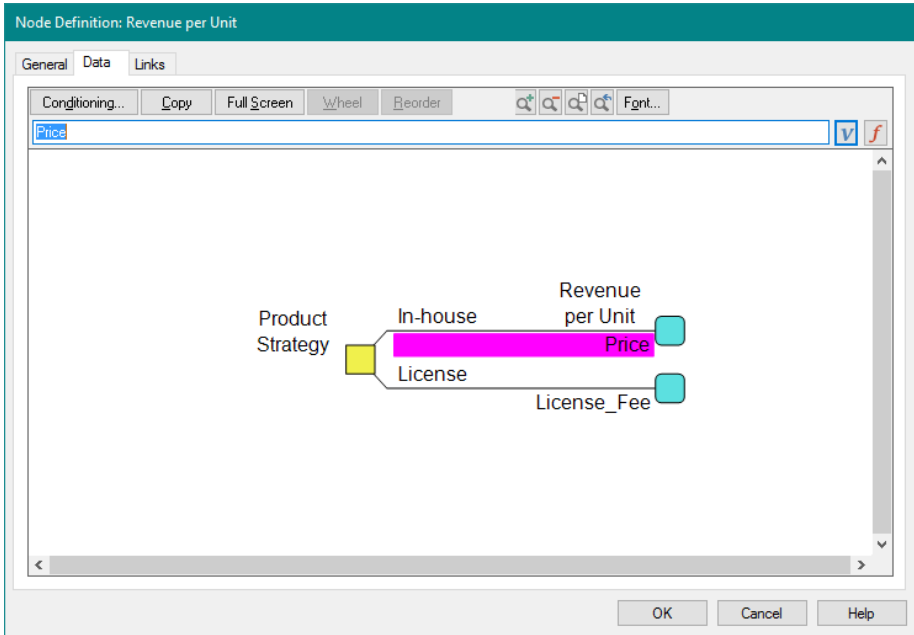


**Figure 4-18. Select Variable Dialog**

⇒ Double-click Price in the list. Press Enter. DPL replaces the number 2 with the Price variable within the data input tree.

Note: In variable names, DPL replaces spaces and punctuation marks with underscores "\_" in multiple word node names so the value node "Unit Sales" has a variable name of "Unit\_Sales". The variable name is what is used in in expressions (formulas). Using the Select Variable dialog saves typing, reduces errors from misspellings and prevents you from having to remember the names of all the variables in your model. Also, DPL variable names are case sensitive; "Costs" is different from "costs".

⇒ With the License alternative selected, click the select Variable button (  ) and double-click License\_Fee in the list. Press Enter. Your data input tree for Revenue per Unit should now look like Figure 4-19.



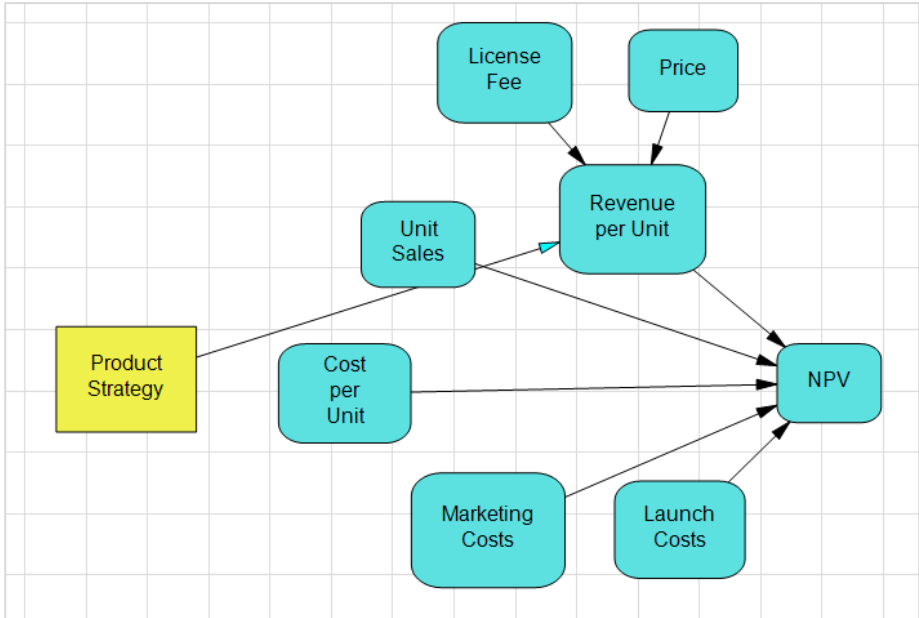
**Figure 4-19. Data Input Tree for Revenue per Unit Node with Conditioning**

You have now told DPL that when Product Strategy is in the In-house alternative, Revenue per Unit is equal to Price, whereas when Product Strategy is in the License alternative, Revenue per Unit is equal to License Fee.

⇒ Click OK to close the Node Definition dialog. Your model should now look something like Figure 4-20.

Note that DPL has drawn an influence arc from Product Strategy to Revenue per Unit to indicate that Product Strategy conditions Revenue per Unit. Also note that the arrowhead of the arc is colored. DPL uses a blue arrowhead to indicate that the values of Revenue per Unit depend on which state or alternative Product Strategy is in. Arrowhead colors are covered in more detail in Section 7.3.





**Figure 4-20. Model with Revenue per Unit Conditioned on Product Strategy**


⇒ Press the Tab key to view the Decision Tree pane.

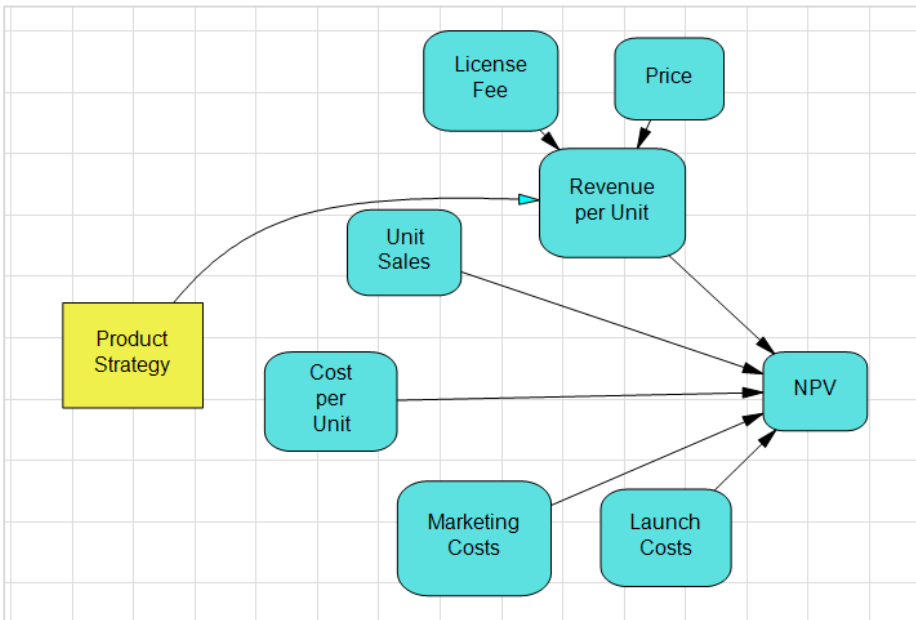
DPL has also updated the Default Tree, adding NPV as a get/pay expression for the decision node Product Strategy. A get/pay expression tells DPL what value to get (i.e., you receive) or pay (i.e., you pay out) at each point in the tree. The get/pay expression is displayed below the branch on which it occurs in the Decision Tree. For now, leave the get/pay expression as is.

It should be noted that value nodes are not displayed directly in the Decision Tree; rather, the names of value nodes appear in get/pay expressions on branches in the Decision Tree. Unlike a decision or chance node, a value node is not an "event" which occurs in the sequence of the tree, it is simply a calculated quantity.

⇒ Press the Tab key to return to the Influence Diagram pane.

You will now neaten up your model by bending the arc from Product Strategy to Revenue per Unit.

- ⇒ Select the influence arc with the left mouse button and without releasing the mouse button begin to drag the arc upward. The mouse cursor will change to the arc bend cursor (  ) and a dashed line will appear to preview the bent arc.
- ⇒ Continue dragging until your arc looks similar to the arc in Figure 4-21.
- ⇒ Release the mouse to fix the arc in place.

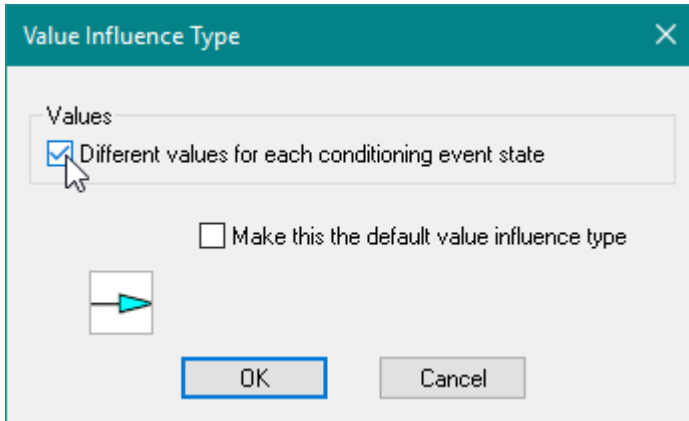


**Figure 4-21. Model with Bent Arc**

When you defined conditioning for Revenue per Unit, you used the Conditioning dialog. DPL also provides a way for you to define conditioning via the arcs in the Influence Diagram. You will now use this procedure to condition Cost per Unit and Marketing Costs on Product Strategy.

- ⇒ Click Influence Diagram | Influence/Arc | Add on the ribbon.
- ⇒ Draw an arc from Product Strategy to Cost per Unit: click on Product Strategy, and then click on Cost per Unit.
- ⇒ Double-click the new influence arc. The Value Influence Type dialog appears.

- ⇒ Check the Different values for each conditioning event state checkbox as shown in Figure 4-22. Notice that the preview arc head is now colored blue.



**Figure 4-22. Value Influence Type Dialog**

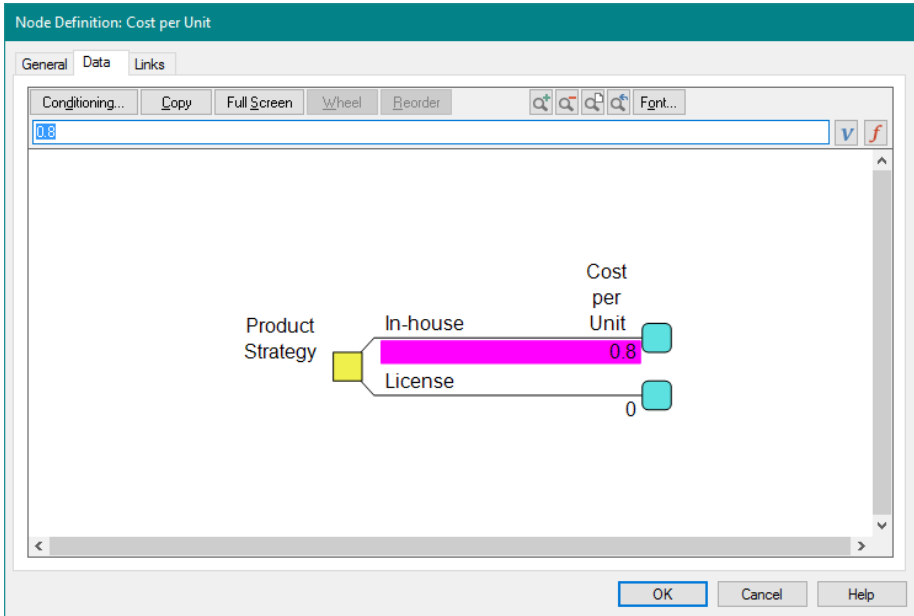
- ⇒ Click OK to close the Value Influence Type dialog.

You have now told DPL that the values for Cost per Unit are conditioned by Product Strategy.

- ⇒ Double-click Cost per Unit. The Node Definition dialog appears with the Data tab selected.

Note that DPL has changed the node data input tree so that it is conditioned on Product Strategy and has copied the data for Cost per Unit to each of these data entry slots.

- ⇒ Arrow down to select the License alternative.
- ⇒ Type "0" and press Enter. Your node data input tree should look like Figure 4-23. (This is a very simple model; if Product Strategy is in the License alternative, costs equal zero.)



**Figure 4-23. Node Data Input Tree for Cost per Unit**

⇒ Click OK to close the Node Definition dialog.

You will now try a different procedure to create an influence arc between Product Strategy and Marketing Costs. To do this you will select multiple nodes in the Influence Diagram by holding down Control and clicking.

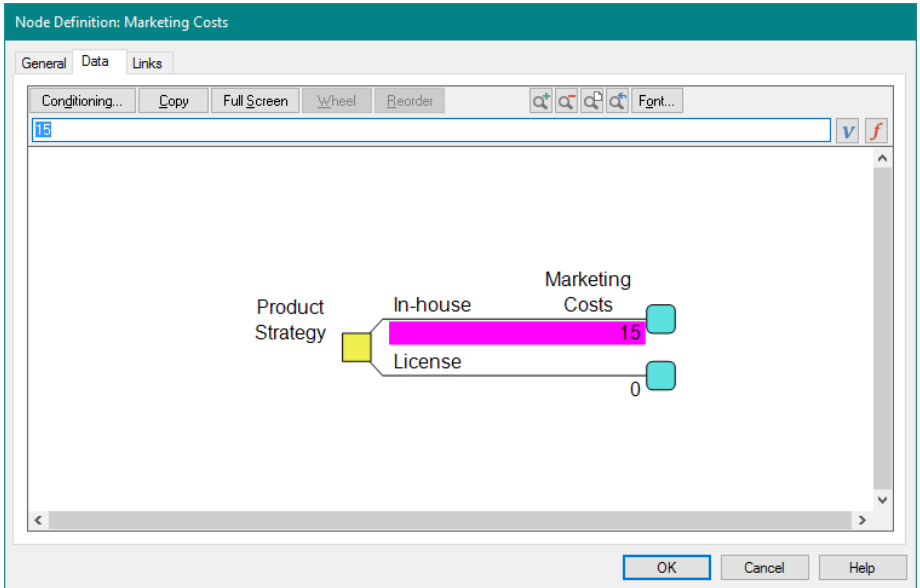
⇒ Click Product Strategy to select it.

⇒ Hold down Ctrl and click Marketing Costs to select both nodes.

⇒ Click Influence Diagram | Influence/Arc | Add. DPL creates an arc from Product Strategy to Marketing Costs.

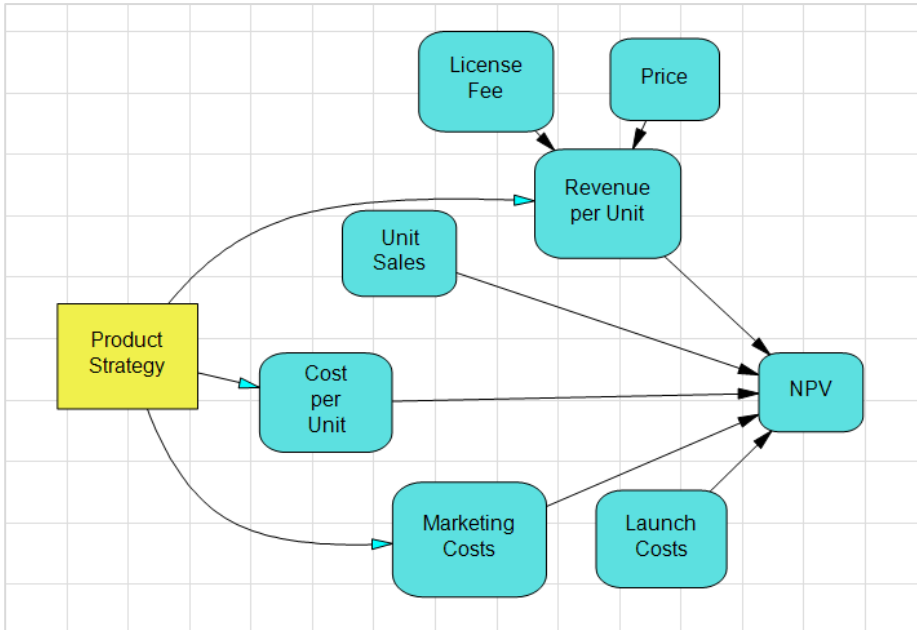
If you have two nodes selected DPL will create an influence arc from the first selected node to the second selected node when you click Influence Diagram | Influence/Arc | Add. Note you can use this method to create multiple arcs at once. If you select more than two nodes and then click the Influence Diagram | Influence/Arc | Add, DPL creates an arc from the first node selected to each subsequent node.

Now repeat the remaining procedure above to make Marketing Costs conditioned on Product Strategy by changing the arc type. Like Cost per Unit, Marketing Costs is zero if Product Strategy is in the License alternative. Your data input tree for Marketing Costs should match Figure 4-24.



**Figure 4-24. Node Data Input Tree for Marketing Costs**

⇒ If you wish, bend the influence arc from Product Strategy to Marketing Costs. Your model should now look something like Figure 4-25.

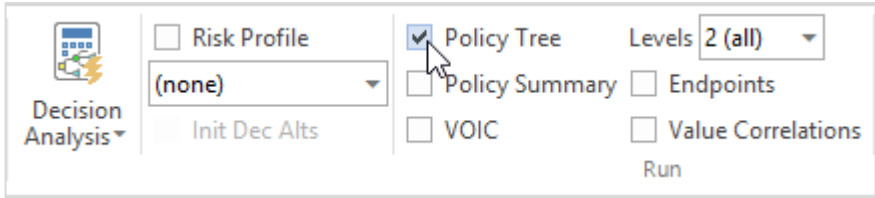


**Figure 4-25. Model with Conditioning from Product Strategy Decision**

Note in this example you conditioned several nodes on the Product Strategy decision. For example, you specified that both Cost per Unit and Marketing Costs were zero if the Product Strategy alternative is License. Another way to do this would be to link the decision node to the spreadsheet with flag values (e.g., 1 and 2) and put the logic for marketing costs, etc. in the spreadsheet (e.g., an Excel formula such as `=if(Product_Strategy=2,0,Marketing_Costs)`). As with any other node type, decision nodes can be linked to Excel.

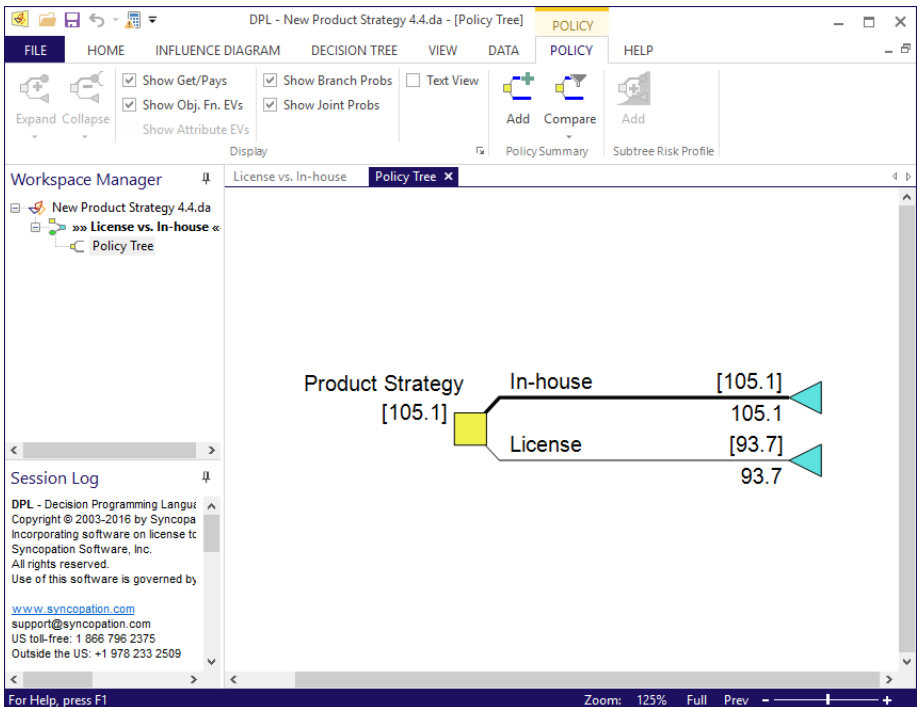
Now that you have added a decision to your model, you will run it again. You may wish to save the workspace first.

- ⇒ For the Decision Analysis options within the Home | Run group make sure Risk Profile is unchecked.
- ⇒ If Policy Tree is unchecked, check it. The Home | Run group should look like Figure 4-26.



**Figure 4-26. Decision Analysis Options for Model with Decision**

- ⇒ Click Home | Run | Decision Analysis or press F10 (F10 is the shortcut for the default run type indicated in the Home | Run | Decision Analysis icon).
- ⇒ DPL runs the model with the newly added decision node and produces the Policy Tree output that you requested. See Figure 4-27.



**Figure 4-27. Policy Tree™ for Model with Decision**

Recall that 105.1 is the same expected value result from when you ran the model in Section 4.3. So the model has not changed much from how it was originally constructed. The optimal decision alternative is to produce the product in-house.

Be sure to save your workspace before proceeding. At the start of Chapter 5 you will be conducting a variety of sensitivity analyses on the model saved at the end of this section.



## 5. Conducting Sensitivity Analyses

Often the first step in a successful analysis is to perform a sensitivity analysis -- one that shows the relative importance of the value drivers. DPL offers a variety of graphical sensitivity analyses that provide you with the insight necessary to prioritize your focus on the variables that matter most in the decision.

DPL's Tornado Diagram's show the effect of varying inputs on the objective function and the optimal decision policy. For a more detailed sensitivity analysis DPL also offers one and two-way Rainbow Diagrams, which can be employed for a particular value or probability.

You will be continuing with the model you saved at the end of Section 4.4.

⇒ Start DPL and open the workspace you saved at the end of Section 4.4 by using File | Open.

Note: if you're starting here and haven't worked through the tutorial in Chapter 4 or are not confident that the model built in Chapter 4 is correct, open the workspace `New Product Strategy_Sensitivity.da` from the Examples folder in your DPL installation directory, usually `C:\Program Files\Syncopation\DPL9\Examples`.

### 5.1 Value Tornado Diagrams

---

You may want to know how sensitive the decision to produce the product in-house is to changes in value for the variables in your model. This can be accomplished by running a Value Tornado.

Note that since the model is deterministic (has no chance nodes -- only value and decision nodes) the Value Tornado diagram is the only type of tornado diagram you can run at this point.

⇒ Click Home | Sensitivity | Tornado (Value Tornado is the default currently indicated by the icon) or press F8 (F8 is the shortcut to run the default tornado type). You will see a blank Value Tornado Setup dialog similar to Figure 5-2 (the figure shows the dialog with the values already entered).

The Value Tornado Setup dialog provides a table in which to enter the low and high settings (range) for each value node that is not a formula. The middle column shows the current setting for each variable.

The Cost per Unit and Marketing Costs variables are not included in the table initially because they are conditioned by the decision and thus have more than one value. You will add these variables to the table now.

⇒ Within the lower half of the dialog in the *Value* section select Cost per Unit from the drop-down list.

Notice that in the *Conditioning* section show in Figure 5-1 it shows that the selected variable is conditioned by Product Strategy. When a value is conditioned by events, you must specify which conditioning states to use for the value. The drop downs in the conditioning section currently show that the In-house alternative conditioning state is selected. The Current value box shows that the value for Cost per Unit given In house is 0.8 (this is written as Cost\_per\_Unit|Product\_Strategy.In\_house).

**Figure 5-1. Adding Conditioned Values within Value Tornado Setup Dialog**

⇒ Click the Add button. A line for "Cost\_per\_Unit|Product\_Strategy.In\_house" is added to the table, i.e., you are going to run the tornado on the value for Cost per Unit given that Product Strategy is In house.

⇒ Again, within the *Value* section select Marketing Costs from the drop-down list and click Add.

"Marketing\_Costs|Product\_Strategy.In\_house" will be added to the table.

⇒ Click the blank cell for the Low value of Launch Costs and enter 20.

⇒ Click the blank cell in the High column of Launch Costs and enter 25.

- ⇒ Continue to fill in the Value Tornado Setup dialog with the rest of the High and Low values shown in Table 5-1. Note that you can use the arrow keys to move around.

<b><u>Value</u></b>	<b><u>Low</u></b>	<b><u>High</u></b>
Launch Costs	20	25
License Fee	0.7	0.8
Price	1.9	2.3
Unit Sales	20	60
Cost per Unit	0.8	1
Marketing Costs	5	20

**Table 5-1. Variables and Ranges for a Value Tornado**

When you are done, the Value Tornado Setup dialog should look like Figure 5-2.

Value Tornado Setup - License vs. In-house

Bars

Value	Low	Current	High
Launch_Costs	20	22	25
License_Fee	0.7	0.75	0.8
Price	1.9	2	2.3
Unit_Sales	20	40	60
Cost_per_Unit Product_Strategy.In_house	0.8	0.8	1
Marketing_Costs Product_Strategy.In_house	5	15	20

Remove

Value: Price (dropdown) Current value: 2 (input) Add (button) Replace (button)

Conditioning

Initial Decision Alternatives

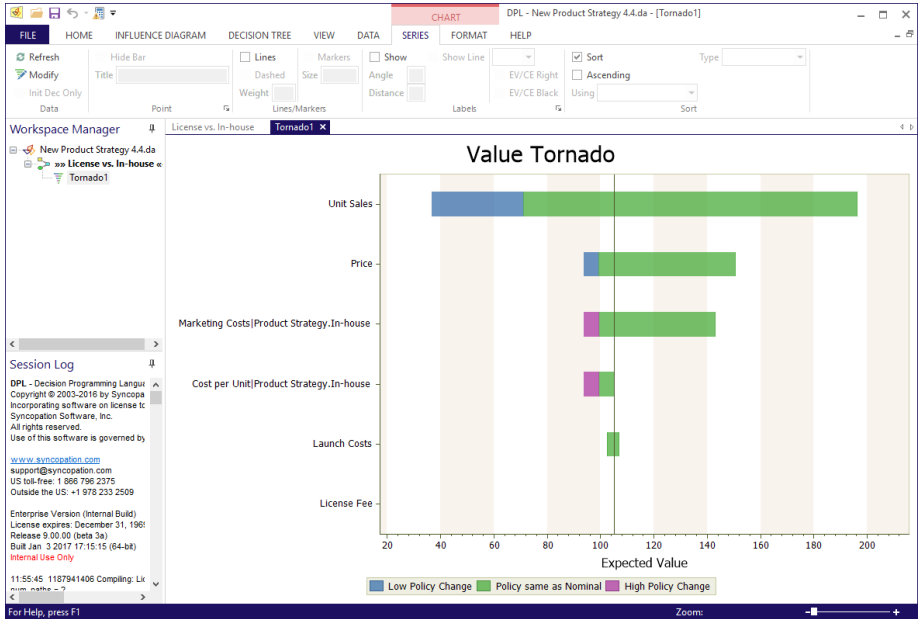
OK (button) Cancel (button)

**Figure 5-2. Value Tornado Setup Dialog**

Note that the Low value for Cost per Unit is the same as the current value. Assume you have reliable information that the cost cannot be less than 0.8.

You have now selected the values to include in the Value Tornado and specified ranges for each.

⇒ Click OK to run the Value Tornado. DPL generates the Value Tornado as displayed in Figure 5-3.



**Figure 5-3. Value Tornado Diagram**

The x-axis (horizontal axis) of the Value Tornado diagram displays the change in the objective function of the model as each of the variables on the vertical axis is changed from the low setting you specified to the high setting you specified. For a given variable (e.g., Unit Sales) it is changed from the low setting to the high setting while all other variables are held at their current settings. DPL runs the model once to establish the value of the objective function with all variables at their current setting (for a model with only value and decision nodes). This is called the Base Result and is indicated by the vertical line in the Value Tornado. See Section 5.2 for a further discussion of the Base Result. Note that the vertical line on the diagram is at 105.1, which is the value you obtained running the model previously. DPL sorts the bars so that the variable that has the biggest impact on the objective function is at the top.

The changes in color indicate that a policy change has occurred for either the low or high setting of the variable. In this model, a policy change occurs when a different alternative is optimal for the upfront decision (the decision at the head, i.e., far left, of the Decision Tree) from that which is optimal for the Base Result policy.

DPL indicates that a policy change has occurred for the low setting of the variable by displaying a portion of the bar for that variable in blue. For a high setting policy change, the end of the bar is displayed in light magenta.

The legend located at the beneath the tornado diagram helps to explain this.

For different values, the blue and/or light magenta may occur on either side of the vertical line representing the Base Result. In Figure 5-3, light magenta and blue both occur to the left of the Base Result. This is because a low setting of Unit Sales decreases the objective function (NPV) and causes a policy change while a high setting of Marketing Costs also decreases the objective function (and causes a policy change).

Note that DPL displays the change in color from green to blue (or light magenta) midway between the value of the objective function for the current setting and the value for the low (or high) setting. For each variable, DPL has only tested a low, nominal and high value and the position of the change in color does not indicate the exact value at which the change in policy occurs. If you would like more precise information on when the change in policy occurs, you could run a Rainbow Diagram (see Section 5.7) to narrow down the range.

## 5.2 Formatting Tornado Diagrams and Other Charts

---

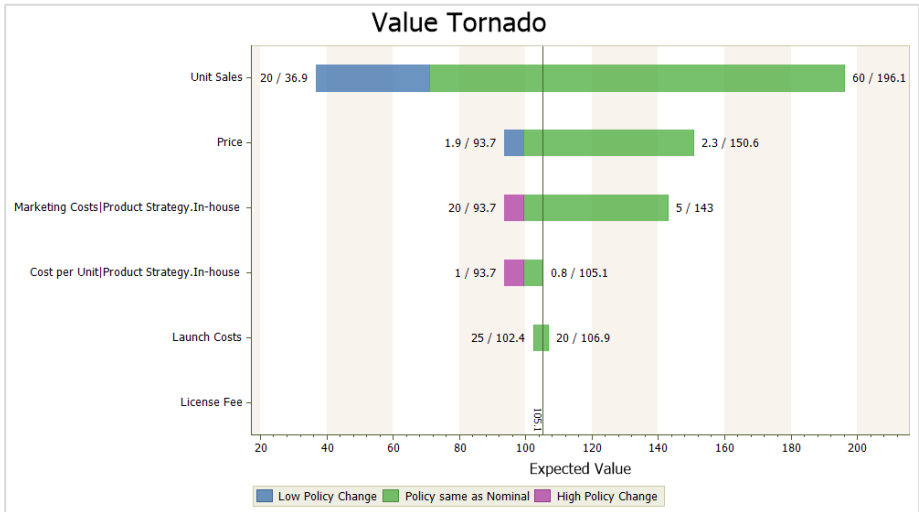
When a tornado diagram or any other graphical output is active, the Chart tab group opens with pink context shading and outline, as shown in Figure 5-4. It contains two tabs: Series and Format.

Within the Format tab you can change several display options: titles, legends, axis, colors, lines, shading, and overall display of the chart. To format the policy change colors, you can chose from color palettes provided with DPL by checking the Use palette checkbox and then choosing a palette from the palette gallery in Chart | Format | Color. You can also choose a custom color via the bucket fill button.

If you have specific color preferences, such as a corporate standard color scheme for presentations, you can create your own custom palette in .xml format and add the file to the location indicated in the Palette tooltip (the tooltip will appear when you hold your mouse over one of the palette selections).

You can format the selected text within the chart via the View | Font group.

- ⇒ To add labels to your chart, check the Show checkbox within Chart | Series | Labels. The labels include the High/Low input values for each variable and the resulting objective function (i.e., NPV) values.
- ⇒ Check the EV/CE Values checkbox in Chart | Format | Display. The tornado is updated as shown in Figure 5-4.



**Figure 5-4. Value Tornado with Values Displayed**

DPL displays the low and high settings for each variable as well as the expected value of the objective function when the variable is at its low and high setting. The variables' low or high settings appear before the "/" character and the expected values appear after. In addition, DPL displays the expected value for the Base Result at the bottom of the vertical line in the Value Tornado. If a variable has no impact on the objective function as it is changed from its low to high setting, DPL displays a zero width bar and no labels for that variable (as is the case for License Fee).

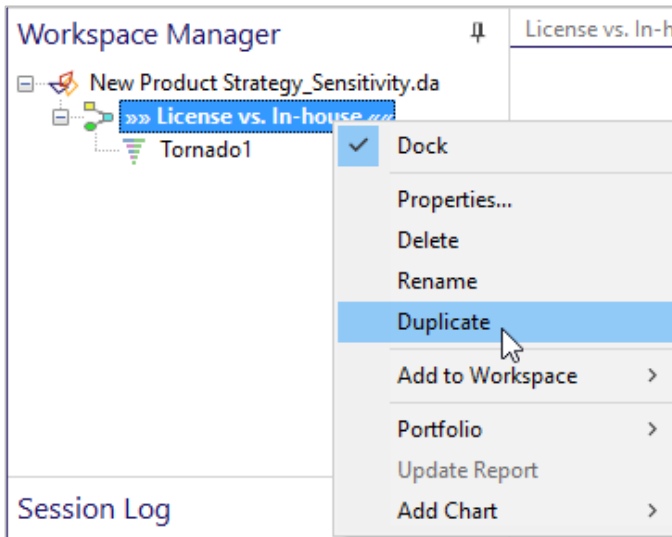
The Value Tornado indicates that four variables result in a change in policy if they are at either their low or high setting. These variables are called decision sensitive. In this sample model, there is only one decision with two alternatives, so you know that a policy change implies that rather than producing the product In-house the optimal alternative is to License it. For more information on determining what policy change has occurred, see Section 5.7.

You have just discovered that several of the variables in your model are decision sensitive. Assume you do not know the value of these variables for certain. You may want to know what the optimal policy is when these

variables are treated as uncertainties. You will now change your model in order to treat these variables as uncertain quantities.

You're about to make significant changes to the model, so you will create a copy of the model as it is and edit the copy. This is good practice when you have results, such as a Value Tornado, which are based on a simpler version of your model and which you may later need to reproduce.

- ⇒ Right mouse-click on the item for the model in the Workspace Manager. The Workspace Manager context menu appears as in fi



**Figure 5-5. Using the Workspace Manager to Duplicate a Model**

- ⇒ Select Duplicate. The model is duplicated with the same name plus "-copy" appended to it.
- ⇒ Click the item for the new model in the Workspace Manager and Press F2 to edit the name.
- ⇒ Rename it "License vs. In-house prob".
- ⇒ Press Enter.

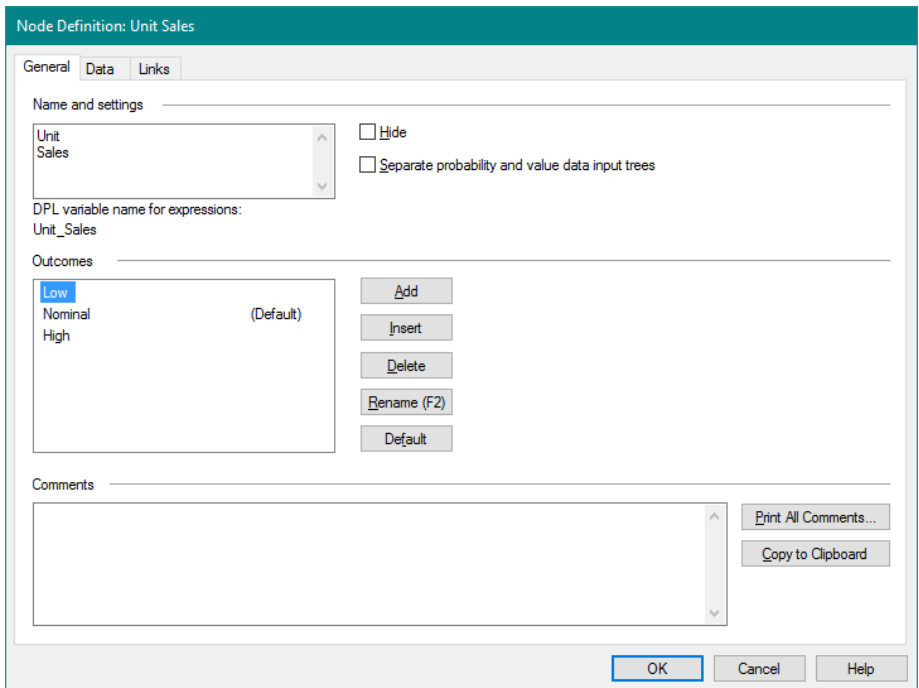
The new model is now active in the Model Window. You may wish to save the workspace at this point.



## 5.3 Discrete Chance Nodes

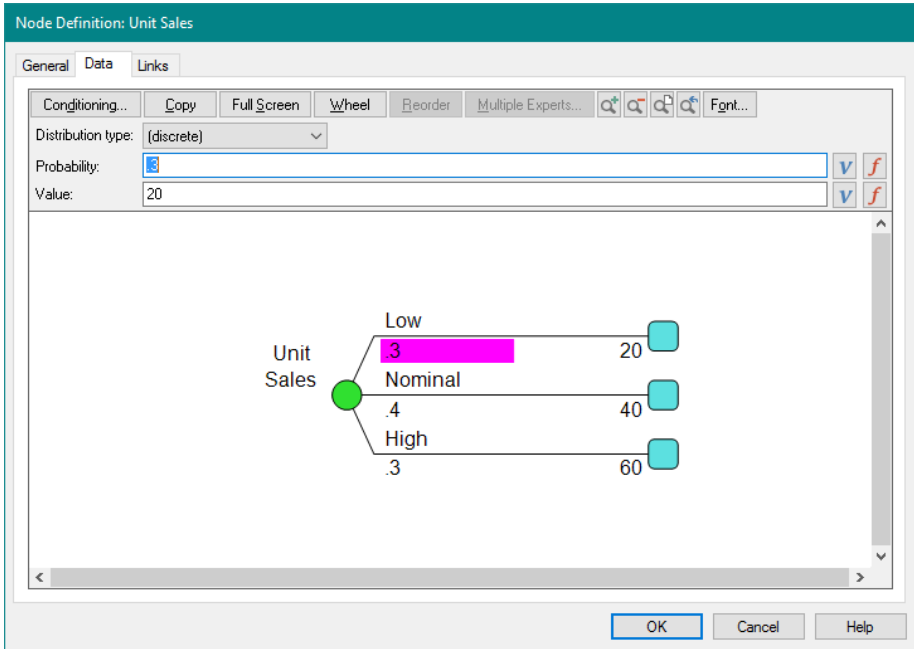
You will now introduce uncertainty into the model by changing some of the value nodes to discrete chance nodes. More specifically, you will be changing the value nodes that are decision sensitive into discrete chance nodes.

- ⇒ Select Unit Sales.
- ⇒ Click the drop-down split of the Influence Diagram | Node| Change To button on the ribbon (see Table 1-2) and select Discrete Chance from the list.
- ⇒ The Node Definition dialog appears as shown in Figure 5-6.



**Figure 5-6. Node Definition Dialog General Tab for a Discrete Chance Node**

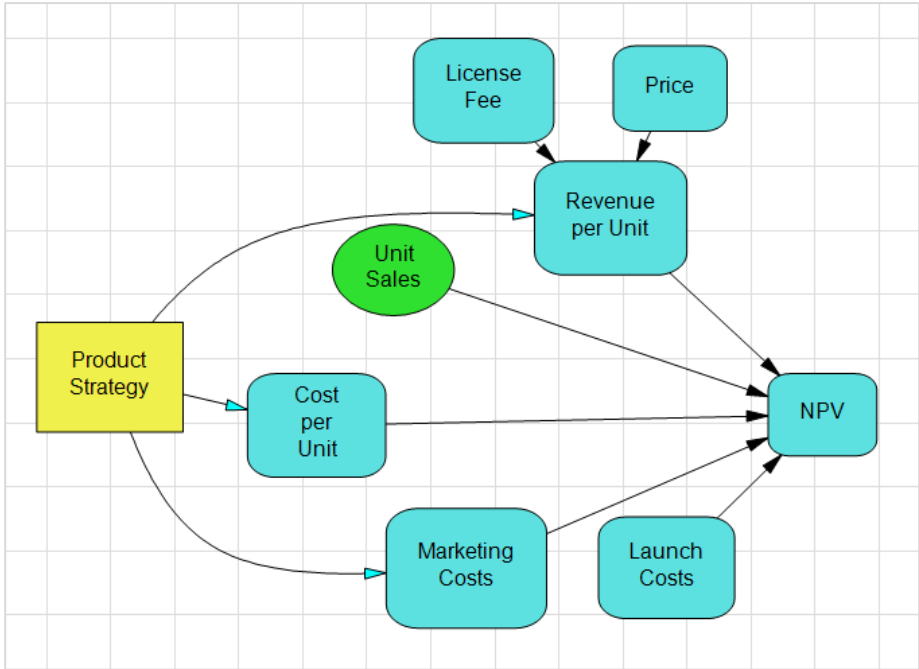
- ⇒ Leave the settings as they are on the General tab and select the Data tab. The Data tab for the Unit Sales discrete chance node is shown in Figure 5-7.



**Figure 5-7. Node Definition Dialog Data Tab for a Discrete Chance Node**

DPL assigns default probabilities to the three chance outcomes, which can be changed in File | Options | General. DPL also assigns the low and high values that you specified in the Value Tornado to the Low and High outcomes (as well as the value that was specified for the value node to the Nominal outcome).

- ⇒ Click OK to accept the probabilities and the previously specified low and high values. Press ESC to deselect Unit Sales. Your model should look something like Figure 5-8.

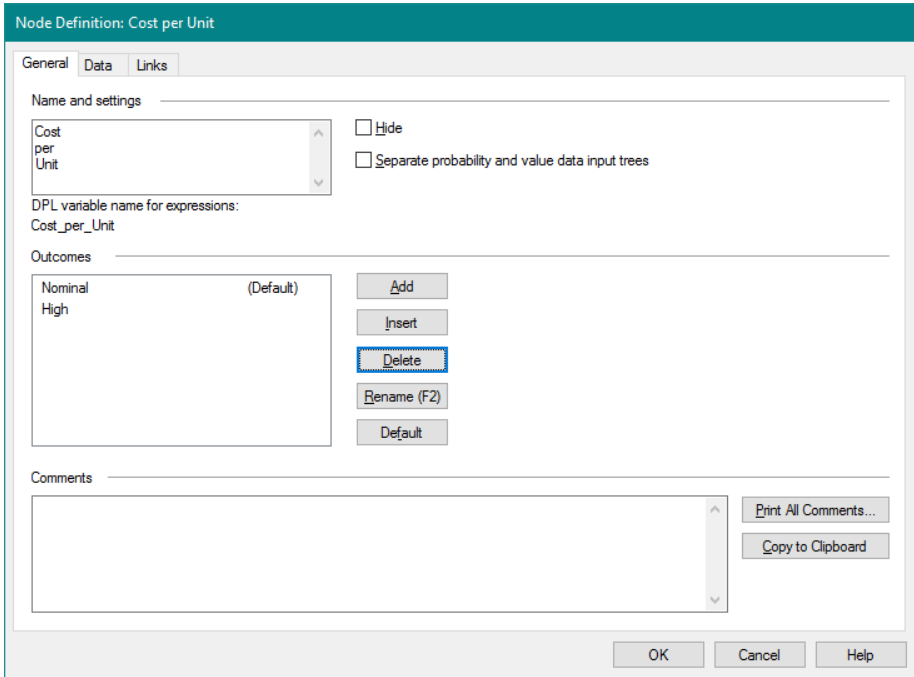


**Figure 5-8. Model with Sales Changed to a Discrete Chance Node**

If you switch to the Decision Tree pane, you'll note that DPL has updated the Default Tree by adding the Unit Sales discrete chance node to it and moving the NPV get/pay expression to the branches for Unit Sales. In the Decision Tree, DPL displays discrete chance nodes as bright green circles.

- ⇒ Select Cost per Unit and change it to a Discrete Chance Node. Note that the last node type changed to was Discrete Chance, so it is now indicated as the default in the Change To icon. Now you can simply click the Influence Diagram | Node | Change to icon.
- ⇒ The Node Definition dialog appears with the General tab selected.
- ⇒ You will now remove one of the chance outcomes that DPL provides by default. On the General tab of the Node Definition dialog, select the Low outcome in the Outcomes list box and click the Delete button.
- ⇒ DPL will notify you that the reduction in states will result in node data being deleted. Click OK to continue.

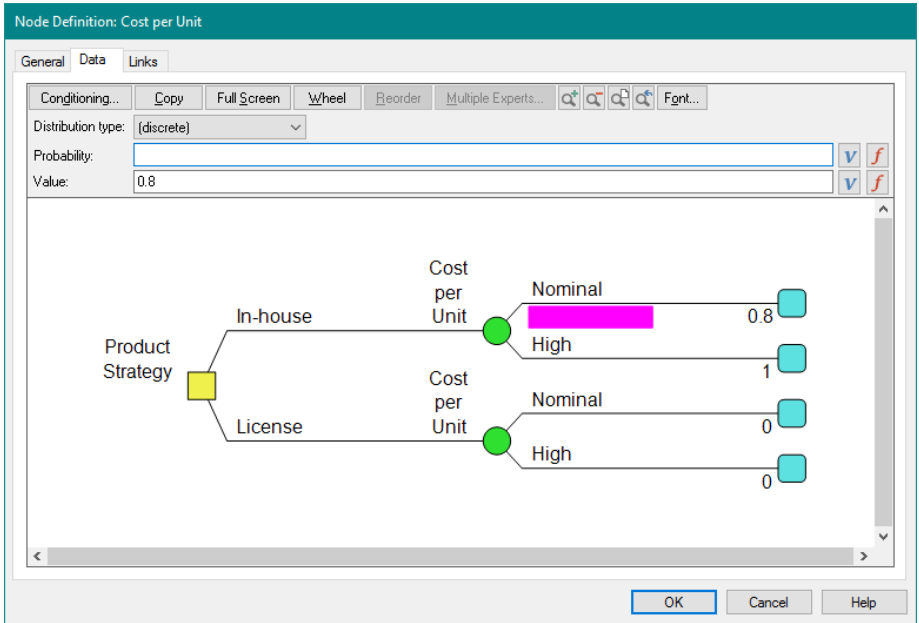
The Node Definition dialog should now look like Figure 5-9.



**Figure 5-9. Node Definition Dialog General Tab for Cost per Unit**

⇒ Select the Data tab of the Node Definition dialog.

Because Cost per Unit is conditioned by the Product Strategy decision, there are two sets of probabilities and values, one for In-house and one for License (See Figure 5-10). No probabilities are given for the Nominal and High outcomes. The default probabilities are only given for chance nodes with the same number of outcomes as specified by the defaults in File | Options | General. However, the Nominal and High value data that you specified for the In-house alternative when you ran the Value Tornado are filled in for the In-house condition state.



**Figure 5-10. Node Definition Dialog Data Tab for Cost per Unit**

⇒ Type "0.6" in the Probability edit box for the Nominal branch of the In-house alternative.

⇒ Arrow down twice to select High.

⇒ Type "0.4" in the Probability edit box for High.

Because both outcomes for the License alternative have a value of zero, the probabilities under the License branch don't affect the results, but they still need to be filled in.

⇒ Arrow down twice to move to the probability for Nominal for the License branch.

⇒ Type "0.5" in the Probability edit box for Nominal.

⇒ Arrow down twice to select High.

⇒ Type "0.5" in the Probability edit box for High.

⇒ Click OK to close the Node Definition dialog.

⇒ Change the Node Type of Marketing Costs to a discrete chance node (Influence Diagram | Node | Change To).

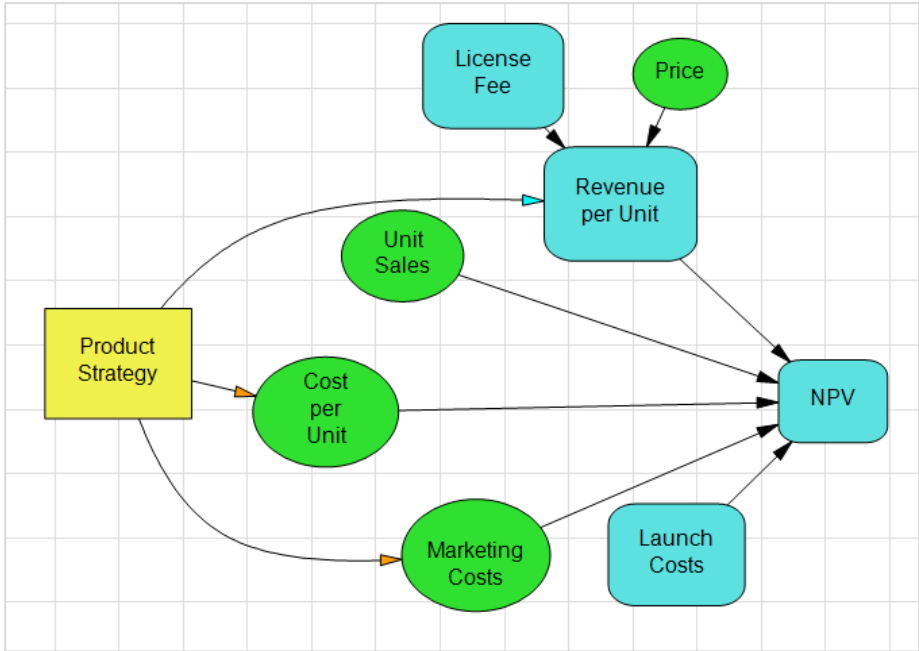
⇒ Accept the default outcomes on the General tab and select the Data tab.

- ⇒ Accept the default probabilities and previously specified values.
- ⇒ Click OK to close the Node Definition dialog.
- ⇒ Repeat the above steps for Price.

Note: If you don't review the data on the Data tab, DPL will warn you that default probabilities have been used when you first run the model and will ask whether you wish to continue the run with those probabilities.

As you saw earlier, you can select multiple nodes in the Influence Diagram. When changing node types, you can select multiple nodes and change their types. You could have done this in this example. When you change multiple nodes at once, DPL does not bring up the Node Definition dialog so you need to edit each node to make further changes as appropriate once you have changed the type. In particular if you change multiple nodes and don't review the node data, you will get the default probabilities warning mentioned above.

Your model should now look something like Figure 5-11. DPL has updated the Default Tree further by adding the discrete chance nodes for Cost per Unit, Marketing Costs, and Price and by moving the NPV get/pay expression to the end of the tree on the branches for Price. Tab over to the Decision Tree pane to view it.



**Figure 5-11. Model with Four Discrete Chance Nodes**

You are now almost ready to run the modified model with uncertainty incorporated into it. There are now two models in the Workspace. DPL will always run an analysis on the model you are viewing or on the model whose output you are viewing. If that model is not the active model in the Workspace Manager (indicated by bold font), DPL will make it the active model.

## 5.4 Base Case Tornado Diagrams

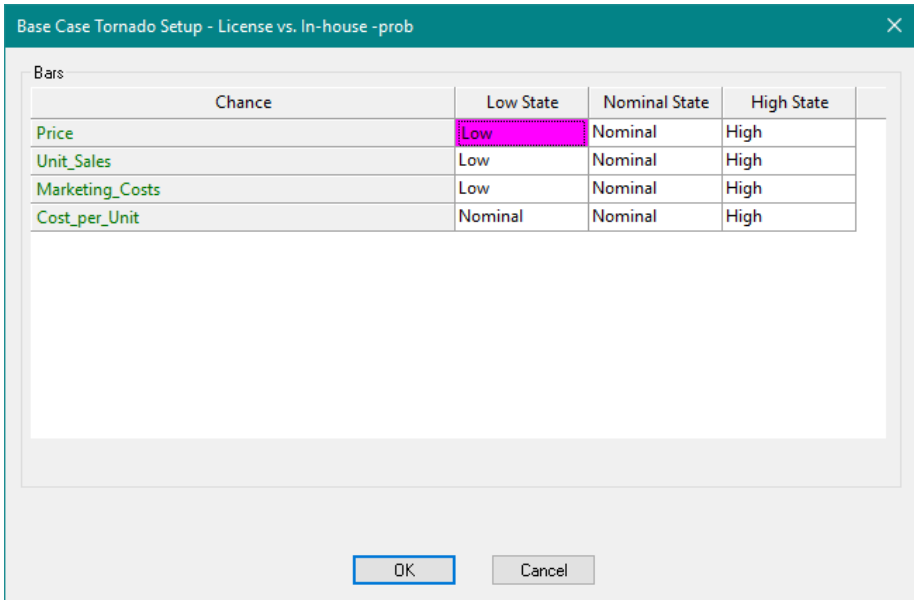
You are now ready to run another sensitivity analysis on your model: the Base Case Tornado Diagram. The Base Case Tornado diagram is designed for models which have several discrete chance nodes, each with a plausible default state (the Nominal or Base Case value) and most of which have three (or more) outcomes.

To compute a Base Case Tornado Diagram, DPL first sets all discrete chance nodes to a state that you define to be Nominal for each node to calculate a Base Case. It then varies each discrete chance node individually to a state you specify to be Low and to a state you specify to be High while

keeping all other discrete chance nodes at their Nominal states. The Base Case Tornado Diagram is useful for understanding the impact of each discrete chance node on the model's objective function.

⇒ To run a Base Case Tornado Diagram, drop-down the Home | Sensitivity | Tornado split button and choose Base Case from the list.

The Base Case Tornado Setup dialog appears as shown in Figure 5-12.



**Figure 5-12. States for Base Case Tornado Diagram Dialog**

Note: the default tornado type for the Home | Sensitivity | Tornado split button changes based upon the last tornado type you have run. The default is now Base Case Tornado.

To run a Base Case Tornado Diagram, you must tell DPL which state (i.e., outcome) is Low, Nominal and High for each discrete chance node. For three-outcome discrete chance nodes, this is usually straightforward. If you have used the default discrete chance node settings provided by DPL, the exercise is easy, as is the case for Marketing Costs, Price and Unit Sales. Figure 5-12 indicates that the states specified to be Low, Nominal and High for Marketing\_Costs, Price and Unit Sales are Low, Nominal and High, respectively.

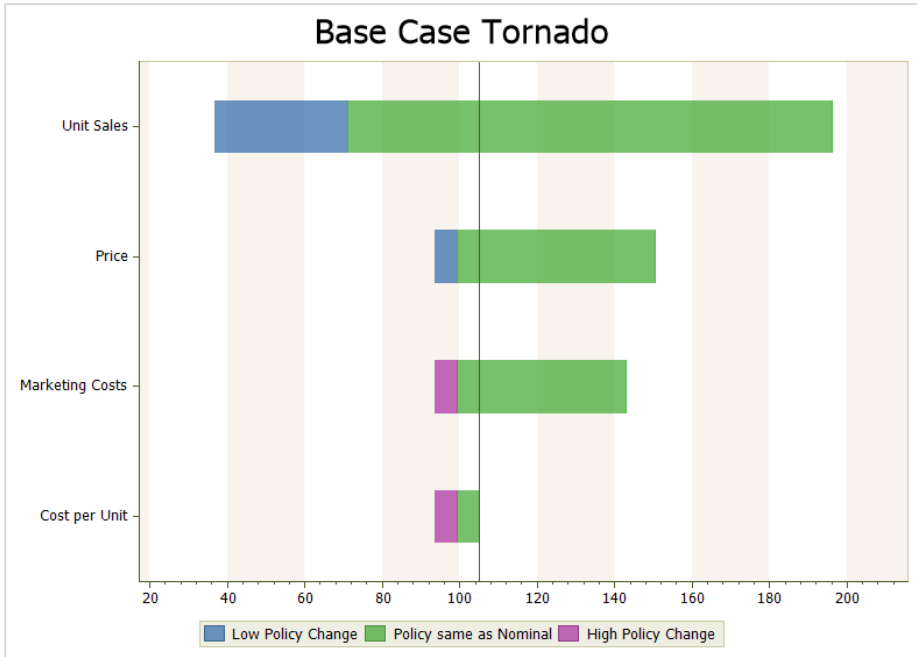


- ⇒ Click in the cell for the Low State of Marketing Costs. You can see that there is a combo box that allows you to type or select the setting for the state of this chance node.
- ⇒ Use the arrow keys to move around the other cells for each state of each chance node, leaving the settings as DPL has suggested.

You will also notice that since Cost per Unit is a two-outcome discrete chance node, one of its outcomes will have to be repeated for the Low, Nominal or High state specification. DPL has suggested using the Nominal state for both the Low and Nominal tornado diagram setting. Leave this setting as is. Note: to start DPL chooses the outcome defined to be Default in the Node Definition dialog to be Nominal for a Base Case Tornado, the first outcome for Low and the last outcome for High. For discrete chance nodes with more than three outcomes, you may need to change the settings in the Base Case Tornado Setup dialog.

- ⇒ Click OK to run the Base Case Tornado Diagram.

DPL produces the Base Case Tornado Diagram as shown in Figure 5-13. In this example the Base Case Tornado is similar to the Value Tornado that you ran on the deterministic model (i.e., the model with no discrete chance nodes) in Section 5.1. This will not always be the case for several reasons. First, the Base Result of a Value Tornado may not be the same as the Base Case for the Base Case Tornado if the states chosen as Nominal for the Base Case Tornado do not match what was used as the current values in the Value Tornado. Second, if you change the high or low values for value nodes that you have changed to discrete chance nodes, the two tornado types will differ.



**Figure 5-13. Base Case Tornado Diagram**

Base Case Tornado Diagrams are useful for understanding how the chance nodes in your model impact the objective function. They are also flexible because you can add bars to a Base Case Tornado Diagram for values in your model. This allows you to compare the impact of chance nodes across their range of states to the impact of changes in values across a range of settings. Lastly, Base Case Tornado Diagrams are very quick to run which may be helpful for larger models with many chance nodes.

While Base Case Tornado Diagrams are relatively simple and easy to understand, they can be limited in terms of showing impacts in a probabilistic model where there is a lot of conditioning and/or where probabilities represent skewed distributions. Base Case Tornado Diagrams don't use probabilities, so if, for example, the probabilities of a chance node reflect a skewed distribution, the Base Case Tornado will show the same results as for a symmetric distribution. A Base Case Tornado is also difficult to use when there is no natural middle case, as in a risk analysis model which considers a number of very unlikely bad outcomes. In later chapters we'll discuss more advanced tornado diagrams that can be insightful for complex models. See Section 14.3, *When to Use Which DPL™ Sensitivity Output*, for a comparison.

## 5.5 Initial Decision Alternatives Tornado Diagrams

---

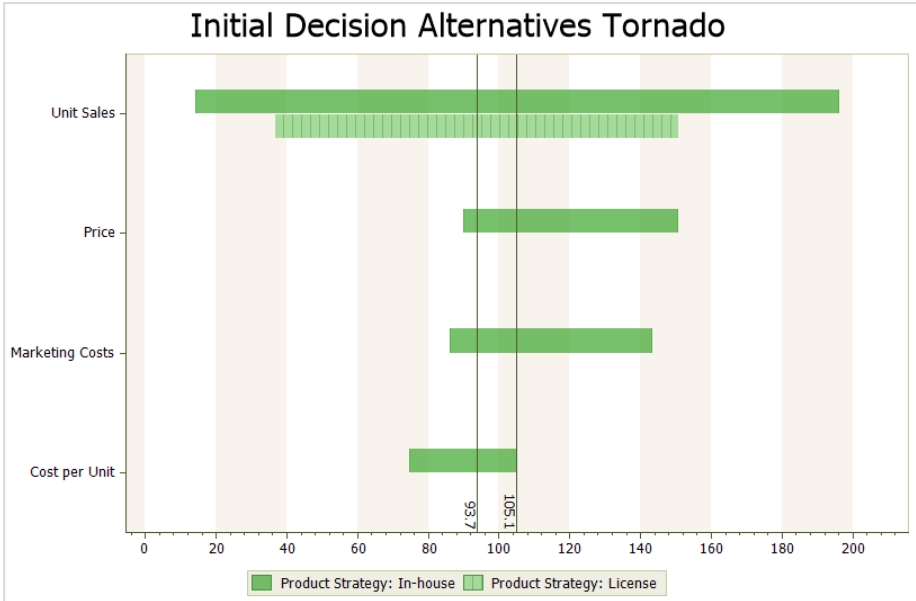
The Initial Decision Alternatives Tornado diagram is an extension of the Base Case tornado that you ran in Section 5.4. To create an Initial Decision Alternatives Tornado, DPL sets the initial decision to each of its possible alternatives and runs a Base Case tornado for each alternative. The resulting Base Case tornadoes are then shown on one diagram.

You will now run an Initial Decision Alternatives tornado using the model you built in the previous sections.

- ⇒ Make sure the "License vs. In-house prob" model is still the active model.
- ⇒ Drop down the Home | Sensitivity | Tornado split button and select Initial Decisions Alternative from the list.

The Initial Decision Alternatives Tornado Setup table appears. It should be identical to the Base Case Tornado Setup table you examined in Section 5.4.

- ⇒ Leave the settings as they are.
- ⇒ Click OK. An Initial Decision Alternatives tornado appears as shown in Figure 5-14.



**Figure 5-14. Initial Decision Alternatives Tornado Diagram**

The tornado shown in Figure 5-14 is an example of a very simple Initial Decision Alternatives tornado. There are two initial decision alternatives (In-house and License), so the tornado diagram essentially consists of two Base Case tornadoes on one chart. The diagram shows that the chance event Unit Sales has the greatest impact on the objective function value, regardless of which initial alternative is pursued. It also shows that the impact on the objective function of Unit Sales is less for the License alternative than it is for the In-house alternative (the bar for License is narrower). Also, the diagram shows that if the License alternative is pursued (despite the fact that it is not the optimal policy), the chance events Marketing Costs and Cost per Unit have no effect on the objective function. This makes sense based on how the model is set up. If the product is licensed, marketing costs and per unit costs are zero.

Initial Decision Alternatives tornadoes are helpful for comparing the impact of each chance event on the objective function across the possible alternatives of the initial decision in the model. Although the tornado you ran in this section is simple and illustrative, Initial Decision Alternatives tornadoes can be quite powerful when used with models that incorporate several strategic alternatives and several chance events. Note in the simple example model there is only one decision in the model. Since DPL runs an Initial Decision Alternatives Tornado by fixing the initial decision at each alternative and running a Base Case Tornado, no policy changes will be

seen in this model. However for models with subsequent decisions after the initial decision (often referred to as downstream decisions), policy changes may occur in an Initial Decision Alternatives Tornado since the optimal alternative may change for a downstream decision.

You may want to save the workspace before you proceed to the next section.

The model you built in Chapter 3 is now ready to produce the full suite of results that are available within the Home | Run group when you run a decision analysis.

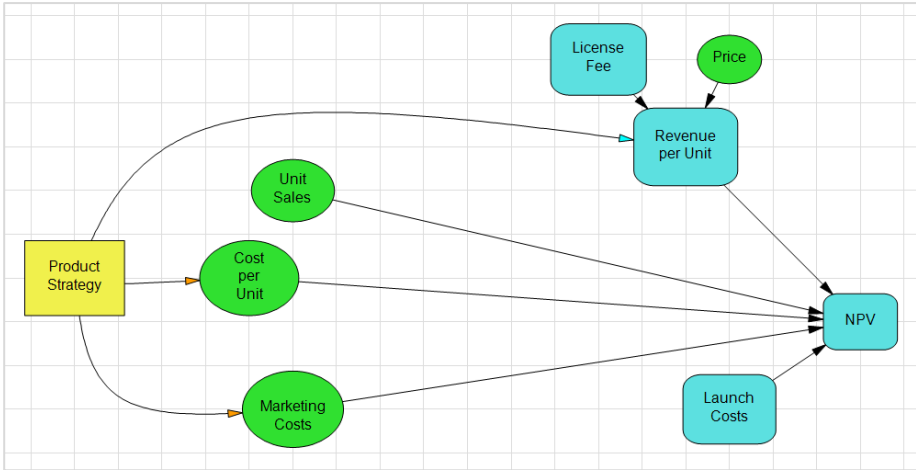
## 5.6 Adding and Analyzing a Downstream Decision

---

After some consideration management has decided to add more flexibility to the model by breaking the Unit Sales variable into two time periods: Sales 1 and Sales 2. They also would like to add an explicit downstream decision to raise or maintain the current pricing after sales in the first period have been observed. If sales are strong in the first time period, it may make sense to raise the price for the product.

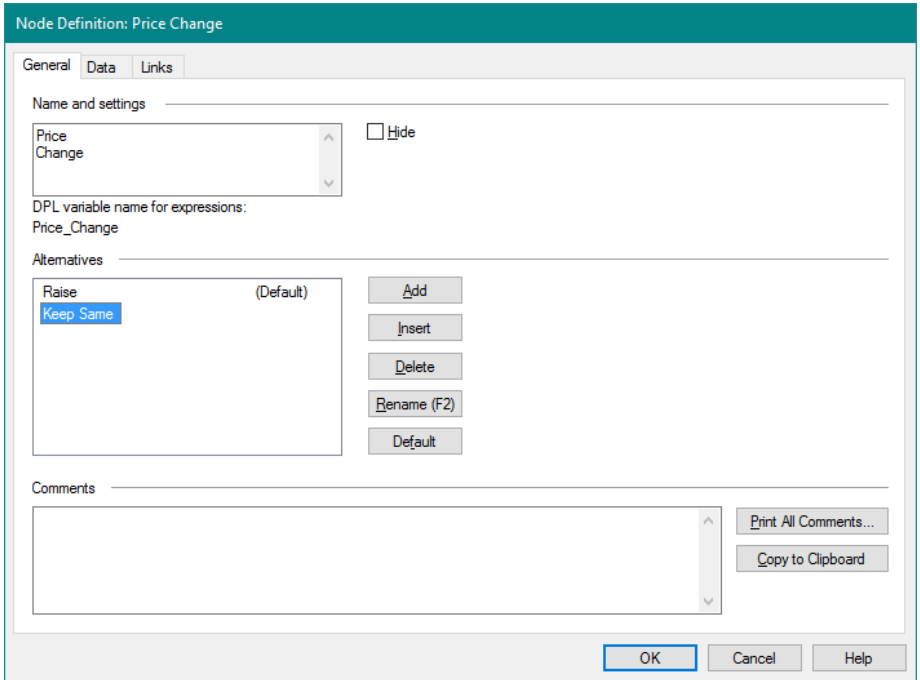
- ⇒ Right mouse-click the item for the model "License vs. In-house prob" (or whatever you named it earlier) in the Workspace Manager and select Duplicate from the context menu.
- ⇒ Rename the new model "License vs. In-house downstream".
- ⇒ In order to add the nodes mentioned earlier, re-arrange your Influence Diagram to give yourself some space between Unit Sales, Revenue per Unit, Marketing Costs and NPV as shown in Figure 5-15.
- ⇒ One way to quickly re-arrange the Influence Diagram is to select multiple nodes at once. While holding down the Ctrl key, click on the Unit Sales, Marketing Costs, Cost per Unit and Product Strategy nodes. All these nodes turn magenta because they are selected as a group.
- ⇒ Use the mouse to drag this group of nodes to the left, freeing up space in the center of the diagram.

You may have to adapt these steps slightly depending on how your diagram is drawn.



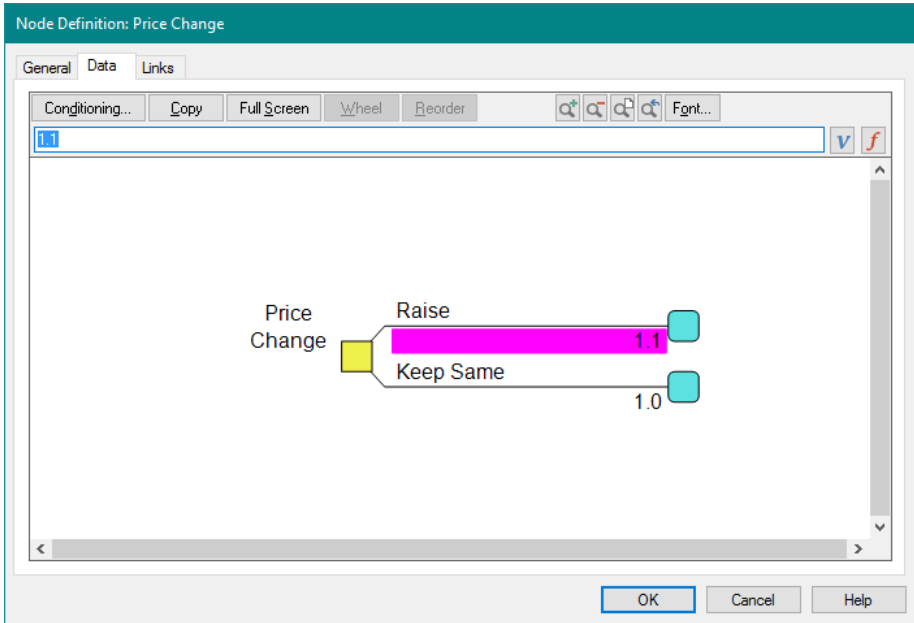
**Figure 5-15. Re-arranged Influence Diagram**

- ⇒ Select Influence Diagram | Node | Add | Decision from the ribbon to add the new decision node.
- ⇒ Click somewhere in the space you created. The Node Definition dialog appears with the General tab selected.
- ⇒ Name the node "Price Change".
- ⇒ Rename the alternatives to "Raise" and "Keep same". The General tab should look like Figure 5-16.



**Figure 5-16. Node Definition Dialog General Tab for Price Change Decision**

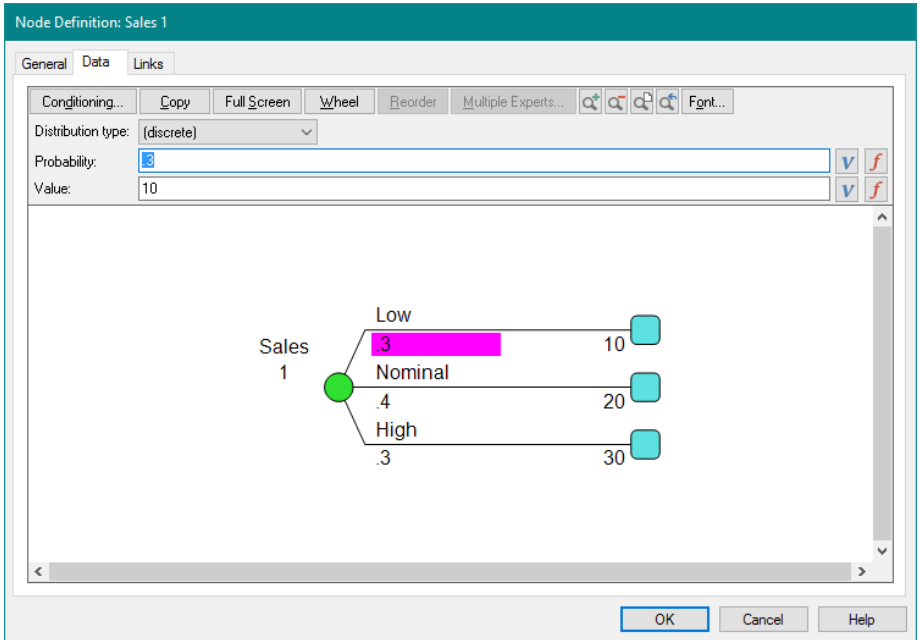
- ⇒ Select the Data tab.
- ⇒ Enter "1.1" for the Raise alternative and "1.0" for Keep same. These numbers will be used as multipliers, so 1.1 means raising prices 10%.
- ⇒ Press Enter. The Data tab should look like Figure 5-17.



**Figure 5-17. Node Definition Data Tab for Price Change Decision**

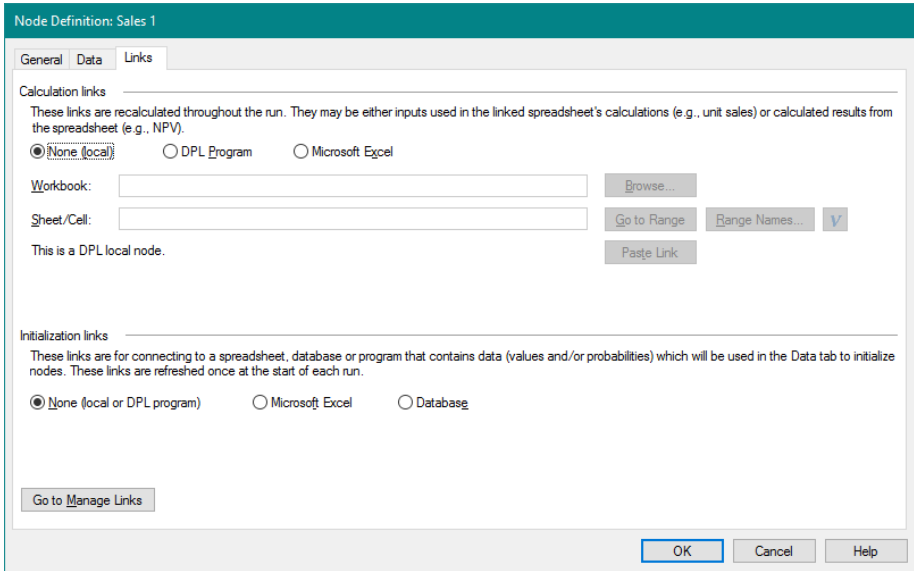
- ⇒ Click OK.
- ⇒ Double-click Unit Sales to edit its definition.
- ⇒ On the General tab, change the name to "Sales 1". Leave the Outcomes as they are.
- ⇒ Change the value for each outcome to half its previous value. The Data tab should look like Figure 5-18 when you are done.





**Figure 5-18. Node Definition Data Tab for Sales 1**

⇒ Switch to the Links tab and under the Calculation links section at the top select the "None (local)" radio button to break this node's link with the spreadsheet (more on this later).



**Figure 5-19. Node Definition Links for Sales 1**

⇒ Click OK to close the Node Definition dialog.

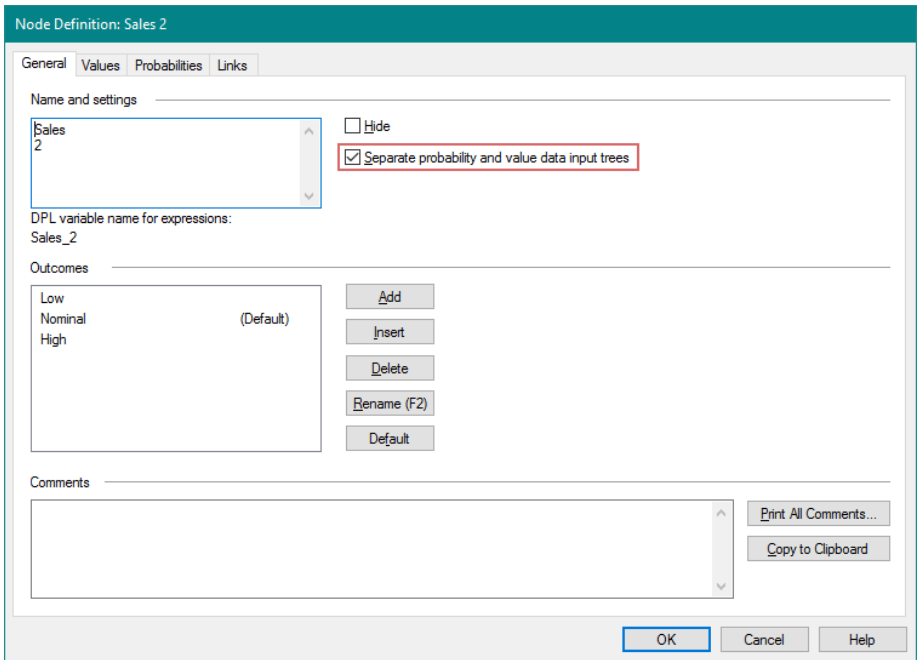
You have divided the sales in half because you are about to introduce a new node called Sales 2. To do this you are going to take advantage of DPL's node copy and paste capability. By the time you are done modifying the model, you will have created a two time period model. Sales 1 will represent sales in the first period and Sales 2 will represent sales in the second period. You have divided the values in half for Sales 1 to reflect the fact that previously Unit Sales represented total sales.

- ⇒ With the Sales 1 node selected (it should be magenta), make a copy of it by pressing Ctrl+C or clicking Home | Edit | Copy.
- ⇒ Paste the new node by pressing Ctrl+V or clicking Home | Edit | Paste.
- ⇒ Select the new node named Sale 1 (copy) and drag it away from Sales 1.

Note: You can also select and copy multiple nodes by holding down Ctrl and selecting the nodes to be copied, as you did earlier when you moved multiple nodes.

- ⇒ Double-click Sales 1 (copy) to edit it.
- ⇒ On the General tab, change its name to "Sales 2".

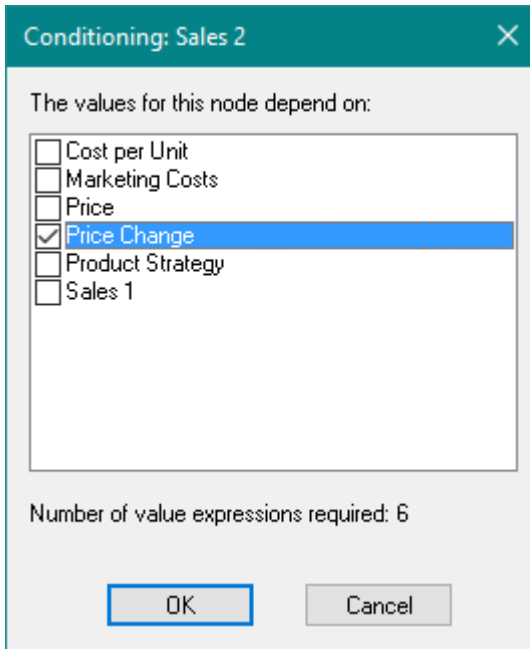
- ⇒ Check Separate probability and value input trees as shown in Figure 5-20.



**Figure 5-20. Node Definition General Tab for Sales 2**

Note that there are now two tabs in place of the Data tab called Values and Probabilities, respectively. This is convenient if you have a discrete chance node whose probabilities are conditioned by one or more nodes and whose values are conditioned by a different set of nodes, or whose probabilities are conditioned but values aren't or vice versa. As you will see in the sample model you are building, the probabilities for Sales 2 will be conditioned by one node while the values will be conditioned by another node.

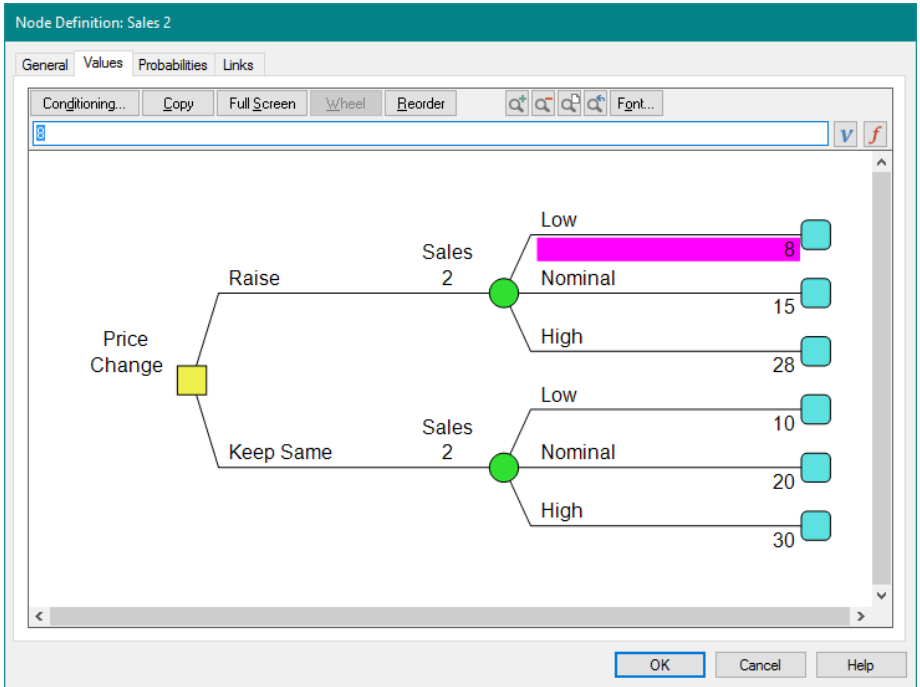
- ⇒ Select the Values tab.
- ⇒ Click Conditioning. The Conditioning dialog appears.
- ⇒ Check Price Change as shown in Figure 5-21.



**Figure 5-21. Conditioning dialog for Sales 2 value**

Note that the Conditioning dialog tells you how many data expressions are required given the nodes selected. Since Sales 2 is a three-outcome discrete chance node and Price Change is a two-alternative decision node, six data expressions are required.

- ⇒ Click OK to close the Conditioning dialog. The data input tree updates on the Values tab to account for the conditioning.
- ⇒ Change the Low, Nominal and High values for the Raise alternative of Price Change to be "8", "15", and "28", respectively. You are assuming that demand (sales) is slightly sensitive to price. Leave the values for the Keep same alternative as is. The data input tree for Values should now look like Figure 5-22.



**Figure 5-22. Node Definition Value Tab After Conditioning**

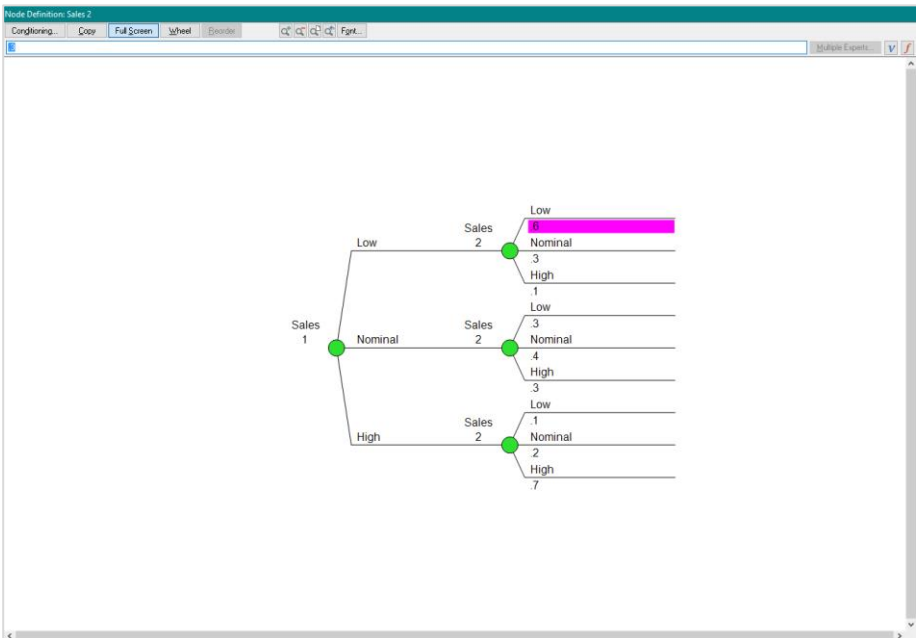
- ⇒ Select the Probabilities tab.
- ⇒ Click Conditioning. The Conditioning dialog appears.
- ⇒ Select Sales 1 and click OK.

As the Conditioning dialog indicated, nine data expressions are required. The probability input tree for this is rather small.

- ⇒ Click the Full Screen button to make entering the data easier. See Figure 5-23.
- ⇒ Enter probabilities for Sales 2 according to Table 5-2.

<b>Sales 1 Outcome</b>	<b>Sales 2 Outcome</b>	<b>Probability</b>
Low	Low	.6
	Nominal	.3
	High	.1
Nominal	Low	.3
	Nominal	.4
	High	.3
High	Low	.1
	Nominal	.2
	High	.7

**Table 5-2. Probability Data for Sales 2**

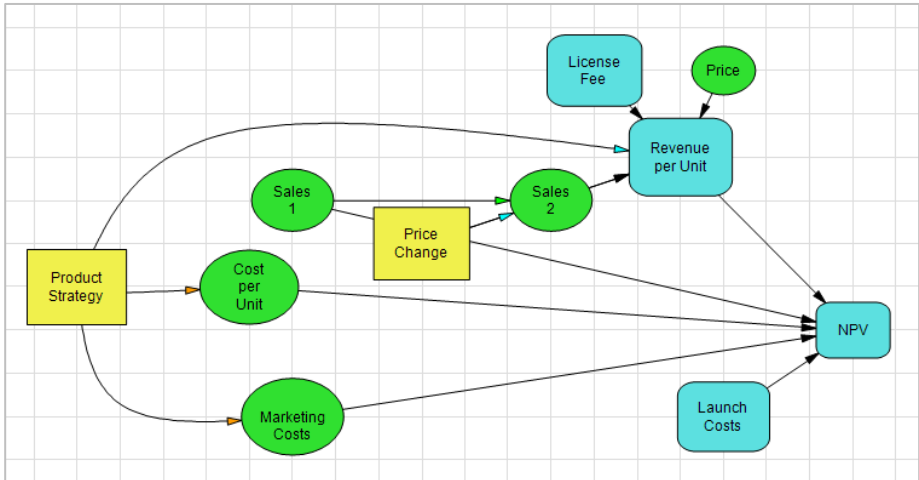


**Figure 5-23. Full Screen Probability Input Tree for Sales 2**

Note that the probabilities for Sales 2 given that Sales 1 is Nominal are unchanged. The probability data in Table 5-2 reflects the belief that if sales in the first period are low (i.e., if the outcome of the Sales 1 chance node is Low) then sales in period two are more likely to be low (i.e., the outcome

of the Sales 2 chance node is more likely to be Low). Similarly if sales in the first period are high they are more likely to be high in the second period.

- ⇒ Click Full Screen again. The Node Definition dialog returns to normal dialog mode.
- ⇒ Click OK. Your model should now look something like Figure 5-24.



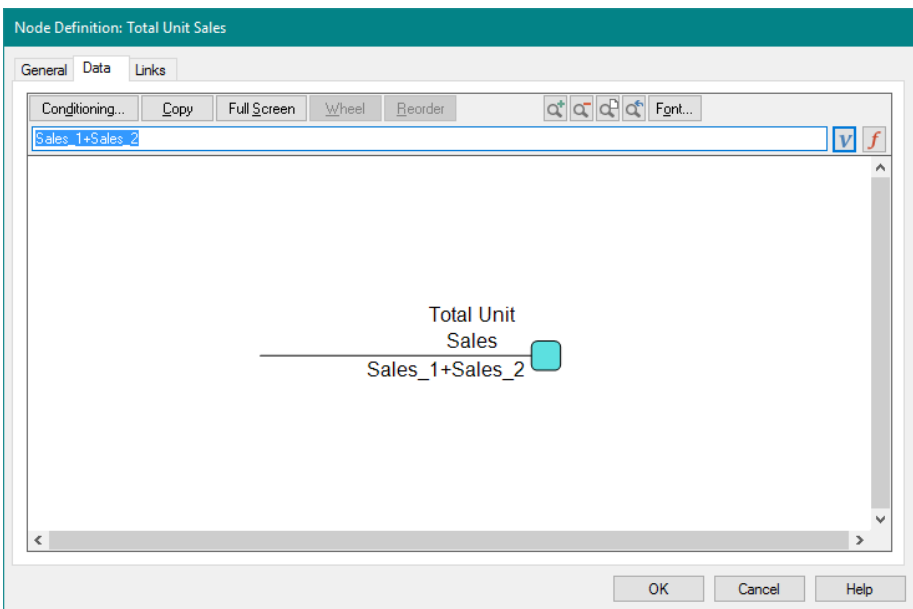
**Figure 5-24. Model After Editing Sales 2**

DPL has indicated the changes that you have just made with colored arcs. A green arc indicates that the influencing node (i.e., Sales 1 – called the predecessor) conditions the probabilities of the influenced node (i.e., Sales 2 – called the successor). As you have seen previously a blue arc indicates that the influencing node (i.e., Price Change) conditions the values of the influenced node (i.e., Sales 2). If the predecessor conditions both the probabilities and the values of the successor, DPL indicates this with a light blue arc. Arc colors are covered in more detail in Section 7.3.

Note that DPL continues to update the Default Tree as you make changes in the Influence Diagram. The Default Tree is not yet correct, but you can ignore that for now. There are several more changes to make to the Influence Diagram.

You need to create a value node to connect to the "Unit\_Sales" driver cell in the spreadsheet. This node will represent total sales in both time periods.

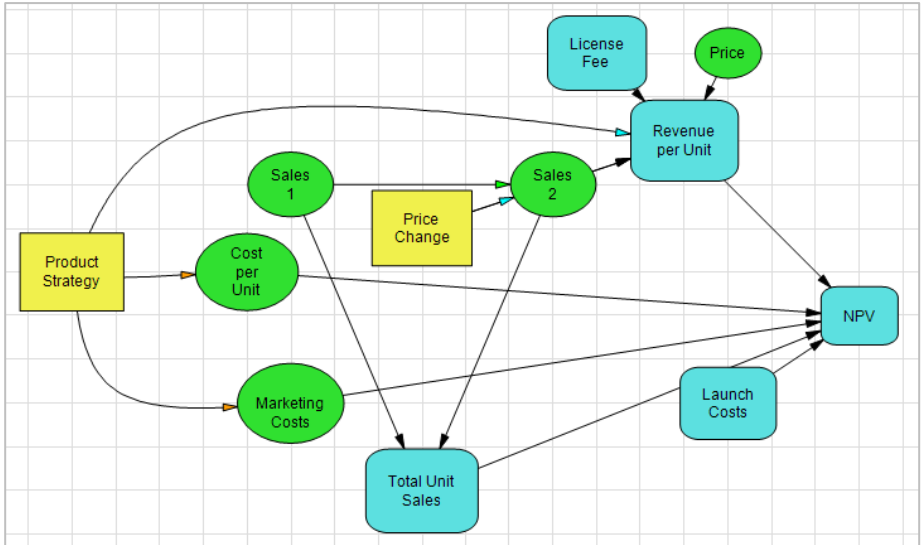
- ⇒ Click the Influence Diagram | Node | Linked Node icon. Note that the default add link node type indicated by the icon is currently Excel-Calculation Linked. If it is not, drop down the split button and select Excel Calculation-Linked... from the list.
- ⇒ In the Range Names dialog, select Unit\_Sales and click OK.
- ⇒ Double-click the new Unit Sales node to edit it.
- ⇒ On the General tab, change the name to "Total Unit Sales".
- ⇒ On the Data tab, enter the formula "Sales\_1 + Sales\_2". Use the variable button to avoid typing mistakes. See Figure 5-25.



**Figure 5-25. Node Definition Data for Total Unit Sales**

- ⇒ Click OK to close the Node Definition dialog.
- ⇒ Move Total Unit Sales to the bottom of the Influence Diagram.
- ⇒ Draw an arc from Sales 1 to Total Unit Sales and Sales 2 to Total Unit Sales.
- ⇒ Delete the arc that goes from Sales 1 to NPV (hint: click near the arrowhead to select it and then press the Delete key).
- ⇒ Draw an arc from Total Unit Sales to NPV. Your model should look similar to Figure 5-26.

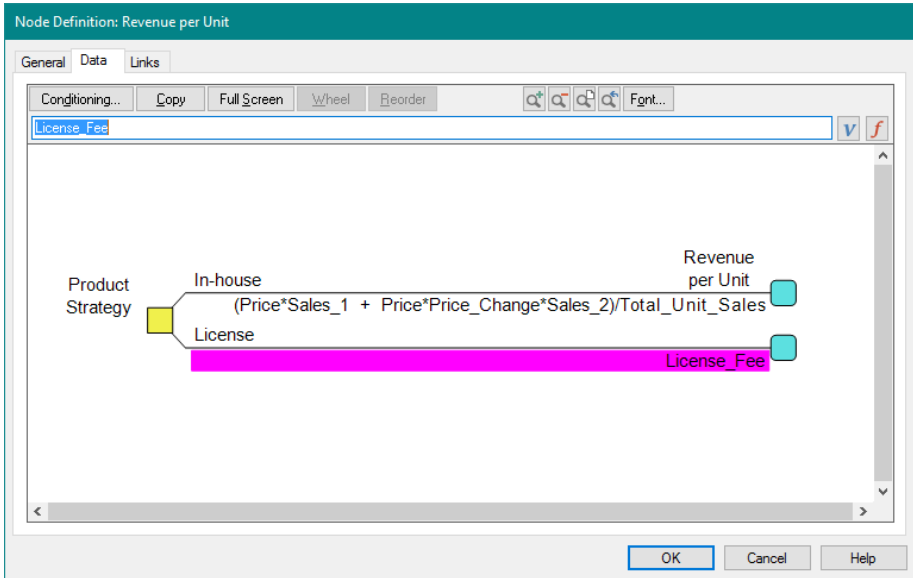




**Figure 5-26. Model with Total Unit Sales**

Now you will update Revenue\_per\_Unit to reflect the price change. The formula for Revenue\_per\_Unit in the In-house case becomes an average selling price.

- ⇒ Double-click Revenue per Unit to edit it.
- ⇒ The In-house formula is "(Price\*Sales\_1 + Price\*Price\_Change\*Sales\_2)/Total\_Unit\_Sales". The License formula doesn't change. Again, use the Variable button for accuracy. See Figure 5-27.
- ⇒ If the data is cut off in the input tree, you can right mouse-click on the tree to get the context menu and select Lengthen Branch.



**Figure 5-27. Node Definition Dialog for Revenue per Unit**

- ⇒ Click OK close the dialog.
- ⇒ Click Influence Diagram | Arc | From Formulas to add arcs from Sales 1, Sales 2, Price Change and Total Unit Sales to Revenue per Unit.

You have now incorporated Sales 2 and Price Change into the expressions for Revenue per Unit. In particular, if In-house is chosen then revenues reflect the price change in the second period.

Your model is nearly complete. However if you were to view the Default Tree DPL is building you'd find that it is not yet correct. The model you are developing is a two time period model. You have an upfront decision to produce the product In-house or License it. You then have a downstream decision to change the price or not. The downstream decision occurs after you have observed sales in the first period. To reflect this, you need to have the Sales 1 discrete chance node appear before (to the left of) Price Change in the Decision Tree.

- ⇒ Drag the splitter bar up from the bottom of the Influence Diagram pane, so that the Influence Diagram and Decision Tree panes are of equal size. Zoom Full within both panes so you can see all of the model in each as shown in Figure 5-28.

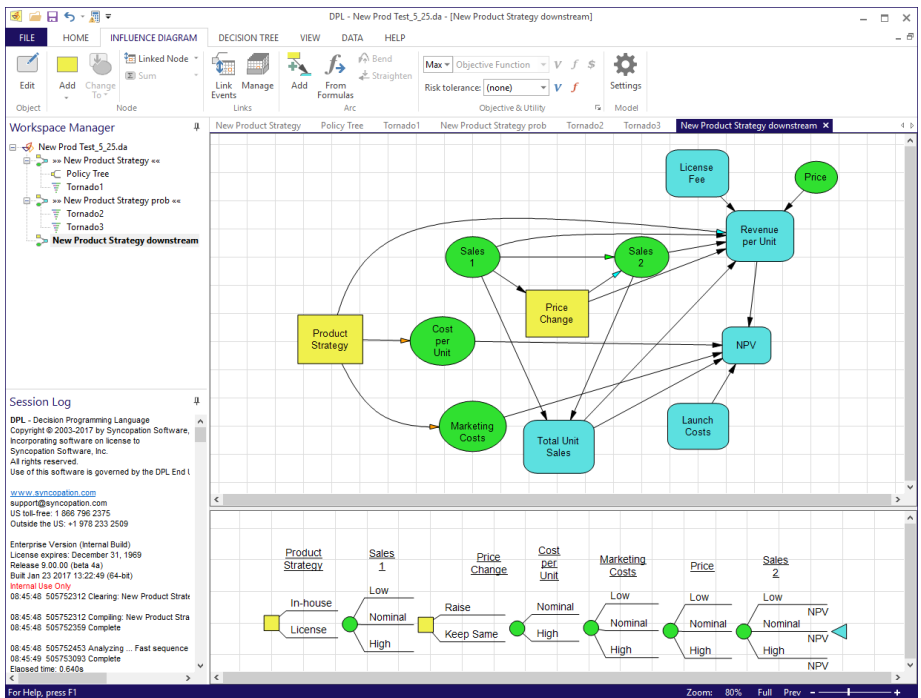
Price Change is currently before Sales in the Default Tree. You will now add a timing arc to the Influence Diagram so that DPL will correct this.

Specifically, you will add an influence arc from Sales 1 to Price Change. As mentioned in Chapter 2, an influence arc can represent dependency, conditioning and/or timing. In this case, the influence arc from Sales to Price Change is a timing arc (Price Change does not depend on and is not conditioned by Sales 1). The arc tells DPL that Sales 1 occurs before Price Change.

- ⇒ Click Influence Diagram | Influence/Arc | Add.
- ⇒ Create an arc from Sales 1 to Price Change.

Note that DPL has moved Sales 1 ahead of Price Change in the Decision Tree.

You may want to move some nodes around again and/or bend some influence arcs to make the Influence Diagram neater. Your model should look something like Figure 5-28. You will now run your updated model.



**Figure 5-28. Completed Model with Downstream Decision**

- ⇒ In the Home | Run group, make sure Policy Tree and Policy Summary are checked.

- ⇒ Click Home | Run | Decision Analysis or press F10. DPL produces the requested outputs.
- ⇒ Review the Policy Tree.

In the revised model, there is a downstream decision. With downstream decisions there is not necessarily a single alternative that is always optimal. Note that in the Policy Tree a different alternative is optimal for the Raise Price decision depending upon the outcome of Sales 1. It is easy to review the relatively small Policy Tree in this example. The Policy Summary provides a useful view of the information in the Policy Tree, particularly for models with larger and/or more complex Policy Trees. Note that the expected value of the model has increased to 111.1.

- ⇒ Double-click on the Policy Summary item in the Workspace Manager to activate the window. The Policy Summary as shown in Figure 5-29 is displayed.



**Figure 5-29. Policy Summary™**

As stated previously, for downstream decisions the Policy Summary displays the probability weighted fraction of scenarios for which each

downstream decision alternative is optimal. In your updated model, Price Change is a downstream decision because it occurs after the Sales 1 discrete chance node. As indicated in Figure 5-29, Raise has a policy dependent probability of 30% while Keep same has a policy dependent probability of 70%. By looking at the Policy Tree again you can determine in which scenarios the Raise vs. Keep same alternative is optimal.

The Policy Summary also displays the policy dependent probabilities of the outcomes for discrete chance nodes. For discrete chance nodes that are not conditioned by others and that do not depend on the optimal policy, the policy dependent probabilities are equal to the input probabilities. This is true of Sales 1, Cost per Unit, Marketing Costs and Price. However, Sales 2 is conditioned by another node (namely Sales 1). You can see that the policy dependent probabilities for Sales 2 are not the same as the input probabilities.

## 5.7 Rainbow Diagram on a Value

The rest of this chapter addresses Rainbow Diagrams, which are another form of sensitivity analysis. A Rainbow Diagram provides more detailed information on how the objective function of the model and the optimal policy change as a single variable is varied across a range of settings. You will now create a Rainbow Diagram for a value node in the model.

- ⇒ Click Home | Sensitivity | Rainbow Diagram (One-Way Rainbow diagram in is the current default indicated by the icon). The Rainbow Diagram Setup dialog appears similar to Figure 5-31 but blank.
- ⇒ Press the Select button to select a variable for the dialog.
- ⇒ Select License\_Fee in the Value for sensitivity drop-down list as shown in Figure 5-30.

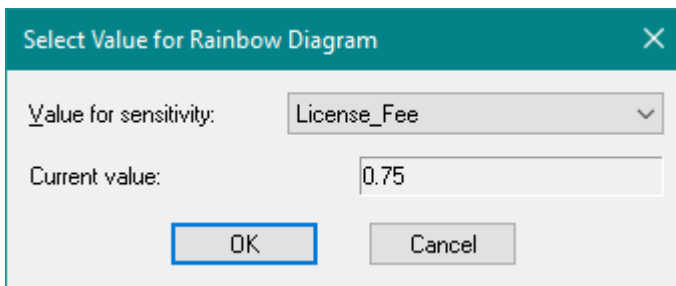


Figure 5-30. Select Value for Rainbow Diagram dialog

- ⇒ Click OK.
- ⇒ In the Run Rainbow Diagram dialog, specify the From: and To: values to be "0.7" and "0.9", respectively.
- ⇒ Set the Step size to be "0.025". The number of steps will update. The Rainbow Diagram Setup dialog should now look like Figure 5-31.

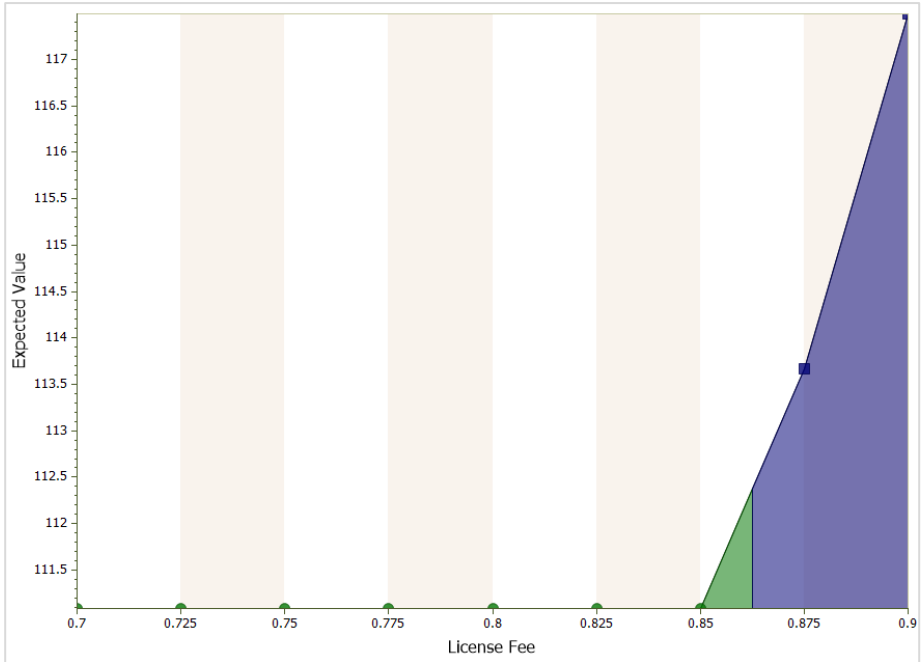
The screenshot shows a dialog box titled "Rainbow Diagram - License vs. In-house downstream". It contains the following fields and controls:

- Variable section:**
  - Variable name: License\_Fee (with a "Select" button)
  - Current value: 0.75
  - From: .7
  - To: .9
  - Number of steps: 9
  - Step size: .025
- Evaluation method section:**
  - Fast sequence evaluation (with a "Change" button)
- Buttons:** OK, Cancel, and Help.

**Figure 5-31. On-way Rainbow Diagram Setup for License Fee**

- ⇒ Click OK to run the Rainbow Diagram.

A Rainbow Diagram displays the value of the model's objective function on the y-axis and the values of the selected variable on the x-axis. The diagram shows how the objective function value changes as the variable setting changes across the specified range. The Rainbow Diagram for License Fee is in Figure 5-32. The diagram indicates that as License Fee increases from 0.7 to 0.85 there is no change in the objective function value, but as License Fee increases from 0.85 to 0.9 there is an increase in the objective function value.



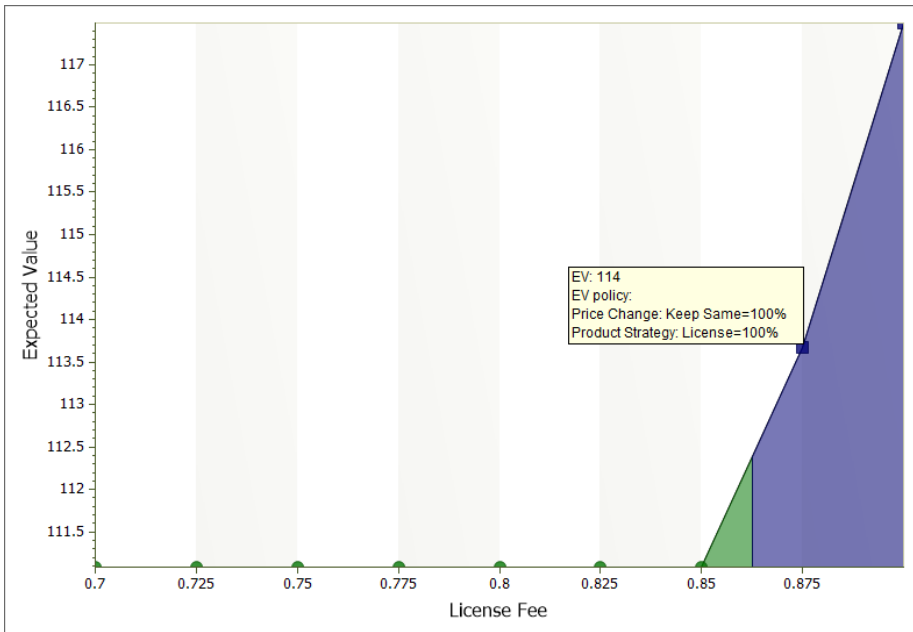
**Figure 5-32. Rainbow Diagram for License Fee**

Rainbow Diagrams also indicate policy changes by changes in color. Since the Rainbow Diagram in Figure 5-32 changes color between 0.85 and 0.875, you know that a policy change occurs in that region. As with Tornado diagrams, the vertical line where the color changes does not indicate the precise point at which the policy changes. The vertical line is drawn at the halfway point of the two tested values of the variable between which a policy change has occurred. In this instance, License Fee is tested at 0.85, which results in one policy, and again at 0.875, which results in a second policy.

Therefore the vertical line is drawn halfway between 0.85 and 0.875. If you want more precise information on where the policy change occurs you can run another Rainbow Diagram with finer increments within this range.

In a Rainbow Diagram, DPL starts with a color and changes to a new color every time there is a policy change. The color of the region does not relate to the value of the variable, i.e., blue does not mean that the high setting of the variable resulted in a policy change. It only indicates that a policy change has occurred. You can use your mouse cursor to find out which policy applies in each colored region if you have Show Tips turned on, i.e. the Show Tips button is shaded blue within the View | Tips group.

- ⇒ Place your mouse cursor over the blue square marker for 0.875 on the curve. A policy tip will appear as indicated in Figure 5-33.



**Figure 5-33. Rainbow Diagram with Policy Tip Showing**

The policy tip tells you the expected value of the model given the value of the variable that the marker represents. For example, the policy tip in Figure 5-33 indicates that the expected value of the model is 114 when License Fee is 0.875. The policy tip also tells you the policy dependent probabilities for each decision in the model given the value of the variable that the marker represents. The decision alternatives are displayed in the policy tip by using the decision node name followed by a colon followed by the decision alternative. E.g., "Product Strategy: License". The policy dependent probability for the alternative follows the equal sign. As Figure 5-33 indicates, the policy when License Fee is 0.875 is to License out production of the product and to keep price the same in time period 2. To keep the policy tip compact, decision alternatives with a policy dependent probability of zero are not displayed.

- ⇒ If you wish, place your mouse cursor over one of the green circular markers on the curve to see what the policy is for these points.

You have completed a Rainbow Diagram on one value in your model. You will now learn how to run a Rainbow Diagram on a probability.



## 5.8 Rainbow Diagram on a Probability

When you ran the Rainbow Diagram in Section 5.7 you may have noticed that the only variables you could select in the Select Value for Rainbow dialog were values. What if you would like to run a sensitivity analysis on a probability? You will need to modify your model slightly to do so.

- ⇒ Double-click on the item for the "License vs. In house downstream" model in the Workspace Manager. The Model Window becomes active.
- ⇒ Add a value node to the model by clicking Influence Diagram | Node | Add. If the default add node type isn't value (as indicated by the icon), use the drop-down list to select Value.
- ⇒ Click anywhere in the Influence Diagram. The Node Definition dialog comes up with the General tab selected.
- ⇒ Name the value "p".
- ⇒ Click the Data tab.
- ⇒ Type "0.6" as shown in Figure 5-34.

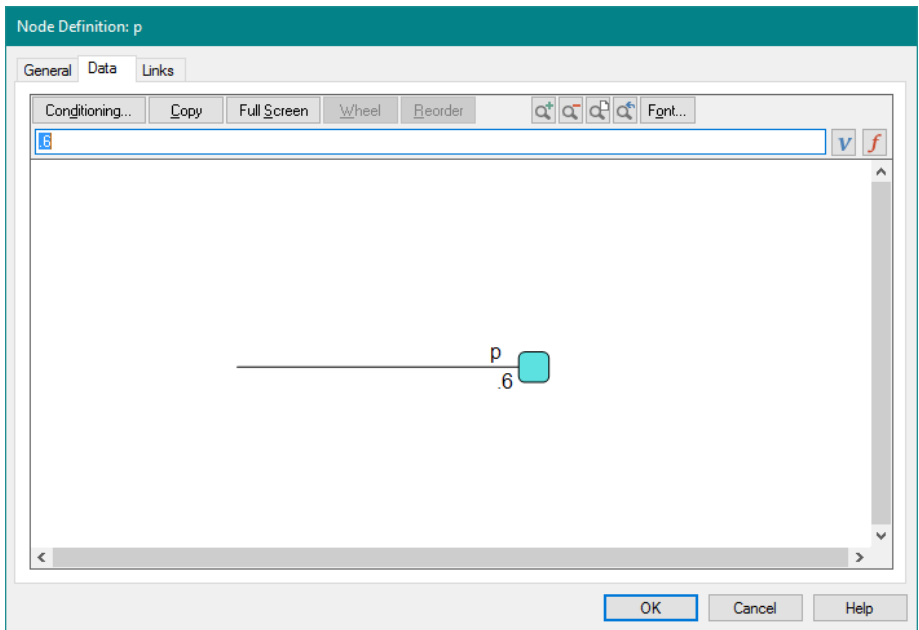
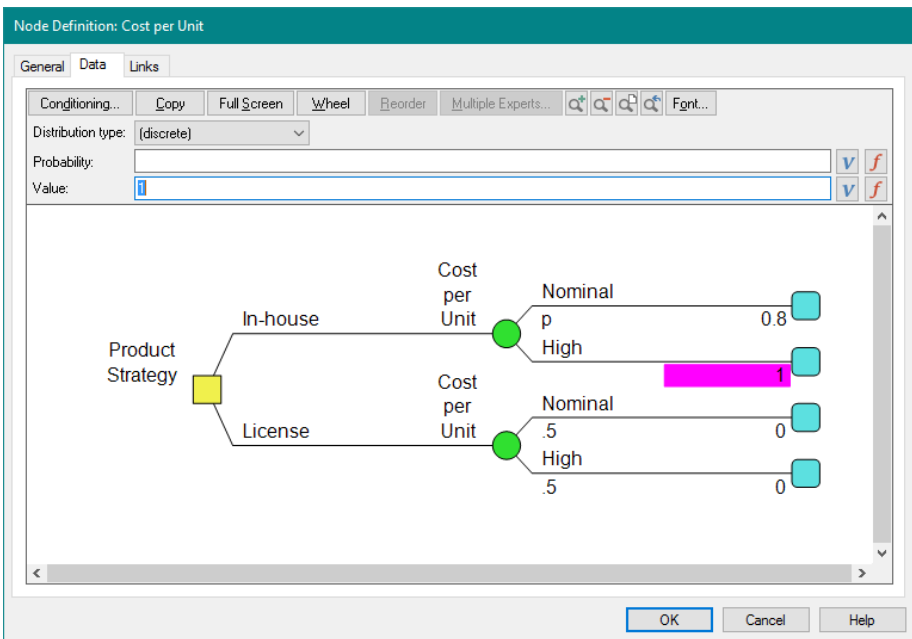


Figure 5-34. Node Definition Dialog Data Tab for p

- ⇒ Click OK.
- ⇒ Double-click the Cost per Unit node to edit it. The Node Definition dialog appears with the Data tab selected.
- ⇒ In place of "0.6" for the probability of Nominal (given In-house), type "p".
- ⇒ Press the down arrow twice to select the probability for High.
- ⇒ Delete the "0.4" that is there for the Probability of High.
- ⇒ Press Enter. Your node data for Cost per Unit should now look like Figure 5-35.

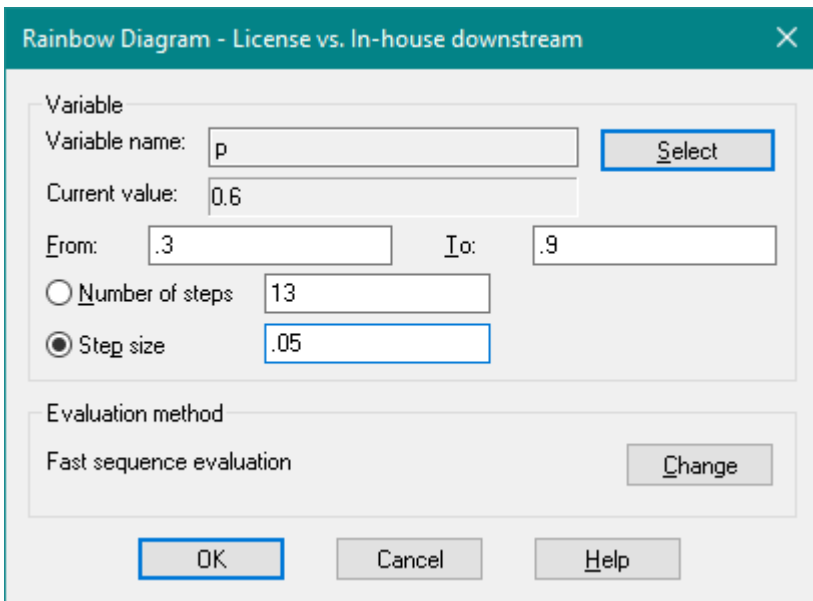


**Figure 5-35. Node Definition Data Tab for Cost per Unit**

Note: in the changes you just made you left the probability of the last outcome of the discrete chance node blank. You can always do this. DPL will assign one minus the sum of the rest of the outcome probabilities to the last outcome. In this instance, you must do this. Rather than entering a number for the probability of Nominal, you have now entered a value. The use of a value such as "p" in the probability expression of an outcome is called a non-constant probability expression. When you use a non-constant probability expression for any of the chance outcomes in a node, you must

leave the last probability expression blank. This helps to ensure that the probabilities of all the outcomes sum to one.

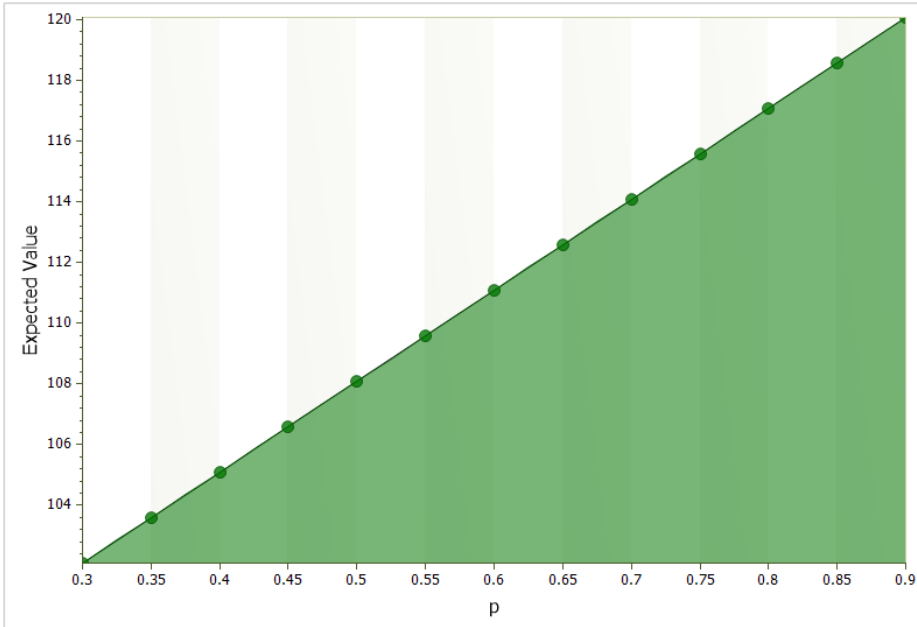
- ⇒ Click OK to close the Node Definition dialog.
- ⇒ Click Home | Sensitivity | Rainbow Diagram. The Run Rainbow Diagram dialog appears.
- ⇒ Click the Select button.
- ⇒ Select p in the Select Value for Rainbow dialog.
- ⇒ Click OK.
- ⇒ In the Run Rainbow Diagram dialog, specify the From: and To: values to be "0.3" and "0.9", respectively.
- ⇒ Set the Step size to be "0.05". The Run Rainbow Diagram dialog should now look like Figure 5-36.
- ⇒ Click OK to run the Rainbow Diagram.



**Figure 5-36. Rainbow Diagram Setup Dialog for p**

DPL produces the Rainbow Diagram for p as shown in Figure 5-37. The Rainbow Diagram indicates that while the objective function increases from approximately 102 to 120, no policy changes occur as p varies from 0.3 to 0.9. In this instance, you can be comfortable with your estimation of the probability of Nominal for Cost per Unit. Over a fairly wide range, it does

not change the optimal strategy. Note, however, that the Rainbow Diagram in Figure 5-37 still assumes that the value for Cost per Unit is no lower than 0.8 and no higher than 1.0 (the Nominal and High values of the outcomes). The Rainbow Diagram tests the sensitivity of the optimal policy to the probability that Cost per Unit is Nominal or High given the specified values of the outcomes.



**Figure 5-37. Rainbow Diagram for p**

The next section discusses Probabilistic Base Case Tornadoes, an additional sensitivity type.

## 5.9 Probabilistic Base Case Tornado Diagram

A Probabilistic Base Case Tornado Diagram is similar to a Base Case Tornado Diagram. However, the Probabilistic Base Case Tornado does not remove the uncertainty from the model when calculating results. To calculate the base run for the output, DPL evaluates the full model with all uncertainty as if it were running a Decision Analysis on the model. Consequently, the base run, or the vertical line in the tornado, will equal the expected value of the model as in a Risk Profile or a Policy Tree. See Sections 3.1 and 3.2 for descriptions of these outputs. To produce the

results for each chance node in the model, DPL runs the model with the chance node it is evaluating fixed to a low state while allowing all the remaining chance nodes to vary across their outcomes. DPL then repeats this with the chance node fixed at a high state and repeats the whole process for each chance node. Because the base run in the Probabilistic Base Case Tornado matches the expected value of the model in a Decision Analysis run, it can be more easily compared to other outputs from a Decision Analysis run and sometimes easier to explain.

The process for setting up the Probabilistic Base Case Tornado is identical to that of the Base Case.

⇒ Drop-down the Home | Sensitivity | Tornado split button and choose Probabilistic Base Case from the list.

The Probabilistic Base Case Tornado Setup dialog appears. The set-up is identical to that of the Base Case Setup Dialog in Figure 5-12.

⇒ Click OK to run the Probabilistic Base Case Tornado Diagram.

DPL produces the Probabilistic Base Case Tornado Diagram as shown in Figure 5-38.

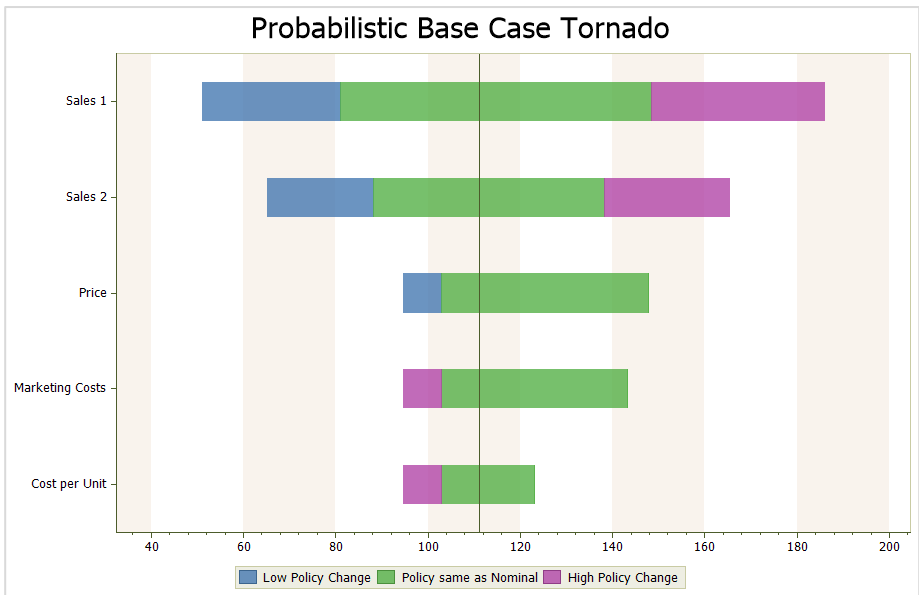


Figure 5-38. Probabilistic Base Case Tornado Diagram

The tornado indicates that all variables are decision sensitive (there is at least one policy change for each bar) and that Sales 1 and Sales 2 have the biggest impact on the objective function. Notice that the vertical line is at 111.1, which is the expected value of the objective function of the model as currently built.

You have now reached the end of the tutorials on building and analyzing DPL decision analysis models. Though the model you developed in this tutorial is fairly simple, you now have the essential skills needed to develop models of decisions of varying complexity.

## 6. Building and Analyzing Monte Carlo Simulation Models

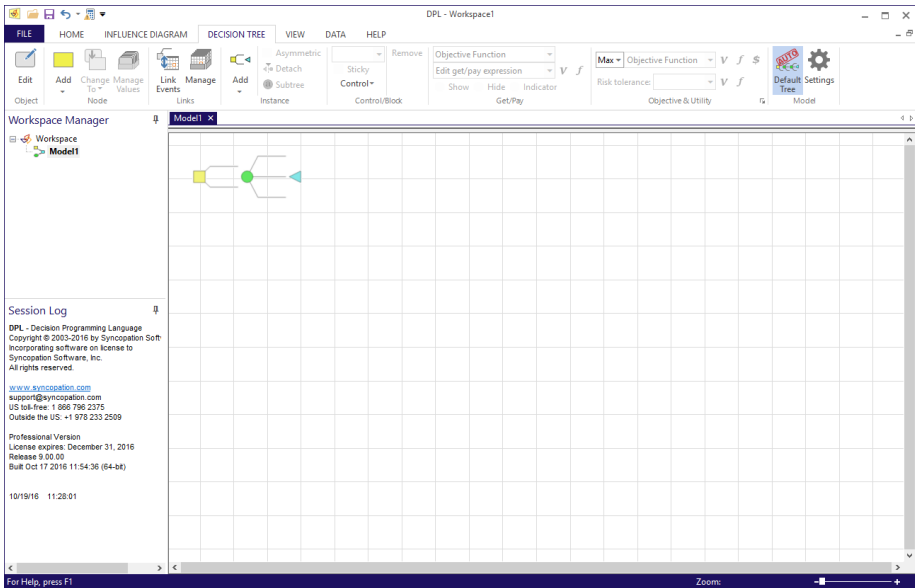
This tutorial focuses on building an Excel-linked DPL model for a financial risk analysis application. It assumes familiarity with cash flow spreadsheets and with the analysis of uncertainty by Monte Carlo simulation. This chapter discusses how to do Monte Carlo simulation in DPL.

⇒ If needed, start DPL.

The left-hand pane is the Workspace Window, and contains the Workspace Manager and Session Log. For more on the Workspace Window, see Section 1.4.2. DPL will load with an empty Decision Tree pane maximized within the Model Window on the right-hand side of the screen as shown in Figure 6-1. This indicates that you are in Decision Tree-focused modeling mode. For on the Model Window and modeling modes, see Section 1.4.3.

The Model Window always contains two panes: 1) the Influence Diagram pane and 2) the Decision Tree pane – though both are not necessarily visible at once. When one pane is maximized, the other is present but minimized and, consequently, hidden from view.

At the moment, Model1 is the only document in your Workspace.



**Figure 6-1. A Blank DPL Workspace in Decision Tree-focused Mode**

The DPL Model you are about to build will be used to characterize the uncertainties of the decision-problem and a linked Excel spreadsheet model will calculate cash flows. The risk analysis model built in this tutorial lends itself to Influence Diagram-focused modeling mode.

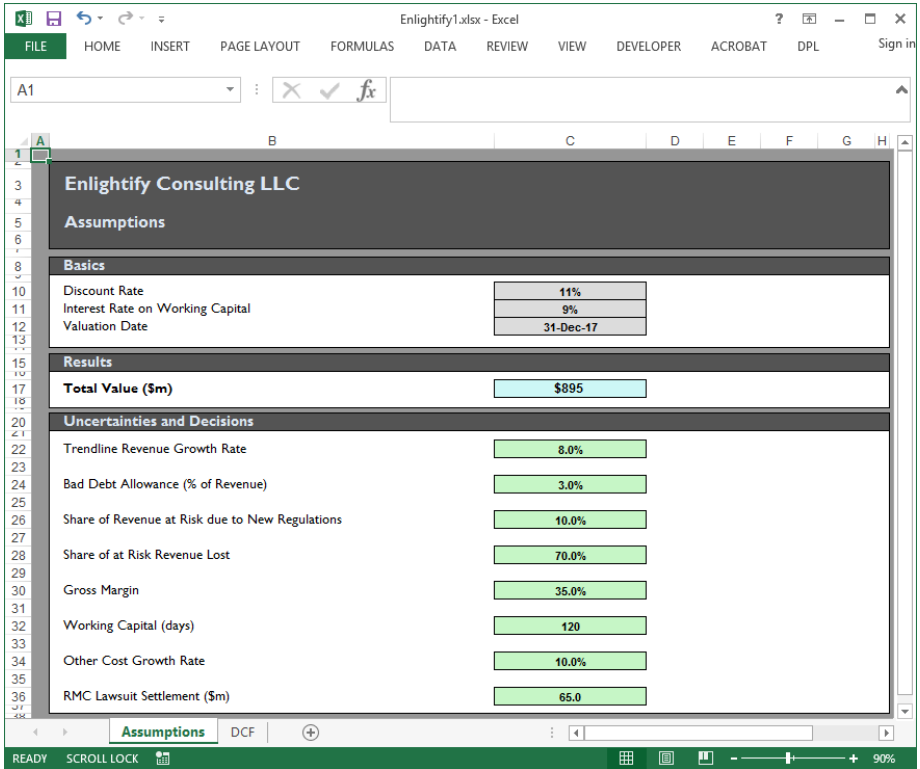
## 6.1 The Excel Cash Flow Model

Before creating the DPL model, take a brief tour of the Excel model provided for this tutorial.

- ⇒ Start Excel as you normally do.
- ⇒ Open the file "Enlightify1.xlsx". This file is found in the "Examples" folder where DPL was installed, usually C:\Program Files\Syncopation\DPL9\Examples.

The directory may vary slightly depending on your license type and operating system.

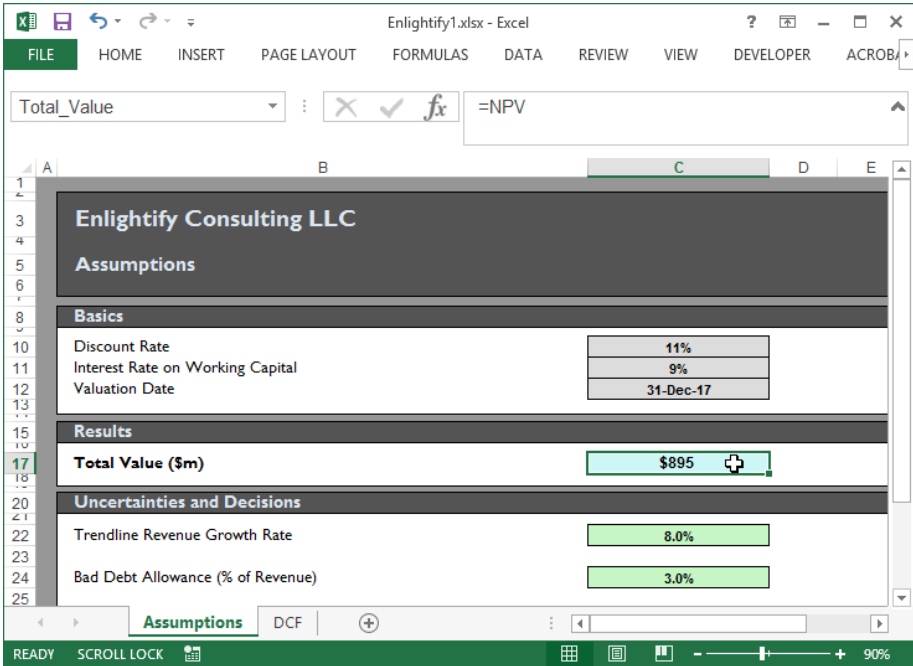




**Figure 6-2. Excel Cash Flow Model for Enlightify: Assumptions Sheet**

This simple cash flow model calculates the value of Enlightify Consulting LLC, a troubled business unit whose financial performance is of concern. The Excel model is built on two sheets, one for key assumptions and value drivers, and another for the cash flow calculations.

In Figure 6-2, you can see that eight of the value drivers have already been identified as sources of uncertainty; the cells are shaded in light green. Each of these cells has been assigned a range name. When using an Excel model with DPL, you must name the cells you intend to link to DPL. If you select a named cell in Excel, the name will appear in the name combo box to the left of the formula bar. In Figure 6-3 you can see that cell C17 has a current value of \$895, and has been given the name "Total\_Value". Please consult your Excel documentation if you are not familiar with named ranges, as they are essential for using DPL with Excel.



**Figure 6-3. Cell C17 is Named Total\_Value**

The calculations sheet (DCF) is shown in Figure 6-4. In this sheet, the model calculates the total value of the Enlightify Consulting business unit by the discounted cash flow method. These calculations are connected to the input cells on the Assumptions sheet. For example, if the Trendline Revenue Growth Rate (cell C22) is changed from 8% to 5%, the total value of Enlightify falls from \$895 million to \$690 million. You may want to spend a few minutes getting familiar with the spreadsheet. Be sure to undo any changes made to the spreadsheet before proceeding.

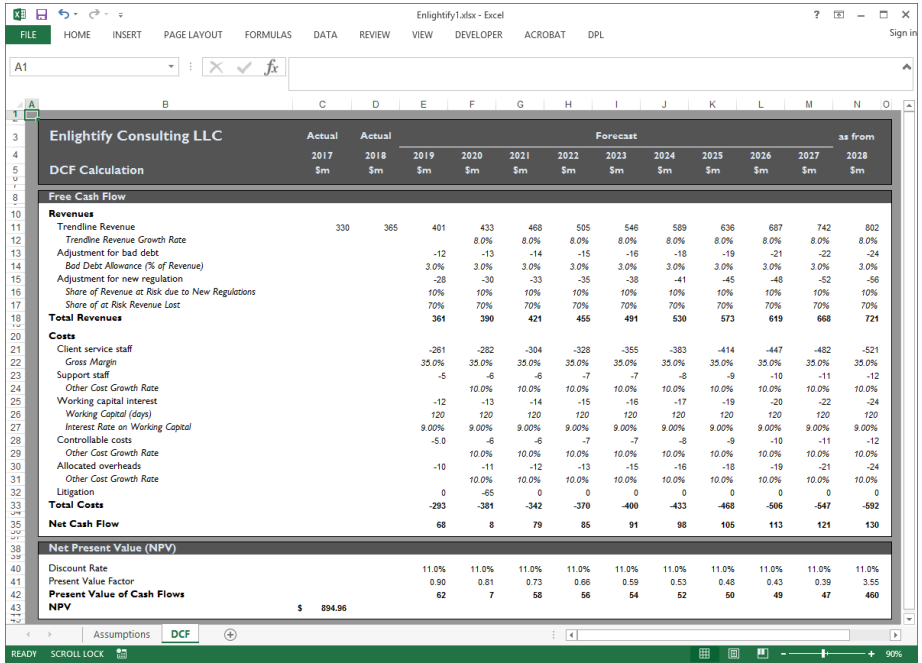


Figure 6-4. Cash Flow Model: DCF Calculations

## 6.2 Creating Linked Values

⇒ Switch back to DPL.

To begin building a DPL model linked to the Enlightify spreadsheet, you'll first need to create and add some linked values to the Influence Diagram. You'll first switch to Influence Diagram-focused modeling mode. To do so:

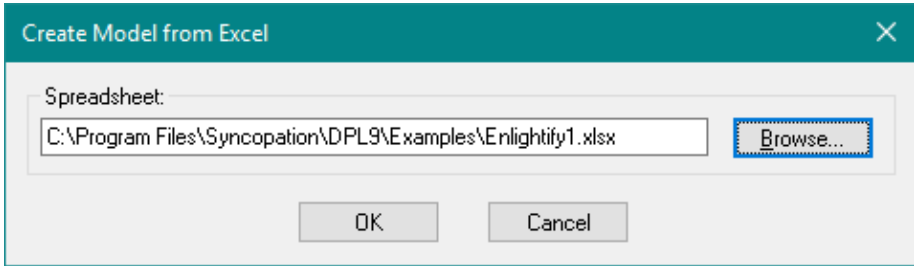
⇒ Press Tab. This activates and maximizes the Influence Diagram pane and makes the Influence Diagram ribbon tab active.

If you wish to have all new models start in this mode, go to File | Options. On the General tab the settings dialog, select the Influence Diagram radio button under *For new models, maximize pane* setting.

⇒ Click the Home | Add to WS | Excel Linked Model...

⇒ In the Create Model from Excel dialog, click the Browse... button and browse to find "Enlightify1.xlsx". See Figure 6-5.

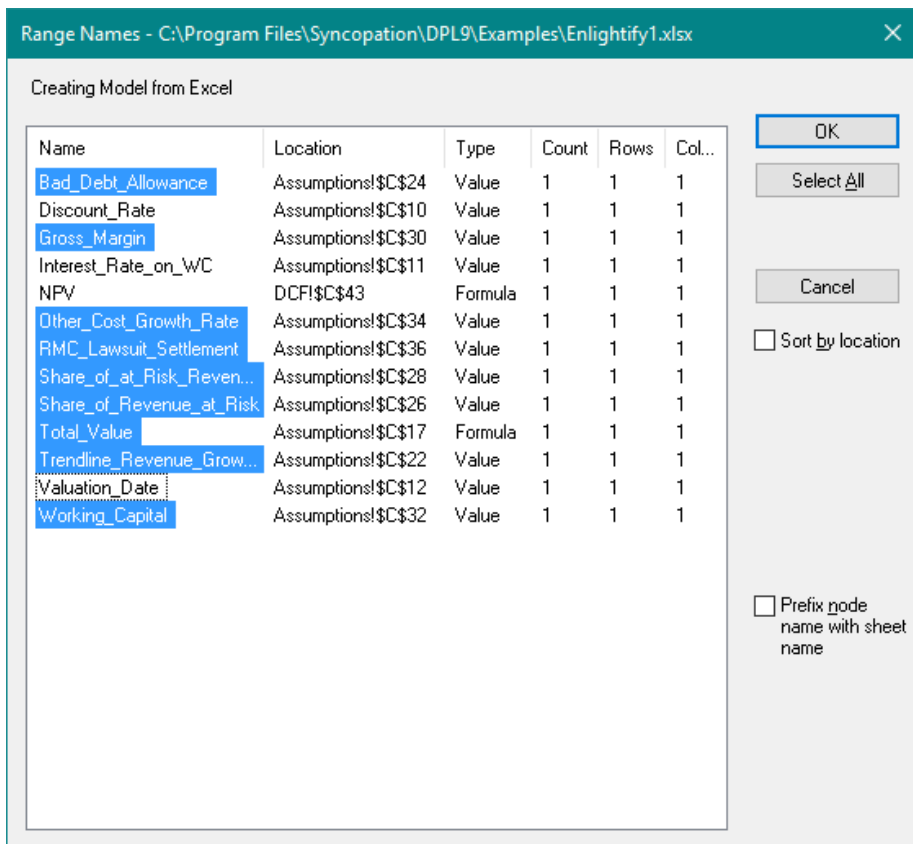
⇒ Click OK to continue.



**Figure 6-5. Create Model from Excel dialog**

DPL will display Range Names dialog which lists all the named ranges in Enlightenment1.xlsx spreadsheet that are suitable for linking. You want to create value nodes for all of these cells except Discount\_Rate, Interest\_Rate\_on\_WC, NPV and Valuation\_Date.

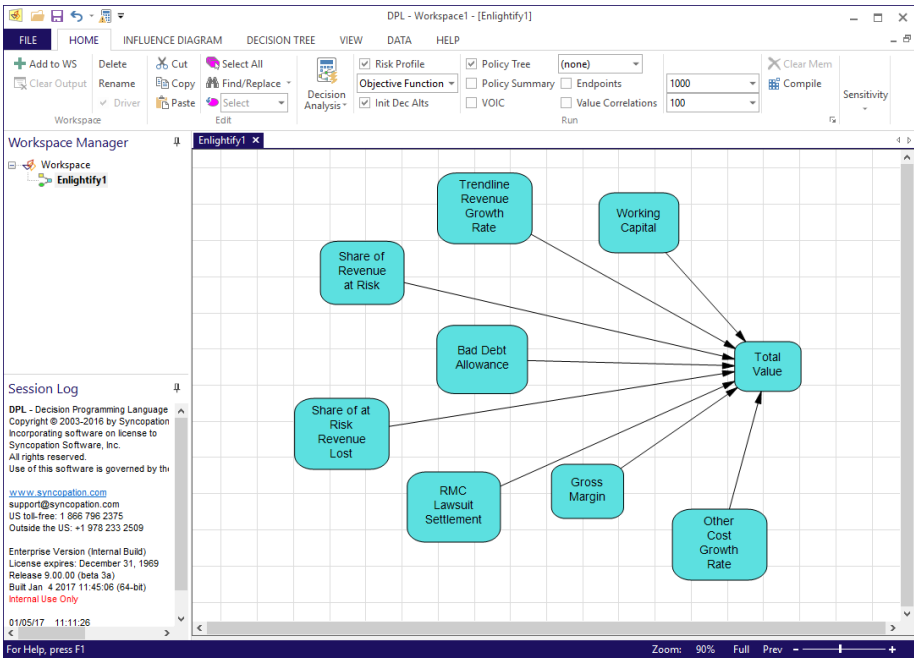
- ⇒ Select the cells as shown in Figure 6-6. To select or de-select cells in the list, you need to hold down the Ctrl key while clicking each one.



**Figure 6-6. Range Name dialog with Values Selected**

⇒ Click OK.

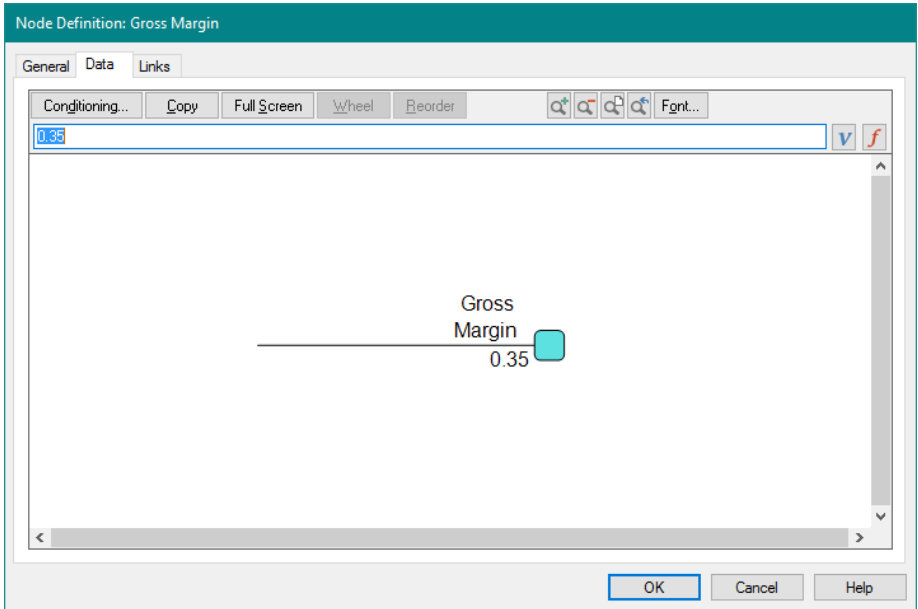
DPL will create a value node for each of the selected cells. A value node is a blue rounded rectangle representing a constant or calculated quantity. DPL has also added arcs based on the relationships in your spreadsheet. Lastly, DPL renames the model in the Workspace Manager to match the spreadsheet name, *Enlightify1*. See Figure 6-7.



**Figure 6-7. Value Nodes Created from Enlightenment1.xlsx Named Cells**

Next, you'll look at how DPL has established the links with Excel and populated the node's data.

⇒ Double-click on Gross Margin. The Data tab of the Node Definition dialog will open.



**Figure 6-8. Node Definition Data for Gross Margin**

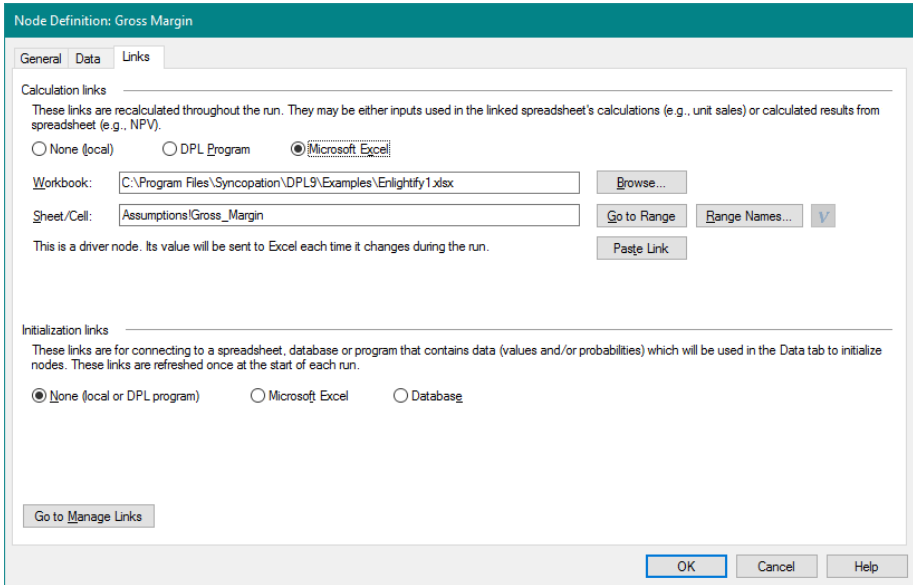
In Figure 6-8, you can see that DPL has brought over the data (0.35 or 35%) from the linked cell named "Gross\_Margin".

⇒ Click on the Links tab.

In Figure 6-9, you can see that DPL has recorded the workbook path and file name, along with the sheet name and cell name to which the Gross Margin node is linked.

DPL also tells you that this is a driver node. When spreadsheet drivers are linked to DPL nodes, they are referred to as driver nodes. Whereas spreadsheet output metrics (e.g., NPV) are linked to metric nodes. Driver nodes are typically linked to cells containing data, while metric nodes are typically linked to cells containing formulas.

If you are unsure whether a DPL node is linked to the correct cell, you can have DPL activate the cell in Excel for you.



**Figure 6-9. Node Definition Links for Gross Margin**

- ⇒ Click the Go to Range button. This activates Excel and selects the linked cell (C30 or Gross\_Margin in this case).
- ⇒ Switch back to DPL.
- ⇒ Click Cancel to close the dialog.

At this point, you've put enough effort into building your DPL model that you should save it.

- ⇒ Select File | Save As and give your Workspace a name (such as Enlightify.da).

## 6.3 Running a Value Tornado Diagram

A Tornado Diagram is typically the first analysis done on a new model. At this point, your model is deterministic (consists only of value nodes), so the Value Tornado Diagram is the only tornado analysis available.

- ⇒ Value Tornado is the default tornado type indicated by the Tornado split button icon. To run a Value Tornado click Home | Sensitivity | Tornado or press F8.



- ⇒ DPL displays a blank Value Tornado setup dialog, similar to Figure 6-10 but without the values filled in for the low and high inputs.

The Value Tornado Setup dialog provides a table in which to enter the low and high settings (range) for each value node that is a constant in the current model. DPL has also filled in the Current setting for each variable in the table.

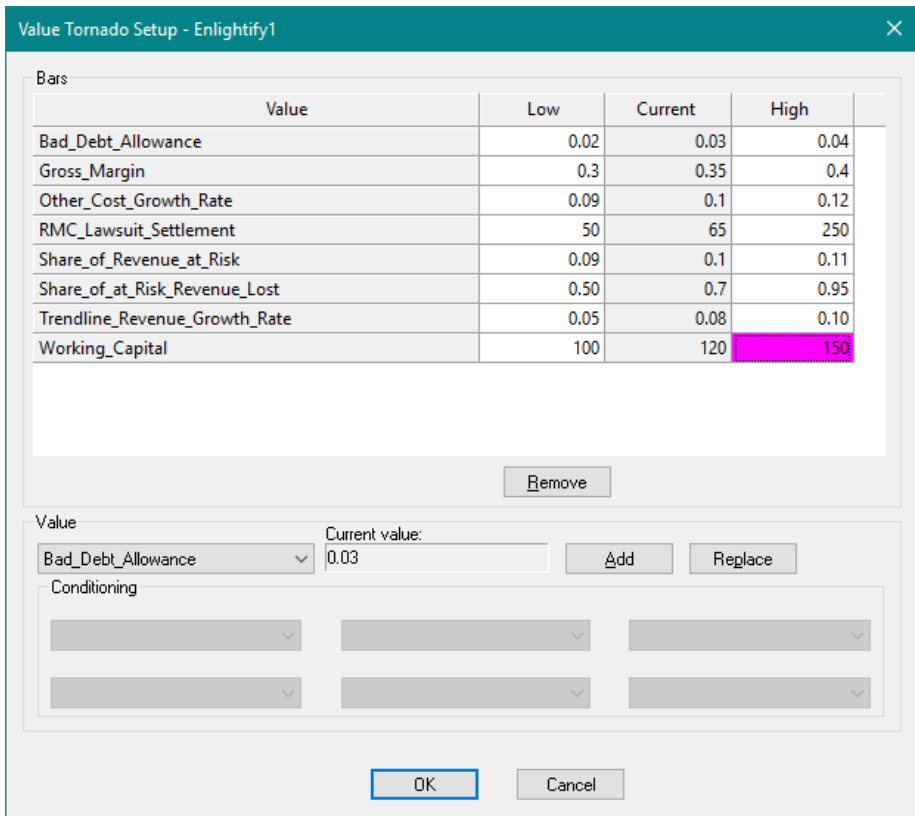
You will now enter the range for each of the value nodes shown in Table 6-1.

- ⇒ Click the blank cell in the Low column of the first value node, Bad Debt Allowance.
- ⇒ Specify 0.02 for the Low value of Bad Debt Allowance.
- ⇒ Specify 0.04 for the High value of Bad Debt Allowance.
- ⇒ Continue to fill in the Value Tornado Setup table with the rest of the High and Low values shown in Table 6-1. Note that you can use the arrow keys to move around the Value Tornado Setup table.

<b><u>Name</u></b>	<b><u>Low</u></b>	<b><u>High</u></b>
Gross Margin	0.30	0.40
Other Cost Growth Rate	0.09	0.12
RMC_Lawsuit_Settlement	50	250
Share of Revenue at Risk	0.09	0.11
Share of at Risk Revenue Lost	0.50	0.95
Trendline Revenue Growth Rate	0.05	0.10
Working Capital	100	150

**Table 6-1. Data for Value Tornado Diagram**

Your completed Value Tornado Setup dialog should look like Figure 6-10.



**Figure 6-10. Completed Value Tornado Setup Dialog**

⇒ Click OK.

DPL displays a Value Tornado diagram (Figure 6-11) for the eight value nodes tested. The bars are sorted so the widest one (corresponding to the highest impact value) is at the top. This diagram will help decide which value drivers need to be modeled probabilistically (as chance nodes). See Section 5.1 for more information on Value Tornadoes.

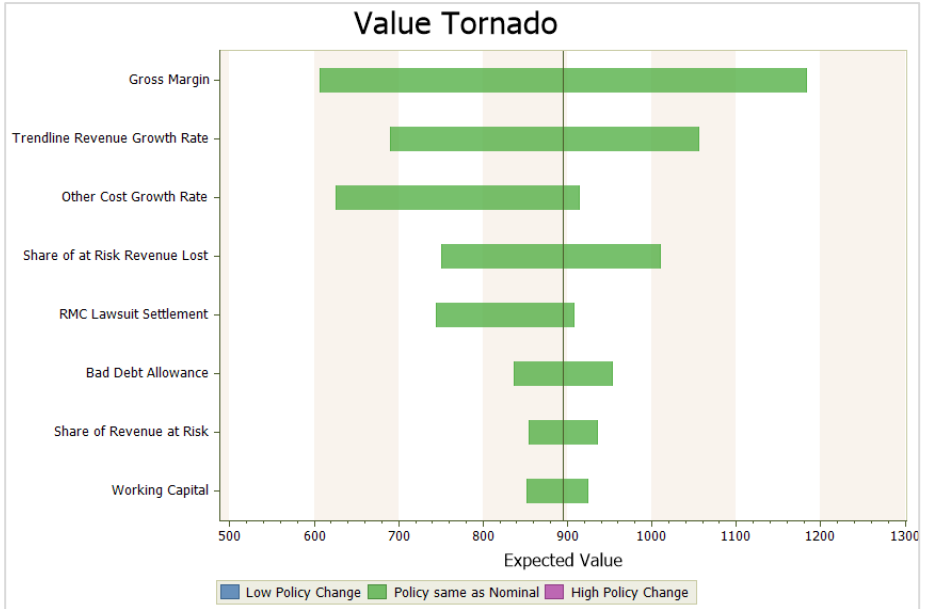


Figure 6-11. Value Tornado Diagram

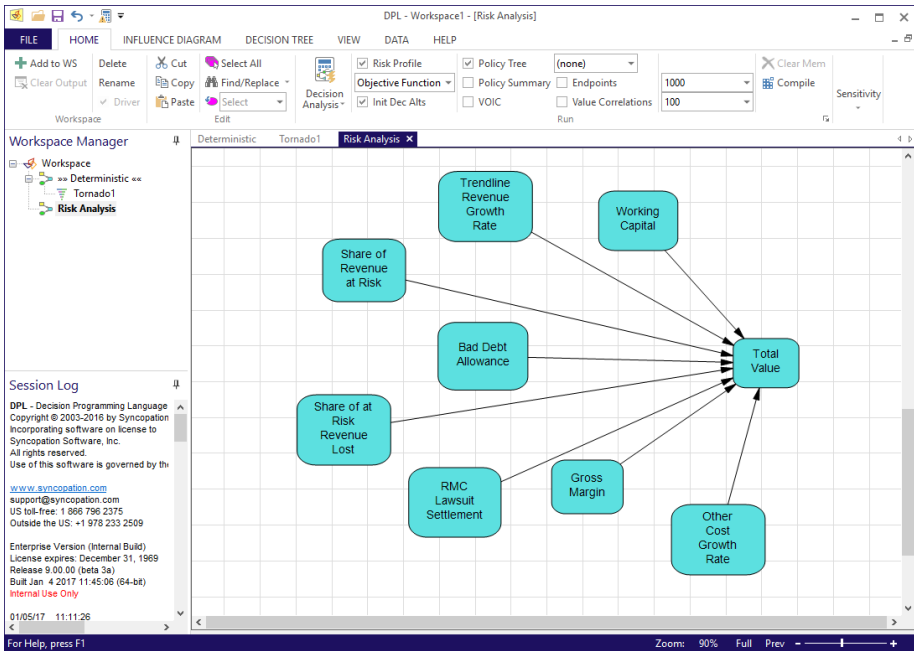
## 6.4 Using the Workspace Manager

DPL allows you to store any number of Models in a Workspace. Before continuing, you'll make a copy of your model as it now stands, so that you'll have it if you need to update the tornado diagram later. Most actions in the Workspace Manager Window are accomplished by right-clicking an item and choosing a command from the context menu. First, you will make a copy of the current model.

- ⇒ In the Workspace Manager window, right-click on "Enlightify1" and select Duplicate.

DPL makes an identical copy of Enlightify1 and names it Enlightify 1 - copy.

- ⇒ Rename the first model to be "Deterministic" by right-clicking on it and selecting Rename.
- ⇒ Rename the second model "Risk Analysis" in the same way.
- ⇒ Activate the Risk Analysis model. Your Workspace should look like Figure 6-12.



**Figure 6-12. Workspace with Two Models**

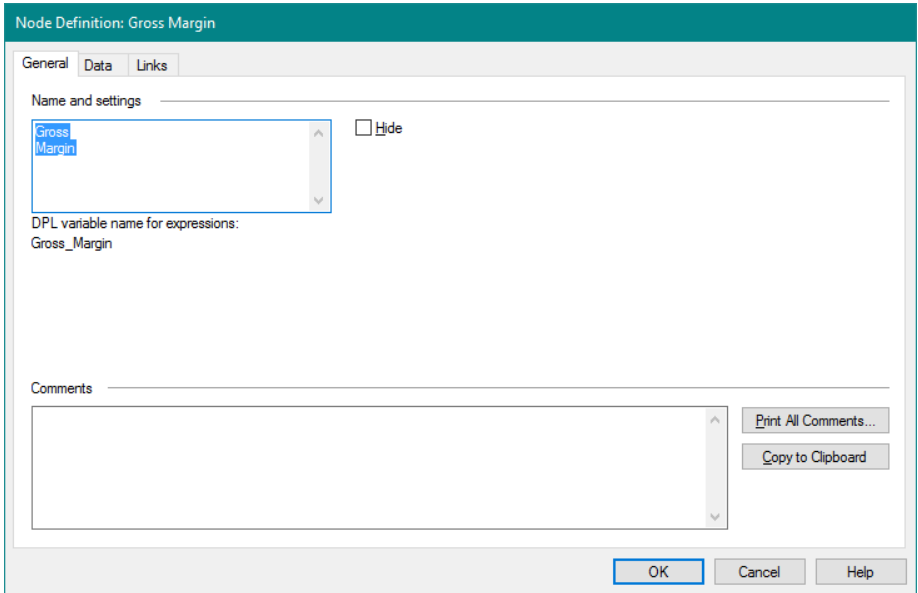
You may notice that the tornado diagram is under Deterministic in the hierarchy -- DPL groups outputs with the models they came from. It's good practice to save your Workspace from time to time.

⇒ Select File | Save.

## 6.5 Continuous Chance Nodes

The tornado diagram indicates which values have the highest impact on Total Value. The next step in the risk analysis is to model the uncertainty in those factors. You'll start with Gross Margin, since it was at the top of the tornado.

⇒ Right-click on Gross Margin, drop-down the Influence Diagram | Node | Change To, and select Continuous Chance from the list. The Node Definition dialog appears as shown in Figure 6-13.



**Figure 6-13. Node Definition Dialog**

Gross Margin is now a chance node, meaning that it is uncertain. DPL has two types of chance nodes, discrete and continuous. Discrete nodes are used for uncertainties which have a finite number of outcomes, such as single events (e.g., regulatory approval) or fixed scenarios (Low, Base Case, High). Continuous nodes can have infinitely many outcomes sampled from named probability distributions (normal, triangular, poisson, etc.) In this chapter, you'll be using continuous chance nodes. DPL draws discrete chance nodes in bright green and continuous chance nodes in dark green.

Next, you'll need to edit the data for Gross Margin.

⇒ Click on the Data tab of the Node Definition Dialog.

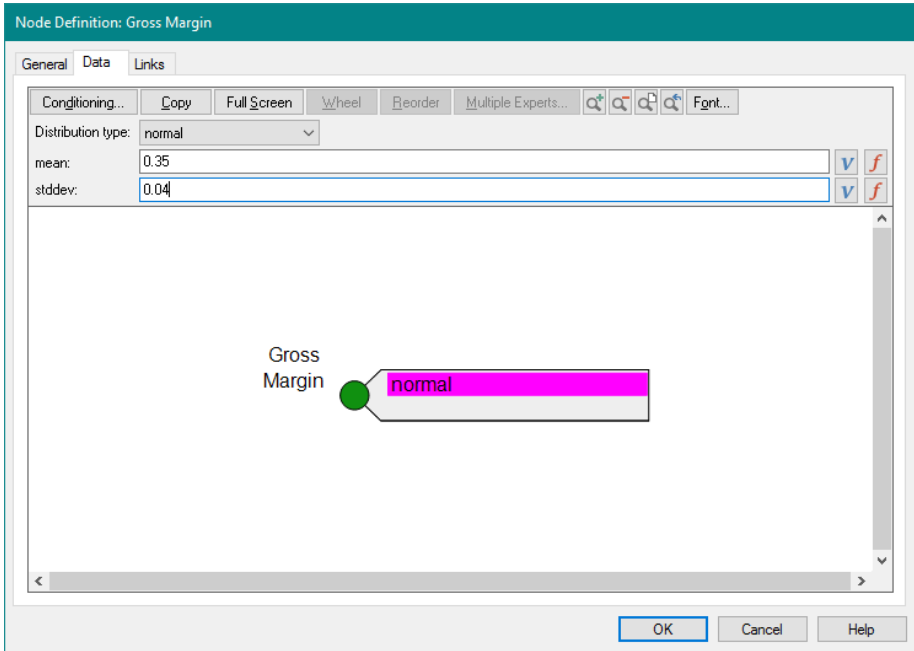
By default, DPL assigns a *Distribution type* of normal with the original value (0.35) specified as the mean. This is fine for the purposes of this tutorial but you'll need to supply the standard deviation (stddev).

⇒ Click the edit box next to stddev.

⇒ Delete "1".

⇒ Type "0.04". The node definition dialog should look like Figure 6-14.

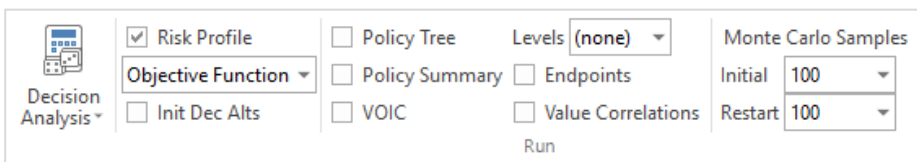
⇒ Click OK.



**Figure 6-14. Node Definition Data for Gross Margin**

Now that your model includes a continuous chance node, you can run a Monte Carlo simulation.

- ⇒ In the Monte Carlo Samples section of the Home | Run group (see Figure 6-15) change the initial number of samples to 100.
- ⇒ Make sure Risk Profile is checked.
- ⇒ Uncheck all other outputs.



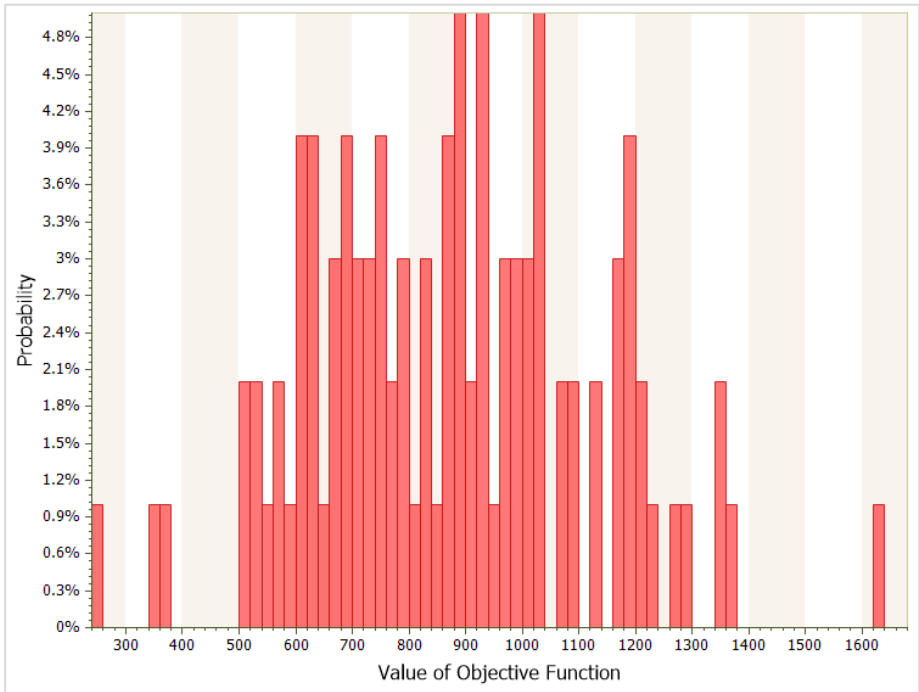
**Figure 6-15. Home | Run options for Monte Carlo Simulation**

- ⇒ Note that the default run type indicated by the Decision Analysis split button changed to Monte Carlo simulation when you introduced a continuous chance node to your model. Select Home | Run | Decision Analysis, or press F10.

DPL checks your model for completeness whenever you run any kind of analysis. DPL will give you an error message if you've mistyped a formula or forgotten a value.

DPL runs the Monte Carlo simulation with 100 samples, which is a very small number, but useful for illustration. When the simulation run is complete, DPL adds two items to the Workspace Manager: a Risk Profile Dataset and a Risk Profile Chart both of which are named "Expected Value". For more information on these items see Section 3.2. DPL will display the Risk Profile Chart (Figure 6-16).

Keep in mind that the sample is random, so your results will be slightly different, but the general shape will be similar. In this simulation, each sample has a weight (probability) of 1/100 or 1%. In the graph, you can see how the histogram bars represent single samples (probability 1%) at the high and low extremes.



**Figure 6-16. Risk Profile Histogram with 100 Samples**

To continue the risk analysis, you need to make the other key value drivers uncertain.

- ⇒ Switch back to the Risk Analysis model by pressing Ctrl+F12 or double-clicking its item in the Workspace Manager.
- ⇒ Change Trendline Revenue Growth Rate, Share of at Risk Revenue Lost and RMC Lawsuit Settlement to continuous chance nodes. You can do this all at once by selecting all three nodes using Ctrl+Click and then choosing Influence Diagram | Node | Change To | Continuous Chance.
- ⇒ Enter the probability distributions and parameters as given in Table 6-2.

Node	Distribution
Trendline Revenue Growth Rate	Normal, mean 0.08, standard deviation 0.01
Share of at Risk Revenue Lost	Triangular, min 0.50, max 0.95, mode 0.70
RMC Lawsuit Settlement	Lognormal, mu 4.0, sigma 1.0

**Table 6-2. Probability Distribution Parameters for Continuous Chance Nodes**

- ⇒ Run a Monte Carlo Simulation with the same settings as Figure 6-15. Note: DPL will issue a warning about deleting unsaved output results.
- ⇒ Click OK to the warning. DPL creates and displays a new Risk Profile Chart.

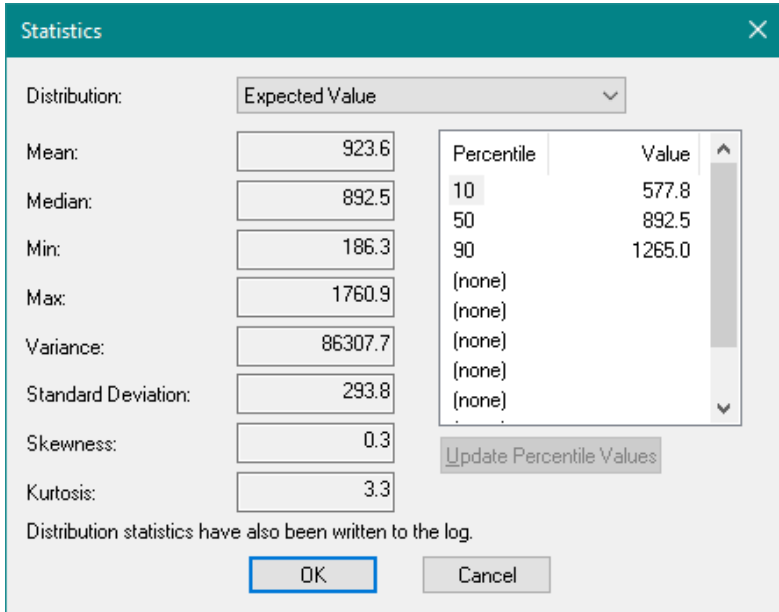
Finally, you will examine the statistics available for the Risk Profile.

- ⇒ Click Data | Distributions | Statistics. The Statistics dialog appears as shown in Figure 6-17.

The Statistics dialog gives you a standard statistical description of the results of your Monte Carlo simulation, including the minimum and maximum values, the variance and standard deviation, the higher moments (skewness and kurtosis), and selected percentiles of the distribution. This information has also been written to the session log.

On the right-hand side of the dialog the Percentiles table displays the 10<sup>th</sup>, 50<sup>th</sup>, and 90<sup>th</sup> percentiles of the distribution by default. You can use the table to determine the values for other percentiles in the distribution. To do so, double-click a (none) item within the Percentile column to put it into edit mode and enter a percentile.






**Figure 6-17. Risk Profile Statistics**

⇒ Click OK to close the Statistics dialog.

Note that because each simulation run will produce somewhat different results and you have run only 100 samples, your results will not match Figure 6-17 exactly, but they should be similar. The Risk Profile statistics are also written to the Session Log. To see this, scroll down to the bottom of the Session Log.

## 6.6 Comparing Risk Profiles


DPL allows you to display up to 32 risk profiles in a single chart window. You'll use this capability now to show how increasing the number of samples increases the accuracy of the results.

- ⇒ In the Workspace Manager, right-click on the Risk Profile Dataset item called "{Expected Value}" (  ).
- ⇒ Choose Rename.
- ⇒ Call this Risk Profile Dataset "100 Samples". The Risk Profile chart's name will be updated as well.

DPL reuses the "Expected Value" dataset each time an analysis is run. If you want to save a distribution, you need to rename it in the Workspace Manager as you've just done. For more information on saving outputs, see Section 3.4.


Risk Profile Charts and Risk Profile Datasets are separate items in the Workspace Manager because one chart can display several datasets. For more information on Risk Profile Dataset and Charts in the Workspace Manager, see Section 3.2.

Next, you'll run simulations with larger numbers of samples.

- ⇒ Run a Monte Carlo Simulation with 1000 initial samples.
- ⇒ Rename the "{Expected Value}" Risk Profile Dataset () to be "1000 Samples".
- ⇒ Repeat the process specifying 10000 initial samples and renaming the Risk Profile Dataset "10000 samples."

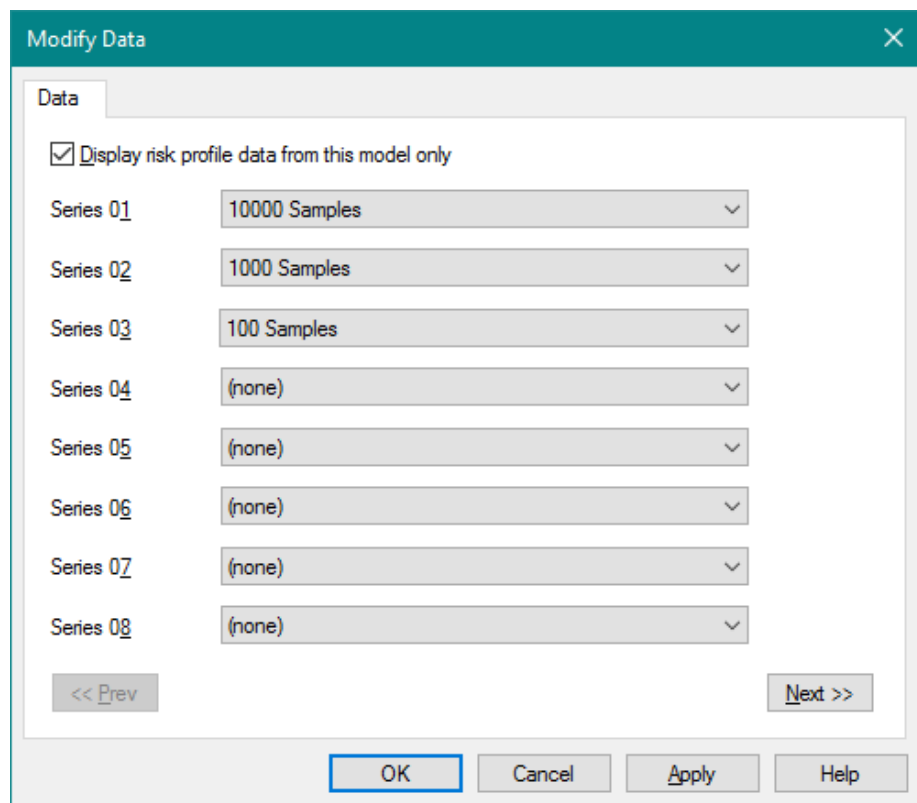
The run with 10000 samples may take a minute or two. You can take the opportunity to switch over to Excel and watch the numbers change. Excel needs to recalculate Total Value 10000 times.

Now you have three Risk Profile Datasets that you can display in a single Risk Profile Chart for comparison.

- ⇒ In the Workspace Manager, rename the "10000 Samples" Risk Profile Chart () to be "Sample Size Comparison".
- ⇒ If it isn't already, double-click the Sample Size Comparison chart to make the window active.

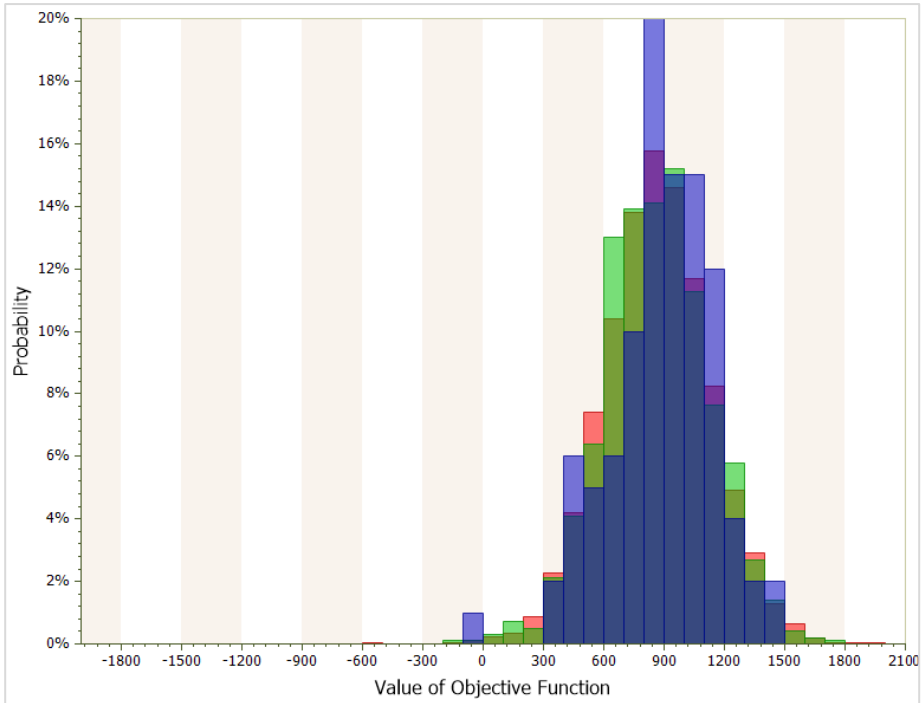
Initially, the chart will display the Expected Value risk profile dataset from the last simulation run. You want it to display all three of the saved risk profiles.

- ⇒ Click Chart | Series | Data | Modify from the menu. The Modify Data dialog is displayed. The 10000 Samples dataset is selected for *Series 01*.
- ⇒ In the drop-down list for *Series 02*, select 1000 Samples.
- ⇒ Select 100 Samples for *Series 03*. The dialog should match Figure 6-18.



**Figure 6-18. Format Diagram Dialog, Series tab**

⇒ Click OK. See Figure 6-19.



**Figure 6-19. Sample Size Comparison**

To make it easier to compare the Risk Profiles:

⇒ Uncheck Format | Display | Histogram and Format | Color | Color Fill

The chart window displays all three of the risk profiles with a single line in cumulative form (Figure 6-20).

You will also add a legend to indicate which color goes with which risk profile.

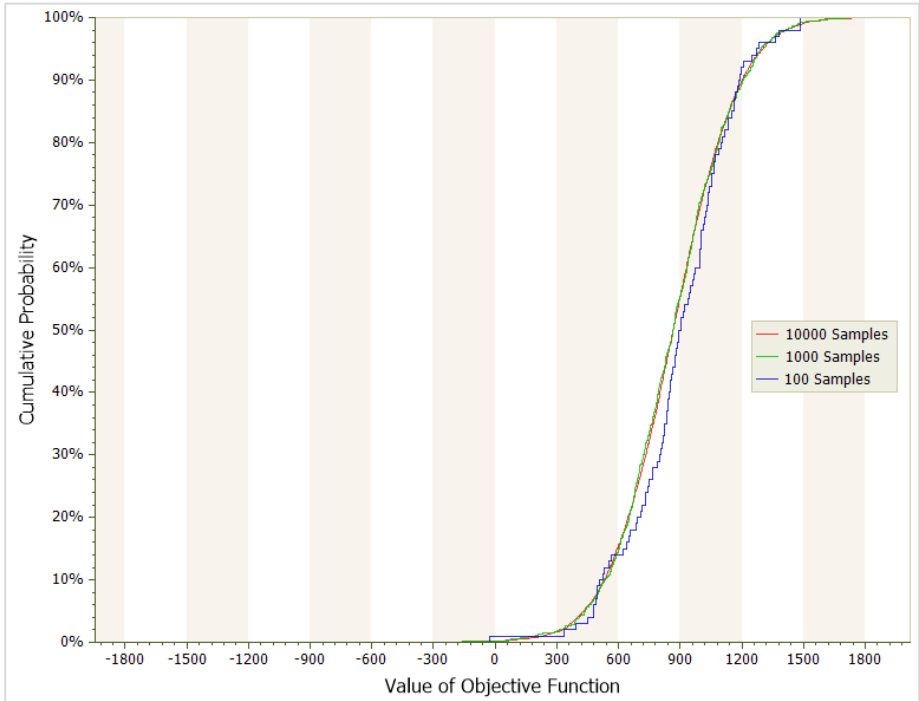
⇒ Check the checkbox next to Show within the Format | Legend group.

⇒ Try out different placements of the legend by using the commands available in the Format | Legend group. If you would like it to be placed as it is in Figure 6-20, choose Right from the drop-down list and check the checkboxes next to Centered and Inside.

It's clear that the three risk profiles are similar in a general sense, but that the 100 Samples curve deviates from the other two. This deviation may be easier to see if the lines were thinner.

⇒ To decrease the weight of the line of the curves, select a curve (it should turn magenta) and go to Series | Lines/Markers. Within the

Weight edit box delete "2" and enter "1". The weight of the selected line will become thinner. Do the same for the other two lines. (See Figure 6-20)



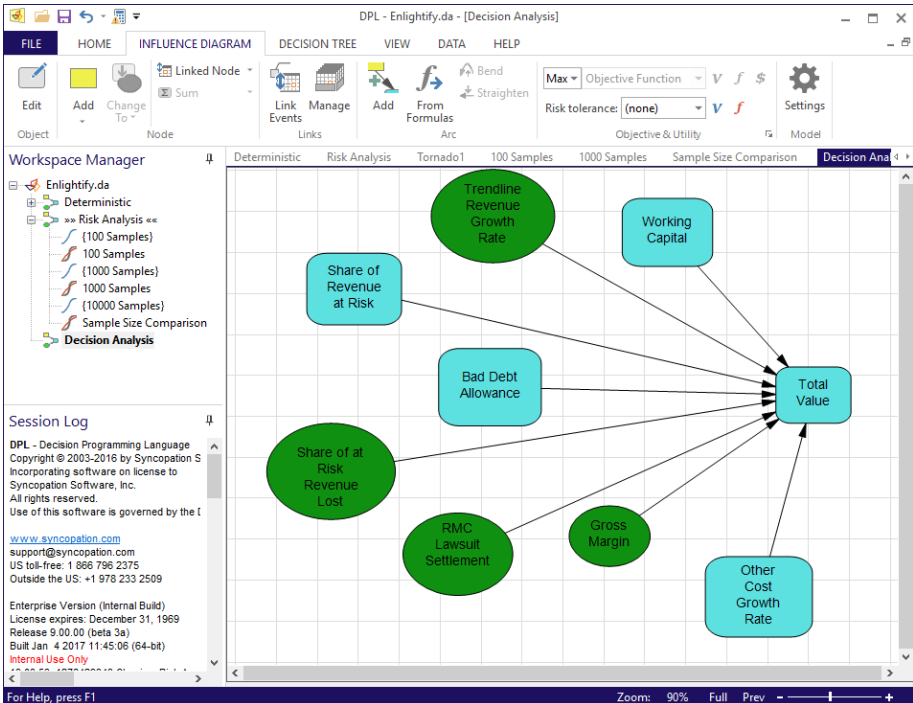
**Figure 6-20. Formatted Sample Size Comparison with Legend**

If you like, you can compare the statistics and percentiles of the risk profiles by clicking Data | Distributions | Statistics again. Use the drop-down list at the top of the dialog to select from the three distributions.

## 6.7 Modeling an Up-Front Decision

Thus far, your analysis has focused on the risks to the business, without considering any decisions management might be able to make to mitigate those risks. Enlightify's parent organization has considered spinning off Enlightify. Doing so would alleviate regulatory pressure, but would probably reduce its rate of growth because cross-selling opportunities ("synergy") would be lost. Next, you'll add this decision to your model. Before making this change, you want to make a copy of the model as it is.

- ⇒ In the Workspace Manager window, right-click on "Risk analysis" and select Duplicate.
- ⇒ Rename the new model "Decision Analysis" by right-clicking on it and selecting Rename.

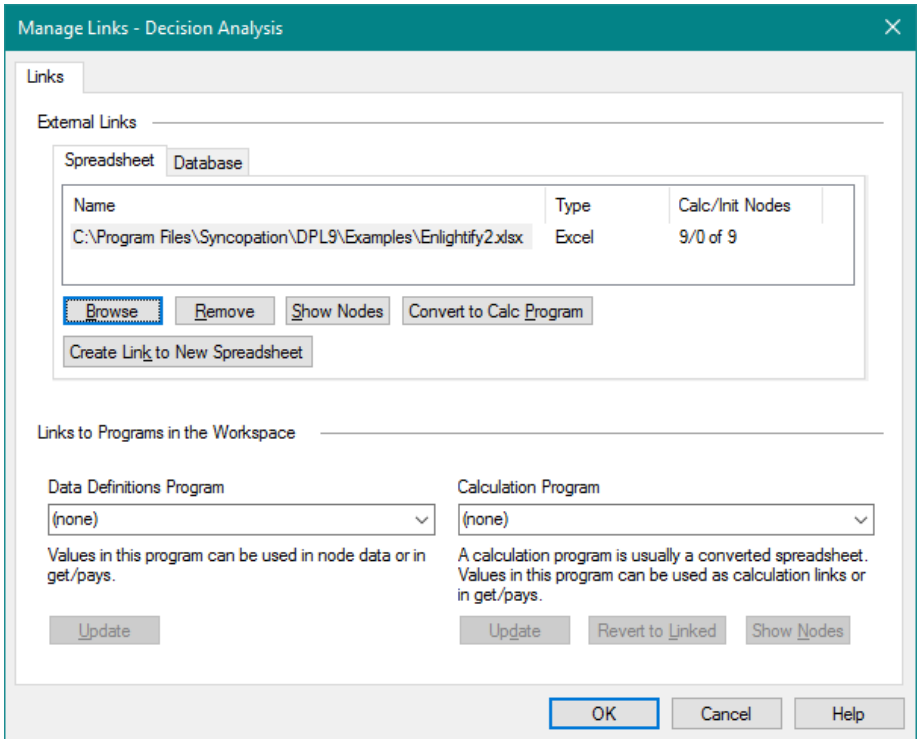


**Figure 6-21. New Model for Decision Analysis**

Assume you've prepared a spreadsheet which includes two new input drivers, one for the spinoff decision and one for its effect on the growth rate. You'll need to link the decision analysis model to the new spreadsheet.

- ⇒ Select Influence Diagram | Links | Manage. The Manage Links dialog is displayed (see Figure 6-22) with the Enlightify1.xlsx file path specified.
- ⇒ Click the Browse button.
- ⇒ Select "Enlightify2.xlsx".
- ⇒ Click Open.

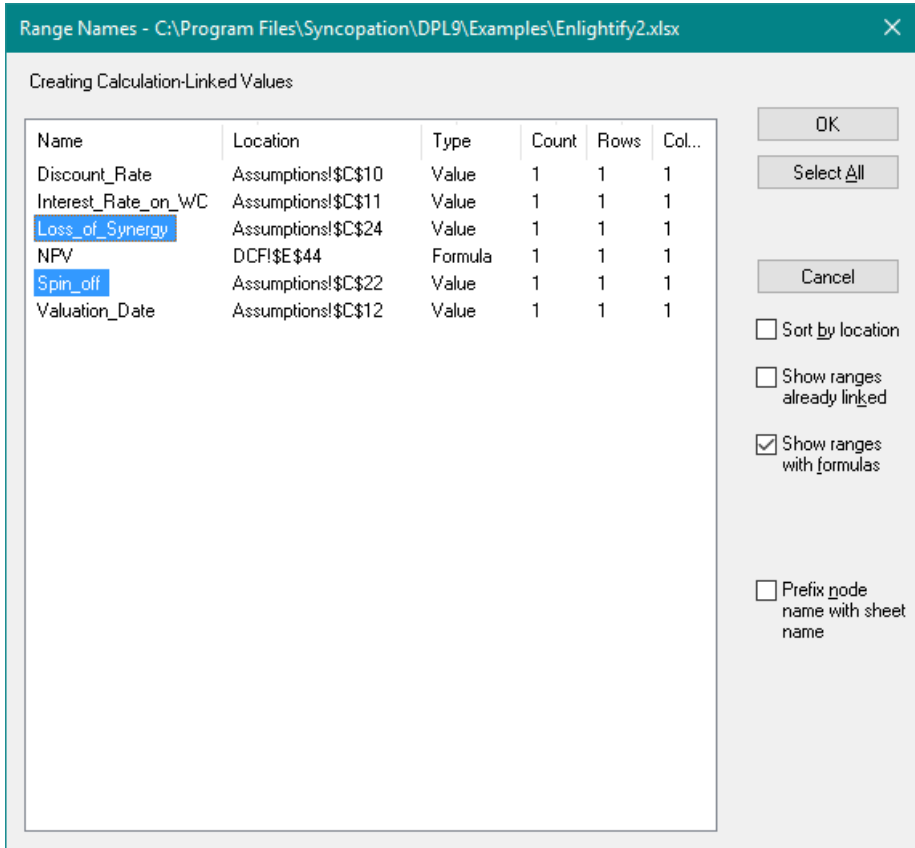
- ⇒ If you can't see the full path and name of the newly linked spreadsheet (Enlightify2), drag the Name column separator to the right. You should see that Enlightenment2.xlsx is now linked to the model.
- ⇒ Click OK.



**Figure 6-22. Manage Links Dialog**

Next, you'll create two new DPL nodes for these new inputs in the spreadsheet.

- ⇒ Click Influence Diagram | Node | Linked Node. Excel Calculation-Linked should be the default (📄). If not, select it from the drop down.
- ⇒ The new spreadsheet is opened and the Range Names dialog is displayed. Select Spin\_off and Loss\_of\_Synergy (hold down the Ctrl key while clicking the second item -- Figure 6-23).



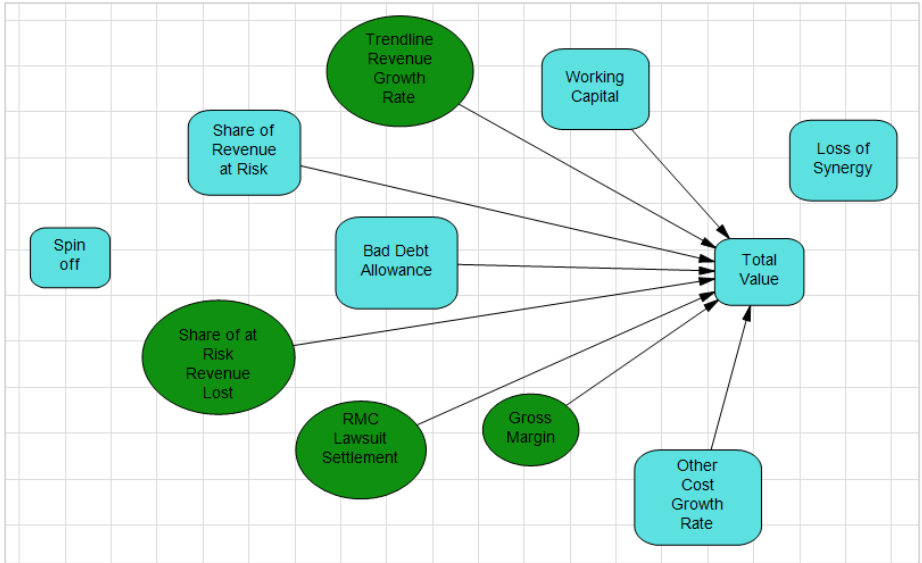
**Figure 6-23. Range Names dialog for Enlightify2**

⇒ Click OK.

DPL initially adds the new nodes to the right.

⇒ Move them so they are positioned as in Figure 6-24.





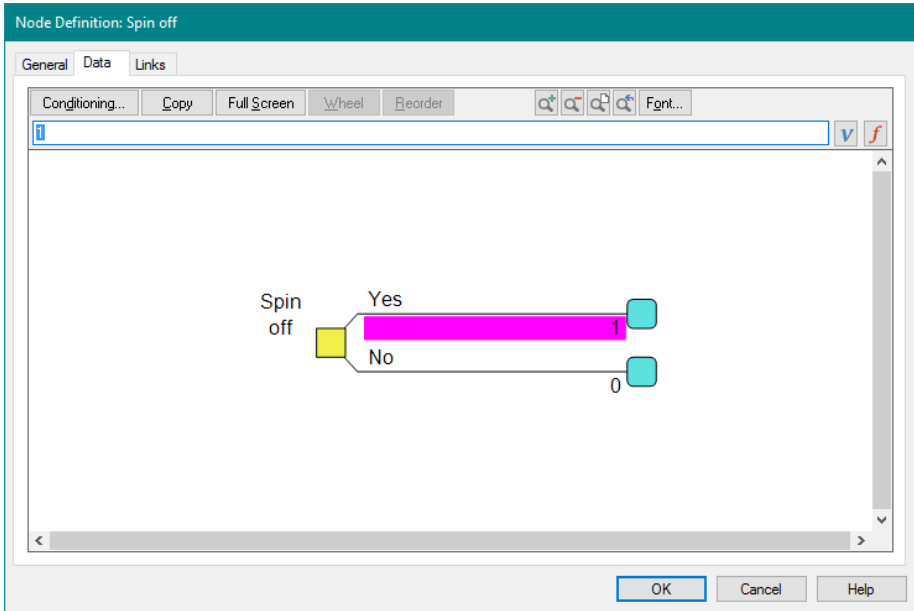
**Figure 6-24. Influence Diagram with Spin off and Loss of Synergy added**

Next, you need to change Spin off into a decision node.

- ⇒ Select Spin off and drop-down the Influence Diagram | Node | Change To split button and choose Decision from the list.

DPL displays the General tab of the Node Definition dialog. The default decision alternatives, Yes and No, are fine, but you need to edit the data. In the spreadsheet, there are flag values associated with the decision alternative. A value of 0 for Spin off means that no action is taken (No), and a value of 1 means that Enlightify is sold (Yes).

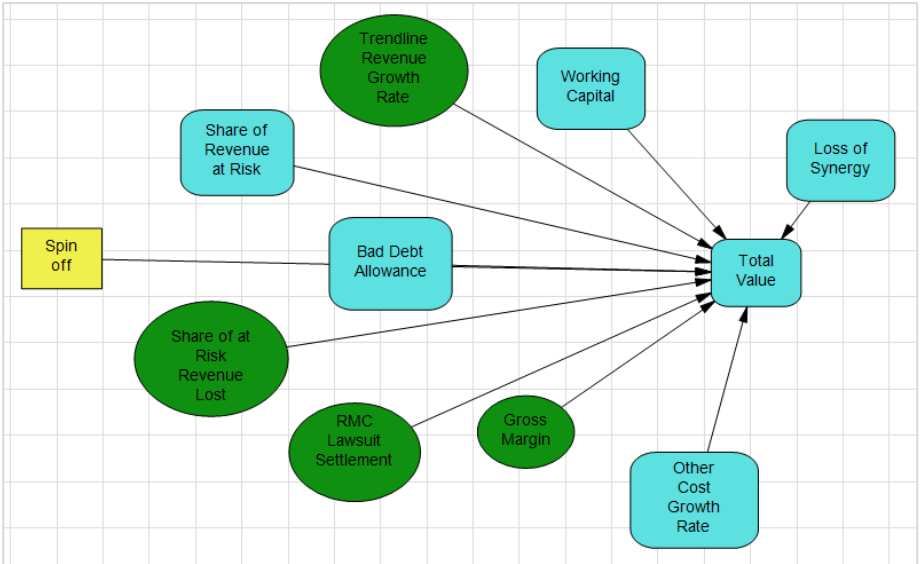
- ⇒ Click on the Data tab in the dialog. You'll see that a value of 1 is entered for both alternatives. Leave the Yes alternative as is.
- ⇒ Press the down arrow to navigate to the No branch.
- ⇒ Type "0" (Figure 6-25).
- ⇒ Click OK.



**Figure 6-25. Node Definition Data for Spin off**

⇒ Select Influence Diagram | Influence/Arc | From Formulas so that DPL will add arcs for the new nodes.

Your model should look like Figure 6-26.



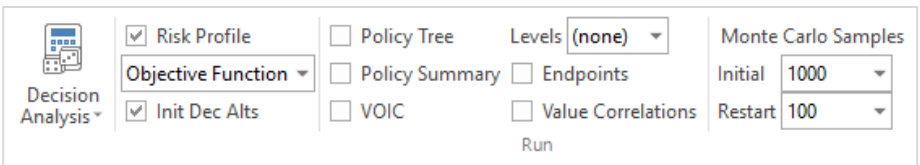
**Figure 6-26. Model with Spin off decision**

Note: Loss of Synergy is simply a linked value node that is set to 0.06 for now. You are linking this value to the model because in the next section, you will see how to perform sensitivity analysis on the assumed value.

You are ready to run the decision analysis model. You may want to save your workspace at this point.

- ⇒ Change the initial number of samples to "1000". Keep Restart at 100.
- ⇒ Make sure Risk Profile and Init Dec Alts is checked within the Home | Run group.

The Home | Run group should look like Figure 6-27.

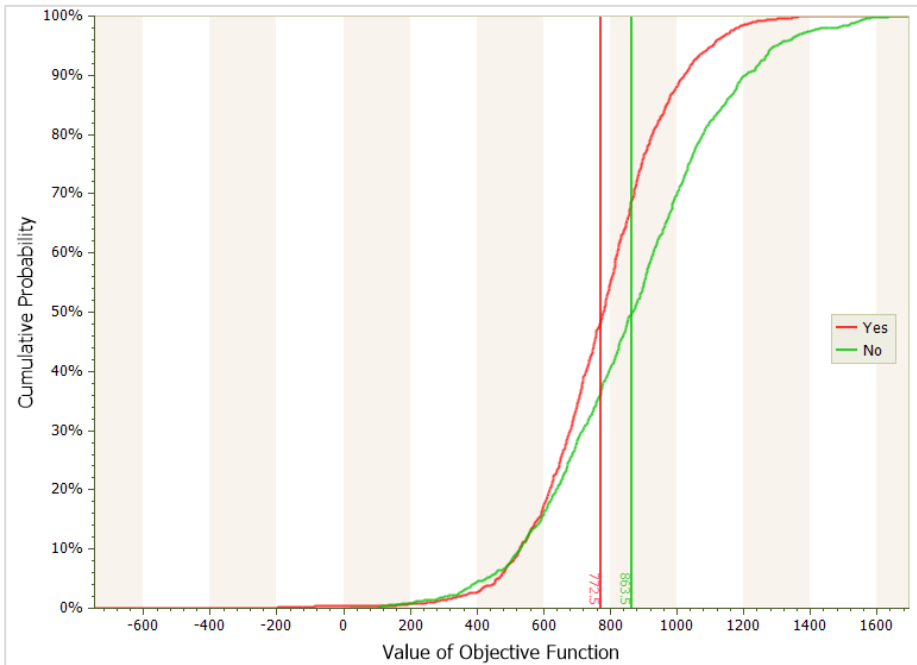


**Figure 6-27. Home | Run options for Monte Carlo Simulation**

- ⇒ Click Home | Run | Decision Analysis (or press F10).

The analysis will take a few seconds. Excel has to recalculate the spreadsheet 2000 times; 1000 for each decision alternative. When the run is finished, DPL will display the Initial Decision Alternatives chart, which displays the probability distributions for each alternative of the initial decision on a single Risk Profile Chart. The chart will initially display the risk profiles in Histogram format, which you'll change.

- ⇒ Uncheck Histogram in Chart | Format | Display.
- ⇒ Check EV/CE Lines.
- ⇒ Check EV/CE Values. A line is added to each risk profile to show the expected value (with its value to the left of it) for each alternative. See Figure 6-28.



**Figure 6-28. Initial Decision Alternatives Chart**

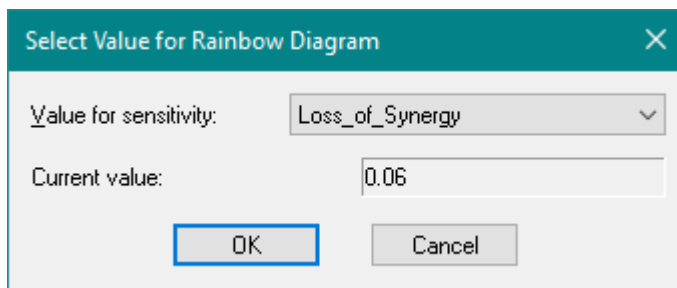
From this Risk Profile Chart, you can see that the risk profile for the No alternative generally lies below and to the right of the risk profile for the Yes alternative. This means that for almost any outcome value on the x-axis, there is a lower probability of the No alternative outcome being less than the x-value than there is of the Yes alternative being less than the x-value. Also, the No alternative has a higher expected value than the Yes alternative.

For most decision makers, the No alternative would be preferable to the Yes alternative. It appears that the parent organization should not spin off Enlightify.

## 6.8 Performing a Sensitivity Analysis

In the previous section, you concluded that it was better not to spin off Enlightify. However, you didn't test the assumption about the loss of synergy that would result from a spin off. DPL's Rainbow Diagram feature allows you to run a model for several settings of an input variable. You'll do this now for the Loss of Synergy variable to test the robustness of the Spin Off decision.

- ⇒ Select Home | Sensitivity | Rainbow Diagram. DPL displays the Rainbow Diagram Setup dialog.
- ⇒ Click Select. DPL displays the Select Value for Rainbow Diagram dialog (Figure 6-29).
- ⇒ Choose Loss\_of\_Synergy from the Value for sensitivity drop list.
- ⇒ Click OK.



**Figure 6-29. Select Value for Rainbow Diagram dialog**

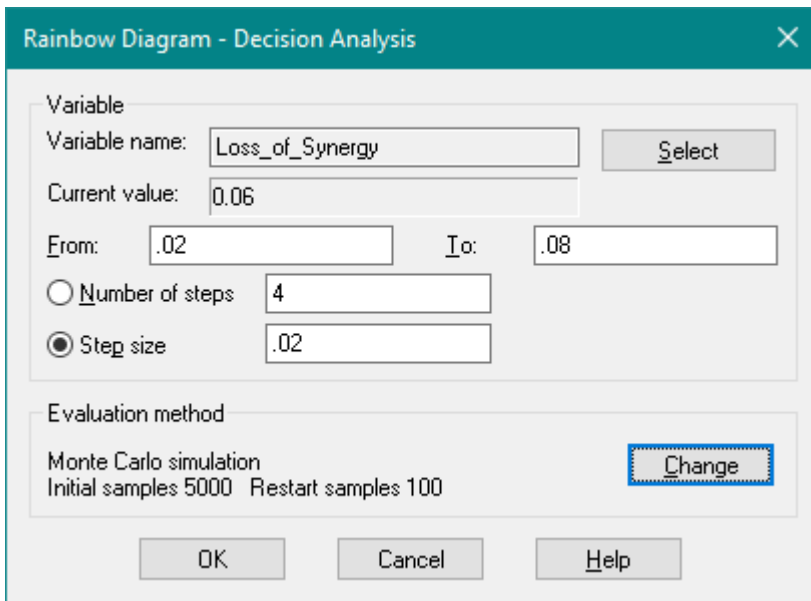
You would like to know how low Loss of Synergy would have to be, in order for spinning off Enlightify to be the optimal alternative.

- ⇒ Type "0.02" for From.
- ⇒ Type "0.08" for To.
- ⇒ Type "0.02" for the Step size.

While you could run the sensitivity analysis with 1000 samples, the randomness of the simulation "draw" might obscure the effect of changing

the Loss of Synergy value. For better results, you'll increase the sample size to 5000.

- ⇒ Within the *Evaluation Method* section click Change. The Run Settings dialog appears.
- ⇒ Under the *Samples for Monte Carlo and discrete tree simulation* section change the Initial number of samples to 5000.
- ⇒ Click OK. The Run Rainbow Diagram dialog should look like Figure 6-30.

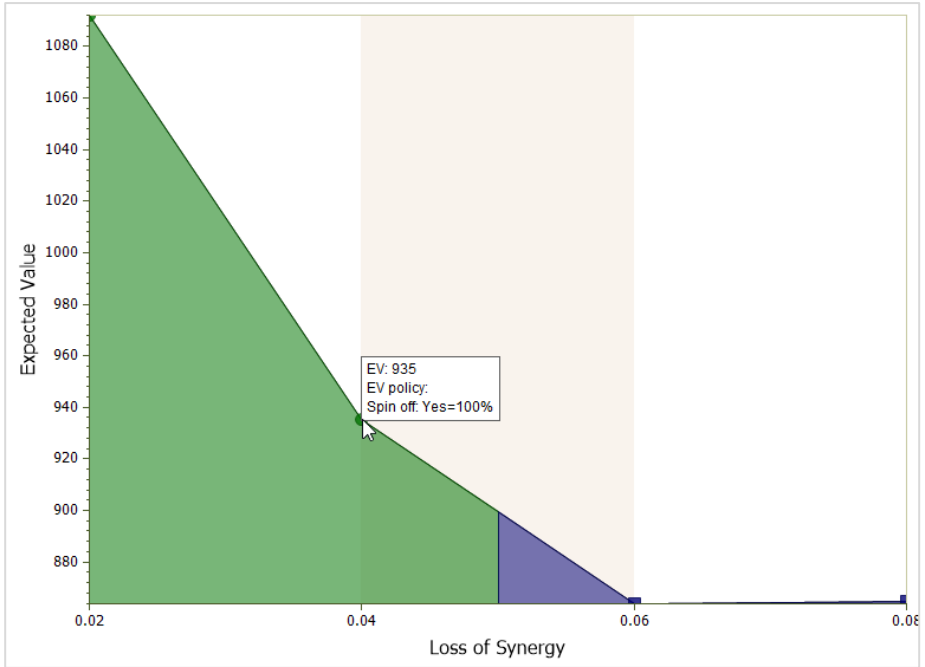


**Figure 6-30. Run Rainbow Diagram dialog**

- ⇒ Click OK to run the Rainbow Diagram.

DPL will run a full simulation for each of the four steps (0.02, 0.04, 0.06, 0.08), a total of 40,000 recalculations of the spreadsheet. This may take a few minutes.

The Rainbow Diagram (Figure 6-31) shows the model's expected value for each of the steps. The color change between 0.04 and 0.06 indicates that the best (highest expected value) decision alternative changed between these two points. This tells you that if you knew Loss of Synergy would be only 4%, it would be better to spin off Enlightify. If you wanted to find the threshold more precisely, you could run another rainbow diagram with a range of 0.04 to 0.06 and a smaller step size.



**Figure 6-31. Rainbow Diagram on Loss of Synergy**

This Rainbow Diagram only has two colors, and the model only has one decision with two alternatives, so it's not hard to figure out the alternative represented by each color. In a more complex diagram, you can use DPL's policy tip feature to see the optimal alternative for each step. If you hold the mouse cursor over the green circle for 0.04 (as has been done in Figure 6-31), DPL will tell you that Spin off is Yes at that step. Note: Show Tips button must be on (shaded blue) within the View | Tips group in order for policy tips to work.



## 6.9 Modeling a Downstream Decision

The decision analysis model used in the previous two sections assumes that the Spin off decision will be made at a time when the four uncertainties are not known. In Figure 6-26 the symmetric, default Decision Tree DPL has been building as you've built your Influence Diagram is displayed (you can press the Tab key to change your view to the Decision Tree pane). As you can see the decision comes first in the Decision Tree sequence.

You might ask how the results would differ if the decision could be made later, after the uncertainty Share of at Risk Revenue Lost is resolved. To answer this question, you will change the sequence of the nodes in the tree so that the decision is preceded by this uncertainty. A decision that comes after one or more uncertainties is called a downstream decision or a real option. For more on modeling real options using downstream decisions, see Chapter 7.

- ⇒ In the Workspace Manager, right-click on "Decision Analysis" and select Duplicate.
- ⇒ Rename the new model to be "Real Option Analysis".
- ⇒ If you've switched over to the Decision Tree pane, switch back to the Influence Diagram pane.

To make the Spin off decision a real option triggered by the Share of at Risk Revenue Lost uncertainty, you will need to tell DPL that the uncertainty comes first in the Decision Tree sequence. The easiest way to do this is by adding a timing arc within the Influence Diagram.

- ⇒ Click Influence Diagram | Influence/Arc | Add. The cursor changes to the begin arc cursor () .
- ⇒ Click on Share of at Risk Revenue Lost. The cursor changes to the end arc cursor () .
- ⇒ Click on the Spin off decision. DPL automatically reorders the Default Tree to reflect the new arc (Figure 6-32).
- ⇒ To view both the Influence Diagram and Decision Tree panes, drag the splitter up from the bottom of the Model Window.



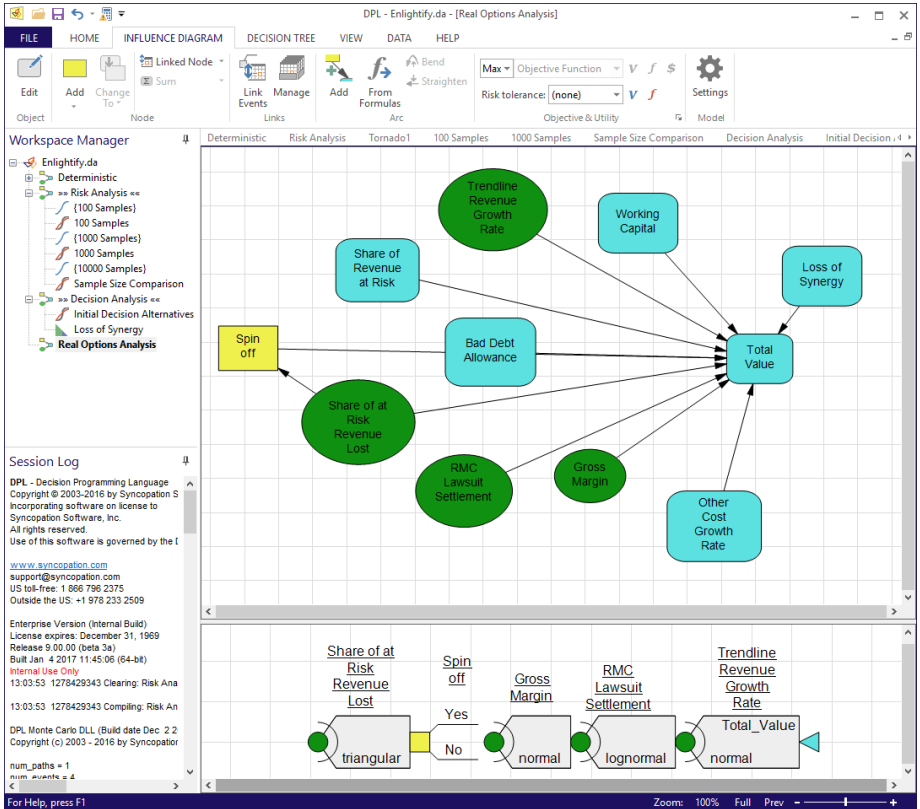


Figure 6-32. Reordered Decision Tree

- ⇒ The Monte Carlo options in the Home | Run group should be modified as shown in Figure 6-33. Make sure Risk Profile is checked. Uncheck Init Dec Alts.
- ⇒ Change Initial number of samples to 500.
- ⇒ Check the Policy Summary checkbox.
- ⇒ Select "3" from the Number of Levels drop-down list.

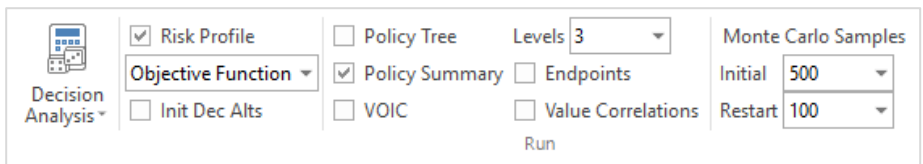


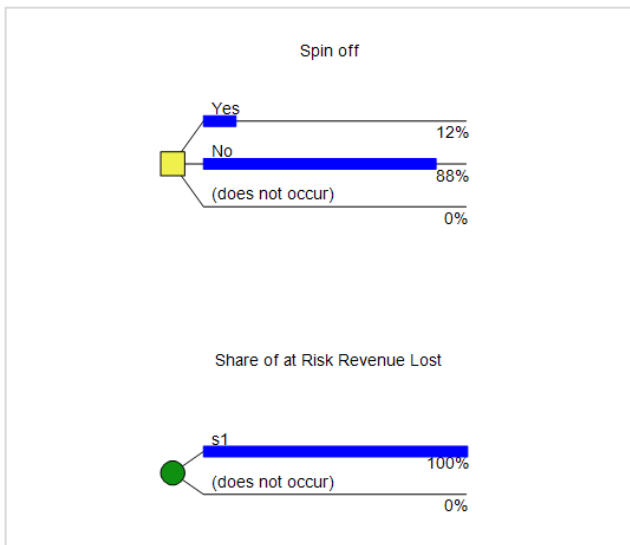
Figure 6-33. Monte Carlo Simulation Options for Downstream Decision

⇒ Click Home | Run | Decision Analysis (or press F10).

The run may take several minutes. DPL must sample each alternative of the downstream decision 100 times, and the decision itself will be sampled 500 times, so the run will require 100,000 recalculations of the spreadsheet.

When the run completes, DPL displays the Policy Summary.

The Policy Summary shows that the option to spin off the business is exercised in about 12% of the scenarios. For more information on Policy Summaries, see Section 3.3.



**Figure 6-34. Policy Summary™ for the Spin off option**

For continuous chance nodes in the Policy Summary, DPL creates two pseudo-states: s1 and (does not occur). The probabilities (or percentages) displayed on these two states indicate how often the chance node occurs in the optimal set of scenarios. In Figure 6-34, the percentage for s1 is 100%. This means that the chance node occurs in all optimal scenarios. This makes sense because the Decision Tree is symmetric (every node occurs on every path). In an asymmetric tree, a continuous chance node might not appear in all optimal scenarios.

If you'd like, you can compare this risk profile dataset, its expected value and other statistics with one from the model without the downstream decision, as described in Section 6.6. Note: to do so you will need to

uncheck *Display risk profile data from this model only* on the Modify Data dialog.

## 7. Conditioning and Learning in Decision Models

When a decision is preceded by a chance node that provides information your model contains learning. For example, before deciding whether to carry an umbrella, you may look out the window and see if there are clouds. Learning about the "clouds" uncertainty provides imperfect information about the uncertainty of concern: whether or not it will rain later that day.

Other typical examples of learning in decision models include medical tests, market research, and sampling of the physical environment. Test drilling for oil is a classic example of learning and is the basis for the tutorial in this chapter.

This chapter assumes you have already been through at least one of the earlier tutorials in this manual or that you are already familiar with DPL. Therefore, basic procedures such as starting DPL and opening and saving workspaces will not be explained in detail here.

This chapter also assumes some familiarity with basic probability theory and Bayes' Rule. The example in this chapter demonstrates DPL's capability to perform Bayesian revision, which can be simply described as reversing the order of conditioning among chance nodes to take advantage of the data available. Please consult DPL's online Help and/or a probability theory or decision analysis textbook if you need further explanation of these topics.

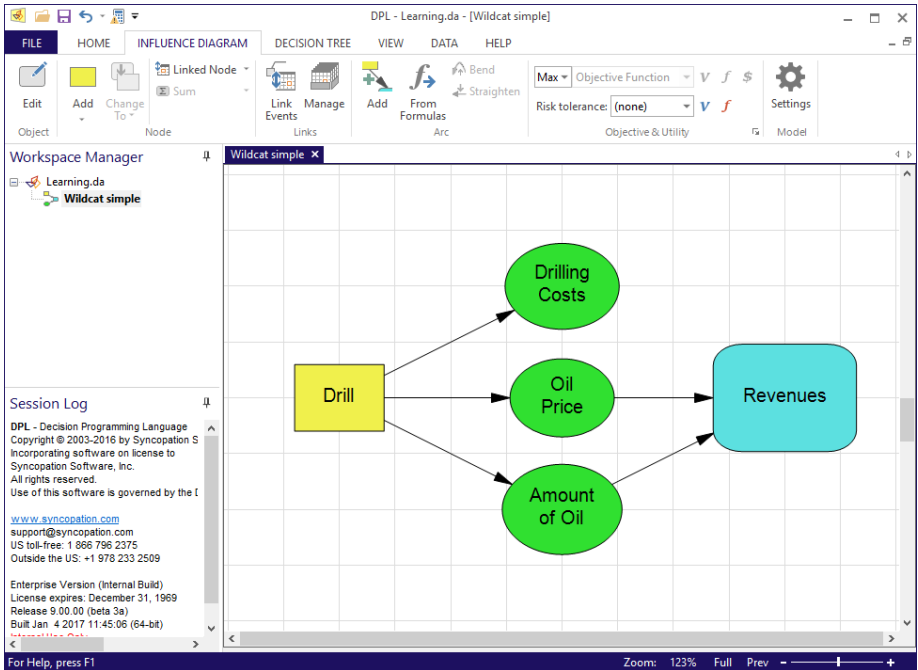
### 7.1 Incorporating Imperfect Information in a Model

---

In this section, you will enhance a simple decision model by adding a node that represents imperfect information about an important value driver. You will see how the "learning" associated with such nodes can significantly change the measured value of an asset.

⇒ Start DPL.

- ⇒ Open the workspace Learning.da. This file is found in the Examples folder where DPL was installed, usually C:\Program Files\Syncopation\DPL9\Examples. See Figure 7-1.



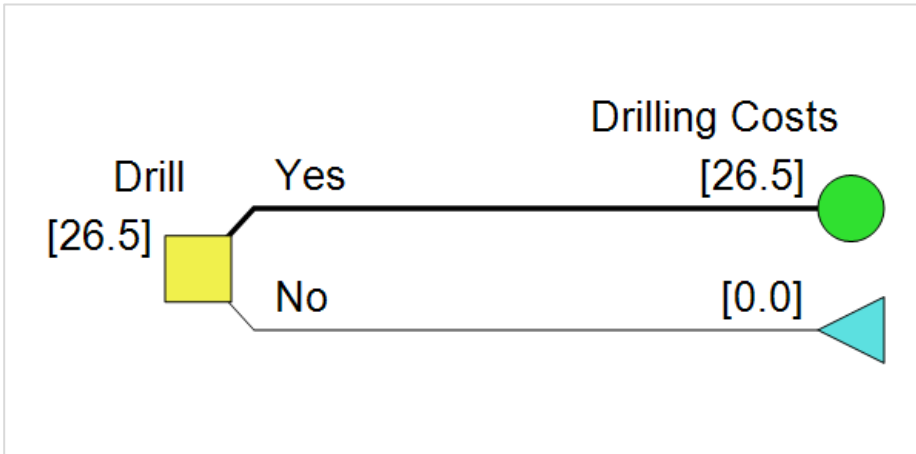
**Figure 7-1. Simple Wildcat Decision Model in Influence Diagram-focused Modeling Mode**

In this tutorial you will be modeling within both panes so you can set the splitter halfway or use the Tab key to switch between maximized panes when prompted. The tutorial's screenshots will reflect the latter option with the Influence Diagram pane maximized initially.

This model represents the decision problem faced by a "Wildcatter" exploring for oil. Assume you are the Wildcatter. In a given field, you must decide whether or not to drill. If oil is found, uncertain costs are incurred. If a "dry hole" is drilled, nothing is gained. If oil is struck, then an amount of revenue is gained which depends on the price of oil and the reserves found.

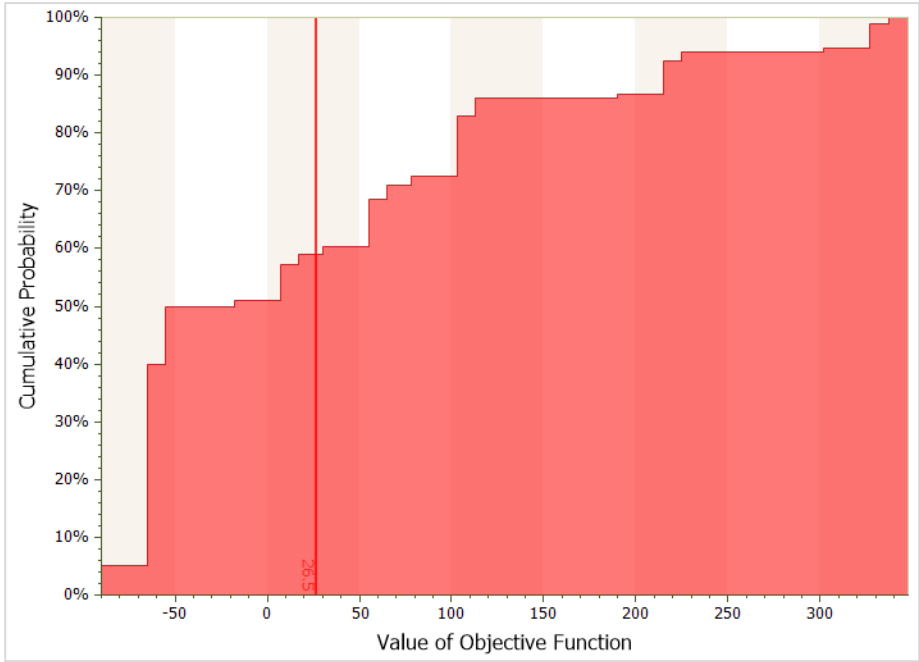
- ⇒ In the Home | Run group, check Risk Profile and Policy Tree. Uncheck the Init Dec Alts checkbox.
- ⇒ Click Home | Run | Decision Analysis.

The Policy Tree tells you that you should drill and that the expected value of this opportunity is \$26.5 million (Figure 7-2).



**Figure 7-2. Policy Tree™**

- ⇒ Activate the Risk Profile Chart (double-click on the item for it in the Workspace Manager).



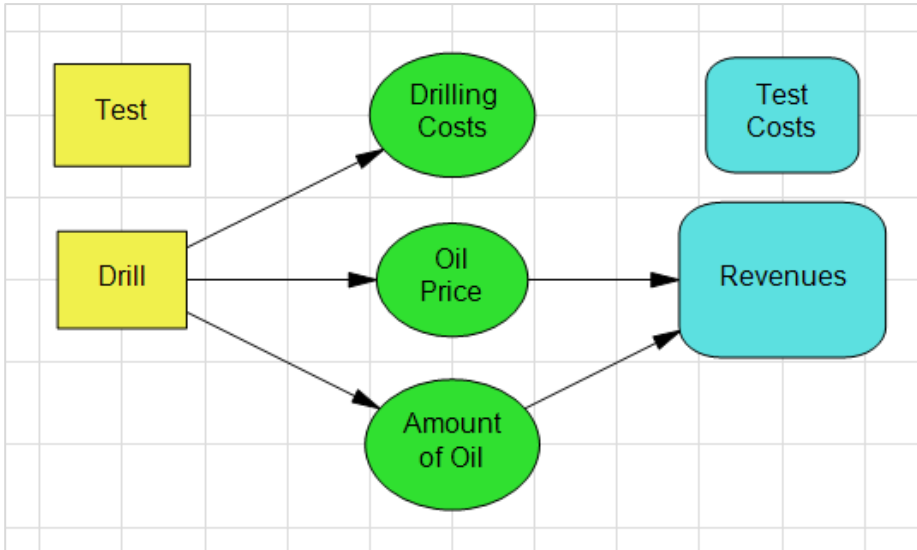
**Figure 7-3. Risk Profile**

While this opportunity has positive value, it is risky. There is a about a 51% chance of losing money (see Figure 7-3). For more on reading Risk Profiles, see Section 3.2.

In reality, you have more choices than just whether or not to drill. You can conduct one of several tests intended to tell you about the likelihood of finding oil and then decide whether or not to drill. You will add a decision and a learning uncertainty for one such test.

- ⇒ Activate the model and add a decision node to the Influence Diagram.
- ⇒ Place it above the Drill decision.
- ⇒ Call the decision node "Test".
- ⇒ Name its alternatives "None" and "Core sample". Click OK to close the Node Definition dialog.
- ⇒ Add a value node, place it above Revenues and name it "Test Costs".
- ⇒ Switch to the Data tab and enter a value for the node of "-6".

In this model, you'll use an additive convention, so all costs will be entered as negative numbers. In this case, the conducting a core sample test will cost \$6 million. Your Influence Diagram should look like Figure 7-4.



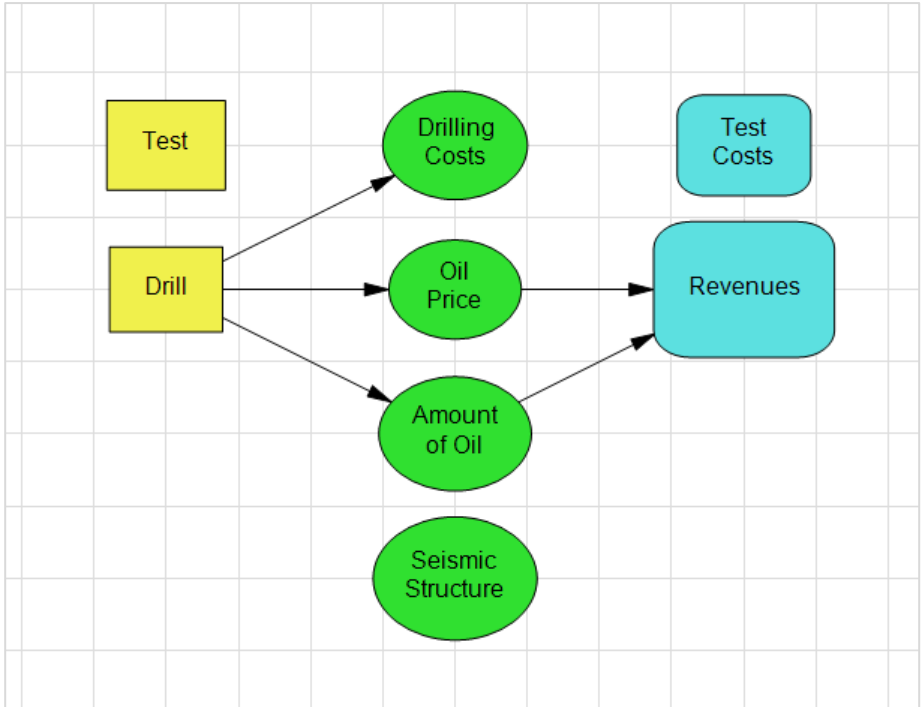
**Figure 7-4. Influence Diagram with Test decision**

You'll now add the learning node associated with the core sample test.

- ⇒ Add a new Discrete Chance Node to the Influence Diagram.
- ⇒ Place it below Amount of Oil.
- ⇒ Name the node "Seismic Structure".
- ⇒ Name its outcomes "None", "Open" and "Closed".
- ⇒ Click OK to close the Node Definition dialog.

Your model should look like Figure 7-5.

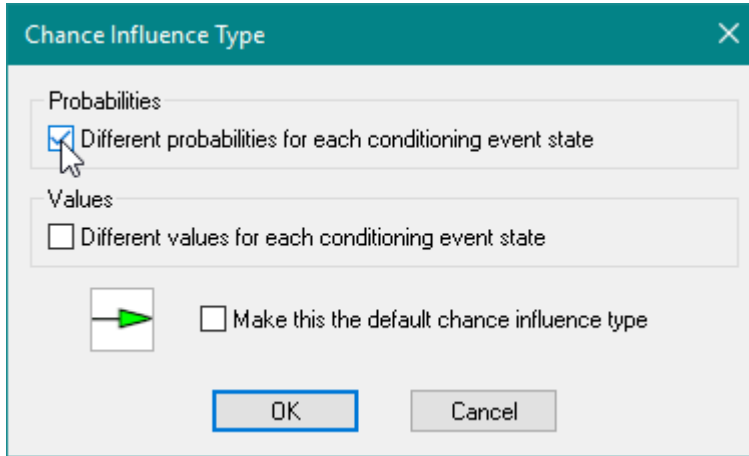




**Figure 7-5. Influence Diagram with Seismic Structure uncertainty**

You know from geological data the relationship between Seismic Structure and Amount of Oil. You need to add an arc from Amount of Oil to Seismic Structure.

- ⇒ Create an arc that goes from Amount of Oil to Seismic Structure.
- ⇒ Double-click on the new arc to change the type of the arc. DPL displays the Chance Event Influence Type dialog (Figure 7-6).
- ⇒ Check the *Different probabilities for each conditioning event state* box.
- ⇒ Make sure *Different values for each conditioning event state* is not checked.



**Figure 7-6. Chance Event Influence Type dialog**

⇒ Click OK.

Now you are ready to enter data for Seismic Structure.

⇒ Double-click on Seismic Structure.

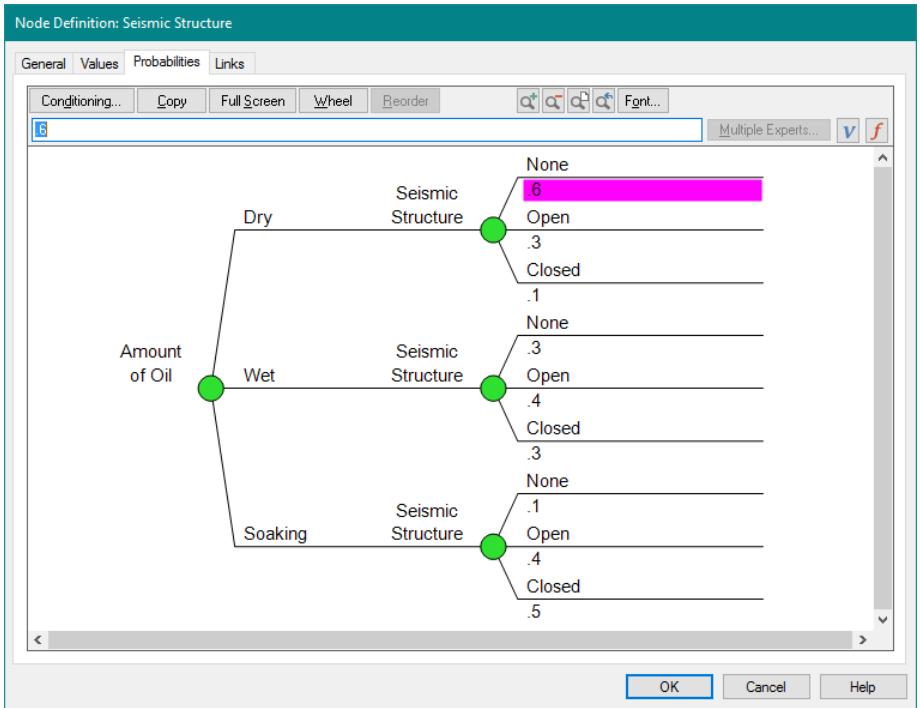
The Node Definition dialog now has separate tabs for Values and Probabilities. You don't need to enter values for this node.

⇒ Select the Probabilities tab. Enter the probabilities as shown in Table 7-1.

<b><u>Amount of Oil</u></b>	<b><u>Seismic Structure</u></b>	<b><u>Probability</u></b>
Dry	None	0.6
	Open	0.3
	Closed	0.1
Wet	None	0.3
	Open	0.4
	Closed	0.3
Soaking	None	0.1
	Open	0.4
	Closed	0.5

**Table 7-1. Probability Data for seismic structure**

Your probability input tree should look like Figure 7-7.



**Figure 7-7. Probabilities Tab of Node Definition dialog for Seismic Structure**

⇒ Click OK.

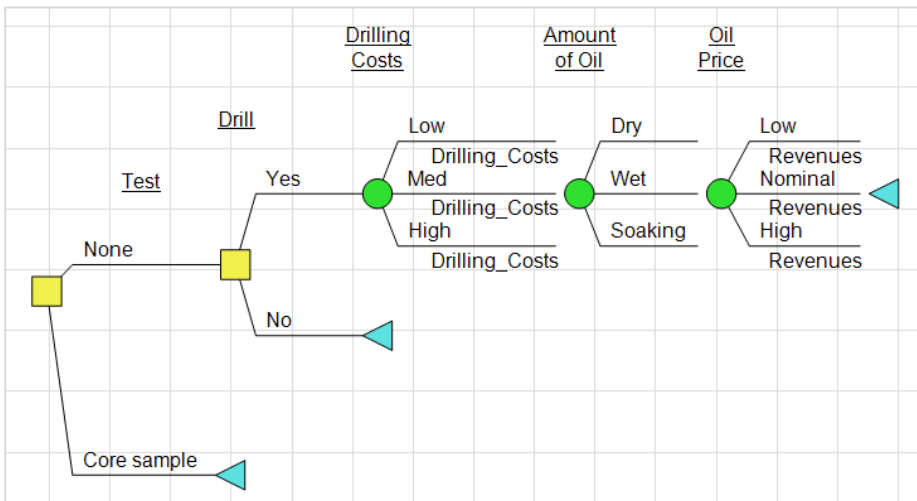
An expert geologist gave you the probability distributions for different seismic structures given different oil reservoir sizes. Had the expert given you the converse, you would have made an arc from Seismic Structure to Amount of Oil. DPL automatically performs Bayesian revision if the nodes are placed in the tree in the opposite of arc order, so you can accept the data in either form.

The Influence Diagram is complete. You now need to update the Decision Tree.

⇒ Press the Tab key to switch over to the Decision Tree pane view.

⇒ Select Decision Tree | Instance | Add | Decision, which is what should be indicated as the default command. If it isn't, drop-down the split button and select Decision... from the list.

- ⇒ Within the Select Decision dialog, select Test from the list. Click OK.
- ⇒ Place the node on top of the Drill decision at the head of the tree. The Test decision should now be at the head of the tree.
- ⇒ Select the branches of the Test decision and check the box for Decision Tree | Instance | Asymmetric. This will detach the node from the rest of the subtree.
- ⇒ Drag the Drill decision node and its subtree and place it on the endpoint for the None branch of the Test decision. Drag the Core Sample branch down to make space for additional nodes. Your model should look like Figure 7-8.

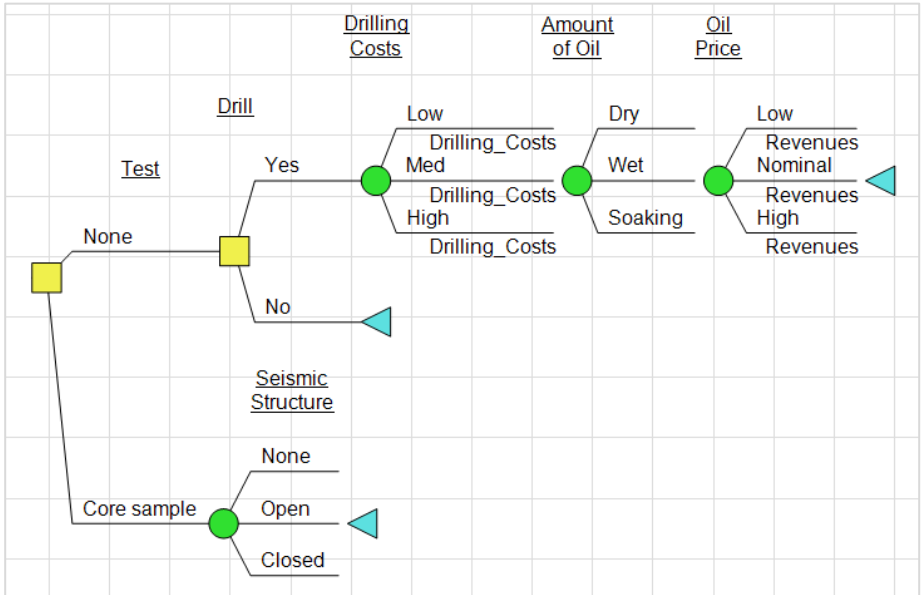


**Figure 7-8. Wildcat Decision Tree with Test Decision Added**

If the Wildcatter decides to conduct the Core sample test, s/he will know the Seismic Structure before making the decision about whether to drill. Therefore, the Seismic Structure chance node needs to be added to the Core Sample branch of the Test decision.

- ⇒ Drop-down the Decision Tree | Instance | Add split button and select Chance... from the list.
- ⇒ From the Select Chance dialog, select Seismic Structure and click OK.
- ⇒ Place the node on the blue triangle for the Core Sample branch of the Test decision.

You may need to drag the Seismic Structure node on the Core sample branch down to make some room. Your Decision Tree should now look like Figure 7-9.



**Figure 7-9. Wildcat Decision Tree with Seismic Structure Uncertainty Added**

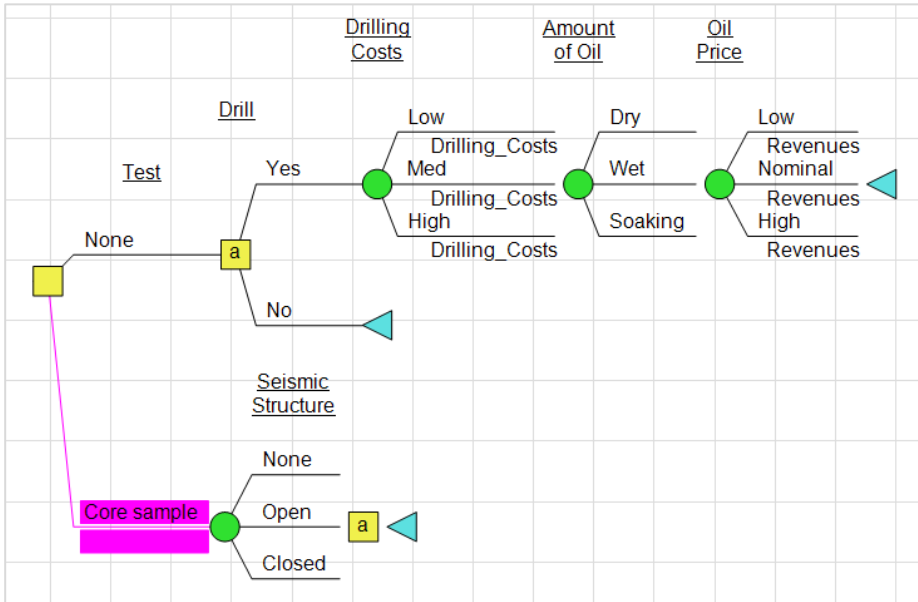
On the Core Sample path through the tree, the Seismic Structure uncertainty is followed by the Drill decision and from there on the structure is the same as the top part of the tree. Rather than repeat that sequence, you'll use the Perform Subtree feature.

- ⇒ Select the Drill decision and the endpoint for the Seismic Structure uncertainty in the tree and click Decision Tree | Instance | Subtree on the ribbon.

DPL labels the Drill decision node with an "a". This is called a Perform Target. DPL also changes the blue triangle after the seismic structure uncertainty to be a yellow decision node and labels it "a". This is called a Perform Reference. Perform Subtrees can also be headed by a chance node (i.e., the Perform Target can be a chance node). If the Perform Target is a chance node, then the Perform Reference is drawn as a chance node. Multiple Perform Subtrees are allowed within the Decision Tree.

Finally, you need to add a Get/Pay expression to the tree to account for the cost of the Core Sample test.

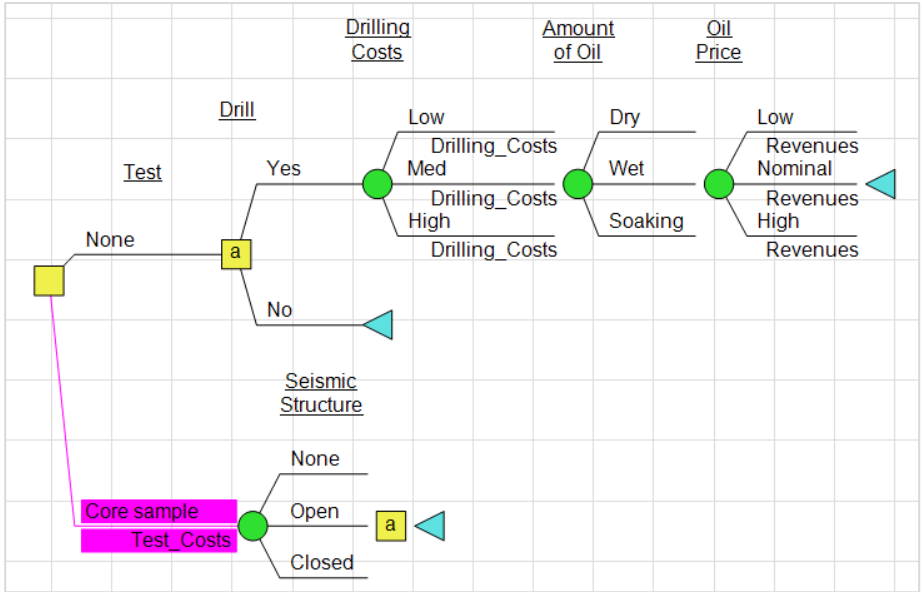
⇒ Select the Core Sample branch of the Test decision as shown in Figure 7-10.



**Figure 7-10. Decision Tree with Core Sample Branch Selected**

⇒ With the branch selected, drop-down the Edit Get/Pay list within the Decision Tree | Get/Pay group and select Test\_Costs.

Your Decision Tree should now look like Figure 7-11.

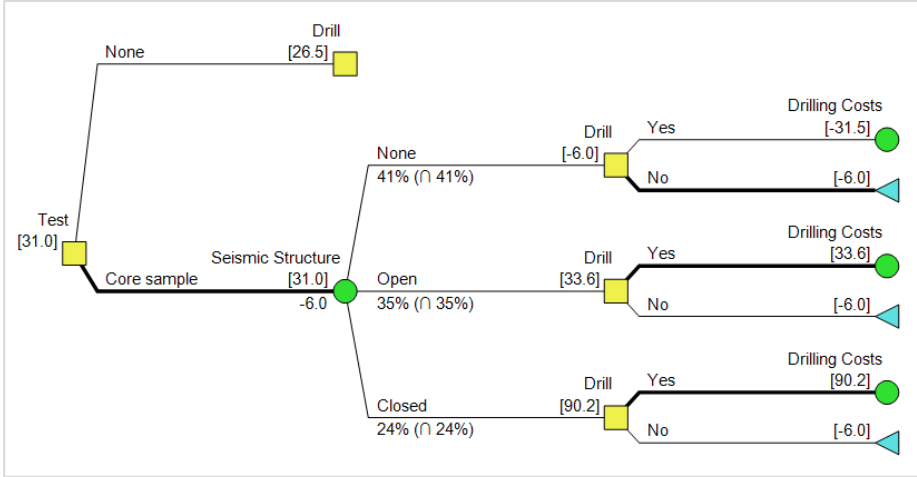


**Figure 7-11. Decision Tree with Test Costs Get/Pay Expression Added**

Now the model is complete and ready for analyses.

- ⇒ In the Home | Run group check Risk Profile and Init Dec Alts.
- ⇒ Check Policy Tree.
- ⇒ Click Home | Run | Decision Analysis.
- ⇒ Click OK to the warning.

The Policy Tree in Figure 7-12 shows that the Core sample test is the optimal alternative. Even including the \$6 million cost of the test, the Core sample alternative has an expected value of \$31 million, \$4.5 million more than drilling without a test.

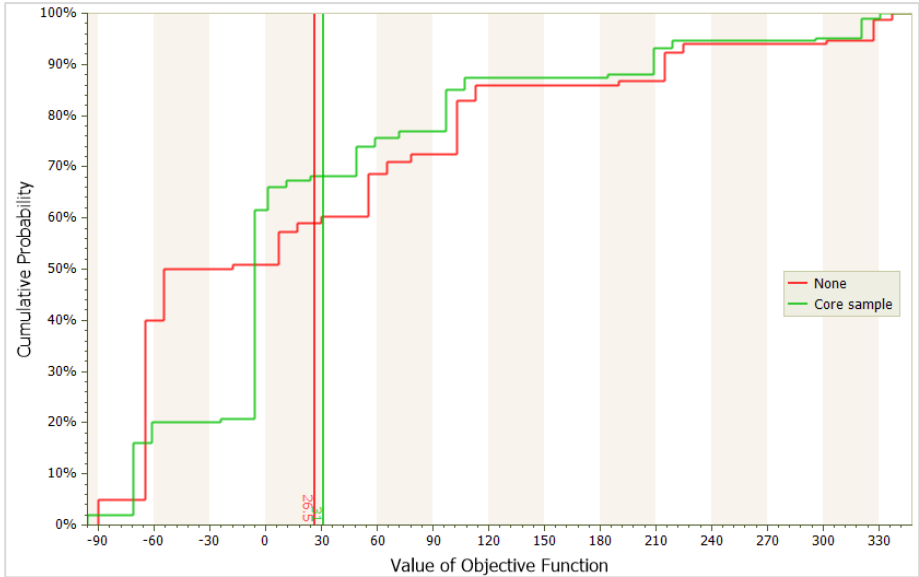


**Figure 7-12. Policy Tree™**

When there is no seismic structure, the optimal policy is not to drill, since doing so has an expected value of -\$31.5 million. The imperfect information about Seismic Structure is valuable because it changes the Drill decision in that scenario.

- ⇒ Navigate to the Risk Profile Chart (labeled Initial Decision Alternatives in the Workspace Manager). See Figure 7-13.





**Figure 7-13. Risk Profiles for the Initial Decision Alternatives**

The Risk Profile shows how the seismic structure information has a risk mitigation effect. The probability of an NPV outcome of -\$50 million or worse is 20% with the Core sample test and 50% without it.

The slight difference between the two alternatives at the higher end of the chart (upper right) simply reflects the cost of the test, which reduces the value of the Core sample alternative slightly in the best-case scenarios.

In the first implementation of the wildcat model above, the Drill decision was being made without any information. By adding the Test decision, the Drill decision becomes a real option: you have the ability but not the obligation to Drill after initially investing money in the testing decision. The reason the expected value of the model increased after you added the Test decision is due to the learning in the imperfect information contained in the Core Sample test. The Drill decision becomes an option to be exercised or not based on the results of the Core Sample outcome.

Save your model to a convenient location as it will be used in the next section.

## 7.2 Perform Subtree and Continue

---

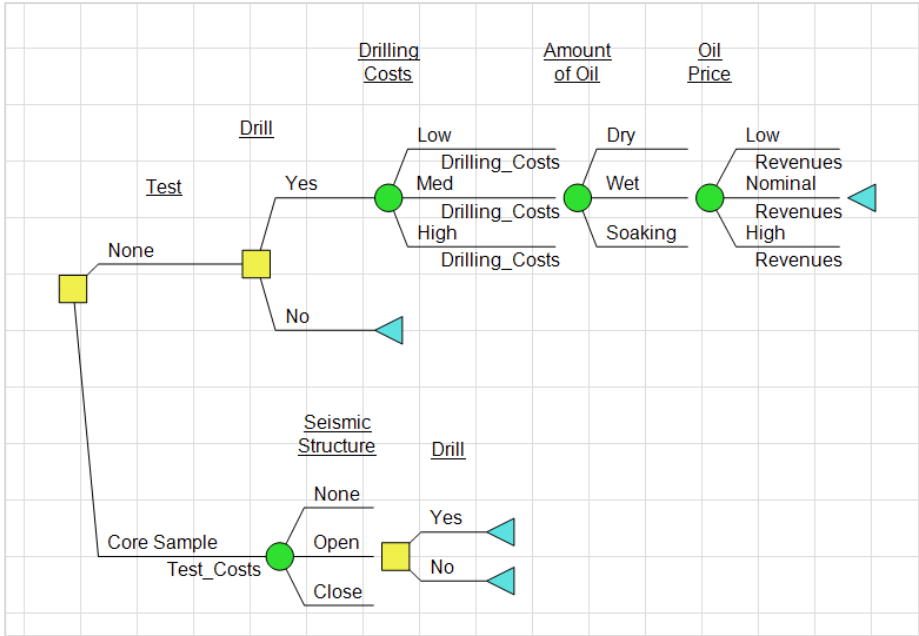
In the previous section you saw how the perform subtree feature allows you to reduce the redundancy in a Decision Tree. In this section you'll see that a perform reference can be followed by one or more other nodes, e.g., perform and continue.

- ⇒ Open the model you saved at the end of Section 7.1. Or,
- ⇒ If you did not complete the previous section you can open the workspace Learning\_done.da. This file is found in the Examples folder where DPL was installed, usually C:\Program Files\Syncopation\DPL9\Examples.

Assume that you've just learned that there will be an additional cost incurred if you proceed with the core sample test and drill. If you perform the test and decide to drill, you'll need to spend an extra uncertain amount on decommissioning the test site. Within the Decision Tree, you'll want this new Site Decommissioning Costs node to follow the Drill decision on the path where you perform the test and then drill. You'll modify the Decision Tree in order to incorporate this uncertain cost.

- ⇒ Activate the Decision Tree pane if it isn't already.
- ⇒ Select the perform reference at the Seismic Structure node and press the Delete key to remove it.
- ⇒ Click Decision Tree | Instance | Add | Decision and choose Drill from the Select Decision dialog.
- ⇒ Place the Drill decision on the Seismic Structure endpoint.
- ⇒ Make the Drill decision asymmetric by checking the Asymmetric box within Decision Tree | Instance.

Your model should now look like Figure 7-14.



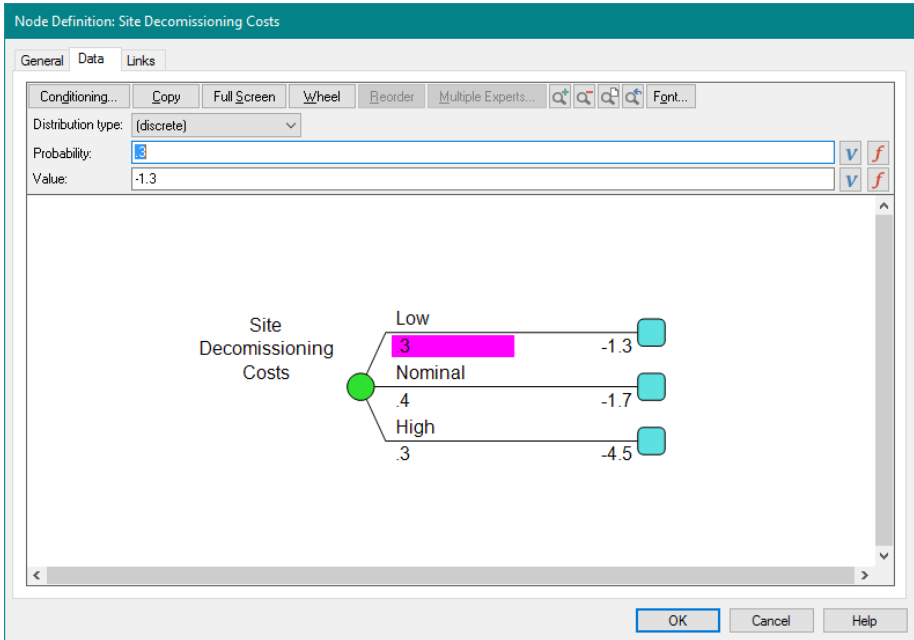
**Figure 7-14. Modified Decision Tree with Perform Reference Removed**

- ⇒ Select the Drilling Costs node and the blue endpoint for the Yes outcome of the Drill decision and click Decision Tree | Instance | Subtree.

Note that when adding new events to the tree via the Select Decision/Chance dialog, you also have the option of creating a new node and adding it to the tree. You'll add the Site Decommissioning Costs node to the Decision Tree using this method now.

- ⇒ Select Decision Tree | Instance | Add | Chance.
- ⇒ Within the Select Chance dialog, click the Create New button.
- ⇒ Place the new node on the blue endpoint that comes after the perform reference on the Yes branch of the Drill decision. This is where you will employ the perform and continue feature.
- ⇒ On the General tab, name the node "Site Decommissioning Costs" and leave the default outcomes.
- ⇒ Switch to the Data tab and enter the following values for the Low, Nominal, and High outcomes, respectively: -1.3, -1.7, and -4.5. Leave the default probabilities.

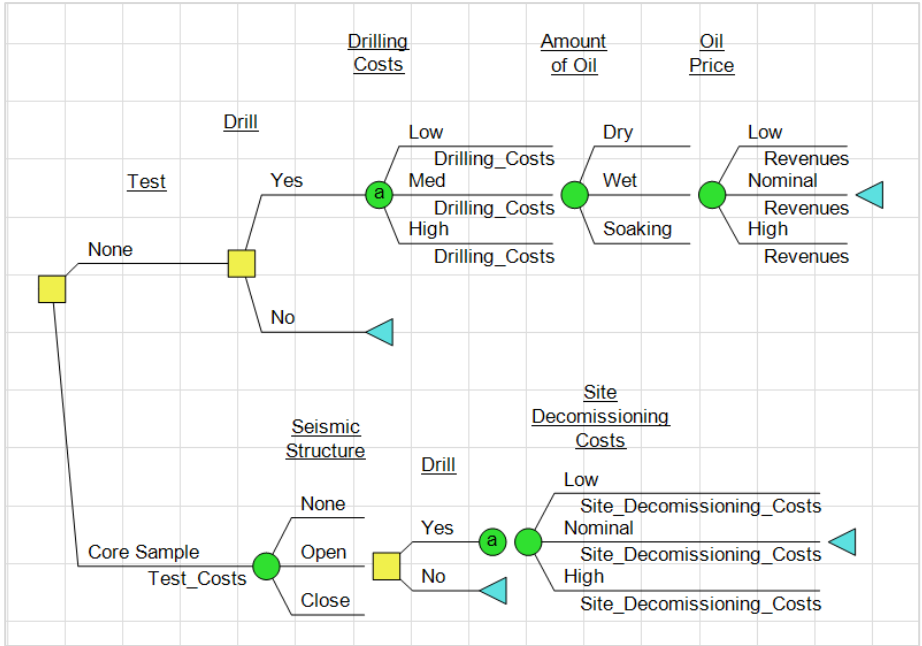
The data input tree should look like Figure 7-15.



**Figure 7-15. Data Input tree for Site Decommissioning Costs Node**

- ⇒ Click OK to close the Node Definition dialog.
- ⇒ Select the branches of the Site Decommissioning Costs node and drop-down the Edit Get/Pay list within the Decision Tree | Get/Pay group and select Site\_Decommissioning\_Costs.

Your Decision Tree is now complete and should look like Figure 7-16.



**Figure 7-16. Wildcat Decision Tree with a Perform and Continue**

- ⇒ Select Home | Run | Decision Analysis to generate an Initial Decision Alternatives chart and Policy Tree.
- ⇒ Click Yes to the warning.

You'll see from the Policy Tree (Figure 7-17) that the optimal decision policy is still to perform the test even with the added costs associated with decommissioning the site afterwards. But the gap between the two initial decision alternatives has narrowed to a slighter margin.

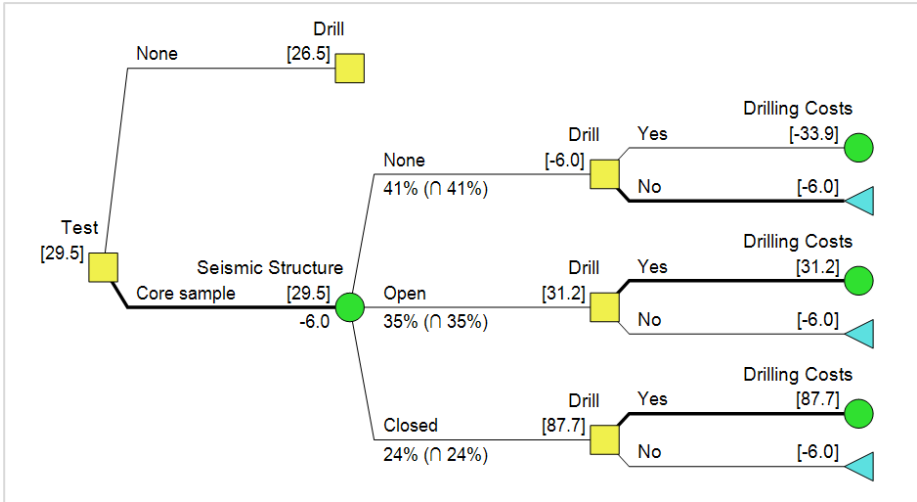


Figure 7-17. Policy Tree™ with Site Decommissioning Costs Added

### 7.3 Summary of DPL™ Influence Arcs

DPL influence arcs are color coded. Table 7-2 summarizes what each color means.





<b>Black</b>		No conditioning
<b>Blue</b>		Values conditioned
<b>Green</b>		Probabilities conditioned
<b>Orange</b>		Both values and probabilities conditioned

Table 7-2. Arc Color Meanings

Arcs appear magenta when selected, so you may need to press Esc to see the actual arc color.

The "from" node is known as the predecessor and the "to" node is called the successor. Because not all predecessors have states, and not all successors have probabilities, some arc types may not be available between a given pair of nodes.

Even when an arc is black and there is no conditioning, there can still be dependence in the calculations if the value of the predecessor node is used in a formula in the successor node. This is the case with the arc from Oil Price to Revenues in the Learning.da model.

Table 7-3 summarizes which type of arc is available between each of the node types in DPL.

		<b>To (successor)</b>				
		<i>Continuous chance</i>	<i>Discrete chance</i>	<i>Decision</i>	<i>Value</i>	<i>Controlled</i>
<b>From (predecessor)</b>	<i>Continuous chance</i>	Black	Black	Black	Black	Black
	<i>Discrete chance</i>	Black, orange	Black, green, blue, orange	Black, blue	Black, blue	Black, blue
	<i>Decision</i>	Black, orange	Black, green, blue, orange	Black, blue	Black, blue	Black, blue
	<i>Value</i>	Black	Black	Black	Black	Black
	<i>Controlled</i>	Black, orange	Black, green, blue, orange	Black, blue	Black, blue	Black, blue

**Table 7-3. Arc Types Available Between Node Types**

## 8. Incorporating Multiple Attributes and Objective Functions

Many decisions can be effectively modeled considering only a single measure of value, such as the net present value (NPV) of cash flows discounted at weighted average cost of capital (WACC). However, some decision problems, including many of those in the public sector, may require you to quantitatively model several measures of value, or you may find that including more than one measure is more enlightening. These value measures are referred to as attributes.

Attributes can be useful even when the focus of the analysis is financial. You may find it useful to look at both free cash flow and earnings, for example. When you wish to track financial measures or other measures over time, DPL's Time Series Percentiles analysis feature lets you use one attribute for each time period to do this; see Chapter 10 of this manual. In large DPL models where multiple attributes are required, DPL's multidimensional value node feature may also be useful; see Chapter 9 of this manual.

The tutorial in Section 8 addresses a decision in which the attributes are not all financial. Chapters 9 and 10 use examples of slightly more complex multiple attribute models in which the focus is financial.

DPL gives you the ability to model up to 1024 attributes. Some or all of these attributes can be weighted and combined in an objective function. DPL selects optimal decision alternatives based on objective function values, maximizing or minimizing as appropriate.

Note that the objective function should not be confused with a utility function for risk tolerance (see Chapter 15). The objective function combines several financial and/or other attributes into a single value, whereas a utility function typically adjusts a single financial value to account for risk aversion.

### 8.1 Incorporating Multiple Attributes

---

You'll begin with a model that has only a single, economic attribute -- Revenues. You'll add a second attribute to this model to account for environmental considerations.

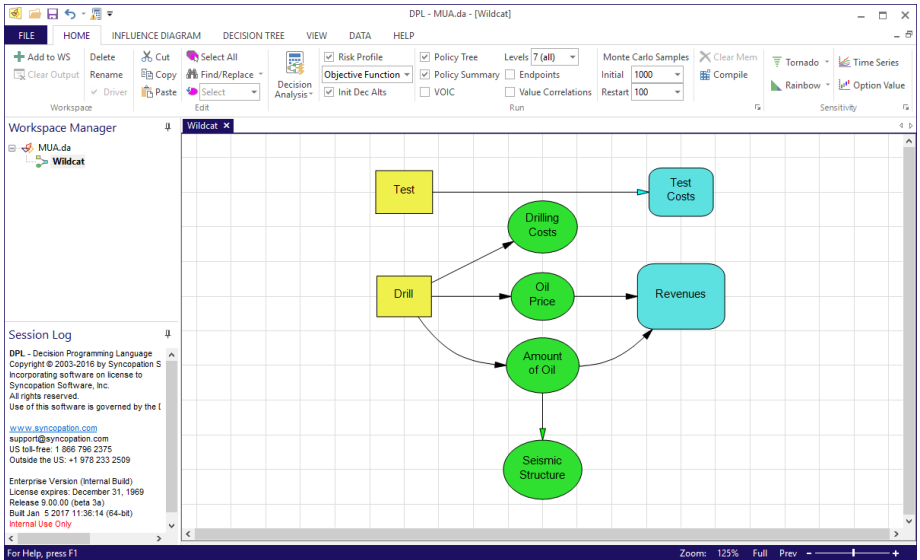


⇒ Open the example file MUA.da.

This file is found in the *Examples* folder where DPL was installed, usually C:\Program Files\Syncoption\DPL9\Examples.

The directory may vary slightly depending on your license type and operating system.

The MUA workspace should open in Influence Diagram-focused modeling mode as shown in Figure 8-1.

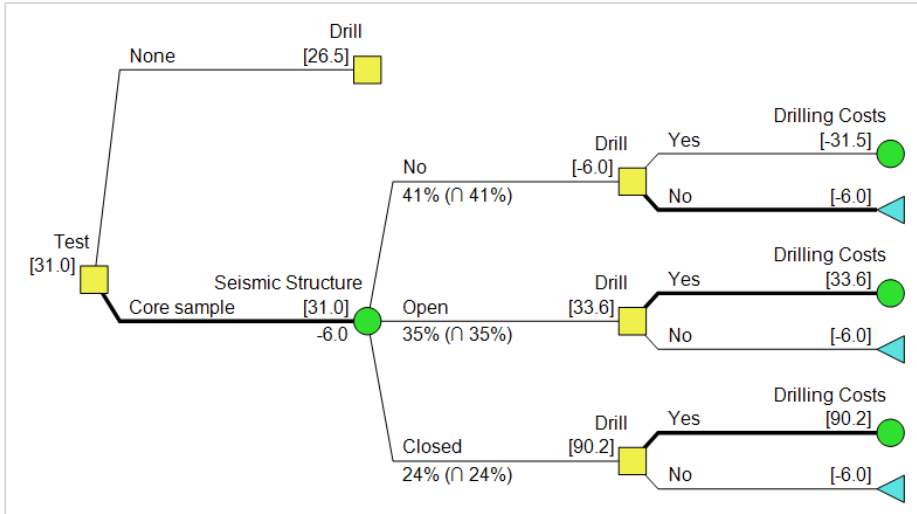


**Figure 8-1. Influence Diagram for Single Attribute Wildcat Model**

⇒ Make sure Policy Tree is checked.

⇒ Click Home | Run | Decision Analysis (or press F10).

Based on the single attribute of profit, the optimal decision policy is to perform the core sample test and then drill unless the outcome is no seismic structure. See Figure 8-2.



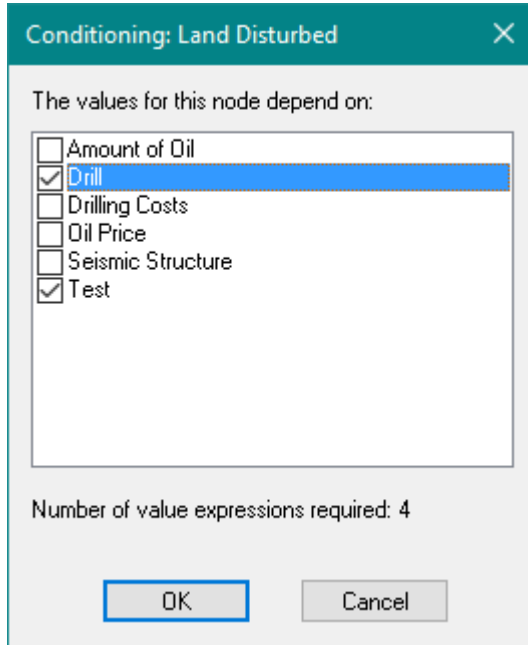
**Figure 8-2. Policy Tree™ for the Single Attribute Model**

The expected value of proceeding with the core sample test is \$31 million, whereas drilling without the test has an expected value of \$26.5 million.

⇒ Activate the Wildcat model by double-clicking its item in the Workspace Window or pressing Ctrl+F12.

The oilfield in question is in a pristine wilderness area. Testing and drilling would require that some areas be cleared and that an access road be built. To take this into account in your decision, you'll add an attribute representing the amount of land that would need to be disturbed.

- ⇒ To add a value node to the model, click Influence Diagram | Node | Add | Value.
- ⇒ Place the node above Test Costs.
- ⇒ Name it Land Disturbed.
- ⇒ Click the Data tab to enter data for the new node.
- ⇒ Click the Conditioning button.
- ⇒ Check Drill and Test. See Figure 8-3.

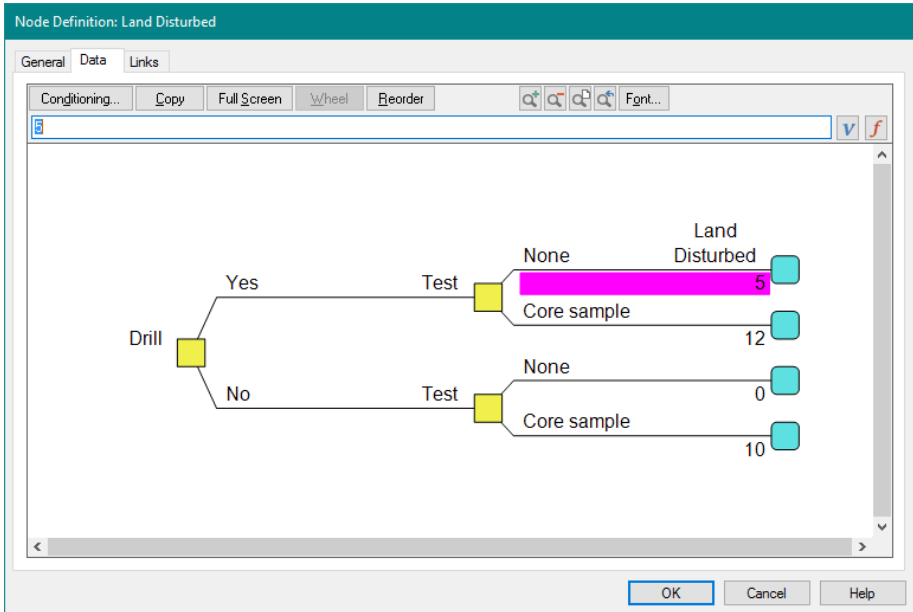


**Figure 8-3. Conditioning for Land Disturbed**

- ⇒ Click OK.
- ⇒ Enter data for Land Disturbed as indicated in Table 8-1. Your value input tree should look like Figure 8-4 when you are done.

	<b><u>Test</u></b>	<b><u>Land Disturbed Value</u></b>
<i>Drill</i>		
<i>Yes</i>	None	5
	Core Sample	12
<i>No</i>	None	0
	Core Sample	10

**Table 8-1. Data for Land Disturbed Value Node Value**



**Figure 8-4. Node Definition Dialog for Land Disturbed**

⇒ Click OK.

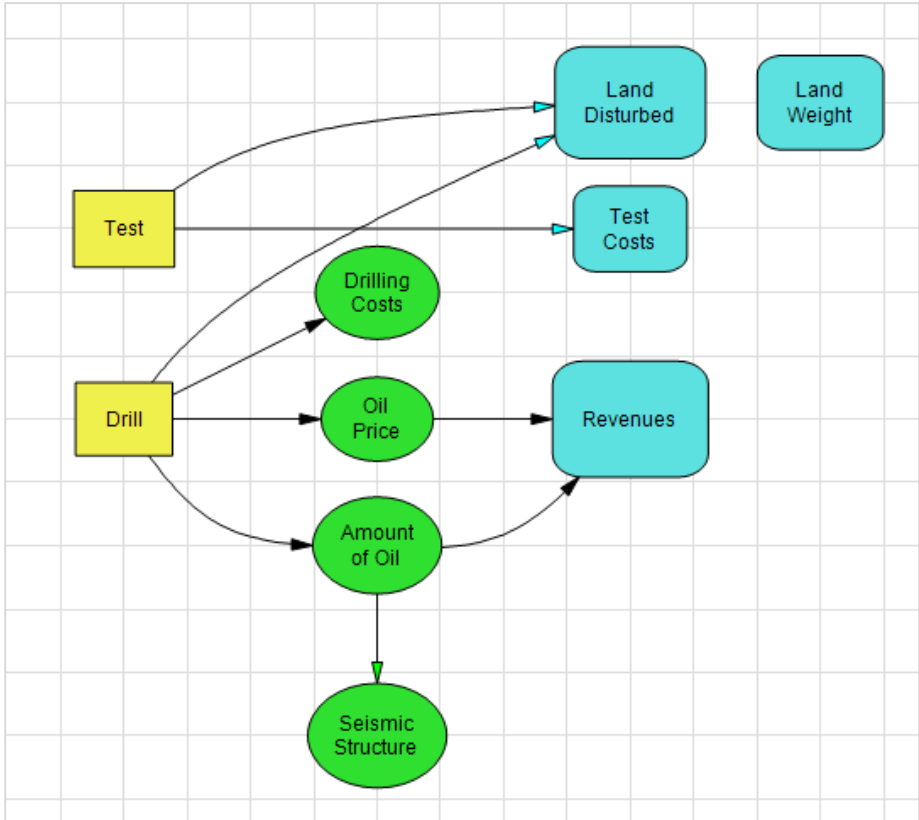
Note: don't be concerned that Drill comes before Test in the data input tree. The actual order of the nodes is dictated by the Decision Tree.

DPL creates new, blue influence arcs from Drill and Test to Land Disturbed. You may want to bend the arcs in your Influence Diagram for better readability.

Next, you'll create a new value node for the weighting of disturbing land.

- ⇒ Click Influence Diagram | Node | Add to add another value node to the model.
- ⇒ Place the node to the right of Land Disturbed.
- ⇒ Name it Land Weight.
- ⇒ Click the Data tab.
- ⇒ Type "-0.5" as the value.
- ⇒ Click OK.

Your Influence Diagram should look something like Figure 8-5. Next, you'll add an additional attribute for Land via the Objective & Utility dialog.






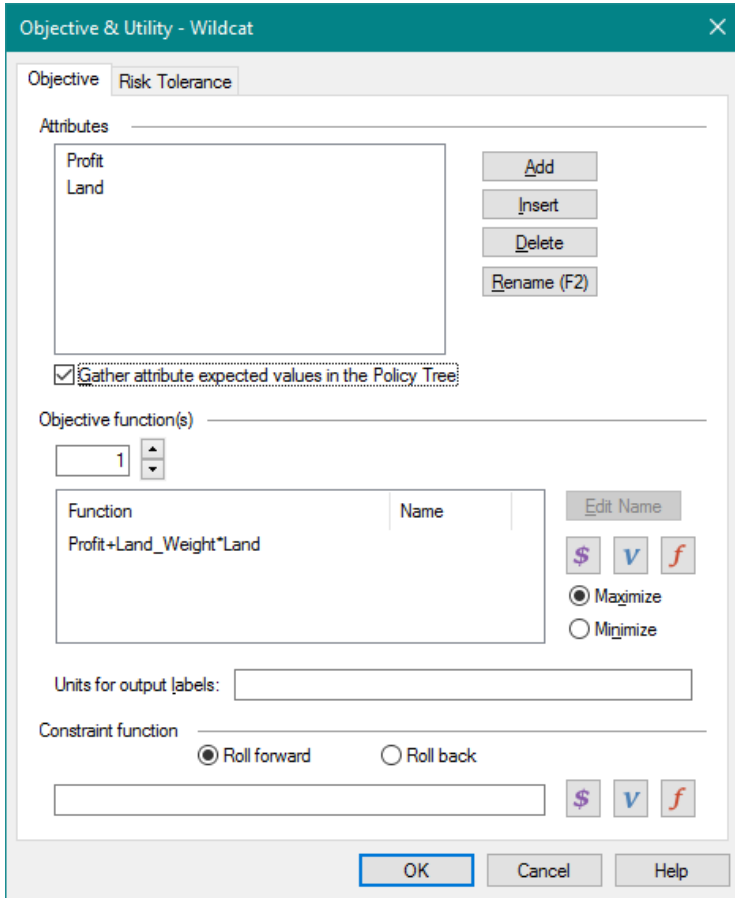
**Figure 8-5. Influence Diagram with New Nodes**

⇒ Click Influence Diagram | Objective & Utility | Settings. DPL displays the Objective tab of the Objective & Utility dialog.

Note: The Objective & Utility dialog shown in Figure 8-6 displays the look of the dialog in the DPL Enterprise version. If you're running DPL Professional the *Objective function* section will be simpler due to the fact that you can only define a single objective function. Whereas, within the DPL Enterprise version, multiple objective functions can be defined for a model.

- ⇒ Under the *Attributes* section, select Attribute1 and rename it to be "Profit".
- ⇒ Click Add to add a second attribute.
- ⇒ Rename the second attribute to be Land.

- ⇒ Under the *Objective function(s)* section, double-click under the Function column within the Objective function edit box to put it into text edit mode.
- ⇒ Click the Attribute button (  ).
- ⇒ Within the Select Attribute dialog double-click on Profit. It is added to the objective function and remains in edit mode.
- ⇒ Type a plus sign ("+") after it.
- ⇒ Click the Variable button (  ).
- ⇒ Double-click Land\_Weight.
- ⇒ Type an asterisk for multiplication ("\*") after it.
- ⇒ Click the Attribute button (  ).
- ⇒ Double-click Land and press Enter.
- ⇒ Check the checkbox next to *Gather attribute expected values in the Policy Tree*. Now, when you run your model and produce a Policy Tree the rolled-back expected values of all attributes can be displayed in the tree.
- ⇒ The Objective & Utility dialog should look like Figure 8-6.



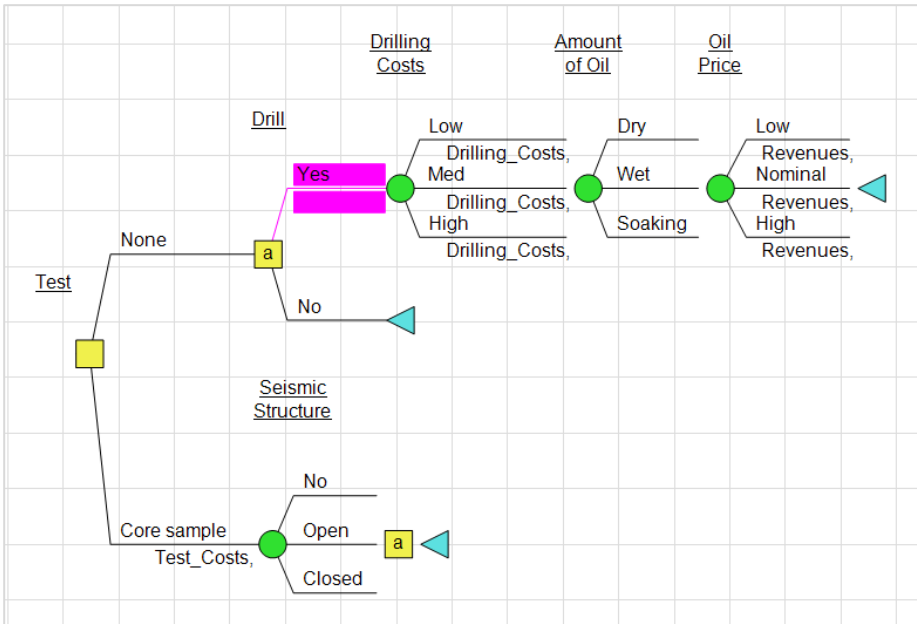
**Figure 8-6. Objective & Utility Dialog**

⇒ Click OK to close the dialog.

You may wonder why you didn't use the value Land Disturbed in the objective function. Land Disturbed will contribute to the Land attribute in a Get/Pay expression in the tree. Separating attributes and values gives you greater modeling flexibility. For example, you might later have several value nodes which all contribute to Land, just as you already have three value nodes that contribute to Profit (Test Costs, Drilling Costs, and Revenues).

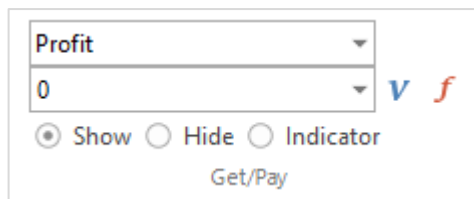
You'll now switch over to the Decision Tree in order to add Get/Pay expressions for the additional Land attribute – as a get/pay expression must be defined for each attribute in the model.

⇒ Press the Tab key to switch over to the Decision Tree pane view.



**Figure 8-7. Decision Tree with Yes Branch of Drill Decision Selected**

⇒ Select the Yes branch of the Drill decision as shown above in Figure 8-7 and take note of the edit boxes within the Decision Tree | Get/Pay group (Figure 8-8).



**Figure 8-8. Decision Tree | Get/Pay Group**

The top drop-down box shown in Figure 8-8 is referred to as the Select Attribute edit box. For single attribute models this box is set to Objective Function and disabled. This makes sense because for single attribute models all get/pay expressions in the tree must apply to the objective function.

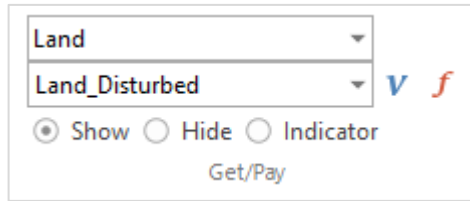
For the current multiple attribute model, the box is enabled and provides a means to specify the attribute for which you'd like to edit the get/pay



expression for the selected branch. Once the attribute is selected, you can then enter a quantity or expression within the lower Edit Get/Pay combo box.

In this case you're editing the get/pay expression for the Profit attribute for the Yes branch of the Drill decision. You don't need to contribute anything to Profit on this branch, so you'll keep the Edit Get/Pay box set to 0.

- ⇒ With the Yes branch of the Drill decision still selected, chose the Land attribute from the Select Attribute drop-down list.
- ⇒ Drop-down the lower Edit Get/Pay combo box and select Land\_Disturbed from the list. The Get/Pay box should look as they do in Figure 8-9.



**Figure 8-9. Decision Tree | Get/Pay Group for Land Attribute**

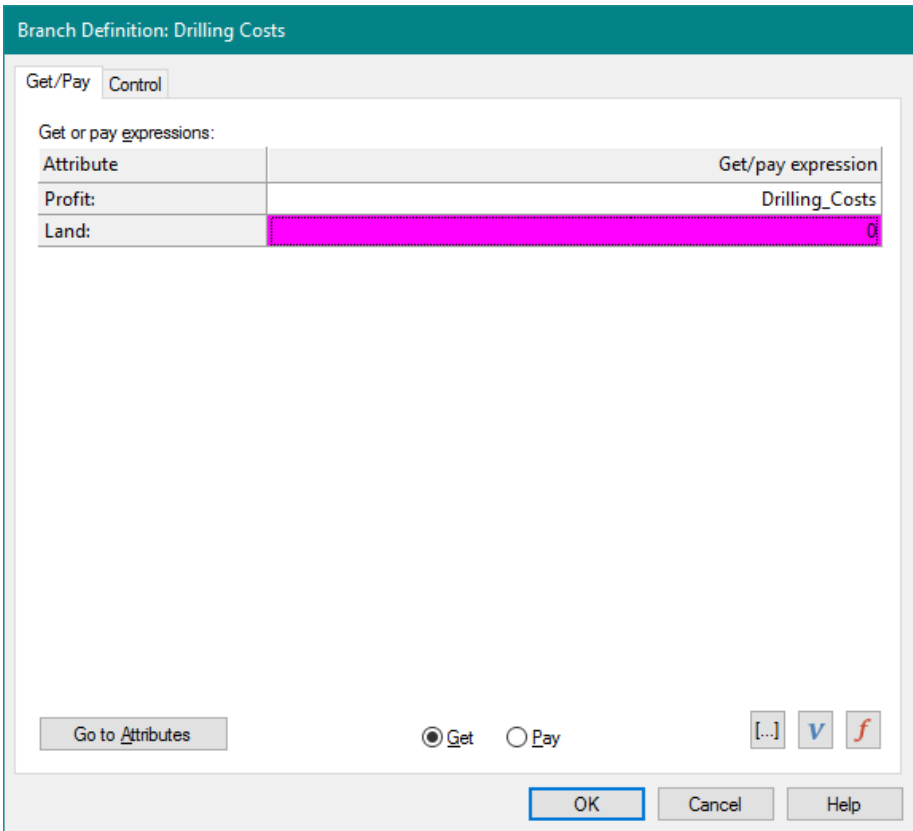
The Get/Pay expression on the Yes branch of Drill now accounts for both the Profit and Land attributes.

Note: Using the ribbon method for specifying get/pay expressions for attributes may be faster but only allows you to see the Get/Pay expression for a single attribute at a time. Whereas, if you to double-click the branch(es) in the tree to launch the Branch definition dialog, you are able to view the expressions for all the attributes simultaneously. You'll use the Branch Dialog for the remaining get/pay expressions.

You also need to change the other Get/Pay expressions, this time adding a 0 placeholder for the Land attribute instead of the Profit attribute.

- ⇒ Double-click on the branches of Drilling Costs.

- ⇒ Leave the get/pay for Profit as is. Type "0" zero in the cell for the Land attribute. The Branch Definition should match Figure 8-10.



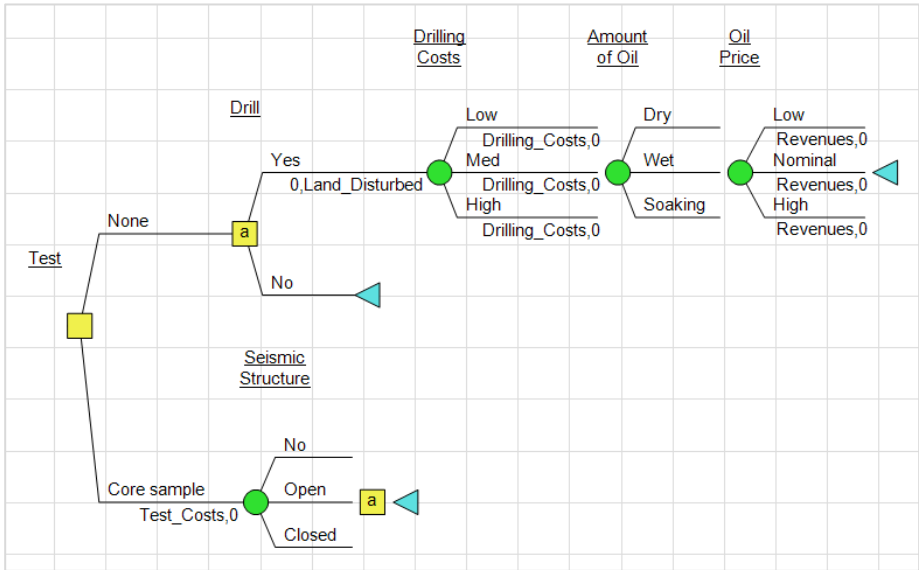
**Figure 8-10. Branch Definition Dialog for Drilling Costs**

- ⇒ Click OK.

Note: When you edit a Get/Pay expression if you leave a cell blank for an attribute, DPL will fill in a zero when you click OK to close the Get/Pay Definition dialog. Try this now.

- ⇒ Double-click on the branches of Oil Price.
- ⇒ Click OK. Note that zero is filled in for Land in the Get/Pay in the Decision Tree.
- ⇒ Repeat the above process for the Get/Pay on the Core Sample branch of the Test decision.

You have created a two-attribute model with an objective function that linearly combines the attributes. Your tree should look like Figure 8-11.

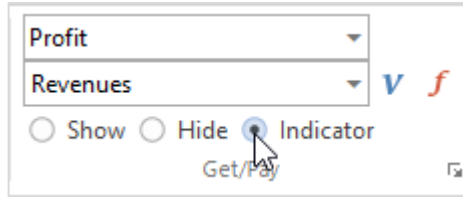


**Figure 8-11. Decision Tree for Model with Two Attributes**

When you change Get/Pay expressions in the Decision Tree, by default DPL expands the branch length of the node where the Get/Pay expression appears. You may not have noticed within this model because the Get/Pay expressions are not particularly lengthy. If you do have lengthy Get/Pay expressions, which you often will when you define multiple attributes, you may end up with a tree that does not display easily in its entirety on a screen.

You can change what is displayed for Get/Pay expressions in the Decision Tree via the Decision Tree | Get/Pay group, with the options being: show the full Get/Pay (*Show*), hide the Get/Pay completely (*Hide*), or display an indicator (\$) to indicate the presence of a get/pay (*Indicator*). The radio button is set to *Show* by default. It's not necessary for this model, but nevertheless a useful feature to be familiar with.

- ⇒ Select the branches of the Oil Price node in the Decision Tree.
- ⇒ Select the *Indicator* radio button in the Decision Tree | Get/Pay group. See Figure 8-12.



**Figure 8-12. Get/Pay Display Options within Decision | Get/Pay Group**

The full display of the get/pay expression of the branches of Oil Price has been replaced by a "\$" symbol, indicating the presence of a get/pay.

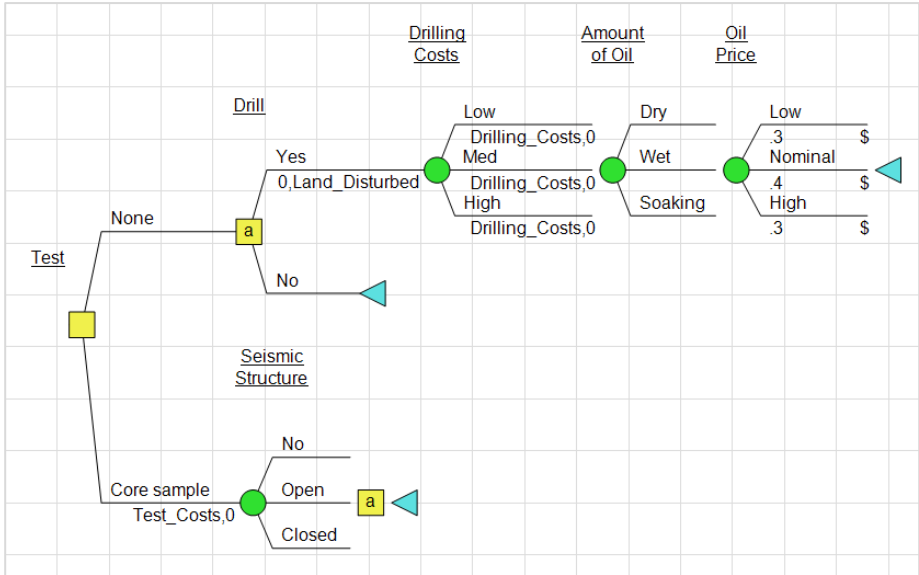
There are a few other useful display options that can be set on a per instance basis within the Decision Tree. To access these settings:

- ⇒ Double-click the Oil Price node. The Tree Instance tab of the Node Definition dialog appears for the node.

Within the *Display Options* section note that you can remove the node name and/or state names for the node instance. You can further underline and/or put a border around the node name. Lastly, you can choose to display the constant probabilities on the branches of uncertainties. Do this now:

- ⇒ Check the box next to "Show constant probabilities" and click OK.

Your Decision Tree should now look like Figure 8-13. Notice that the probabilities for each outcome of the Oil Price chance node are displayed on their respective branches.



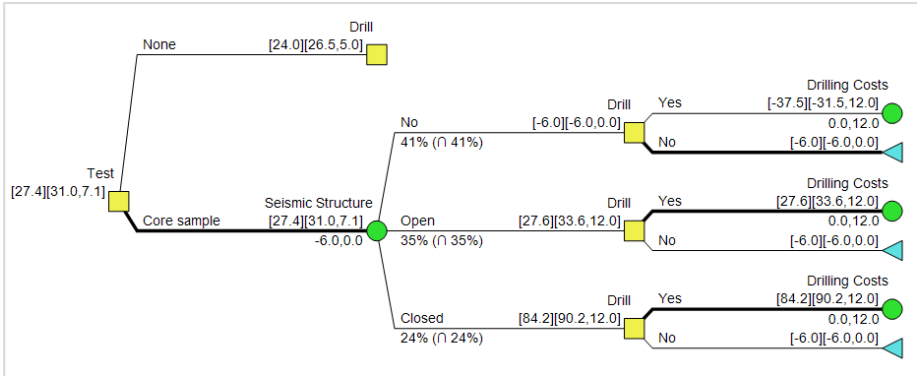
**Figure 8-13. Oil Price Instance with Get/Pay Indicator and Constant Probabilities Displayed**

You are now ready to run your model. Remember, with the *Gather attribute expected values in the Policy Tree* checkbox checked within the Objective & Utility box, DPL will produce a Policy Tree displaying the expected value of the attributes at each point in the Policy Tree.

- ⇒ Make sure Policy Tree is checked.
- ⇒ Click Home | Run | Decision Analysis.
- ⇒ Click OK to the warning.

DPL will warn you that the objective function and utility functions must be a linear combination of the attributes for sequence evaluation to work properly. If they are not, use Full Tree Enumeration as the evaluation method.

- ⇒ Click Yes. The Policy Tree will look like Figure 8-14.



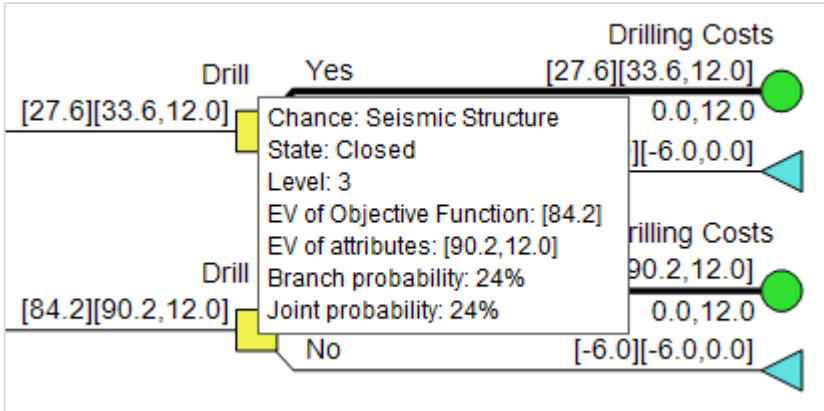
**Figure 8-14. Policy Tree™ for the Two Attribute Model with Attribute Expected Values Displayed**

The expected values of both alternatives in Figure 8-14 are lower than in the original Policy Tree (Figure 8-2), reflecting a "penalty" of \$0.5 million per unit of land. The Core sample Test is still the preferred alternative, although the gap has narrowed.

The tree in Figure 8-14 illustrates how DPL's Policy Tree can display attribute expected values. At each node along the tree, two sets of values are now shown in brackets: in the first set of brackets is the expected value of the objective function, in the second set of brackets is the expected value of each of the two attributes. For example, if the optimal policy (Core Sample) is followed and Seismic Structure is Open, the expected value of the objective function at that point is 27.6; the expected values of the Profit and Land attributes are 33.6 and 12.0, respectively.

If you did not want to see the expected value of each attribute at every node, you can turn off their display by unchecking the Policy | Display | Show Attribute EVs checkbox. If there is a specific subset of attributes you'd like displayed, you can specify these on an individual basis within the Format Policy Display dialog (Policy | Display | Settings). See Section 3.1 for more on Policy Tree display options.

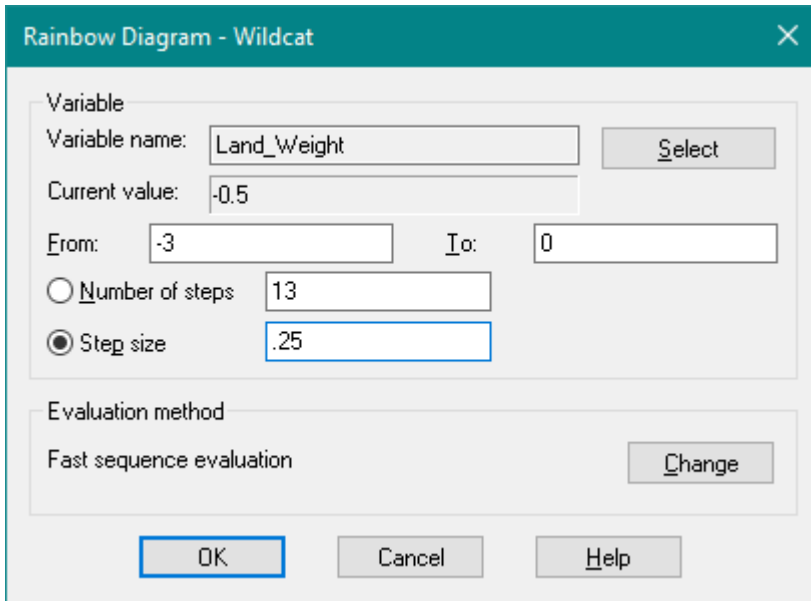
You can hold your mouse cursor over any node to see the expected values associated with that node more clearly in a tips box. See Figure 8-15.



**Figure 8-15. Policy Tree™ Displaying Attributes and Tips**

Finally, you will run a Rainbow Diagram on the Land Weight value. You want to know how much more penalty there would need to be associated with land disturbed for the optimal alternative to change, i.e., how much more negative Land Weight would need to be.

- ⇒ Click Home | Sensitivity | Rainbow Diagram (One-way). The Rainbow Diagram dialog appears.
- ⇒ Click Select. The Select Value for Rainbow dialog appears.
- ⇒ Select Land\_Weight from the Value for sensitivity drop-down list.
- ⇒ Click OK.
- ⇒ Type "-3.0" for From.
- ⇒ Type "0.0" for To.
- ⇒ Type "0.25" for Step size. The Run Rainbow Diagram dialog should look like Figure 8-16.



**Figure 8-16. Run Rainbow Diagram Dialog**

⇒ Click OK.

The Rainbow Diagram (Figure 8-17) shows that Land Weight would have to be less than -2 for the decision policy to change. This means the policy of testing before deciding to drill is fairly robust – there would have to be a substantial change in the attribute weighting for it to change.



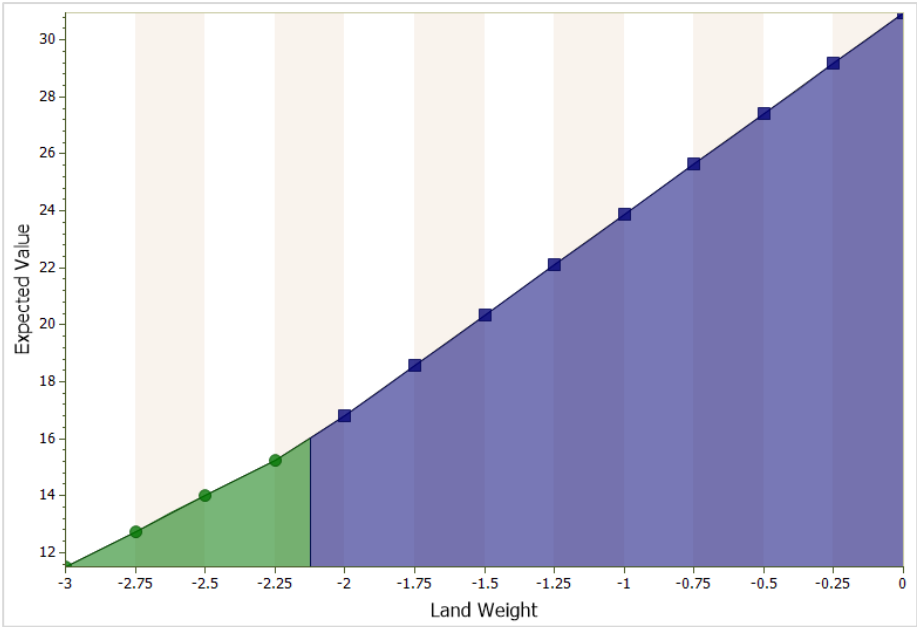


Figure 8-17. Rainbow Diagram on Land Weight

## 8.2 Using a Constraint Function

The objective function gives you considerable flexibility in defining how much you like/dislike each attribute, but there may still be situations where you want to place an absolute limit on one or more attributes.

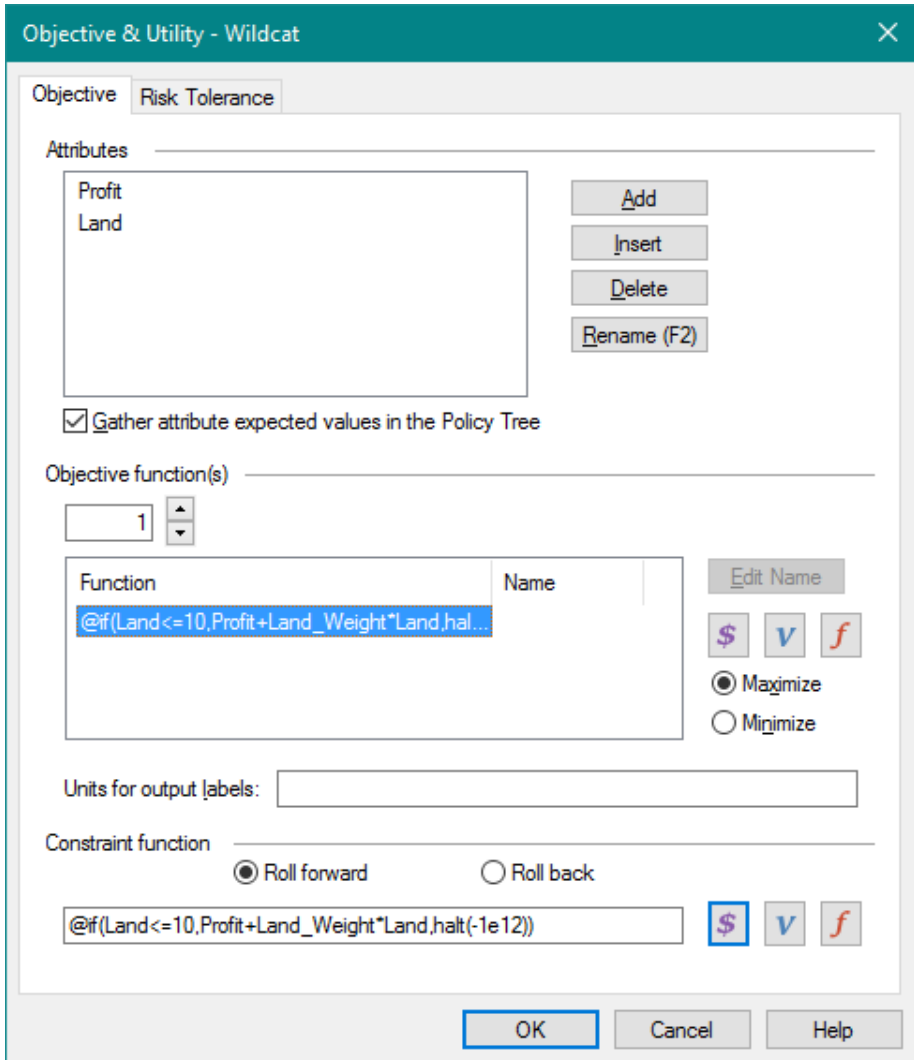
In DPL, you do this by specifying a constraint function. A constraint function is normally an @if statement with a "halt" function for its False value. The halt function prevents DPL from continuing down that path. This serves to limit the size of the tree as well as enforce the constraint. The syntax for the @if function is @if(condition, True value, False value).

- ⇒ Activate the Wildcat model.
- ⇒ Open the Objective & Utility dialog (Decision Tree | Objective & Utility | Settings). The Objective tab of the Objective & Utility dialog will appear.
- ⇒ Click in the Constraint function edit box.
- ⇒ Type "@if(Land<=10,Profit+Land\_Weight\*Land,halt(-1e12))".

DPL is case sensitive, so make sure your spelling, capitalization and punctuation are correct.

When a constraint function is used, the objective function and the constraint function will normally be the same.

⇒ Copy the constraint function up to the objective function edit box. See Figure 8-18.

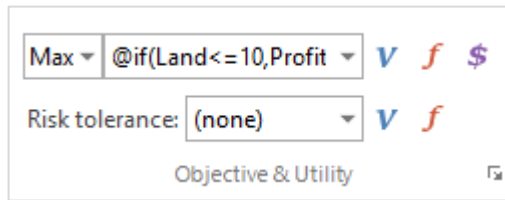


**Figure 8-18. Objective & Utility dialog with Constraint Function**

When the halt function is evaluated, DPL will stop evaluating the current Decision Tree path and will insert an expected value of - 1e12. In other words, it would cost you one million trillion to violate the constraint, thus enforcing that Land attribute will never be greater than 10.

⇒ Click OK.

Notice that the new objective function is displayed in the Objective Function combo box in the Decision Tree | Objective & Utility group (Figure 8-19). In addition to the Objective & Utility dialog, you can edit the objective function using the Objective & Utility | Objective Function combo box on the Influence Diagram or Decision Tree tabs either by selecting a single attribute from the drop-down list or by typing an expression involving more than one attribute and/or value from the model into the edit box. The select attribute, value and function buttons can be used for these purposes.



**Figure 8-19. Influence Diagram/Decision Tree | Objective & Utility Group**

⇒ Click Home | Run | Decision Analysis.

⇒ Click OK to the warning.

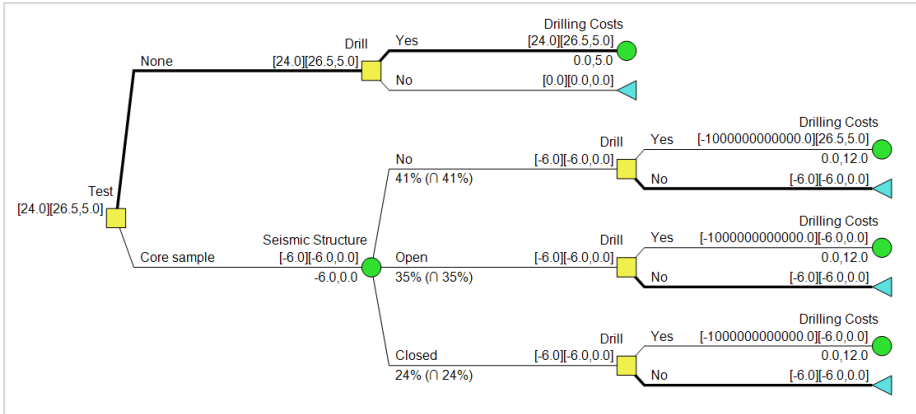
DPL will warn you that for fast sequence evaluation to work properly, the conditions under which the "halt" function is executed must depend only on attributes or constant values.

⇒ Click Yes to the warning.

The constraint has changed the optimal decision alternative. If you were to conduct the Core sample test and then decide to drill, the land constraint would be violated.

⇒ In the Policy Tree, select the Seismic Structure node and drop-down the Policy | Expand split. Select Expand Subtree from the list to expand that section of the tree.

You may need to Zoom Full to see the expanded Policy Tree. See Figure 8-20.



**Figure 8-20. Policy Tree™ for Model with Constraint**

DPL cannot expand the subtree for Seismic Structure further beyond Drilling Costs because the halt function prevented DPL from continuing to evaluate that path. If you double-click on Drilling Costs, the Policy Tree will not expand further.

## 8.3 Modeling Multiple Objective Functions (Enterprise only)

Similar to the case with attributes, most decision-problems can be aptly modeled by maximizing (or minimizing) a single objective function, e.g. maximizing the NPV of a new product development opportunity or minimizing the mortality rates in the development of a medical device. However, there are situations where you may want to take into account the actions of a partner or adversary that makes one or more of the decisions in the model. These agents may have objectives that are different than your own.

For example, imagine you are a large pharma company that is seeking to buy a compound from a small firm. Both of you want the project to succeed but your partner will likely have different financial incentives from your own. The Enterprise version of DPL 9 offers the ability to optimize multiple objective functions within a single model to better model the actions of partners and adversaries.

In the tutorial that follows, you will modify a multi-attribute model that analyzes an airport defense decision. More specifically the model includes a

single decision: which, if any, airports to defend (none, top 10, or top 50) against attacks. The single objective function of the model is to minimize fatalities and costs. You will make the necessary changes to the model to incorporate a new decision made by the adversary along with their underlying objectives.

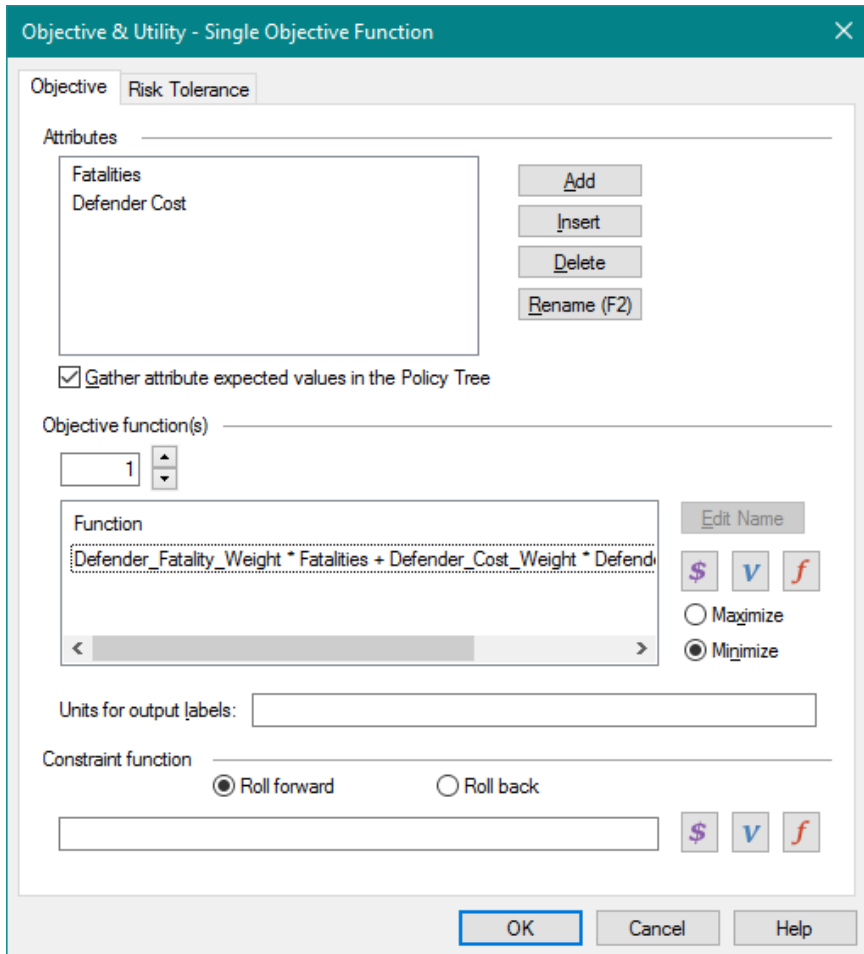
If you have the DPL Professional version you will not be able to work through the tutorial that follows. If you have a license of DPL Professional and would like to use the multiple objective function feature, contact Syncopation's sales team at [sales@syncopation.com](mailto:sales@syncopation.com) to discuss feature upgrade options.

Note that the objective function should not be confused with a utility function for risk tolerance (see Chapter 15). The objective function combines several financial and/or other attributes into a single value, whereas a utility function typically adjusts a single financial value to account for risk aversion.

- ⇒ Open the model Airport Defense.da from the Examples folder of your DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples.

Take a moment to look over the model.

- ⇒ Open the Objective & Utility dialog (Figure 8-21) to view the attributes and objective function of the model.

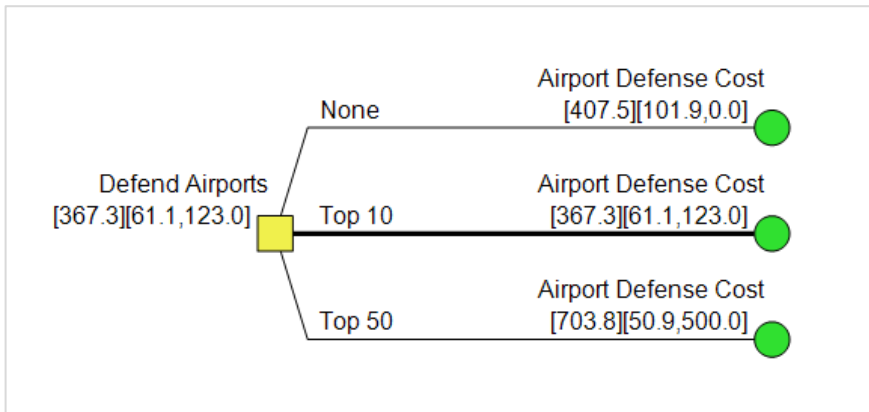


**Figure 8-21. Objective & Utility dialog for Airport Defense Model with Single Objective Function**

There are 2 attributes defined in the model: Fatalities and Defender Costs. If you look closely at the formula for the objective function, you'll find that it is the weighted sum of the attributes. Note further that the radio button for the objective function is set to *Minimize*. The defender obviously would like to prevent attacks, and consequently, avoid fatalities. While the decision maker would ideally want to defend all airports in order to avoid any and all fatalities – that simply isn't economically feasible in the real world. So the decision maker must make the best possible decision while taking into account the tradeoff between the costs associated with defending the airports and the potential loss of life if an attack were to be

carried out. You'll now run the model and take a look at the resulting Policy Tree.

- ⇒ Make sure Policy Tree is the only output selected within the Home | Run group.
- ⇒ Click Home | Run | Decision Analysis analyze the model and generate the results.
- ⇒ Click OK to the fast sequence evaluation warning. The Policy Tree looks like Figure 8-22.



**Figure 8-22. Policy Tree for Airport Defense Model with Single Objective Function**

The optimal decision policy is to defend the top 10 airports in the country. The Policy Tree displays the expected value for each attribute in brackets to the right of the expected value of the objective function. The expected value of the Fatalities attribute is listed first. You can see that you will have a few less fatalities (51 vs. 61) if you were to defend the top 50 airports. But look at the values for the Defender Cost attribute. Saving those lives would cost 400% more than defending just the top 10 airports. It's certainly a difficult decision to be made. Let's examine further down the tree.

- ⇒ Right-click on the Airport Defense Cost chance node at the end of the Top 10 decision alternative and choose Expand to Level...
- ⇒ Within the Expand Policy Tree dialog enter "5" for the level.

Now you can see through to the Attack Airport uncertainty. While the outcome to this variable is uncertain to you – it's really a decision that will be made by the attacker. Underlying this decision are the attacker's own

set of objectives that would clearly conflict with your own. To gain a better picture of the problem, you will incorporate the attacker's decision into the model and set up additional attributes that will be combined into a second objective function for the attacker.

- ⇒ Activate the Defender model and then duplicate it within the Workspace Manager.
- ⇒ Re-name the new model to be "Attacker/Defender".

First thing you'll do is change the Attack Airport discrete chance node into a decision node.

- ⇒ Select the Attack Airport node and choose Influence Diagram | Node | Change to | Decision.
- ⇒ DPL will tell you that the node you want to change conditions other nodes and that data will be lost for those nodes. Click OK to continue.

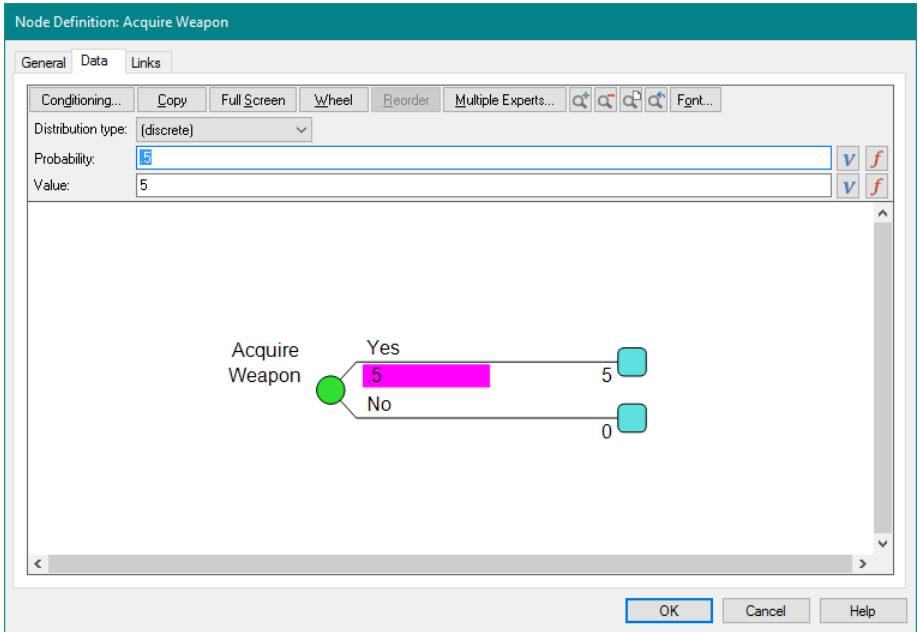
Experts in the field provided information on the metrics the Attackers would seek to optimize with an attack: attackers typically seek to maximize both fatalities and media exposure. Additionally, the attacker does not have infinite resources for financing attacks so minimizing costs should also be part of the objective function. The appropriate weights will be assigned to these attributes in order to form an overall quantity to maximize or minimize for the model. You'll add the necessary nodes to the model to account for these variables.

- ⇒ Add a discrete chance node to the model. Place it above the Attack Airport decision.
- ⇒ Name the node "Attack Cost" on the General tab of the Node Definition dialog.
- ⇒ Enter the values of 10, 12, and 15 for the Low, Nominal, and High outcomes, respectively. Leave the default probabilities as they are.
- ⇒ Click OK to close the Node Definition dialog.

There will also be a cost incurred by the attacker if they were to acquire a weapon. You'll enter this cost within the node data for Acquire Weapon.

- ⇒ Double-click the Acquire Weapon node to edit it.
- ⇒ On the Data tab, enter 5 for the value input for the Yes outcome and 0 for the No outcome (Figure 8-23).





**Figure 8-23. Updated Data Input Tree for Acquire Weapon node**

⇒ Click OK to close the Node Definition dialog.

You already have a variable for fatalities in the model but not for media exposure. You'll add this now.

⇒ Click Influence Diagram | Node | Add | Discrete Chance.

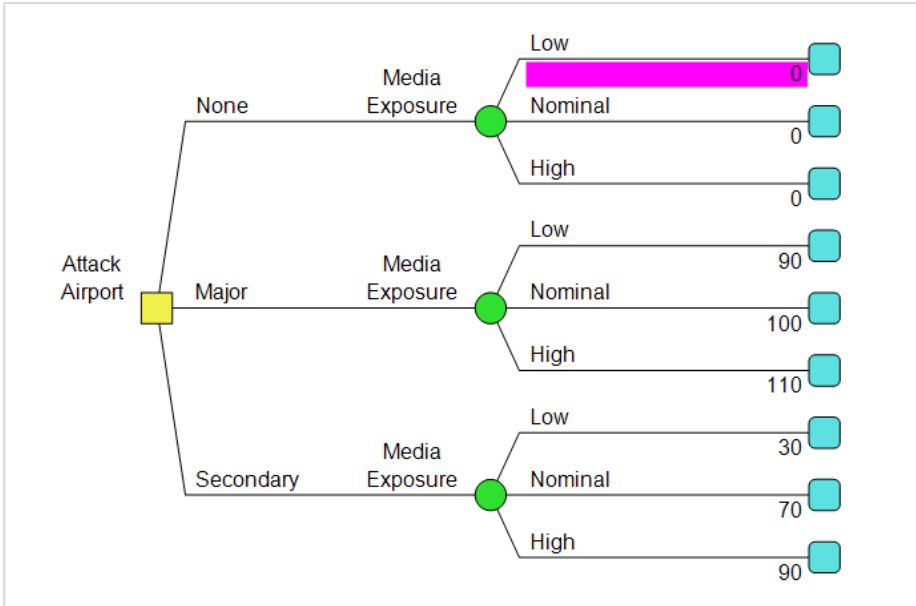
⇒ Place the node below Fatalities and name it "Media Exposure".

⇒ Check the *Separate probability and value data input tree* box.

The level of media exposure is dependent upon the whether the attack is carried out and, if it is, whether it occurs at a major airport or a secondary location.

⇒ On the Values tab, click the Conditioning button. Select Attack Airport from the list. You will now have 9 inputs to define.

⇒ Enter the values to match what is shown within the data input tree in Figure 8-24.



**Figure 8-24. Value Input Tree for the Media Exposure Chance Node**

⇒ Click OK to close the Node Definition dialog.

Prior to defining the additional attributes and objective function you'll need to establish weights for the fatalities, media exposure and costs for the attacker.

⇒ Select the Defender Cost Weight node and press Ctrl+C to copy it to the clipboard.

⇒ Press Ctrl+V to paste the copied node. Move it to the right of the Defender Cost Weight node.

⇒ Double-click the new node to edit it.

⇒ Re-name the node to be "Attacker Cost Weight".

⇒ Select the Data tab. Note that a value of 1 has been entered. This is what you want so click OK to close the Node Definition dialog.

⇒ Copy the Defender Fatality Weight node, paste it, and move it the right.

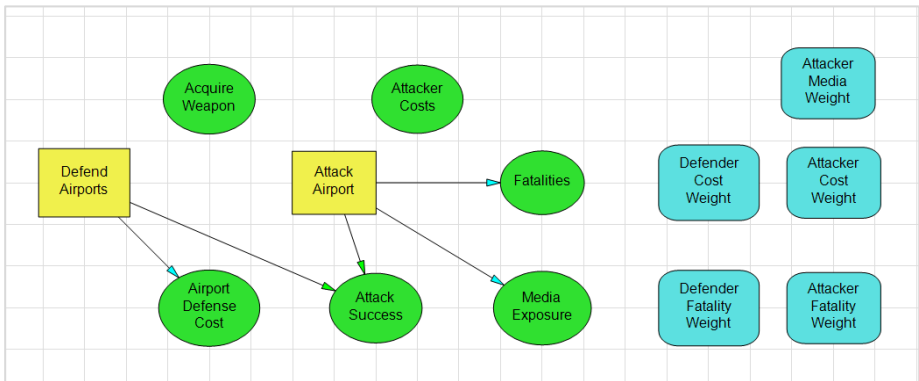
⇒ Edit the new node so it's named "Attacker Fatality Weight".

⇒ Within the Data tab, you'll find a value of 4 has been entered. Change this value to a "1". Press OK to close the Node Definition dialog.

Now you'll add one last weight to the model for the media exposure variable.

- ⇒ Choose Influence Diagram | Node | Add | Value and place it above the Attacker Cost Weight Node.
- ⇒ Name the new value node "Attacker Media Weight".
- ⇒ On the Data tab, enter a value of "1". Click OK to close the Node Definition dialog.

In this model, the attacker puts equal weight on fatalities and media exposure. Your Influence Diagram should now look like Figure 8-25.



**Figure 8-25. Influence Diagram with Attacker Decision and Weights Incorporated**

Now you'll add two more attributes to the model: one for Media Exposure and one for Attacker Costs.

- ⇒ Open the Objective & Utility dialog.
- ⇒ Within the *Attributes* section of the dialog click the Add button to add a new attribute.
- ⇒ Name it "Media Exposure".
- ⇒ Click the Add button again to add another attribute.
- ⇒ Name it "Attacker Cost".

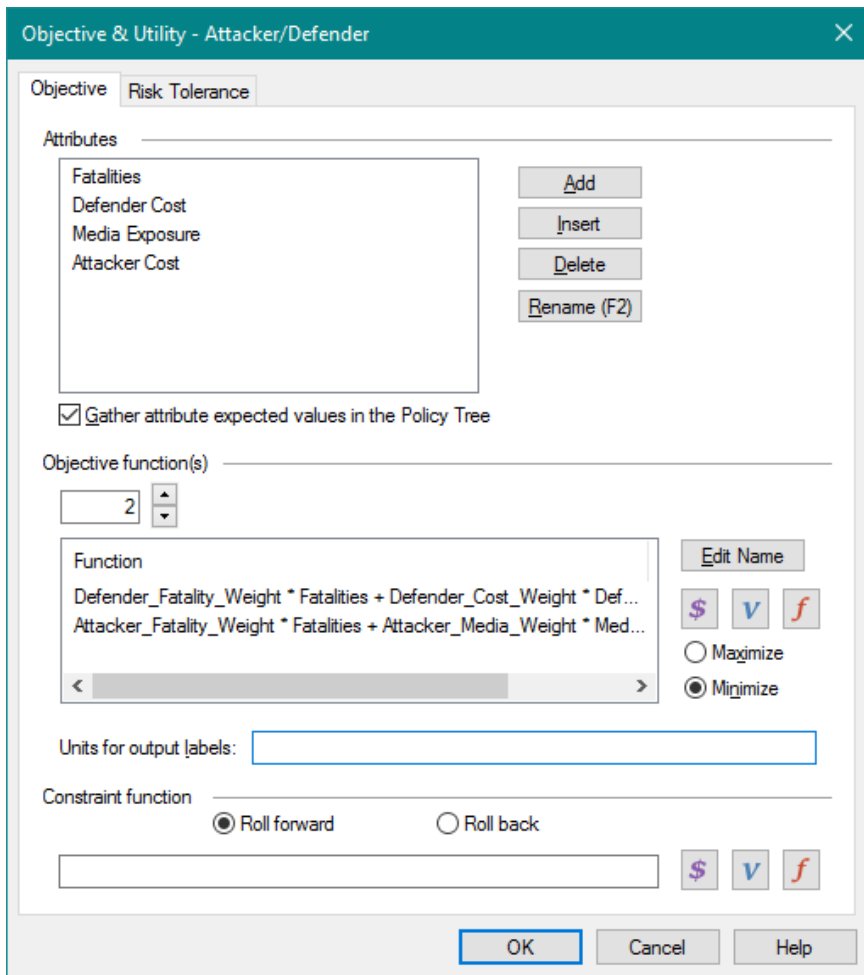
Now that the necessary variables and attributes have been added you can define the second objective function for the model.

- ⇒ Just beneath the *Objective function(s)* heading use the spin box to increase the number of objective functions to 2.

⇒ Enter the following formula for this second objective function using the Select Attribute and Variable buttons:

$$\text{Attacker\_Fatality\_Weight} * \text{Fatalities} + \text{Attacker\_Media\_Weight} * \text{Media\_Exposure} - \text{Attacker\_Cost\_Weight} * \text{Attacker\_Cost}$$

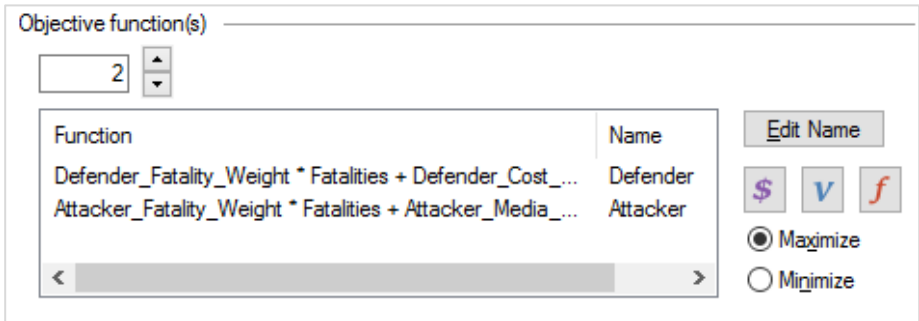
Note that by default DPL has selected the Maximize radio button for the newly defined objective function. In this case, the attacker is seeking to maximize this metric so you'll leave that as it is. The Objective & Utility dialog should look like Figure 8-26.



**Figure 8-26. Objective & Utility dialog with Second Objective Function Added**

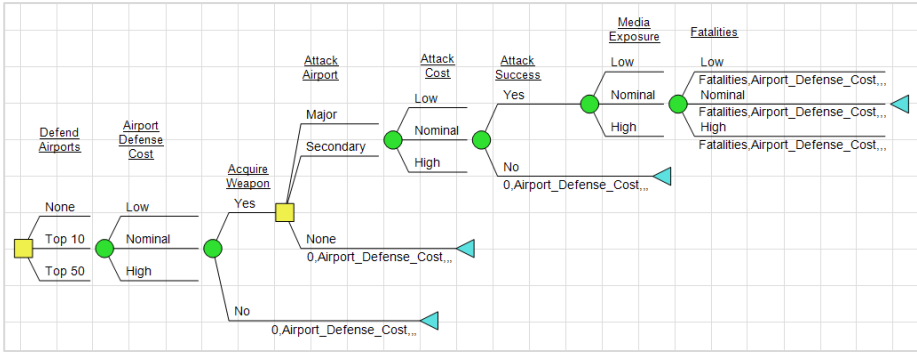
Now that two objective functions are defined, you'll want to assign a name to each of them for clarity.

- ⇒ Select the first objective function listed (it should be the formula that includes the Defender weights) and click the Edit Name button.
- ⇒ Within the Enter Name dialog type "Defender". Click OK to close the dialog.
- ⇒ Do the same for the second objective function listed, naming it "Attacker". The objective function list box should now look like Figure 8-27



**Figure 8-27. Objective Function list box with Two Objective Functions Defined**

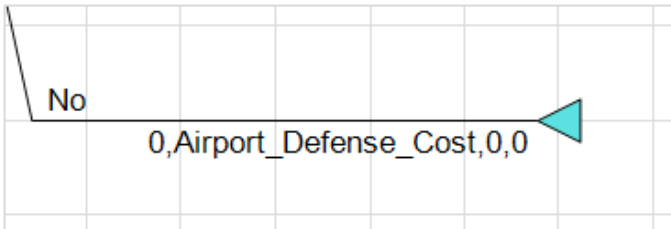
- ⇒ Click OK to close the Objective & Utility dialog.
- You'll now modify the Decision Tree to incorporate these new variables.
- ⇒ Switch over to the Decision Tree pane.
  - ⇒ Choose Decision Tree | Instance | Add | Chance...
  - ⇒ Select Attack Cost from the Select Chance variable list.
  - ⇒ Place the node on top of the Attack Success chance node so that it follows the Attack Airport decision.
  - ⇒ Choose Decision Tree | Instance | Add | Chance... again and select Media Exposure from the list.
  - ⇒ Drop it on the Fatalities chance node so that it is placed just before it. Your Decision Tree should now look like Figure 8-28.



**Figure 8-28. Decision Tree with Attacker Variables added**

Recall that you need to define a get/pay expression for each attribute. You'll add expressions to the branches that contain a get/pay expression for the two additional attributes.

- ⇒ Double-click on the No branch of the Acquire Weapon chance node. Click OK to close the dialog. DPL will supply zeroes to the blank cells for the Media Exposure and Attacker Cost attributes (Figure 8-29).



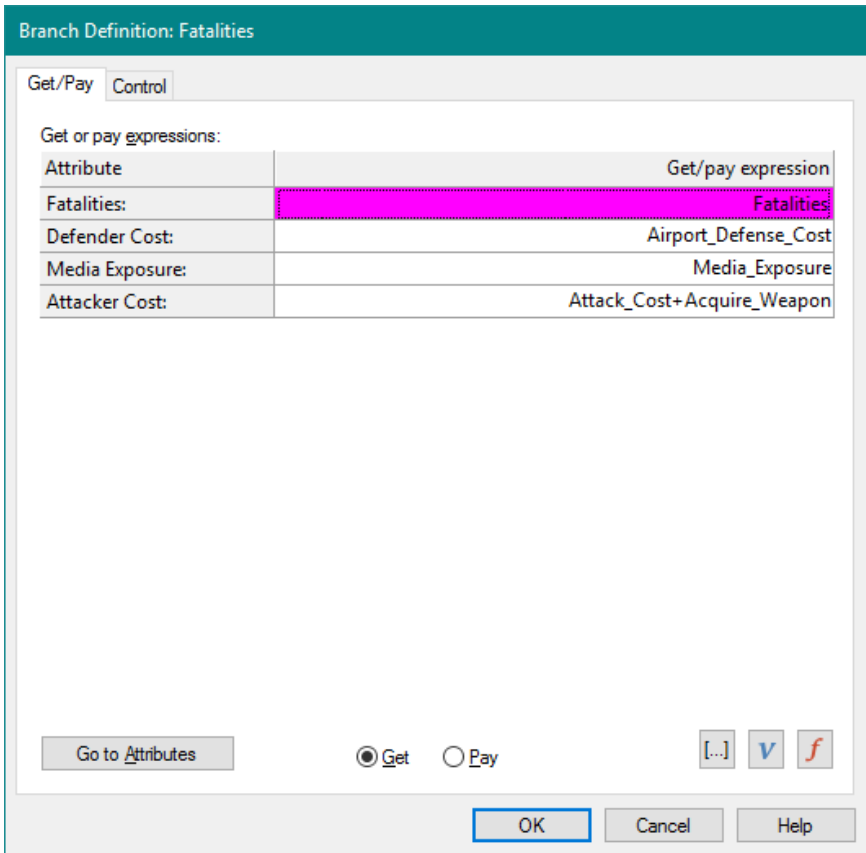
**Figure 8-29. Get/Pay Expression for No Outcome of Acquire Weapon**

- ⇒ Do the same for the None branch of the Attack Airport decision.
- ⇒ Double-click the No branch of the Attack Success node.

At this point in the tree the Attacker would have incurred the costs associated with the attack, whether it was deemed successful or not.

- ⇒ Enter a "0" for the Media Exposure input.
- ⇒ For the Attacker Cost attribute enter "Attack\_Cost + Acquire\_Weapon" by using the Select Variable button.
- ⇒ Click OK to close the Branch Definition dialog.
- ⇒ Double-click on the branches for the Fatalities node.

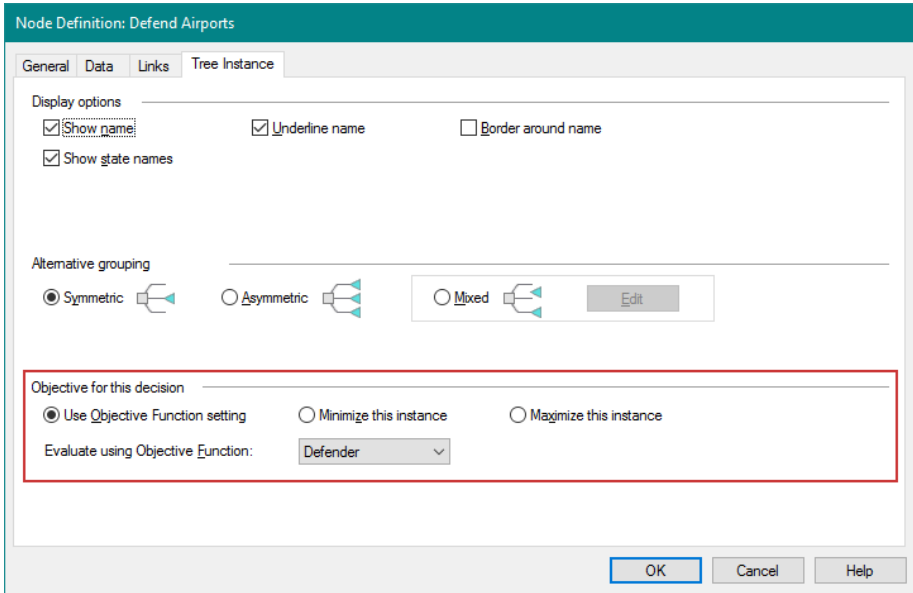
- ⇒ Select the input for the Media Exposure attribute and press the Select Variable button. Select Media\_Exposure from the list.
- ⇒ For the Attacker Cost attribute you'll enter to following formula once again: "Attack\_Cost + Acquire\_Weapon". The Branch Definition dialog for the Fatalities node should now look like Figure 8-30.



**Figure 8-30. Get/Pay tab of Branch Definition dialog for Fatalities node**

Now there is just one last step to take to fully incorporate the Attacker's decision and objective function into the model. You need to tell DPL what objective function to use (Attacker or Defender) for each of the decision instances in the tree.

- ⇒ Double-click the Defend Airports decision node to open the Tree Instance tab of the Node Definition dialog (Figure 8-31).



**Figure 8-31. Objective for this Decision Setting for Defend Airports Decision Instance**

Within the *Objective for this decision* section, note that the radio button is set to *Use Objective Function setting*. This means it will use the setting specified within the Objective & Utility dialog (Maximize or Minimize) for given decision instance. Furthermore, the *Evaluate using Objective Function* drop-down box is set to Defender. So for the Defend Airports decision, DPL will choose the alternative that minimizes the Defender objective function as defined in the Objective & Utility dialog.

⇒ The settings are correct for this decision. Click OK to close the Node Definition dialog.

The Attack Airport decision is made by the attacker. You will modify the model to reflect this.

⇒ Double-click the Attack Airport Decision Tree node to access the Tree Instance tab.

⇒ Within the *Objective for this decision* section, the radio button should be set to *Use Objective Function setting*.

⇒ Select Attacker from the Evaluate using Objective Function drop-down.

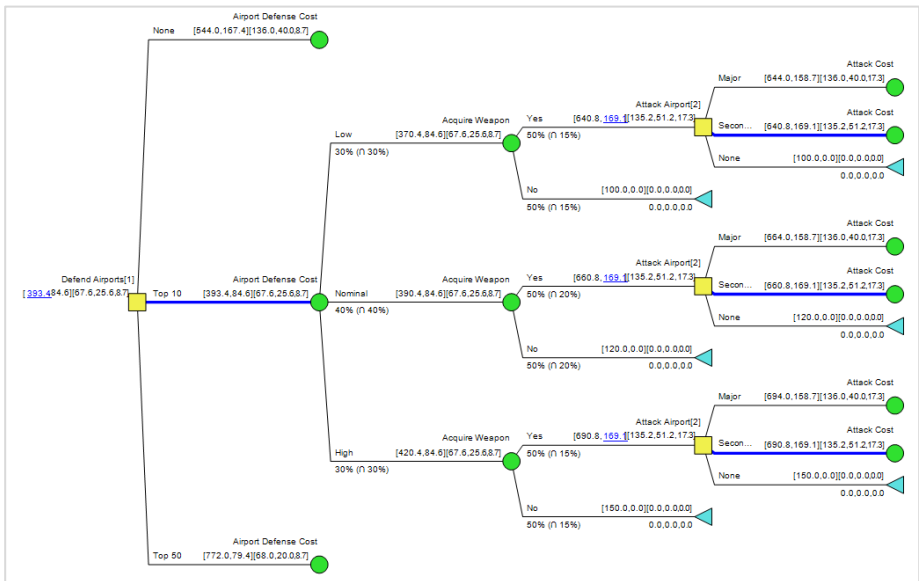
⇒ Click OK to close the dialog.

Now you are ready to run an analysis on the model and generate results.



- ⇒ Within the Home | Run group make sure that Risk Profile and Policy Tree are checked. The rest of the outputs should be un-checked.
- ⇒ Below the Risk Profile checkbox, set the Select Risk Profile Quantity box to All Obj. Fns. This tells DPL to generate a Risk Profile for each Objective Function in the model.
- ⇒ Click the Decision Analysis button to run the analysis and generate the requested results.

DPL will display a Policy Tree (Figure 8-32). You may first notice the blue shaded branches and values. These are an indication of the optimal decision policy and the objective function that is used at that decision.



**Figure 8-32. Policy Tree™ for Multiple Objective Function Model**

At the head of the Policy Tree you can see that the expected value of the model has increased slightly and the optimal decision policy is to defend only the top 10 airports. Notice that the first Expected Value listed is in blue font – as this decision is made using the Defender’s objective function.

Looking further down the tree you can see that at the Attack Airport decision, the second Expected Value listed is shaded blue meaning that the Attacker objective function is used for this decision. The optimal policy, given that the Defender defends the top 10 airports, is to attack a secondary airport for maximum effect and least cost.

Within the Workspace Manager you'll find that DPL generated two Risk Profile charts and datasets, one for each of the objective function. The data and chart are given the name of the Objective Function. Look at these results if you'd like.

## 9. Multidimensional Value Nodes

Multidimensional value nodes allow you to model series, one-dimensional arrays and two-dimensional arrays in the Influence Diagram. As with scalar value nodes, these value nodes can contain numeric constants and/or formulas. Each of these types of multidimensional value nodes will be demonstrated in the sections that follow.

Regardless of its dimensionality, any value node in DPL can be linked to a DPL program or an Excel spreadsheet. Value nodes that are linked to Excel spreadsheets can be either metric nodes that are calculated by Excel and returned to DPL, or driver nodes whose data DPL sends to Excel for use in Excel's recalculation.

This chapter assumes that you are familiar with DPL model building basics, linking DPL models to Excel, and naming ranges in Excel. If you are new to DPL and/or naming ranges in Excel, you may wish to review the tutorials contained within Chapters 2 through 0 of this manual and/or your Excel documentation, before proceeding with this chapter.

If you are building cash flow models and wish to analyze the uncertainties associated with your cash flows over time, you may wish to work through Chapter 9.1.1 of this chapter (and the other sections, as needed) and then proceed to Chapter 10, Time Series Percentiles.

### 9.1 Creating and Linking a One-Dimensional Value Node

---

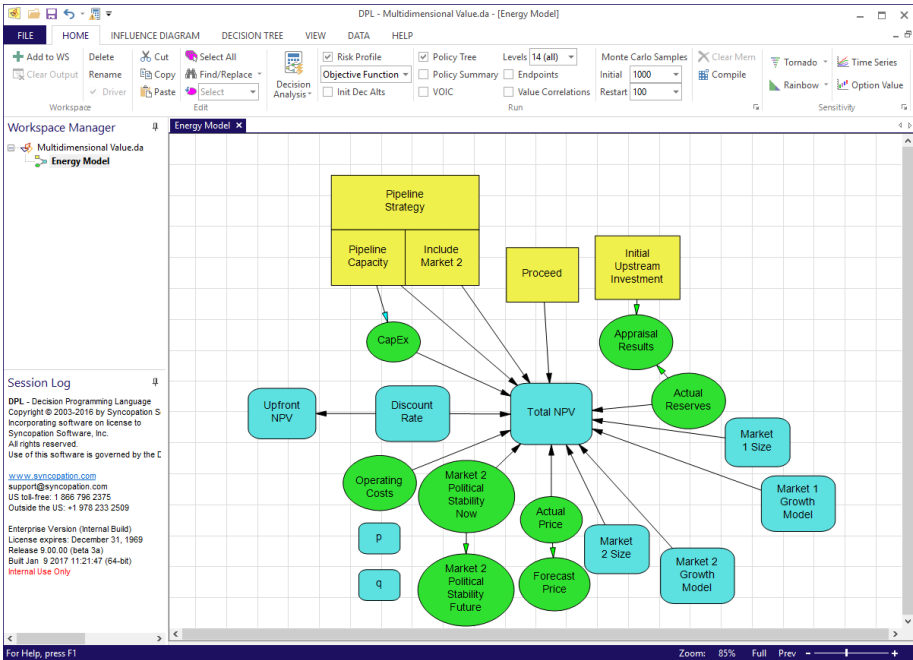
This section contains brief tutorials on setting up one-dimensional Excel-linked array and series value nodes in DPL. One-dimensional Excel-linked arrays and series can be either metric or driver nodes. The tutorial will look at an example of a one-dimensional array metric node, a series metric node and a series driver node.

#### 9.1.1 A One-Dimensional Metric Node

You will begin by examining a model of a decision about a gas pipeline investment.

⇒ Select File | Open.

- ⇒ Navigate to the Examples folder where DPL is installed, usually C:\Program Files\Syncopation\DPL9\Examples.
- ⇒ Select Multidimensional Value.da and click Open. You should see the Energy Model as depicted in Figure 9-1.



**Figure 9-1. Multidimensional Value.da Model**

This model contains a few features that may be unfamiliar to you, depending on your experience with DPL. You don't need to understand all the logic in the DPL model or the linked Excel cash flow model in order to complete this tutorial, but you should note the following components of the model. If any of these components are unfamiliar, you may wish to review previous chapters of this manual. Specifically, this model contains the following.

- Several chance nodes, some of which are conditioned by other chance nodes (see Section 2.5). Not all the nodes are linked to the Excel cash flow model.
- Several value nodes which serve various purposes: metric nodes, driver nodes, and parameters (p and q) that can be used for sensitivity analysis (see Section 5.8).

- A strategy table comprising two decisions: pipeline capacity, and whether to expand the pipeline project to a second, riskier market (see Appendix B).
- Asymmetry -- for example, the "Market 2 Political Stability Future" node only matters if the project is expanded to Market 2 (see Chapter 2).

Right now the model has two Excel-linked metric nodes: Upfront\_NPV and Total\_NPV. You will look at the Excel links and add another metric node: a value node for the 10-year annual cash flow.

⇒ Launch Excel.

⇒ Open the file "Multidimensional Value.xlsx". This file is found in the "Examples" folder where DPL was installed, usually C:\Program Files\Syncopation\DPL9\Examples.

The default directory may vary slightly depending on your license type and operating system.

⇒ View the Assumptions sheet (Figure 9-2).

Basic Parameters		Color coded blue, yellow or green ==> currently linked to DPL										
Discount Rate	10%											
Market 1 Growth Model Choice	3											
Market 2 Growth Model Choice	3											
Market 1 Size Year 1	110											
Market 2 Size Year 1	180											
Annual % Growth Market 1, Option 2	5.0%											
Annual % Growth Market 2, Option 2	5.0%											
Annual Growth Assumptions for Market Growth Mod.		2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	
Market 1 Size: Option 2		110	116	121	127	134	140	147	155	163	171	
Market 2 Size: Option 2		180	189	198	208	219	230	241	253	266	279	
Year by Year % Growth for Market 1, Option 3		0.04	0.05	0.05	0.05	0.04	0.04	0.03	0.03	0.03	0.03	
Year by Year % Growth for Market 2, Option 3		0.04	0.05	0.05	0.05	0.04	0.04	0.03	0.03	0.03	0.03	
Other Assumptions												
Annual Price Growth Market 1	2%											
Annual Price Growth Market 2	2%											
Annual Op Costs Growth Market 1	4%											
Annual Op Costs Growth Market 2	3%											
Start year	2017											
value: p	0.55											
value: q	0											
Upfront Cost Escalation	1											
Initial Marketing Spend	500											

**Figure 9-2. Assumptions Sheet of Multidimensional Value.xlsx**

The Assumptions sheet contains various market, price, and cost growth assumptions; numerous other parameters; and several "switch" cells for decisions and uncertainties. Some cells are linked to DPL driver nodes and the color coding indicates whether they are linked to value, decision, or chance nodes. You can also see the two cells linked to DPL metric nodes: Upfront\_NPV and Total\_NPV.

You will run the model to generate results for comparison later on.

- ⇒ Switch to DPL.
- ⇒ In the Home | Run group, Fast Sequence evaluation should be the default indicated in the Decision Analysis icon and Risk Profile, Init Dec Alts, and Policy Tree should be selected.
- ⇒ Click the Home | Run | Decision Analysis icon or press F10. The model may take a couple of minutes to run.

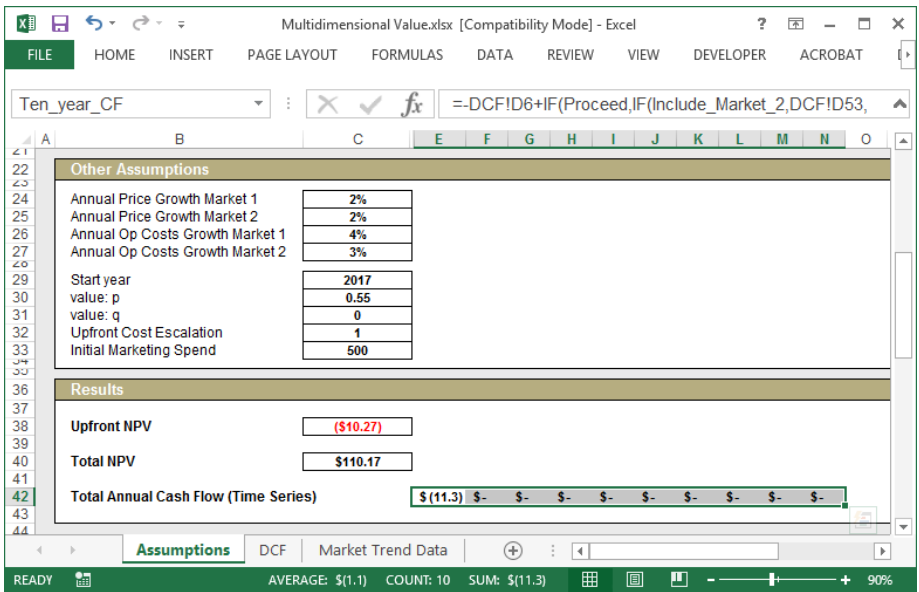
⇒ Examine the Risk Profile and Policy Tree results.

You will now examine a named range in Excel that contains 10 cells. It will be used for the value node you are going to create in DPL corresponding to the annual cash flow.

⇒ Switch back to Excel.

⇒ In the Assumptions sheet in Excel, select the range E42:N42. This is the first 10 years of annual cash flows in the model. It is labeled Total Annual Cash Flow (Time Series) in the Results section.

⇒ Note that it says Ten\_year\_CF in the Name combo box (located at the left end of the Excel edit bar). This is the name of this range. See Figure 9-3.



**Figure 9-3. Excel Cash Flow Ten\_year\_CF Range**

⇒ Switch back to DPL.

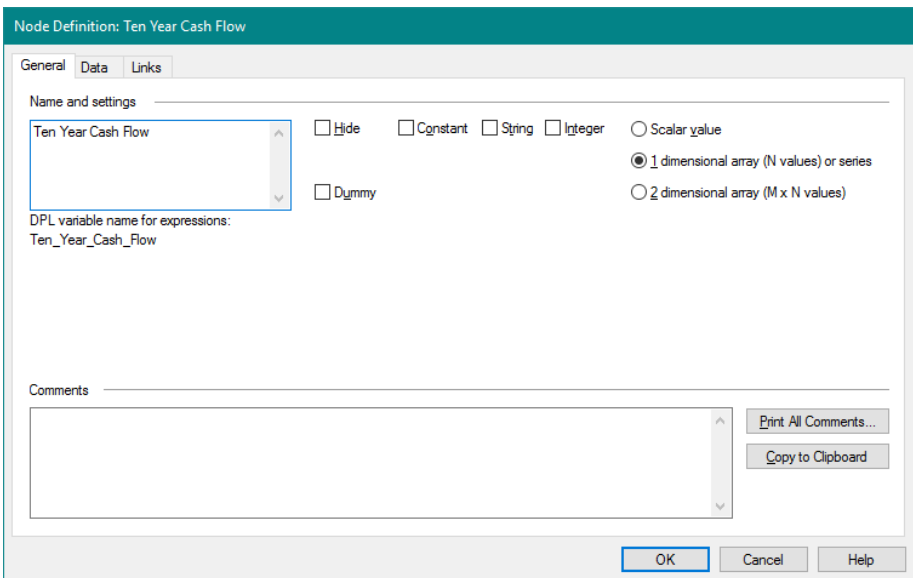
⇒ Activate the Energy Model.

⇒ To add a new value node to the model, click Influence Diagram | Node | Linked Node. Excel calculation linked should be default add link type indicated by the icon (🔗). If it isn't, use the drop-down to select it.

⇒ In the Range Names dialog, select Ten\_year\_CF.

Note the values within Row and Column for the Ten\_year\_CF range. It is a one dimensional array. DPL will automatically create a multidimensional value node based on the dimensionality of the named range selected.

- ⇒ Click OK.
- ⇒ DPL places the Ten Year CF node off to right. You may move it to nearer to the Market Size 1 node if you'd like.
- ⇒ Double-click the new node to examine its definition.
- ⇒ Switch to the General tab. Note that DPL has created a one-dimensional array or series. Change the name to Ten Year Cash Flow. The General tab should now look like Figure 9-5.

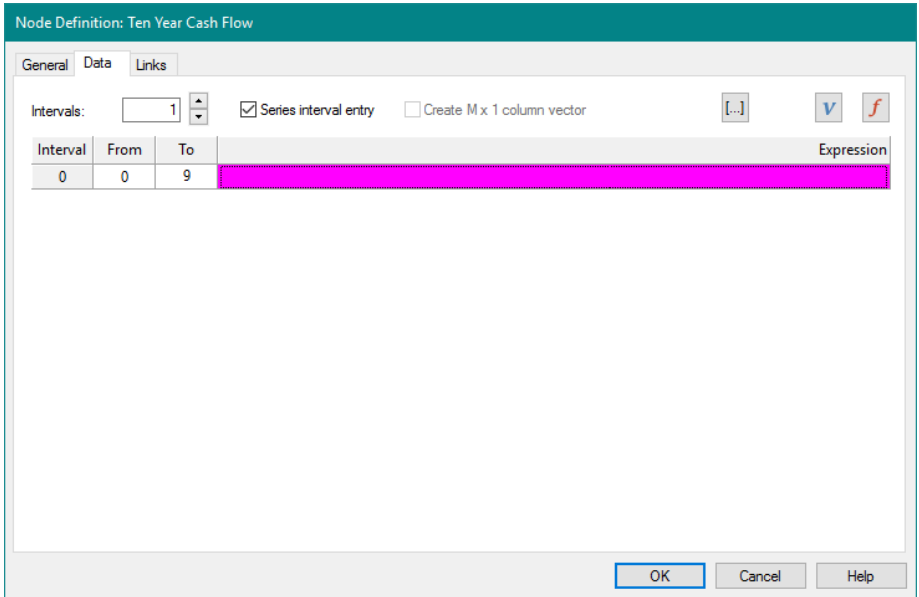


**Figure 9-4. Node Definition Dialog for 1 Dimensional Array**

- ⇒ Switch to the Data tab of the Node Definition dialog.

Note that DPL has created a series by setting the Series interval entry to be true. And it has created a single interval which goes from 0 to 9. So DPL has created a single interval series with ten elements. See Figure 9-5. This matches what DPL found in Excel when it examined the named ranges, e.g., that Ten\_year\_CF is a ten cell named range with formulas. DPL will create a series metric node if it finds formulas in the named range. It will create an array driver node if it finds numbers in the named range. You will learn more about series interval entry in the next section.

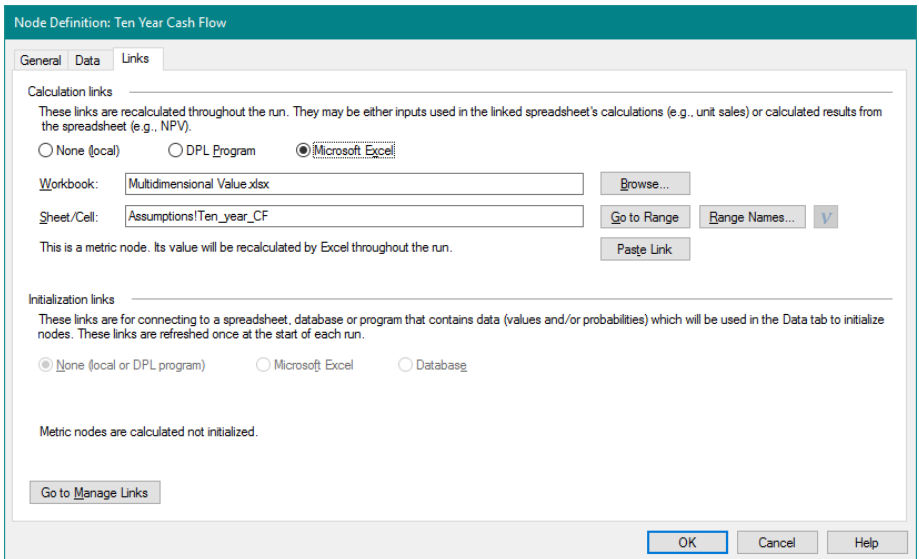




**Figure 9-5. Data Tab for Single Interval Series**

⇒ Switch to the Links tab of the Node Definition dialog

Note that DPL has linked the node to Ten\_year\_CF. See Figure 9-6.

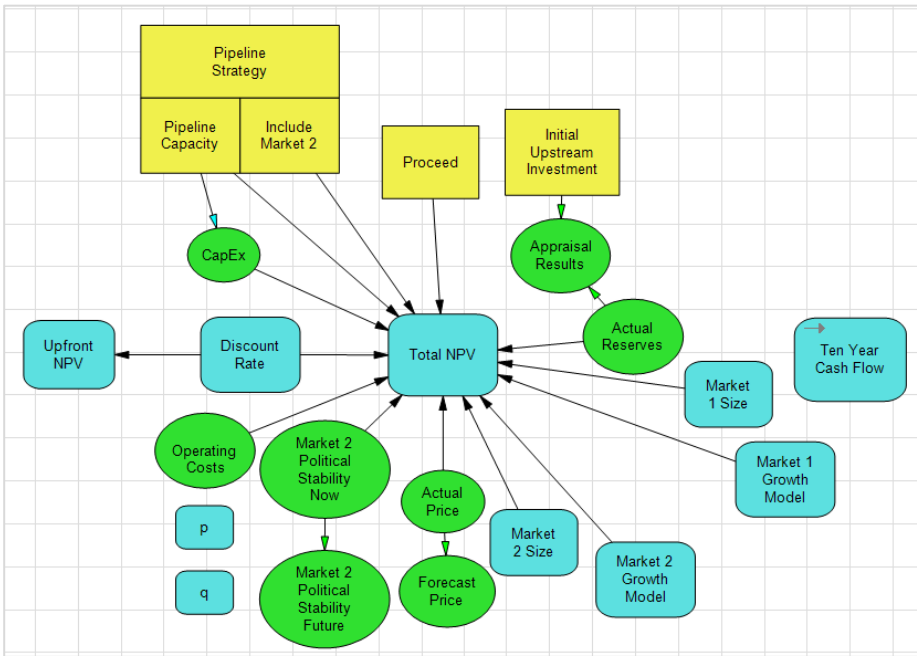


**Figure 9-6. Node Definition Dialog with Linked Range Name**

DPL tells you that the linked node is a metric node. As mentioned above, DPL created a metric because it found formulas in the named range and hence did not initialize the node with any data on the Data tab.

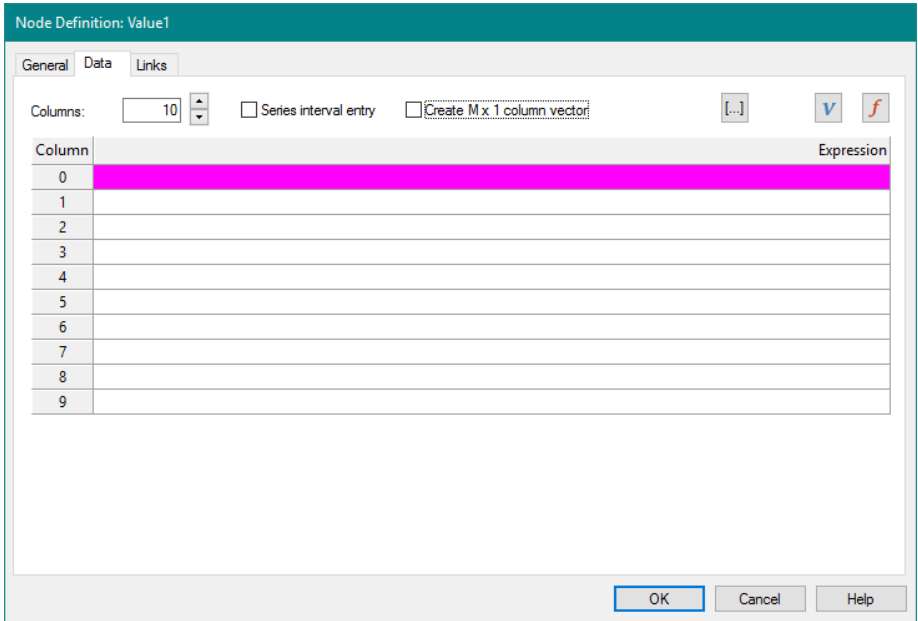
⇒ Click OK.

Your model should now look like Figure 9-7. You have created a 1 dimensional (1 x 10) series metric value node that is linked to a 10-cell range in Excel. You can see in Figure 9-7 that the value node Ten Year Cash Flow has a vector arrow on it to indicate that it is a 1-dimensional node rather than a scalar value.



**Figure 9-7. Model with 1 x 10 Value Node**

Note when linking a one dimensional value node to a named range in Excel you may use either an array or a series node. DPL automatically creates a series node when it finds formulas in the range. But if you were creating the node and linking it using the Node Definition dialog you can use either type to link the node to Excel but you need to be sure the dimensionality of the node matches the named range. See Figure 9-8 for an example of what the Node Definition Data tab looks like for a one dimensional metric array.



**Figure 9-8. Data Tab for One-Dimensional Metric Node**

As your model is now set up, the Ten Year Cash Flow value node will import ten years of cash flow values from the Excel spreadsheet each time the model is run. However, further set up of the model is needed before you see these values in analysis results. At this point you might want to analyze the expected values and ranges of uncertainty around these annual cash flows. To do this, you need to add 10 attributes to your model to correspond to the annual cash flows. Defining attributes in your model allows you to run Time Series Percentiles to visualize how the uncertainty in the cash flows evolves over time.

⇒ Do File | Save as to save the Workspace with a new name.

If you would like to run a Time Series Percentiles now, skip to Chapter 10.

If you would rather learn more about multidimensional value nodes first, continue to the next tutorial.

### 9.1.2 A One-Dimensional Series Driver Node

This tutorial provides an example of a one-dimensional value node containing a series. A series in DPL is a one-dimensional vector of values that may depend on the states of events, may use formulas, or may just contain constants. Series are different from one-dimensional arrays

because they are defined using intervals and/or relative subscripts as will be demonstrated in this tutorial. The ability to use relative subscripts in a series makes it easy to define elements of the series in terms of other elements, e.g., sales in time period  $t$  is equal to sales in time period  $t - 1$  times a growth factor.

⇒ Either continue with the example you saved in Section 9.1.1 above, or if you closed the files re-open Multidimensional Value.da and Multidimensional Value.xlsx.

Note: If you completed the Time Series Percentiles tutorial in Chapter 10 and saved your model, you may use that model for the remaining tutorials in this chapter, however, your model will look slightly different from the figures in the following sections.

⇒ In Excel, select the Assumptions tab of the spreadsheet and look at the Basic Parameters and Annual Growth Assumptions sections near the top. See Figure 9-9.

Basic Parameters		Color coded blue, yellow or green ==> currently linked to DPL										
Discount Rate		10%										
Market 1 Growth Model Choice		3										
Market 2 Growth Model Choice		3										
Market 1 Size Year 1		110										
Market 2 Size Year 1		180										
Annual % Growth Market 1, Option 2		5.0%										
Annual % Growth Market 2, Option 2		5.0%										
Annual Growth Assumptions for Market Growth		2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027
Market 1 Size: Option 2		110	116	121	127	134	140	147	155	163	171	179
Market 2 Size: Option 2		180	189	198	208	219	230	241	253	266	279	293
Year by Year % Growth for Market 1, Option 3		0.04	0.05	0.05	0.05	0.04	0.04	0.03	0.03	0.03	0.03	0.02
Year by Year % Growth for Market 2, Option 3		0.04	0.05	0.05	0.05	0.04	0.04	0.03	0.03	0.03	0.03	0.02

**Figure 9-9. Assumptions Sheet, Parameters and Growth Assumptions**

Some of the parameters (the ones shaded in blue) are already linked to value nodes in DPL. The parameters Market 1 Growth Model Choice and Market 2 Growth Model Choice are linked, so these values (which are currently set to 3) are sent from the DPL model. If these values are set to 2, a simple annual growth rate (the value in cells C9 and/or C10) is used for the market size growth. If they are set to 3, a different growth rate may be used for each year. These rates are currently set in the block of cells

that begins with cell E18. The growth model parameters can be set differently by year and for each market.

In this section of the tutorial, you will look at the way the simple annual growth rates are set up in Excel, and change the DPL model so that this logic is done in DPL instead, using a series in a one-dimensional value node.

⇒ Examine the formulas for Market 1 Size: Option 2 and Market 2 Size: Option 2 (these begin in cells E15 and E16).

You'll see that the market sizes are initialized using cells that are also linked to value nodes in DPL: Market\_1\_Size and Market\_2\_Size. After the first year, the values Mkt\_1\_Growth and Mkt\_2\_Growth, which are not currently linked to DPL value nodes, are used to grow the market each year.

In financial and cash flow models, as well as in other types of models, you will often see growth trends represented in series such as this one. As this model is currently set up, the series is defined in Excel rather than in DPL. However, in some situations, you might want the series values to be defined in DPL so that they can depend on other values in your DPL model or so that you can more easily do a sensitivity analysis on them.

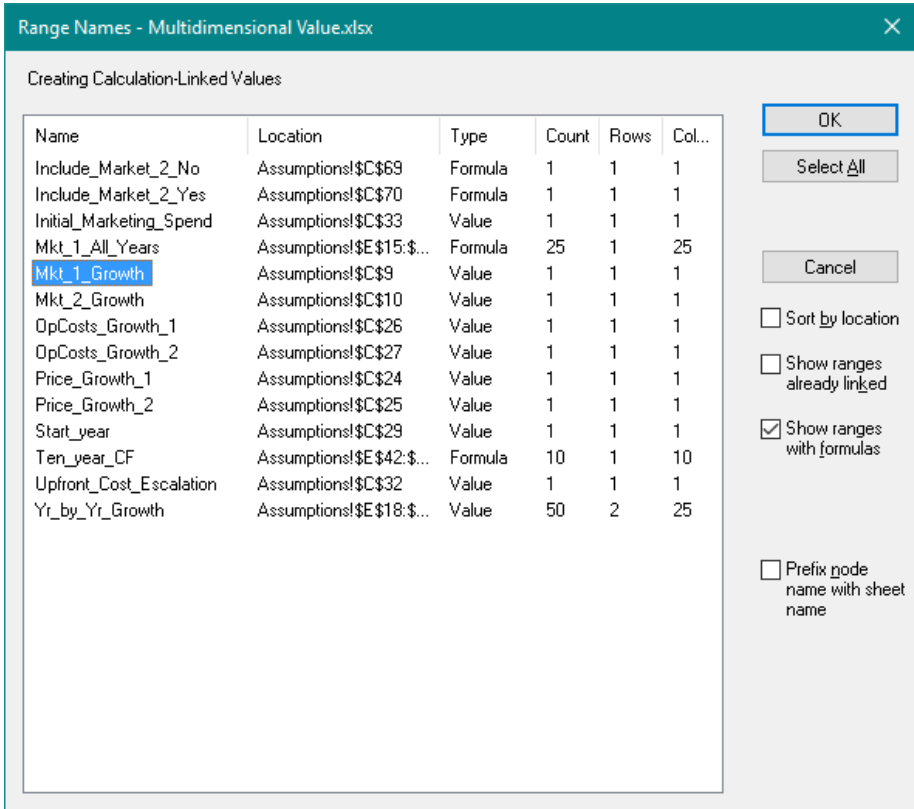
You will now see how series like this one can be defined quickly and easily in a one-dimensional value node in DPL.

⇒ Switch to DPL.

⇒ Select Influence Diagram | Node | Linked Node. The Range Names dialog appears.

⇒ Select Mkt\_1\_Growth from the list. See Figure 9-10.

⇒ Click OK.

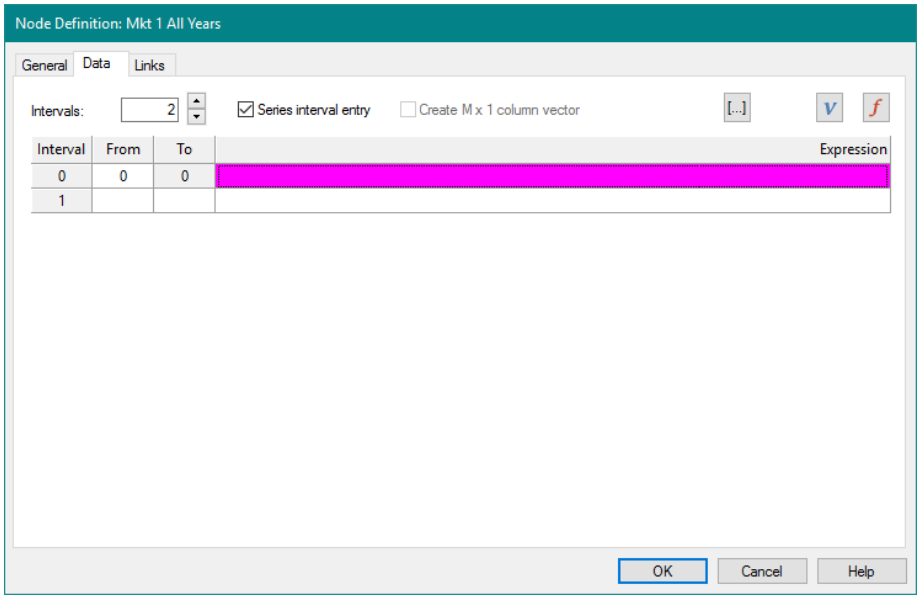


**Figure 9-10. Range Names Dialog**

DPL adds the value node, names it Mkt 1 Growth, and initializes it with 0.05, which is the growth rate (5%) from the spreadsheet. The node is now a driver node whose value will be sent to Excel. If you wish to use a different value for Mkt 1 Growth, you would now change it in DPL instead of in Excel since the value in DPL will be sent to Excel and overwrite any change made there. You will use this value in the series you are about to create.

- ⇒ Click Influence Diagram | Node | Add to add a new, local value node. Place the node near the Mkt 1 Growth node you just created.
- ⇒ In the General tab, name the new node Mkt 1 All Years.
- ⇒ Select *1 dimensional array (N values) or series* in the Dimensions section.
- ⇒ Switch to the Data tab.
- ⇒ Check the *Series interval entry* checkbox.



⇒ Change the number of intervals to 2. See Figure 9-11.



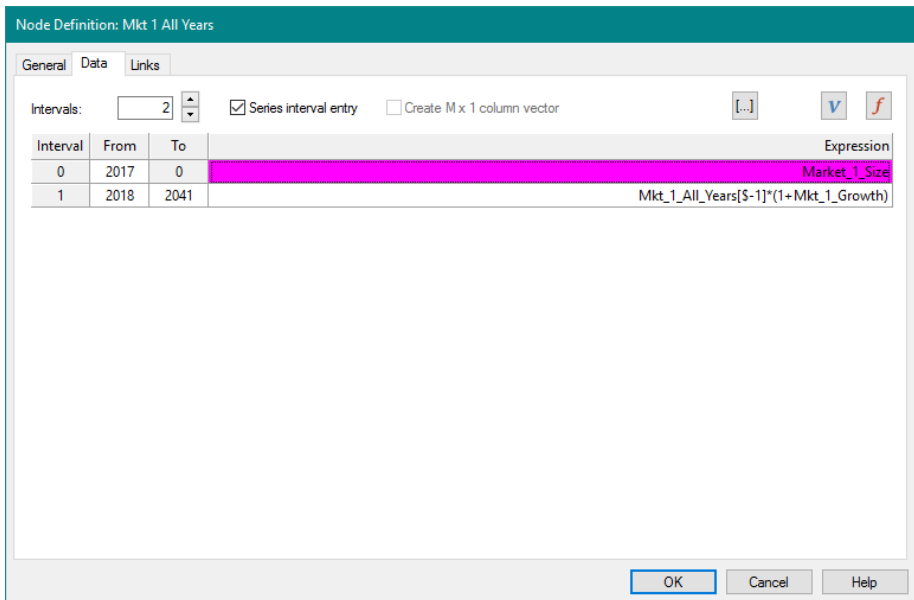
**Figure 9-11. Node Definition Data for a Series with 2 Intervals**

Series are defined using intervals. Your series has two intervals: you will set it up so that the first interval is for the first year, and the second is for every year thereafter. The intervals in a series do not have to start at 0, as DPL arrays do. Your series will use the cash flow years for the subscripts for clearer illustration. You will also see how to use a relative subscript in this example.

- ⇒ In the From cell for interval 0, type "2017".
- ⇒ In the From cell for interval 1, type "2018". Note that DPL fills in the "To" cell for the preceding interval so that the interval subscripts are consecutive.
- ⇒ In the To cell for interval 1, type "2041". Note: you must specify the To boundary for the last interval.
- ⇒ Click in the Expression cell for interval 0.
- ⇒ Click the Select Variable () button.
- ⇒ Double-click on Market\_1\_Size in the list.
- ⇒ Double-click in the Expression cell for interval 1 for text edit mode.

- ⇒ Click the Select Variable () button again.
- ⇒ Double-click on Mkt\_1\_All\_Years in the list.
- ⇒ In the Expression cell for interval 1, edit the expression to read "Mkt\_1\_All\_Years[\$ - 1]\*(1+Mkt\_1\_Growth)". Note: you can use the Select Variable () button again if you wish, to bring in the Mkt\_1\_Growth variable.

The dialog should now look like Figure 9-12.



**Figure 9-12. Node Definition Data for Mkt 1 All Years Node**

Take a moment to understand how this series is defined. The series definition refers to itself using the relative subscript symbol "\$". The term "\$ - 1" indicates the previous element in the series. (In DPL, series are the only type of expression in which you can use relative subscripts. See Chapter 15 of this manual for more information.) This series is now defined to produce exactly the same result that the Excel formulas for Market 1 Size (option 2) are producing.

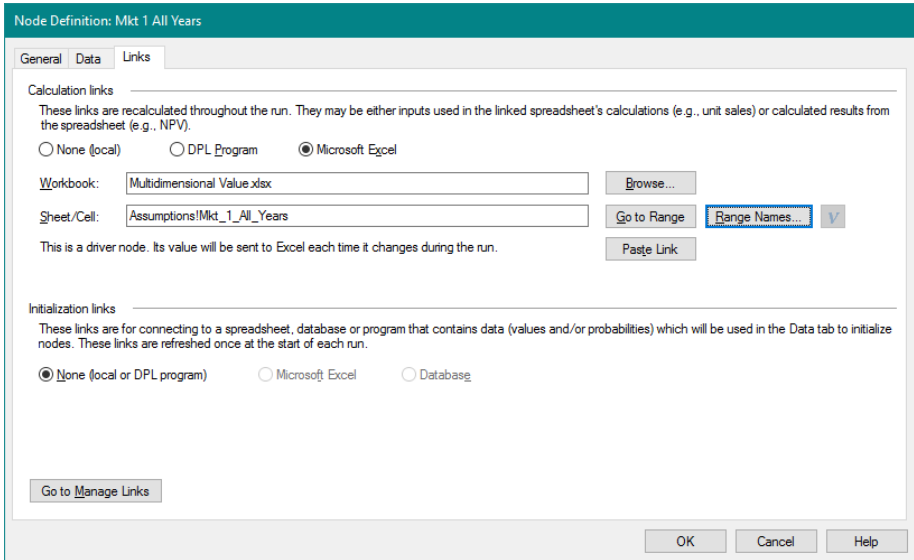
The series is indexed according to the years in the cash flow model: 2017 through 2041. This was done for clarity of definition. For purposes of the analysis, you could also have indexed the series from 0 to 24 or from any other integer to that integer plus 24. DPL will only export the actual values of the series expressions to Excel.



Now you need to link this node to the appropriate range in Excel so that DPL can send the values to Excel. There is already a named range in Excel to do this.

- ⇒ If you'd like to see the named range, switch back to Excel.
- ⇒ Select cells E15 through AC15.
- ⇒ Look in the Name combo box in Excel and note it says Mkt\_1\_All\_Years. (Note: there is no requirement that you name this range exactly the same as the value node in Excel. Any range name acceptable to Excel is fine.)
- ⇒ Switch back to DPL.
- ⇒ Click the Links tab of the Node Definition dialog.
- ⇒ Under Calculation Links, select *Microsoft Excel*.
- ⇒ Click the Range Names... button.
- ⇒ In the Range Names dialog, check the checkbox for Show ranges with formulas. (The range you defined in Excel currently has a formula in it.)
- ⇒ Select Mkt\_1\_All\_Years from the list. You can see that it has a "Count", or dimension of 25 cells.
- ⇒ Click Select.
- ⇒ Click Yes for the warning. DPL is telling you that you have just defined a driver node that will overwrite a formula in Excel.

The Links tab of the Node Definition dialog should now look like Figure 9-13.



**Figure 9-13. Node Definition Links for Mkt 1 All Years Node**

- ⇒ Click OK.
- ⇒ Create Influence arcs from Mkt 1 Growth and Mkt 1 All Years to Total NPV.

The Mkt 1 All Years node is now fully defined and linked to Excel.

Note that the Excel formulas for the Mkt\_1\_All\_Years range now will not be used because the values are being determined in DPL and exported to Excel instead. You can tell DPL to reset the Excel cells back to their original values/formulas after a model run. In general, it is a good practice to do this. If you are overwriting cells with formulas as in this case, it is a particularly good idea to do this to maintain the spreadsheet with its initial functionality.

- ⇒ Go to Influence Diagram | Model | Settings.
- ⇒ Check *Reset Excel values/formulas after run*, if it is not already checked.
- ⇒ Click OK.
- ⇒ Make sure Fast sequence evaluation is the default run type and Risk Profile, Init Dec Alts, and Policy Tree are selected.
- ⇒ Click Home | Run | Decision Analysis to run the model.

After a few minutes, DPL will return precisely the same decision analysis results that you obtained at the beginning of Section 9.1.1. Check the Session Log to compare results if you like.

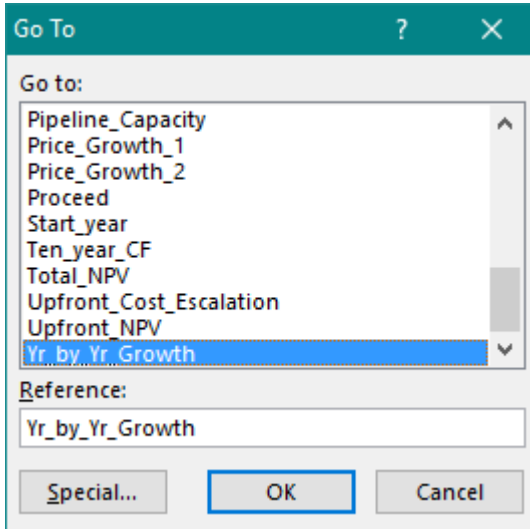
This tutorial illustrated how series can be used in DPL's 1-dimensional value nodes to give you more control and more power in building models. In some situations, you might want to develop more complex series using expressions that refer to other value nodes in your DPL model, or you might not want to have to switch to Excel every time you want to change an assumption or a formula. In this example, you needed to create the node and link it via the Node Definition dialog because you wanted to create a driver node and the range you were linking to in Excel contained formulas. Had you used Influence Diagram | Node | Linked Node, the Range Names dialog would have assumed you wanted to create a metric node.

## 9.2 Creating and Linking a Two-Dimensional Value Node

---

In the sections above, you modified a DPL model to include two new value nodes: a one-dimensional metric node, and a one-dimensional driver node. In this section you will define a two-dimensional array that is a driver node and link it to Excel. The process for defining two-dimensional arrays is very similar to the steps you used to define the one-dimensional arrays.

- ⇒ Either continue with the example you saved in the previous section, or if you closed the files re-open Multidimensional Value.da and Multidimensional Value.xlsx.
- ⇒ In Excel, select the Assumptions sheet.
- ⇒ Press F5 (the Go To button in Excel). Scroll down to the Yr\_By\_Yr\_Growth range and select it. See Figure 9-14.



**Figure 9-14. Excel Go To Dialog**

⇒ Click OK to see where this range is.

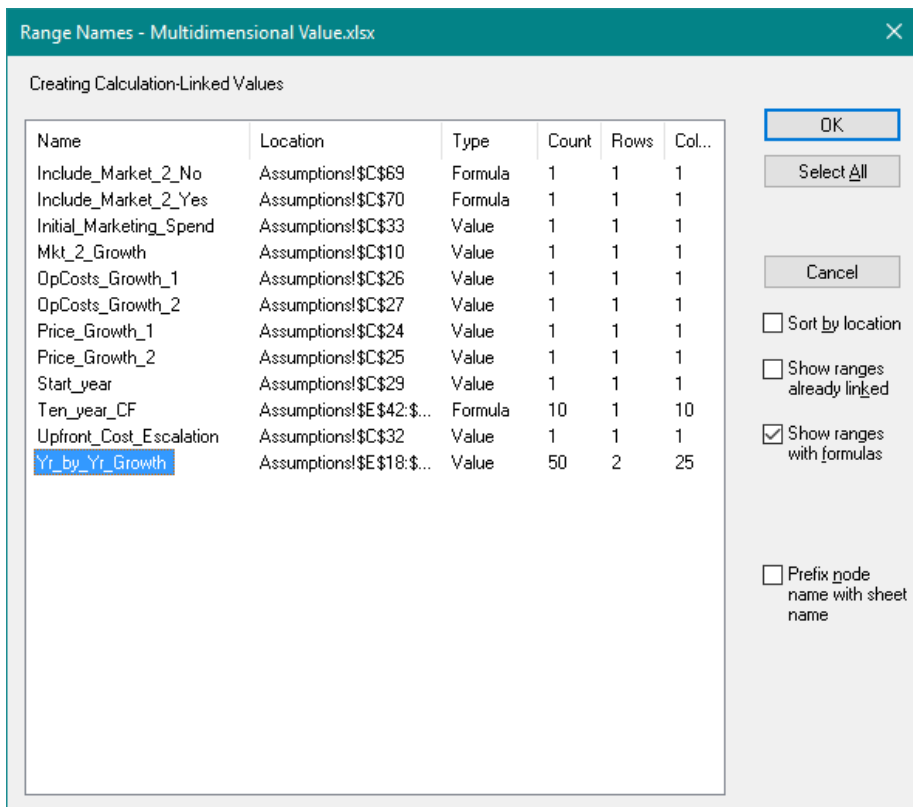
This two-dimensional range contains values that represent growth rates for the two energy markets. The Excel spreadsheet uses these if the market growth model choice is set to 3. This option allows the user to enter a separate market growth rate in each year for each of the two markets.

⇒ Switch back to DPL without changing anything in Excel.

⇒ Click Influence Diagram | Node | Linked Node to add a new Excel Calculation-Linked value node to the model.

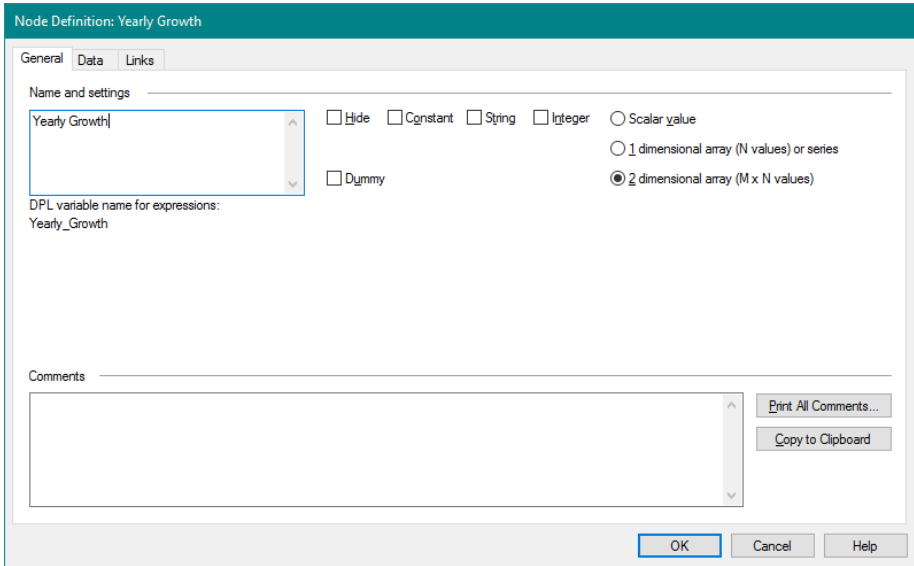
⇒ In the Range Names dialog select Yr\_by\_Yr\_Growth. See Figure 9-15.

⇒ Click OK. DPL places the node to the right in the Influence Diagram. Move it if you'd like.



**Figure 9-15. Range Name Dialog for 2 Dimensional Array**

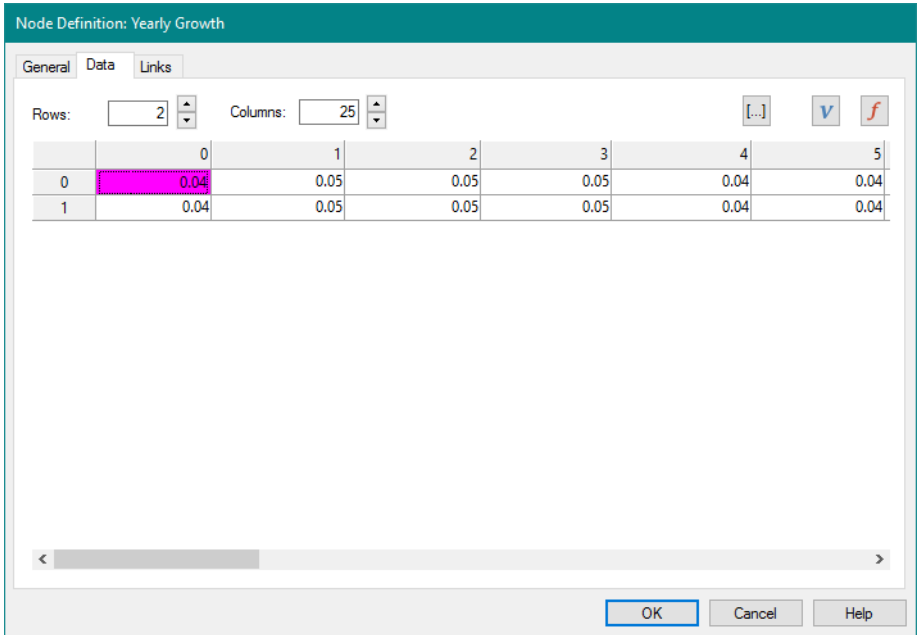
- ⇒ Double-click the new node to edit it. Switch to the General tab of the Node Definition dialog, name the node Yearly Growth. Note that DPL has created it as a 2 dimensional array. See Figure 9-16.



**Figure 9-16. Node Definition Dialog for 2 Dimensional Array**

⇒ Switch to the Data tab of the Node Definition dialog

Note DPL has created a 2 by 25 array and filled in the data from Excel. See Figure 9-17.



**Figure 9-17. Data Tab for 2-Dimensional Array with Values**

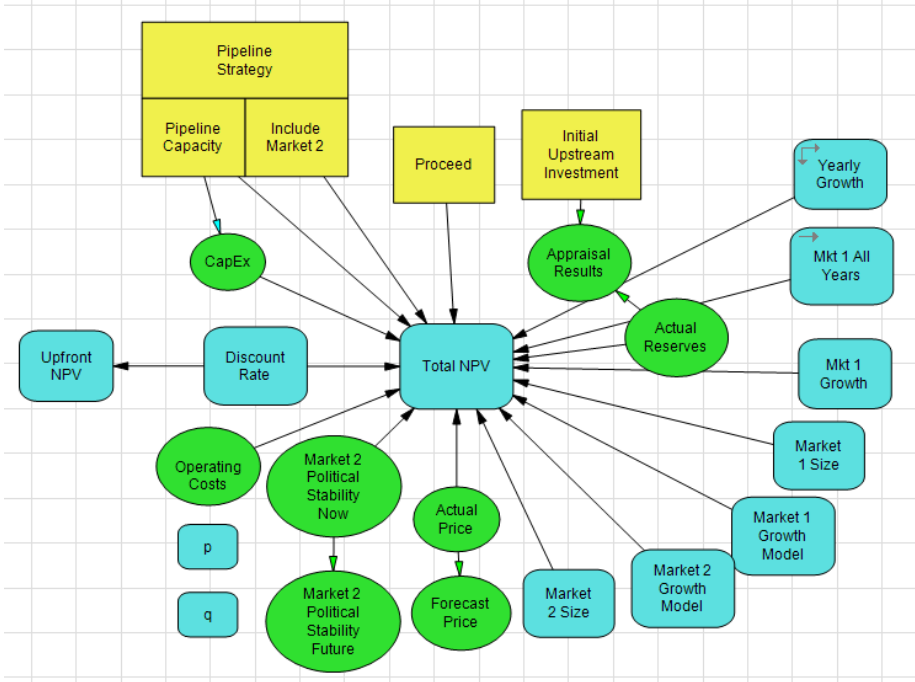
⇒ Switch to the Links tab

DPL has linked the node to the Yr\_By\_Yr\_Growth range.

⇒ Click OK.

⇒ Create an Influence arc from Yearly Growth to Total NPV.

Your model should now look like Figure 9-18. Yearly Growth is a value node with perpendicular vector arrows on it to indicate that it is a 2-dimensional node.



**Figure 9-18. Model with 2 x 25 Value Node**

Your model is set up so that DPL exports the 50 values in the Yearly Growth node to Excel. You will now run the model.

- ⇒ Make sure Risk Profile, Init Dec Alts, and Policy Tree are selected.
- ⇒ Click Home | Run | Decision Analysis or press F10.

Note: unless you have changed some of the input data, your results should be the same as those from model runs in earlier in the chapter.

This tutorial has shown you how to set up and link a two-dimensional value node in DPL. In this example, the values in the node were populated from Excel and are constants representing growth rates. However, each element of the node could contain any expression using DPL variables or functions.

Chapter 10 of this manual includes more information on how to set up Get/Pay expressions in DPL when an array or series is used to define the Get/Pay, as well as an example using DPL's Time Series Percentiles feature in conjunction with a one-dimensional value node.



# 10. Time Series Percentiles and Multiple Metrics

Time Series Percentiles are useful for understanding how the range of uncertainty in performance metrics, such as gross sales or net profits, evolves over time. The first section of this chapter discusses how to run and view Time Series Percentiles for the optimal decision policy. It continues with the example from Section 9.1.1 which uses a 1-dimensional value node that has been previously defined and linked to Excel.

Time Series Percentiles require the use of DPL's multiple attributes feature, as described in Chapter 8. This chapter also covers how to quickly generate Risk Profiles for multiple attributes simultaneously as well as more information on how to use the Get/Pay tab of the Branch Definition dialog with multiple attributes. Note that these features can be used in any multi-attribute DPL model, not just models in which the attributes correspond to metrics over time.

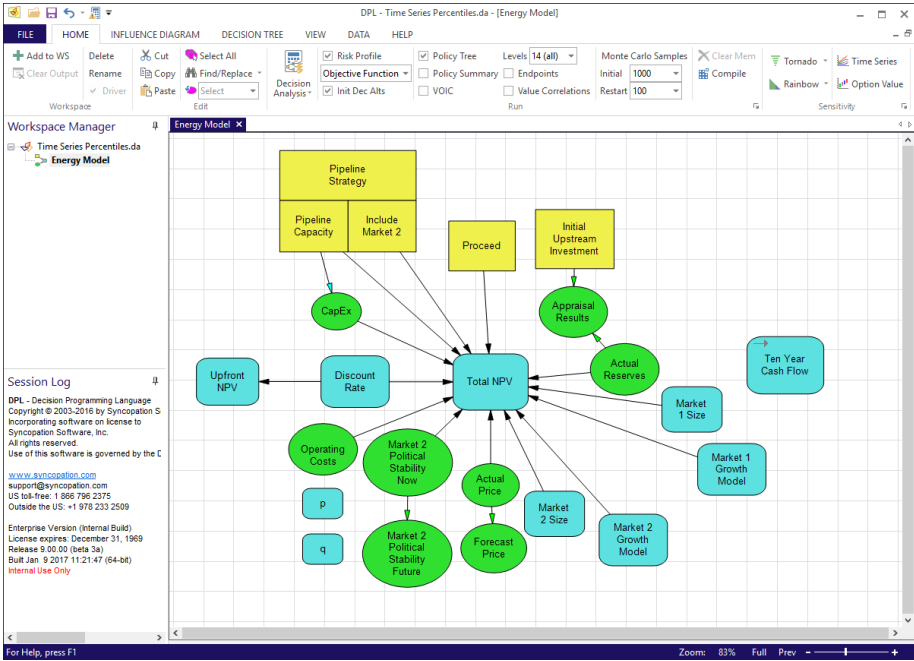
Previously you were only able to generate Time Series Percentiles for the optimal decision policy. With the DPL 9 Release you're now able to visualize cash flow over time for all your initial decision alternatives to gain a complete picture of how uncertainty evolves over time. Within Section 10.3 you open a new example model and generate a Time Series Percentiles chart for all of the initial decision alternatives in the model.

## 10.1 Time Series Percentiles

---

- ⇒ Select File | Open.
- ⇒ Open the workspace Time Series Percentiles.da. This file is found in the Examples folder where DPL was installed, usually  
C:\Program Files\Syncopation\DPL9\Examples.

DPL opens the Workspace as shown in Figure 10-1. This model may look familiar as it matches the model built at the end of Section 9.1.1.



**Figure 10-1. Energy Case Model Workspace**

- ⇒ Launch Excel.
- ⇒ Open the spreadsheet called Time Series Percentiles.xlsx. This file is also found in the Examples folder where DPL was installed, usually C:\Program Files\Syncopation\DPL9\Examples.

The first step to run Time Series Percentiles is to create a value node for the metrics you are interested in seeing over time. This was done in Section 9.1.1. The Ten Year Cash Flow value node is already linked to the spreadsheet and set up to import 10 years of cash flow results into DPL. To review this, do the following.

- ⇒ Switch back to DPL.
- ⇒ Double-click on the Ten Year Cash Flow value node to view its definition.
- ⇒ On the Data tab you'll see this node is a single interval, 10 element series with no data in it.
- ⇒ Click on the Links tab. This node (like the other linked nodes in the model) is linked to Time Series Percentiles.xlsx. It is a metric node since it has no data.

⇒ Click Cancel.

Time Series Percentiles use DPL's attributes interface. The next step to run Time Series Percentiles is to define attributes for each time period variable. You will need 10 additional attributes, one for each year. For more information on attributes, please see Chapter 8.

⇒ Click on Influence Diagram | Objective & Utility | Settings, (i.e., the dialog box launcher for the group). The Objective tab of the Objective & Utility dialog appears as shown in Figure 10-2.

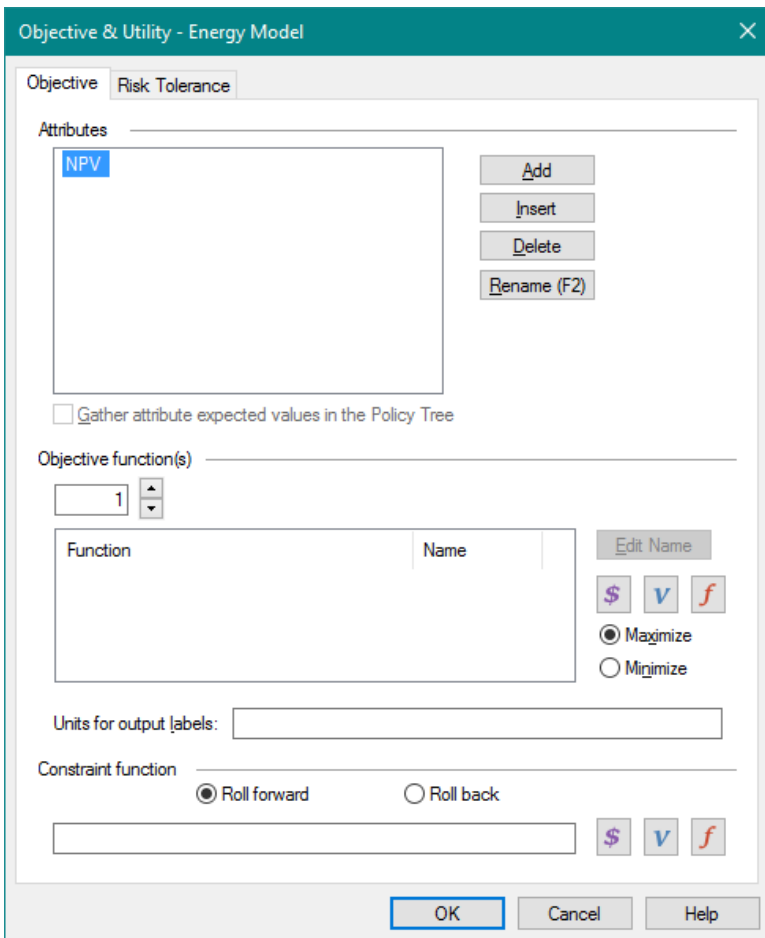
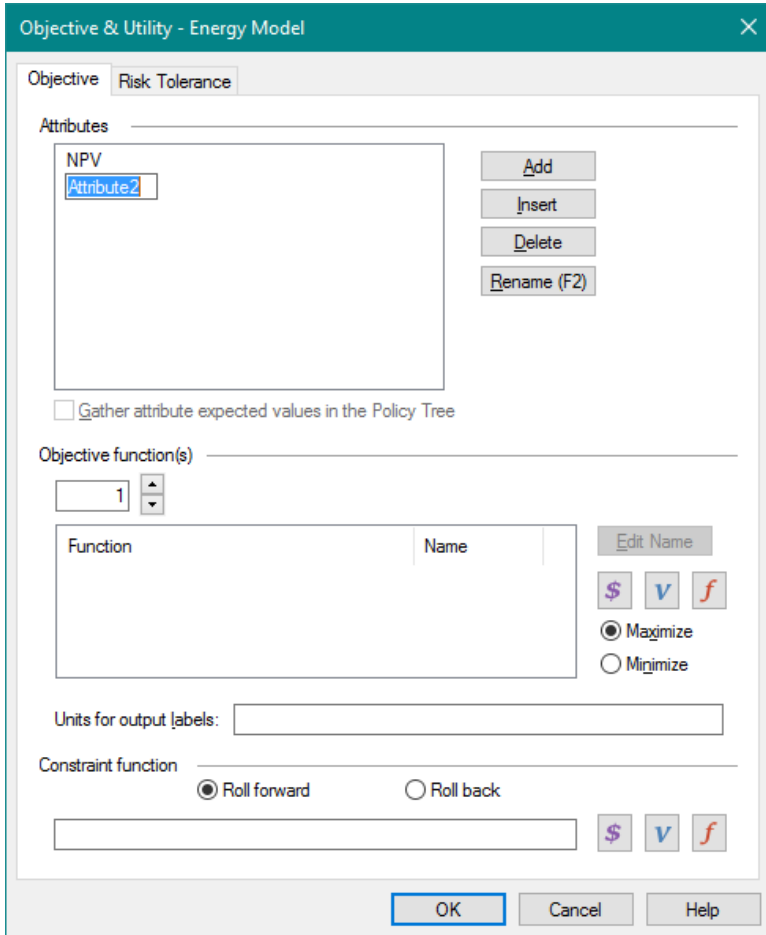


Figure 10-2. Objective & Utility dialog

Currently there is only one attribute defined in the model. It is called NPV. When there is only one attribute defined in the model and no explicit objective function has been defined, DPL maximizes the Get/Pay expressions specified in the Decision Tree. Therefore, the objective function for the model is to maximize NPV. Furthermore, the model contains two Get/Pay expressions: Upfront\_NPV (which applies if the Proceed decision is No), and Total\_NPV (which applies if the Proceed decision is Yes). So DPL is maximizing Upfront\_NPV and Total\_NPV depending upon the path through the tree.

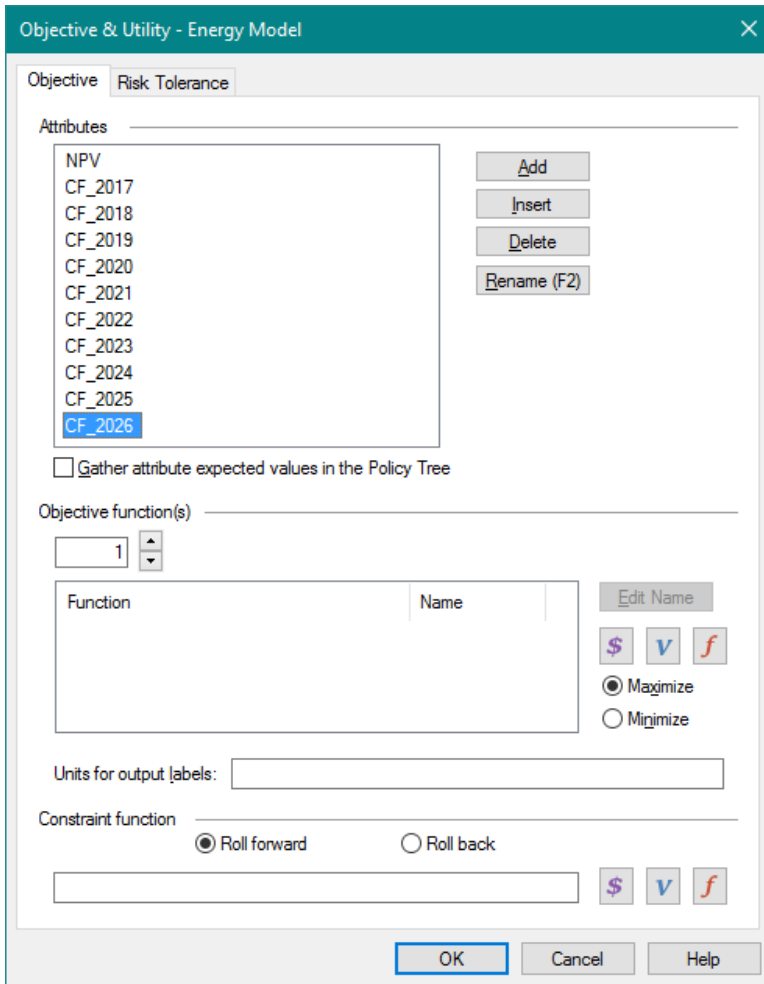
You will now add 10 attributes to the model for the cash flow information from Excel.

- ⇒ Click Add to create another attribute. DPL names it Attribute2 by default. See Figure 10-3.



**Figure 10-3. Objective & Utility dialog with Attribute2 Added**

- ⇒ DPL automatically puts you in edit mode. Type "CF\_2017" to rename Attribute2.
- ⇒ Press Enter.
- ⇒ Click Add again. DPL will add an attribute named CF\_2018, which is what you want.
- ⇒ Click Add eight more times to create attributes named CF\_2019, etc., until you have created ten new attributes altogether, the last of which will be named "CF\_2026". The Objective & Utility dialog should look like Figure 10-4.



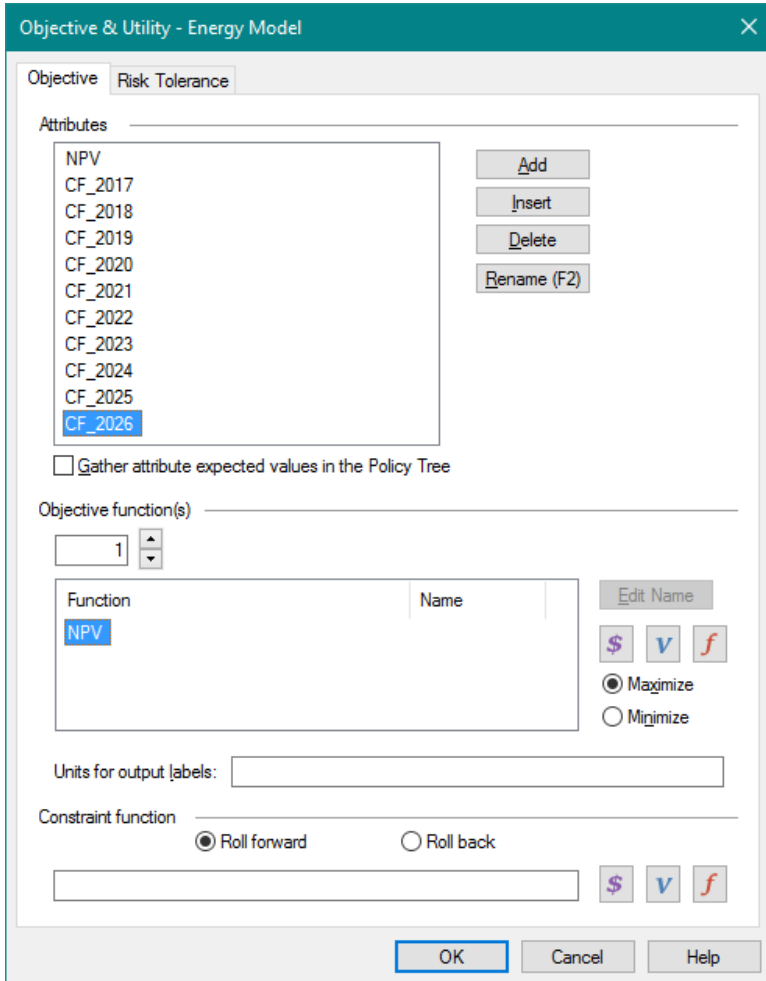
**Figure 10-4. Objective & Utility dialog with Ten New Attributes**

Note: DPL will automatically increment from the previous attribute name any name that ends with a number when you add an attribute. This makes it easy to add sequential attributes as in this example. If you did not want a sequentially named attribute, you can always rename it.

Now that there are multiple attributes defined in the model, you need to tell DPL explicitly what the objective function is. If you do not specify an objective function when there are multiple attributes defined, DPL will maximize the sum of the attributes by default. You do not need the ten new attributes for the objective function. The objective function is still simply NPV.

- ⇒ Click in the Objective function edit box.
- ⇒ Type "NPV" or use the select attribute button (\$) to select NPV for the objective function.

The maximize radio button is already selected. DPL will now maximize NPV for the objective function. The Objective & Utility dialog should look like Figure 10-5. You will make one further change so that you can view attribute expected values in the Policy Tree.



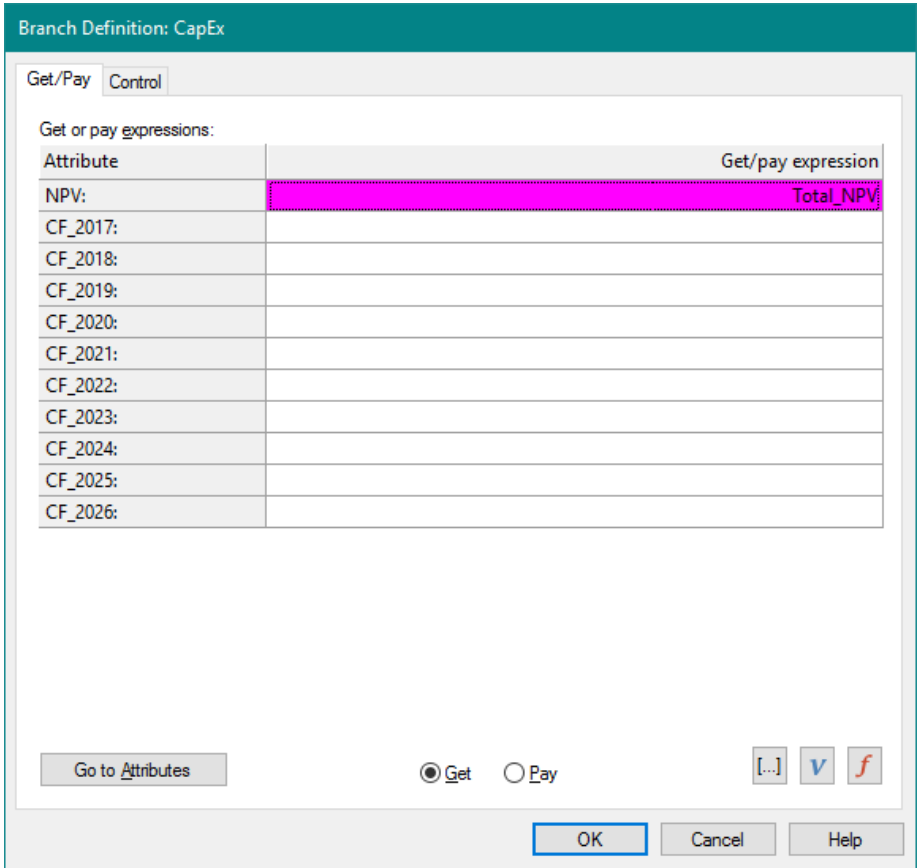
**Figure 10-5. Objective & Utility dialog with 10 Attributes and Objective Function Defined**

- ⇒ Check *Gather attribute expected values in Policy Tree* underneath the attributes list box.
- ⇒ Click OK.

The last step to set up Time Series Percentiles is to change the Get/Pay expressions in the Decision Tree to tell DPL which values should be used for each attribute. For more information on specifying Get/Pay expressions for multiple attributes, see Chapter 8, as well as the last section of this chapter.

- ⇒ Press the Tab key to switch to the Decision Tree pane view. You may notice that there are several commas (one for each attribute) on the branches that previously contained a get/pay expression.
- ⇒ Double-click the branches of the CapEx node. The Get/Pay tab of the Branch Definition dialog appears as shown in Figure 10-6.






**Figure 10-6. Get/Pay tab of Branch Definition Dialog**

Now that there are multiple attributes defined in the model, DPL provides room on the Get/Pay tab of the Branch Definition dialog for separate expressions for each attribute. When you have multiple attributes in a model, you must specify a value for each of them in every Get/Pay expression (or use zero as a placeholder). The Get/Pay expression as it is currently defined only has a value (Total\_NPV) for the first attribute (called NPV).

Your model is set up so that the ten new attributes are determined by a single, 1 x 10 value node. You will now set up the get/pay expressions using this value node.

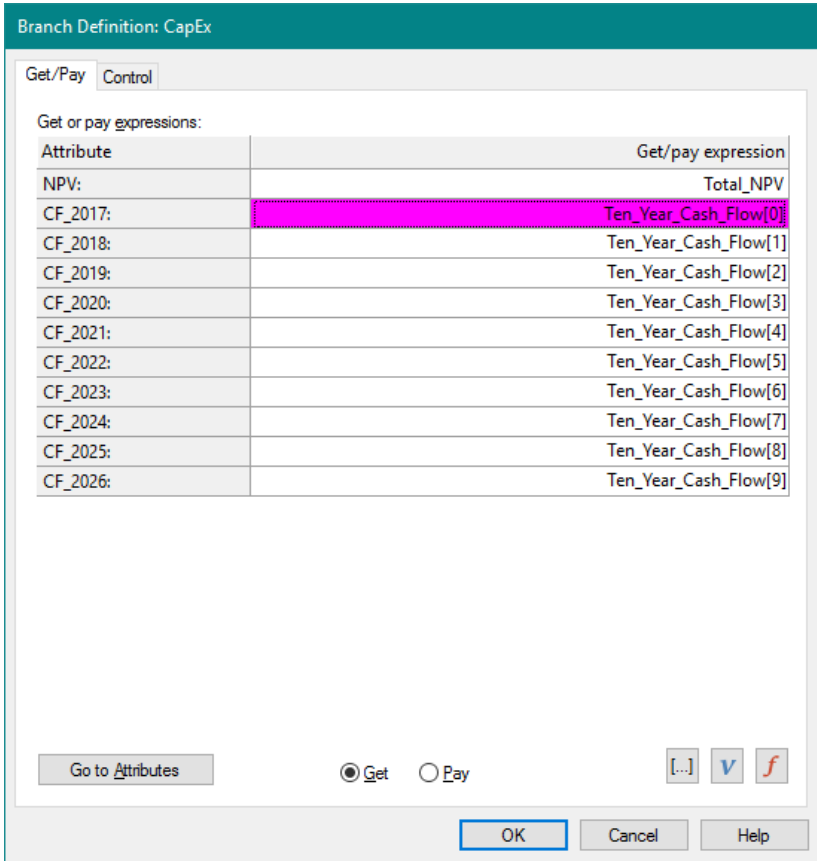
- ⇒ Click in the expression cell for the CF\_2017 attribute (the second cell).
- ⇒ Click the variable button (  ).

- ⇒ Select `Ten_Year_Cash_Flow` from the list of variables.
- ⇒ Click OK to return to the Branch Definition dialog.

The Get/Pay for `CF_2017` is now defined as `Ten_Year_Cash_Flow`, but as you know, this variable is a 10-element series. You want the first element (`Ten_Year_Cash_Flow[0]`) to be used for the attribute `CF_2017`. You can have DPL do this for you and set up the expressions for the remaining attributes with one button click.

- ⇒ Click the Fill Down button ().

DPL initializes the second attribute and the remaining nine attributes to be the elements of the `Ten_Year_Cash_Flow` value node: `Ten_Year_Cash_Flow[0]` through `Ten_Year_Cash_Flow[9]`. The dialog should now look like Figure 10-7.



**Figure 10-7. Get/Pay tab of Branch Definition Dialog with New Get/Pay Expressions**

More information is given in Section 10.3 on how to use the Fill Down button.

- ⇒ Click OK. Zoom Full the Decision Tree pane to view the entire get/pay expression.

You can see in the Decision Tree pane that DPL has added the ten new attributes to the Get/Pay expression on the CapEx node of the Decision Tree. This causes the branches of the node to be quite long so you'll display an indicator instead.

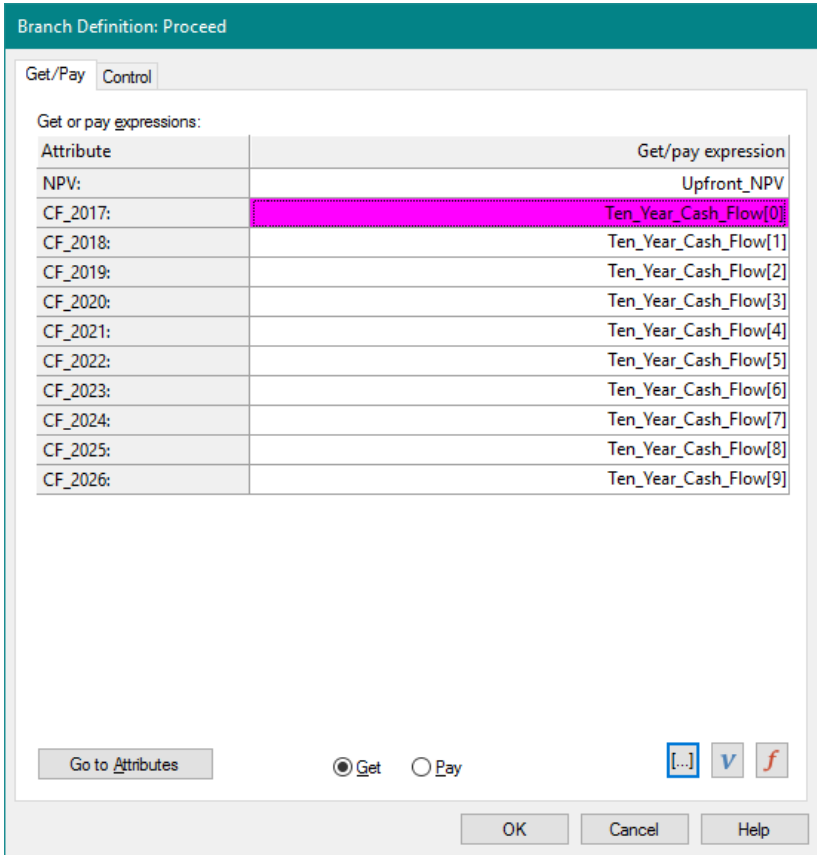
- ⇒ Select the branches of the CapEx node in the Decision Tree pane.

- ⇒ In the Decision Tree | Get/Pay group, select the *Indicator* radio button. (Note that you could alternatively choose to show nothing by selecting the Hide radio button.)
- ⇒ Click Zoom Full again.

An indicator (\$) that a Get/Pay expression exists on these branches is now displayed on the CapEx node's branches. The branches have been shrunk since the full get/pay no longer appear there.

Next you will repeat this process for the other tree branch that has a Get/Pay expression: the No branch of the Proceed decision.

- ⇒ First, select the No branch of the Proceed decision node in the Decision Tree, and again, select the *Indicator* radio button in the Decision Tree | Get/Pay group.
- ⇒ Repeat the above process to add the Ten\_Year\_Cash\_Flow value node to the Get/Pay expression defined for the No branch of the Proceed decision. Note that the value for the NPV attribute for this branch is Upfront\_NPV. See Figure 10-8.



**Figure 10-8. Get/Paytab of Branch Definition Dialog for Proceed = No, with New Get/Pay Expressions**

Note that the same value node is being used for the ten years of cash flow in both Get/Pay expressions. This works because the logic is set up in the Excel spreadsheet so that these ten annual cash flow values reflect whether the Proceed decision is Yes or No. If you'd like to see this logic you can set the Proceed decision switch, cell C53, to 1 and the cash flows will change. The model was set up this way for ease of illustration. In other situations, the spreadsheet might have been set up differently, so that an entirely different value node (or group of value nodes) would be used for each Get/Pay in the Decision Tree.

Note: you can copy and paste Get/Pay expressions from one branch to another. There are two ways to this. If you wish to copy the entire Get/Pay expression for all attributes (or if it is a single attribute model), select the branch whose Get/Pay you'd like to copy and press Ctrl+C and then select

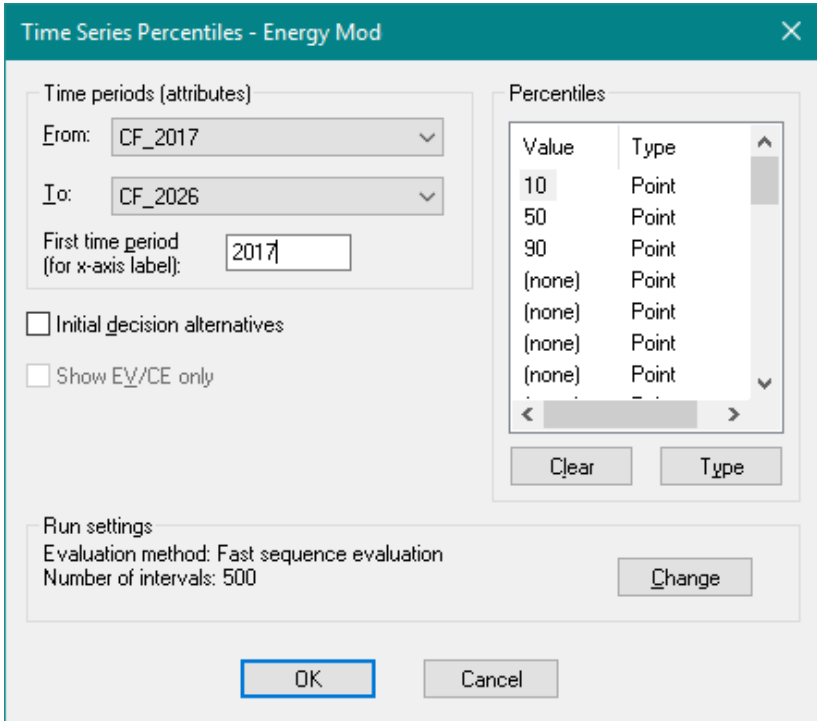
the branch where you would like to paste the Get/Pay and press Ctrl+V. If you'd like to copy a subset of the Get/Pay expression you can do this via the Get/Pay tab of the Branch Definition dialog. Double-click the branch for which you'd like to copy a portion of the Get/Pay, select a range of cells in the Get/Pay Definition dialog and press Ctrl+C. Then double-click the branches where you wish to paste the Get/Pay expressions, select the appropriate cell in the Branch Definition dialog and press Ctrl+V.

You have now specified a value for each of the model attributes on both branches of the tree that require a Get/Pay expression. You are ready to run Time Series Percentiles.

- ⇒ If you wish, save your Workspace under a new name of your choice. (This is not essential, but you may find it useful for future reference, if you intend to use Time Series Percentiles frequently.)
- ⇒ Click Home | Sensitivity | Time Series. The Time Series Percentiles dialog appears.

In the Time Series Percentile dialog, you need to confirm the first and last attribute to include in the run. If you name your attributes with numbers at the end, DPL will attempt to set the From and To drop-down lists in the *Time Period (attributes)* section for you. If the settings are incorrect, change the From and To drop-down lists.

- ⇒ Confirm that the *From:* drop-down list is set to CF\_2017. If not, select it.
- ⇒ Confirm that the *To:* drop-down list is set to CF\_2026. If not, select it.
- ⇒ DPL will also set the *First time period (for x-axis label)* edit box based on the initial attributes selected for the To and From attributes. Confirm it is set to 2017. The dialog should look like Figure 10-9.



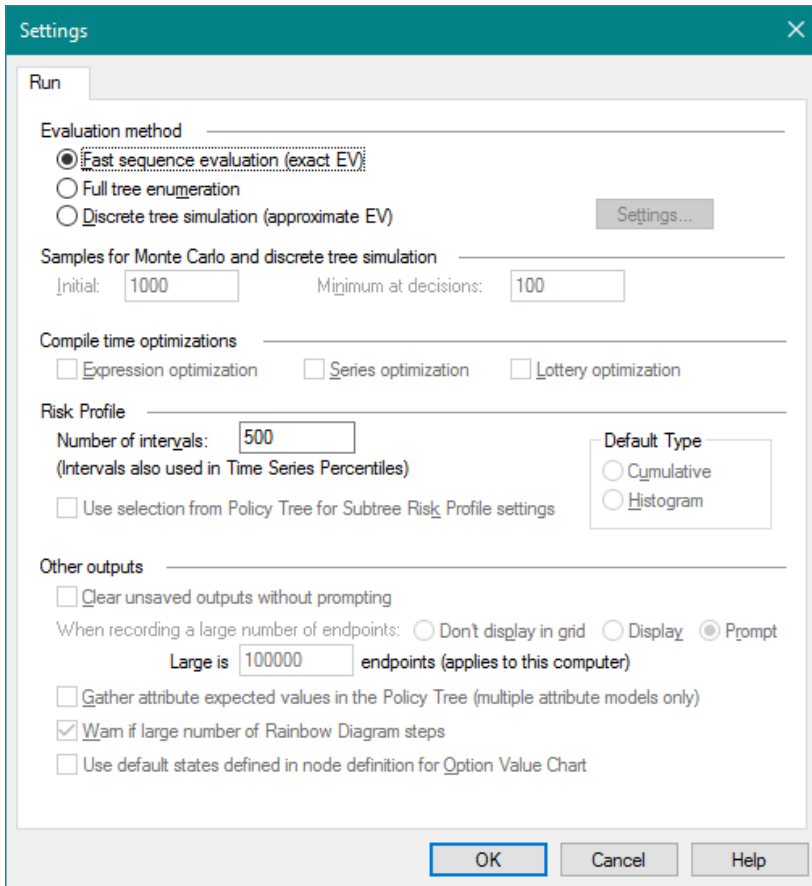
**Figure 10-9. Time Series Percentiles Dialog After Edits**

The first time period on the x-axis of the Time Series Percentiles chart is labeled using the data entered into the First time period box. Time periods are labeled sequentially after that.

DPL displays the current evaluation method and the number of intervals in the *Run settings* section.

In the *Percentiles* box you can specify the percentiles you would like included in the chart. You may edit the existing three to change them to a different percentile, type a number in a slot that says "(none)" to add one or select a number and press Clear to remove it. If you'd like two percentiles displayed as a range bar, select a percentile and use the Type button to toggle its type. You must specify both a range min and range max to display a range bar. You may also set the type by right-mouse clicking the Percentiles box and using the context menu.

- ⇒ Within the *Run Settings* section, click the Change button to see the Run Settings dialog. See Figure 10-10.



**Figure 10-10. Run Settings dialog**

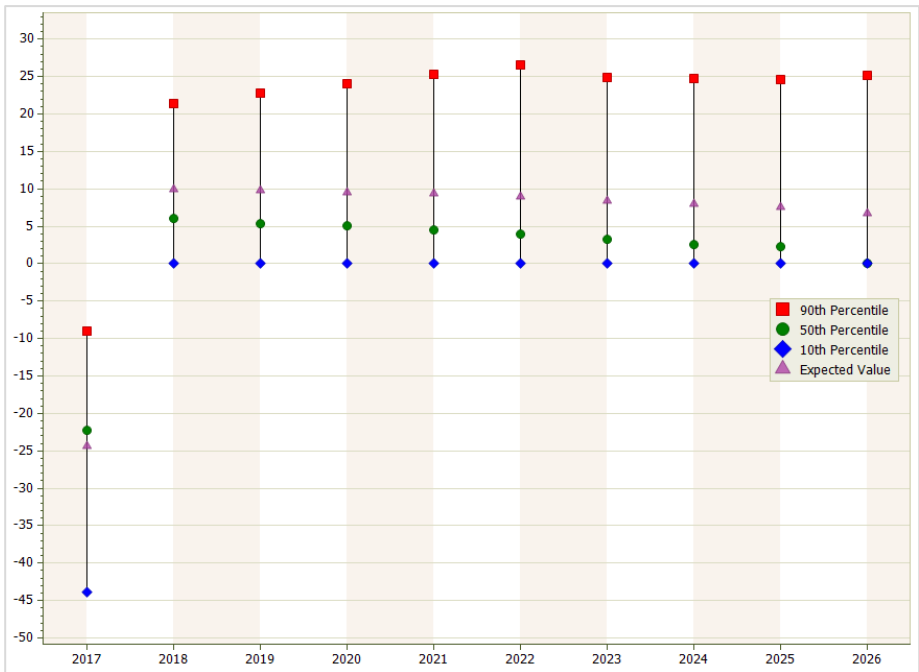
Risk Profiles underlie the Time Series Percentiles output. In the Run Settings dialog, you can change the number of intervals to be used to accumulate Risk Profiles for the Time Series Percentiles chart within the *Risk Profile* section of the dialog. This setting is the number of "bins" into which the outcome values will be grouped. The accuracy of the percentiles depends on the number of intervals setting. The more intervals you use, the greater the accuracy, however, more intervals requires more memory and will have an impact on model runtime. Usually, 500 intervals should suffice. If you have extremely long tails in your distributions, you may wish to increase the number.



Note the number of intervals setting is also used for Risk Profiles in a Decision Analysis run. If you change the setting here, your new setting will also be used the next time you run a Decision Analysis and generate a Risk Profile. If you'd like to change these settings prior to a Decision Analysis run you can access the Run Settings dialog via Home | Run | Settings. Also see Section 3.2 for information on Risk Profiles and the Number of intervals setting.

- ⇒ Make sure the Number of intervals is 500.
- ⇒ Click OK twice to run a Time Series Percentiles chart.

Note: adding attributes to a model does have a performance impact. After a minute or two, DPL produces the Time Series Percentiles Chart shown in Figure 10-11.



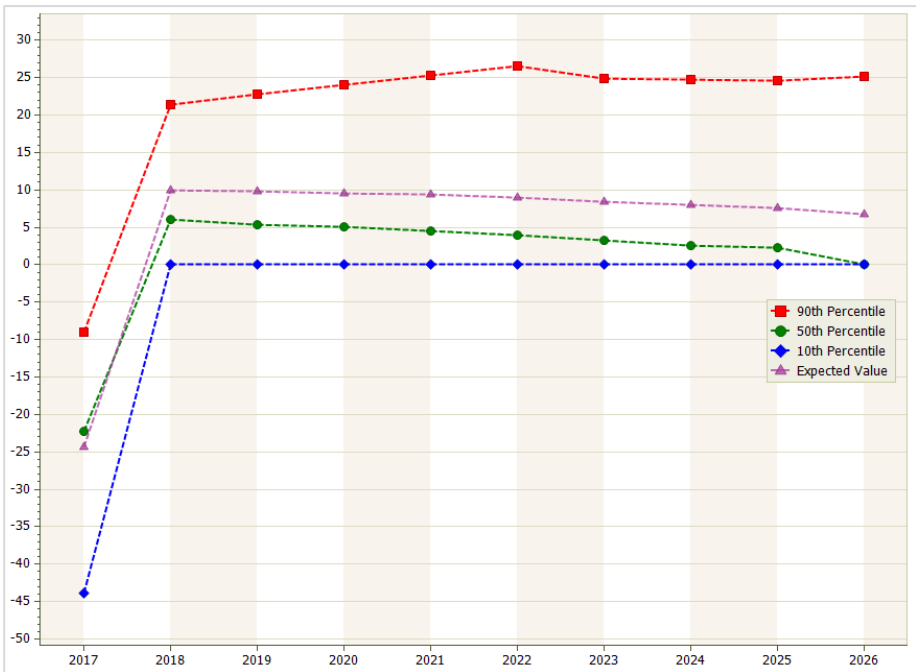
**Figure 10-11. Time Series Percentiles Chart**

DPL displays the 10th, 50th and 90th percentiles (or the percentiles/percentile ranges you specified) plus the expected value of the time series variable for each time period. In this case, the time series variable is annual cash flow, so Figure 10-11 displays the three percentiles and expected value of cash flow in each of the time periods. The 10th percentile is the value such that there is a ten percent chance that the time

series variable is less than that amount. The 90th percentile is the value such that there is a ten percent chance that the time series variable is greater than that amount. The 50th percentile is the value such that there is an equal chance that the time series variable is less than or greater than that amount. The expected value series is the expected value of the variable for the time period.

Finally, you can change several display options to tailor the appearance of the Time Series Percentiles chart to suit your needs.

- ⇒ With the Time Series Percentiles chart still active uncheck Error Bars in Chart | Format | Display.
- ⇒ In Chart | Series | Lines/Markers, make sure Lines and Dashed are checked. The chart will look like Figure 10-12.



**Figure 10-12. Revised Time Series Percentiles Chart**

The revised chart connects the points for each series (10<sup>th</sup> percentile, etc.) over time. Technically, how the uncertainty in the attributes evolves between time periods cannot be derived from this output. That is, the percentiles for points in time in between the discrete time periods are not known. However, you may find this depiction of the percentiles more useful than the error bar depiction for communication purposes.

Also note that within the Chart | Format tab, you can change the Titles, Subtitles, Axis labels, Legend, marker/line color, and background display of the Time Series Percentiles chart. To format the font within the chart select the font and use the commands in the View | Font group. Try this if you like.

See Figure 10-13 for an example of what the Time Series Percentile dialog looks like when you specify two of the percentiles as the minimum and maximum of a range bar.

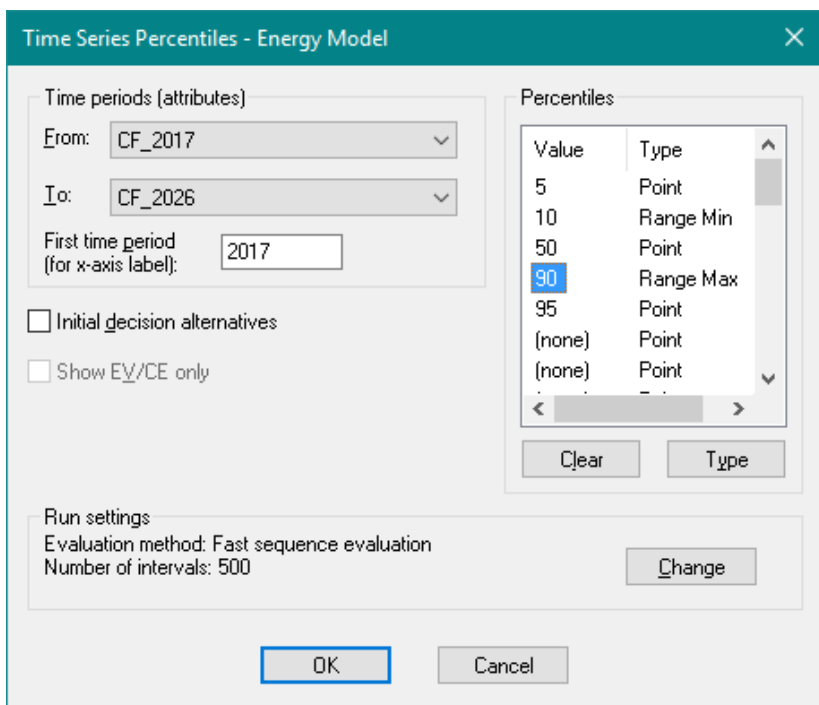
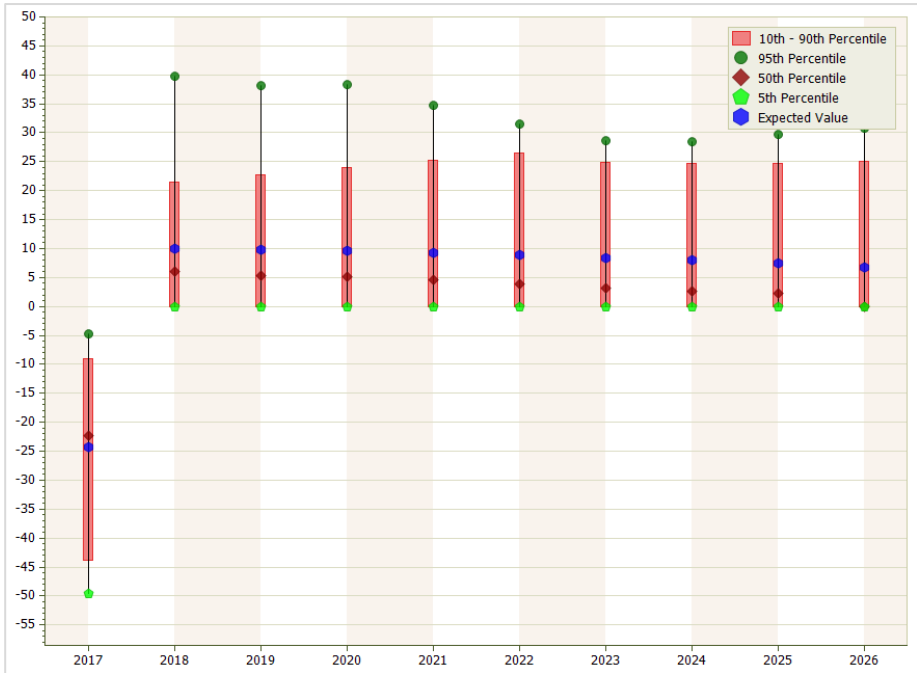


Figure 10-13. Time Series Percentiles Dialog for Range Bar

The settings in Figure 10-13 produce the Time Series Percentile Chart displayed in Figure 10-14. You can see that the 95<sup>th</sup> percentile of the cash flows is greater than the 90<sup>th</sup> while in most time periods the 5<sup>th</sup> percentile is the same as the 10<sup>th</sup>. As specified, the 10<sup>th</sup> and 90<sup>th</sup> percentiles are displayed as the minimum and maximum of the range bar.



**Figure 10-14. Time Series Percentiles Chart with Range Bars**

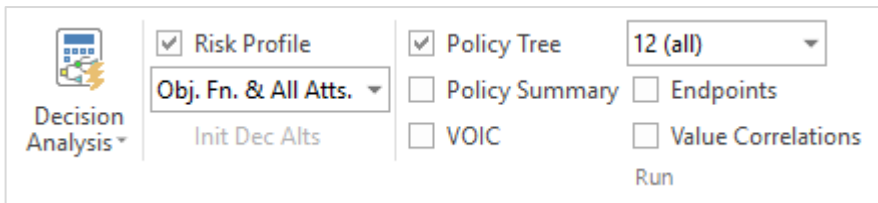
The Time Series Percentiles Chart does not provide all the information you might need about the full risk profile for each of the ten attributes (cash flows); it gives you up to seven percentiles plus the expected values. However, more information about the uncertainty in each attribute is available by generating Risk Profiles from a Decision Analysis run. You will examine this feature in the next section.

## 10.2 Risk Profiles and Policy Trees™ for Multiple Attributes

In the previous section, you set up a Time Series Percentiles analysis to look at ten attributes (years of cash flow) in an energy model. In some situations, you might want more detailed information about expected value given a particular scenario or the full risk profile (range of uncertainty) for one or more attributes. You will continue the example from the previous section to see how this information can be obtained via a single Decision Analysis run.

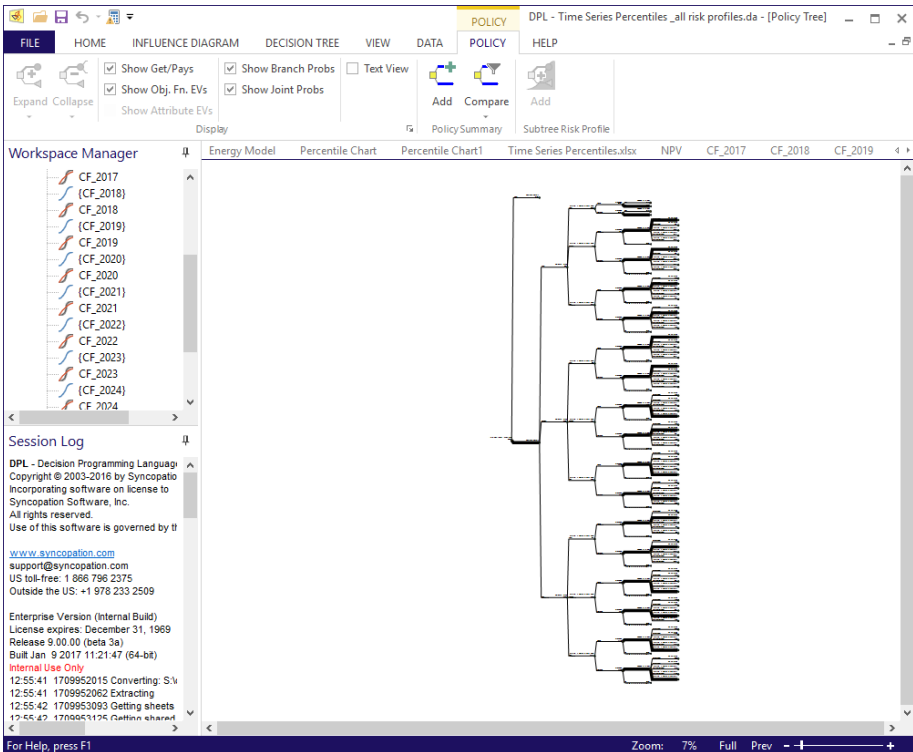
### 10.2.1 Controlling the Display of Attribute Expected Values

- ⇒ Activate the "Energy Model" in the Model Window.
- ⇒ In the Home | Run group, make sure that Risk Profile and Policy Tree are the only selected outputs.
- ⇒ Use the Select Risk Profile Quantity drop-down list (just below the Risk Profile checkbox) to select Objective Function and All Attributes ("Obj. Fn. & All Atts."). This will generate the full Risk Profile for the objective function and every attribute in your model. See Figure 10-15.
- ⇒ Click Home | Run | Decision Analysis or press F10.



**Figure 10-15. Generating Risk Profiles for the Objective Function and All Attributes**

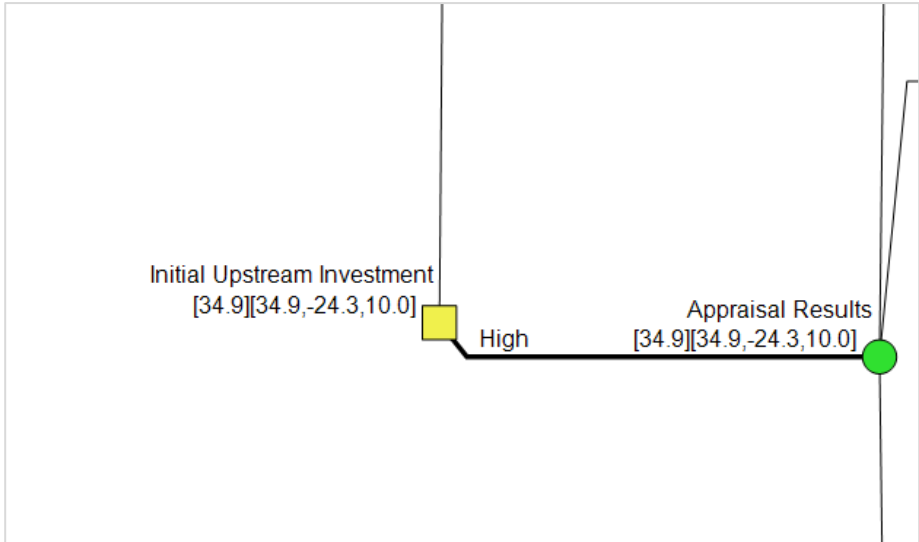
The model may take a few minutes to run. As mentioned earlier, adding attributes to a model does have a performance impact, however, requesting a single Risk Profile vs. multiple Risk Profiles does not. When it completes, you will see Risk Profile charts and datasets for the objective function plus all the attributes in the Workspace Manager, as in Figure 10-16. The Policy Tree output will be displayed.



**Figure 10-16. Workspace Manager with All Risk Profiles and Policy Tree™ Displayed**

Before viewing some of the Risk Profiles, let's take a moment to learn how to control which attributes are shown within the Policy Tree. Initially DPL will only display the first 3 attributes defined within the Objective & Utility dialog, the first of which is NPV – which is also the objective function for the model.

- ⇒ Do a Ctrl+Shift+Drag (zoom by selection) on a small section of the Policy Tree in order to view the attribute values displayed as shown in Figure 10-17.



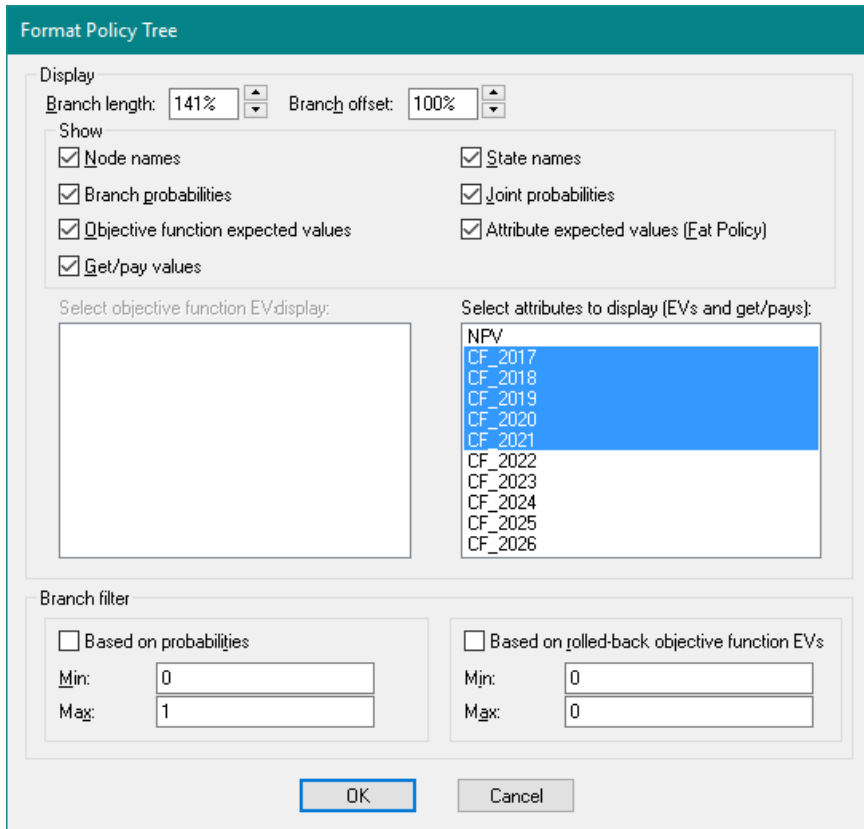
**Figure 10-17. Zoom by Selection on Attribute Values within the Policy Tree™**

The expected value of the objective function (NPV) is listed first within the first set of brackets. Then the expected values of the first three attributes are listed within the second set of brackets to the right of the objective function expected value. As mentioned above, the value for the objective function and the first attribute are the same (both are NPV). This is a redundancy you may want to correct. Let's assume that you are most interested in the displaying the values for the first 5 years of cash flow.

⇒ Select Policy | Display | Settings (the dialog box launcher).

Note that within the *Select attributes to display* box the first three attributes included in the display are selected. Via the list box you can pick any single, subset (can be non-consecutive), or all attributes to be displayed within the Policy Tree.

⇒ Select the attributes CF\_2017 through CF\_2021. Your selection should match Figure 10-18.



**Figure 10-18. Format Policy Tree Dialog with 5 Attributes Selected for Display**

- ⇒ Click OK to close the dialog and update the Policy Tree display.
- ⇒ Do Ctrl+Shift+Drag to zoom the same area as before (Figure 10-19).

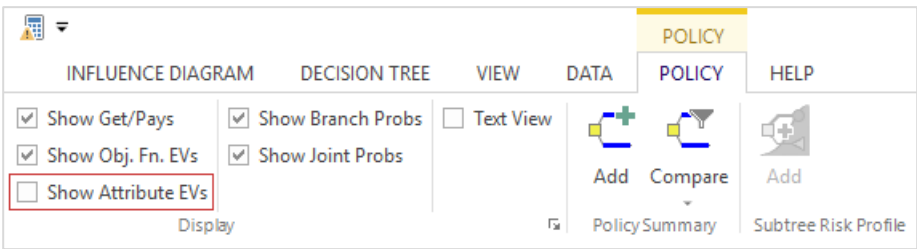




**Figure 10-19. Zoom by Selection on 5 Attribute Values within the Policy Tree™**

You'll see that DPL has updated the display to include the values for the five attributes selected within the Format Policy Tree dialog.

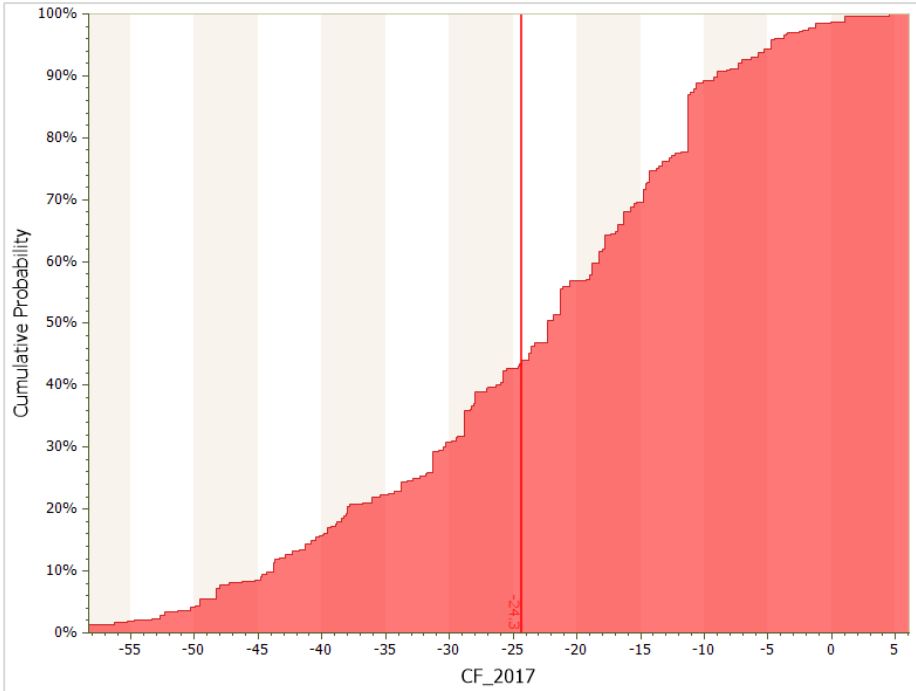
If you wanted to turn off the display of attributes completely so that only the Expected Value is displayed (and no attributes values) you could uncheck the Show Attribute EVs box within the Policy Display group (Figure 10-20).



**Figure 10-20. Policy | Display | Show Attribute EVs turned off**

**10.2.2 Viewing and Modifying Attribute Risk Profiles**

- ⇒ Double-click on the Risk Profile chart for CF\_2017 to open it. You can see that there is a substantial amount of uncertainty regarding the cash flow in 2017; due to the initial investments required, the net cash flow will probably be negative. See Figure 10-21.

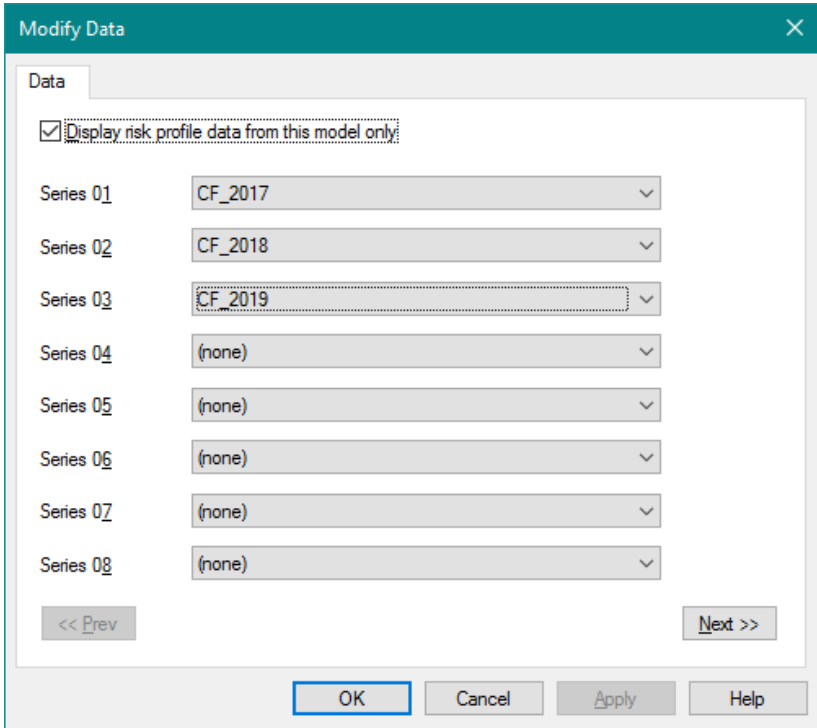


**Figure 10-21. Risk Profile for CF\_2014**

If you want to see detailed statistics on the probability distribution for cash flows in 2017, you can click [Data | Distribution | Statistics](#).

For comparison purposes, you might want to view multiple attributes on one Risk Profile chart.

- ⇒ With the CF\_2017 chart still active, click [Chart | Series | Modify](#).
- ⇒ You can display up to 32 Risk Profile datasets on a single Risk Profile chart. Leave the first series set to CF\_2017. Use the drop-down lists to fill in CF\_2018 and CF\_2019 for Series 2 and Series 3 in the [Modify Data](#) dialog. See Figure 10-22.



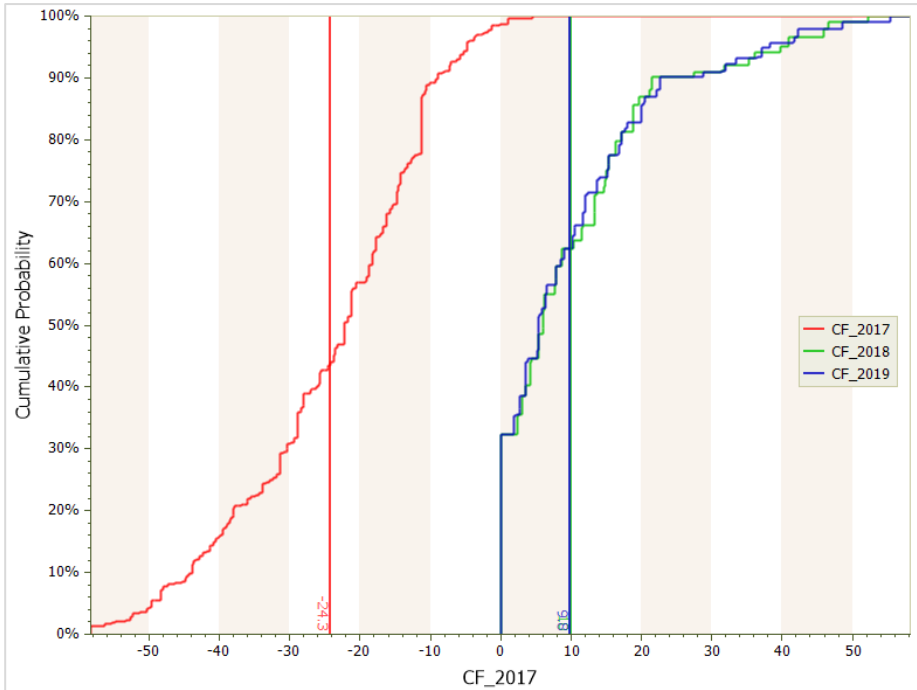
**Figure 10-22. Modify Data dialog to Display Multiple Risk Profiles**

- ⇒ Press Ok.
- ⇒ In the Chart | Format | Legend group, check the checkbox next to Show. You can use the other checkboxes and drop-down list within the group to format the placement and display of the legend.

At times it may be easier to view the multiple probability distributions on a single chart as lines without a color fill. To turn off color fill:

- ⇒ Uncheck the Color Fill checkbox in the Chart | Format | Color group.

The Risk Profile should look something like Figure 10-23.



**Figure 10-23. Risk Profiles for 2017-2019 Cash Flows**

You can see from the Risk Profiles that the probability distributions for 2018 and 2019 cash flow are very similar; each cash flow has about a 30% chance of being zero, and neither will be more than 60. The probability distribution for 2017 cash flow is quite different since this is the year when the initial investments are made. As you saw earlier, the net cash flow in this year will very likely be negative, and could even be less than -50.

⇒ You can close your Workspace without saving changes; it is not needed for the last sections of this chapter.

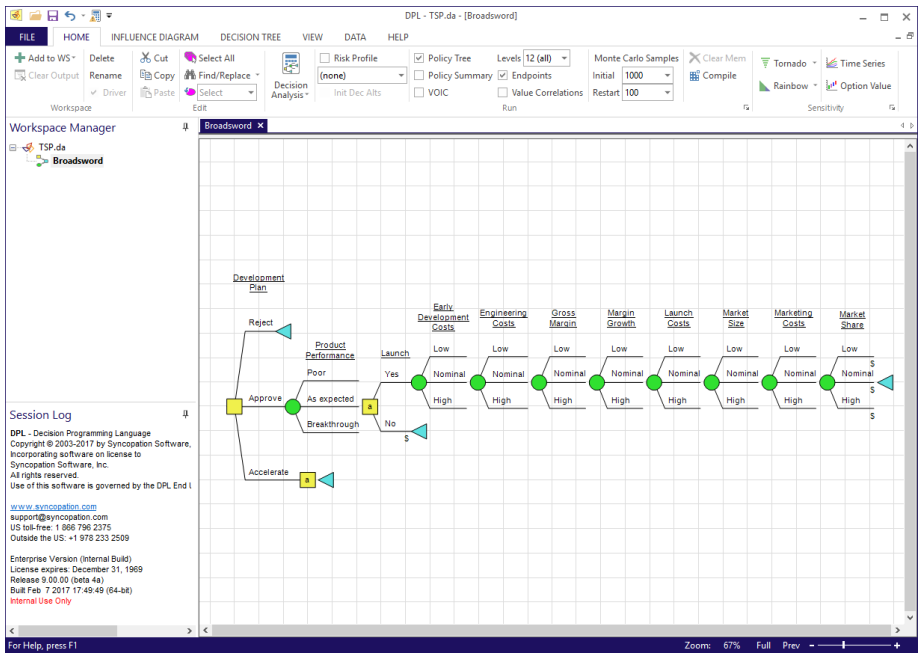
## 10.3 Initial Decision Alternatives Time Series Percentiles

In this tutorial, you use a new example model that analyzes a new product development decision where there is significant uncertainties about its performance and commercial viability if launched. Alternatives on the table include terminating the project, proceeding as planned, and accelerating to

reach the market one year sooner. There is also an explicit downstream launch decision. A spreadsheet and model have been prepared to evaluate the development alternatives.

- ⇒ Select File | Open.
- ⇒ Open the Broadword\_TSP.da. This file is found in the Examples folder where DPL was installed, usually C:\Program Files\Syncoption\DPL9\Examples.

DPL opens the Workspace as shown in Figure 10-24

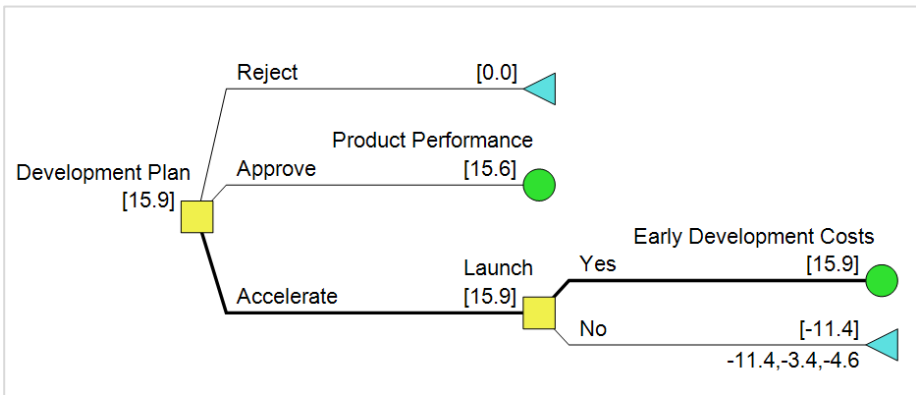


**Figure 10-24. New Product Development Model**

The model contains two asymmetric decision nodes, an initial decision regarding the development plan and a downstream exit option to launch the product or not. For the Approve alternative of the Development Plan decision, the Launch decision occurs after you've learned about the product's performance. You can only gain this product performance information if you proceed as planned (i.e., you don't accelerate the project). You'll run the model to see which development strategy looks optimal.

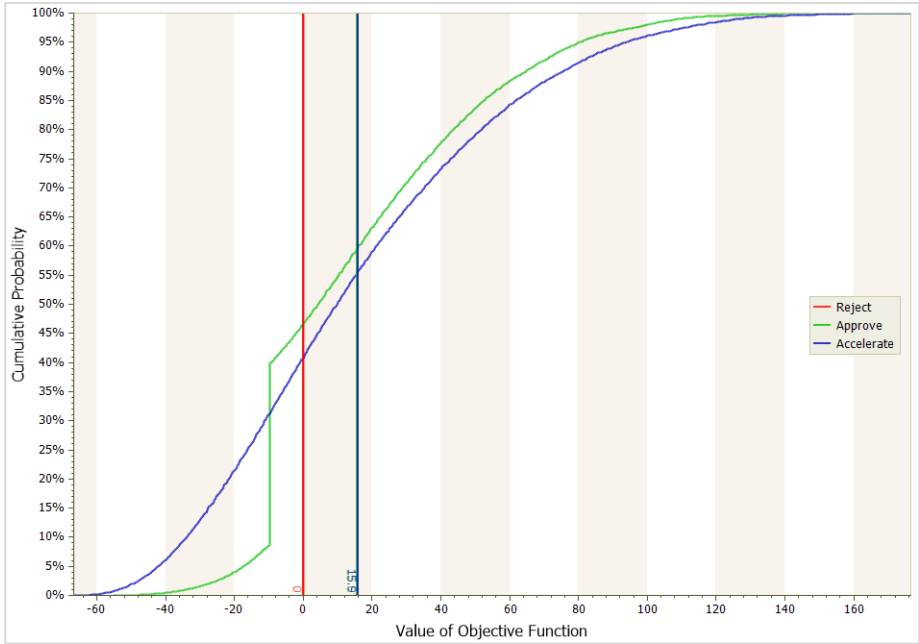
- ⇒ Within the Home | Run group make sure Risk Profile, Init Dec Alts, and Policy Tree are checked.
- ⇒ Select Home | Run | Decision Analysis (or press F10).

DPL will display the Policy Tree (Figure 10-25). You can see that the optimal decision policy is to accelerate the development project and proceed with launching the product. But you should note that there is a very small difference between the expected values for the Approve and Accelerate alternatives (15.9 v. 15.6). Let's look at a comparison of the Risk Profiles for the alternatives.



**Figure 10-25. Policy Tree for New Product Development Model**

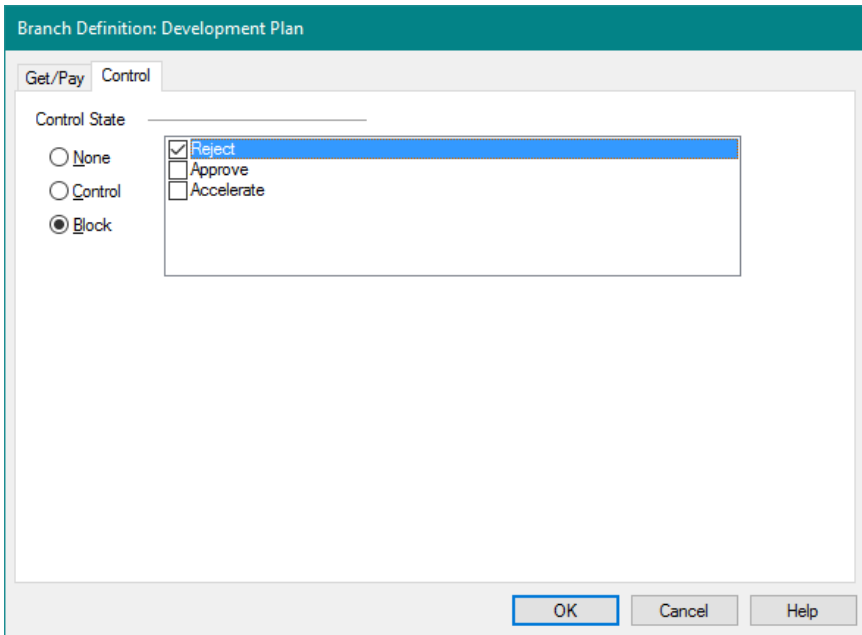
- ⇒ Double-click the item for the Initial Decision Alternatives within the Workspace Manager to view the output.



**Figure 10-26. Initial Decision Alternatives Chart for New Product Development Model**

The Risk Profiles for the Approve and Accelerate alternatives again show this to be a difficult choice. The Accelerate alternative is mostly to the right of the of the Approve alternative but it is more risky, as indicated in the lower percentiles. To gain a clearer picture of how the uncertainty for each alternative evolves over time, you'll generate a Time Series Percentiles chart for each Initial Decision Alternative. You'll block the Reject decision alternative (i.e., it will be ignored during the analysis) because it would simply be a line at zero. You'll do this now.

- ⇒ Activate the Broadsword model.
- ⇒ Within the Decision Tree, double-click the Reject branch of the Development Plan decision.
- ⇒ Switch to the Control tab of the Branch Definition dialog.
- ⇒ Set the *Control State* radio button to Block.
- ⇒ Check the box next to the Reject alternative as shown in Figure 10-27.



**Figure 10-27. Control Tab of Branch Definition Dialog Set to Block**

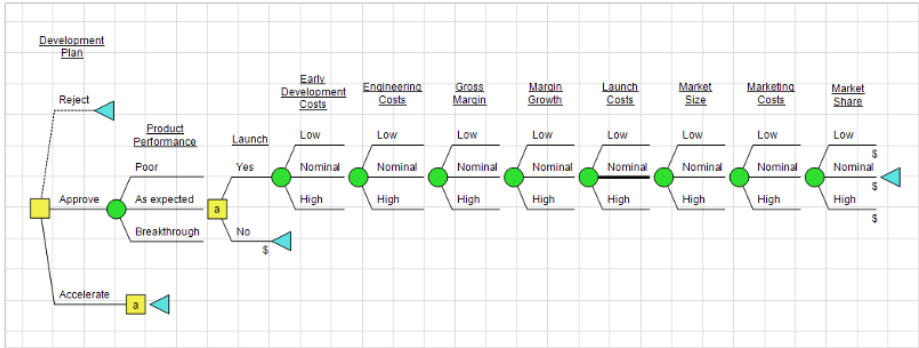
⇒ Click OK to close the dialog.

DPL has redrawn the Reject branch as a dotted line to indicate that it is a blocked alternative to be ignored. Furthermore, you have good information that launch costs will be less volatile than expected so you'll control this uncertainty to its nominal outcome. Modeling variables deterministically when appropriate can help to speed up your analysis.

⇒ Select the branches of Launch Costs.

⇒ Select the Nominal outcome within the Decision Tree | Control/Block | Branch Control drop-down. Your Decision Tree should now look like Figure 10-28.





**Figure 10-28. Decision Tree with Branches Blocked/Controlled**

You're now ready to generate the Time Series Percentiles for the remaining two initial decision alternatives. The steps for setting up time series percentiles outlined in Section 10.1 have already been completed for the model. You may wish to review that section if you haven't already. You will now create a Time Series Percentiles chart.

- ⇒ Select Home | Sensitivity | Time Series.
- ⇒ With numerically named attributes, DPL will attempt to set the *From:* *To:* attributes for you. They should be set at Y19 and Y28, respectively.
- ⇒ DPL also set the *First time period* to be 19 but you want 2019. Enter "2019" within the *First time period* box.
- ⇒ Check the box labeled *Initial decision alternatives*. This tells DPL to generate percentiles for both the Approve and Accelerate alternative (remember that Reject alternative will be ignored).
- ⇒ Check the box labeled *Show EV/CE only*. For now, you'll generate only the expected values of cash flows for each alternative.

⇒ Press OK to run the analysis. See Figure 10-29.



**Figure 10-29. Time Series Percentiles for Initial Decision Alternatives, EVs only**

You can see there is a large dip in cash flows in the year 2021 and 2022 for the Accelerate and Approve alternatives, respectively. The cash flow for Accelerate bottoms out much lower – resulting in more risk. But cash flow bounces back more steeply and quickly in 2022 if you Accelerate. To gain more information, you'll generate another chart with additional percentiles.

- ⇒ Activate the Broadword model.
- ⇒ Select Home | Sensitivity | Time Series. The *From:*, *To:* and *First time period* should all be as before (Y19, Y28, 2019, respectively.)
- ⇒ Check the *Initial Decision Alternatives* box.
- ⇒ Within the Percentiles list box select 50 within the Value column and press the Clear button. You only want to the 10<sup>th</sup> and 90<sup>th</sup> percentiles displayed in the chart.
- ⇒ Press OK to generate the output (Figure 10-30).

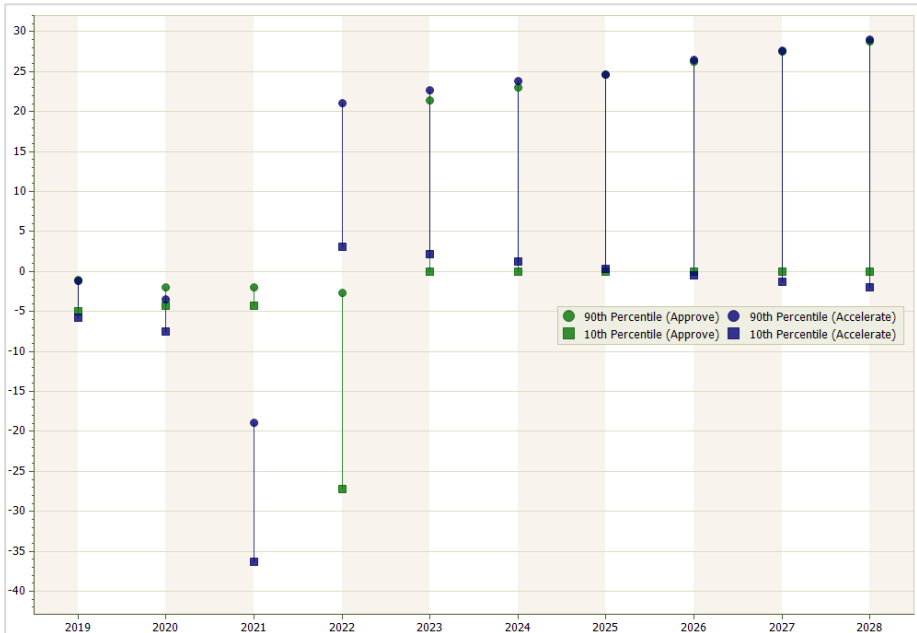


**Figure 10-30. Time Series Percentiles for Initial Decision Alternatives, 10<sup>th</sup>, 90<sup>th</sup>, and EVs**

The chart displays error bars representing the 10<sup>th</sup> and 90<sup>th</sup> percentiles plus the Expected Value for each time period within the time series variable (cash flow). You already know how the Expected Values compare from the first chart generated. You can hide these bars from display for a less cluttered chart if you'd like.

- ⇒ Select any one of the green markers representing the Expected Value of the Approve alternative. This will cause all of the markers for this series to be selected.
- ⇒ Check the box labeled Hide within the Series | Point group.

⇒ Now do the same for the Expected Value of the Accelerate alternative. Your output should now match Figure 10-31.



**Figure 10-31. Time Series Percentiles for Initial Decision Alternatives, with EVs removed**

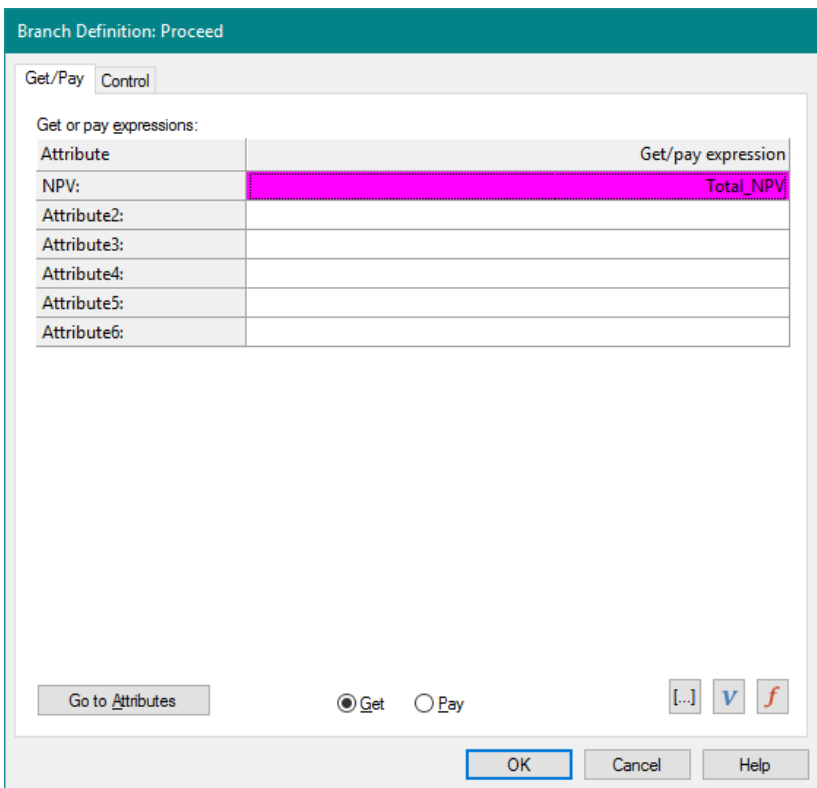
If you wanted to bring the EVs back to the display, you would click the dialog box launcher for the Series | Point group. Within the Unhide dialog select the items you wish to add back into the chart display and click OK.

## 10.4 Using Multidimensional Value Nodes in Get/Pay Expressions

As discussed in Chapter 8 and earlier in this chapter, any time you use DPL's multiple attributes feature, whether to set up Time Series Percentiles or to track multiple attributes for the objective function, you must set up your Get/Pay expressions to match the attributes you have defined. On any Decision Tree branch that has a get/pay, you need to have exactly as many Get/Pay expressions as the number of attributes you have defined.

If you use multidimensional value nodes (arrays or series) in a multiple attribute model, you may find that you need to refer to elements of these value nodes in your Get/Pay expressions. This section provides a brief discussion of how to do this using the Get/Pay tab of the Branch Definition dialog and it's Fill Down ( [...]) button, which you also used earlier to set up Time Series Percentiles.

Assume that you have set up a project evaluation model in DPL with six attributes. The first attribute corresponds to the objective function, and is called NPV. Management is concerned about the up-front spend on this project, so you must also track the other five attributes, which represent corporate investments in each of the first five years of the project. You have left the attribute names at their default for now: Attribute2 through Attribute6. You are setting up Get/Pay expressions on the Proceed branch of the tree. Initially, the Get/Pay tab of the Branch Definition dialog for this branch would look something like Figure 10-32.



**Figure 10-32. Get/Pay tab of Branch Definition Dialog for a Six-Attribute Model**

You need to define values for each of the five attributes other than NPV (which is already defined as Total\_NPV). There are several ways in which you might use a multidimensional value node to fill in get/pay expressions.

Assume first that you have defined a 1 x 5 (row) value node called Invest\_1x5 as shown in Figure 10-33.

Node Definition: Invest\_1x5

General Data Links

Columns: 5  Series interval entry  Create M x 1 column vector [...] [V] [f]

Column	Expression
0	30
1	20
2	18
3	15
4	12

OK Cancel Help

**Figure 10-33. A 1 x 5 Value Node Definition**

In order to use this value node to fill in the Get/Pay expressions on the Proceed branch, you would take the following steps:

- ⇒ Double-click the branch to open the Get/Pay tab of the Branch Definition dialog.
- ⇒ Click in the get or pay expressions cell for Attribute2.
- ⇒ Click the Select Variable button () and select Invest\_1x5 from the list. Click OK.
- ⇒ Click the Fill Down () button in the Get/Pay Definition dialog.

DPL will fill in the array elements Invest\_1x5[0], Invest\_1x5[1], and so forth for the Get/Pay expressions. The Get/Pay Definition dialog would now look like Figure 10-34. If there is data in any of the subsequent cells, you will get an overwrite data prompt before DPL fills in the subsequent values.

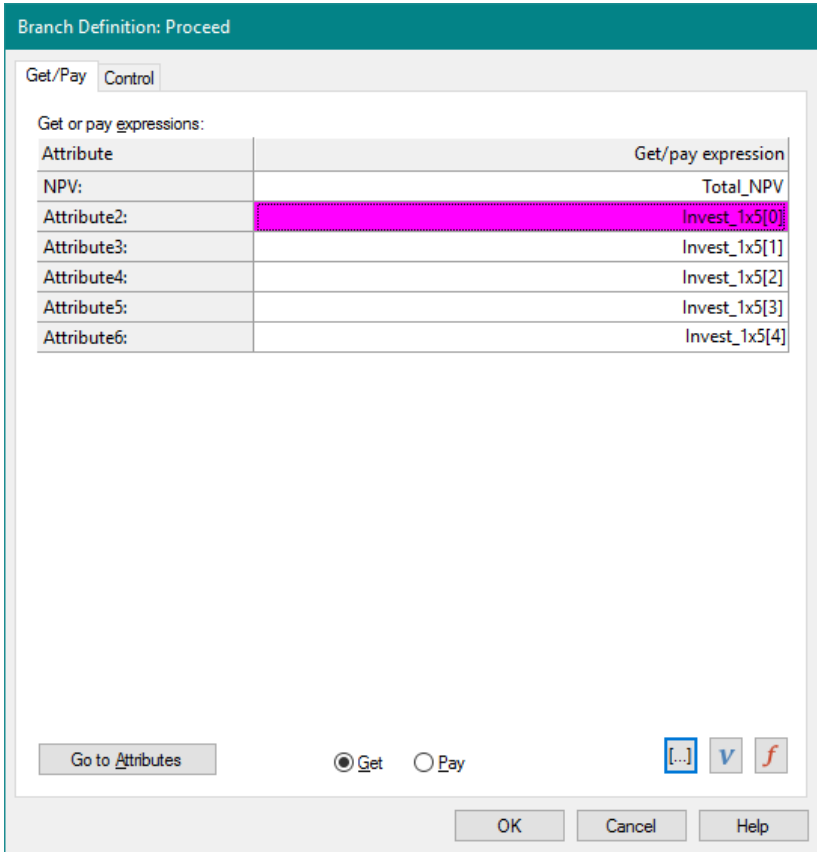



Figure 10-34. Get/Pay Definition Dialog with Array Values Filled In

The procedures for filling in Get/Pay expressions with array elements using the Fill Down button are very similar for arrays with other dimensionality. For example:

- Suppose you have defined a 5x1 column vector (array) called Invest\_5x1. Just as with the row vector, you would select Invest\_5x1 from the variables list and enter it for Attribute2. The Fill Down button () will then fill in the other four attributes with elements of the column vector, which are denoted Invest\_5x1[0][0], Invest\_5x1[1][0], etc.
- Suppose you have defined a 2x5 (two-dimensional) array called Invest\_2x5. As with the one-dimensional arrays, you would select Invest\_2x5 for Attribute2, and use the Fill Down button. In this case DPL will ask you whether to fill in with the first row or the first column of the array.
- You can also start filling in from an element of an array other than the first element. To do this, enter, for example, Invest\_1x10[5] for the first attribute you wish to fill in, and again use the Fill Down button to fill in the subsequent array elements. In this example, DPL would fill in array elements 5 through 9 for the five attributes.
- The Fill Down button works with series value nodes as well as array value nodes.

You may wish to experiment on your own to fully see how the Fill Down button in Get/Pay expressions works.



# 11. Using the Endpoint Database™

When DPL runs a Decision Analysis, it creates a variety of results and charts that help you understand, interpret, and display insights from your model. Underlying these results is an Endpoint Database, which DPL can also store (or "record") from a model run. An Endpoint Database can be envisioned as a simple tabular data set in which the records (or rows) correspond to endpoints (single paths) in the Decision Tree, and the fields (or columns) correspond to information about each endpoint, such as the state of each event for that path, the values of the objective function and each attribute for that path, and so forth. You can record, view, sort, and re-use an Endpoint Database. Endpoint Databases can also be imported from or exported to other applications.

At times it can be advantageous to re-use an Endpoint Database for further analysis; this is called playing endpoints. In order to play endpoints, you must select Full Tree Enumeration from Endpoints for your evaluation method. There are a number of changes you can make to a model after recording endpoints that will still allow you to play the endpoints to produce additional results. For example, you can reorder the Decision Tree to see the value of information for one or more chance nodes. The ability to record and play endpoints can save you a lot of analysis time, especially with DPL models linked to large spreadsheets.

The first two sections in this chapter provide an overview of the Endpoint Database and demonstrate how to play recorded endpoints. The last three sections describe how to reconnect, export, and import an Endpoint Database.

A model with a large number of endpoints will result in a large Endpoint Database. The displaying of a sizeable Endpoint Database in a report at the completion of a run consumes additional memory and may take some time. In addition, saving DPL Workspace files with large Endpoint Databases in them (particularly those displayed in a report) require significant disk space to save.

If you try to record and save databases with tens of thousands of endpoints (or more), you will see DPL messages warning you about the amount of memory and disk space that will be required. Extremely large Endpoint Databases (i.e., with hundreds of thousands or millions of endpoints) may be infeasible or inadvisable to record and save depending on your computer.

Also please note that if you save your model while completing this tutorial, you may see that it will take a few minutes to save and it may result in a fairly large (20-30 megabyte) saved Workspace file. If you want to be sure not to save a file this large, you should select Home | Run | Clear Mem (or press F9) before saving the Workspace file. However, this will clear the Endpoint Database and you will have to re-run the model the next time you open the Workspace. Other tips for using DPL's Endpoint Database with large models are given later in this chapter.

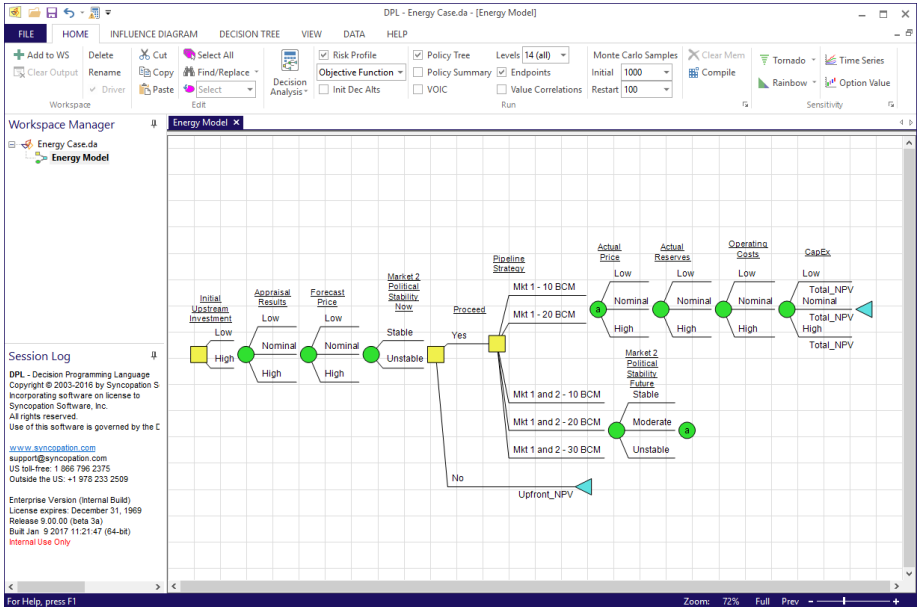
This tutorial assumes that you are familiar with the basics of using DPL Professional and with most of the material contained in the earlier tutorials of this manual.

## 11.1 Recording and Viewing Endpoints

---

For this tutorial, you will use the energy model similar to the one used in Chapters 9 and 10 of this manual. However, rather than continuing with the multi-attribute version of that model, you will use a version in which the objective function, NPV, is the only attribute.

- ⇒ Select File | Open.
- ⇒ Open the workspace Energy Case.da. This file is found in the Examples folder where DPL was installed, usually  
C:\Program Files\Syncopation\DPL9\Examples. See Figure 11-1.



**Figure 11-1. Energy Case Model Workspace**

In order to record endpoints, DPL uses the Full Tree Enumeration evaluation method. If you check the Endpoints box within the Home | Run group, DPL will automatically set the evaluation method to Full Tree Enumeration.

- ⇒ Within the Home | Run group, check the checkbox next to Endpoints if it is not already.
- ⇒ Request a Risk Profile and a Policy Tree as well.
- ⇒ Leave everything else including Init Dec Alts unchecked.
- ⇒ Click Home | Run | Decision Analysis or press F10 to run a Decision Analysis.

DPL will open the linked spreadsheet if it is not already and run the model. It may take a few minutes, as this model has more than 32,000 endpoints and Excel has to recalculate results each time. The expected value of this model is 34.9, as you can see in the Session Log or by zooming in on the Policy Tree.

When the model run is complete, you will see an icon for an Endpoint Database in the Workspace Manager.

⇒ Double-click on this icon to display the Endpoint Database. Note that the Grid tab appears and becomes active in the command ribbon under a green shaded context tab indicator. See Figure 11-2.

The screenshot shows the 'Endpoint Database' grid in the Syncopation software. The grid has the following columns: 0 (Endpoint), 1 (Proceed), 2 (Initial\_Upstr...), 3 (Pipeline\_Str...), 4 (Include\_Ma...), 5 (Pipeline\_Ca...), 6 (CapEx), 7 (Actual\_Price), and 8 (Actual\_Res...). The rows are numbered 0 to 25. The first row (0) is highlighted in pink. The 'GRID' tab is active in the ribbon.

0	1	2	3	4	5	6	7	8
Endpoint	Proceed	Initial_Upstr...	Pipeline_Str...	Include_Ma...	Pipeline_Ca...	CapEx	Actual_Price	Actual_Res...
0	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
1	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
2	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
3	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
4	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
5	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
6	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
7	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
8	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
9	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
10	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
11	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
12	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
13	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
14	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
15	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
16	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
17	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
18	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
19	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
20	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
21	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
22	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
23	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
24	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low
25	Yes	Low	Mkt_1__10...	No	BCM_10	Low	Low	Low

Figure 11-2. Endpoint Database™

The first column in the Endpoint Database indicates the endpoint number. Following that the Endpoint Database has a column for each event (decision, chance or controlled node) in the model. In a multi-attribute model, following the event columns there will be a column for each attribute. In a single attribute model, these columns do not appear. The next to last column is the probability column and the final column is the objective function column. The columns display the following for each of the endpoints of the Decision Tree:

- Event columns: the state of each decision node, chance node and controlled node in the model (the column labels are the event name);
- Attribute columns: the value of each attribute at the endpoint (the column labels are the attribute names);

- Probability column: the probability of the endpoint (i.e., the probability of the path in the Decision Tree represented by the endpoint);
- Objective function column: the objective function value at the endpoint.

As you examine the Endpoint Database, you will notice that there are some blank cells in the event columns. Blank cells may occur if there is asymmetry in the tree. Blank cells in a column indicate that the state of the event heading the column is not defined, i.e., the event does not occur on that path. For example, the first few thousand rows in the Endpoint Database correspond to paths where the state of the chance event "Market 2 Political Stability Future" is not defined because this chance event does not occur for these paths. The alternatives for Pipeline Strategy are either Mkt 1 – 10 BCM or Mkt 1 – 20 BCM for these first few thousand endpoints. Paths through the tree with these two alternatives do not include the Market 2 Political Stability Future chance event.

You can navigate around the Endpoint Database report as you would an Excel spreadsheet.

- ⇒ Press Ctrl+End to scroll down to the bottom, rightmost cell. You can see that there are 32,112 endpoints in the database.
- ⇒ Press Ctrl+Home to go to the top, leftmost cell.

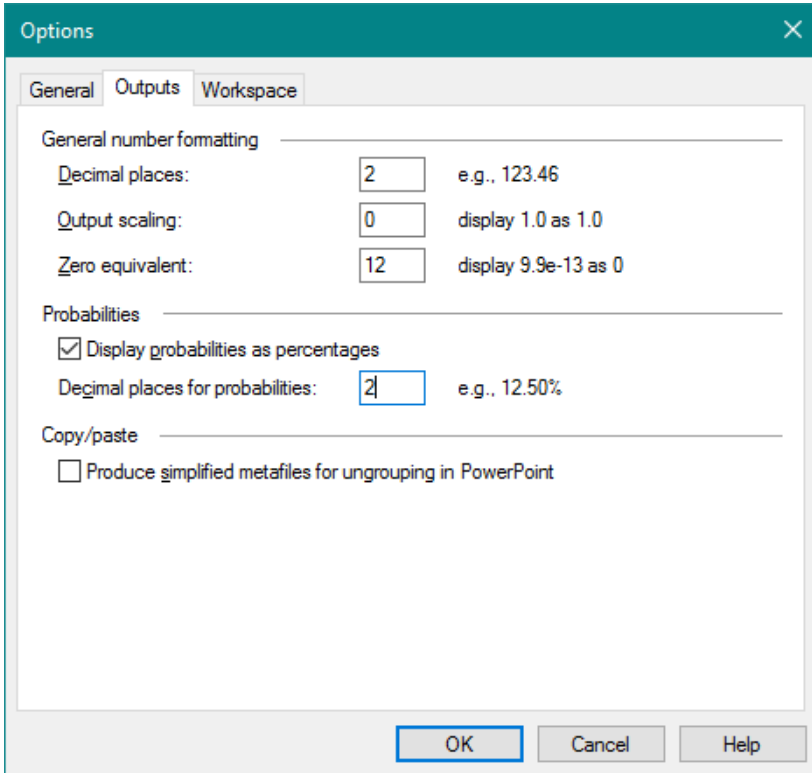
Note that by default, the Endpoint column (or column 0) is frozen so that when you scroll to the right the Endpoint column remains visible. You can freeze multiple columns or no columns by entering the number of columns you'd like to freeze in the Freeze column dialog (Grid | Format | Freeze). For example, if you enter "3" within the dialog and then scroll to the right (you may need to make the application window smaller to scroll), columns 0, 1, and 2 will remain visible.

You can drag the vertical lines separating the columns (in the first row) to change their width. You can also click Grid | Format | Columns | Width to specify the width for a particular the column. If you'd like to specify a width to be applied to all columns in the database check the checkbox next to Apply to all in the Column Width dialog.

- ⇒ Select Grid | Format | Row | Hide/Unhide or Grid | Format | Column | Hide/Unhide and experiment with hiding and unhiding a row or column.

Depending on how your display options are set, you may notice that most of the probabilities appear as 0%. The objective function value also may have fewer or more decimal places than you need to see.

- ⇒ Go to File | Options and select the Outputs tab.
- ⇒ Change the decimal places for both General number formatting and Probabilities to 2. See Figure 11-3.



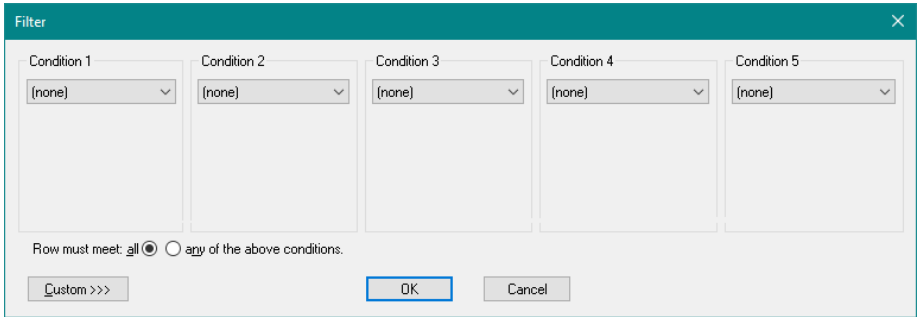
**Figure 11-3. Outputs Tab of the File Options Dialog**

- ⇒ Click OK. The Endpoint Database report updates to reflect the change.

### 11.1.1 Filtering the Endpoint Database™

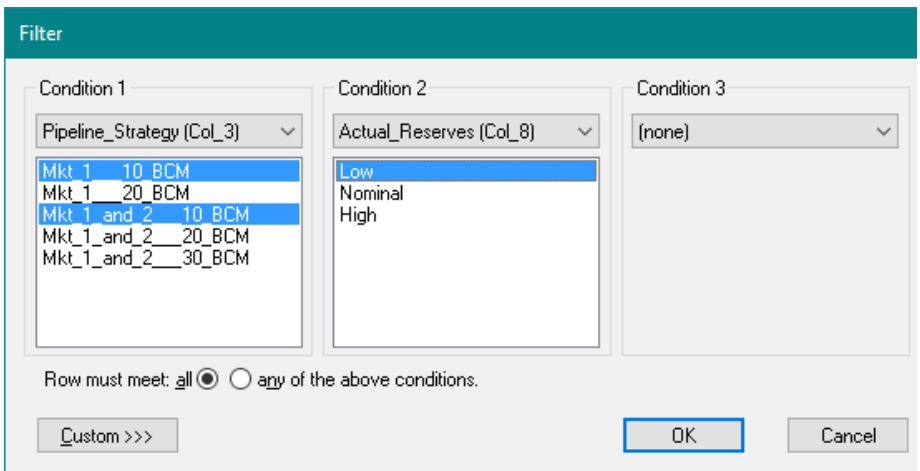
You can filter the Endpoint Database.

- ⇒ Click Grid | Sort & Filter | Filter. The Filter dialog comes up as shown in Figure 11-4.



**Figure 11-4. Filter Dialog for Endpoint Database™**

- ⇒ For Condition 1, use the drop-down list to select Pipeline Strategy (Col 3).
- ⇒ Select both of the strategies that include "10\_BCM" (hold down the Ctrl key and click on the first and third strategy states).
- ⇒ For Condition 2, select Actual Reserves (Col\_8) and select the Low outcome.
- ⇒ Leave the radio button for *Row must meet:* to "all". The left half of the dialog should look like Figure 11-5.



**Figure 11-5. Completed Filter Dialog**

- ⇒ Click OK. DPL displays the filtered Endpoint Database.

The endpoints shown in the Endpoint Database are filtered to the endpoints for which the Pipeline\_Strategy is in either the Mkt\_1\_\_10\_BCM

or `Mkt_1_and_2_10_BCM` alternatives and for which the outcome of `Actual_Reserves` is Low. Notice that the Grid | Sort & Filter | Filter button is shaded blue, indicating that the inputs have a filter applied.

Using the filter dialog, you can combine up to five filter conditions. In the example above, you defined a filter in which the endpoints must meet all of the conditions, e.g., a logical AND between the conditions. You may also define a filter in which the endpoints meet any of the conditions using the radio button below the conditions, i.e. a logical OR is used between conditions in the filter.

You can also create a Custom filter by clicking the Custom >>> expand button as shown in the filter dialog. Custom filters allow you to define more complex filtering rules so that you can view more specific subsets of the Endpoint Database. To see how to do this, you will modify the filter you just created to become a custom filter.

⇒ Click Grid | Sort & Filter | Filter.

⇒ Press the Custom button.

If you have defined a filter using the controls at the top of the dialog as you have just done, DPL will translate that filter into a custom filter and show the logic in the Custom filter edit box as shown in Figure 11-6.

The screenshot shows a dialog box titled "Filter" with a teal header. It contains three dropdown menus labeled "Add", "Condition 2", and "Condition 3", each with "(none)" selected. Below these is a radio button group: "Row must meet: all" (selected) and "any of the above conditions". A "Custom <<<" button is highlighted with a dashed border. Below it is a text area labeled "Custom filter" containing the logic: `((Col_3 == 0 || Col_3 == 2) && (Col_8 == 0))`. At the bottom right are "OK" and "Cancel" buttons.

**Figure 11-6. Custom Filter Logic**



In custom filters, the columns (i.e. DPL events, attributes, etc.) are denoted Col\_0, Col\_1, etc., according to their order in the report. The states of each column are integers numbered starting at zero. The filter operators are as shown in Table 11-1. In particular, the filter you created uses the operators "==" (equal to), "||" (logical OR) and "&&" (logical AND).

Filter Operator	Meaning
&&	logical AND
	logical OR
==	Equal to
!=	Not equal to
<, >	Less than, Greater than
<=, >=	Less than or equal to, Greater than or equal to
!	logical NOT

**Table 11-1. Filter Operator Definitions**

- ⇒ Place your cursor in the Custom filter edit box so that it is between the last two right parentheses.
- ⇒ Type " && (".
- ⇒ Use the drop-down list below the *Add* label to add Actual\_Price (Col\_7) to the filter.
- ⇒ Place your cursor after the Col\_7 in the Custom filter edit box and type " < 2)".
- ⇒ The filter should now read:

```
((Col_3 == 0 || Col_3 == 2) && (Col_8 == 0) && (Col_7 < 2))
```

You have added an AND condition such that Col\_7 is less than 2. This last condition corresponds to Actual\_Price being Low (0) or Nominal (1). Note: you could have created this filter without using custom logic.

- ⇒ Click OK.

DPL refreshes the Endpoint Database, restricting it further to endpoints in which Actual Price is Low or Nominal.

Custom filters can be used to "drill down" further in the Endpoint Database, letting you see narrowly defined sets of endpoints; you can combine AND with OR conditions and you can include more conditions than the five provided in the dialog.

⇒ Click Grid | Sort & Filter | Clear to clear the filter and display the entire Endpoint Database.

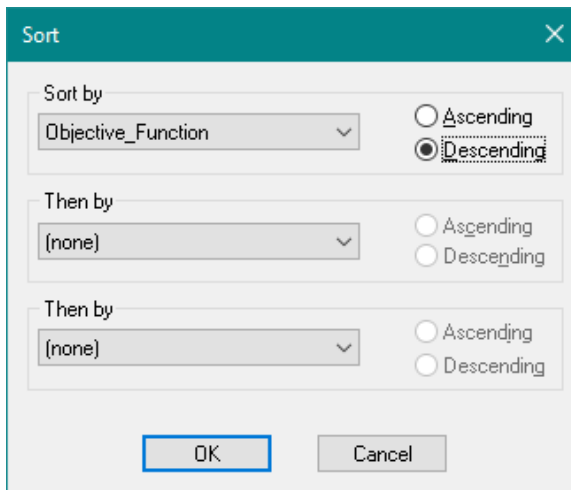
### 11.1.2 Sorting the Endpoint Database™

You can sort the Endpoint Database via the Sort dialog or directly from the ribbon. Within the Sort dialog, you may sort using up to three columns. Among many other things, this feature is useful for finding out the maximum and minimum values of your objective function and which paths lead to these best case or worst case outcomes.

⇒ Select Grid | Sort & Filter | Sort.

⇒ In the *Sort by* drop-down list, select Objective Function.

⇒ Click the Descending radio button. See Figure 11-7.



**Figure 11-7. Sorting the Endpoint Database™**

⇒ Click OK.

Note to sort by a single column it is usually quicker to simply click the top-most numbered row of the column you'd like to sort by. The cell will display an arrow next to the column number that indicates sort order, descending or ascending. You can also select a cell within the desired column and click the Ascending (A↓) or Descending (Z↓) buttons within the Grid | Sort &

Filter group for the same purpose. The Sort dialog is useful when you'd like to sort by more than one column.

If it isn't visible, scroll to the right of the report. You can see that the maximum value of the objective function is 338.36; this occurs when all outcomes are favorable (actual prices are High, operating costs are Low, and so forth).

⇒ Select Grid | Sort & Filter | Sort again and sort by (none) to return the Endpoint Database to its default display.

Endpoints in the Endpoint Database are sorted by default in the order in which DPL recorded them. You can also click in the top row of column 0 of the report to return the sort order to the default recorded order.

## 11.2 Playing Endpoints

---

If you are running a DPL model that is relatively large and is linked to an Excel spreadsheet that is also relatively large, you can save time by recording and playing an Endpoint Database. Playing endpoints works whenever DPL can re-use the Endpoint Database to get the results you need. The Endpoint Database stores all the value model information that is needed to roll-back the Decision Tree. Typically, calculating this value model information (particularly if this information comes from an Excel spreadsheet that needs to be re-calculated) is the most time consuming part of a DPL run. When you play endpoints, DPL only needs to look up values rather than recalculate them which results in much quicker runtime.

If you change your value model (i.e., add a value node which contributes to an attribute or the objective function) after you record endpoints, you will not be able to play endpoints.

If you have not changed your value model, playing endpoints is usually possible. The exception is when you have changed the structure of the model such that there are "new" endpoints. This can occur if you record endpoints for an asymmetric tree and then reorder nodes.

Reordering nodes in a symmetric tree, changing probabilities, adding conditioning events, branch controlling, and branch blocking do not result

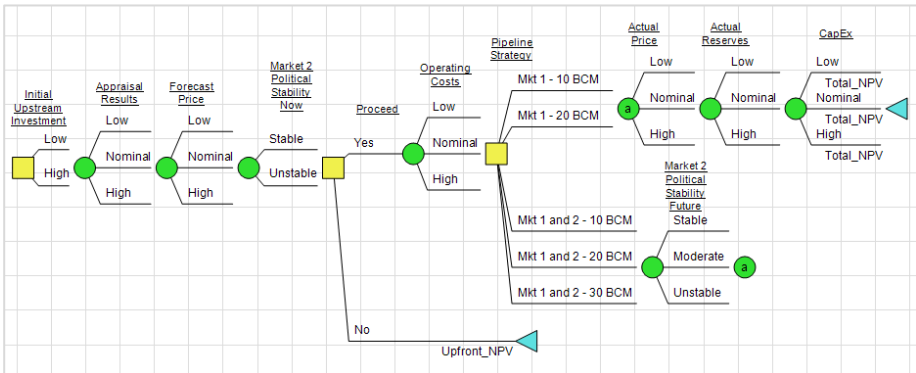
in new endpoints. Depending upon how you reorder, you may also be able to reorder nodes in an asymmetric tree and play the endpoints. Therefore, you can often make significant changes to your model and still be able to play a recorded Endpoint Database. This section demonstrates how you can play endpoints to save time when you need to do "what-if" analyses or quickly test changes to your model.

Before beginning this section, you should have an Endpoint Database recorded and available from the previous section.

### 11.2.1 Reordering Nodes

First you will reorder the tree so that Operating Costs are known before the Pipeline Strategy must be decided.

- ⇒ Switch back to the Model window (Ctrl + F12 or double-click on the Energy Model item in the Workspace Manager).
- ⇒ Click on the Operating Costs chance node in the Decision Tree.
- ⇒ Drag the node so that the chance node is positioned over the Pipeline Strategy decision node. Notice that the cursor changes to a reorder cursor when you hover over another node.
- ⇒ Release the mouse button to place the node in front of the Pipeline Strategy decision. The reordered tree should look like Figure 11-8.



**Figure 11-8. Reordered Decision Tree**

- ⇒ In the Home | Run group, select Risk Profile and Policy Tree. You can de-select Endpoints.

- ⇒ Drop-down the Home | Run | Decision Analysis split button and select Full Tree Enumeration from Endpoints from the list (or press the keyboard shortcut for Playing Endpoints: Alt+F10).
- ⇒ Click Yes to the warning.

DPL takes just a few seconds to play the endpoints and produce the results.

- ⇒ Examine the Policy Tree and/or the Session Log.

You can see that the expected value increased slightly, to about 35.2. If Operating Costs are known before the Pipeline Strategy decision has to be made, the gain in value is about 0.3 (35.2 - 34.9).

Note that you could have renamed and saved the original Policy Tree and Risk Profile. Since you did not rename them, they were over-written when you played endpoints.

- ⇒ Activate the Energy Model.

You will switch your model back to its original state. But before doing so take a moment to drop-down the Undo split button where you will find a list of the actions last executed. In this case you will see Undo Reorder Node as the last action taken. When there are multiple actions in the list you could choose to undo more than one of the previous actions in the list.

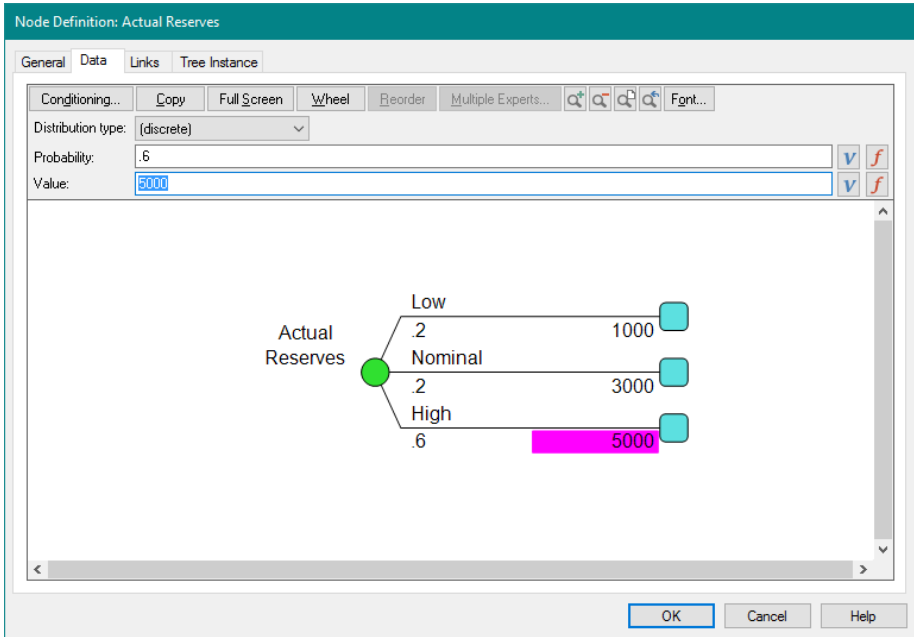
You can set the number of undo levels in File | Options | General. DPL allows you to set the number of Undo levels separately for graphics windows and for text windows (DPL Code).

- ⇒ Click the Undo button on the Quick Access menu located at the top left of the application window. This will undo the last action executed.

### 11.2.2 Changing Probability Data

You can change any of the probabilities in your model and play endpoints instead of re-running the entire model. You will change the probability distribution for Actual Reserves to be more optimistic.

- ⇒ Edit the probability data for Actual Reserves so that it looks like Figure 11-9. Note that only the probabilities, not the values, are changing.



**Figure 11-9. Revised Probabilities for Actual Reserves Node**

⇒ Click OK.

Note that although Full Tree Enumeration from Endpoints (or playing endpoints) was the last evaluation method used DPL does not update the Run | Decision Analysis split button. Therefore, each time you wish to run a Full Tree Enumeration from Endpoints, you must drop-down the Home | Run | Decision Analysis split button and select it from the list or press the keyboard shortcut Alt+F10.

⇒ Run a Full Tree Enumeration from Endpoints.

⇒ Click Yes to the warning.

The expected value has increased substantially to 46.8.

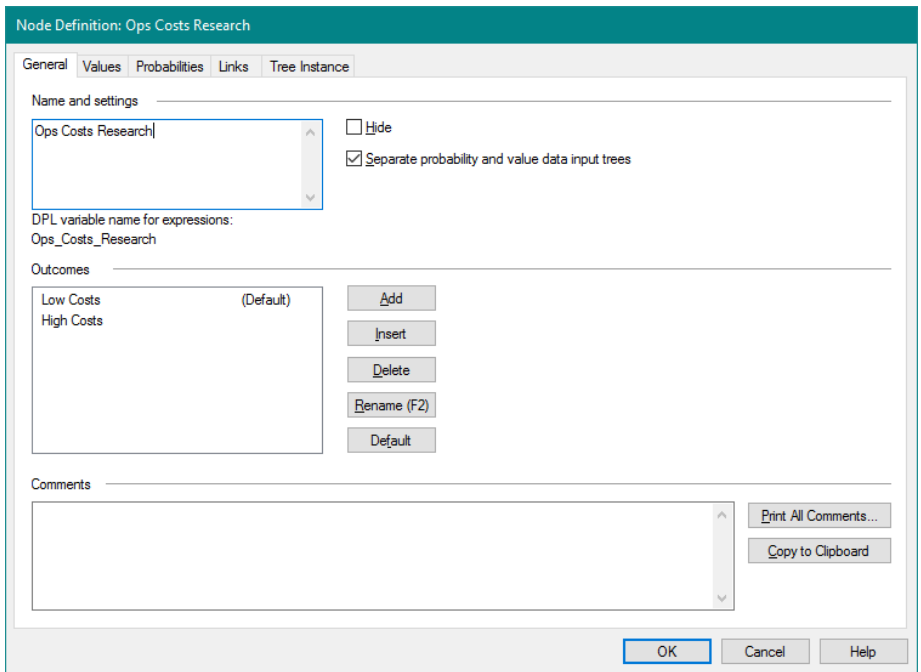
⇒ Switch back to the Model and select Undo from the Quick Access menu.

### 11.2.3 Adding Learning Events

You can add a chance event to the model and play endpoints as long as the new chance event only probabilistically conditions or is probabilistically

conditioned by other chance events in the model. If the conditioning is value-wise, you will not be able to play endpoints.

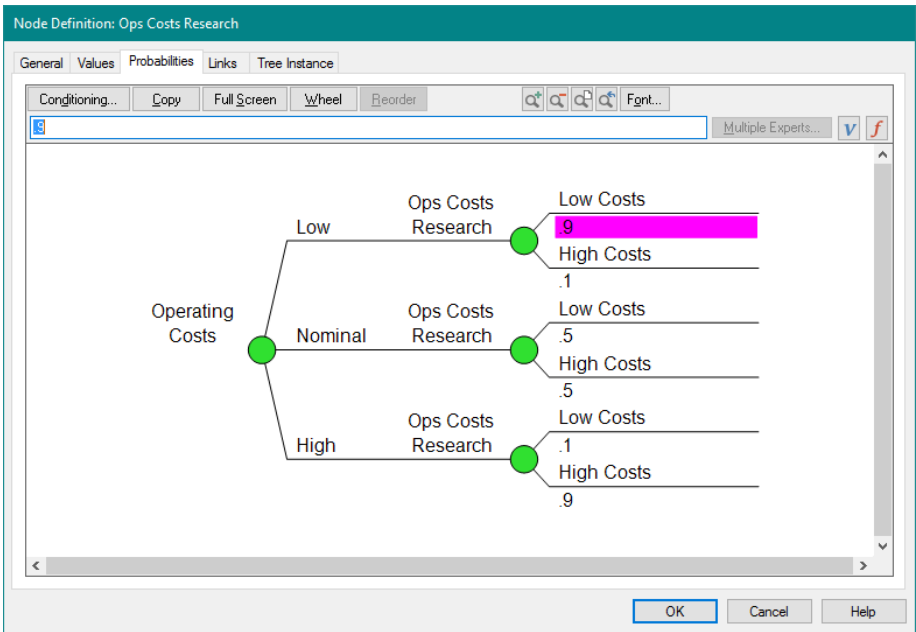
- ⇒ Within the Decision Tree add a new chance node to the model and place it just before the Proceed Decision.
- ⇒ Name the new node Op Costs Research.
- ⇒ Within the Outcomes box specify two: Low Costs and High Costs. Set Low Costs to be default.
- ⇒ Check the Separate probability and value data input trees – as only the probabilities will be conditioned. The General tab show now look like Figure 11-10.



**Figure 11-10. Node Definition, General Tab for Ops Costs Research**

- ⇒ Switch to the Probabilities tab.
- ⇒ Click the Conditioning button.
- ⇒ Select Operating Costs.
- ⇒ Click OK.

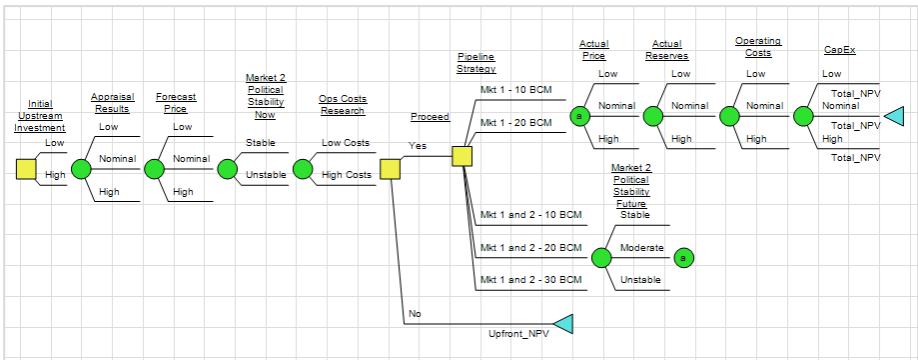
⇒ Edit the probability data for Op Costs Research so that it looks like Figure 11-11.



**Figure 11-11. Probability Data for Op Costs Research Node**

⇒ Click OK.

The edited Decision Tree should look like Figure 11-12.



**Figure 11-12. Decision Tree with Reordered Op Costs Research Node**

You are ready to play endpoints with the conditioning node.



- ⇒ Drop-down the Home | Run | Decision Analysis split and select Full Tree Enumeration from Endpoints or press Alt+F10.

You will see a warning message. DPL is telling you that you have added a new event since the recording of the Endpoint Database. If the conditioning relationship between the new event and existing events were not purely a probabilistic one, you would not be able to play endpoints. However, the Op Costs Research node serves only one purpose which is to update the probabilities of the Operating Costs node located farther down the Decision Tree given the results of the research. In this situation, you can play endpoints.

- ⇒ Click Yes for the warning.

You can see from the Policy Tree or the Session Log that the expected value of the model is slightly higher than the original value (35.04 compared with 34.9). Doing research on operating costs before making the Proceed decision adds a small amount of value. The added value, 0.1, might be less than the cost of the research in this example. This exercise gives you an upper bound on how much you would be willing to spend to gain the quality of information as specified by the Op Costs Research node.

Before continuing, you will remove the new node.

- ⇒ Activate the Energy Case model.
- ⇒ Click the Undo button to remove the Ops Costs Research node from the model.

#### 11.2.4 Saving Workspace Files with Endpoint Databases™


As mentioned earlier, Endpoint Databases can require a large amount of memory and disk space when a Workspace file is saved with an Endpoint Database in it. The display of an Endpoint Database also contributes a fair amount to the storage requirements.

If you need to save a Workspace file which contains a sizeable Endpoint Database, you can reduce the storage requirements by saving the Workspace file that contains an Endpoint Database but without the report.

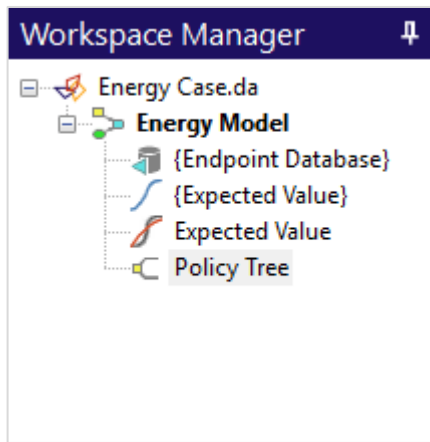
You may remove the report by doing the following.

Right-click on the Endpoint Database item in the Workspace Manager and select Delete. (You are not really deleting the Endpoint Database itself; you are merely deleting the report view of it.)

Alternatively, you can click on the "✕" button within its document tab in the Document Navigator. If the report window is not maximized, you can

click on the close button (  ) at the top right of the window or icon. A dialog will appear confirming that you would like to delete the Endpoint Database. Again, you are confirming deletion of the report view, not the Endpoint Database itself.

Your Endpoint Database is still stored but the view is not. The icon for the item in the Workspace Manager will change to a blue triangle with a database can behind it instead of a blue triangle with a grid behind it as shown in Figure 11-13. Also, DPL will change the caption to be "{Endpoint Database}" to indicate that the database is no longer displayed in a window. This is analogous to the way risk profile datasets (not charts) are shown.



**Figure 11-13. Endpoint Database™ Without Grid View in Workspace Manager**

You can restore the view of the Endpoint Database by double-clicking on the icon in the Workspace Manager. You can also play endpoints without them being displayed in a report as you will see in the next section.

If you select the Endpoint Database item at this point and press Delete, then the Endpoint Database itself will be deleted.

### 11.2.5 Risk Profiles

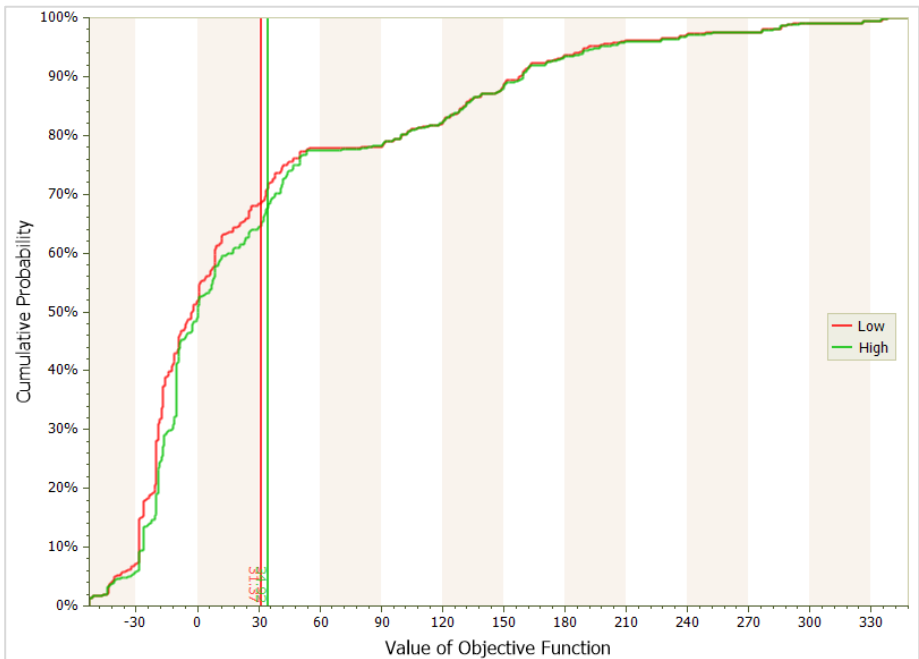
On occasion, you will want to play endpoints for the purpose of generating output charts that you did not generate with the initial decision analysis run. You can generate four types of outputs by simply playing the endpoints: Risk Profiles, Policy Trees, Policy Summaries, and Expected Value of Perfect Information and Control charts.

If your model has multiple attributes, you may want to play endpoints in order to generate Risk Profiles for attributes other than the objective function. See Chapters 8 through 10 of this manual for more information on models with multiple attributes.

You can also play endpoints to generate an Initial Decision Alternatives chart. You will do this now.

- ⇒ Make sure Risk Profile and Init Dec Alts is checked.
- ⇒ Also check Policy Tree and Policy Summary (for comparison with the results of the next sub-section of this tutorial).
- ⇒ Run a Full Tree Enumeration from Endpoints. Click Yes to the prompt.

DPL quickly creates an Initial Decision Alternatives chart for the energy case, as shown in Figure 11-14. You can see from this Risk Profile that the two initial decision alternatives have very similar risk profiles and that neither alternative is substantially riskier than the other. There is little reason for the decision maker not to choose the initial alternative with higher expected value, which is High Initial Upstream Investment.



**Figure 11-14. Initial Decision Alternatives Risk Profile from Playing Endpoints**

### 11.2.6 Branch Controlling and Blocking

You can very quickly generate what-if analyses by controlling the outcome of an event to a particular state, or blocking a specific branch of a decision, and playing endpoints.

You should have a Policy Tree and a Policy Summary from the previous endpoint play, with an expected value of 34.9.

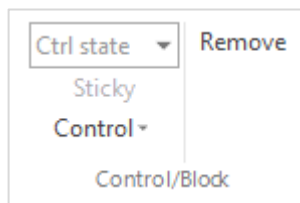
- ⇒ Right-click on the Policy Tree and rename it to Original Policy Tree.
- ⇒ Right-click on the Policy Summary and rename it to Original Policy Summary.

You have just saved the original results for comparison with your what-if analysis. You will evaluate how the model results change when two of the chance nodes are set to their worst-case outcomes.

You can control branches directly on the ribbon via the Decision Tree | Control/Block group as seen in Figure 11-15. You can also employ branch control via the Control tab of the Branch Definition dialog. To access the latter, right-click the branches of a node in the Decision Tree and select Control/Block... from the context menu. For the purposes of this tutorial you will use the Command Ribbon. See Table 1-5 for explanation of the commands found within the Decision Tree | Control/Block ribbon group.

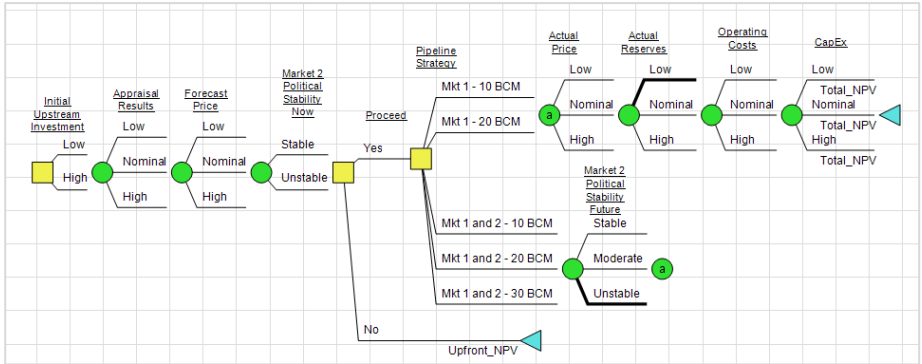
In the Decision Tree, branch control the Actual Reserves chance node to the Low state.

- ⇒ Select the branches of the Actual Reserves node in the Decision Tree and select Low from the Control drop-down list within Decision Tree | Control/Block group (See Figure 11-15).



**Figure 11-15. Decision Tree | Control/Block Ribbon Group**

- ⇒ Also branch control the Market 2 Political Stability Future node to Unstable. Your Decision Tree should look like Figure 11-16.



**Figure 11-16. Decision Tree with Branches Controlled for Two Chance Nodes**

- ⇒ Within the Home | Run group uncheck Risk Profile.
- ⇒ Make sure Policy Tree and Policy Summary are checked.
- ⇒ Run a Full Tree Enumeration from Endpoints. Click Yes to the prompt.

You can see from the Policy Tree that the expected value of the decision has declined to only about 4.5. You can compare the what-if Policy Summary with the one you saved (Original), and see that the optimal initial decision alternative has changed to the Low level of investment, and that the policy dependent probabilities of the downstream strategy have changed significantly as well. For example, the No branch of the Proceed decision is chosen much more often.

- ⇒ Activate the model and select the branches for both Actual Reserves and Market 2 Political Stability Future by using the Ctrl key.
- ⇒ Select Decision Tree | Control/Block | Remove to return the model to its earlier state.
- ⇒ Delete the Original Policy Tree by right-clicking its item in the Workspace Manager and selecting Delete.
- ⇒ Delete the Original Policy Summary.

## 11.3 Reconnecting Endpoints to a Model

In some situations, you may find it useful to rename and save an Endpoint Database, just as you can rename and save other model outputs in DPL. When you save an Endpoint Database, you can later reconnect your model to the saved database and play its endpoints. You might need to do this, for example, if you make changes to your value model (e.g., in your linked Excel spreadsheet) but then wish to revert to the original version to compare results. You will do a simple example of this now.

- ⇒ Double-click on the Endpoint Database icon in the Workspace Manager.
- ⇒ Click Yes to the prompt in order to display it in a grid.
- ⇒ Right-click on the Endpoint Database item in the Workspace Manager and Rename it to Saved Endpoints.

DPL makes a copy of the Endpoint Database under the new name. See Figure 11-17.

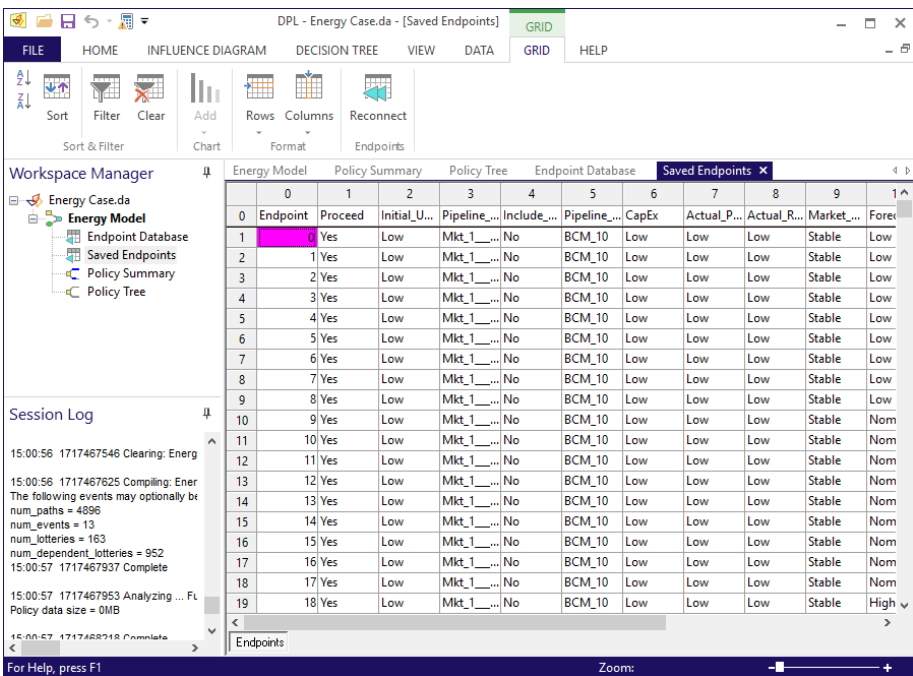


Figure 11-17. Workspace Manager with Saved Endpoint Database™

At this point, the renamed Endpoint Database (Saved Endpoints) is not connected to your model; it is essentially a backup copy of your Endpoint Database. Note: because it is not connected to your model, the Endpoint Database displayed in the report Saved Endpoints does not exist anywhere except in the report. If you delete an Endpoint Database that is not connected to a model, the report and the Endpoint Database it displays will be deleted. I.e., it is not possible to have a disconnected Endpoint Database that is not displayed in a report.

The original Endpoint Database (still named Endpoint Database) is currently connected to your model. You will now change your value model and re-run, over-writing this database.

- ⇒ Switch to Excel to edit the Energy Case.xlsx spreadsheet.
- ⇒ In the Assumptions tab, change both the Annual Price Growth assumptions (cells C24 and C25) to 5%.
- ⇒ Also, change the Annual Op Costs Growth assumptions (cells C26 and C27) so that they are both 2%.
- ⇒ Change the value of Initial Marketing Spend (cell C33) to 650.

The "Other Assumptions" section of the spreadsheet should look like Figure 11-18.

Other Assumptions	
Annual Price Growth Market 1	5%
Annual Price Growth Market 2	5%
Annual Op Costs Growth Market 1	2%
Annual Op Costs Growth Market 2	2%
Start year	2017
value: p	0.55
value: q	0
Upfront Cost Escalation	1
Initial Marketing Spend	650
Results	
Upfront NPV	(\$11.64)
Total NPV	\$210.01

**Figure 11-18. New Assumptions in Energy Case.xlsx**

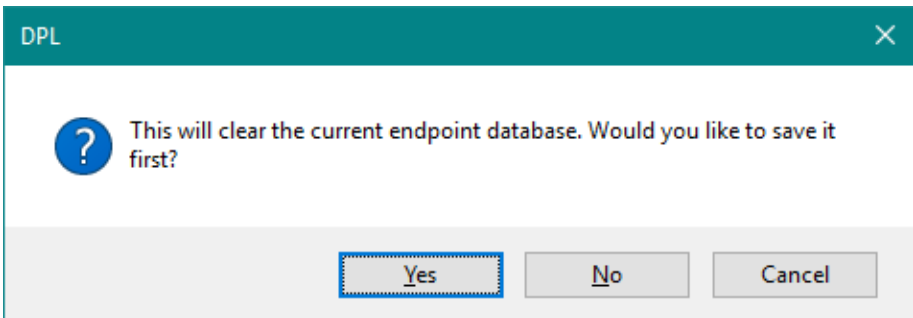
- ⇒ Switch back to DPL.

- ⇒ Make sure that Endpoints, Policy Tree, and Policy Summary are checked within the Home | Run group and run a Full Tree Enumeration. Click Yes to the prompt.

DPL runs your model with the new assumptions in Excel, and overwrites the Endpoint Database with a new one. As you can see from the Policy Tree, the model results are dramatically different. The expected value is now over 100. From the Policy Summary, you can tell that the optimal policy has changed as well. Most of the time, the optimal pipeline strategy is to pursue Market 1 only with a pipeline capacity of 20 BCM.

You learn the new assumptions are incorrect (they were too optimistic). You need to immediately produce a Value of Information and Control chart for the model using your original assumptions. Rather than going back to Excel, removing the changes, and re-running the model, you can generate this in a few clicks by simply reconnecting and playing the Endpoint Database you saved earlier.

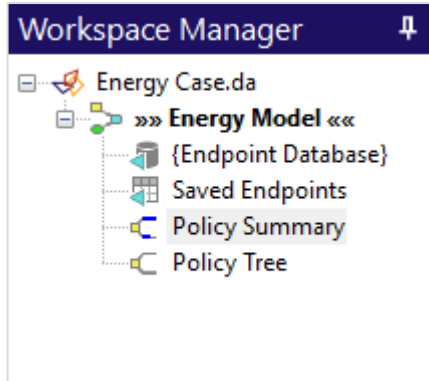
- ⇒ Activate (double-click on) the Saved Endpoints item in the Workspace Manager.
- ⇒ Click Grid | Endpoints | Reconnect.
- ⇒ DPL will ask if you wish to save the Endpoint Database currently connected to the model (the one you created with the optimistic assumptions), as shown in Figure 11-19. Click No.



**Figure 11-19. Save Endpoint Database™ Prompt**

DPL reconnects the Saved Endpoints database and gives it the default name, Endpoint Database, as shown in Figure 11-20. Note that the connected endpoints are not yet displayed in a report. Note also that Saved Endpoints is still saved as a separate item.





**Figure 11-20. Workspace Manager after Reconnecting Endpoint Database™**

- ⇒ Check Risk Profile, Init Dec Alts, Policy Tree, Policy Summary, and VOIC within the Home | Run group and run a Full Tree Enumeration from Endpoints to generate results from the saved and reconnected Endpoint Database:

DPL instantly creates new outputs from the original Endpoint Database. You can examine the Policy Tree or session log to see that the expected value is back to 34.9, as before. In addition, DPL has generated a new Value of Information and Control chart from the reconnected endpoints. Examine this chart if you like. Value of Information and Control charts are explained in the Section 3.5 of this manual.

You do not need to save your model for use in the last two sections of this chapter. If you do save the model in its current state, it will be a large file because there are two Endpoint Databases (one of which is displayed in a report) in the file.

- ⇒ Right-Click on the Saved Endpoints item and select Delete. The Endpoint Database in this report was not connected to a model so it will be deleted entirely.
- ⇒ Close the Excel file, Energy Case.xlsx, without saving changes.

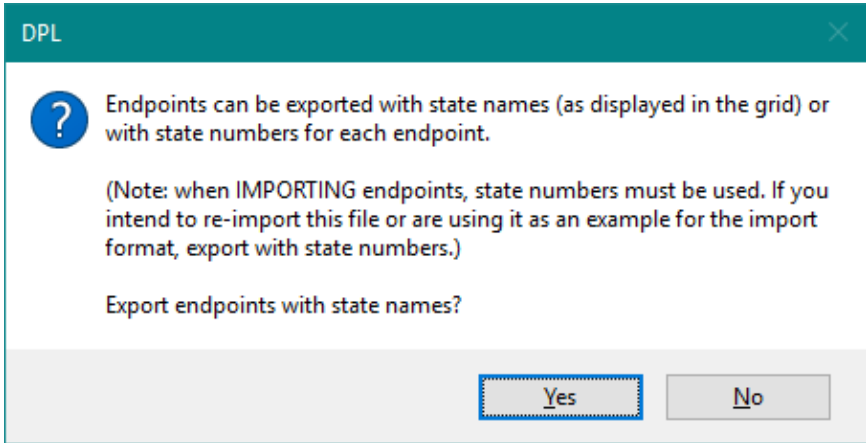
## 11.4 Exporting Endpoints

---

As you have seen, the Endpoint Database report looks a lot like an Excel spreadsheet. In fact, you can quickly and easily export the Endpoint Database to a comma separated value (.CSV) file which can be read by Excel or other applications.

There are two ways to export endpoints: with state names for each event (as you see in the report) or with state numbers. If you plan to use the .CSV file in Excel or another application where you need to see the state names, you should use the state names option. However, if you wish to import the .CSV file back into DPL later (as you will do in the next section), you need to choose the state numbers option. You will do both in this tutorial.

- ⇒ If you have the energy model open with a recorded Endpoint Database (from the previous sections), double-click on the Endpoint Database item to make sure the report is displayed. Click Yes if you are prompted to display the report.
- ⇒ If you do not have the energy model currently open or you do not have a recorded Endpoint Database, open it and/or run it and record endpoints as described in the beginning of this chapter.
- ⇒ Activate the Endpoint Database if it is not already active.
- ⇒ Go to the Data tab, drop down the Export list and select Data.
- ⇒ Choose a folder location and name the file Endpoints1.csv. Click Save.
- ⇒ You will be prompted whether to use state names or state numbers as shown in Figure 11-21.



**Figure 11-21. Saving Exported Endpoints**

⇒ Click Yes to use state names. DPL saves the file Endpoints1.csv.

You can open this file in Excel or in other applications and see how it has stored the endpoint data for you. Note: when opening in Excel you may need to change the file type drop down list to All Files to select the Endpoints1.csv file.

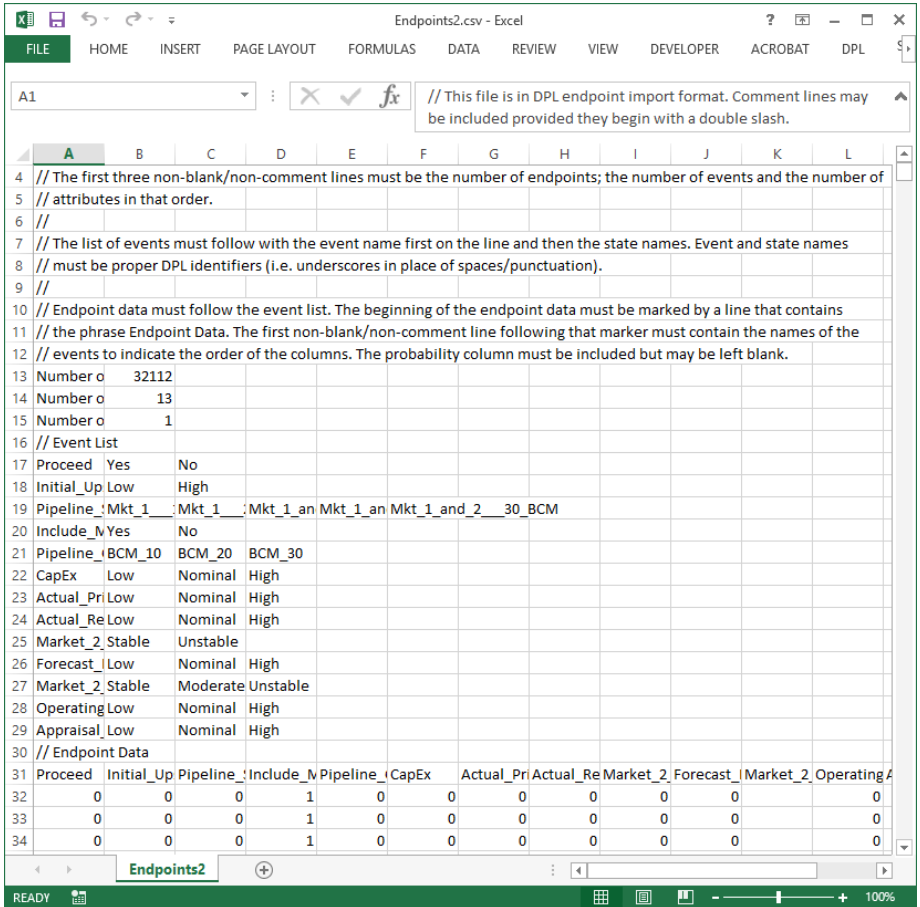
Figure 11-22 shows the rows at the top of the worksheet describing the number of endpoints and other information about the database, including the state names for each event. Following that is a row with column labels, and then the 32,112 rows of data that you have examined previously in DPL.

	A	B	C	D	E	F	G
1	Number of endpoints	32112					
2	Number of endpoints	13					
3	Number of active endpoints	1					
4	// Event Data						
5	Proceed	Yes	No				
6	Initial_Upstream	Low	High				
7	Pipeline_Strat	Mkt_1_10	Mkt_1_20	Mkt_1_and_2	Mkt_1_and_2	Mkt_1_and_2_30	BCM
8	Include_Mark	Yes	No				
9	Pipeline_Cap	BCM_10	BCM_20	BCM_30			
10	CapEx	Low	Nominal	High			
11	Actual_Price	Low	Nominal	High			
12	Actual_Reserve	Low	Nominal	High			
13	Market_2_Percent	Stable	Unstable				
14	Forecast_Prior	Low	Nominal	High			
15	Market_2_Percent	Stable	Moderate	Unstable			
16	Operating_Cost	Low	Nominal	High			
17	Appraisal_Rate	Low	Nominal	High			
18	Endpoint Data						
19	Proceed	Initial_Upstream	Pipeline_Strat	Include_Mark	Pipeline_Cap	CapEx	Actual_Price
20	Yes	Low	Mkt_1_10	No	BCM_10	Low	Low
21	Yes	Low	Mkt_1_10	No	BCM_10	Low	Low
22	Yes	Low	Mkt_1_10	No	BCM_10	Low	Low

**Figure 11-22. Endpoint Database™ in Excel with State Names**

Next, you will export the endpoint data again using state numbers instead of state names.

- ⇒ Switch back to DPL (if you were viewing the data in Excel).
- ⇒ Activate the Endpoint Database again, if needed.
- ⇒ Export the Endpoint Database and give the file the name Endpoints2.csv this time. Click Save.
- ⇒ Click No to the DPL prompt in order to save state numbers.
- ⇒ Examine the file in Excel. It should look like Figure 11-23.



**Figure 11-23. Endpoint Database™ in Excel with State Numbers**

Note that at the top of this file, there are several comment lines explaining the file format. This format is required in order for DPL to re-import the endpoint data. As you can see in Figure 11-23, the file (other than the comment lines) is very similar to the previous file you exported; however, instead of state names, the data (other than the last two columns, which are the probabilities and the objective function values) indicates the settings of each state using state numbers. The state numbers are indexed beginning with zero for each event.

If you should need to import endpoint data that was created by an application other than DPL (such as a statistical package or just within Excel), you will need to make sure that the data is in the correct format. The easiest way to do this is to export a database from DPL using state numbers as you just did, and then look at the format and the comment

lines at the top of the file. Also, if you intend to do any post processing of the exported Endpoint Database, it may be easier to do so with one that has been exported with state numbers.

In the next section, you will see how to import endpoints.

## 11.5 Importing Endpoints

---

There are two ways to import endpoints: from a DPL Workspace (.DA) file or from a .CSV file. In addition, you can import and merge multiple Endpoint Databases. This section contains three brief tutorials to demonstrate how importing works.

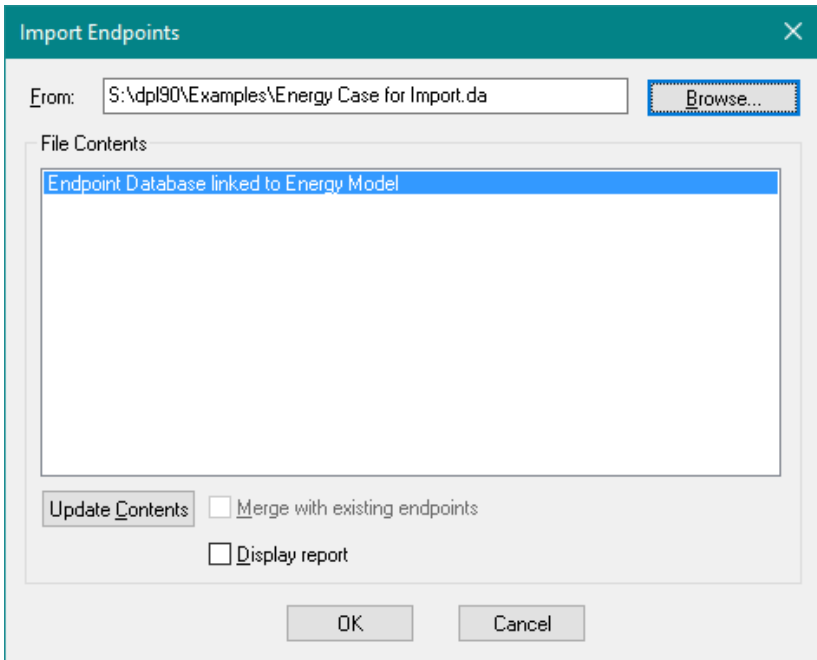
When you import an Endpoint Database, DPL does not display to you any endpoint probabilities contained in the database. These probabilities are not used when you play endpoints. Further, if the endpoints were generated in an application other than DPL, probability data will most likely be missing.

### 11.5.1 Importing from DPL™ Workspace Files

You can import an Endpoint Database from a previously saved DPL Workspace (.DA) file and connect it to your model. As you might expect, the Endpoint Database structure must match the structure of your model in order for DPL to be able to successfully import and play the imported endpoints. In this tutorial you will continue with the energy case model, but you will import an Endpoint Database containing different results.

- ⇒ You should still have the energy model open (if not, open Energy Case.da from the DPL examples folder).
- ⇒ Switch to the Energy Case model. You should have no Endpoint Database.
- ⇒ If needed, press F9 to clear model memory and click OK for the warning.
- ⇒ With the model still activated, select Home | Workspace | Add to WS | External Endpoints.
- ⇒ Browse to find the file: Energy Case for Import.da located within the Examples folder on your DPL installation directory. This DPL file contains an Endpoint Database that was previously created with the energy model.

- ⇒ Select the file and click Open. You will see the Import Endpoints dialog; see Figure 11-24.



**Figure 11-24. Importing Endpoints from a DPL Workspace File**

- ⇒ Leave Display report unchecked.
- ⇒ Click OK.

DPL imports the Endpoint Database. You can see in the Workspace Manager that Imported Endpoint Database is present, although the report is not shown.

- ⇒ Check Policy Tree and Policy Summary in the Home | Run group.
- ⇒ Run a Full Tree Enumeration from Endpoints.

The expected value of this model is approximately 50 and the policy is different from the energy model you have been using. Note that you played endpoints, instead of running the model. If you were to run the model at this point, you would revert to your earlier results (expected value of 34.9), because your model is still linked to the original Excel file.

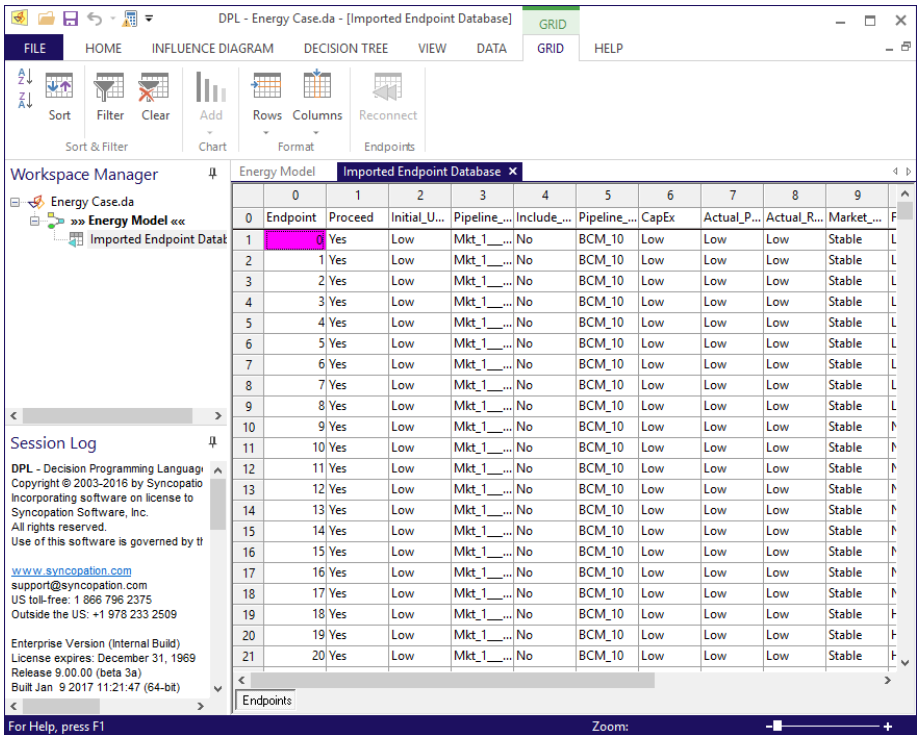
- ⇒ Before proceeding to the next tutorial, switch to the Energy Model and Home | Run | Clear Mem to clear model memory. Press Ok to the warning. The imported Endpoint Database is no longer available.

### 11.5.2 Importing Endpoints from .CSV Files

In this section you will see how to import the Endpoint Database from a .CSV file that was exported from DPL. The steps are simple.

- ⇒ With the Energy Model activated and model memory cleared, select Home | Workspace | Add to WS | External Endpoints.
- ⇒ Navigate to the Examples folder underneath the DPL installation directory.
- ⇒ Change the File Type drop-down list to Comma Separated Value Files (\*.CSV).
- ⇒ Select endpoints\_import.csv from the list. Click Open.
- ⇒ In the Import Endpoints dialog, check Display report so that you will be able to see the imported endpoints this time.
- ⇒ Click OK. DPL displays the endpoint report and calls it Imported Endpoint Database, as shown in Figure 11-25.





**Figure 11-25. Endpoint Database™ Imported from a .CSV File**

As mentioned above, DPL does not display any probabilities for imported endpoints. You can see this is the case in the endpoints you just imported.

- ⇒ Scroll to the right to see the probability column. It is blank.
- ⇒ Within the Home | Run group, check the Policy Tree box and then play the endpoints to check that the model results are as before, with an expected value of 34.9.
- ⇒ Before continuing to the last tutorial, switch to the Energy Model and press F9 to clear model memory again.

### 11.5.3 Merging Two Endpoint Databases™

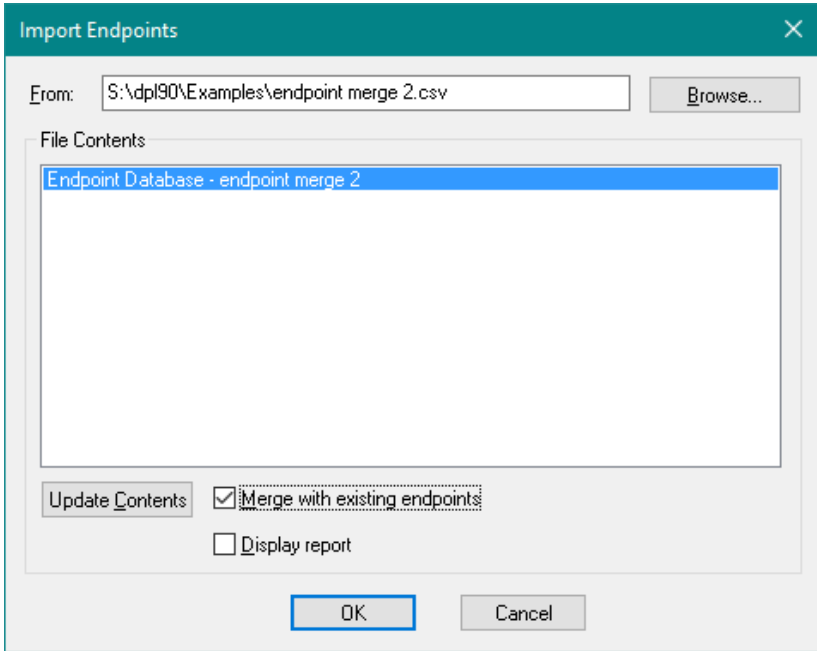
In this tutorial you will import and merge two separate Endpoint Databases from two different .CSV files. You might need to do this, for example, if you have created a model that includes an initial decision with two alternatives, and you have used other applications (such as statistical or database software) to calculate values for each alternative. As long as the Endpoint Databases are compatible with your DPL model structure and have the

required format (as documented in the first several lines of comments in the .CSV files), you can merge the databases in DPL and use them in conjunction with your model.

Note that you can also merge Endpoint Databases from two or more DPL Workspace files. You may wish to do this if you have a model with particularly long run times. You could put the model on several different computers and record endpoints with, for example, an upfront decision controlled to a different alternative on each computer. You could then quickly conduct a complete analysis after merging the separate Endpoint Databases.

In this example, you will merge two databases that were created using the energy model. Each database contains half (16,056) of the endpoints in the overall database, so each .CSV file contains a comment section and required formats at the top, followed by the column heading row and then 16,056 rows of data.

- ⇒ If you like, examine the files `endpoint merge 1.csv` and `endpoint merge 2.csv` which are in the DPL Examples folder.
- ⇒ Close the .CSV files.
- ⇒ Switch to DPL.
- ⇒ Make sure that the energy model is open and activated, and has no Endpoint Database in it. Press F9 to clear model memory if needed.
- ⇒ Select `Home | Workspace | Add to WS | External Endpoints`. Browse for the file `endpoint merge 1.csv` (you will have to use the Files of Type drop-down to look at files with the extension .csv, as you did earlier). Open the file.
- ⇒ In the Import Endpoints dialog, keep `Display report` unchecked. Select `Endpoint Database - endpoint merge 1` and click OK. DPL loads this database.
- ⇒ Select `Home | Workspace | Add to WS | External Endpoints` again. This time locate and open the file `endpoint merge 2.csv`.
- ⇒ In the Import Endpoints dialog, check `Merge with existing endpoints`, as shown in Figure 11-26. Click OK.



**Figure 11-26. Merging an Endpoint Database™**

DPL updates the Endpoint Database. Note that the Workspace Manager does not change. There is still one Imported Endpoint Database.

- ⇒ Double-click on the Imported Endpoint Database to create and display the report. Click Yes for the prompt.
- ⇒ Scroll to the bottom of the Endpoint Database (Ctrl + End). You can see that the Endpoint Database has 32,112 endpoints.
- ⇒ Play Endpoints to see results.

Your model has the full, merged Endpoint Database and results available. The expected value is still 34.9.

- ⇒ Close the Workspace without saving changes.

The tutorials in this chapter so far have provided you with the basic steps you need to record, manipulate, view, play, import, export and merge Endpoint Databases.

## 11.6 Recording Endpoint in Parallel using Multiple Servers (Enterprise only)

---

The Enterprise version of DPL includes a feature which allows you to record endpoints in parallel on several other computers, keeping your own computer free for other work and resulting in quicker runs.

To understand how this feature works, imagine that you have a model with 100,000 endpoints linked to a complex spreadsheet, and that your model takes an hour to run. Your office mate is out to lunch so you have two computers to work with. Using the techniques described earlier in this chapter, you could record the first 50,000 endpoints on one computer, the last 50,000 endpoints on the other computer, paste the two together, import them into DPL, and then run an instant decision analysis replaying those endpoints.

In theory, this approach might save you a half an hour, but given all the manual cut-and-paste work the wiser course might be to start a conventional run and go to lunch yourself. Fortunately, in DPL 9 Enterprise there is a feature which makes this process automatic and, with the use of several computers, can speed up runs substantially.

The parallel endpoint recording feature is recommended for models which take a long time to run because of a complex value model (typically a big Excel spreadsheet). If a model takes a long time to run because it has millions of endpoints, the overhead associated with transferring endpoints between computers will likely negate any benefit from performing calculations in parallel.

When recording endpoints in parallel, your computer is called the client and the other computers are called servers. To set up parallel endpoint recording, the following requirements must be satisfied:

- You must have at least two computers, your computer (the client) and one or more other computers (the servers). The servers should have similar specifications, since each will be assigned an equal number of endpoints and if one is slow it will become the bottleneck.
- All the computers must have read and write access to a mapped share drive, ideally via a fast connection on a LAN.
- If the DPL model is linked to a spreadsheet, all of the computers must have Microsoft Excel installed (need not be the same version).

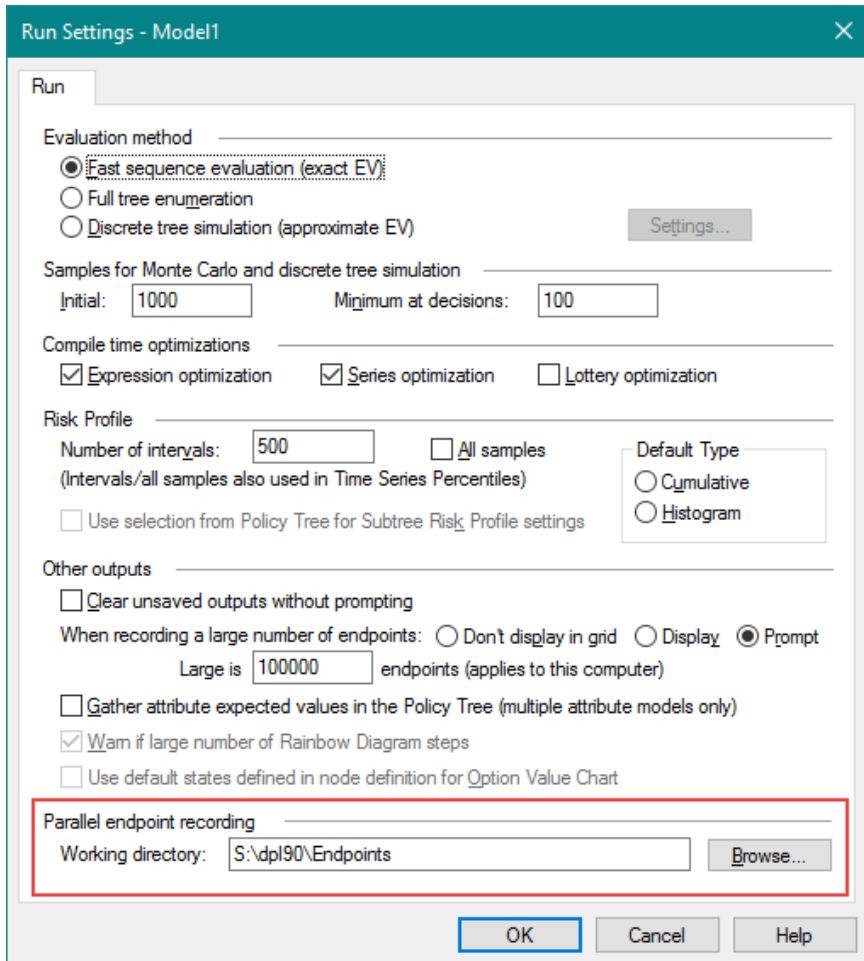
To begin your parallel endpoint recording session, perform these steps:

1. Make sure the DPL model and linked spreadsheet are stored in a folder on a network share drive. This should be a working folder for DPL runs only and should not contain any other files (DPL creates various files in this directory and assumes it can delete freely when cleaning up). It's recommended that you use a workspace with only one model.
2. On all of the computers, open DPL, select Home | Run | Settings (the dialog box launcher) to open the Run Settings dialog.
3. Near the bottom of the dialog, under the *Parallel endpoint recording* section, specify the working directory for parallel endpoint recording.
4. On the server computer(s), you'll give the command Home | Decision Analysis | Act as Endpoint Server. On the client computer, you'll give the command Home | Decision Analysis | Record Endpoints using Servers, kicking off the parallel recording of endpoints on the server machines.

When the parallel run completes, DPL will import and merge all of the endpoints on the client machine. You can then replay the endpoints using Home | Decision Analysis | Full Tree Enumeration from Endpoints to quickly obtain analysis results.

If you have multiple machines available and can satisfy all of the requirements stated above, you can proceed with the following tutorial in which you'll complete a parallel endpoint recording session for the Energy case model.

- ⇒ On the machine that will serve as the client. Make sure that the energy model is open and activated and has no Endpoint Database in it. Press F9 to clear model memory if needed.
- ⇒ Set up a folder on a network share drive as described above. Save the Energy Case workspace and spreadsheet to the new folder. Be sure the folder contains only the Energy Case.da workspace and the Energy Case.xlsx spreadsheet file.
- ⇒ Open DPL on the machine(s) that will act as the servers.
- ⇒ On each computer (servers and client), select Home | Run | Settings to open the Run Settings dialog. Within the *Parallel endpoint recording* section, browse for the working directory that contains the DPL model and linked spreadsheet as shown in Figure 11-27.



**Figure 11-27. Working Directory for Parallel Endpoint Recording specified within the Run Settings dialog**

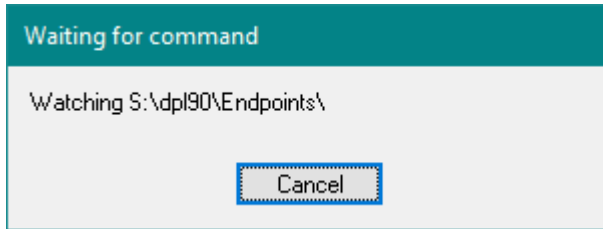
- ⇒ On each of the server machines, choose Home | Decision Analysis | Act as Endpoint Server.

This will close the current workspace on the server machines. You will receive a save prompt. The Working directory setting is not a Workspace level setting so you do not need to save the Workspace in order for the setting to be saved. The server machines should not have a workspace open during this process. In this example, DPL should be open with a blank Workspace on the server machine(s) so the Workspace does not need to be saved. If you had a Workspace open on a server machine as you started

this process, you could say Yes to the save prompt without disrupting the process. DPL will save the Workspace, close it and put the server machine in waiting mode.

⇒ Say No to the Save prompt.

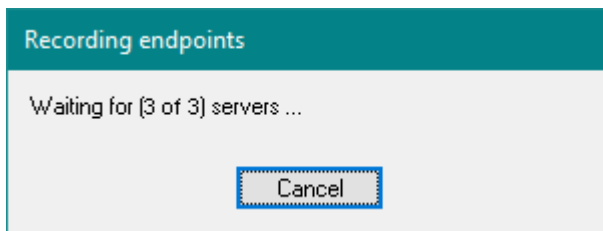
A dialog will launch that states "Watching <<your working directory>>" (Figure 11-28). Further, you will see that the Session Log has the text "waiting for command..." written to it.



**Figure 11-28. Waiting for command dialog on Server Machines**

⇒ On the client machine, choose Home | Decision Analysis | Record Endpoints using Servers. The model will compile on the client machine, so if the spreadsheet isn't already open, DPL will launch Excel and open the linked spreadsheet. The Servers will commence recording endpoints for the model.

The client machine will display a Recording Endpoints dialog similar to Figure 11-29 indicating the number of servers that the client machine is waiting for.

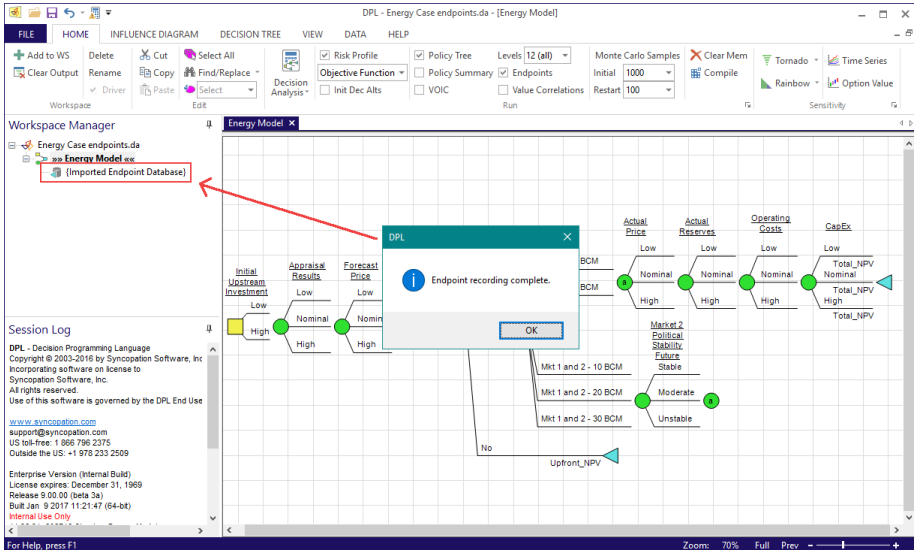


**Figure 11-29. Waiting for Servers dialog on Client Machine**

Note you must put the servers in waiting mode before you set the client to Record Endpoints using Servers. If you don't, DPL on the client machine will indicate that no servers were found and abort the parallel recording run.

Once complete, the client machine displays the "Endpoint recording is complete" message. There will be a new item included in the Workspace Manager for an "{Imported Endpoint Database}". See Figure 11-30.

⇒ Click OK to the "Endpoint recording is complete" message.



**Figure 11-30. Endpoint Recording Complete dialog with Imported Endpoint Database added to the Workspace**

When the run is complete, the Server machines revert back to the "Watching <<your working directory>>" message.

If you'd like, you can play endpoints once again on the client machine to ensure the imported, merged endpoints that were recorded in parallel on multiple servers generate results that match those from the previous sections.

⇒ Click the Cancel button within the Waiting for Command dialog on the server machines. This takes the servers out of Endpoint Server mode.



## 12. Advanced Decision Analysis Results

This chapter discusses additional decision analysis results and outputs available in DPL including further features of Policy Trees and Policy Summaries; Subtree Risk Profiles; and Option Value Diagrams. Chapter 3 provides an overview of DPL's fundamental decision analysis outputs. You may wish to review that chapter before proceeding with this chapter.

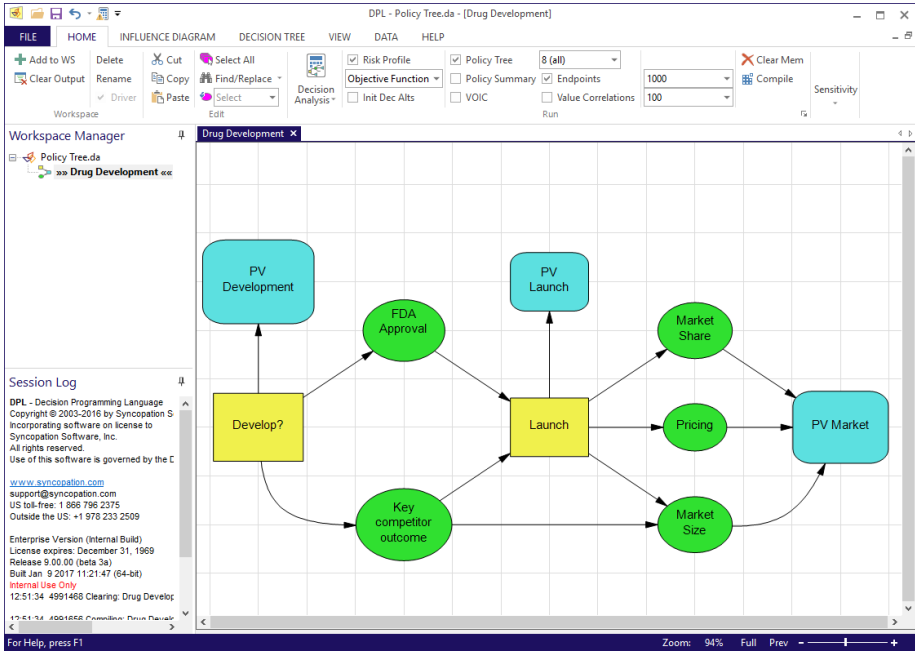
This chapter assumes that you are familiar with the basics of using DPL, i.e., with the material covered in Chapters 2 through 0 of this manual.

### 12.1 Policy Tree™ Features

---

You will start by looking at the Format Display features for pruning Policy Trees to make them easier to examine and analyze.

- ⇒ Select File | Open.
- ⇒ Open the workspace Policy Tree.da. This file is found in the Examples folder where DPL was installed, usually  
C:\Program Files\Syncopation\DPL9\Examples. See Figure 12-1.

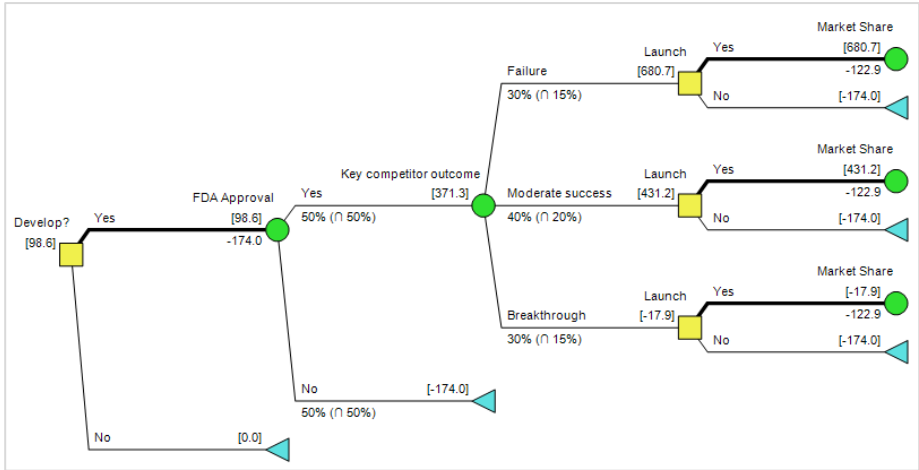


**Figure 12-1. Policy Tree.da**

This is a drug development example.

- ⇒ Make sure Risk Profile, Policy Tree, and Endpoints are checked in the Home | Run group.
- ⇒ Make sure that Init Dec Alts and all other outputs are unchecked.
- ⇒ Run a Decision Analysis.

DPL launches Excel if it is not already running and loads Policy Tree.xlsx, runs the model, and produces the requested outputs. The Policy Tree as shown in Figure 12-2 will be active.



**Figure 12-2. Policy Tree™ for Drug Development Model**

You can use the Format Policy Tree dialog (Policy | Display | Settings) to prune some of its branches. Suppose you are interested in seeing the scenarios in which the objective function (the net present value of the cash flows in this example) is positive. To do this you first need to have an estimate of the maximum value for the objective function.

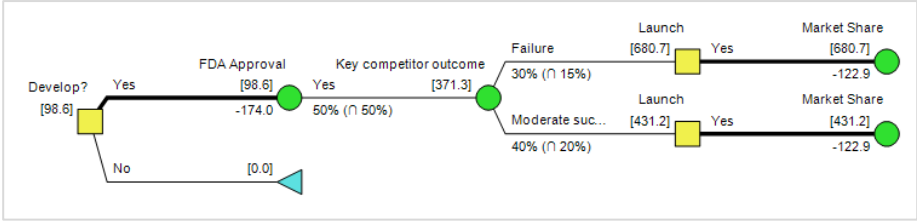
- ⇒ Double-click the Risk Profile Chart item (📊) labeled Expected Value in the Workspace Manager. The Risk Profile Chart for the objective function is activated.
- ⇒ Note the maximum value on the x-axis. It is approximately 3500.
- ⇒ Double-click the Policy Tree item in the Workspace Manager to activate it again.
- ⇒ Click Policy | Display | Settings.

The Format Policy Tree dialog appears as shown in Figure 12-3. In addition to allowing you to modify the appearance of the tree by changing the branch length and branch offset and by specifying which items to display, the Format Policy Tree dialog allows you to filter branches based on probabilities or by the objective function Expected Values.

**Figure 12-3. Format Policy Tree Dialog**

- ⇒ Within the *Branch filter* section of the dialog, check the *Based on rolled-back objective function EVs* checkbox.
- ⇒ Leave the *Min:* edit box in the rolled-back objective function values section filled in with "0".
- ⇒ In the *Max:* edit box, type in "3500" (the value you noted from the Risk Profile).
- ⇒ Click OK.

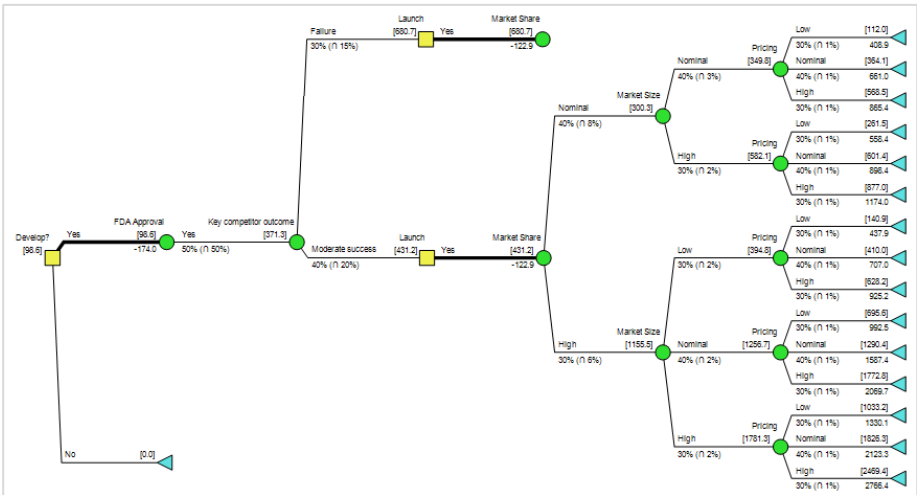
As shown in Figure 12-4, DPL filters out any branch in the Policy Tree whose rolled-back expected value is negative. Note that in Figure 12-4 there is no longer a branch for the No outcome of FDA Approval. This is because if the FDA does not approve the drug, then the expected value is negative. Similarly, there is no longer a branch for Key competitor outcome of Breakthrough. The expected value is less than zero if the key competitor achieves a breakthrough outcome. Lastly, there are no branches shown for Launch equal to No in either of the scenarios where the key competitor fails or has moderate success.



**Figure 12-4. Policy Tree™ Filtered on Rolled-Back Expected Values**

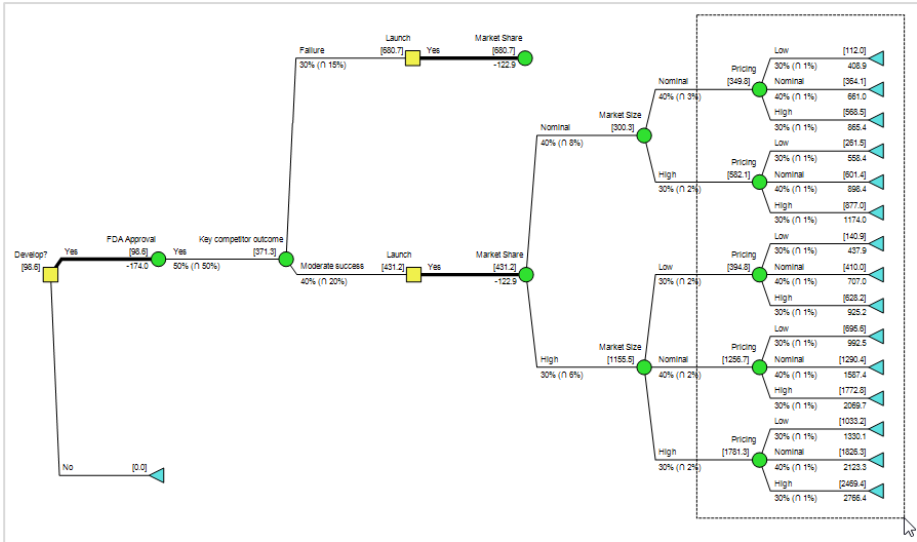
- ⇒ Select the Market Share chance node at the end of the Moderate success branch of Key competitor outcome and the Yes branch of Launch.
- ⇒ Drop-down the Policy | Display | Expand split button and select Expand Subtree from the list.
- ⇒ Click the Full button within the zoom controls or press Ctrl+L to Zoom Full.

The Policy Tree should now look like Figure 12-5. The Policy Tree is difficult to read at this zoom level. You will zoom in on a portion of the Policy Tree.



**Figure 12-5. Filtered Policy Tree™ with Expanded Subtree**

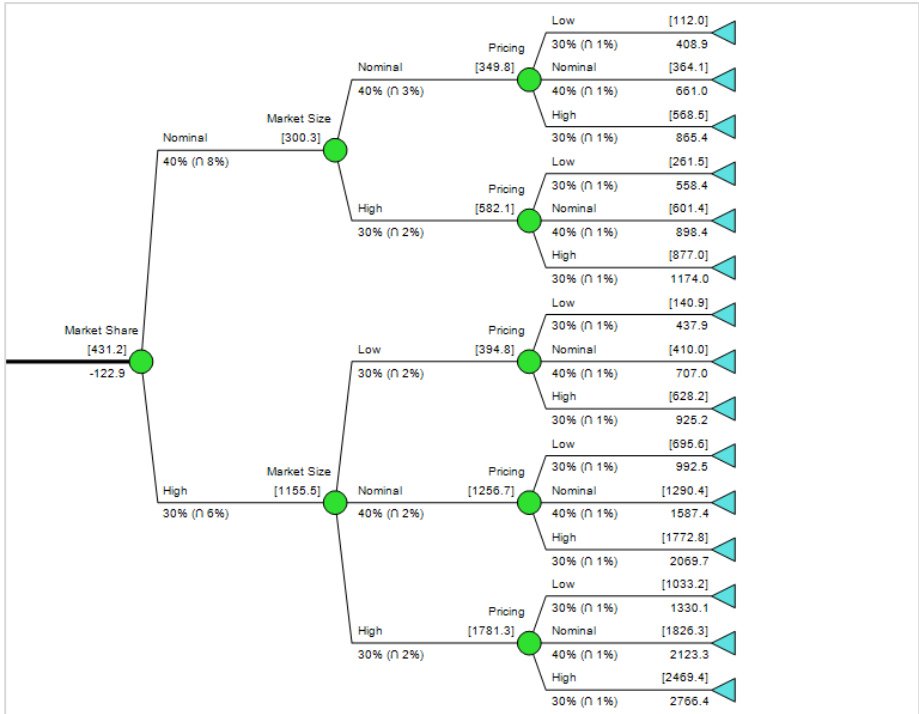
- ⇒ While holding Ctrl + Shift, left-mouse click above the Market Size node at the end of the Nominal branch of Market Share and without letting go, drag down and to the right to create a rectangle around the endpoints of the expanded subtree as shown in Figure 12-6.



**Figure 12-6. Using the Mouse to Zoom a Portion of the Policy Tree™**

⇒ Release the mouse button.

DPL zooms to the portion of the Policy Tree you selected as shown in Figure 12-7.



**Figure 12-7. Policy Tree™ Zoomed to Expanded Subtree**

You can see from Figure 12-7 that there is no Low branch for Market Share because of the filter. The rolled-back expected value at that node is negative. Similarly, there is no Low branch for Market Size at the end of the Nominal branch for Market Share; the rolled-back expected value is negative at that point as well.

The branch filtering feature can be used to give you information on which scenarios lead to negative expected values. Note that if a branch does not appear because its rolled-back expected value is outside the range of the filter, this means that the expected value over all the scenarios further down the subtree is outside the range of the filter; it does NOT mean that EVERY scenario further down the subtree is outside the range of the filter.

You can also filter branches based on probabilities, as shown in the dialog in Figure 12-3. If you do this, branches with probabilities outside the range you specify will be hidden from display. The filtering is handled similarly to the rolled-back expected value filtering. That is, if a branch is filtered out, all subsequent branches (further down the subtree) will also be hidden from display, regardless of whether their probabilities are in the range you specify.

- ⇒ Click Policy | Display | Settings.
- ⇒ Uncheck Based on rolled-back obj. fn. expected values.
- ⇒ Click OK to clear the filter.

## 12.2 Subtree Risk Profile™

---

Subtree Risk Profiles can be used to view the risk profile for all of the scenarios below any particular branch in the Policy Tree. A subtree is defined as a combination of fixed states of one or more events in your model. Events that are not defined as part of the subtree are allowed to vary as in a Decision Analysis run, generating a Subtree Risk Profile chart.

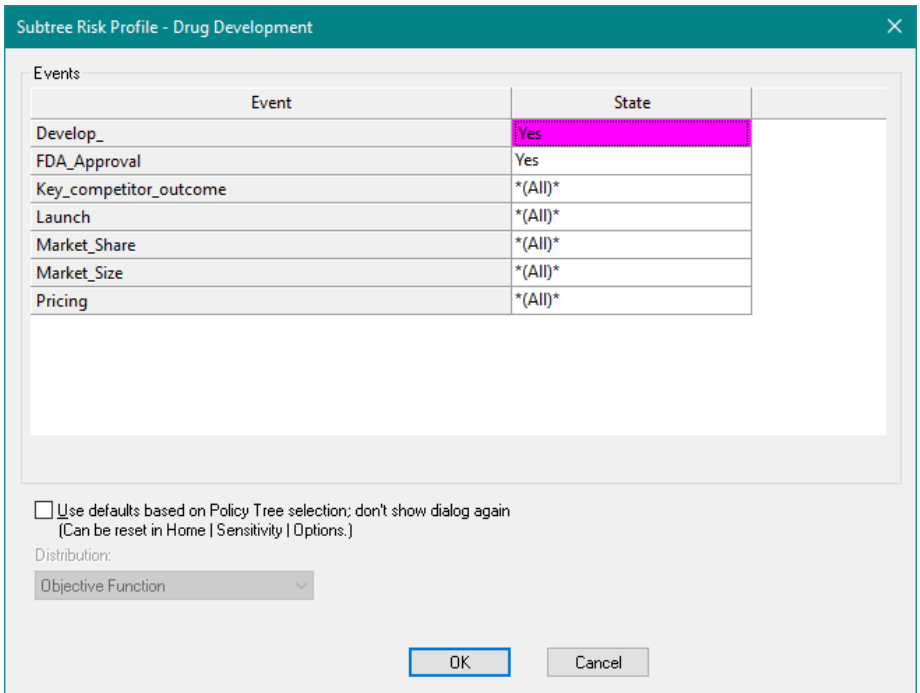
Subtree Risk Profiles use an Endpoint Database (see Chapter 11 of this manual) in conjunction with a Policy Tree, so you must run Full tree enumeration and Record Endpoints in order to create them.

- ⇒ Make sure you have an Endpoint Database, Policy Tree and Risk Profile from the example in the previous section. If not, create one of each now by re-running the model.
- ⇒ Right-click on the Risk Profile Dataset item called {Expected Value}.
- ⇒ Rename it Full Profile.

You have created a copy of the risk profile for the full model for comparison with a risk profile for a subtree.

- ⇒ Double-click on the Policy Tree to make it active.
- ⇒ Select the chance node for Key competitor outcome. (You may need to scroll left slightly to see it.)
- ⇒ Click Policy | Subtree Risk Profile | Add. You will see the Subtree Risk Profile dialog as in Figure 12-8.

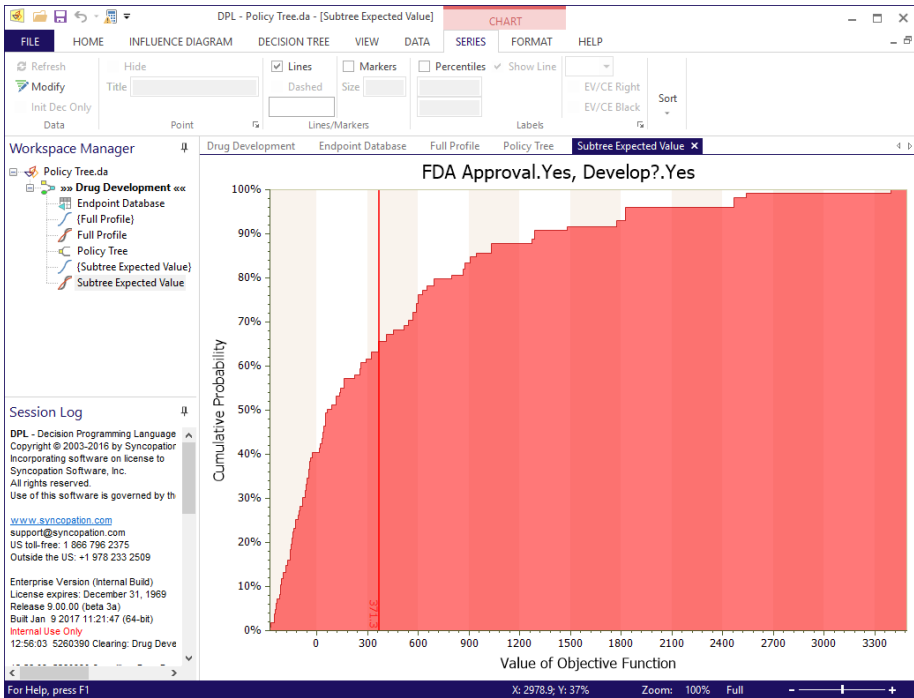




**Figure 12-8. Subtree Risk Profile™ Dialog**

You asked for a Subtree Risk Profile from the point in the Policy Tree where the initial decision (Develop) is Yes and the outcome of FDA Approval is Yes. The dialog is initialized to these settings and indicates that this is the subtree you have selected. All other events in the tree are denoted *\*(All)\** which means that they are allowed to vary as usual; only the initial decision and the first chance node are fixed.

⇒ Click OK to generate the Subtree Risk Profile, See Figure 12-9.



**Figure 12-9. Subtree Risk Profile™ Chart**

You will see a Risk Profile Dataset and a Risk Profile Chart item, both called Subtree Expected Value. Note: Subtree Risk Profile charts and datasets will overwrite Expected Value charts and datasets. Since you renamed the Expected Value dataset to Full Profile it did not get overwritten.

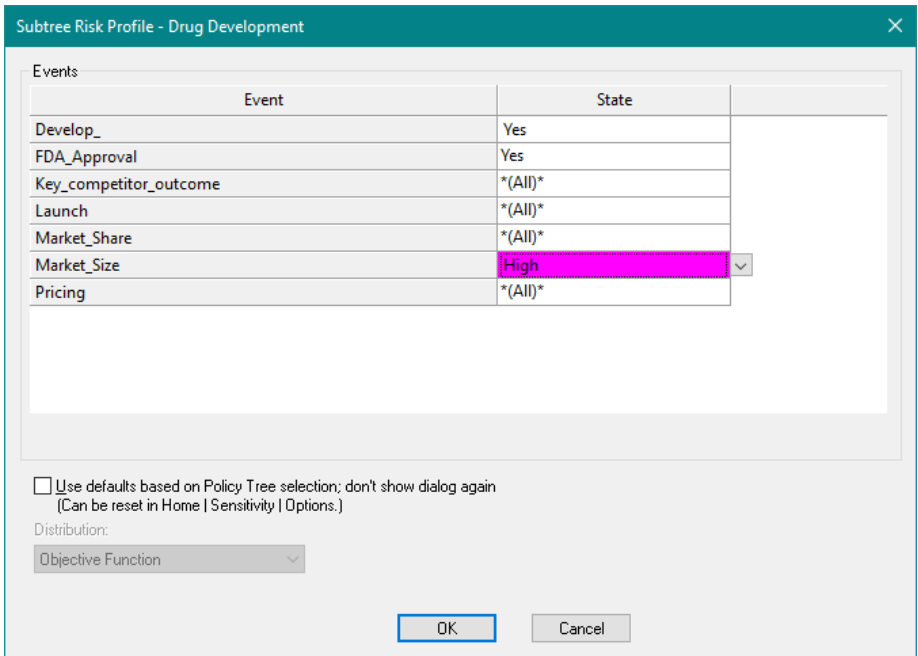
As you would expect, the risk profile and expected value for this Subtree are very different from the full risk profile because you have removed the possibility that the product is not approved. You can view both the Subtree Risk Profile and the Full Profile within a single chart for comparison via the Modify Data dialog (Chart | Series | Data | Modify).

Also note that the Subtree Risk Profile has a title indicating the states of the events that are not varying, i.e. Develop and FDA Approval are both set to Yes.

You will generate one more Subtree Risk Profile using the dialog.

- ⇒ Open the Policy Tree again.
- ⇒ This time right-click on the chance node for Key competitor outcome and select Subtree Risk Profile from the context menu to open the dialog.

⇒ Use the combo box to select the High setting for Market Size. See Figure 12-10.



**Figure 12-10. Subtree Risk Profile™ Dialog with Market Size High**

⇒ Click OK.

DPL generates a new Subtree Risk Profile. Note that this Subtree Risk Profile replaced the previous one, since you did not rename the previous one. Finally, note that this Subtree Risk Profile corresponds to a set of endpoints that cannot be defined just by starting from a location on the Policy Tree since the three events whose states are fixed are not at the head of the Policy Tree and consecutive.

The Subtree Risk Profile dialog has a checkbox option to use the default settings for each event based on where you are in the Policy Tree. If you do not want to be prompted with the Subtree Risk Profile dialog, check this option. When you select Policy | Subtree Risk Profile | Add, the Subtree Risk Profile will automatically be created using your selection in the Policy Tree and the default settings for the remaining events. You can also change this setting within the Run Settings dialog when accessed via the Home | Sensitivity | Settings. Note that the dialog allows you to create subtree risk profiles for some combinations of event states that don't

correspond to any branch in the Policy Tree, such as the one you just created.

- ⇒ Close the Workspace without saving changes. You will not need to use it later in this chapter.

## 12.3 Policy Summary™ Features

---

As discussed in Section 3.3, the Policy Summary provides information about the probability-weighted percentage that each decision alternative is selected in the set of optimal endpoints or scenarios. For example, in the model discussed in that section, the Yes alternative of the Launch decision is optimal in a probability-weighted 10% of the scenarios. You may wish to know in which scenarios the Launch alternative is optimal. In that model, the Risk Profile indicates that the objective function (NPV in this case) is negative in nearly 50% of the scenarios. You may wish to know which scenarios lead to losing money. DPL provides several features that allow you to compare the full set of optimal scenarios to those scenarios filtered on specific criteria. These comparison features help you find out the type of information alluded to in the above examples.

This section covers the Comparison (filtering) features of the Policy Summary output.

- ⇒ Select File | Open.
- ⇒ Navigate to the Examples folder underneath the DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples.
- ⇒ Select Wildcat.da.

DPL opens the Workspace for the Wildcat model as shown in Figure 12-11.

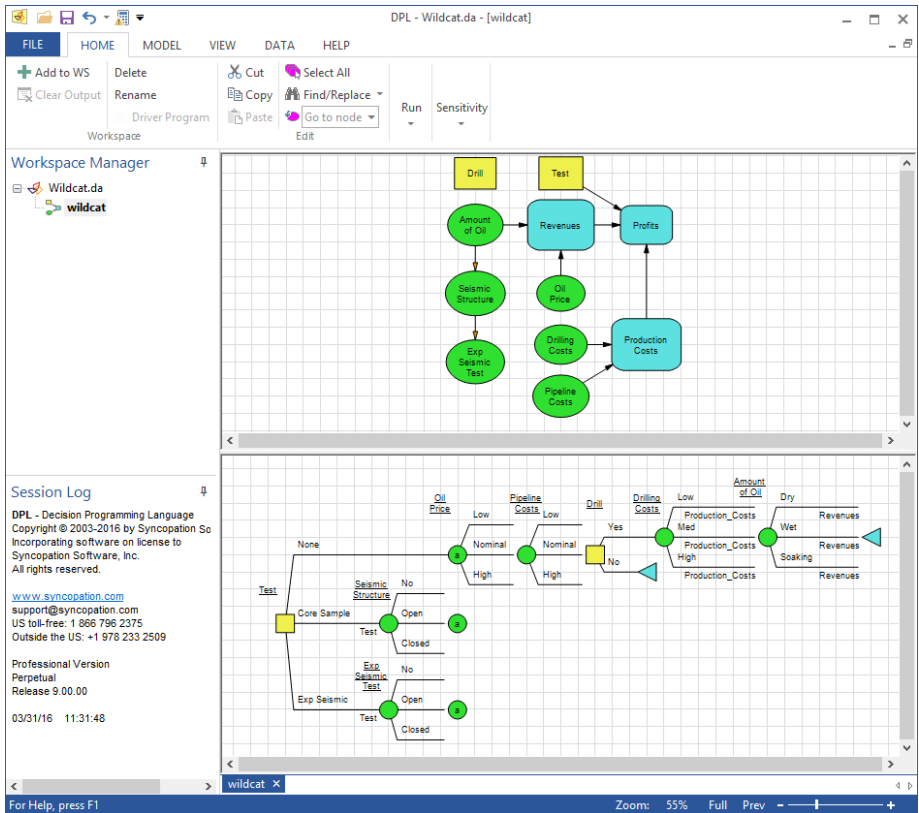


Figure 12-11. Wildcat Model

The Wildcat model is a simple model of oil exploration. The Decision Tree is asymmetric (see Section 2.2). There is an upfront decision to conduct one of two tests or proceed without testing. If a Test is conducted, information on the seismic structure is gained which provides an indication of the amount of oil. If no test is conducted, this information is not gained. After either testing or not, uncertainties regarding oil price and pipeline costs are resolved. These are followed by a decision to drill or not and further uncertainties regarding drilling costs and the amount of oil.

The results of the two tests are conditioned on the actual amount of oil present, however, the results of the tests are available before the amount of oil is known. DPL uses Bayesian revision (see Chapter 7) to calculate the unconditional probabilities of the test outcomes.

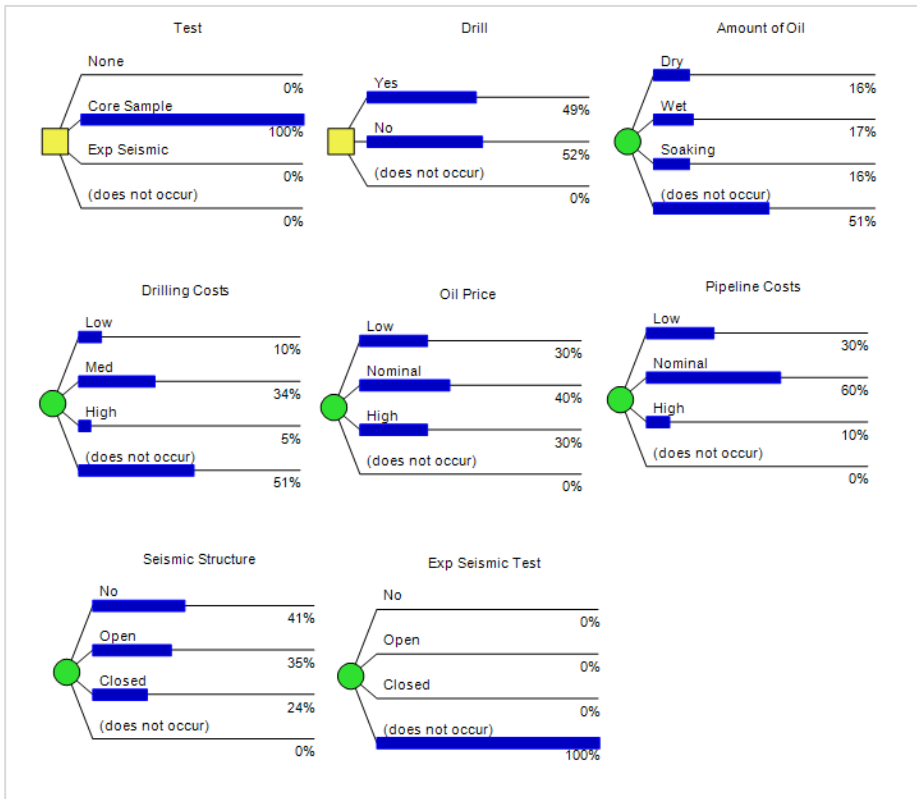
⇒ In the Home | Run group check Risk Profile and Policy Tree only. Make sure to uncheck Init Dec Alts.

⇒ Click Home | Run | Decision Analysis to run an analysis and generate the outputs.

DPL runs the model and produces the requested outputs. The Policy Tree will be active.

As mentioned within Section 3.3, you could have requested the Policy Summary output when you ran the Decision Analysis. Or, you can optionally choose to create a Policy Summary from the Policy Tree, as you'll do now.

⇒ With the Policy Tree active, select Policy | Policy Summary | Add from the command ribbon. DPL creates a Policy Summary as shown in Figure 12-12.



**Figure 12-12. Policy Summary™ for the Wildcat Model**

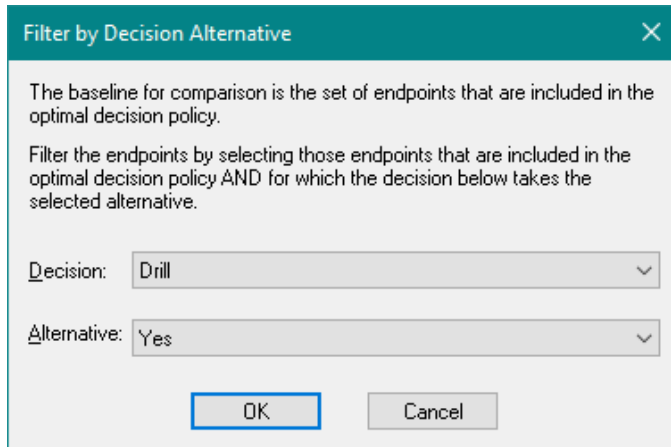
DPL displays the policy dependent probabilities for each state of each event in the decision model in the Policy Summary. It also displays these probabilities graphically by drawing a blue bar proportionate in length to the probability on each branch. As the Policy Summary indicates, in just under half of the scenarios it is optimal to drill and in just over half it is optimal to not drill. The Policy Tree contains all the information needed to create a Policy Summary and Policy Summary comparisons.

You will now use DPL's Filter by Decision Alternative feature to give you information on which event states are correlated with drilling.

⇒ Activate the Policy Tree again.

⇒ Select Policy | Policy Summary | Compare | By Decision Alternative...

The Filter by Decision Alternative dialog appears as shown in Figure 12-13.

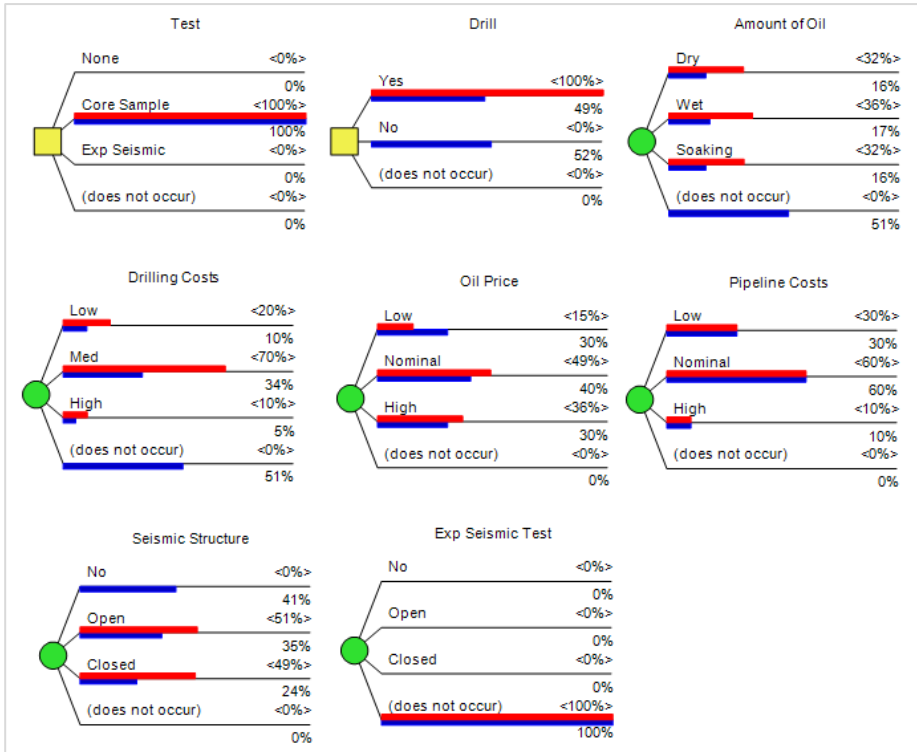


**Figure 12-13. Filter by Decision Alternative Dialog**

The dialog allows you to select which decision and alternative you would like to use for the filter. As the dialog indicates, the endpoints will be filtered to select the endpoints that are included in the optimal decision policy AND for which the selected decision takes the selected alternative.

⇒ The decision and alternative in which you are interested (Drill and Yes) are already selected. Click OK.

DPL creates a new filtered Policy Summary as shown in Figure 12-14. Note that the original Policy Summary is still in the Workspace Manager; it was not overwritten.



**Figure 12-14. Policy Summary™ with Comparison Bars for Drill.Yes**

In comparison mode, DPL displays graphically the policy dependent probabilities for the full set of optimal endpoints with a blue bar below each branch. It also displays graphically the policy dependent probabilities for the filtered endpoints with a red bar above each branch. DPL displays the numeric values for the filtered policy dependent probabilities in angle braces (e.g., <32%>) above the end of each branch. As before, the policy dependent probabilities for the unfiltered set of optimal endpoints are displayed below the branch.

Chance outcomes or decision alternatives that have a red bar that is longer than their blue bar are more likely in the filtered scenarios than they are in the full set of optimal scenarios. These event states are positively correlated with the decision alternative used for the filter. For example, in Figure 12-14 you can see that Seismic Structure.Open and Seismic Structure.Closed both have red bars that are longer than the blue bars. This means that both these two outcomes are more likely given that the Yes alternative to Drill is optimal. This tells you that these outcomes are positively correlated with the decision to drill. You can also see that oil

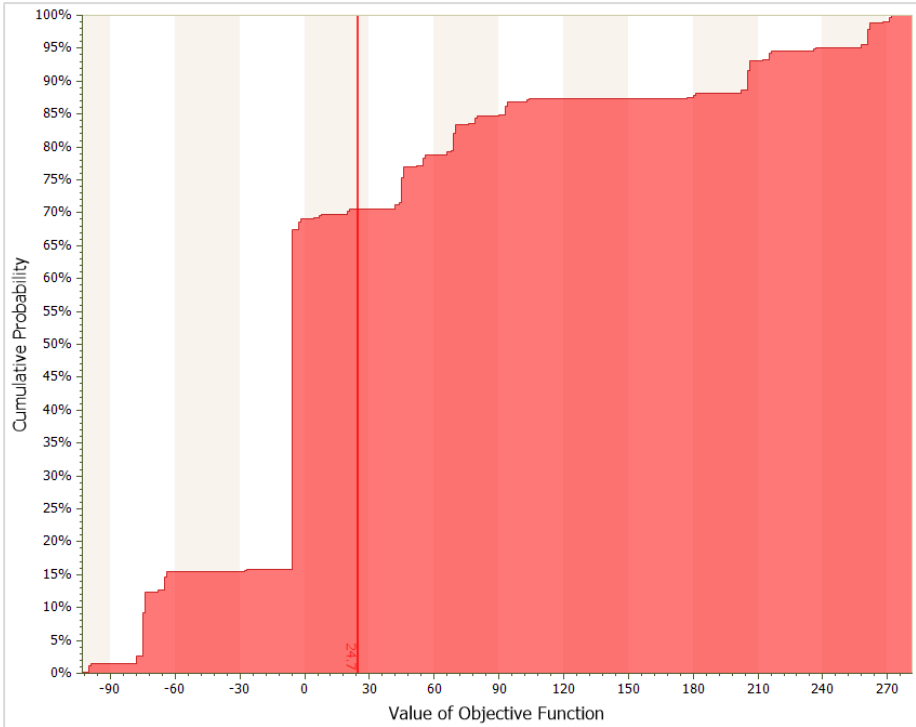


price is more likely to be nominal or high in scenarios where drilling is the optimal decision alternative.

Conversely, chance outcomes or decision alternatives that have a red bar that is shorter than their blue bar are less likely in the filtered scenarios than they are in the full set of optimal scenarios. These event states are negatively correlated with the decision alternative used for the filter. For example, in Figure 12-14 you can see that Seismic Structure.No has no red bar at all. This means that this outcome never occurs given that the Yes alternative to Drill is optimal. This information can be used to help develop a decision rule, i.e., if Seismic Structure is No, then do not drill. You can also see that oil price is less likely to be low in scenarios where drilling is the optimal decision alternative (in fact it is 50% less likely: 15% vs. 30%).

⇒ Double-click the Risk Profile Chart item in the Workspace Manager labeled Expected Value.

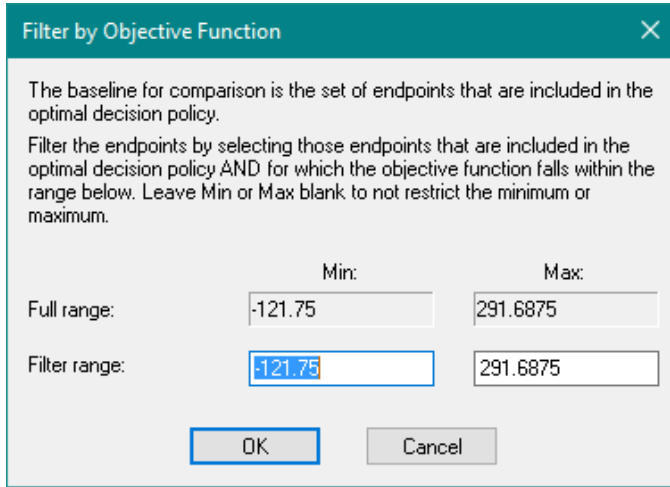
DPL activates the Risk Profile chart for the Wildcat model as shown in Figure 12-15. The Risk Profile Chart indicates that in about 10% of the scenarios the objective function exceeds 200.



**Figure 12-15. Risk Profile Chart for Wildcat Model**

You will now use DPL's Filter by Objective Function feature to find out more about which scenarios lead to big gains.

- ⇒ Double-click the Policy Tree item in the Workspace Manager to activate it again.
- ⇒ Select Policy | Policy Summary | Compare | By Objective Function... The Filter by Objective dialog appears as shown in Figure 12-16.



**Figure 12-16. Filter by Objective Function dialog**

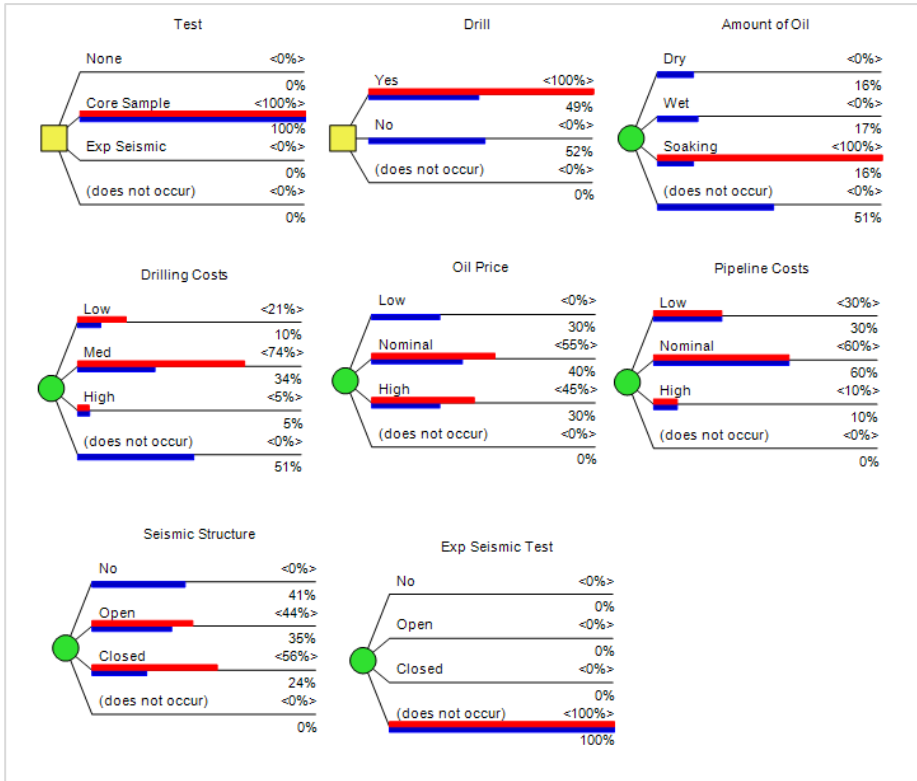
DPL displays the full range of the risk profile of the model to help you select which range you would like to specify.

- ⇒ In the *Min* edit box for *Filter range* type "200".
- ⇒ Leave the *Max* edit box for *Filter range* at its specified value.

Note that the low and high for the range are inclusive values. DPL will filter based on the objective function being greater than or equal to the low value and less than or equal to the high value.

- ⇒ Click OK.

By leaving the high setting for the filter range at the high setting for the full range, you are asking DPL to filter the Endpoint Database for all optimal endpoints with an objective function value greater than or equal to 200. DPL creates another filtered Policy Summary as shown in Figure 12-17.



**Figure 12-17. Policy Summary™ with Comparison Bars for Objective Function Filter**

The Objective Function  $\geq 200$  Policy Summary Comparison indicates that the Soaking outcome of Amount of Oil occurs in 100% of the filtered scenarios. The comparison also indicates that the No outcome of Seismic Structure never occurs in scenarios where the objective function exceeds 200. You can also see that the Yes alternative of Drill occurs in 100% of the scenarios. This information tells you that if Seismic Structure is in the No outcome there is zero chance that the objective function will exceed 200. In order for the objective function to exceed 200, you must drill and the amount of oil must be soaking.

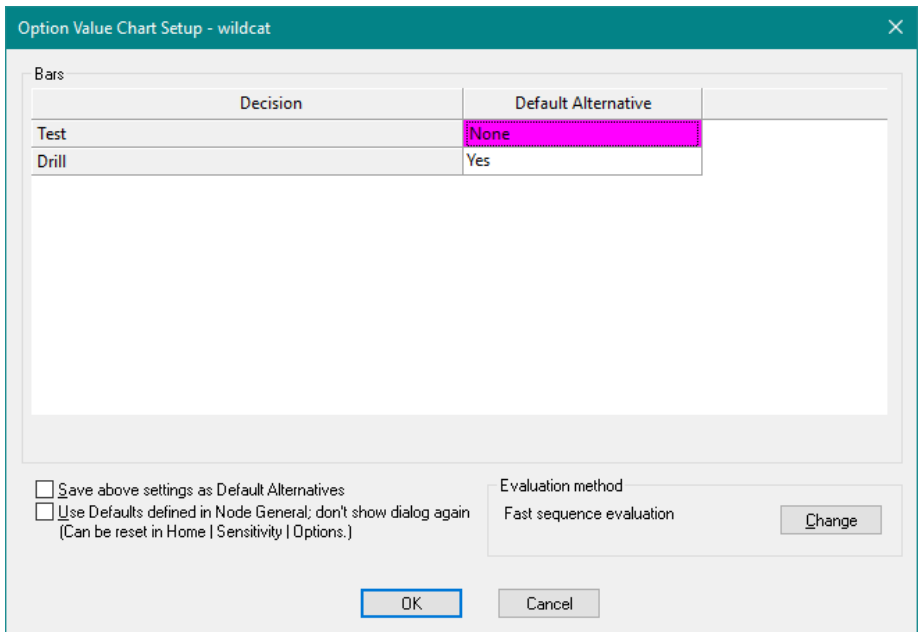
Note that DPL gives each filtered Policy Summary a name that indicates which filter is active and how it is set. You can rename or delete these comparisons within the Workspace Manager. DPL will not overwrite them when you run the model, and they are not cleared when you clear model memory.

## 12.4 Option Value Charts™

Option Value Charts display the value of each "option" (decision) in your model. In DPL, the value of each option (decision) is defined as the difference in the objective function value when the model is run with that decision uncontrolled vs. the value when the model is run with the decision controlled to a fixed alternative.

You will run an Option Value Chart using the Wildcat model from the previous section. The process DPL uses for calculating option values will be discussed in the context of this example chart.

- ⇒ Open the model from the previous section (Wildcat.da) and activate the model.
- ⇒ Click Home | Sensitivity | Option Value. The Option Value Chart setup dialog appears as in Figure 12-18.

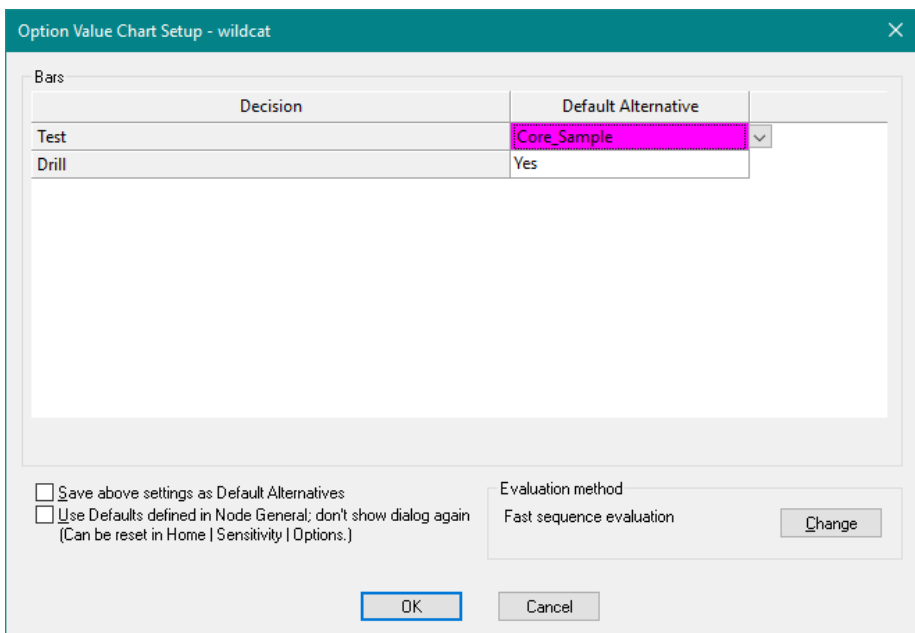


**Figure 12-18. Option Value Chart™ Setup Dialog**

This dialog is asking you to specify the default alternative states for each decision in your model for use in your Option Value Chart. Note that the default states the dialog suggests are the same as the default states for each decision as specified on the General tab of the Node Definition dialog. You can check the checkbox at the bottom left of the dialog if you wish not to be prompted for the defaults each time. However, when you use Option Value Charts, you will usually want to examine (and possibly change) the default states.

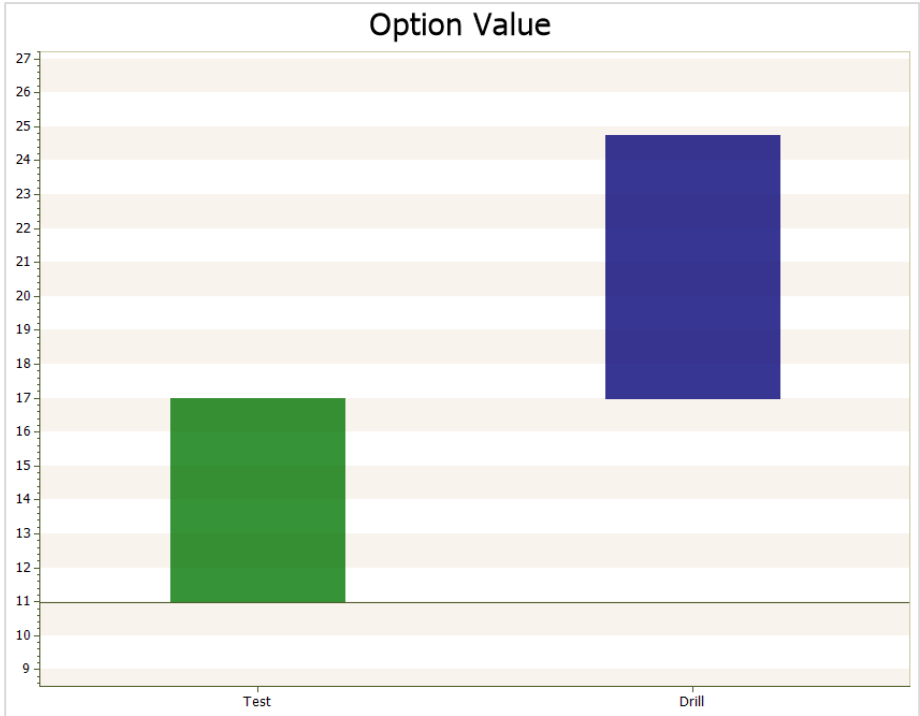
You will change one of the default settings for purposes of generating the chart.

- ⇒ Use the combo box for the Test decision to change the default alternative to Core Sample.
- ⇒ Leave the default alternative for Drill set at Yes. The dialog should look like Figure 12-19.



**Figure 12-19. Option Value Chart™ Setup with Changes**

- ⇒ Click OK. DPL creates the Option Value Chart as in Figure 12-20.



**Figure 12-20. Option Value Chart™ for Wildcat Model**

DPL calculated the option values in this chart by first running the model with the Test decision controlled to Core Sample and the Drill decision controlled to Yes. If this is always the strategy, then the objective function value is approximately 11, which is the value at the bottom of the first bar on the chart.

DPL then ran the model with the first decision (Test) uncontrolled, and the Drill decision still controlled to Yes. If the Drill decision is always Yes, then there is no value associated with testing, so the optimal alternative for Test is None. The objective function in this case is 17, which is the value at the top of the first bar (and bottom of the second bar) on the chart.

Finally, DPL uncontrolled both decisions and ran the model again. In this case the objective function value is 24.7, which is the value at the top of the second bar on the chart, and is also the value you will get if you simply run the Wildcat model as usual. This value reflects the optimal initial decision alternative, which is to Core Sample, as well as full flexibility in the downstream option to Drill.

If there were more decisions in the model, DPL would continue from left to right in the tree, uncontrolling decisions one at a time and adding a bar for each decision. The cumulative height of the last bar in the chart is always equal to the objective function when it is run with all decisions uncontrolled. However, the starting value for the first bar as well as the lengths of the individual bars may vary depending on your choice of default alternatives.

Note that the choice of the default alternatives can make a significant difference in the appearance of the chart. For each decision (option) in your model, the length of the bar can be interpreted as the incremental value of having flexibility in that option, given that all previous options are also flexible (uncontrolled) and all subsequent options are controlled to the default alternatives you select.

You may find that you wish to change the default alternatives in the Option Value dialog, and save the new settings. To do this, check the first checkbox in the lower left corner of the Option Value Chart Setup dialog. This accomplishes the same thing as changing the default alternatives using the General tab of the Node Definition dialog.



## 13. Risk Tolerance

While there are theoretical arguments for making decisions based on expected value alone, often decisions are made with some consideration of the degree of risk involved in the different alternatives. In many cases, good risk visualization is enough and a judgment about risk can be made after looking at Risk Profiles, Time Series Percentiles, Tornado Diagrams, and other results.

In situations where explicitly modeling attitudes toward risk is desired, DPL gives you the ability to model your risk aversion (or risk seeking) in terms of a utility function. To model attitudes toward risk using an exponential utility function (a standard form well supported by research), you need only specify a single number: the risk tolerance coefficient.

This chapter assumes you have already been through at least one of the earlier tutorials (Chapters 1 through 0) in this manual, or that you are already familiar with DPL. Therefore, basic procedures such as starting DPL and opening and saving workspaces will not be explained in detail here.

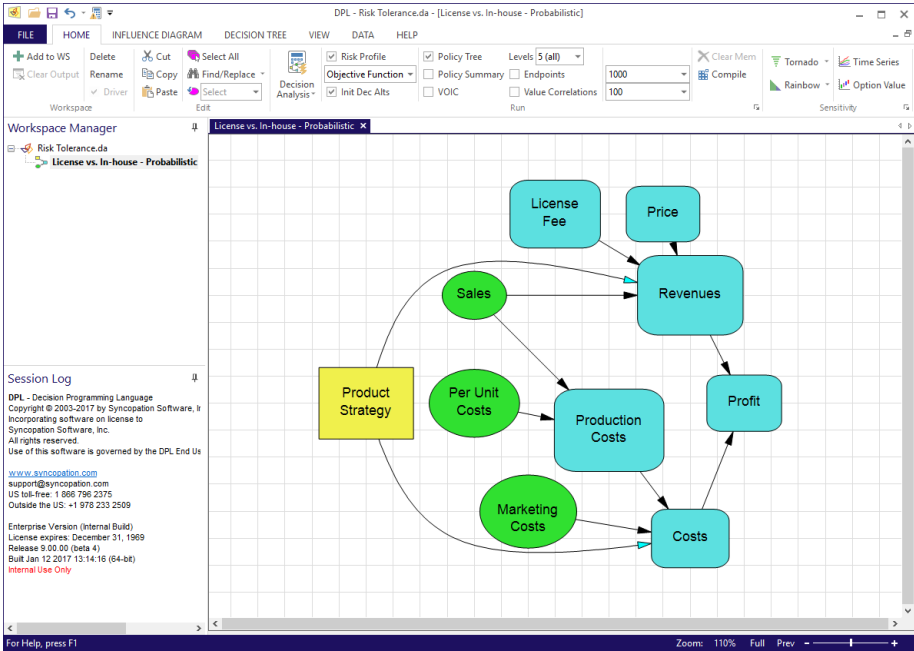
### 13.1 Incorporating a Risk Tolerance

---

You'll start with a file containing a model similar in concept to the one introduced in Chapter 4.

- ⇒ Start DPL.
- ⇒ Open Risk Tolerance.da in the Examples folder underneath where DPL is installed, usually C:\Program Files\Syncopation\DPL9\Examples.

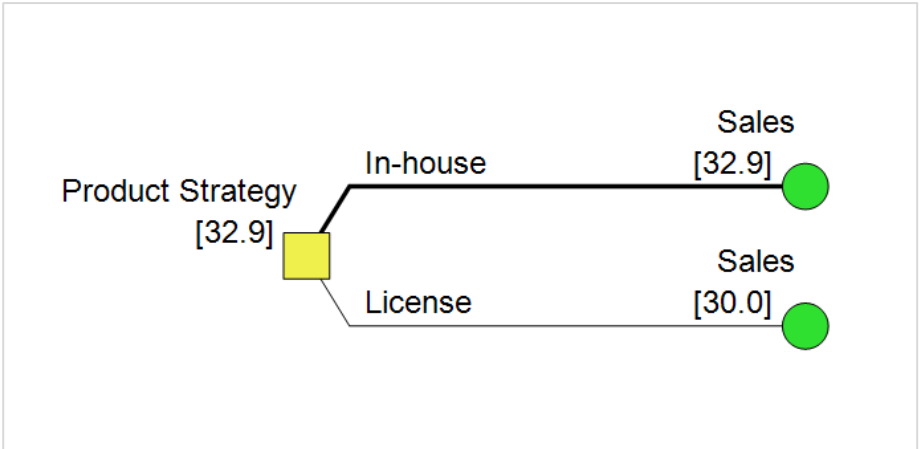
DPL opens the Workspace as shown in Figure 13-1.



**Figure 13-1. License vs. In-house Model**

Run the model to review the current results.

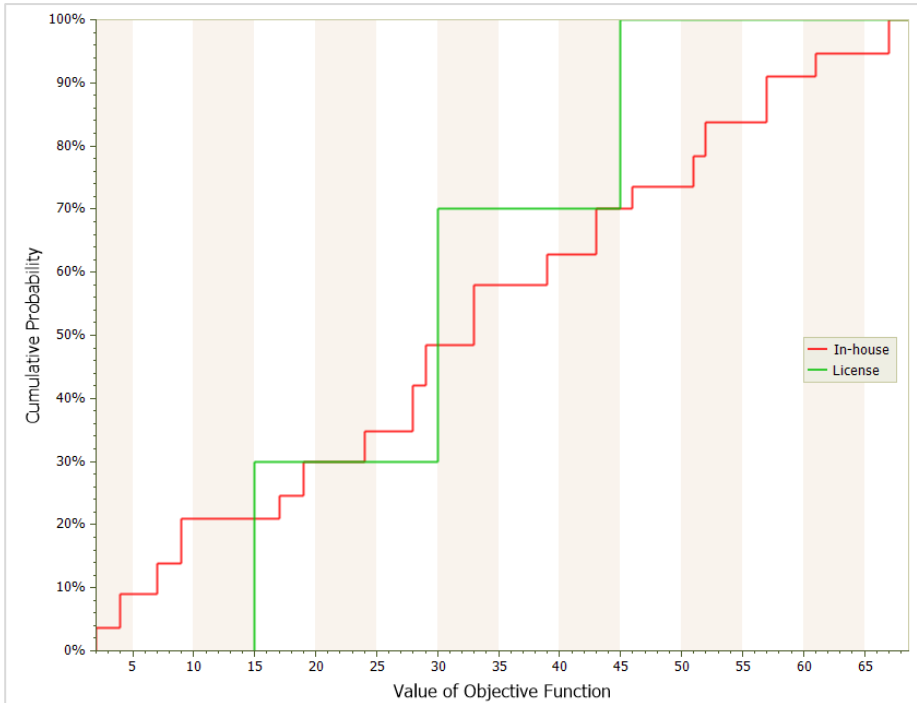
- ⇒ Make sure Risk Profile, In Dec Alts and Policy Tree are checked in the Home | Run group.
- ⇒ Select Home | Run | Decision Analysis (or press F10).



**Figure 13-2. Policy Tree™**

The Policy Tree (Figure 13-2) shows that In-house is the optimal alternative. The expected value of In-house is \$2.9 million more than the expected value of License.

- ⇒ Navigate to the Risk Profile Chart called Initial Decision Alternatives.
- ⇒ Uncheck EV/CE Lines in Chart | Format | Display.



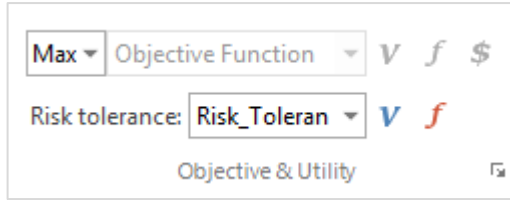
**Figure 13-3. Risk Profiles**

The Risk Profiles of the two alternatives show a more ambiguous result (Figure 13-3). In-house is better in terms of expected value, but it is also more risky. Would a different attitude toward risk ever lead you to choose License? You will add a new value node to the model that contains data for your risk tolerance.

- ⇒ Add a Value node to the model, place the node near Profit, and name it "Risk Tolerance".
- ⇒ Switch to the Data tab and enter "50" for the value.
- ⇒ Click OK to close the Node Definition dialog.

Note that within the Influence Diagram | Objective & Utility group there is a combo box labeled *Risk tolerance*.

- ⇒ Drop-down the Risk tolerance box and select Risk\_Tolerance from the list. See Figure 13-4.



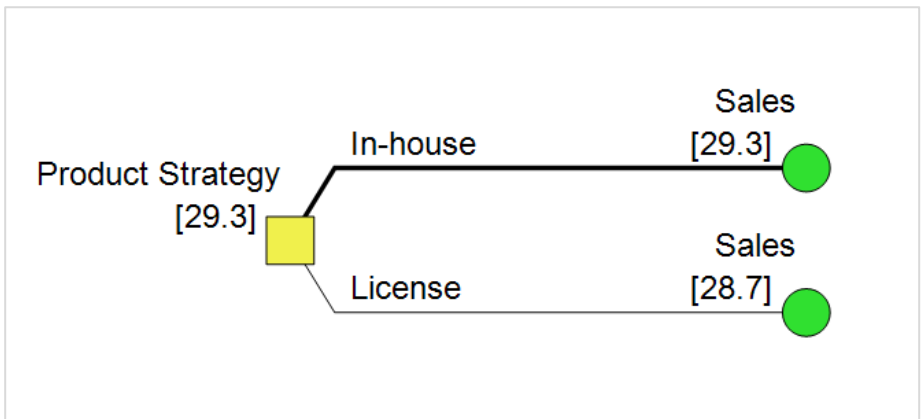
**Figure 13-4. Risk Tolerance Combo Box**

You might ask where the 50 comes from. In rough terms, it's the amount you (or your company) are willing to put at risk. You can assess your risk tolerance coefficient directly using the following rule. Imagine that a fair coin will be tossed. If you call it correctly, you win  $R$ , if not, you lose half of  $R$ . What is the largest  $R$  for which you would be willing to play this game? The largest  $R$  for which you would be willing to play is the risk tolerance coefficient.

Risk tolerance coefficients are difficult to assess and are never very precise. You should always test your risk tolerance coefficient using sensitivity analysis (as you will later in this tutorial). For this reason, it is good to define a value node for your risk tolerance rather than specifying the number directly. If you did the latter, you would not be able to run a sensitivity analysis on it.

You'll now run the model and observe how the results have changed.

⇒ Select Home | Run | Decision Analysis. Click Yes for the warning.



**Figure 13-5. Policy Tree™ with Certain Equivalent results**

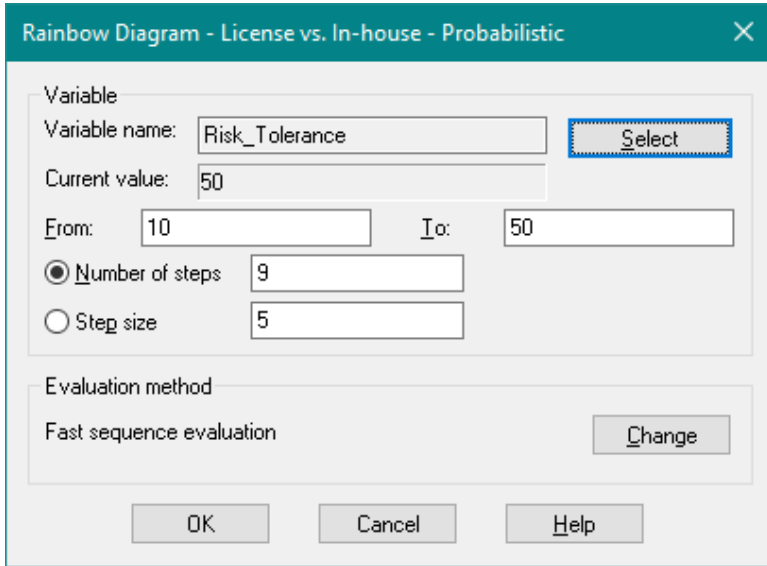
In-house is still the preferred alternative, but the gap has narrowed. The results in Figure 13-5 are Certain Equivalents, which are being shown because risk tolerance is now incorporated in your model. Pursuing the In-house alternative might result in an outcome better or worse than 29.3, but a decision maker with a risk tolerance of 50 would be indifferent between the investment and a perfectly safe 29.3. If, as is usually the case, the decision maker is risk averse, the Certain Equivalent will be less than the Expected Value.

Within the View | Results group, the CE button is shaded blue to indicate that the results shown in the Policy Tree are for Certain Equivalents. If you would like to see the Expected Value results (Figure 13-2) again, click View | Results | CE to turn off the Certain Equivalents view. DPL updates the Policy Tree to show the EV results.

In a model with a risk tolerance, DPL always calculates both EV (Expected Value) and CE (Certain Equivalent) results. With a risk tolerance coefficient of 50, the optimal decision is unchanged. How much lower would it have to go (i.e., how much more risk averse would you have to be) for the License alternative to have a higher certain equivalent than the In-house alternative?

- ⇒ Click Home | Sensitivity | Rainbow Diagram.
- ⇒ Click the Select button to choose Risk\_Tolerance for your *Value for sensitivity* and then click OK.
- ⇒ Type "10" for *From*, "50" for *To*, and "5" for *Step size*.

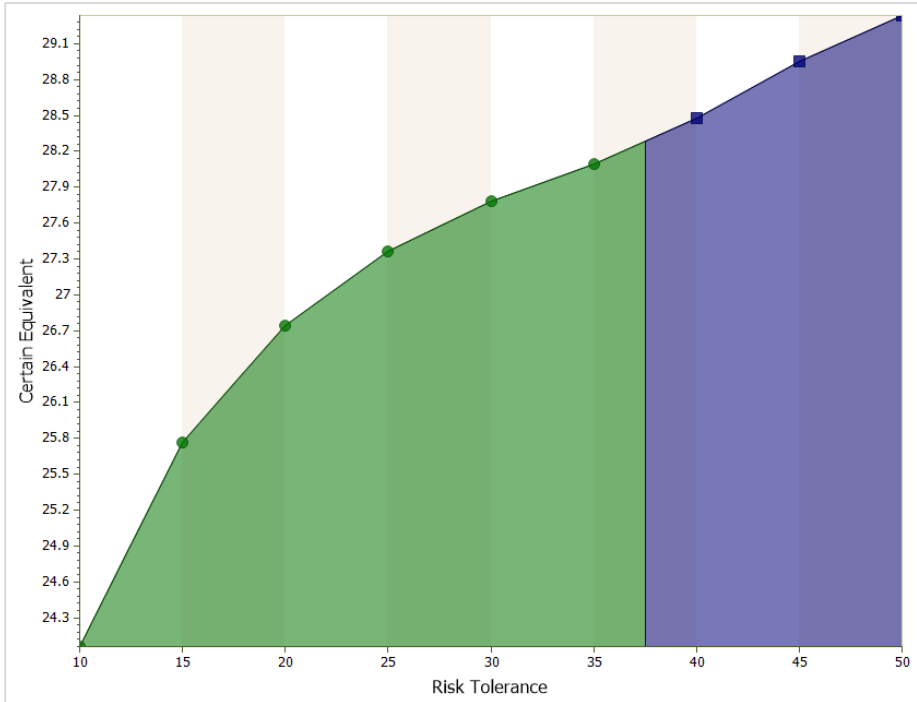
The Run Rainbow Diagram dialog should look like Figure 13-6.



**Figure 13-6. Run Rainbow Diagram dialog**

⇒ Click OK.

The rainbow diagram in Figure 13-7 shows that the optimal decision policy would change if your risk tolerance dropped to 35. The precise threshold is between 35 and 40. You could run another Rainbow Diagram with a smaller step size if you need to know more precisely the value at which the policy changes.

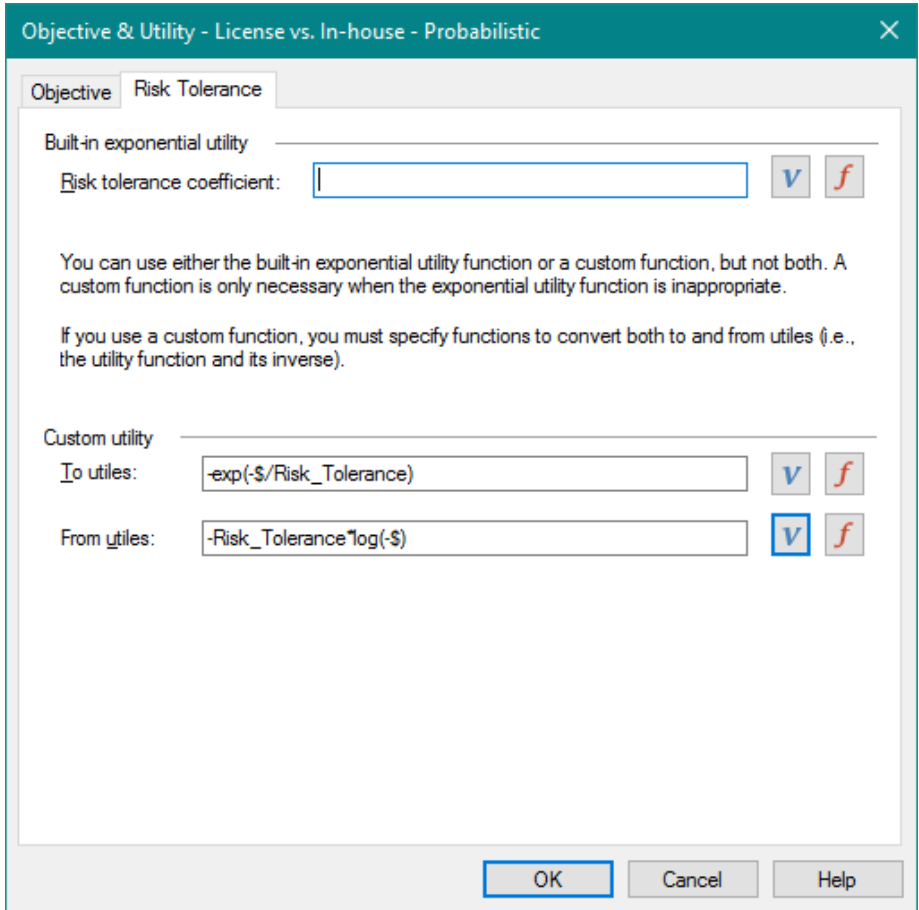


**Figure 13-7. Rainbow Diagram on Risk Tolerance**

## 13.2 Advanced Utility Functions

If you want to specify a utility function other than the standard exponential, you can do so within the Risk Tolerance tab of the Objective & Utility dialog (Objective & Utility | Settings). You need to specify both the function and its inverse, using the objective function variable "\$". Utility functions cannot depend on the state of events although they can (and generally should) depend on or use constant values such as Risk Tolerance. Figure 13-8 shows the definition of a user-defined utility function equivalent to the built-in exponential utility function.





**Figure 13-8. User-defined Utility Function**

Note: Most applications that require modeling risk aversion can be done so by specifying a single risk tolerance or utility function for the entire Decision Tree. However, there is a theoretical argument for changing the degree of risk aversion throughout the tree as the company's situation changes. DPL provides this capability, although it is hidden by default since it is seldom used. To use this feature, check the checkbox next to *Risk tolerance and utility functions by branch* within the Decision Tree tab of the Model Settings dialog (Decision Tree | Model | Settings) under the *Advanced Setting* section. The Branch Definition dialog will then have a Risk Tolerance tab on which you can set either a risk tolerance or a custom utility function from that point forward in the subtree. The risk tolerance/custom utility function applies until another one is specified further down the subtree or until the end of the subtree is reached.

## 14. Advanced Sensitivity Analyses

Five types of sensitivity analyses (Value Tornado, Base Case Tornado, Probabilistic Base Case Tornado, Initial Decision Alternatives Tornado, and Rainbow Diagram) were discussed in Chapter 5 and other earlier chapters. This chapter covers two further types of slightly more advanced sensitivity analyses that DPL offers and provides some guidance on when it is most appropriate to use each of DPL's sensitivity analysis outputs.

### 14.1 Two-Way Rainbow Diagrams

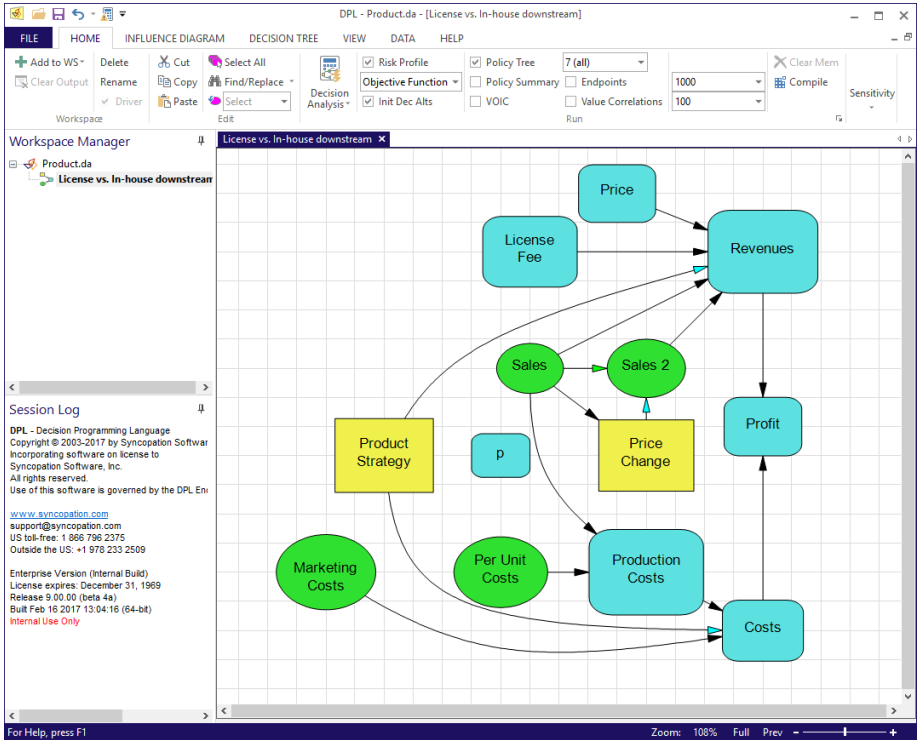
---

A Two-Way Rainbow Diagram is a sensitivity analysis that is run on two selected values from a model. The diagram displays the policy changes for each combination of the two values selected. In addition, you can use the Show Tips feature to see the change in the objective function of the model for each combination of the two values.

You will now run a Two-Way Rainbow Diagram.

- ⇒ Select File | Open.
- ⇒ Navigate to the Examples folder underneath the DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples.
- ⇒ Select Product.da.

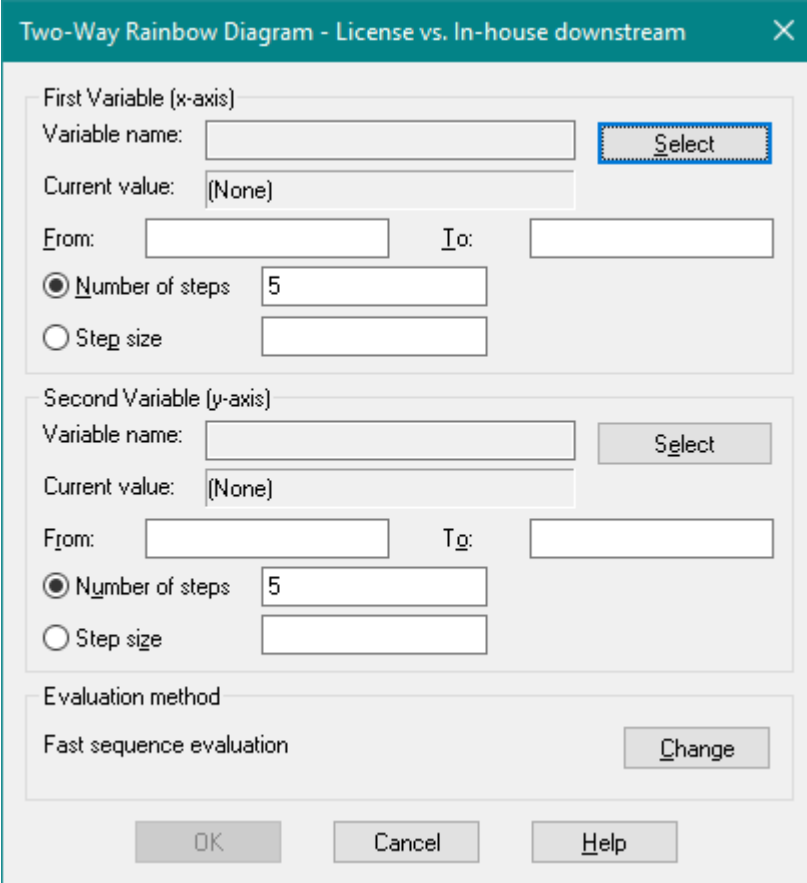
DPL opens the Workspace as shown in Figure 14-1.



**Figure 14-1. Product.da Workspace**

This model is similar in concept to the model introduced in Chapter 4.

- ⇒ Drop-down the Home | Sensitivity | Rainbow split button and select Two-Way. The Two-Way Rainbow Diagram Setup dialog appears as shown in Figure 14-2.



**Two-Way Rainbow Diagram - License vs. In-house downstream** [X]

**First Variable (x-axis)**

Variable name:

Current value:

From:  To:

Number of steps

Step size

**Second Variable (y-axis)**

Variable name:

Current value:

From:  To:

Number of steps

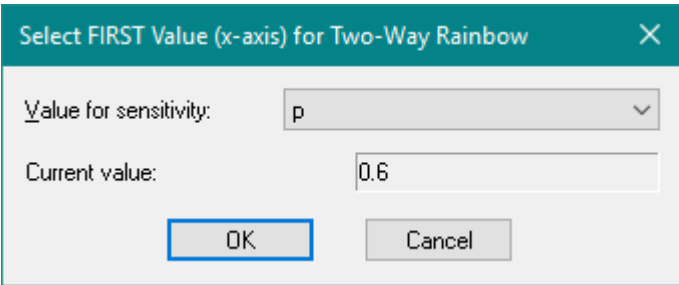
Step size

**Evaluation method**

Fast sequence evaluation

**Figure 14-2. Two-Way Rainbow Diagram Setup Dialog**

- ⇒ Click the Select button in the *First Variable (x-axis)* section. The Select FIRST Value dialog appears as shown in Figure 14-3.



**Select FIRST Value (x-axis) for Two-Way Rainbow** [X]

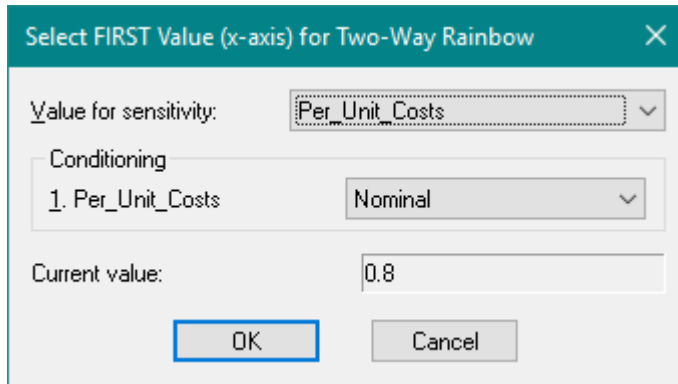
Value for sensitivity:  [v]

Current value:

**Figure 14-3. Select Value Dialog for Two-Way Rainbow**

⇒ Select Per\_Unit\_Costs in the Value for sensitivity drop-down list.

The Select Value dialog updates to show you that the value Per\_Unit\_Costs is conditioned as shown in Figure 14-4. In this instance, Per Unit Costs is a discrete chance node that has probability and value data. Therefore the value for Per\_Unit\_Costs is conditioned by which state (or outcome) the Per Unit Costs discrete chance node is in. To run a Two-Way Rainbow Diagram on a value that is conditioned, you must specify the conditioning states for each conditioning event. For example, if value node C were conditioned by discrete chance node A and decision B, in the Conditioning section of the Select Value dialog you would need to specify an outcome for A and an alternative for B in order to run a Two-Way Rainbow Diagram on C.



**Figure 14-4. Select Value Dialog Displaying Conditioning for Per\_Unit\_Costs**

⇒ In the drop-down list in the *Conditioning* section, select High. The *Current Value* increases to 0.9.

⇒ Click OK.

The Two-Way Rainbow Diagram Setup dialog updates to indicate the value and the conditioning states (if any) that you have selected. DPL displays a value and its conditioning using the notation value|event.state. You have selected the Per\_Unit\_Costs value for the High state of the Per\_Unit\_Costs chance event, so DPL displays Per\_Unit\_Costs|Per\_Unit\_Costs.High in the *Variable name* label box. If the value you have chosen is conditioned by multiple events, this is indicated by separating each event.state with commas, i.e., value|event1.state, event2.state, etc.

In the *Current value* label box, DPL displays the current value for the variable given its conditioning. See Figure 14-5.

Two-Way Rainbow Diagram - License vs. In-house downstream

First Variable (x-axis)

Variable name:

Current value:

From:  To:

Number of steps

Step size

Second Variable (y-axis)

Variable name:

Current value:

From:  To:

Number of steps

Step size

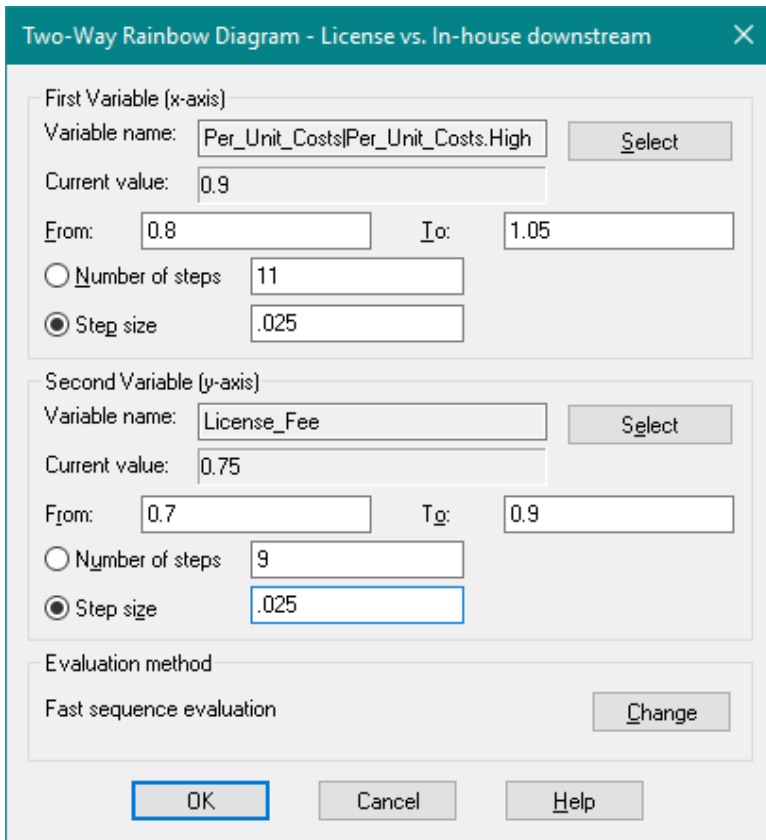
Evaluation method

Fast sequence evaluation

**Figure 14-5. Two-Way Rainbow Diagram After Selecting First Variable**

- ⇒ In the *From:* edit box in the *First Variable (x-axis)* section, type "0.8".
- ⇒ In the *To:* edit box, type "1.05".
- ⇒ In the Step Size edit box, type ".025".
- ⇒ Click the Select button in the *Second Variable (y-axis)* section. The Select Value dialog appears again.
- ⇒ Select License\_Fee in the Value for sensitivity drop-down list.
- ⇒ Click OK.
- ⇒ In the *From:* edit box in the *Second Variable (y-axis)* section, type "0.7".
- ⇒ In the *To:* edit box, type "0.9".
- ⇒ In the Step Size edit box, type ".025".

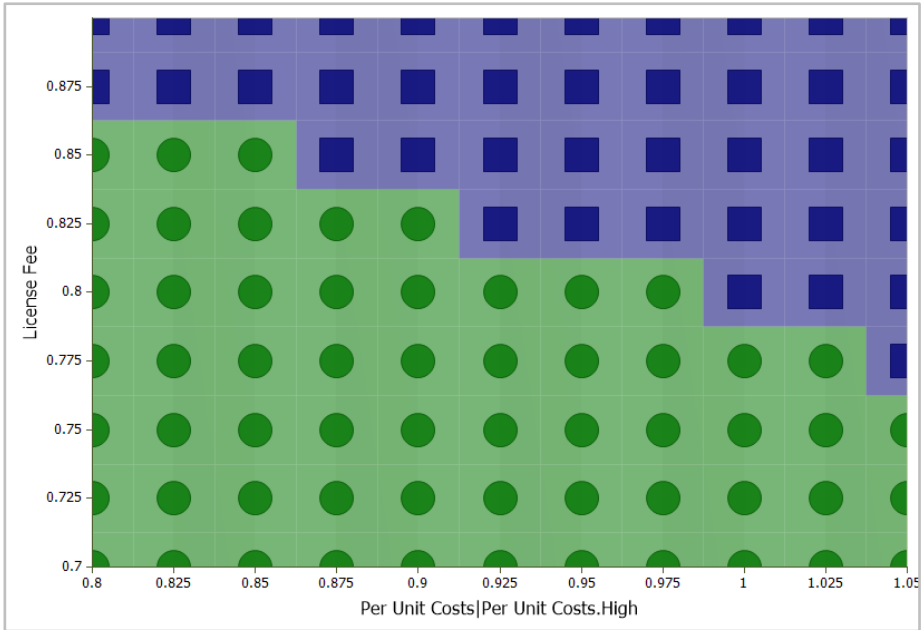
The Two-Way Rainbow Diagram Setup dialog should look like Figure 14-6.



**Figure 14-6. Run Two-Way Rainbow Dialog after Specifying Values**

⇒ Click OK to run the Two-Way Rainbow Diagram.

DPL produces the Two-Way Rainbow shown in Figure 14-7. DPL creates a Two-Way Rainbow Diagram by running the model for each combination of the two values selected for the diagram. To create the Two-Way Rainbow Diagram in Figure 14-7, DPL ran the model 99 times.



**Figure 14-7. Two-Way Rainbow Diagram**

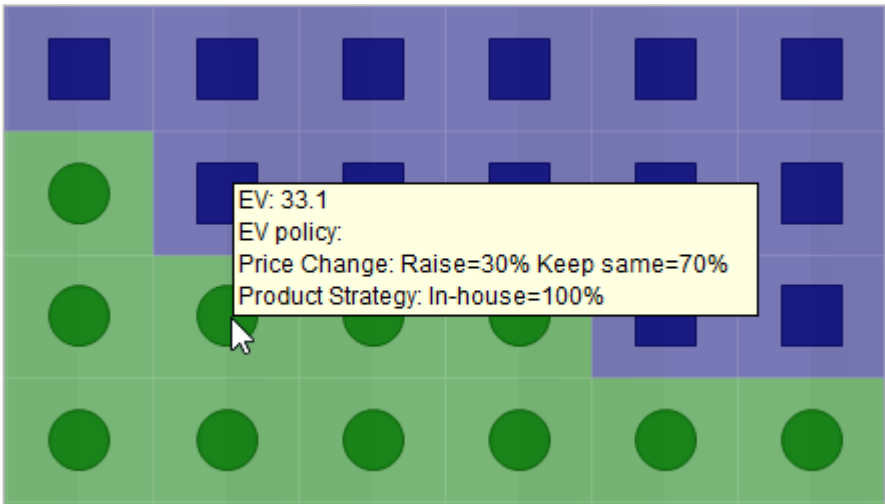
A Two-Way Rainbow Diagram displays policies using different colors and marker types for different policies at the x- and y-values on the axes. The policy at all points with the same color and marker type is the same. A color/marker change indicates a change in policy. The Two-Way Rainbow Diagram in Figure 14-7 indicates that the optimal policy depends on both the value of `Per_Unit_Costs.High` and `License_Fee`. This can be seen because the change in policy as `License_Fee` increases occurs at lower and lower values as `Per_Unit_Costs.High` increases. For example, when `Per_Unit_Costs.High` is 0.8, 0.825, or 0.85, a change in policy occurs as `License_Fee` increases from 0.85 to 0.875, but when `Per_Unit_Costs.High` is 0.875, a change in policy occurs as `License_Fee` increases from 0.825 to 0.85.

To see which policy is optimal at each point and what the expected value of the model is at each point in the Two-Way Rainbow Diagram, you must have Show Tips turned on.

- ⇒ To check if you have Show Tips turned on navigate to the View tab. If the Show button within the Tips group is shaded blue, Show Tips is turned on.
- ⇒ Move your mouse over one of the markers in the Two-Way Rainbow Diagram.



A policy tip appears as shown in Figure 14-8.



**Figure 14-8. Two-Way Rainbow Diagram with Policy Tip**

The policy tip tells you the expected value of the model given the values of the two variables that the marker represents. For example, the policy tip in Figure 14-8 indicates that the expected value of the model is 33.1 when License\_Fee is 0.8 and Per\_Unit\_Cost.High is 0.925. The policy tip also tells you the policy dependent probabilities for each decision in the model given the values of the variables that the marker represents. The decision alternatives are displayed in the policy tip by using the decision node name followed by a colon followed by the decision alternative. E.g., "Product Strategy: In-house".

The policy dependent probability for the alternative follows the equal sign. As Figure 14-8 indicates, the policy when License\_Fee is 0.8 and Per\_Unit\_Cost.High is 0.925 is to produce the product in-house and downstream to raise the price in 30% of the scenarios and to keep price the same in 70% of the scenarios. To keep the policy tip compact, decision alternatives with a policy dependent probability of zero are not displayed.

The information displayed in the policy tip is also available in the Session Log for all the points.

- ⇒ Make the Workspace Window wider by dragging its edge and the Session Log longer by dragging the splitter between the two panes.

The expected value of the objective function and the policy information is written to the Session Log for each point. See Figure 14-9.

Session Log		
Per_Unit_Costs = 0.925	License_Fee = 0.700	Expected value = 33.138
EV:		
Price_Change: Raise: 30% Keep_same: 70% (does not occur): 0%		
Product_Strategy: In_house: 100% License: 0% (does not occur): 0%		
-----		
Per_Unit_Costs = 0.925	License_Fee = 0.725	Expected value = 33.138
EV:		
Price_Change: Raise: 30% Keep_same: 70% (does not occur): 0%		
Product_Strategy: In_house: 100% License: 0% (does not occur): 0%		
-----		
Per_Unit_Costs = 0.925	License_Fee = 0.750	Expected value = 33.138
EV:		
Price_Change: Raise: 30% Keep_same: 70% (does not occur): 0%		
Product_Strategy: In_house: 100% License: 0% (does not occur): 0%		
-----		
Per_Unit_Costs = 0.925	License_Fee = 0.775	Expected value = 33.138
EV:		
Price_Change: Raise: 30% Keep_same: 70% (does not occur): 0%		
Product_Strategy: In_house: 100% License: 0% (does not occur): 0%		
-----		
Per_Unit_Costs = 0.925	License_Fee = 0.800	Expected value = 33.138
EV:		
Price_Change: Raise: 30% Keep_same: 70% (does not occur): 0%		
Product_Strategy: In_house: 100% License: 0% (does not occur): 0%		
-----		
Per_Unit_Costs = 0.925	License_Fee = 0.825	Expected value = 33.248
EV:		
Price_Change: Raise: 0% Keep_same: 100% (does not occur): 0%		
Product_Strategy: In_house: 0% License: 100% (does not occur): 0%		

**Figure 14-9. Session Log with Policy Information from Two-Way Rainbow Diagram**

⇒ Return the Session Log to its default size.

## 14.2 Event Tornadoes

Event tornadoes provide information about how much each chance node in a model contributes to the overall uncertainty in the outcome of the objective function of the model. There are two types of Event Tornadoes in DPL: Deterministic and Probabilistic.

DPL creates a Deterministic Event Tornado by running the model once to establish the minimum amount of uncertainty in the model (called the Minimum Model Range). DPL calculates the Minimum Model Range by replacing all the chance events in the model with their expected values. Then, DPL runs the model one additional time for each chance event in the model (call this event the sensitivity event). In each of these subsequent runs, DPL replaces all of the chance events in the model except the sensitivity event with their expected values. It leaves the sensitivity event unchanged. In effect, in each sensitivity event model run the only uncertainty left in the model is the uncertainty associated with the sensitivity event (unless Always Gamble has been set for chance events, see below).

DPL follows an analogous procedure to create a Probabilistic Event Tornado. DPL runs the model once making no changes to the model to establish the original amount of uncertainty in the model (called the Original Model Range). Then, DPL runs the model one additional time for each chance event in the model (the sensitivity event). However, for Probabilistic Event Tornadoes, in these subsequent runs DPL leaves all of the chance events in the model unchanged except for the sensitivity event. The sensitivity event is replaced with its expected value. In effect, in each sensitivity event model run the reduction in uncertainty from the Original Model Range is due to the removal of the uncertainty associated with the sensitivity event.

If either Don't Gamble or Always Gamble is set for a chance event, this has an impact on how the Event Tornado is run. The Don't Gamble and Always Gamble checkboxes are found under the *Special handling of this chance* section on the Tree Instance tab of the Node Definition dialog. To access the Tree Instance tab for a chance node, double-click the chance node in the Decision Tree

For Probabilistic and Deterministic Event Tornadoes, Don't Gamble and Always Gamble have the following effects. If Don't Gamble is set for a chance event, then it is not gambled on regardless of whether or not it is the sensitivity event (i.e., it is always replaced with its expected value in the Event Tornado). If Always Gamble is set for a chance event, then it is gambled on regardless of whether or not it is the sensitivity event (i.e., it is never replaced with its expected value in the Event Tornado).

Note: because a Deterministic Event Tornado replaces chance events with their expected value, it may not be appropriate to use for models that use a risk tolerance and hence have certain equivalents. For such models, consider using a Probabilistic Event Tornado.

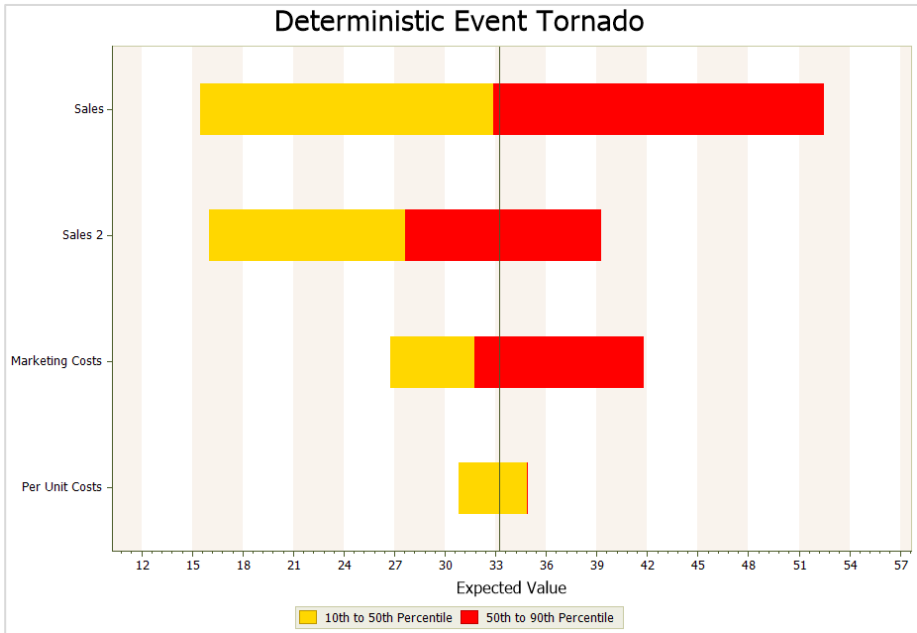
You will now run a Deterministic Event Tornado.

⇒ If Product.da is not already open, select File | Open.

This file is found in the *Examples* folder where DPL was installed, usually C:\Program Files\Syncopation\DPL9\Examples.

⇒ In the Home | Sensitivity group, drop-down the Tornado split button and select Event (Deterministic) from the list.

DPL produces the Deterministic Event Tornado shown in Figure 14-10.



**Figure 14-10. Deterministic Event Tornado**

The width of each bar indicates the difference between the 10th percentile and the 90th percentile of the risk profile of the objective function for the model when the sensitivity event labeling the bar is the only uncertainty in the model (unless chance events have been set to *Always Gamble*). The change in color occurs at the 50th percentile. The bar is yellow between the 10th and the 50th percentiles, and red between the 50th and 90th percentiles. The black vertical line running through all the bars indicates the expected value established in the run for the Minimum Model Range.

A bar may not change color if the risk profile is highly skewed for a particular chance event. For example, if the 50th percentile were equal to the 90th percentile, then the entire bar would be yellow.

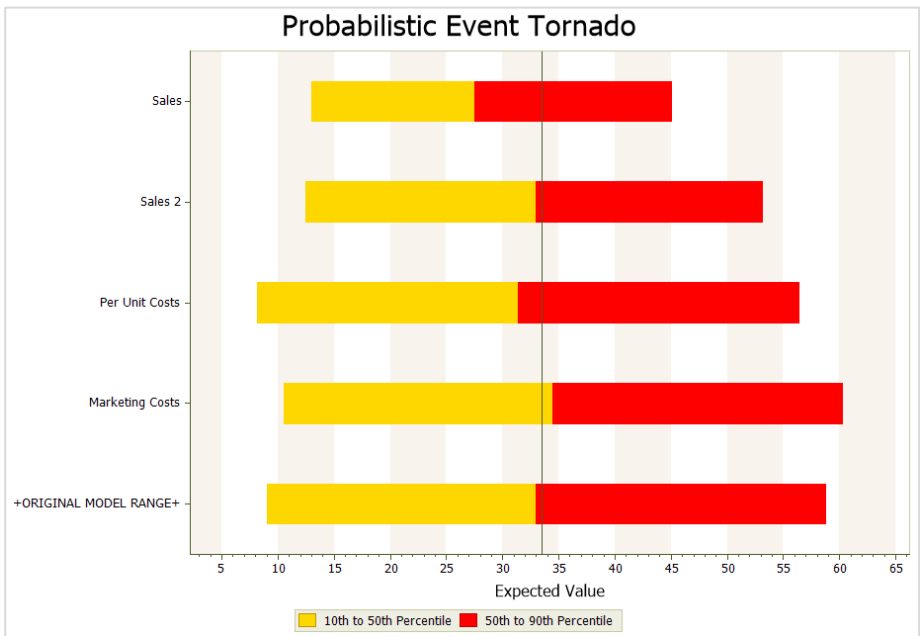
The narrower a bar is for a chance event in a Deterministic Event Tornado, the less impact the chance event has on the overall uncertainty in a model. Bars nearer the top of a Deterministic Event Tornado contribute more to the overall uncertainty in a model.

If a chance event appears in a Deterministic Event Tornado with a "bar" of no width, then the 10th and 90th percentiles are equal. The most likely reason for this is that Don't Gamble has been set for the event, though it could be because the 10th and 90th percentiles are truly equal. The latter may happen in an asymmetric tree when an uncertainty only occurs for a particular decision alternative, and that alternative is never optimal.

You will now run a Probabilistic Event Tornado.

⇒ In the Home | Sensitivity group, drop-down the Tornado split button again. This time, select Event (Probabilistic) from the list.

DPL produces the Probabilistic Event Tornado shown in Figure 14-11.



**Figure 14-11. Probabilistic Event Tornado**

The width of each bar indicates the difference between the 10th percentile and the 90th percentile of the risk profile of the objective function for the model when the sensitivity event labeling the bar is the only uncertainty removed from the model (unless other chance events have been set to

Don't Gamble). The change in color occurs at the 50th percentile. The bar is yellow between the 10th and the 50th percentiles; red between the 50th and 90th percentiles. The black vertical line running through all the bars indicates the expected value established in the run for the Original Model Range.

As in a Deterministic Event Tornado, a bar may not change color if the risk profile is highly skewed for the model with the sensitivity event replaced by its expected value.

The narrower a bar is for a chance event in a Probabilistic Event Tornado, the greater the reduction in uncertainty is due to the "removal" of the chance event. Bars nearer the top of a Probabilistic Event Tornado contribute more to the overall uncertainty in a model.

## 14.3 When to Use Which DPL™ Sensitivity Output

---

Table 14-1 and Table 14-2 provide a summary of when it is most appropriate to use each of DPL's sensitivity outputs as well as when to be careful using a particular sensitivity output.

<b><u>Output</u></b>	<b><u>Particularly good for:</u></b>	<b><u>Things to be aware of:</u></b>
<i>Value Tornado</i>	Determining which values in a deterministic model should be treated as uncertainties, understanding the probabilistic impact of changes to values in probabilistic models.	
<i>Base Case Tornado</i>	Understanding the deterministic impact of chance events, comparing the deterministic impact of changes to values to chance events.	Can be difficult to interpret for highly asymmetric models. All conditioning and probabilities are ignored. Results are sensitive to the definition of the Nominal state for each chance event.
<i>Probabilistic Base Case Tornado</i>	Comparing to other results since the base run (i.e. vertical line) will equal the expected value of the model as in a Risk Profile or a Policy Tree. Ease of explanation.	Takes longer to run.
<i>Deterministic Event Tornado</i>	Understanding the probabilistic contribution of each chance event to overall model uncertainty. Understanding the impact of a chance event while preserving conditioning. Useful for models with a lot of conditioning.	Inappropriate to use for models with risk tolerance.
<i>Probabilistic Event Tornado</i>	Same as Deterministic Event Tornado.	Appropriate to use for models with risk tolerance. Can be difficult to explain.

**Table 14-1. Summary of When to Use Each of DPL's Tornado Diagrams**

<b>Output</b>	<b>Particularly good for:</b>
<i>Rainbow Diagram</i>	Understanding the impact on the objective function of varying a single value across a number of settings, determining if policy changes occur as a result of varying a single value across a number of settings
<i>Two-way Rainbow Diagram</i>	Understanding how the interaction between two variable impacts the objective function, determining if policy changes occur as a result of varying two values simultaneously.

**Table 14-2. Summary of When to Use Each of DPL's Rainbow Diagrams**

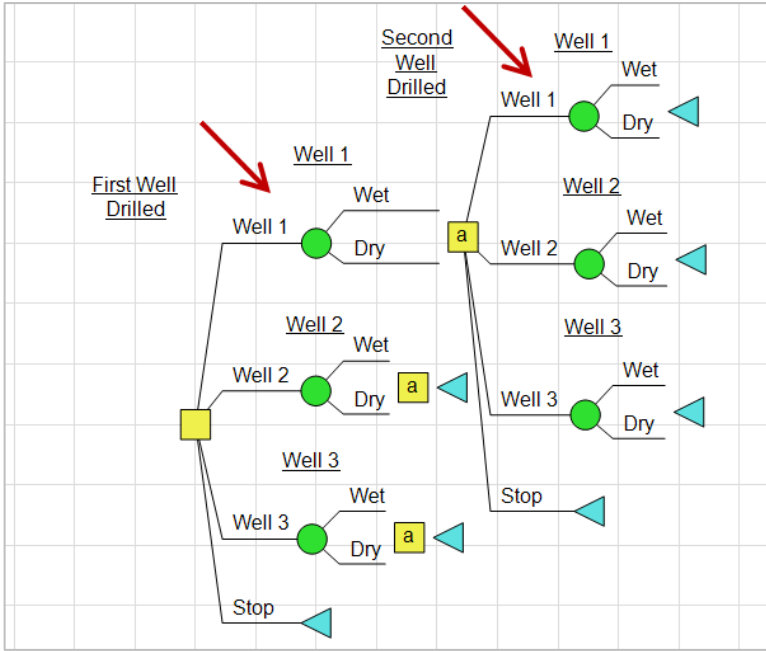


## 15. Pruned Sequential Asymmetric Trees (Enterprise only)

At times decision problems lend themselves to a sequential structure that can make them difficult to model with standard Decision Trees. An example of this is a common problem within the oil & gas industry: deciding how best to explore an oil field. Typically there is a cluster of potential well sites that can be thought of as a highly dependent set of uncertainties. If you drill a well at one site and strike oil, it's more likely that a nearby site will also have oil. If you drill a "dry hole" or two, you'll probably give up on the field rather than spend more money drilling wells that increasingly look like poor prospects. This makes intuitive sense but the many possible states of wells drilled can make the modeling tedious.

The difficulty stems from the fact that in Decision Tree modeling you normally aren't allowed to have the same event (decision or chance) more than once on any path in the tree. But if you were to consider the problem described above, at each decision you should be able to pick from among all the wells except the one you've already drilled.

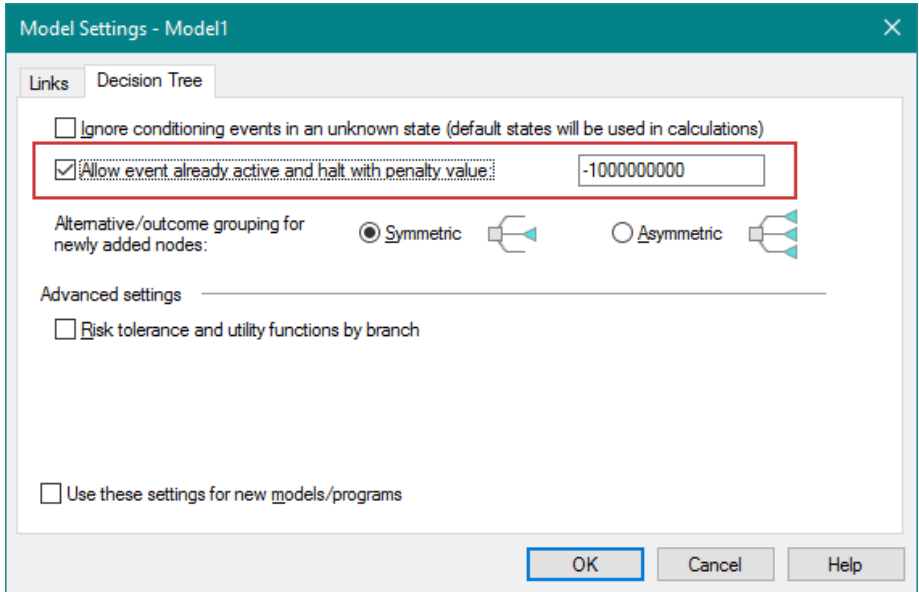
In Figure 15-1, consider the two instances of Well 1 in the tree (red arrows). All of the paths leading to the second instance (right red arrow) have already encountered the first instance of Well 1 (left red arrow), so it's already in a known state on those paths – therefore it doesn't make sense for it to be uncertain again. You could work around this by adding another decision (say, "Second Well Drilled given that Well 1 was First"), but that would quickly become impractical as the number of wells increased. The Enterprise version of DPL includes a feature that allows you to model these types of sequential problems more adeptly.



**Figure 15-1. Three Well Sequential Drilling Decision Tree Example**

To model these sequential Decision Tree problems, first turn on the *Allow event already active and halt with penalty value* (Figure 15-2) setting via Decision Tree | Model | Settings. We also refer to this as the "pruned sequential tree" feature. With this setting turned on DPL will stop evaluating a path in the tree (i.e., halt) if it encounters the same event twice, and will apply a user-defined penalty value (typically a large negative number) at that point. This serves to automatically filter out paths that don't make sense (e.g., drilling Well 1 twice).

Note: This is a DPL Enterprise feature. If you have the DPL Professional version this setting will not be available within the Model Settings dialog and you will not be able to complete this tutorial. If you're interested in this feature please contact Syncopation's Sales Team at [sales@syncopation.com](mailto:sales@syncopation.com) to discuss feature upgrade options.

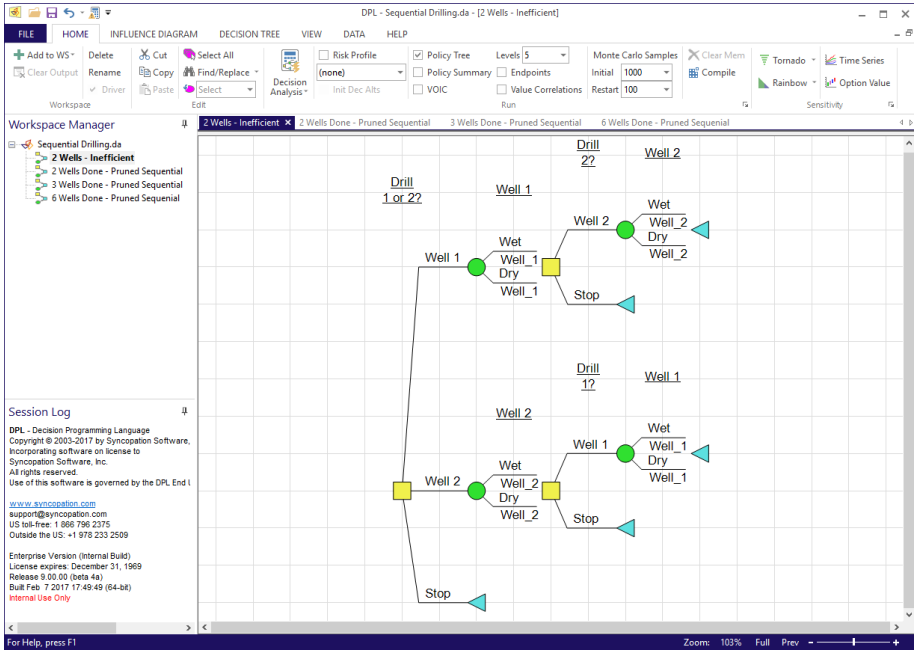


**Figure 15-2. Allow Event Already Active Setting in Modeling Settings | Decision Tree**

## 15.1 Tutorial: Modeling Sequential Drill Decisions

In the tutorial that follows, you will make modifications to a model initially structured in a way that does not utilize the pruned sequential tree feature. You'll make the edits necessary, turn on the Allow event already active setting, and generate results for comparison with those generated from the original model structure.

- ⇒ Start DPL.
- ⇒ Open the file Sequential Drilling.da from the Examples folder within your DPL installation directory (see Figure 15-3).

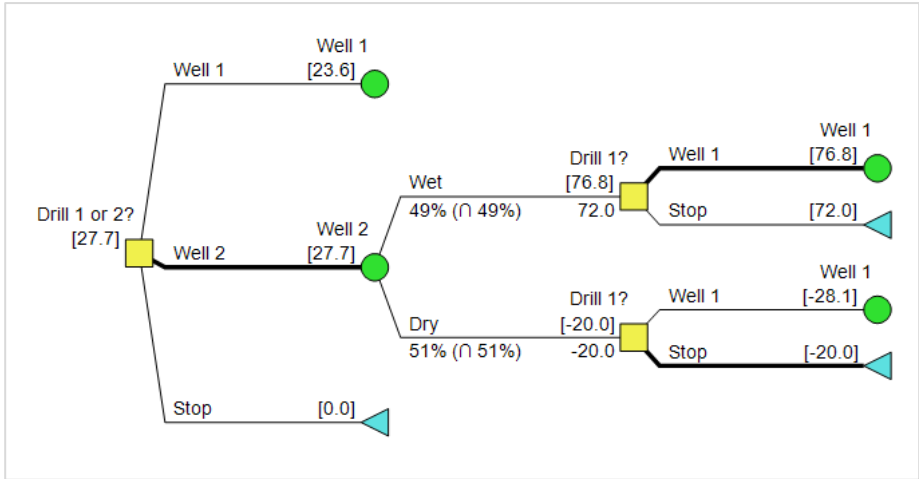


**Figure 15-3. Inefficient 2-Well Sequential Drilling Model**

There are four models within the Workspace. The "2 Wells – Inefficient" model should be active in the Model Window. This model analyzes a 2-well sequential drilling decision and is structured in a way that will work without the pruned sequential tree setting turned on. For two wells, the model isn't overly complex but with more wells this method of modeling grows onerous very quickly.

Let's run the model and generate a Policy Tree.

- ⇒ Within the Home | Run group make sure Policy Tree is the only output selected.
- ⇒ Click Home | Run | Decision Analysis (or press F10). DPL will run the analysis and activate the Policy Tree (Figure 15-4).



**Figure 15-4. Policy Tree™ for Two Well Sequential Drilling Model**

You can see that the optimal decision policy is to drill Well 2 first. If you find oil, then you should proceed with drilling Well 1. If Well 2 is found to be dry then you stop there. The overall Expected Value of the model is 27.7.

Now you will modify this two well model in order to take advantage of the pruned sequential tree setting. The modification will result in a model that is more efficient, compact, and scalable.

- ⇒ Activate the 2 Wells – Inefficient model by double-clicking on its item in the Workspace Manager (or press Ctrl+F12).
- ⇒ Press the Tab key to switch to the Influence Diagram.
- ⇒ Select both the Drill 1? And Drill 2? decision nodes and press the Delete key to delete both nodes.

DPL will warn you that references to these nodes will be removed from the Decision Tree as well, which is fine.

- ⇒ Click OK to the prompts.
- ⇒ Double-click the Drill 1 or 2? decision to edit it.
- ⇒ On the General tab, re-name the node to be "First Drill" and click OK to close the Node Definition dialog.
- ⇒ Select First Drill node and press Ctrl+C to copy the node.
- ⇒ Press Ctrl+V to paste the node. Double-click the pasted node to edit it.
- ⇒ On the General tab, name the node "Second Drill".

⇒ Click OK to close the Node Definition dialog.

With the pruned sequential tree setting you're able to model this particular decision-problem with only two asymmetric decision nodes instead of three.

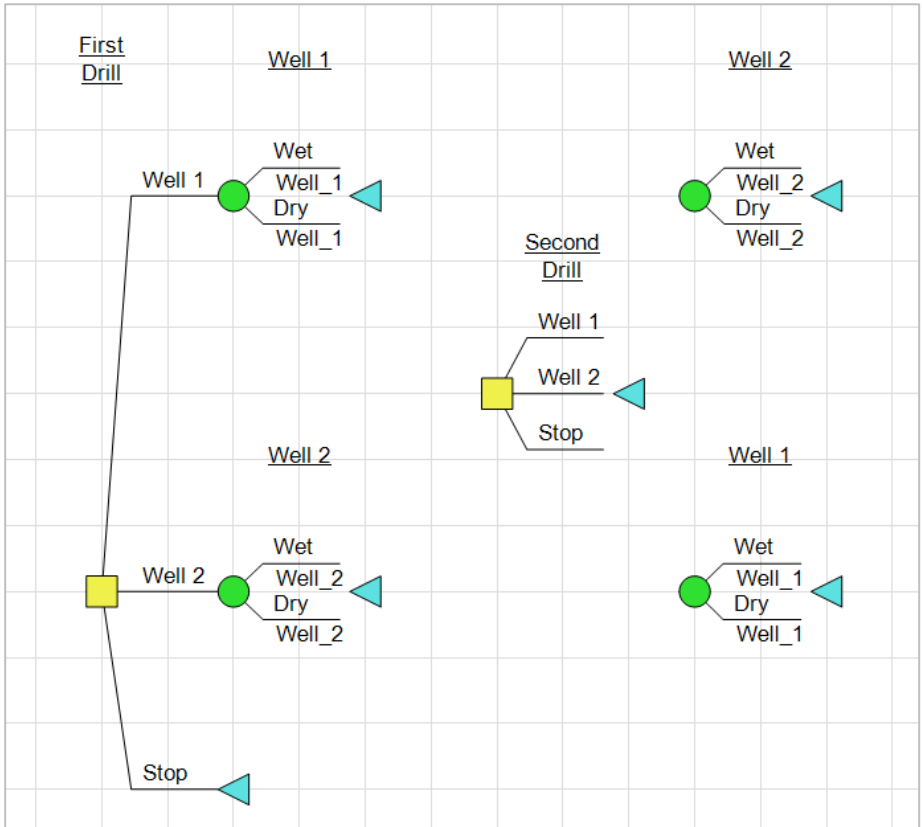
⇒ Press the Tab key to switch to the Decision Tree.

⇒ Select Decision Tree | Instance | Add | Decision.

⇒ Within the Select Decision dialog, select Second Drill and click OK.

- ⇒ Place the node in whitespace within the Decision Tree pane as shown in Figure 15-5.

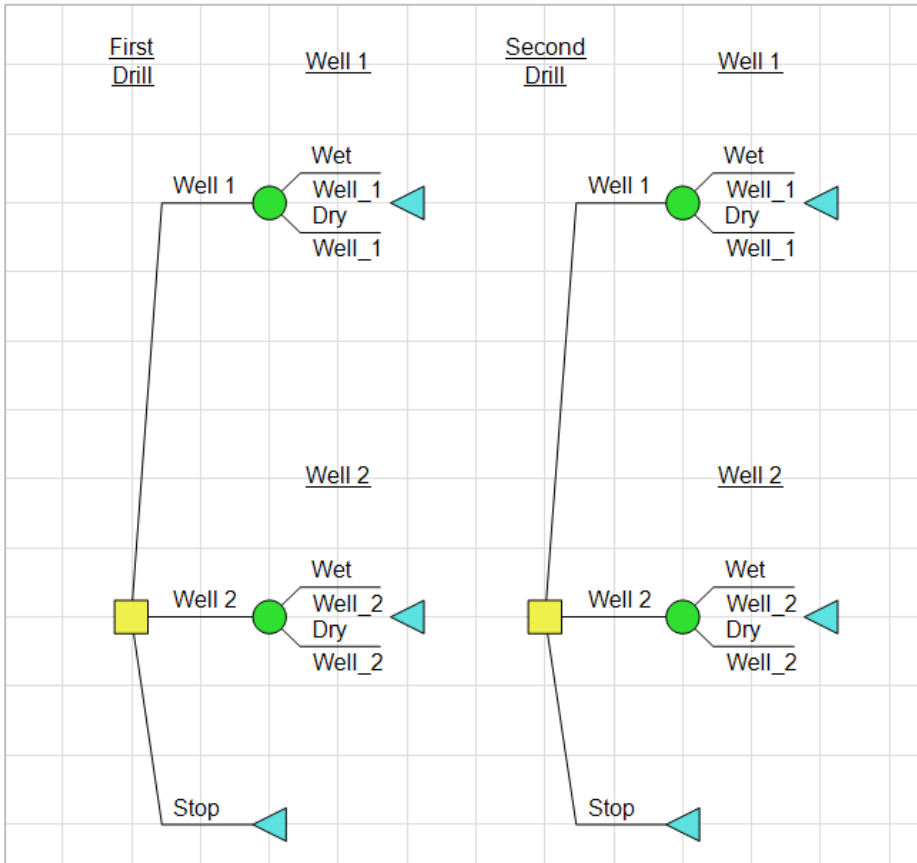
Note that when you deleted the two decisions from the Influence Diagram, they were also deleted from the Decision Tree. The subsequent Well 1 and Well 2 chance nodes remain but are detached from the tree. You'll need to first make the Second Drill decision asymmetric prior to reconnecting the chance nodes.



**Figure 15-5. Decision Tree with Second Drill Decision Added and Detached Nodes**

- ⇒ Select the branches of the Second Drill decision node and check the Asymmetric box within Decision Tree | Instance.
- ⇒ Move the Second Drill decision and blue endpoints so that each aligns with those of the First Drill decision (Figure 15-6).

- ⇒ Drag and drop the Well 1 chance node onto the blue endpoint node for the Well 1 alternative of the Second Drill decision.
- ⇒ Drag and drop the Well 2 chance node onto the blue endpoint node for the Well 2 alternative of the Second Drill decision. Your Decision Tree should now match Figure 15-6.

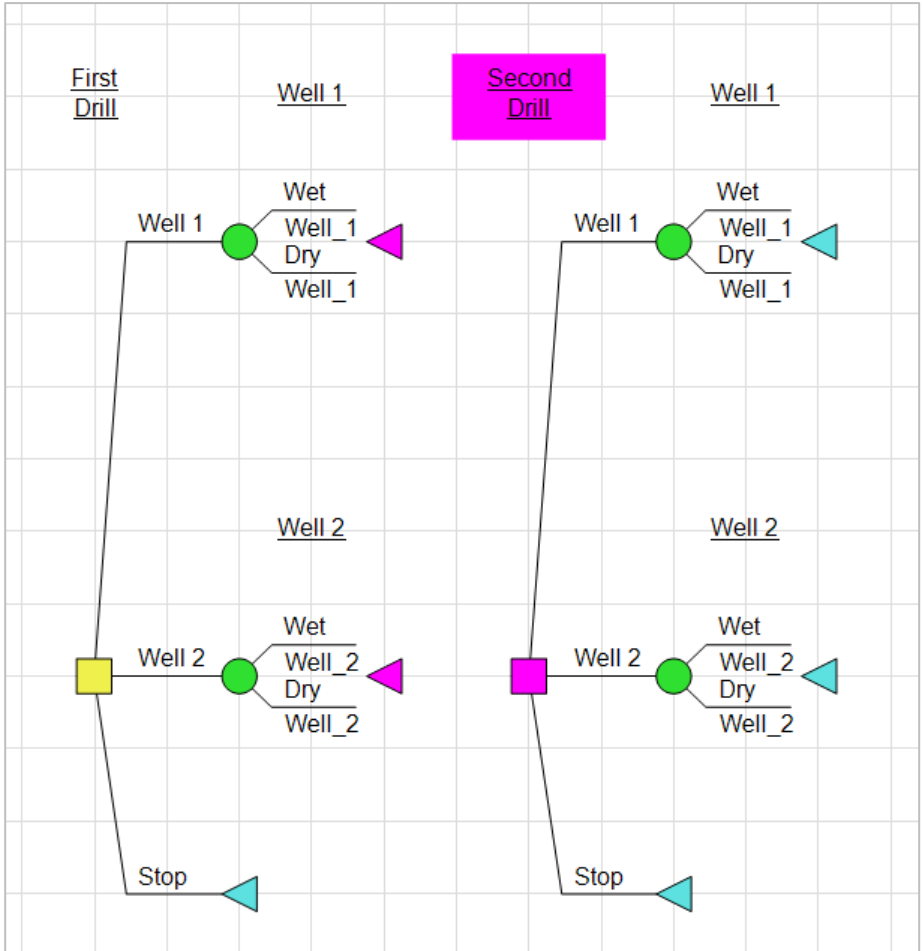


**Figure 15-6. Decision Tree with Chance Nodes Attached to Second Drill Decision**

You'll now use DPL's perform subtree feature to copy the subtree you just created that starts at Second Drill to the blue endpoint nodes of the Well 1 and Well 2 chance nodes after the First Drill decision.

- ⇒ Select the Second Drill decision node and the blue endpoints for the Well 1 and Well 2 chance node for the First Drill decision all at once using Ctrl+Click as shown in Figure 15-7.





**Figure 15-7. Decision Tree with Items Selected for Performing a Subtree**

⇒ Click the Decision Tree | Instance | Subtree button.

DPL create a perform subtree reference at both endpoints and a perform subtree target at Second Drill. At this point the model structure is complete. The only thing left to do is to turn on the *Allow event already active and halt with penalty value* setting. But before doing so you'll run the model with its current structure and settings to see what happens with it turned off.

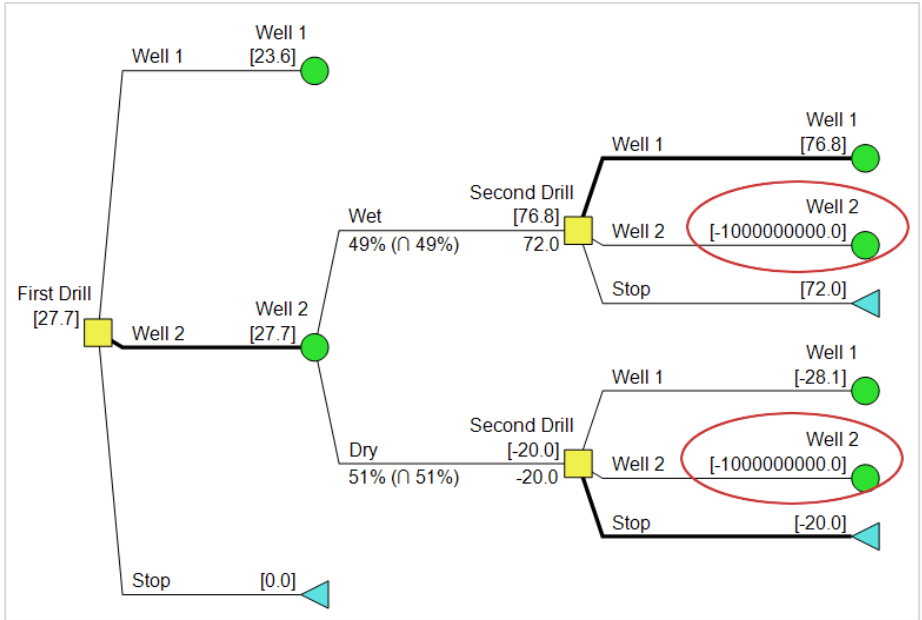
⇒ Make sure Policy Tree is the only output requested within the Home | Run group.

- ⇒ Click the Home | Run | Decision Analysis button (or press F10).
- ⇒ Click OK to the clear output prompt.

You will receive a warning from DPL that states "Event already active in earlier lottery/decision." As mentioned before, without the Allow Event Already Active setting on DPL will stop analyzing the tree when it encounters the same event (chance or decision), in this case Well 1, twice on the same path.

- ⇒ Click OK to the warning.
- ⇒ Select Decision Tree | Model | Settings.
- ⇒ On the Decision Tree tab, check the *Allow event already active and halt with penalty value* box.
- ⇒ Click OK to close the dialog.
- ⇒ Run the model again (Home | Run | Decision Analysis or F10) to generate a Policy Tree.

DPL will run the model and display the Policy Tree (Figure 15-8). Note that the tree is nearly identical to the one generated at the beginning of the section (Figure 15-4). The only difference being the expected values circled within the Figure 15-8. The halt penalty has been applied to the Well 2 alternative of the Second Drill decision due to the fact that Well 2 has already been drilled, i.e., it is the optimal alternative for the First Drill decision and so should not be considered in the Second Drill decision. The pruned sequential tree setting only provides a small amount of modelling convenience in this simple two well case but in a situation with three or more wells (which is almost always the case) this feature saves a lot of time and effort and results in much more manageable Decision Trees.



**Figure 15-8. Policy Tree™ for Pruned Sequential Tree**

Note that the model you completed in the section should match the "2 Wells Done – Pruned Sequential" model contained within the Workspace. There is also an example of a 3-well and 6-well case that is set up to employ the pruned sequential tree setting. These examples models are based on data derived from a paper by Bickel & Smith (2006) on the subject of optimizing sequential drilling decisions. (Bickel, J. Eric and James E. Smith. 2006. Optimal Sequential Exploration: A Binary Learning Model. *Decision Analysis*. 3(1), 16.)

## 16. Converting Spreadsheets to DPL™ Code

This chapter is intended mainly for experienced DPL users who have built large, Excel-linked DPL models, and who need to reduce model runtimes. This chapter discusses how to convert a spreadsheet to DPL code, provides guidance on good spreadsheet practices if you intend to convert to code, and alerts you to pitfalls to be aware of.

Chapters 2 and 4 discuss the basics of DPL-linked spreadsheets, how to build a model from Excel and how to modify it by adding linked nodes to the model. Throughout those chapters, the model uses Excel as its calculation engine for the cash flows that DPL rolls up in the Decision Tree.

DPL communicates with Excel via OLE Automation. If you have a large model, this communication protocol can be slow. Also, DPL's analytical engine is optimized for performance on Decision Trees with numerous paths, whereas Excel is designed for a single interactive recalculation. For these reasons, if you have a large model and wish to substantially reduce runtime, you may want to convert your spreadsheet to code.

### 16.1 How to Convert Spreadsheets

---

- ⇒ Open Software.da.
- ⇒ This file is found in the Examples folder where DPL was installed, usually C:\Program Files\Syncopation\DPL9\Examples.

DPL opens the Workspace as shown in Figure 16-1.

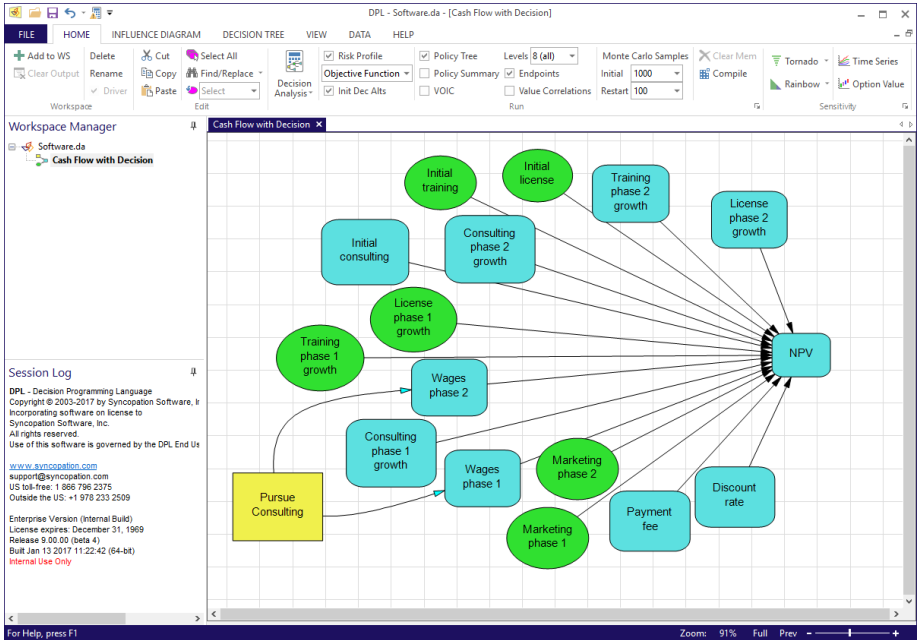


Figure 16-1. Software.da Workspace

⇒ In the Home | Run group, select Risk Profile, Init Dec Alts, Policy Tree, and Endpoints.

⇒ Run a Full Tree Enumeration for the evaluation method.

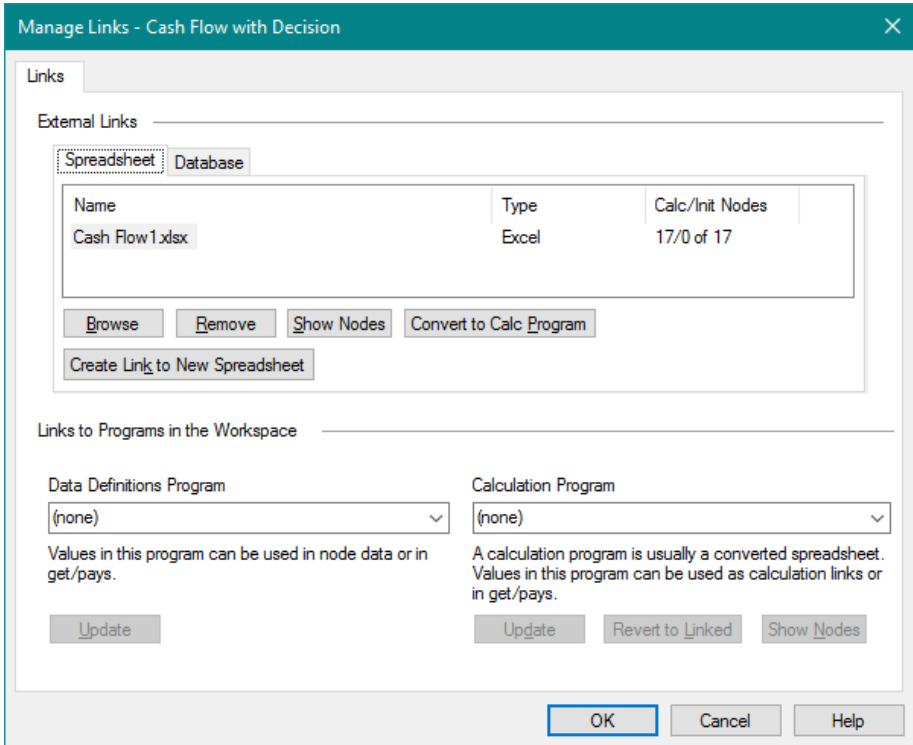
You will run the model in order to see how long it takes to run linked to the Excel spreadsheet.

⇒ After the model runs, look at the Session Log to see how long the model took to run. Near the bottom of the Session Log will be a line with "Elapsed time:". Note down the number.

⇒ Double-click the item for the model in the Workspace Manager to activate the Model Window.

The model is linked to an Excel spreadsheet called Cash Flow1.xlsx. You will convert this spreadsheet to code via the Manage Links dialog.

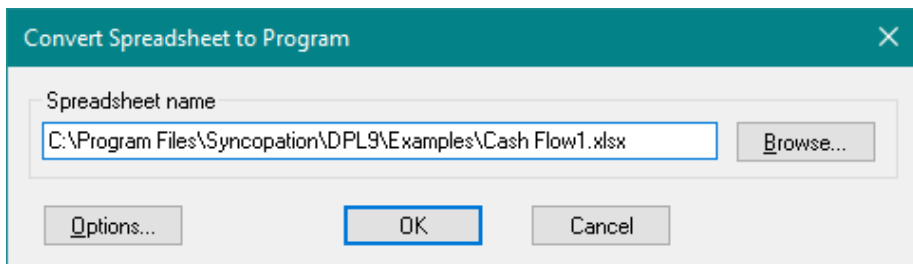
⇒ Activate the Cash Flow with Decision model and click Influence Diagram | Links | Manage. The Manage Links Dialog appears as shown in Figure 16-2.



**Figure 16-2. Manage Links Dialog**

In the *External Links* section, the Manage Links dialog displays the list of Excel spreadsheets linked to the model. In most instances, there will only be one Excel spreadsheet linked to a model. However, you can have multiple spreadsheets linked to a model. To establish a link to another spreadsheet, you can click *Create Link to New Spreadsheet*. In this instance, there is only one spreadsheet link. Note if no path is specified for the spreadsheet name then DPL assumes the spreadsheet is in the same directory as the Workspace file.

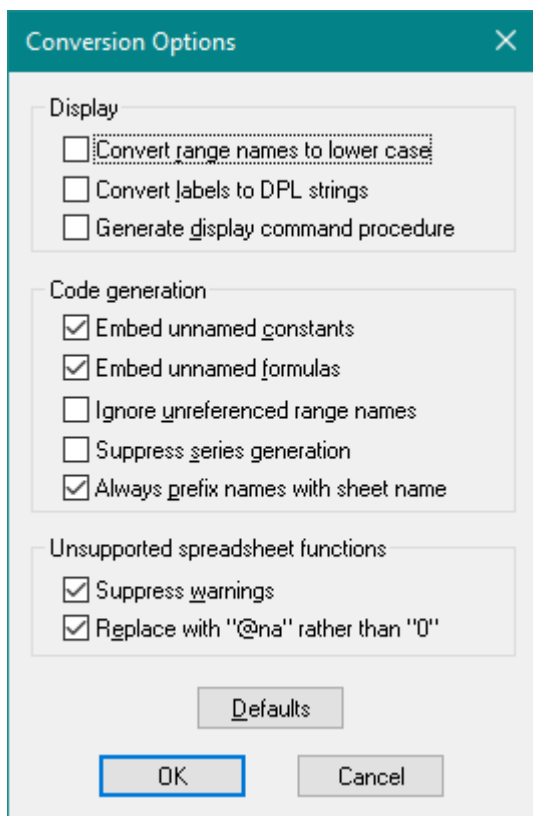
- ⇒ Make sure *Cash Flow1.xlsx* is selected in the *Linked Spreadsheets* list box.
- ⇒ Click *Convert to Calc Program* button. The *Convert Spreadsheet to Program* dialog appears as shown in Figure 16-3.



**Figure 16-3. Convert Spreadsheet to Program Dialog**

The dialog has the spreadsheet name that you are about to convert in the Spreadsheet name box. If needed, you could change the spreadsheet by clicking the Browse... button.

- ⇒ Click the Options... button. The Conversion Options dialog appears as shown in Figure 16-4.



**Figure 16-4. Conversion Options Dialog**

You will rarely need to change any of the conversion options settings. Table 16-1 through Table 16-3 describe what each option does.

<b><u>Default</u></b>	<b><u>Option</u></b>	<b><u>Description</u></b>
Off	Convert range names to lower case	Converts e.g., Growth_Rate to growth_rate. Should not be used with Convert to Calc Program from Model Links dialog.
Off	Convert labels to DPL strings	Converts cells which contain strings but are not used in calculations.
Off	Generate display command procedure	Generates statements which can be used for (advanced) debugging in the Command Window.

**Table 16-1. Display Conversion Options definitions**



On	Embed unnamed constants	Cells which contain constants used in calculations will not normally be defined as DPL values. DPL normally prefers <pre>value B3 = 1 + 2;</pre> to <pre>value B1 = 1; value B2 = 2; value B3 = B1 + B2;</pre> Uncheck this option if you want the constant values.
On	Embed unnamed formulas	As above but pertaining to formulas.
Off	Ignore unreferenced range names	Check this to have DPL ignore (i.e., not convert) cells or ranges which are named but not used in other calculations. This saves some memory but can be annoying, as output cells like "Total_NPV" are often unreferenced.
Off	Suppress series generation	DPL series are designed for fast calculation of similar formulas, as are often found in spreadsheet rows. However, a whole series (row) must be evaluated at once, so DPL series can be recursive even though the source spreadsheet has no circular references. Check this option if you see the message "series definitions involving X are recursive". DPL will generate less efficient code using values and arrays.
On	Always prefix names with sheet name	Convert e.g., NPV on the DCF sheet to DCF_NPV rather than just NPV. Leave this on when using Convert to Calc Program from the Model Links dialog so that you can Revert to Linked if necessary.

**Table 16-2. Code Generation Conversion Options definitions**

On	Suppress warnings	Uncheck to see a message box for every warning.
On	Replace with "@na" rather than "0"	Unsupported functions are normally replaced with @na. This is a safety feature: if the expression needs to be calculated, an error will stop the run. Uncheck this to have unsupported functions replaced with zero.

**Table 16-3. Unsupported Functions Conversion Options definitions**

If you change any of the options, your new settings will be used the next time you convert unless you change them again. You can click the Defaults button to revert to the default settings.

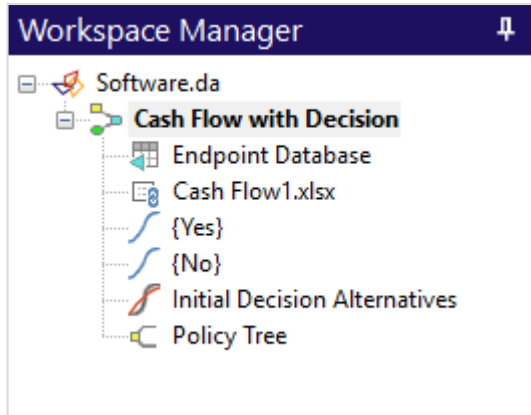
- ⇒ Leave the settings as they are.
- ⇒ Click Cancel.
- ⇒ Click OK in the Convert Spreadsheet to Program dialog.

DPL creates a DPL program file with the spreadsheet converted to DPL code in it. For large spreadsheets, this may take some time. While converting, DPL displays the Converting dialog, which allows you to cancel the process. The DPL program with the converted spreadsheet code has the same name as the spreadsheet (i.e., Cash Flow1.xlsx in this instance).

After conversion, DPL removes the spreadsheet from the Linked Spreadsheets list box and puts the name of the DPL program with the converted spreadsheet code in the *Calculation Program* edit box under the *Links to Program in the Workspace* section.

- ⇒ Click OK to close the Manage Links dialog.

Note that DPL has created an item in the Workspace Manager for the DPL program with the converted spreadsheet code. DPL places the item for the DPL program under the model that references it. See Figure 16-5. The icon for a referenced program has a link on it indicating that it is linked to the model beneath which it is displayed.



**Figure 16-5. Workspace Manager with Item for Referenced Program**

A DPL program can be linked to more than one model. If you have programs in your Workspace that are not linked to any model, they are displayed beneath a Workspace Manager item called Unreferenced programs. Programs beneath the Unreferenced programs item in the Workspace Manager are displayed with an icon that does not have a link.

- ⇒ Run the model with the converted spreadsheet. You should notice a marked improvement in runtime.
- ⇒ Look at the Session log again to see how much time was saved.
- ⇒ Look for the "Elapsed time:" entry near the bottom.

The reduction in run time is dramatic. This can be particularly useful for large models.

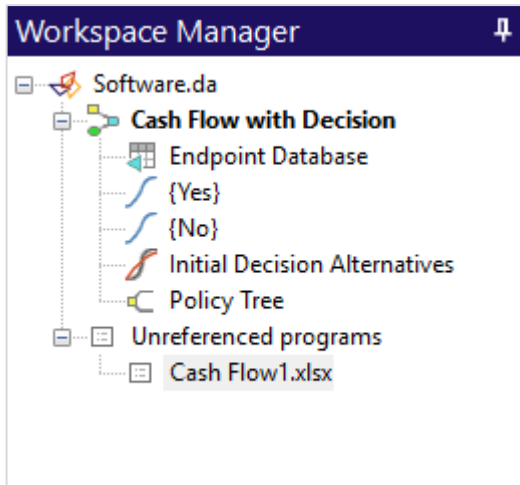
Once a linked spreadsheet model has been converted to a DPL program, subsequent changes to the converted model can only be made by changing the DPL program. See Chapter 15 for a discussion of DPL programs and code. Usually, if you plan to continue modifying the linked spreadsheet and re-running the model, you will not want to modify the converted DPL program. Instead, you can save a copy of the linked model (either as a separate .da file, or by duplicating the model within your Workspace) and revert to the saved, linked model each time you wish to continue with changes to the spreadsheet. Then you can re-convert the spreadsheet to a DPL program each time you want to run the model with your most recent changes.

If you forget to save a version of the linked model, and you follow the steps above to convert to a DPL program, there is an easy way to "undo"

this. You will "undo" the conversion now and re-link your DPL model to your spreadsheet.

- ⇒ Switch back to the Cash Flow with Decision model.
- ⇒ Select Influence Diagram | Links | Settings.
- ⇒ In the *Calculation Program* section, click the Revert to Linked button. You will need to indicate the location and name of the linked spreadsheet. DPL will re-link your model as it was originally.
- ⇒ Click OK for the warning.
- ⇒ Locate the Cash Flow1.xlsx file when prompted by DPL. Click Ok to close the Manage Links dialog.

After you revert to the linked model, you will see in the Workspace Manager that the converted spreadsheet program has been moved to the Unreferenced programs section, because it is no longer used by any of the models in the Workspace Manager. See Figure 16-6.



**Figure 16-6. Workspace Manager with Unreferenced Programs**

If you prefer, you can delete an unreferenced program from the Workspace by clicking on it and deleting it. DPL will ask you to confirm that you wish to delete the item.

## 16.2 Spreadsheet Practices for Easier Conversion

---

While DPL supports the functions most commonly used in financial spreadsheets, Excel's functionality is vast, so it isn't practical for DPL to implement every Excel function. If you know in advance that the spreadsheet you are building will be linked to DPL, you can follow the guidelines in Table 16-4 to ease spreadsheet conversion.

Be sure to contact Syncopation technical support if you have questions about these suggestions or other conversion issues. It is not uncommon for experienced DPL users to need a little help converting very large or complex Excel spreadsheets.

<b><u>Spreadsheet modeling issue</u></b>	<b><u>Suggestion</u></b>
<i>Table lookups</i>	<b>Use</b> VLOOKUP, HLOOKUP, INDEX. <b>Don't use</b> OFFSET.
<i>Dates</i>	<b>Use</b> numerical calculations. Excel date functions are not supported.
<i>Iteration</i>	Spreadsheets that iterate cannot be converted, and tend to be slow even when linked. Iteration is often used for recursive calculations such as interest, which can be solved explicitly.
<i>Rounding</i>	<b>Use</b> ROUND. <b>Don't use</b> ROUNDUP, ROUNDDOWN.
<i>Miscellaneous</i>	<b>Use</b> array formulas in place of SUMIF, COUNTIF, RANK.

**Table 16-4. Spreadsheet Modeling Suggestions for Easier Conversion**

While following these suggestions may represent a significant change of practice, keep in mind that a converted spreadsheet often runs 10-100 times as fast as a linked one, so for a large model it can be worth the effort.

Syncopation sometimes makes enhancements to DPL's spreadsheet conversion between major releases. Contact Syncopation's technical support team at [support@syncopation.com](mailto:support@syncopation.com) for the most up to date information on spreadsheet conversion and supported functions.

## 17. DPL™ Programs and Code

The DPL language (DPL "code") is at the heart of DPL. It provides a simple, compact and precise way to define almost any decision analysis model using text rather than graphics. You don't have to know anything about DPL code to use DPL, most users don't. However, if you're building large, complex models, or are converting spreadsheets, you may find it useful to look at DPL code and/or work with it. In a DPL Workspace, DPL code is contained within DPL programs.

This chapter provides an overview of DPL programs and their components followed by a brief tutorial on converting DPL models to programs. The last three sections of this chapter are intended mainly as a reference; they provide more detailed documentation of the definition and sequence sections of programs as well as advanced techniques for using DPL programs.

### 17.1 What is a DPL™ Program?

---

A DPL program is a text based description of the components of a decision analysis model. A DPL program may contain a complete description of a decision analysis model with decision and chance events; conditioning; values and a sequence section or it only may contain a set of data definitions to be used in another program.

When you build a model and then run any form of analysis on it, DPL converts the model to a DPL program written in DPL code under-the-covers. It then compiles the program, checking that the model is complete, consistent, and structurally sound. Then, it uses the analysis engine to evaluate the model and create the requested graphical outputs.

You don't have to start with building a graphical Influence Diagram/Decision Tree within the Model Window. If you prefer, you can write a DPL program directly in a Program Window. Most people, however, prefer to use the graphics and data management tools provided in the Model Window. There are three situations in which you may wish to work with a DPL program directly:

- Converting spreadsheets -- DPL is able to convert Excel spreadsheets into DPL programs. Models using converted spreadsheets run much more quickly than models linked to Excel.

DPL can evaluate models that combine graphic models and DPL programs, so you can continue to use the Model window to build Influence Diagrams and Decision Trees while using a converted spreadsheet.

- Documenting your model -- if you need a printed, hard-copy version of your complete model, you can print the Influence Diagram, Decision Tree, and all the data trees from the Model window. Or, you can convert your model to a DPL program and print it out. The DPL program is usually much more compact.
- Quality assurance -- you may find that a DPL program's compact presentation of all the data and formulas in your model makes verifying that everything is complete and correct an easier task. DPL can generate programs automatically from Influence Diagrams/Decision Trees.

## 17.2 Overview of DPL Program Components

---

A typical DPL program is divided into two sections: a definition section and a sequence section. In the definition section, you enter a collection of statements that describe the elements of the decision problem—decisions, chance events, data—and how they work together. The definition section is analogous to the Influence Diagram. In the sequence section, you describe the sequence in which decisions are made, uncertainties are resolved, and values are calculated. The sequence section is analogous to the Decision Tree. It is not necessary to include both sections in every program.

If you generate a DPL program from a graphical DPL model, the Influence Diagram is described in a definition section. The Decision Tree is described in a sequence section. Most programs created from graphical models will have both a definition section and a sequence section. If you convert a spreadsheet to a DPL program file, it will have the appropriate format to be included in the definition section and it will not contain a sequence section.

### 17.2.1 The Definition Section

Most DPL programs start with a definition section. This section, which specifies and initializes the components of a decision problem, usually contains the bulk of the model.

There are two components you can define in the definition section:



- Events, which include decisions, chance events, and controlled events
- Values, which include constants, scalars, series, arrays with string or numeric data

Strategy tables are handled as a set of decisions in DPL programs.

### 17.2.2 The Sequence Section

DPL sequence sections are systematic English-language descriptions of Decision Tree structures. There are two tasks to accomplish in the sequence section of a DPL program. First, you must specify the chronological order in which the events, or nodes, occur. Second, you must specify the actions, such as payment or receipt of values, which are associated with each branch.

### 17.2.3 Comments

Comments are strings of characters that are ignored by the DPL compiler. They allow you to insert explanatory notes in a DPL program to organize it or to help other people understand it. You can create comments that occupy several lines, but you cannot nest one inside another. Comments can appear anywhere a blank or tab is allowed.

There are two types of comments in DPL. The first type is bracketed by two pairs of characters, `/*` and `*/`. This type of comment can occupy more than one line. The first occurrence of `*/` ends the comment. This type of comment may be used in the data entry fields of the Model window as well as in programs and command statements. The second type of comment begins with a pair of forward slash characters (`//`) and ends at the end of the line. By definition, this type of comment cannot occupy more than one line.

### 17.2.4 Spelling/Case Sensitivity

It is very important to pay attention to spelling in a DPL program. DPL is case sensitive, which means that "Oil\_Price" and "oil\_price" are not equivalent. All DPL reserved words are lower case (*decision*, *chance*, and *value* are all reserved words). A reserved word cannot be used as an identifier. User-defined identifiers can be in lower or upper case. Refer to the online Help for more information on identifiers and reserved words.

## 17.3 Converting a Model to a Program

The easiest way to familiarize yourself with DPL programs is to convert an existing Influence Diagram/Decision Tree model to a program and inspect the code that results. As you become more proficient with DPL, you may find it helpful to convert your models to programs from time to time for debugging, documentation, or other purposes.

In this section, you will convert the energy case to a DPL program.

- ⇒ Open Energy Case.da.
- ⇒ This file is found in the Examples folder where DPL was installed, usually C:\Program Files\Syncopation\DPL9\Examples.
- ⇒ Select Home | Workspace | Add to WS | Program from Model

DPL converts the model to a DPL program and displays the converted model in a Program Window. Note that DPL did not replace the model with the program; both the model and the program are contained in the Workspace and can be seen as separate items in the Workspace Manager.

Below is the text of the DPL program for the energy case. The text is included here for reference, but it will be easier for you to read in the DPL Program Window because of its width. Use the scroll bar at the bottom of the window to scroll right and see all the text.

- ⇒ If you like, select File | Print and print the text of the energy case program. You may need to print in Landscape format to see the entire program.

```
string Excel_1="C:\\Program Files\\Syncopation\\DPL9
\\Examples\\Energy Case.xlsx";
excel(Excel_1,"Assumptions!Discount_Rate") value Discount_Rate=.1;
value p=.55;
value q=0;
excel(Excel_1,"Assumptions!Market_1_Size") value Market_1_Size=110;
excel(Excel_1,"Assumptions!Market_2_Size") value Market_2_Size=180;
```

```

excel(Excel_1,"Assumptions!Mkt_1_Model")
value Market_1_Growth_Model=3;
excel(Excel_1,"Assumptions!Mkt_2_Model")
value Market_2_Growth_Model=3;
excel(Excel_1,"Assumptions!Proceed") decision Proceed.{Yes~,No}=
  1,      // Proceed.Yes
  0;      // Proceed.No
excel(Excel_1,"Assumptions!Initial_Upstream_Investment")
  decision Initial_Upstream_Investment.{Low~,High}=
    620,   // Initial_Upstream_Investment.Low
    630;   // Initial_Upstream_Investment.High
decision Pipeline_Strategy.{Mkt_1___10_BCM~,Mkt_1___20_BCM,
  Mkt_1_and_2___10_BCM,Mkt_1_and_2___20_BCM,
  Mkt_1_and_2___30_BCM};
excel(Excel_1,"Assumptions!Include_Market_2")
  decision Include_Market_2.{Yes~,No}=
    2500,  // Include_Market_2.Yes
    0;     // Include_Market_2.No
excel(Excel_1,"Assumptions!Pipeline_Capacity")
  decision Pipeline_Capacity.{BCM_10~,BCM_20,BCM_30}=
    10,    // Pipeline_Capacity.BCM_10
    20,    // Pipeline_Capacity.BCM_20
    30;    // Pipeline_Capacity.BCM_30
chance CapEx.{Low,Nominal~,High}={.2,.4,.4};
excel(Excel_1,"Assumptions!CapEx")
  value CapEx|Pipeline_Capacity,CapEx=
    // Pipeline_Capacity.BCM_10
    650,   // CapEx.Low
    1000,  // CapEx.Nominal
    1750,  // CapEx.High
    // Pipeline_Capacity.BCM_20
    750,   // CapEx.Low
    1100,  // CapEx.Nominal
    2000,  // CapEx.High
    // Pipeline_Capacity.BCM_30
    850,   // CapEx.Low
    1200,  // CapEx.Nominal
    2200;  // CapEx.High
excel(Excel_1,"Assumptions!Actual_Price")
  chance Actual_Price.{Low,Nominal~,High}={.3,.4,.3},=
    10,    // Actual_Price.Low
    25,    // Actual_Price.Nominal
    35;    // Actual_Price.High
excel(Excel_1,"Assumptions!Actual_Reserves")
  chance Actual_Reserves.{Low,Nominal~,High}={.3,.4,.3},=
    1000,  // Actual_Reserves.Low
    3000,  // Actual_Reserves.Nominal
    5000;  // Actual_Reserves.High
chance Market_2_Political_Stability_Now.{Stable~,Unstable}={p};
chance Forecast_Price.{Low,Nominal~,High}|Actual_Price=
  {.75+q,.15-q},    // Actual_Price.Low
  {.125+q,.75-q},  // Actual_Price.Nominal
  {.1+q,.15-q};    // Actual_Price.High
value Forecast_Price|Forecast_Price=
  15,    // Forecast_Price.Low
  25,    // Forecast_Price.Nominal
  35;    // Forecast_Price.High
chance Market_2_Political_Stability_Future.

```

```

    {Stable~,Moderate,Unstable}|
    Market_2_Political_Stability_Now=
    {.7,.2,.1}, // Market_2_Political_Stability_Now.Stable
    {.05,.1,.85}; // Market_2_Political_Stability_Now.Unstable
excel(Excel_1,"Assumptions!Market_2_Stability_Future")
    value Market_2_Political_Stability_Future|
    Market_2_Political_Stability_Future=
    1, // Market_2_Political_Stability_Future.Stable
    .5, // Market_2_Political_Stability_Future.Moderate
    .05; // Market_2_Political_Stability_Future.Unstable
excel(Excel_1,"Assumptions!Operating_Costs")
    chance Operating_Costs.{Low,Nominal~,High}={.3,.4,.3},=
    18, // Operating_Costs.Low
    20, // Operating_Costs.Nominal
    22; // Operating_Costs.High
chance Appraisal_Results.{Low,Nominal~,High}
|Initial_Upstream_Investment,Actual_Reserves=
// Initial_Upstream_Investment.Low
{.45,.3,.25}, // Actual_Reserves.Low
{.275,.45,.275}, // Actual_Reserves.Nominal
{.25,.3,.45}, // Actual_Reserves.High
// Initial_Upstream_Investment.High
{.85,.1,.05}, // Actual_Reserves.Low
{.075,.85,.075}, // Actual_Reserves.Nominal
{.05,.1,.85}; // Actual_Reserves.High
excel(Excel_1,"Assumptions!Total_NPV") value Total_NPV;
excel(Excel_1,"Assumptions!Upfront_NPV") value Upfront_NPV;

```

sequence:

```

decide to Initial_Upstream_Investment then
gamble on Appraisal_Results then
gamble on Forecast_Price then
gamble on Market_2_Political_Stability_Now then decide
to Proceed.Yes then decide
    to Pipeline_Strategy.Mkt_1__10_BCM and
    set Pipeline_Capacity.BCM_10 and
    set Include_Market_2.No and
    StrategyTableSubtree0:
    a: gamble on Actual_Price then
    gamble on Actual_Reserves then
    gamble on Operating_Costs then
    gamble on CapEx and get Total_NPV
to Pipeline_Strategy.Mkt_1__20_BCM and
    set Pipeline_Capacity.BCM_20 and
    set Include_Market_2.No and
    perform StrategyTableSubtree0
to Pipeline_Strategy.Mkt_1_and_2__10_BCM and
    set Pipeline_Capacity.BCM_10 and
    set Include_Market_2.Yes and
    StrategyTableSubtreel:
    gamble on Market_2_Political_Stability_Future
    then perform a
to Pipeline_Strategy.Mkt_1_and_2__20_BCM and
    set Pipeline_Capacity.BCM_20 and
    set Include_Market_2.Yes and
    perform StrategyTableSubtreel
to Pipeline_Strategy.Mkt_1_and_2__30_BCM and

```

```

        set Pipeline_Capacity.BCM_30 and
        set Include_Market_2.Yes and
        perform StrategyTableSubtree1
to Proceed.No and get Upfront_NPV

```

⇒ Spend a few minutes examining the program you have created. Below are some of the characteristics of the code that you should notice.

The first line of the program tells you that the model is linked to a spreadsheet and gives the spreadsheet's filename and path. DPL gives the spreadsheet the label "Excel\_1" and the value, chance, and decision nodes that are linked to this spreadsheet are preceded by the "excel" keyword and the name of the linked cell; for example:

```

excel(Excel_1,"Assumptions!Market_1_Size")
value Market_1_Size=110;

```

The majority of the code in the definition section is representing the Influence Diagram, as described earlier. See Section 17.4 for more information on the definition section.

DPL automatically inserts comments (the text following each //) indicating the settings of each event state whose probability and/or value is being defined; for example:

```

Actual_Price.{Low,Nominal~,High}={.3,.4,.3},=
10, // Actual_Price.Low
25, // Actual_Price.Nominal
35; // Actual_Price.High

```

Comments such as the above are not necessary but they usually make it easier to read and interpret the DPL program.

The last elements in the definition section are the definitions of the two Excel-linked value nodes that import the cash flow results:

```

excel(Excel_1,"Assumptions!Total_NPV")
value Total_NPV;
excel(Excel_1,"Assumptions!Upfront_NPV")
value Upfront_NPV;

```

⇒ Scroll down until you see the line that says "sequence:".

The section of the program after the line "sequence:" represents the Decision Tree. This section tells DPL the order in which it should evaluate the nodes and which Get/Pay expressions to evaluate during the analysis.

Notice that since the first four events on the tree are symmetric, the DPL code for this portion of the tree is very simple:

```

decide to Initial_Upstream_Investment then
gamble on Appraisal_Results then

```

```
gamble on Forecast_Price then
gamble on Market_2_Political_Stability_Now then decide ...
```

Following this section, the DPL program describes the asymmetric portion of the tree. In this last part of the program, the indenting of the program text matters. See Section 17.5 for detailed information on the sequence section.

If the Program Window is active in the Model Window, DPL will analyze the program when you run an analysis.

- ⇒ With the Program Window active, run the program as you normally would run a model (Home | Run | Decision Analysis or press F10). The program will open the linked Excel spreadsheet and run the model, producing the same initial results you saw in Chapter 11. The expected value of the model is still 34.9.

You will make a change to the program that affects the value model, and run it one more time.

- ⇒ Activate the Program Window to edit the program.
- ⇒ In the second line of the program, change the discount rate from 0.10 to 0.05. The code should now look like this:

```
excel(Excel_1,"Assumptions!Discount_Rate") value
Discount_Rate=.05;
```

- ⇒ Run the program again and inspect the new results.

The results are quite different because a lower discount rate implies a higher value for investments with long-term payback. The expected value is now roughly 57, and the optimal policy is always to proceed with the pipeline strategy regardless of the outcomes of the initial uncertainties.

Keep in mind that the program and graphic model are not linked, so changes in one do not appear in the other. Also, it is not possible to go the "reverse direction" and generate a graphic model from a program.

Also note that the program in this example is still linked to the Excel spreadsheet for evaluation of the cash flow results. That is, when you ran the program, the program had to open the Excel spreadsheet to calculate `Upfront_NPV` and `Total_NPV`, just as the DPL model would do.

If you become comfortable with using DPL programs, you may find them useful for quick "what-if" analyses, such as the change you just made to the discount rate. You can always revert back to the original model by deleting the program from the Workspace Manager and re-generating it from the original model.

The remaining sections of this chapter are intended for DPL users who need a more thorough understanding of DPL programs.

## 17.4 Defining Components in the Definition Section

---

The definition section is a list of definitions of events, variables, and constants. This section can also include comments. DPL is a very flexible language; there are only a few rules for ordering, punctuation, and spacing.

### 17.4.1 Definition Order

You may define the elements of a decision problem in any order; you do not have to group all values together or have all decisions appear at the beginning of the definition section. You can refer to previously defined events, values, or other elements in subsequent data definitions and in the sequence section. The only limitation is that you cannot refer to a name before it is defined.

### 17.4.2 Punctuation and Spacing

DPL requires that statements that define elements of a decision problem end with a semicolon. (In contrast, semicolons are not required in the second part of a DPL program, where you define the sequence of events.) Definitions also have punctuation to separate the parts; refer to the punctuation in the examples as a guide. DPL will usually catch punctuation errors during compilation. If it does find such an error, it will return you to the error location.

DPL doesn't care how many lines a definition covers or how many spaces or tabs you include between the words in a definition. The only restriction on spacing is that identifiers (the names of events, states, values, series, and so on) must not include blanks or tabs. Everywhere else in the definition section, you may space things however you like. Note that this is not the case in the sequence section if the model is asymmetric; see Section 17.5.

### 17.4.3 Excel-Linked Events and Values

Values and events linked to Excel are preceded by an `excel()` clause, as in the following.

```
excel(Excel_1,"Assumptions!Market_1_Size")
value Market_1_Size=110;
```

The `excel()` clause shows the workbook path (`Excel_1`) and the sheet and cell (`Assumptions!Market_1_Size`) to which the value or event is linked. In most cases, all nodes are linked to the same workbook, and the path and file name of that workbook are defined as a string named `Excel_1`. If the Excel linked value is a metric node, then there will be no assignment (`"="`) following the value as in the `Total_NPV` example above.

### 17.4.4 Decisions

The following statement defines a decision in DPL.

```
decision Research_Funding.{Low,Medium~,High};
```

The tilde (`~`) character indicates that the Medium state is the default state. The tilde is optional; however, it is recommended if you will be using Base Case Tornado diagrams or Asymmetric Tree Dependency Checking. If no state is marked with the tilde, the first state is the default.

If you wish to associate value expressions with the states of a decision, add them to the definition following the state names.

```
decision Research_Funding.{Low, Medium~, High}
= 10000,25000,35000;
```

### 17.4.5 Chance Events

The following statement defines a chance event in DPL.

```
chance Research_Time.{Short,Medium~,Long}
= {0.3,0.5};
```

In the statement above, only two probabilities are specified. If no probability is given for the last state, DPL will calculate it as one minus the sum of the other probabilities.

Like decisions, chance events can have default states, indicated by the tilde (`~`) character.

If you wish to associate value expressions with the states of a chance event, enter them after the probability distributions. A chance event with both probabilities and values assigned to its states is sometimes called a random variable.



```
chance Research_Time.{Short,Medium~,Long}  
  = {0.3,0.5}  
  = 1, 5, 10; //in months
```

Chance events may be drawn from named distributions. To do so, do not enter probabilities and value expressions. Instead, use the distribution name and enter value expressions for the parameters.

```
chance A.{High, Medium, Low} = beta(1,2);
```

For more information on named distributions, refer to the online Help.

Chance events may also be continuous. A continuous chance node is defined in two parts, one for the chance event and one for the associated value.

```
chance B.{1}=;  
value B=DPLMC.normal(10,3);
```

The first line above defines a chance event with a single unnamed state (this is how DPL knows the event is continuous — discrete events have two or more states). The second line tells DPL that the chance event is drawn from a normal distribution, and that the mean and standard deviation are 10 and 3 respectively. DPLMC refers to DPL's Monte Carlo library, DPLMC.DLL, which is part of the DPL installation. For a list of available continuous distributions, see the online Help.

If the probabilities or values associated with a chance event depend on the states of other events, you include this information in the chance event's definition. You must define the conditioning events before you define a chance event that depends on them. You must provide probability distributions and value expressions for each combination of conditioning event states.

For example, the following statement defines a chance event that depends on a decision called `Research_Funding` and a chance event called `Research_Time`, both defined earlier. Both `Research_Funding` and `Research_Time` are conditioning events.

```
chance Research_Cost.{Low,Moderate,Outrageous}
  given Research_Funding,Research_Time
  =          //Research_Funding Research_Time
  {0.6,0.2}, //Low           Short
  {0.4,0.3}, //Low           Medium
  {0.2,0.4}, //Low           Long
  {0.4,0.3}, //Medium        Short
  {0.4,0.3}, //Medium        Medium
  {0.2,0.4}, //Medium        Long
  {0.2,0.4}, //High          Short
  {0.2,0.4}, //High          Medium
  {0.1,0.4}; //High          Long
=
40, 50, 80, //Low           Short
45, 60, 85, //Low           Medium
40, 60, 80, //Low           Long
50, 95, 110, //Medium        Short
80, 100, 125, //Medium        Medium
85, 105, 130, //Medium        Long
90, 105, 130, //High          Short
95, 120, 140, //High          Medium
95, 140, 160; //High          Long
```

This statement defines nine sets of probabilities, one for each combination of the conditioning events' states (`Research_Funding` and `Research_Time`). The order in which you specify the probabilities depends on the order in which you list the conditioning events after the keyword `given`. (You can substitute a vertical bar "|" for the keyword `given`.) DPL uses the first set of probabilities when all conditioning events are in their first state. The next set of probabilities applies when the last conditioning event in the list changes to its second state. After associating sets of probabilities with all the states of the last conditioning event DPL varies the states of the next-to-last conditioning event and so on. This continues until probabilities are associated with all combinations of states. This method for determining the order of the various combinations of states is sometimes called row major order or the odometer principle. This means that the states of the last event vary most rapidly. The same applies for the values.

The comments in the earlier statement show the states of the conditioning events. These comments are ignored by DPL, and they do not determine the order in which you specify the probabilities.

### 17.4.6 Controlled Events

The following statement defines a controlled event in DPL.

```
controlled AchieveOrbit.{Yes,No};
```

If you wish to associate value expressions with the states of a controlled event, enter them after the state names.

```
controlled AchieveOrbit.{Yes,No} = 500,-10;
```

### 17.4.7 Values

The following statement defines a value in DPL.

```
value Base_Price = 585.30;
```

The following statements define two events and a value, `Inflation_Factor` that depends on the two events.

```
chance MoneySupply.{Lower, Constant, Higher}
    = {.35,.55};
chance TaxRate.{Unchanged, Increased} = {.1};
value Inflation_Factor given MoneySupply, TaxRate
    =
// TaxRate.Unchanged,    TaxRate.Increased
1.14,                    1.12, //MoneySupply.Lower
1.11,                    1.09, //MoneySupply.Constant
1.09,                    1.08; //MoneySupply.Higher
```

The order in which you list the numbers depends on the order in which you list the conditioning events. DPL uses the first number when all conditioning events are in their first state. The next number applies when the last conditioning event changes to its second state. After associating numbers with all the states of the last conditioning event, DPL varies the state of the next-to-last conditioning event and so on. This continues until a number is associated with all combinations of states. Once again, the comments are ignored by DPL and do not determine the order in which you specify the values.

## 17.4.8 Shortcuts

### Statenames

If you don't want to name the states of an event, you can define events without state names. DPL will supply default state names. For example, the following statement defines a decision called Invest that has two alternatives.

```
decision Invest.{2};
```

In the program you can refer to these alternatives as Invest.s1 and Invest.s2. If an event has only two states, DPL also allows you to call them t (for true) and f (for false), or y (for yes) and n (for no), respectively.

If you do not specify the number of alternatives or list their names, DPL will generate two unnamed states. The following definition is equivalent to the one above.

```
decision Invest;
```

If you define a chance event without specifying state names, you should still provide probabilities.

```
chance MarketPrice.{4} = {0.1, 0.3, 0.4};
```

If you define a chance event without probabilities, DPL will assign a uniform distribution to it.

```
chance MarketPrice.{4};
```

For this example, each state of MarketPrice will have a probability of 0.25.

### Events and Values with the Same Name

Although you cannot have two events with the same name or two values with the same name, you can have an event and a value with the same name. For example, this definition:

```
decision Research_Funding.{Low,Medium,High}
= 30, 52.5, 80;
```

could be replaced with these two:

```
decision Research_Funding.{Low,Medium,High};
value Research_Funding given Research_Funding
= 30, 52.5, 80;
```

Values defined with events and values defined separately can be used in exactly the same way. DPL can tell whether you are referring to the value or to the event by the context.

Zeros

DPL also allows you to under specify the number of values. If there are fewer initialization expressions than expected, DPL will set the remaining values to 0.

```
chance Fire_Damage.{High, Medium, None}
  = {0.2, 0.5}
  = 50000,10000;
```

DPL will automatically assign a value of 0 to the final state of this chance event. Beware: The fact that DPL does this can cause problems if you have incorrectly specified the values because you will not get an error message. For example, the following definition will not work as expected.

```
value Profit | Revenues, Costs
  = Revenues - Costs;
```

This definition will result in Profit being assigned the difference between Revenues and Costs only when Revenues and Costs are each in their first states. All other times, Profit will be assigned a value of 0. The correct definition is:

```
value Profit = Revenues - Costs;
```

Default...Case Notation

DPL provides another way to specify the probabilities for a chance event that depends on other events. This approach can simplify the task of specifying the probabilities when the conditioning events have many combinations of states. For example, rather than specifying each probability distribution individually:

```
chance A.{S1, S2, S3, S4} = {.1, .2, .3};
chance B.{Yes, No} | A = {.1}, {.1}, {.1}, {.5};
```

you can define a default distribution and then define only the exceptions.

```
chance A.{S1, S2, S3, S4} = {.1, .2, .3};
chance B.{Yes, No} | A =
  default: {.1},
  case A.S4: {.5};
```

If there are multiple conditioning events, you can use logical OR (||) and logical AND (&&) to specify combinations of events for the cases.

As with the probability distributions in chance event definitions, you can save time and effort in a value definition by defining one value as the default for the value and identifying particular cases for other definitions.

### 17.4.9 Series

A series is a one-dimensional vector of numbers whose elements are initialized in groups or intervals.

```
series Sales =
  from 2014 : 0,
  from 2015 : 100000,
  from 2017 to 2020 : 150000;
```

In this example, the series is called `Sales`. `Sales` contains three intervals: one that has one element (corresponding to the year 2014), one that has two elements (corresponding to years 2015 and 2016), and one that has four elements (corresponding to years 2017, 2018, 2019, 2020).

A series has at least one interval definition that contains an expression that gives the number represented by the series in that interval. The interval definitions also contain the interval boundaries—in this case 2014, 2015, 2017, and 2020. The lower bound of the first interval and the upper bound of the last interval define the lower and upper bounds of the series, and they must be constant expressions. The other interval boundaries are called intermediate interval boundaries, and they may be variable expressions. Only the upper bound for the last interval must be explicitly defined; the upper bounds for the other intervals are calculated by subtracting one from the lower bound of the next interval.

The interval bounds and subscripts must be integers, and they can cover any range that you designate, including negative numbers. Each interval boundary must exceed the previous one. All interval ranges must lie between  $-32768$  and  $+32767$ . If necessary, DPL will truncate subscripts and interval bounds to integers.

A series also has a subscript (a way to reference particular elements of the series) which can range between the lower and upper boundaries of the series. In this example, the subscript may range from 2014 to 2020. A basic subscript refers directly to a particular element in the series. For example, the following refers to the value for `Sales` in 2017:

```
Sales[2017]
```

A relative subscript, contains a special identifier (`$`) that represents the current subscript of the series being initialized.

```
series Sales =
  from 2014 : 0,
  from 2015 to 2024 : Sales[$ - 1] + 10000;
```

In this case, when DPL evaluates the value of Sales for 2015, the \$ symbol stands for 2015, so DPL looks up the value of Sales for 2014 (that is, Sales[2015-1]) and adds 10,000 to it.

You must initialize each element of a series, which means that each interval must be assigned an expression. The expression can include the names of variables previously defined in the model and any function supported by DPL. Unlike those of an array (described later in this chapter), DPL does not allow the elements of a series to be un-initialized.

In addition to using the \$ symbol in the name of a series element, you can use the \$ symbol to represent the number of the current index in an expression.

```
series Years =
  from 2014 to 2024: [$];
```

Most definitions in DPL can only reference objects defined earlier in a program. However DPL allows you to reference a series before defining it. That makes it possible to define several series recursively. In the following example, the definition of Series1 references Series2 before Series2 is defined.

```
series Series2;
series Series1 =
  from 1: 1,
  from 2 to 10: Series2[$ - 1] + 1;
series Series2 =
  from 1 to 10: Series1[$] * 2;
```

With these definitions, Series1[2] equals 3, Series2[2] equals 6, Series1[10] equals 1023, and Series2[10] equals 2046. The first definition in the example indicates that the variable Series2 is a series rather than an array.

You can refer to series in three ways. Assume the series Cash\_Flows is defined as:

```
series Cash_Flow=
  from 2014: 10000,
  from 2015 to 2023: Cash_Flow[$-1] * 1.1;
```

You may refer to Cash\_Flow as follows.

- With a subscript, referring to a single element: e.g.,value CF2014 = Cash\_Flow[2014];

CF2014 is 10000.

- Without a subscript, referring to the entire series: e.g.,value CFTotal = @sum(Cash\_Flow);

CFTotal is 159374.25.

- With a subrange specification, referring to a selected subset of the elements in the series: e.g., value CFSubTotal = @sum(Cash\_Flow{2017..2020});

CFSubtotal is 61771.71.

For more information on using series, refer to the online Help.

### 17.4.10 Arrays

You can define one- or two-dimensional arrays of numbers in DPL. The following statement defines a one-dimensional array in DPL:

```
array A = {1, 2, 3};
```

The first element of this array is referred to as A[0], and it equals 1. The second element is A[1], and it equals 2 and so on. Commas separate elements in a one dimensional array.

The following statement defines a two-dimensional array in DPL:

```
array B =
  { 1,  2,  3;
    4,  5,  6;
    7,  8,  9;
    10, 11, 12};
```

To refer to the element in the first row and first column of array B, you use the syntax A[0][0] (which equals 1). The element in the first row and second columns is A[0][1] (which equals 2) and so on. Commas separate elements in a row in two-dimensional array. Semicolons separate rows in a two-dimensional array. Note: the last row does not end with a semicolon.

The elements of an array, unlike those of a series, cannot be initialized in groups. You must initialize each element of an array separately, although you can leave some elements un-initialized. If you do not initialize an array element, it is called a null element.

The subscripts of an array are always in a range from 0 to a positive integer. If necessary, DPL will truncate the number represented by an array subscript to produce an integer.



If you would like to declare an array that has multiple rows and one column, you must do this as a two-dimensional array. The array `Payments` below is an array with four rows and one column.

```
array Payments = {100;
                  150;
                  200;
                  240};
```

To access the third row of this array you use `Payments[2][0]`. You may not omit the `[0]`.

Note: one-dimensional arrays are treated like two-dimensional arrays with one row. Therefore you can refer to the elements of array `A` declared above as either `A[0][1]` or `A[1]`. The `[0]` is optional.

You can initialize each element of an array with an expression. The expression can contain values that depend on events and any of DPL's functions. DPL allows you to reference an array before defining it. This makes it possible to define several arrays recursively. Because you can define both series and arrays recursively, you must first specify whether the variable being referenced before its definition is a series or an array (if you don't, DPL will assume it is a series). Then, use the variable name, and finally, provide the complete definition. For example:

```
array a;
series s =
    from 1: a[0][0],
    from 2 to 4: a[1][0];
array a = {@sum(s{2..4}), 0, 0;
          1, 2, 3};
```

You can refer to arrays in three ways. Assume the series `Costs` is defined as:

```
array Costs =
    {10000, 20000, 30000, 40000;
     12500, 22500, 32500, 42500;
     15000, 25000, 35000, 45000};
```

- With subscripts, referring to a single element: e.g., value `Year1Cost = Costs[0][0]`;

`Year1Cost` is 10000.

- Without subscripts, referring to all the elements in the array: e.g., value `CostTotal = @sum(Costs)`;

`CostTotal` is 330000.

- With a subrange specification, referring to a selected subset of the elements in the array: e.g., value `CostSubTotal = @sum(Costs{0..1}{1..3});`

`CostSubTotal` is 187500.

### 17.4.11 Strings

You can define a variable called a string and initialize it with a sequence of characters.

```
string Report_Title = "Fuel Price Analysis";
```

Strings can depend on events.

```
decision Strategy.{US, Europe};
string Currency given Strategy = "Dollars", "Euros";
```

You can use a string name in place of a string literal anywhere that DPL allows string literals. You can use strings and string literals as the argument of a display function, as arguments of the lookup functions (`@hlookup`, `@vlookup`), and as the arguments of the relational operators.

If strings are used as arguments of the relational operators (`<`, `>`, `=`, and so on), the ordering is determined by the standard ASCII values for characters. For example:

```
"a" < "z"
"a" > "A"
```

Both relations are true (that is, they both result in a value of 1).

If a string name appears in a math expression, DPL will generate a "Not a number" error message.

### 17.4.12 Named Constants

A named numeric constant represents a number. When you define a named constant, you assign it an identifier and an initial value. DPL stores the initial value at the time it compiles a program, and it does not allow the value to change thereafter.

The following statement defines a named numeric constant in DPL.

```
const BaseYear = 2014;
```

You can use a named numeric constant, or constant expressions containing named numeric constants, in places where a variable is not allowed. These include DPL statements that define the number of attributes associated with alternatives and outcomes, the number of alternatives for a decision, the number of outcomes for a chance event, and the number of elements in a series.

Unlike values, named numeric constants cannot depend on events. Once you define a named numeric constant, you cannot change it with a DPL command or perform a sensitivity analysis with respect to it. However, you can change the definition of a named numeric constant in a DPL program file and then recompile the program. This allows you to consistently and easily change parameters of a decision analysis throughout a DPL program.

### 17.4.13 The Integer Keyword

The DPL keyword `integer` can be used in the definition of constants, values, series, and arrays that must have integer values. Whenever DPL evaluates an expression that will be assigned to an integer variable, the expression result is rounded to the nearest integer value. For example:

```
integer value Years_Remaining = years * 1.83;
integer Years_Remaining
    = years * 1.83; // same as above
integer series index =
    from 2014: 1,
    from 2011 to 2016: 2.6 * factor;
```

For most purposes, the `integer` keyword and the `@round` function have the same effect; the `integer` keyword is just a convenience. For example, if you want all the values in an array to be integers, you can use the `@round` function in the definition of each array item, or you can define the array to be an integer array, and the rounding will be automatic.

If you have a variable that must be an integer and you are using the "dont" specification to remove lotteries from the sequence section, you must use the `integer` keyword in that value's definition.

Don't confuse the `integer` keyword with the `@int` function: the `@int` function truncates an expression value to the next lower integer; the `integer` keyword rounds an expression value to the nearest integer.

You cannot define the values associated with a chance, decision, or controlled event to be integers. The following code is incorrect:

```
//incorrect code
integer chance A.{3} = {.2, .3} = 1, 2, 3;
```

If you wish to define integer values conditioned by a chance, decision or controlled event, do the following:

```
//correct code
chance A.{3} = {.2, .3};
integer value A given A = 1, 2, 3;
```

## 17.5 Defining Event Sequences in the Sequence Section

---

The sequence section starts with the keyword `sequence`.

```
sequence:
```

If you are using multiple attributes, defining an objective, or defining a constraint function, the word `sequence` is followed by the appropriate specifications.

```
sequence (attributes = 2, objective = $1 * $2):
```

There is one important limitation on the organization of the sequence section: keywords marking the states of a single decision or chance event must begin in the same column. Apart from this, you may define a sequence section with as many or as few lines as you like.

### 17.5.1 Decisions

You specify a decision in the sequence of events with the keyword `decide` or the keyword phrases `make a decision`, `make a max decision`, or `make a min decision`. The first three are equivalent. You can use the one that makes the program easiest to read. Next, you identify each alternative (or group of alternatives) with the keywords `on`, `to`, or `about`. You can use whichever keyword makes the program easiest to read and you can use different keywords with different alternatives of the same decision.

```
decide
  to Release_Product.Yes ...
  to Release_Product.No ...
```

### 17.5.2 Chance Events

You specify a chance event in the sequence section of a program with the keywords `chance` or `gamble` or the keyword phrases `take a chance` or `take a gamble`. All of these keywords or keyword phrases are equivalent. You can use the one that makes the program easiest to read.

Next, you identify each outcome (or group of outcomes) with the keywords `on` or `of`. You can use whichever keyword makes the program easiest to read, and you can use different keywords with different outcomes of the same chance event.

```
gamble
  on Market_Forecast.High ...
  on Market_Forecast.Medium ...
  on Market_Forecast.Low ...
```

### 17.5.3 Branches

If a decision or chance node is symmetric, DPL code provides a way to use schematic notation, as in a Decision Tree. The event name appears by itself, and the individual states are not listed. For example,

```
gamble on Demand then ...
```

The above code is equivalent to the node being placed in the Decision Tree as shown in Figure 17-1.

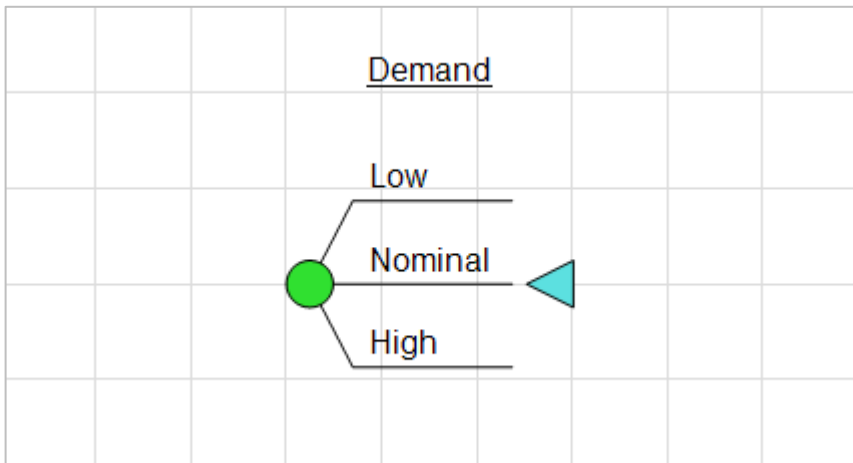
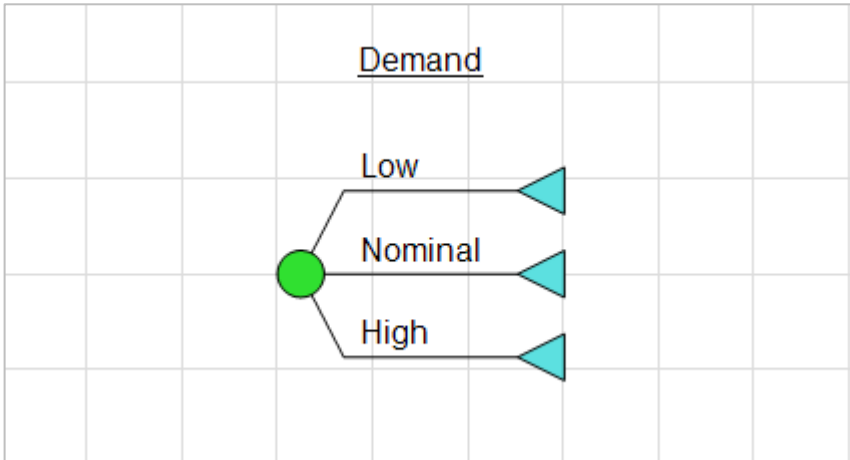


Figure 17-1. Node Placed with Symmetric Grouping

If a decision or chance node is asymmetric, then each state must be listed:

```
gamble
  on Demand.High then
  on Demand.Medium then
  on Demand.Low then
```

The above code is equivalent to the node being placed in the Decision Tree as shown in Figure 17-2.

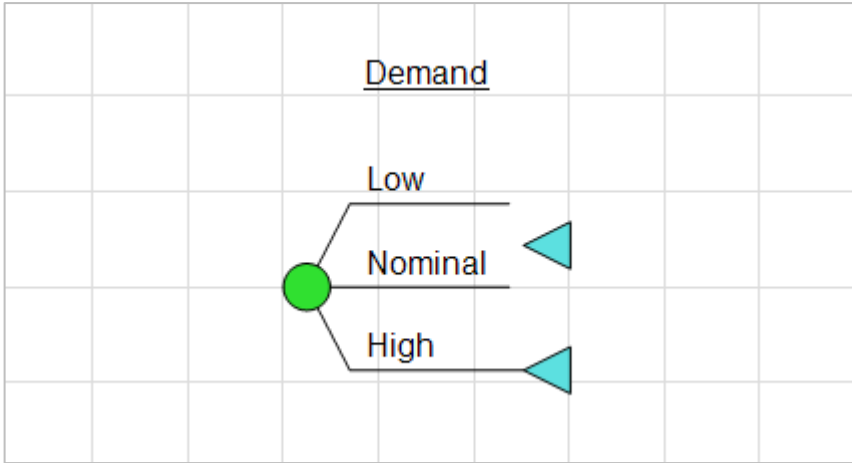


**Figure 17-2. Node Placed with Asymmetric Grouping**

In the asymmetric case, DPL will generate an error message if you do not describe every alternative associated with a decision or chance event. If the name of a decision or chance event appears with the name of only one state, the names are separated by a period. If the name appears with the names of two or more states, the name is followed by a period and a list of the state names. The state names are enclosed in braces and separated by commas.

```
gamble
  on Demand.{Low, Medium } then ...
  on Demand.High then ...
```

The above code is equivalent to the node being placed in the Decision Tree as shown in Figure 17-3.



**Figure 17-3. Node Placed with Mixed Grouping**

The keywords representing a decision's alternatives or a chance event's outcomes must start in the same column. Also, they cannot start in the same column as the keywords representing states of another event if you specify states for that event both earlier and later in the sequence section. DPL uses the columns in which these keywords occur to distinguish the states of one event from those of another. If there is any ambiguity about which event is associated with an event state, DPL associates the state with the event that appeared most recently in the program. The following sequence section is incorrect:

```
sequence: // incorrect code
decide
    to Get_Info.Yes and pay Info_Cost then
        gamble on Market_Forecast then decide
    to Release_Product.Yes and
        gamble on Demand and get Profit
    to Release_Product.No
    to Get_Info.No then decide
    to Release_Product.Yes and
        gamble on Demand and get Profit
    to Release_Product.No
```

The keywords for `Get_Info` and `Release_Product` should not start in the same column. The following is correct:

```
sequence: // correct code
decide
  to Get_Info.Yes and pay Info_Cost then
    gamble on Market_Forecast then decide
      to Release_Product.Yes and
        gamble on Demand and get Profit
      to Release_Product.No
  to Get_Info.No then decide
    to Release_Product.Yes and
      gamble on Demand and get Profit
    to Release_Product.No
```

#### 17.5.4 Duplicating Parts of a Tree (Performing a Subtree)

You can duplicate repeated sections of the sequence section with a `perform` link. You can define a `perform` link by defining labels in the sequence section of a DPL program and referring to them later in the program with the verb `perform`.

A label is an identifier followed by a colon (:). DPL determines which identifiers represent labels from their context in the sequence section. All labels must be unique. These labels are the equivalent to Perform Targets discussed in Chapter Section 7.1.

```
decide
  to Get_Info.Yes and pay Info_Cost then
    gamble on Market_Forecast then Rel: decide
      to Release_Product.Yes then
        gamble on Demand and get Profit
      to Release_Product.No
  to Get_Info.No then
    perform Rel
```

A label must appear in a DPL program before a `perform` clause that references it.

#### 17.5.5 Ending a Path in a Decision Tree

DPL provides two equivalent keywords for ending a path through a Decision Tree. The keywords are `quit` and `stop`. You can use whichever keyword makes the program easier to read.



In some circumstances, you must use `quit` or `stop` to end a path through a Decision Tree. In most circumstances, these keywords are optional.

### 17.5.6 Changing the Structure of a Tree with "Dont"

Occasionally, you may wish to temporarily remove a lottery from the sequence section of your model by attaching a `dont gamble` specification. To "dont" a gamble, simply type the word `dont` (no apostrophe!) in front of the keywords `gamble` or `take a chance on`.

```
chance A.{3} = {.2, .3} = 10, 20, 30;
sequence:
dont gamble on A and get A
```

This tree now has one path. The expected value is 23.

The `dont` prefix only applies to the lottery it precedes. It does not apply to other occurrences of the same lottery elsewhere in the sequence section.

### 17.5.7 Getting or Paying the Value of an Expression

You can associate a gain or loss with each branch (event state) in a Decision Tree with a `get/ pay` expression.

```
... and get 20000
... and pay @sum(Cost1, Cost2, Cost3)
... and receive Revenues - Costs
... and lose Damages
```

If you tell DPL to keep track of several attributes throughout a decision problem, you must provide one expression for each attribute whenever you specify a gain or loss. (Multiple attributes are described later in this section.) The number of expressions following the keywords `get`, `receive`, `pay`, or `lose` must equal the number of attributes specified in the program. If more than one expression follows one of these keywords, the expressions are separated by commas.

```
... and get Cash_Flow1, Cash_Flow2
```

### 17.5.8 Setting the State of an Event

At any point in the sequence of events, DPL allows you to set the state of an event with the keyword `set`. If you set the state (outcome) of a chance event, that outcome effectively has a probability of 1.0 at that point in the Decision Tree, regardless of the initial probabilities contained in the event's definition. If you set the state (alternative) of a decision, that alternative is

selected regardless of the expected values associated with other alternatives. (DPL does not even explore the other alternatives to see if they are optimal.) The only way you can include a controlled event in the sequence section of a program is by setting its state.

You can only set the state of a decision or chance event if it has not occurred previously along that path through the Decision Tree. Also, once you set the state of a decision or chance event, you cannot use keywords like `decide` or `gamble` to make the event occur at a subsequent point along the same path. In contrast, you can set the state of a controlled event several times along any path.

```
decide
  to Advertise.Yes and pay Ad_Cost then
    set Demand.High and get Profit
  to Advertise.No then
    gamble on Demand and get Profit
```

### 17.5.9 Using a Utility Function to Describe Risk Attitude

The expression for DPL's built-in exponential utility function is  $-\exp(-V/T)$ , where  $V$  is a value and  $T$  is the risk tolerance coefficient. To use it, you only need to specify the decision maker's risk tolerance coefficient.

```
value RiskTolerance = 100;
...
sequence:
  use tolerance RiskTolerance and
  decide
  to ...
```

The expression for the risk tolerance coefficient cannot contain values that depend on events. Also, this expression cannot contain the state function. Because the specification occurred at the beginning of the sequence section, it applies to the entire tree. It is possible (albeit unusual) to specify different risk tolerance coefficients for different paths through the tree.

You can define your own utility function for converting values into utiles. If you do so, you must also define the inverse utility function for converting utiles to values. The expressions for the utility function and inverse utility function use a dollar sign (\$) to represent the arguments of these functions. For example, the following defines a utility function that is equivalent to the built-in exponential utility function.

```
use utility -exp(-$/RiskTolerance),
  -RiskTolerance*log(-$)
```

The expressions for the utility function and inverse utility function cannot contain values that depend on events. Also, these expressions cannot contain the state function. The utility function defined above is the same as the default exponential utility function provided by DPL.

You are responsible for ensuring that the two functions are the inverse of each other. DPL will perform an analysis with functions that are not the inverse of each other, but the results may be meaningless.

### 17.5.10 Multiple Attributes, Objective Functions, and Constraints

You specify the number of attributes and an objective function (if any) immediately following the keyword sequence at the beginning of the sequence section of a program.

```
sequence (attributes = 2,  
         objective = $1 * Cost1 + $2 * Cost2):
```

The constant expression that represents the number of attributes must produce a number between 1 and 1024, after truncation to an integer. If the number is not an integer, DPL will truncate it. If you do not specify the number of attributes, DPL assumes there is only one attribute.

DPL provides predefined identifiers for the attributes, which you can use in the expression for the objective function. In this expression, \$1 represents the first attribute, \$2 represents the second attribute, and so on. If you do not provide an expression for the objective function, DPL uses a default function equal to the sum of the attributes. If you provide an expression, it need not contain an identifier for each attribute, and any attribute identifier can appear more than once. When the identifier for an attribute is missing from the expression, that attribute does not affect the optimal decision policy. The predefined identifier \$0 may be used to refer to the joint probability of the path for which the function is being evaluated.

When you specify more than one attribute for a decision problem, you must provide an expression for each attribute whenever you associate a gain or loss with an event state. The expressions follow the keywords `get`, `receive`, `pay`, and `lose`. The expressions are separated by commas.

The following is an example of a DPL program that specifies two attributes:

```

value P = 0.02;
value MeltCost = 1.0e6;
decision Mode.{Normal,Emergency};
value OperatingCost | Mode = 20,50;
chance SystemFailure = {0.01};
chance CoreMelt | SystemFailure,Mode
    = {0.1}, {P}, {0.01}, {P};
chance Warning.{Clear,Ambiguous,None} | SystemFailure
    = {0.75,0.2}, {0.05,0.2};
sequence (attributes = 2,
    objective = $1 * MeltCost + $2):
gamble on Warning then
decide on Mode and
pay prob(CoreMelt.y),OperatingCost

```

DPL compiles the expression for the objective function before it processes the Decision Tree. This means that the expression cannot depend on the states of any events. If the expression contains values, they cannot depend on the states of events. Also the expression cannot contain the state function, which returns a number equal to the state of an event.

When there is only one attribute, DPL's default objective function is equal to that attribute. (The default objective function is the sum of the attributes, which is equal to the first attribute when there is only one.) However, you can specify an expression for the objective function even where there is only one attribute. This allows you to multiply all the gains and losses by a scale factor before DPL uses them in an analysis. It also lets you use any expression to transform the sum of the gains and losses before applying a utility function (if any) and determining the optimal policy.

The phrases `attributes =`, `objective =` and `constraint =` are optional.

```
sequence (2, $1*Cost1 + $2*Cost2):
```

You can also specify a constraint function.

```

sequence (attributes = 2,
    objective = $1*Cost1 + $2*Cost2,
    constraint = $1*Cost1 > 20000 ?
    halt(100000): $1*Cost1 + $2*Cost2):

```

Because the constraint is only checked before evaluating a node in the tree, and not at the endpoints, you may also wish to put the constraint expression in the objective function and copy it in the constraint function with an asterisk:

```
sequence (attributes = 2,
  objective = $1*Cost1 > 20000 ?
  halt(100000): $1*Cost1 + $2*Cost2,
  constraint = *)
```

## 17.6 Advanced Techniques for Programs

---

If you do a lot of programming with DPL, you may find yourself working with many programs or copying some parts from one program to another. DPL offers several advanced techniques for working with your program files that can simplify this activity. One technique allows you to break your programs into multiple parts, which is especially convenient if you have certain definitions (constants, for example, or a converted spreadsheet model) that are used repeatedly. Another technique allows you to include several variations in a single program file and choose at compile time which variant should be compiled. All of these techniques involve the use of compiler directives, which are instructions for the compiler which you embed in a program file.

If you are using these advanced DPL programming techniques and cannot find answers to your questions in this section or in DPL online Help, please contact technical support.

### 17.6.1 What is a Compiler Directive?

DPL code is not procedural. This means that a DPL program simply describes a decision problem, but does not include instructions on how to solve the problem. In this way, DPL is more like a spreadsheet than C++ or Java, which are procedural.

However, although a DPL program does not contain instructions for solving a decision analysis problem, it can contain directions to the compiler that control the process by which the compiler reads a DPL program and prepares it for analysis by the DPL engine.

These instructions are called compiler directives. Every compiler directive begins with a # symbol in the first column of a line of a program file. There are compiler directives that:

- Include additional programs in a compiled program
- Leave parts of a program file out of the compiled version
- Control optimization and spreadsheet evaluation

### 17.6.2 Including the Contents of One DPL™ Program in a Second Program

DPL code includes a compiler directive that allows a DPL program to be broken into several parts. This can allow you to maintain a list of constants that you use in several models, or use a single value model in several programs, or update a spreadsheet model, convert it to code again, and run a program without editing the DPL models or program.

The compiler directive to include a program is:

```
#include programname
```

The instruction may be coded in DPL programs and command procedures. As with all DPL compiler directives, this instruction must begin in the first column of the line. In addition, there can be only one space between `#include` and `programname`.

When the compiler encounters the `#include` directive during compilation or command execution, it will suspend processing of the current input model or program and take its input from the program specified as the argument of the directive. When the compiler completes the processing of the included program, it will return to the suspended program and continue. Included programs may be nested to any level; however, recursive `#include` directives are not allowed.

If the compiler encounters an error in an included program, DPL will place the cursor at the error location.

When compiling a program, the DPL compiler will ignore duplicate variable definitions encountered in an included file and will use the first declaration of the variable found.

### 17.6.3 Conditional Compilation

Conditional compilation involves an if-then-else instruction for the compiler. When the compiler encounters an `#if` expression statement, it evaluates the expression. If the expression is true (does not equal 0), it compiles the code immediately following until it encounters a `#else` or `#endif` statement. If it finds a `#else` statement, it skips the code that follows the `#else` and searches for a `#endif` statement.

If the `#if` expression is false (equals 0), then the compiler searches for the first occurrence of a `#else` statement or a `#endif` statement. It skips the code between the `#if` statement and the `#else` or `#endif` statement. If it finds a `#else` statement first, it compiles the code between `#else` and the next `#endif`. Once the compiler finds a `#endif` statement, it starts compiling again.

### 17.6.4 Setting Compile-Time Options from Within a Program

DPL allows you to control the settings of compile-time options from a DPL program file. Although these options can also be set within the Run Settings dialog (Home | Run | Settings), this feature can be very useful if forgetting to set an option results in long run times or unexpected results. Like all compiler directives, the `#options` option list directive must begin in the first column of a line. In general, this directive should be used only once and near the beginning of a program file.

The option list consists of a sequence of letters separated by commas, indicating the options to be set. There are five compile-time options that can be set in an options directive:

- `e` (for expression optimization)
- `s` (for fast sequence evaluation)
- `l` (for lottery optimization)
- `x` (Excel lookup function convention)
- `i` (ignore case when comparing strings)

If the letter is preceded by a minus sign, the option is turned off. If the letter is preceded by a plus sign (or nothing) the option is turned on. These settings override the settings in the Run Settings dialog. DPL will use the settings within the Run Settings dialog to set options that are not included in the option list. The letters may be in either upper or lower case.

### 17.6.5 Command Window

The Command Window allows you to view any number or calculation within any scenario in a model. You can view any number in a model, including values; probabilities; series and array elements; and expressions. You can also temporarily change any value in a model. These two capabilities provide a powerful and flexible environment for testing and debugging.

To open a Command Window, select Home | Workspace | Add to WS | Command Program. You will see a blank window in which you can enter various types of commands to execute. A Command tab appears and becomes active on the ribbon as well. The results of your commands are displayed in the Session Log. Refer to the DPL online Help for detailed descriptions of specific commands.



## 18. Running Excel Macros from DPL™ (Enterprise only)

### 18.1 When to Use Excel Macros

---

Most spreadsheet models are built in such a way that the outputs (e.g., NPV) are updated as part of a normal Excel recalculation. However, some models may include calculations that are difficult or impossible to express solely in terms of Excel formulas but can be readily programmed as Excel Visual Basic for Applications (VBA) macros. For this reason, DPL provides support for running an Excel macro in lieu of the Excel Calculate command normally sent on each path through the Decision Tree.

### 18.2 Tutorial: Building a DPL™ Model for a Spreadsheet Updated by a Macro

---

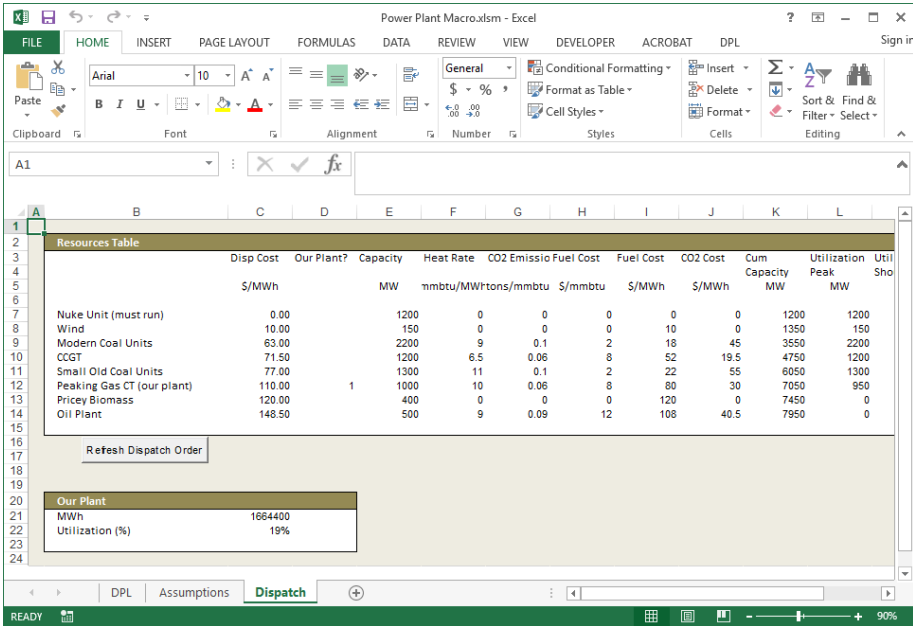
In this example, assume you are the owner of a gas-fired combustion turbine electricity generating plant. The plant is part of a system consisting of many generating stations using various energy sources: nuclear, wind, coal, gas, etc. The system operator dispatches these power plants efficiently based on their variable costs per generated megawatt hour (MWh). The lowest variable cost plants run nearly all the time whereas the more expensive ones run only during periods of peak demand. The plant you own is a "peaking" unit that typically runs about 20% of the time.

For planning purposes, you would like to estimate how many hours the plant will be operating next year. The problem is made difficult by the uncertainty in fuel prices as well as the level of a recently enacted carbon tax.

#### 18.2.1 The Dispatch Spreadsheet

Assume you have a spreadsheet that approximates the logic employed by the system operator in dispatching the power plants in the system.

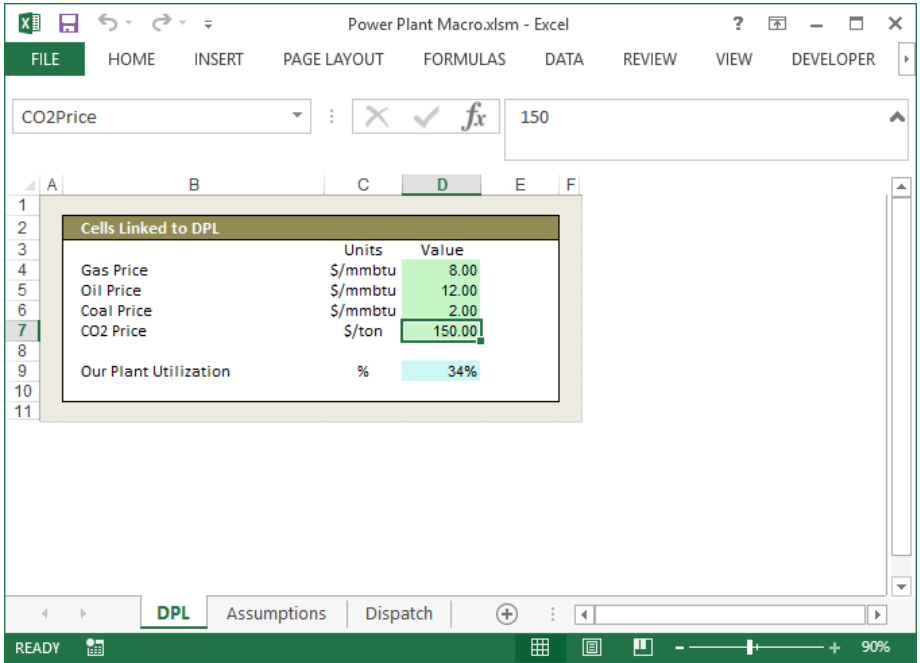
⇒ Open PowerPlantMacro.xlsm and select the Dispatch tab.



**Figure 18-1. Power Plant Dispatch Sheet**

Under base case assumptions, the plant falls in the bottom half of the dispatch order and runs about 19% of the time (cell C22). The units immediately ahead of the plant are old, less efficient coal-fired plants. A high carbon tax might mean that your plant runs more since it would push up the costs of plants currently above yours to a level that's more than your costs.

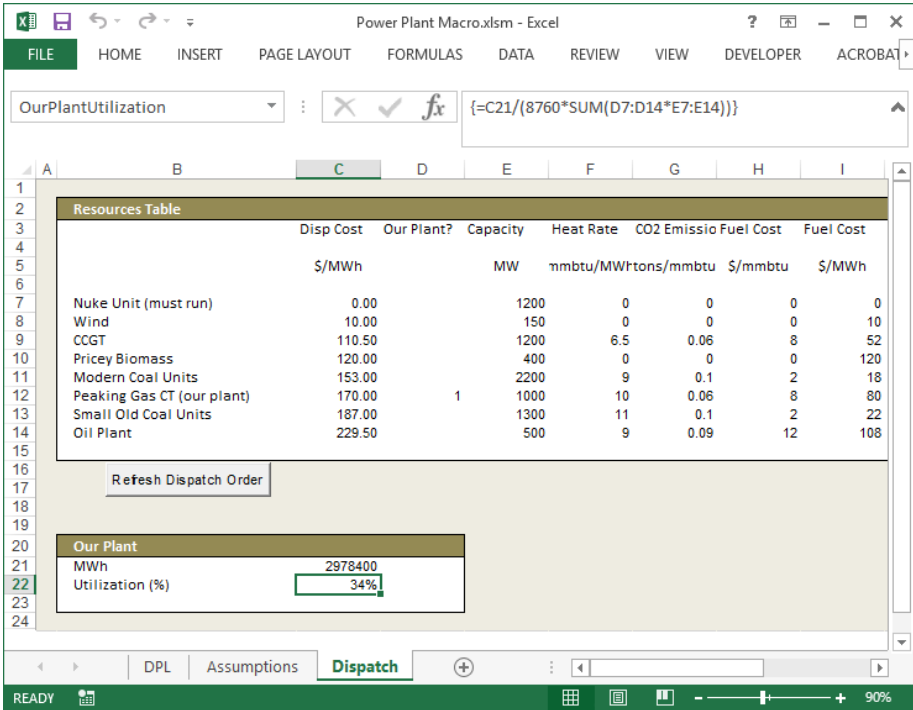
- ⇒ Select the DPL tab.
- ⇒ Change the green CO2 Price cell (cell D7) to 150.



**Figure 18-2. Power Plant DPL Sheet**

- ⇒ Select the Dispatch tab.
- ⇒ Press the Refresh Dispatch Order button.

Note that with the higher carbon tax the plant moves up in the dispatch order and runs 34% of the time.



**Figure 18-3. Power Plant Dispatch Sheet Updated**

The Refresh Dispatch Order button runs a macro that sorts the power plants by their variable costs to update the dispatch order.

- ⇒ Click Developer | Code | Visual Basic. If the Developer tab isn't visible in the Excel Command Ribbon go to File | Options | Customize Ribbon and click the checkbox next to Developer in the Customize the Ribbon section on the right.
- ⇒ Once Visual Basic is open, in the left-hand pane, double-click on Module1.

The SortResources() macro is displayed in the Visual Basic Editor.

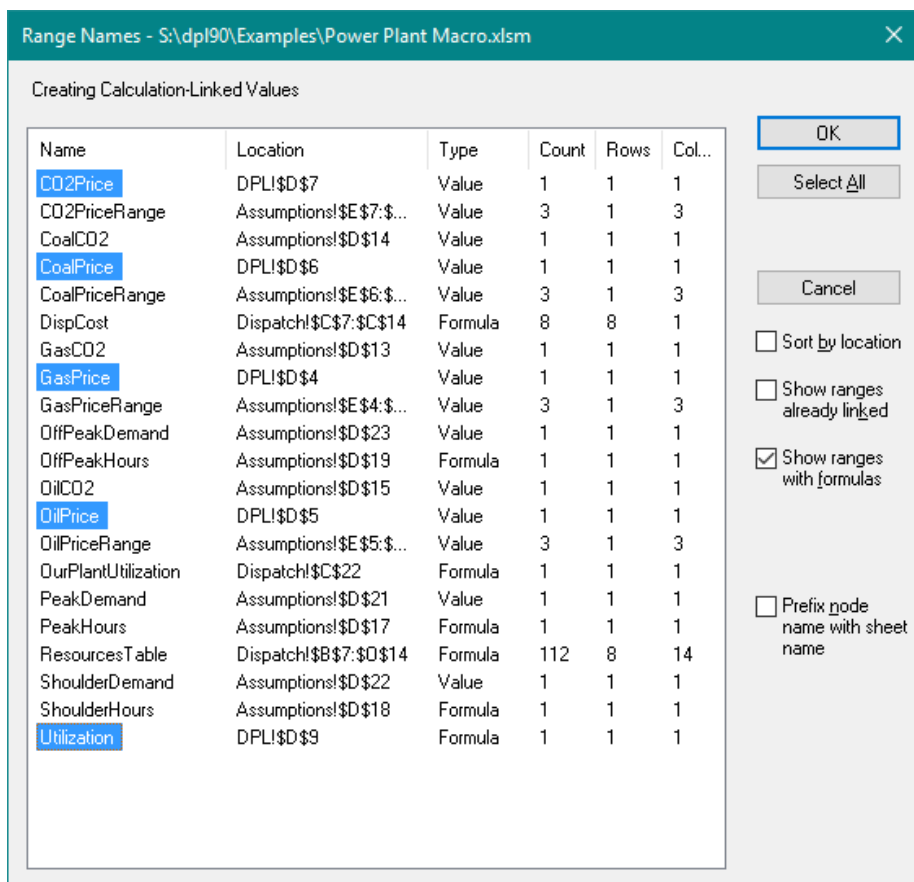
```
Sub SortResources ()
    Calculate
    Worksheets("Dispatch").Activate
    Names("ResourcesTable").RefersToRange.Select
    Selection.Sort Key1:=Names("DispCost").RefersToRange
    Calculate
End Sub
```

- ⇒ Close the Visual Basic Editor.
- ⇒ Select the DPL sheet and change CO2 Price back to 50.

## 18.2.2 Building the DPL™ Model

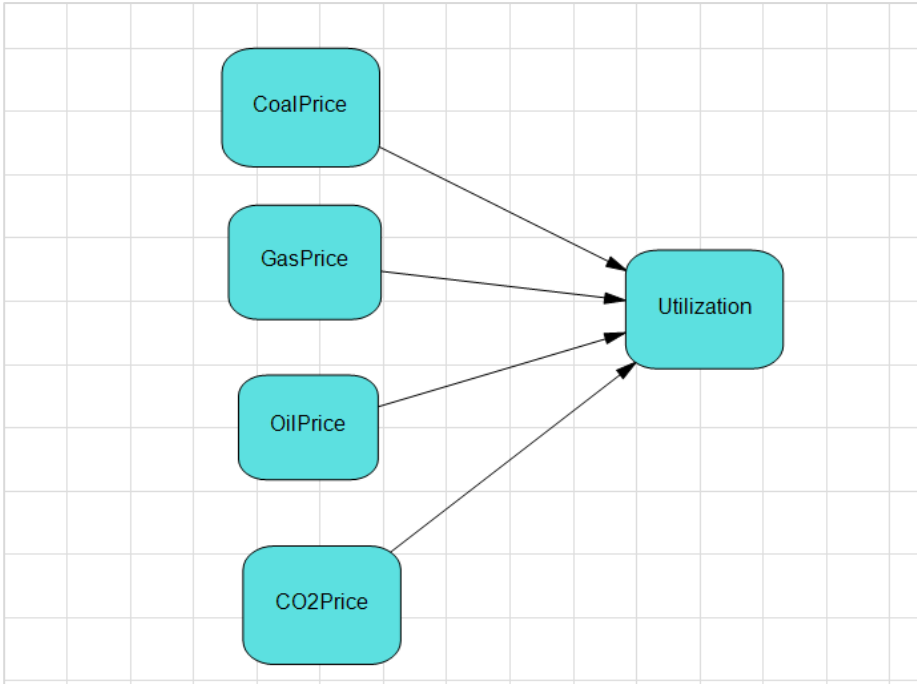
You will now create a DPL Model for the spreadsheet.

- ⇒ Start DPL Enterprise.
- ⇒ Click Influence Diagram | Node | Linked Node
- ⇒ Press the Browse button and find Power Plant Macro.xlsm, then click Open and then OK.
- ⇒ In the Range Names dialog, select CO2Price, CoalPrice, GasPrice, OilPrice and Utilization (see Figure 18-4), then click OK.



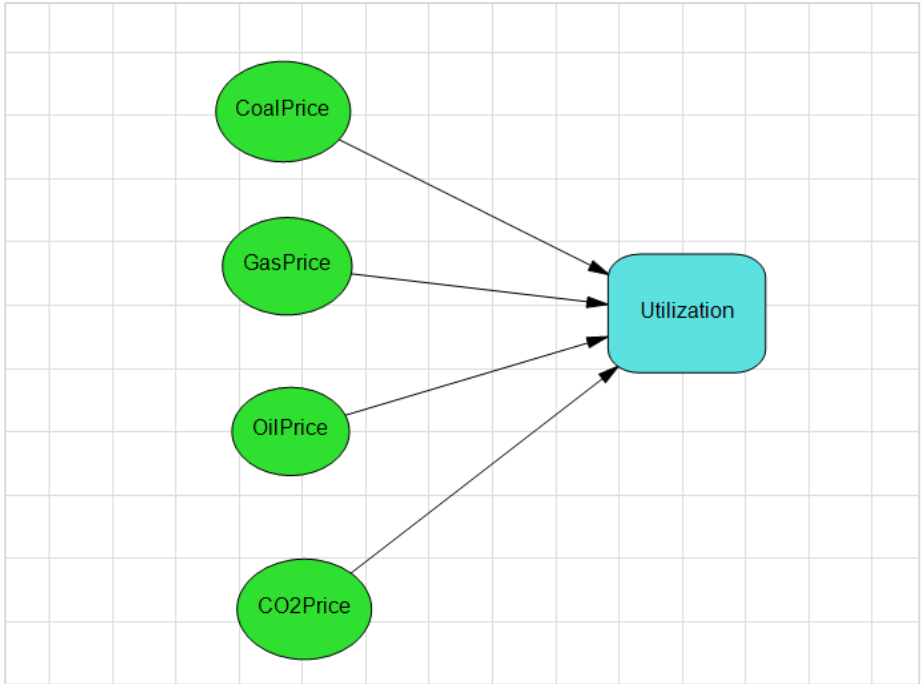
**Figure 18-4. Range Names Dialog**

- ⇒ Click Influence Diagram | Arc | From Formulas.
- ⇒ Move your nodes so your Influence Diagram looks like Figure 18-5.



**Figure 18-5. Deterministic Influence Diagram**

- ⇒ Select the nodes CoalPrice, GasPrice, OilPrice and CO2Price (hold down the Ctrl key as you click each to select all the nodes at once).
- ⇒ Drop-down the Influence Diagram | Node | Change To split button and select Discrete Chance from the list.
- ⇒ Click in whitespace to de-select the nodes. See Figure 18-6.



**Figure 18-6. Probabilistic Influence Diagram**

You now need to provide data for the chance nodes. The Assumptions sheet in the spreadsheet has Low-Nominal-High ranges for each of them. You'll use Initialization Links (see Appendix A.3) to tell DPL to use the range data in the spreadsheet.

- ⇒ Double-click on CoalPrice to bring up the Node Definition dialog.
- ⇒ Switch to the Links tab and in the *Initialization links* section click Microsoft Excel. See Figure 18-7.

Node Definition: CoalPrice

General Data Links

Calculation links

These links are recalculated throughout the run. They may be either inputs used in the linked spreadsheet's calculations (e.g., unit sales) or calculated results from the spreadsheet (e.g., NPV).

None (local)
  DPL Program
  Microsoft Excel

Workbook:

Sheet/Cell:

This is a driver node. Its value will be sent to Excel each time it changes during the run.

---


Initialization links

These links are for connecting to a spreadsheet, database or program that contains data (values and/or probabilities) which will be used in the Data tab to initialize nodes. These links are refreshed once at the start of each run.

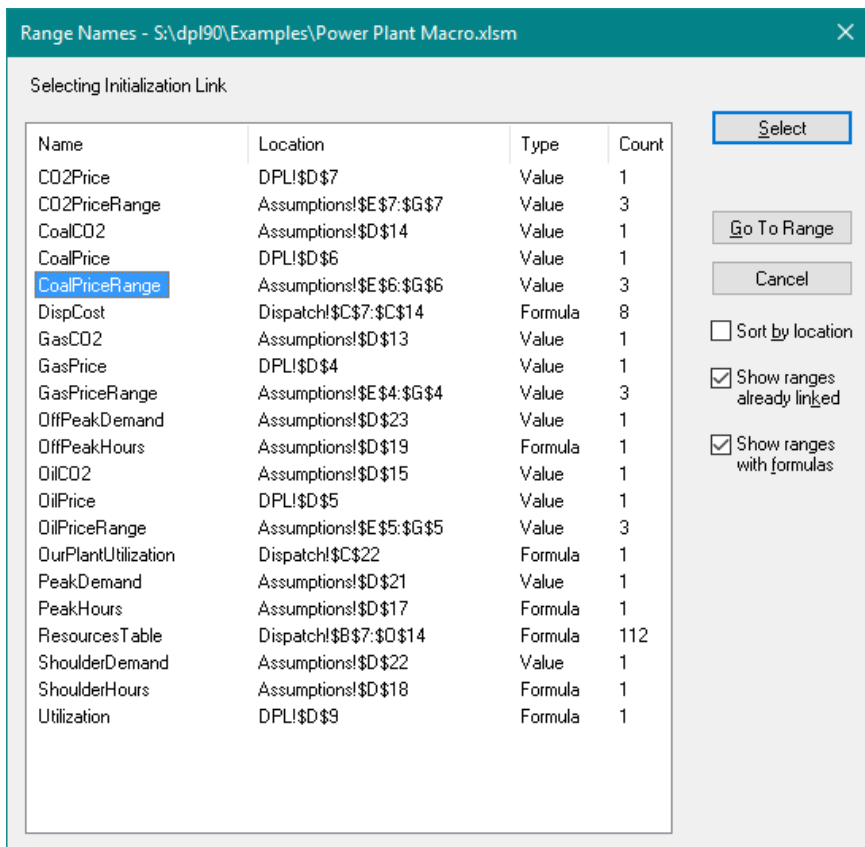
None (local or DPL program)
  Microsoft Excel
  Database

Workbook:    Same as calc links workbook

**Figure 18-7. Node Definition Links for CoalPrice**

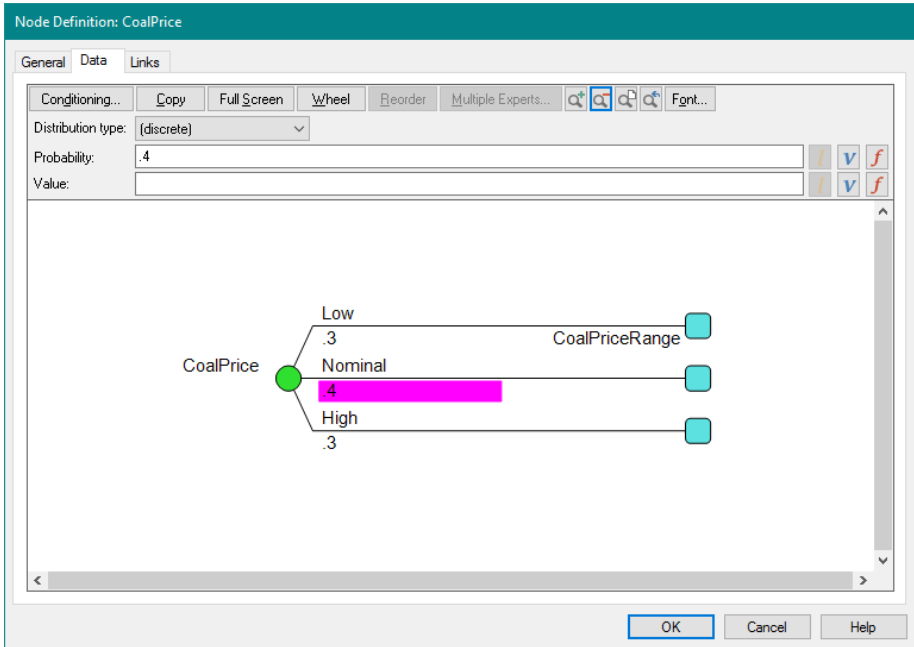
- ⇒ Switch to the Data tab.
- ⇒ Click the link button (  ) for the Value edit box (second link button down).
- ⇒ In the first column, select CoalPriceRange (see Figure 18-8) and then click the Select button.





**Figure 18-8. Range Names Dialog**

You have now set up the initialization links so that the CoalPrice chance node will use the three values in the spreadsheet for its Low, Nominal and High branches. You will use the default probabilities of .3, .4, .3 for this example. See Figure 18-9.



**Figure 18-9. Node Definition Data for CoalPrice**

- ⇒ Click OK to close the Node Definition dialog.
- ⇒ Repeat the preceding steps to establish initialization links for the GasPrice, OilPrice and CO2Price nodes, using GasPriceRange, OilPriceRange and CO2PriceRange, respectively.
- ⇒ Save your DPL model.

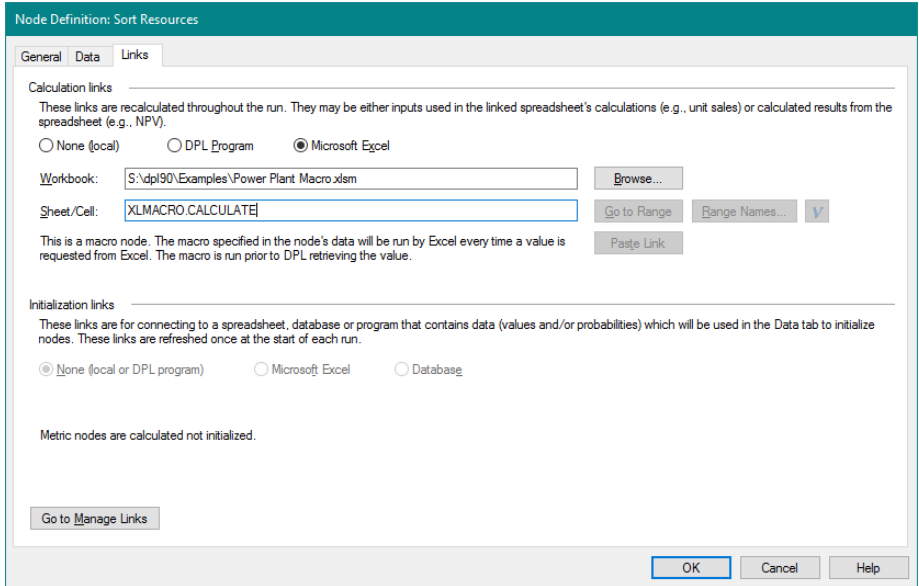
### 18.2.3 Connecting the Calculation Macro

The DPL model is now set up and could be run, however the results would not be correct since a simple recalculation wouldn't sort the table. You need to make the SortResources macro run at the end of each path in the Decision Tree. To do that, you will create a special macro node. As you will see in the following, you indicate to DPL that the node is a macro node on the Links tab. You specify the macro to be run on the Data tab.

- ⇒ Create a new, local value node (Influence Diagram | Node | Add).
- ⇒ In the General tab, name the node Sort Resources.
- ⇒ Switch to the Links tab and in the *Calculation links* section click Microsoft Excel.

DPL fills in the Workbook edit box for you.

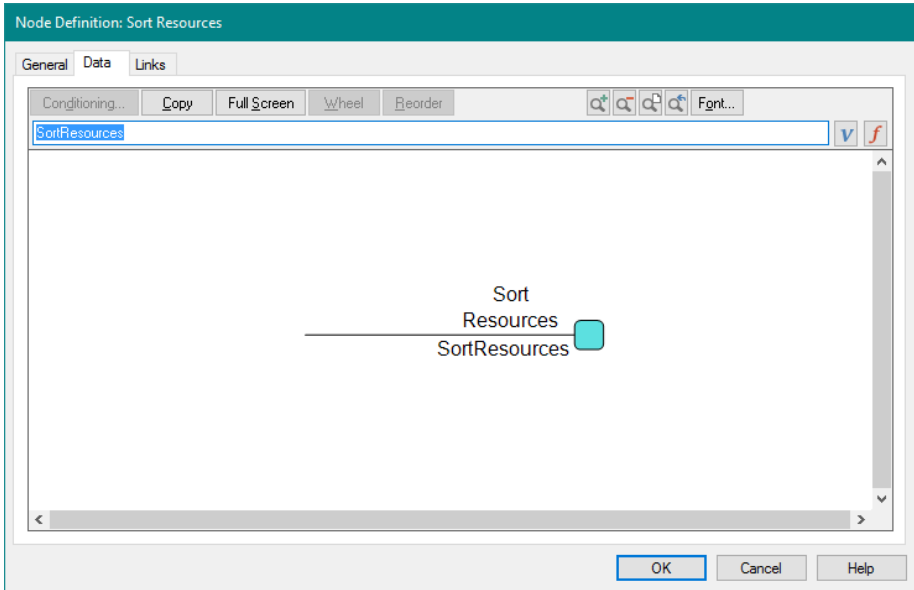
⇒ In the Sheet/Cell edit box, type in "XLMACRO.CALCULATE". See Figure 18-10.



**Figure 18-10. Node Definition Links for the Sort Resources Macro Node**

DPL recognizes the special cell name as the code for a calculation macro node as indicated by the note beneath the *Sheet/Cell* edit box. You will now specify the name of the macro to run on the Data tab.

⇒ Select the Data tab and type SortResources. See Figure 18-11.

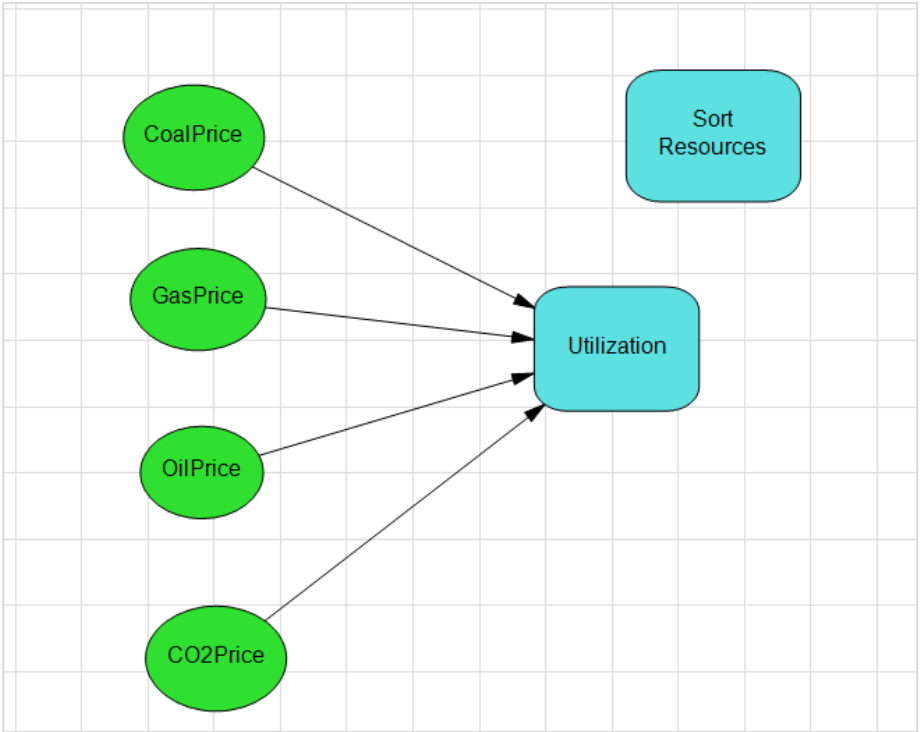


**Figure 18-11. Node Definition Data for the Sort Resources Macro Node**

SortResources is the name of the update macro.

Note: Calculation macros must be VBA Subs without any parameters. If you need to pass parameters to your macro, just add one or more DPL driver nodes and have the macro check the values of the cells to which those nodes are linked. The macro name is case sensitive. The data that you specify in the node for macro name must match exactly the macro name in Excel VBA.

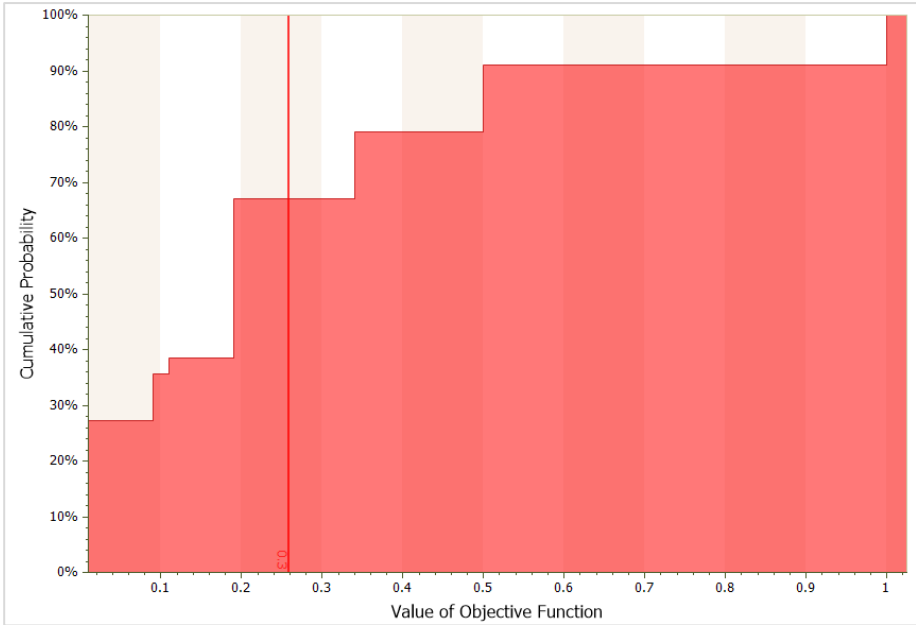
- ⇒ Click OK to close the Node Definition dialog. Your model should now look similar to Figure 18-12.



**Figure 18-12. Influence Diagram with Sort Resources Macro Node**

You are now ready to run the model.

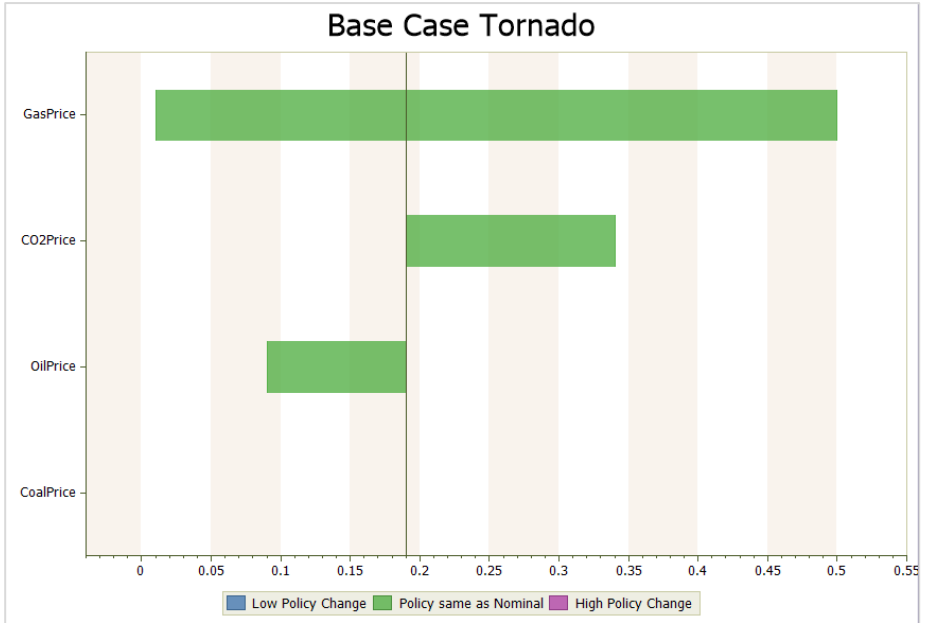
- ⇒ In the Home | Run group, make sure Risk Profile is checked and uncheck Policy Tree.
- ⇒ Click Home | Run | Decision Analysis.



**Figure 18-13. Risk Profile**

The Risk Profile in Figure 18-13 shows a broad range of outcomes. You can run a tornado diagram to see which of the chance nodes is contributing the most uncertainty.

- ⇒ Drop-down the Home | Sensitivity | Tornado split button and select Base Case from the list.
- ⇒ Click OK to accept the default Low/Nominal/High assignments.



**Figure 18-14. Tornado Diagram**

Figure 18-14 indicates that GasPrice is the most sensitive variable, which is what you expect since you're running a gas-fired plant, but CO2 and Oil prices are also significant. Coal price is not sensitive. As it is modeled, the risk in the dispatch cost of a coal plant is primarily driven by CO2 emissions.

# 19. Multiple Experts (Enterprise only)

## 19.1 Why Use Multiple Experts?

---

Decision analysis requires the assessment of probability distributions using data, expert judgment, or most often, a combination of these. Probability distributions are defined by the DPL analyst, but to acquire the necessary knowledge and/or data, the analyst often needs to consult experts.

Different experts often have different opinions. For chance nodes with two states (also known as binomial chance events), DPL provides a built-in capability to aggregate probability assessments from several experts. This chapter explains and demonstrates how this multiple experts feature works.

For purposes of this chapter, the quantity being assessed (a probability of a particular event occurring) will be called a **likelihood** to avoid confusion with the other probabilities involved in the calculations.

Probability assessments are often difficult and experts may have significantly differing views. Given a variety of experts and their assessments, you would like to come up with some sort of weighted average likelihood to use in the analysis. One possibility is to simply apply weights to each expert and compute a straightforward weighted average. However, this method does not directly incorporate any measure of the degree of confidence in each expert's assessment. It also does not capture the overlap of knowledge across experts.

DPL's aggregation method uses data provided by you to calculate the weights and the overall expected likelihood. The weights used by DPL are determined by the reliability of the expert (measured by the assessments the expert provides) and the amount of shared information among the experts (measured by an overlap factor). DPL's method is simple and quick to use.

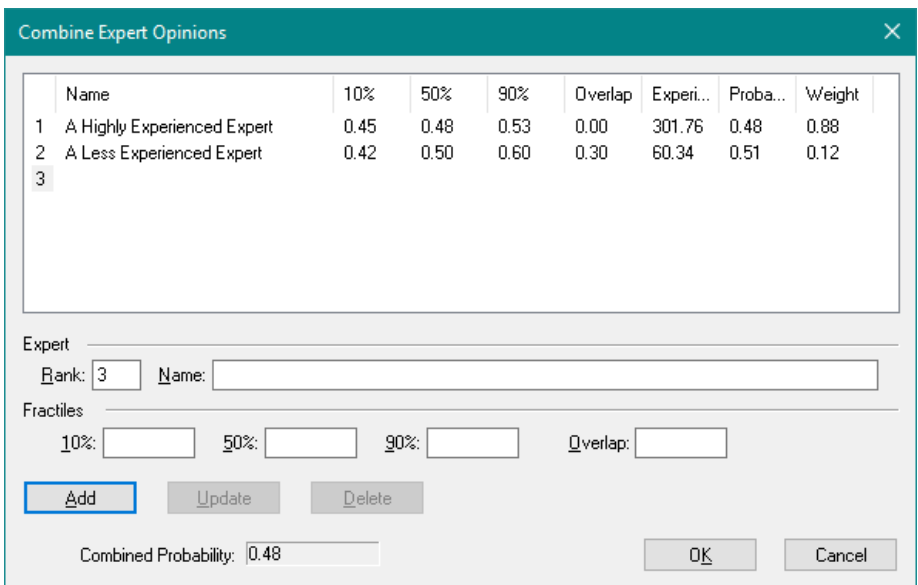


## 19.2 Overview of DPL's™ Multiple Experts Feature

Through a process of careful questioning, a distribution of likelihoods is extracted from each expert. To use DPL's multiple experts feature, each expert must provide his/her assessment of the 10th, 50th, and 90th percentiles for the likelihood in question. You rank the experts from most "reliable" to least and enter the 10-50-90 percentile values (referred to as fractiles) from the distribution for each expert.

An overlap factor is also required for all experts entered after the first (most reliable) expert. This quantity represents the amount of shared information the experts are believed to have.

DPL approximates each expert's distribution as a Beta distribution and computes an expected value of the weighted average of all the distributions.



**Figure 19-1. DPL Dialog for Combining Expert Opinions**

Look at the top row of the Combine Expert Opinions dialog in Figure 19-1. The following headings appear: Name, 10%, 50%, 90%, Overlap, Experience, Probability, and Weight. Each of these is a quantity used to

calculate the overall likelihood of the event. The fractiles and overlap factors are entered by you and the other quantities are calculated by DPL.

In Figure 19-1, a highly experienced expert and a less experienced expert have each assessed likelihoods for an uncertain event. The less experienced expert has provided a wider range around his or her 50% (median) point estimate. The combined probability for the two experts is 0.48.

The subsections that follow provide descriptions of each quantity in this dialog. The last section of this chapter provides a brief tutorial on using DPL's Multiple Experts feature.

### 19.2.1 The Overlap Factor

The overlap factor is provided by you. It is a number between zero and one representing the fraction of the information provided by that expert that overlaps the information already represented by other experts. For example, if two academic experts are entered who are in the same department, have read the same books, and have attended the same seminars, the second one entered may have an overlap factor close to 1. An overlap factor of zero implies that the expert with zero overlap has only information or data that is entirely unknown to other experts; this is not typically the case.

Determining the overlap factors is a challenge left to you. It is important to consider using overlap factors in order to avoid over-representing any one source of information.

### 19.2.2 The Experience Index

The experience index is calculated by DPL. This number represents the amount of information in each expert's assessment and is based on the 10-50-90 fractiles.

This quantity is most easily explained by the "colored balls in an urn" model of the Beta distribution. Suppose the quantity being assessed is the probability that you will pick a red ball from an urn with an unknown percentage of red balls. An expert draws  $n$  balls from the urn, where  $n$  is different for each expert. Based on the size of the sample drawn ( $n$ ) and the portion of the balls drawn that are red, the expert constructs his or her own distribution of the likelihood of drawing a red ball. As  $n$  increases, the accuracy of the expected value of the likelihood increases.

In DPL, the quantity  $n$  is labeled the experience index. Note that  $n$  is determined by the distribution, rather than the distribution being determined by  $n$ . It is an estimate of the total "n" (or alpha plus beta) in a Beta distribution that approximately corresponds to the fractiles. The precise formula for the experience index is:

$$n = \frac{E(p) - E(p^2)}{E(p^2) - (E(p))^2}$$

where  $p$  represents the probability (likelihood) being assessed, and  $E(x)$  is the expected value of  $x$ .

### 19.2.3 The Probability

The probability column contains the expected value of the likelihood of the event for each expert, calculated from the 10-50-90 fractiles.

### 19.2.4 The Weight

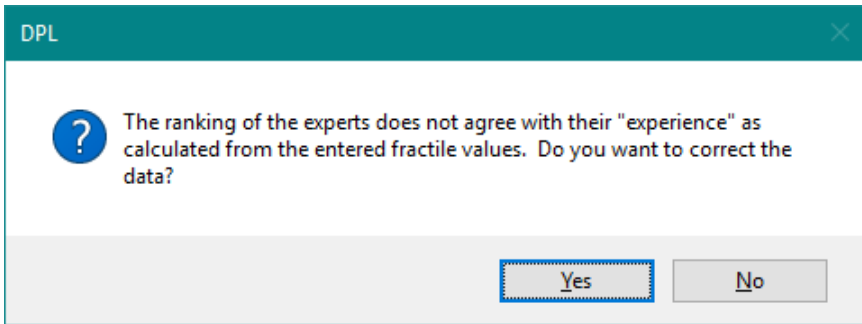
The weights for all experts should add up to 1. The weight is based on the overlap factor and the experience index. First, each expert's experience index is adjusted by subtracting off the amount of overlap (multiplying by  $1 - \text{overlap factor}$ ). Then the weight for the  $k$ th expert represents the fraction of all total experience that is attributable to that expert, that is:

$$w_k = \frac{n_k}{\sum n_i}$$

where  $n_i$  is the adjusted experience index of the  $i$ th expert.

### 19.2.5 Ranking

The experts should be ranked in order of descending reliability and entered in this order, i.e., the most reliable should be entered first. This way the overlap factor will apply to the less reliable experts. If the experience indices are not in descending order when all data has been entered, DPL will put up a warning box. See Figure 19-2.



**Figure 19-2. DPL Warning About Expert Rankings**

In general, more reliable experts should have higher experience indices. DPL is checking to make sure the entries are what you intended.

If an expert believed to be less experienced has a very high experience index, it may be that this expert is not well calibrated and has provided 10-50-90 fractiles that are closer together than they should be given his/her knowledge. You must exercise judgement to ensure that the opinion of an overconfident expert is not overweighted.

## 19.3 Tutorial: Using Multiple Experts to Assess Early Product Approval

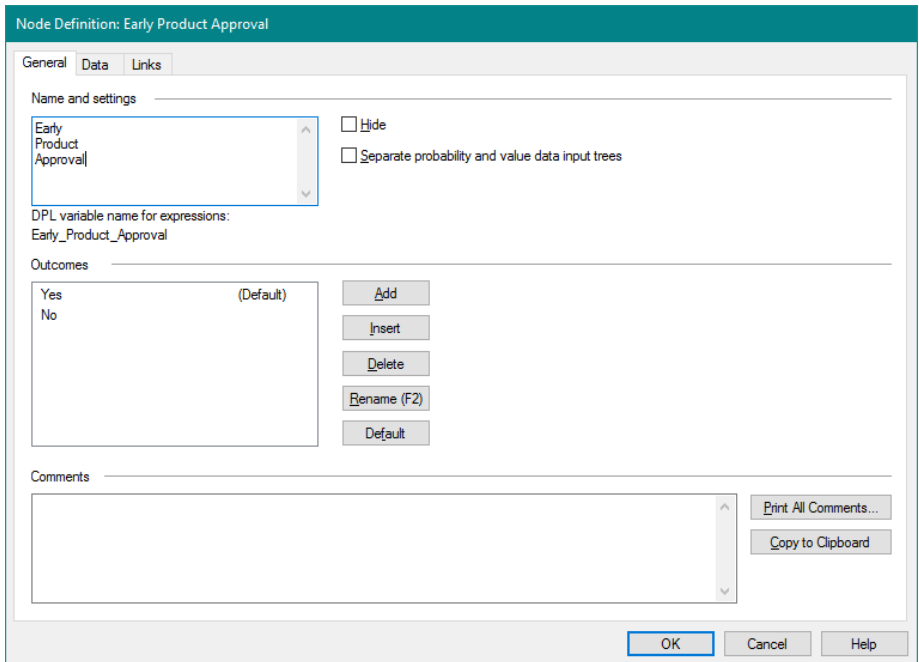
---

Suppose you are a decision analyst at a firm that is preparing to launch an exciting new product early next year. Plans are in place for an expensive product launch, including millions of dollars in advertising and promotions and a large kickoff meeting for the global sales team. Government regulators were on track to approve the product at the end of the calendar year, so the product launch has long been scheduled for the first quarter of the next year. You've done an extensive DPL analysis of decisions surrounding the product launch but the launch timing was never in question.

You just received a call with new information. Because of the national election in November, there are rumors that the regulators in your industry are going to speed up approvals during the summer and fall which could lead to your product being approved and launched a few months early (i.e., only a few months from now). If this happens, the promotional events and other pre-launch investments may need to be accelerated at significant cost. You need to adjust your DPL analyses accordingly and the NPV of the product will change.

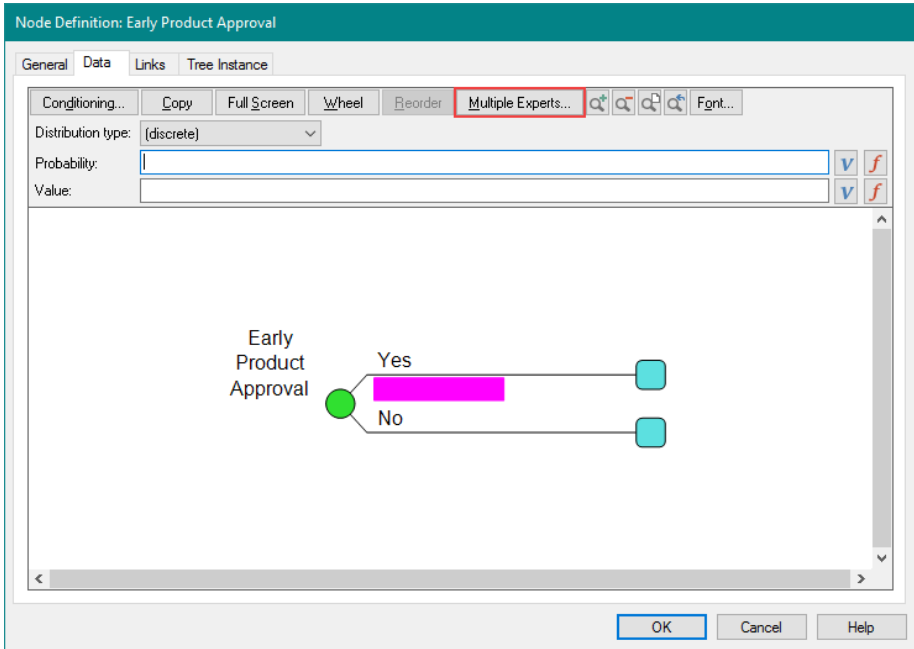
You decide to consult a few external and internal experts to better understand the likelihood that the product will be approved (and launched) in the current calendar year so that you can incorporate this new uncertainty into your models.

- ⇒ Start DPL Enterprise (if it isn't already open).
- ⇒ If necessary, save your previous Workspace and close it.
- ⇒ Select File | New to open a blank Workspace.
- ⇒ Within File | Options | Outputs under *General number formatting* increase the number of decimals places to 2.
- ⇒ Create a discrete chance node named Early Product Approval.
- ⇒ Modify the default outcomes so that the node has two outcomes: Yes and No, as in Figure 19-3.



**Figure 19-3. Chance Node Definition with Two States**

- ⇒ Click the Data tab and delete any probabilities remaining on the Yes and No branches.
- ⇒ The dialog should look like Figure 19-4. Note that the Multiple Experts button is enabled.



**Figure 19-4. Node Data with Multiple Experts Enabled**

⇒ Click the Multiple Experts button. The Combine Expert Opinions dialog appears.

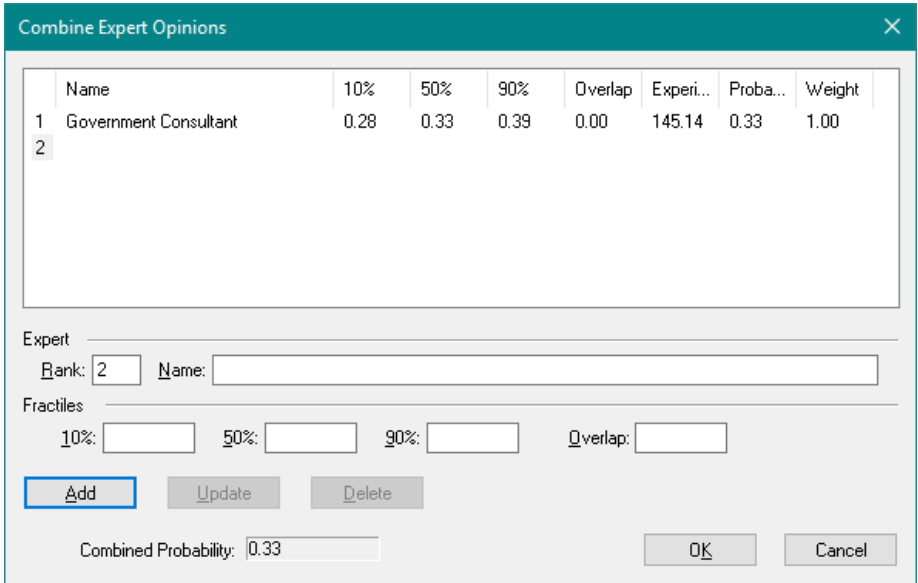
You will enter data for three experts you interviewed about the likelihood of early product approval.

⇒ In the *Name* edit box for the first expert, enter Government Consultant.

⇒ Click in the three *Fractiles* edit boxes, and enter the values 0.28, 0.33, and 0.39.

⇒ Leave the overlap blank (it will default to zero since this is the first expert).

⇒ Click the Add button. The dialog should look like Figure 19-5.



**Figure 19-5. Combine Expert Opinions Dialog with First Expert's Data Entered**

Note that DPL has calculated the experience index to be about 145 for this expert; this is analogous to saying the expert has about 145 observations from which to draw conclusions. DPL has also calculated a weighted likelihood of 0.33 for this expert.

DPL prompts you to enter the second (rank 2) expert.

- ⇒ For the rank 2 expert, enter the name In-house Expert. Enter the following fractiles: 0.2, 0.28, 0.4.
- ⇒ Enter 0.2 for the overlap. You believe that your in-house expert (a former government regulator) has about 20% overlap, i.e., 80% "new" information compared with the government consultant.
- ⇒ Click the Add button. DPL updates the calculations.
- ⇒ Before examining results, add the third expert, named Third Opinion. The fractiles are: 0.15, 0.25, and 0.5 and the overlap is 0.5.

Click the Add button. The final results are shown in

- ⇒ Figure 19-6.

Combine Expert Opinions
✕

	Name	10%	50%	90%	Overlap	Experi...	Proba...	Weight
1	Government Consultant	0.28	0.33	0.39	0.00	145.14	0.33	0.80
2	In-house Expert	0.20	0.28	0.40	0.20	39.37	0.29	0.17
3	Third Opinion	0.15	0.25	0.50	0.50	11.25	0.29	0.03
4								

Expert

Rank:  Name:

Fractiles

10%:  50%:  90%:  Overlap:

Combined Probability:

**Figure 19-6. Final Combined Expert Opinions**

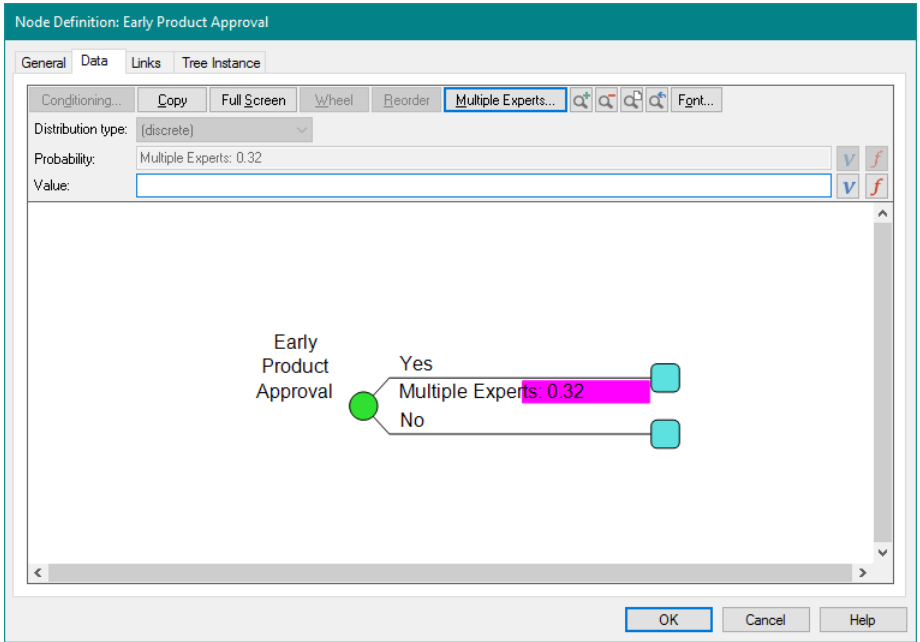
DPL has calculated that the In-house expert has an experience factor of about 39. The Third Opinion expert has an experience factor of only about 11.

The weights applied to the three experts are calculated and shown in the far right column. The government consultant's assessment gets nearly 80% of the weight, while the other two experts get about 17% and 3%, respectively. The overall combined probability (likelihood) is 0.32. You decide to use this probability in your analysis and to also conduct sensitivity analysis to see how decision sensitive it is.

⇒ Click OK to accept the probability as it is.

As shown in Figure 19-7, DPL has applied the combined probability to this chance node and noted that it comes from the Multiple Experts feature. You can proceed to use this node in your model as you would use any other two-state chance node.





**Figure 19-7. Node Data with Multiple Experts Data Defined**

Note: As shown in Figure 19-7, when multiple experts data is in place, you can no longer enter probabilities directly. In order to remove the multiple experts data from the node, you need to go back into the Multiple Experts dialog and delete each expert. To do this, you select each expert by clicking on his/her number in the far left column and then click Delete. When there are no experts remaining in the dialog, click OK and the node will be cleared of the multiple expert's data.

## 20. Estimating Probabilities from Data (Enterprise only)

The probabilities for chance nodes in a DPL model can come from a variety of sources. When the uncertainty is a one-of-a-kind, truly unique event, and no structured data is available, probabilities are often obtained by direct subjective assessment by an expert in the subject matter.

When structured data is available, DPL can infer the probabilities and probabilistic relationships (i.e., Influence Arcs) for a data set consisting of a number of outcome observations (rows) for two or more uncertain events (columns).

### 20.1 Tutorial: Moving from Analyzing Datasets to Decisions

---

You are going to be working with a dataset derived from a direct marketing campaign conducted by a banking institution. Within the campaign, existing bank customers were contacted and offered a subscription to a bank term deposit. The example dataset is a .CSV file. (Moro, S., Cortez, P. and Rita, P. A Data-Driven Approach to Predict the Success of Bank Telemarketing. *Decision Support Systems*, Elsevier. 62:22-31, June 2014.)

#### 20.1.1 Estimating Probabilities for an External Dataset

- ⇒ Open the Bank Marketing.csv file in Excel. This file is found in the Examples folder where DPL was installed, usually C:\Program Files\Syncopation\DPL9\Examples.

	A	B	C	D	E	F	G	H	I
1	Age	Job	Marital St	Education	Default	Housing	Personal	Previous	Success
2	50-59	managem	married	tertiary	no	yes	no	unknown	no
3	40-49	techniciar	single	secondary	no	yes	no	unknown	no
4	30-39	entrepreur	married	secondary	no	yes	yes	unknown	no
5	40-49	blue-colla	married	unknown	no	yes	no	unknown	no
6	30-39	unknown	single	unknown	no	no	no	unknown	no
7	30-39	managem	married	tertiary	no	yes	no	unknown	no
8	18-29	managem	single	tertiary	no	yes	yes	unknown	no
9	40-49	entrepreur	divorced	tertiary	yes	yes	no	unknown	no
10	50-59	retired	married	primary	no	yes	no	unknown	no
11	40-49	techniciar	single	secondary	no	yes	no	unknown	no
12	40-49	admin.	divorced	secondary	no	yes	no	unknown	no
13	18-29	admin.	single	secondary	no	yes	no	unknown	no
14	50-59	techniciar	married	secondary	no	yes	no	unknown	no
15	50-59	techniciar	married	unknown	no	yes	no	unknown	no
16	50-59	services	married	secondary	no	yes	no	unknown	no
17	50-59	retired	married	primary	no	yes	no	unknown	no
18	40-49	admin.	single	unknown	no	yes	no	unknown	no
19	50-59	blue-colla	married	primary	no	yes	no	unknown	no
20	60-99	retired	married	primary	no	yes	no	unknown	no

**Figure 20-1. Bank Marketing.csv Example File in Excel**

The Bank Marketing.csv file contains 9 columns of data (Figure 20-1). The first 8 columns (columns A-H) contains input data which takes the form of personal bank customer information (age, job, marital status, and education), loan status (existing personal or housing loans), and the outcome of a previous marketing campaign. The 9<sup>th</sup> column (I) contains data for the target event: whether or not the bank customer subscribed to a bank term deposit (yes or no). If you navigate to the end of the dataset (Ctrl+End), you'll find that it includes over 45,000 rows or observations.

It costs the bank money (e.g., sales staff time) to initiate contact, market, and sell products to existing customers. The goal of bringing together this set of data and DPL is to identify the set of customer inputs that are the best predictors of campaign success, which in this case is the customer

subscribing to a term deposit, and thereby make better decisions about the marketing campaign.

In addition to building a better understanding of the dataset, DPL's Decision Tree analysis capabilities provide you the ability to explicitly add decisions (Sell Subscription?) and other key variables to the model (Costs, Profit) so you can develop decision rules based on the relationships you've discovered in the data.

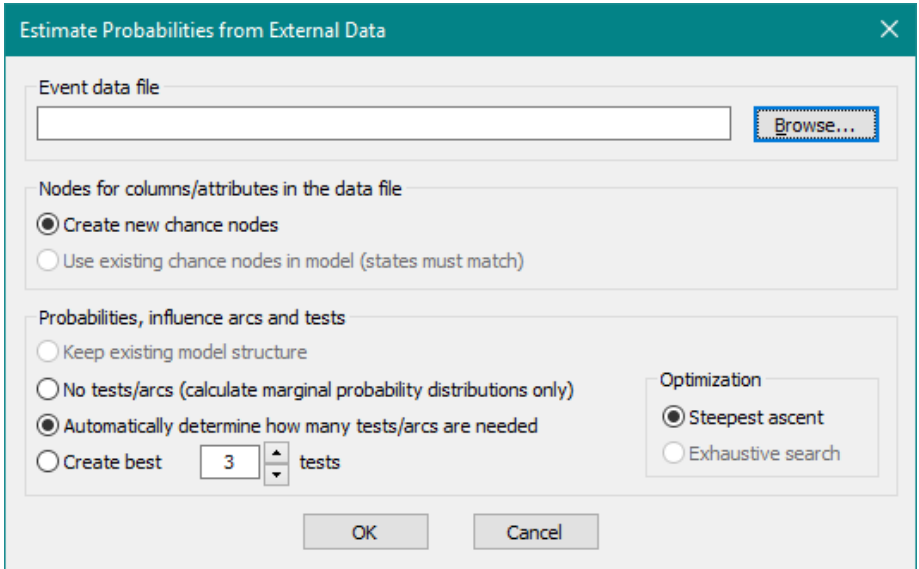
Armed with these tools, the bank will be able to direct resources to the customers most likely to buy, enhancing customer relationships by offering the right products to the right people and boosting the efficiency and overall success of the campaign.

Starting from a blank workspace, the first thing you'll do is modify a couple of model-level settings. More specifically, you're going to change some number formatting.

- ⇒ Close the Bank Marketing.csv. DPL won't be able to access the dataset if the file is open in Excel.
- ⇒ Open a blank DPL Workspace.
- ⇒ Click File | Options.
- ⇒ Switch to the Outputs tab and under *General number formatting* change the number of decimal places to 2.
- ⇒ Click OK to close the File Options dialog.
- ⇒ If the Influence Diagram pane is not maximized, hit Tab to maximize it and change to Influence Diagram-focused modeling mode.

Note that you can make Influence Diagram-focused modeling mode the default for new models on File | Options | General by selecting the Influence Diagram radio button for the *For new models, maximize pane* setting.

- ⇒ Choose Data | Learning | Estimate Probs. DPL will open the Estimate Probabilities from External Data dialog (Figure 20-2).



**Figure 20-2. Estimate Probabilities from External Data dialog**

The Estimate Probabilities dialog allows you to choose the external dataset you'd like to estimate probabilities from and provides options for creating the resulting nodes, probabilities, influence arcs and conducting tests.

- ⇒ Click the Browse... button and navigate to the Examples folder (C:\Program Files\Syncopation\DPL9\Examples). Select the Bank Marketing.csv example file and click Open.

Beneath the path and filename DPL displays a note on how many events are contained within the data file.

Within the *Nodes for columns/attributes in the data file* section, you can choose whether or not you'd like DPL to create new chance nodes from the dataset. Since you're starting from a blank workspace your only option is to have DPL create the chance nodes for you from the dataset. It is possible to use an existing model – as long as the chance nodes and states match those in the dataset.

Within the *Probabilities, influence arcs, and tests* section, the first option listed, *Keep existing model structure*, is greyed out because you are starting from a blank model. If you had already created a model from the dataset, you could choose this setting to maintain the existing model structure (i.e., arcs and tests) and update the probabilities.

The next option, *No tests/arcs*, tells DPL not to generate any tests or arcs for the dataset. With this selected, DPL will simply calculate the marginal

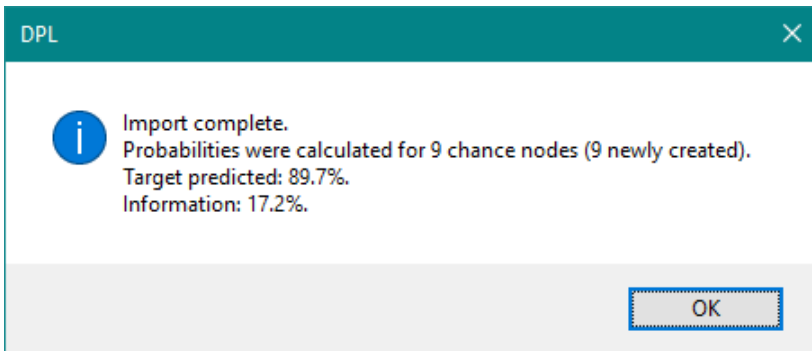
probability distributions for each event (column) in the dataset. The probabilities can then be viewed within each node's definition.

This third option is selected by default and tells DPL to *Automatically determine how many tests/arcs are needed* to estimate probabilities for the dataset. For the purposes of this tutorial, you'll choose this setting.

Lastly, instead of having DPL automatically determine how many tests are appropriate, you can explicitly set the number of tests you'd like by setting the radio button to the *Create best [n] tests* and setting the spin box to the desired number of tests.

Note that the *Steepest ascent* setting is selected within the *Optimization* box and is currently the only option available. The *Exhaustive search* setting is only available if you explicitly set the number of tests (e.g., *Create best [n] tests*) to 3 or more. With the Steepest ascent setting, DPL will choose the arcs one at a time based on the inputs that provide the best test. With the Exhaustive search, DPL will look at all of the possible combination of inputs in order to generate the [n] arcs that provide the best tests.

⇒ Leave the default settings as they are and click OK to import the dataset and estimate probabilities.



**Figure 20-3. Import Complete dialog for Estimating Probabilities**

Once the process is complete DPL will bring up the Import complete dialog (Figure 20-3). The dialog tells you for how many chance nodes probabilities were calculated and of those nodes how many were newly created. In the current model, probabilities were calculated for 9 chance nodes, all of which were newly created.

Beneath this, the dialog lists percentages for Target predicted (89.7%) and Information (17.2%). The Information percentage is a quantification of the extent to which the tests conducted resolve uncertainty about the target

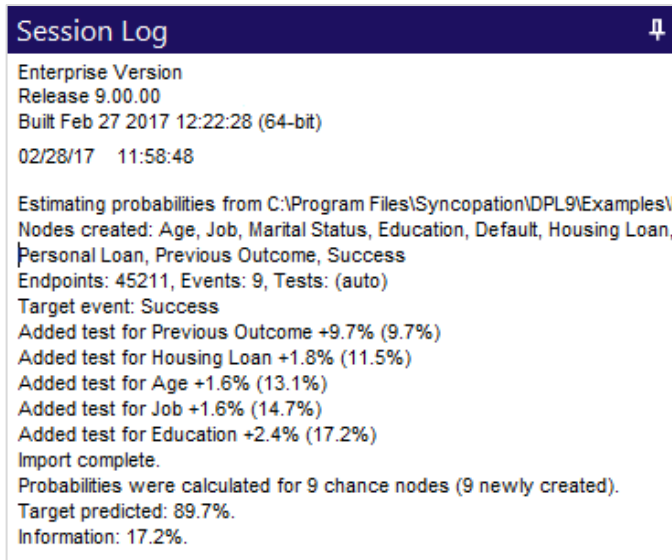
event. If knowing the outcomes of the tests would always tell you the outcome of the target, the Information would be 100%.

⇒ Click OK to close the Import complete dialog.

More detailed information regarding the import is contained within the Session Log (Figure 20-4). You can see that DPL created 5 tests for the dataset, one for each of the following variables:

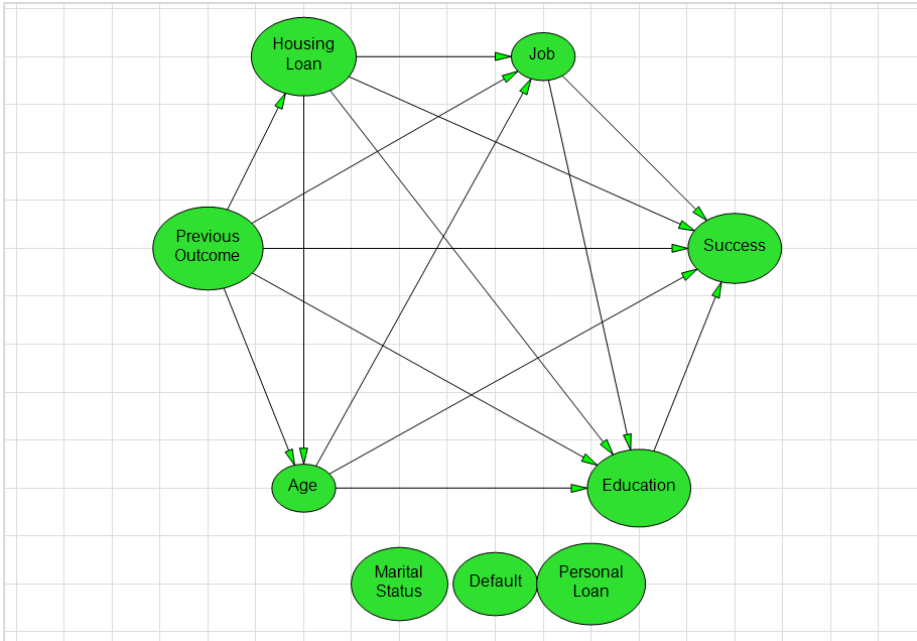
- Previous Outcome
- Housing Loan
- Age
- Job
- Education

The incremental information gained by the addition of each test is listed, and sums to 17.2%.



**Figure 20-4. Session Log with Information on Probabilities**

Looking at the Influence Diagram you can see that DPL has added 9 chance nodes to the model. Six of the nodes (the 5 tests and target event) have a number of influence arcs between them. These are the variables that provide you with information about the target event. The three nodes at the bottom with no arcs provide less information about the target.



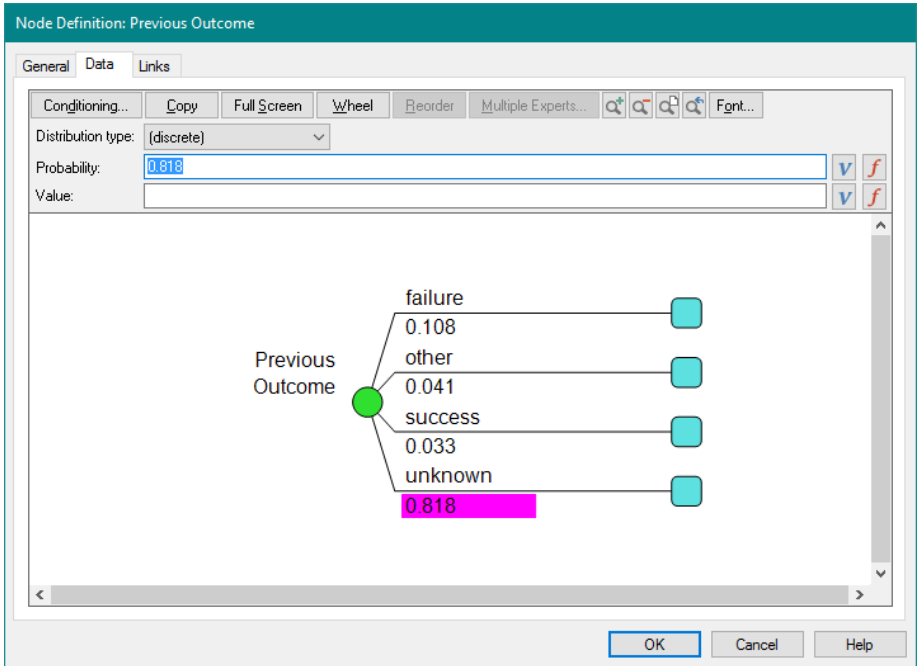
**Figure 20-5. Influence Diagram generated from Bank Marketing Dataset**

According to the Session Log the first "test" added was for the Previous Outcome variable. Previous Outcome refers to the outcome of the previous marketing campaign for the particular customer, with the possible states being failure, other, success, and unknown.

⇒ Double-click the Previous Outcome node to view its probability input tree (Figure 20-6).

It should be noted that more than 80% of customers are in the "Unknown" state (Figure 20-6). These customers neither bought nor refused the previous offer, either because they weren't contacted or didn't reply. In this tutorial you will be focusing your analytic efforts on this particular subset of customers.

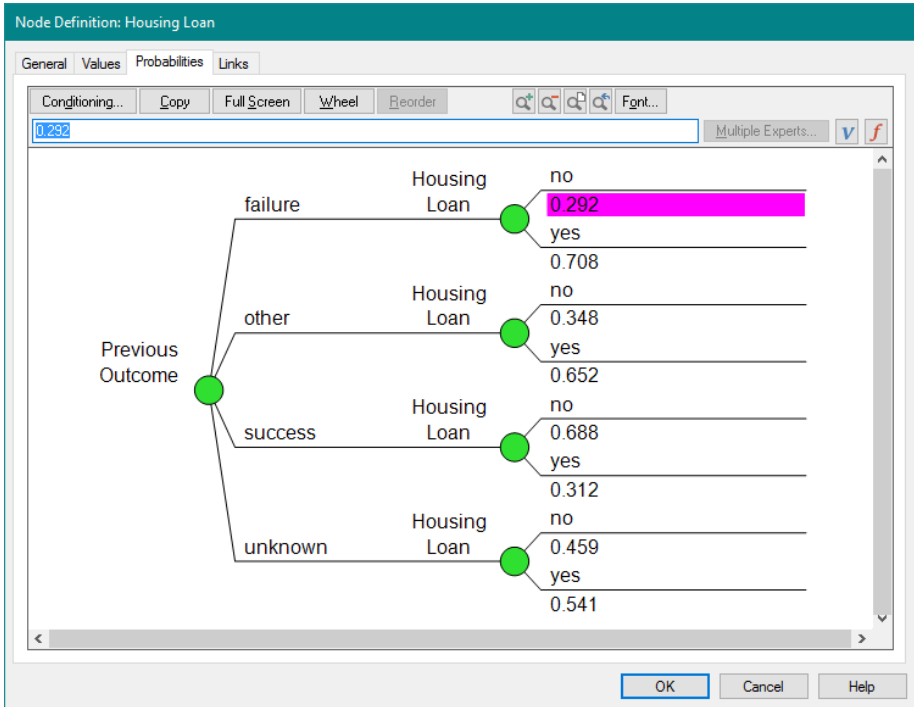




**Figure 20-6. Probabilities for Previous Outcome Node**

The probabilities contained within the node data for each chance node come from fractions of records within the dataset, with each record weighted equally.

- ⇒ Double-click Housing Loan to view its probability input tree (Figure 20-7).



**Figure 20-7. Conditioned Probabilities for Housing Loan Node**

The second test added to the model was for the Housing Loan variable. You can see from its probability input tree that Housing Loan is conditioned by the Previous Outcome variable (the first test). For example, when Previous Outcome is "unknown" then there is close to a 50/50 split between those with and without a housing loan (~46% vs. ~54%).

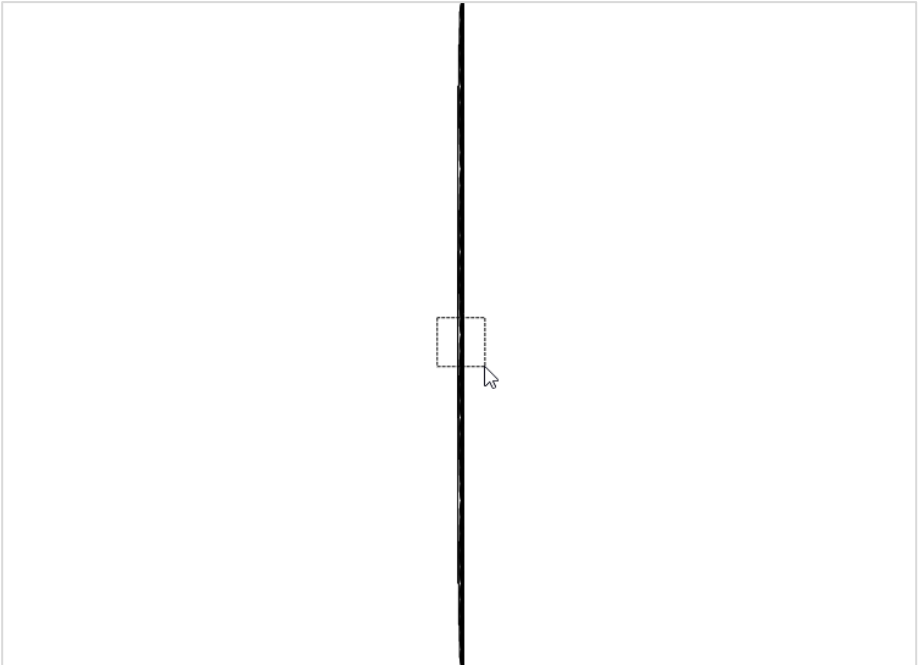
With each additional test added to the model, more conditioning and general complexity is introduced. Consequently, the probability input trees for the third through fifth tests and target event are quite complex and are therefore difficult to display but they need not be edited directly.

You will now run a preliminary analysis on the model in order to generate a Policy Tree output. This output will tell you approximately what percent of customers contacted will subscribe to the term deposit. Note that the model has over 3,840 endpoints, all of which are displayed initially.

- ⇒ Within the Home | Run group un-check Risk Profile, make sure Policy Tree is checked.
- ⇒ Click Home | Run | Decision Analysis to run the model. The fully expanded Policy Tree will be displayed.

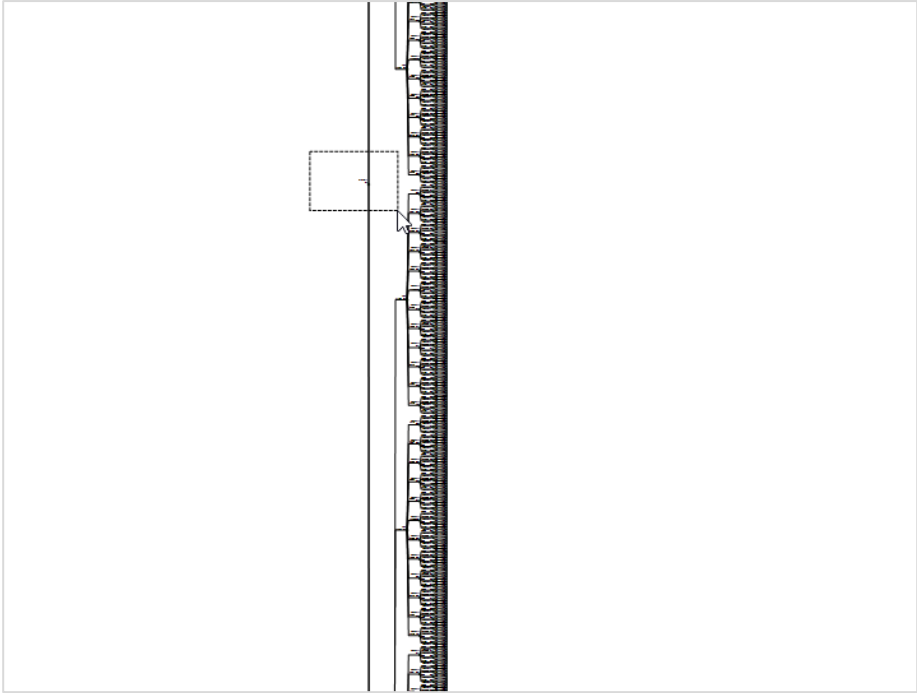
Obviously you can't see much with the tree fully expanded at this zoom level, so you will zoom in to a section by selecting the area near the head of the tree, collapse the tree, and then expand it to just a few levels for better viewing.

- ⇒ Do a Shift+Ctrl+Drag (zoom by selection) on a small section of the middle of the Policy Tree as shown in Figure 20-8.



**Figure 20-8. Zoom by Selection on Fully Expanded Policy Tree**

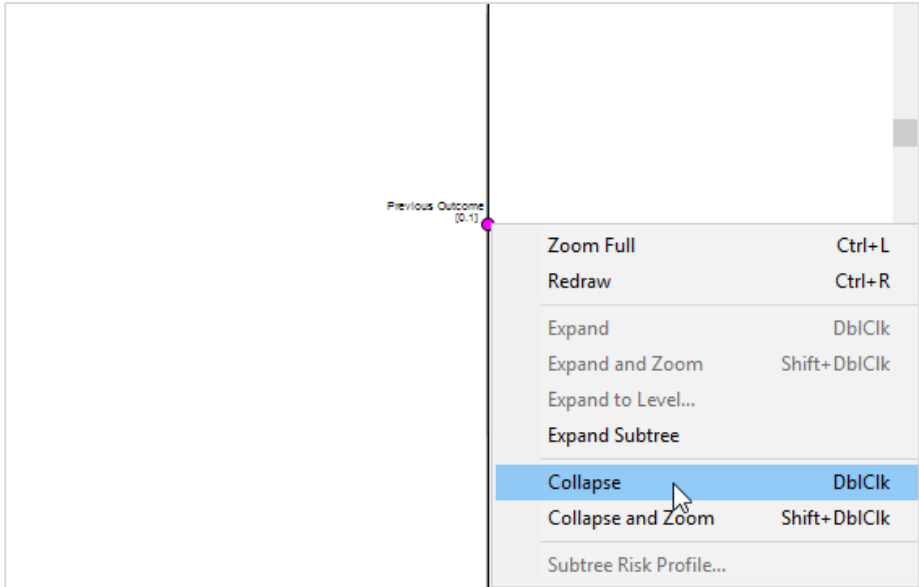
- ⇒ You'll likely see the node at the head of the tree, do one more Shift+Ctrl+Drag (zoom by selection) on a small section that includes the head of the Policy Tree as shown in Figure 20-9.



**Figure 20-9. Second Zoom by Selection on Fully Expanded Policy Tree**

Now you should see the node (Previous Outcome) at the head of the Policy Tree (Figure 20-10). Now you'll collapse the tree at this node.

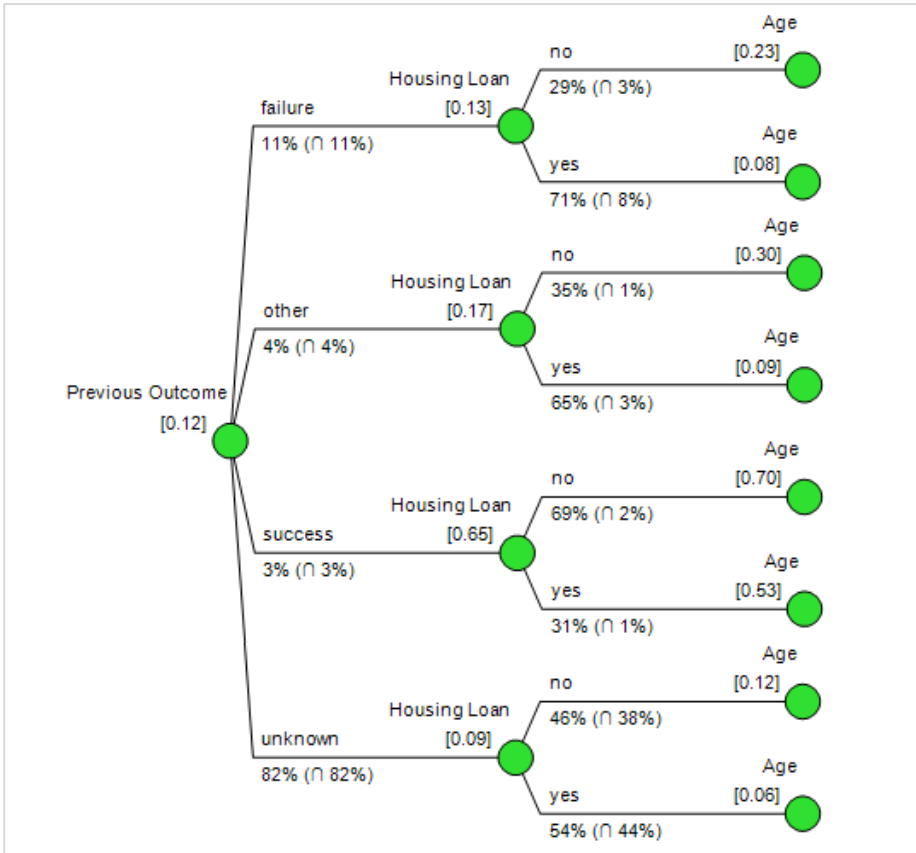
- ⇒ Right-click on the Previous Outcome node and select Collapse from the context menu.



**Figure 20-10. Collapsing Policy Tree at the Previous Outcome Node**

- ⇒ Right-click the Previous Outcome node again and select Expand to level... from the context menu.
- ⇒ Within the Expand Policy Tree dialog enter a "3" and click OK.

Now you can easily view probabilities and Expected Values for the first 3 levels of the Policy Tree (Figure 20-11). Note that the probabilities near the head of the tree are those you viewed earlier within the node data. The overall expected value (i.e., probability of target event being yes) is approximately 12%. However, this includes all of the states of the Previous Outcome node. Recall earlier that you are really only interested in those customers within the "unknown" category. So you'll activate the Decision Tree and use branch control to exclude the other states of this node.



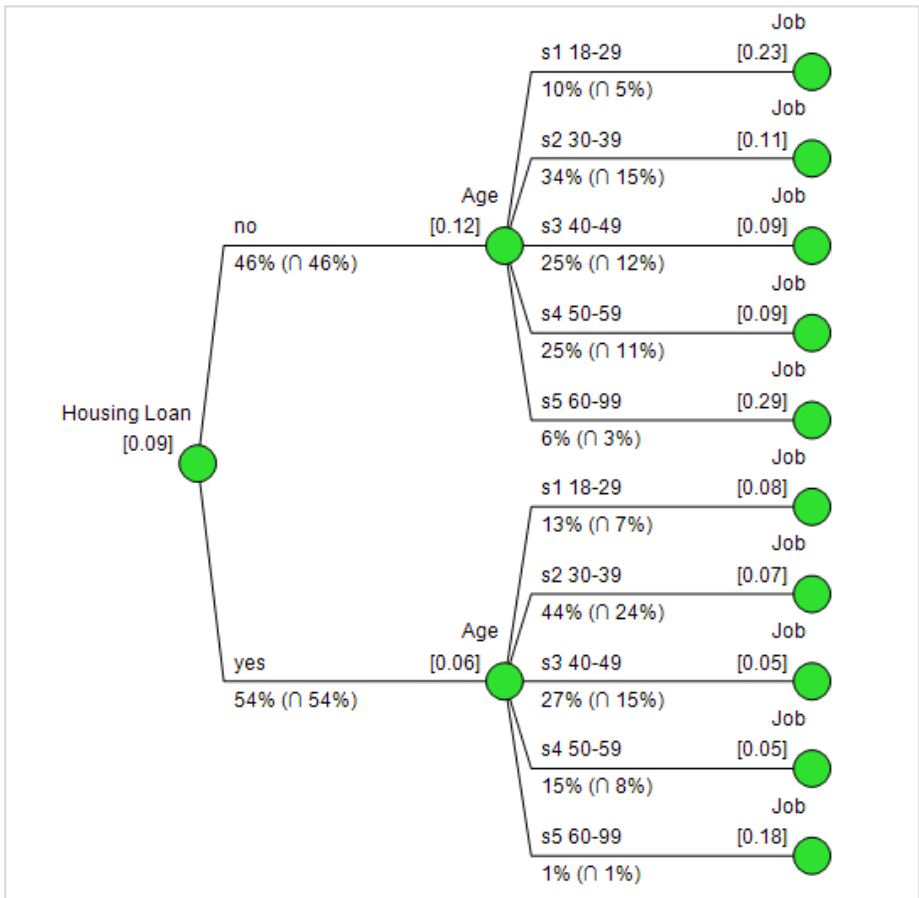
**Figure 20-11. Policy Tree expanded 3 Levels**

- ⇒ Activate the model and press the Tab key to switch to the Decision Tree pane.
- ⇒ With the branches of the Previous Outcome node selected, drop-down the Branch Control combo box within the Decision Tree | Control/Block group and select "unknown" from the list.

You'll run another analysis on the model that includes just the customers that fall within the "unknown" state.

- ⇒ Click Home | Run | Decision Analysis.
- ⇒ As done previously, zoom by selection until you can see the node at the head of the Policy Tree, which is Housing Loan now that you've applied branch control to the Previous Outcome node.

⇒ Collapse the tree at this node, and then expand to level 3 as before (Figure 20-12).



**Figure 20-12. Policy Tree expanded 3 Levels with Previous Outcome Controlled to "unknown" State**

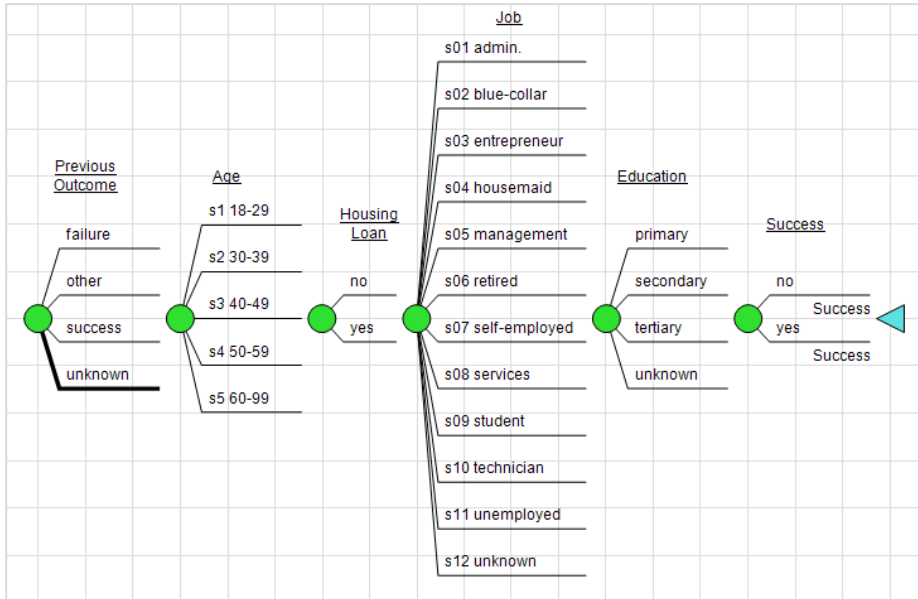
The expected value of the model is now 0.09, meaning that approximately 9% of customer within the "unknown" category that are contacted will subscribe to the term deposit.

### 20.1.2 Moving from Data Analysis to Decision Making

Now that the data is represented in a model, you can quickly and easily manipulate it to focus on the information at hand and the decisions you need to make. For example, you may want to see the Age variable before

the Housing Loan because that's what makes most business sense (as a bank, you likely know all of your customer's demographic information, so it's easy to check their age). This can be done through a simple re-order of the Decision Tree.

- ⇒ Activate the Decision Tree.
- ⇒ Select the Age node, drag it and drop it on top of the Housing Loan node. Your tree should now match Figure 20-13.



**Figure 20-13. Decision Tree after Age Node Re-ordered**

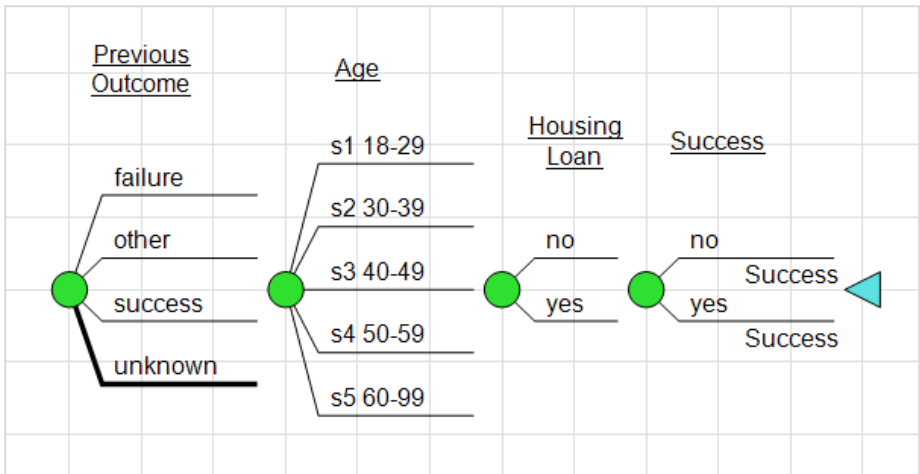
Recall that the Age node had influence arcs coming into it and going out of it within the Influence Diagram (i.e., it is both a successor to other nodes and a predecessor to other nodes). More specifically, the Housing Loan node probabilistically conditions the Age node (i.e., a probability conditioning arc (green head) went from Housing Loan to Age). In the Decision Tree now, these nodes are in the reverse order of the conditioning arc. Through a technique called Bayesian Revision, DPL automatically "flips" the arcs in order to calculate the probabilities correctly within the rolled-back tree.

You'll make some further modification to the Decision Tree in order to make it more concise, which will result in a Policy Tree that is more compact and easier to read. More specifically, you will remove the Job and Education nodes from the tree since they've been found to be less



consequential. DPL will still be able to do the necessary calculations with these variables removed.

- ⇒ Hold down Ctrl, select the Job node and the Education node so both are selected.
- ⇒ Press the Delete key to remove them from the tree. This will cause the subtree after it to detach from the rest of the tree.
- ⇒ Re-attach the Success node to the tree by dragging and dropping the node onto the endpoint for the Housing Loan node.

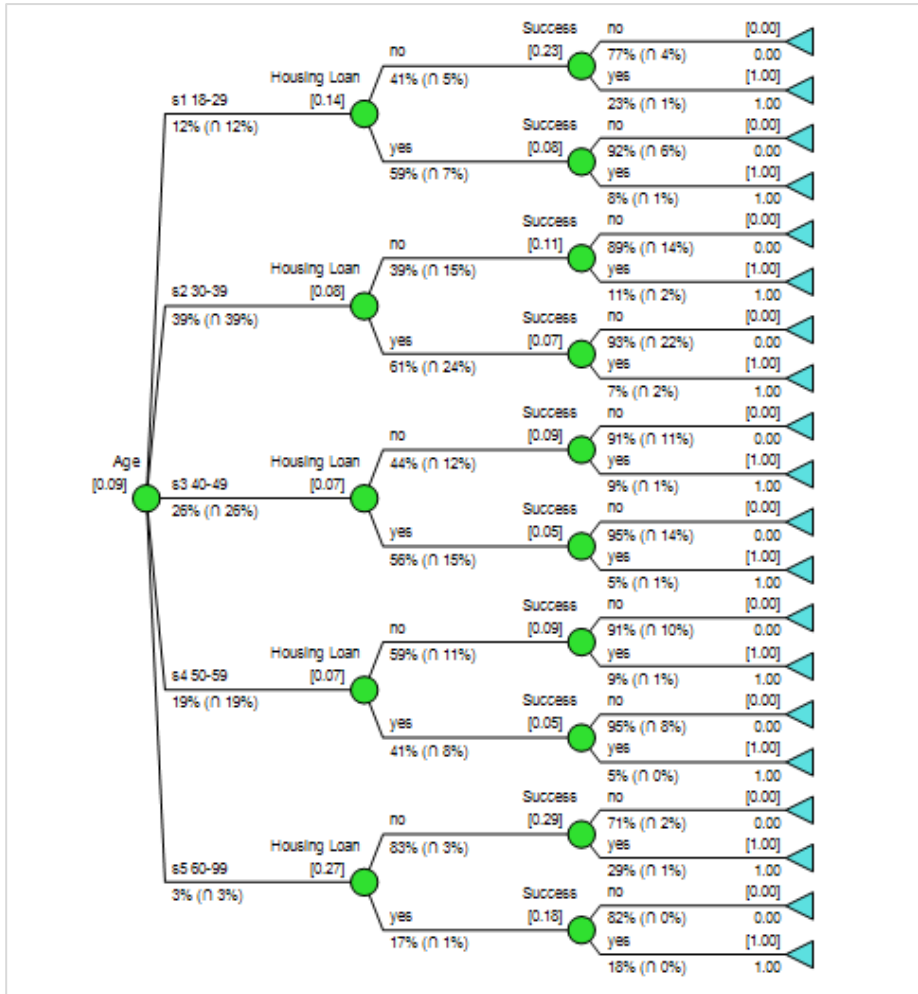


**Figure 20-14. Decision Tree with Education and Job Node Removed**

No modifications need to be made within the Influence Diagram. You'll run the model again and look at the Policy Tree output.

- ⇒ Click Home | Run | Decision Analysis.

With the Decision Tree trimmed to just the essential variables, the Policy Tree is now readable without having to zoom and collapse (Figure 20-15).



**Figure 20-15. Fully Expanded Policy Tree for Trimmed Decision Tree**

Next, you will add a few nodes to the model to incorporate a downstream decision: whether or not to sell the product to the customer.

- ⇒ Activate the Influence Diagram.
- ⇒ Add a new value node (Influence Diagram | Node | Add).
- ⇒ Name the node "Marketing Costs" and enter a value of "-20" within the Data tab.
- ⇒ Create another new value node, name it "Profit", and enter a value of "150" on the Data tab.

This means that it will cost the bank \$20 for an employee to attempt to sell a product to a given customer. If a given bank customer subscribes to the term deposit, the bank will see a \$150 profit. Now you'll add the decision to the model.

- ⇒ Add a new Decision node to the model and name it "Sell?". Leave the default alternatives of Yes and No. Click OK to close the Node Definition dialog.
- ⇒ Press the Tab key to switch to the Decision Tree pane.
- ⇒ Select Decision Tree | Instance | Add | Decision.
- ⇒ Within the Select Decision dialog select Sell? and click OK.
- ⇒ Place the node on top of the Success node.

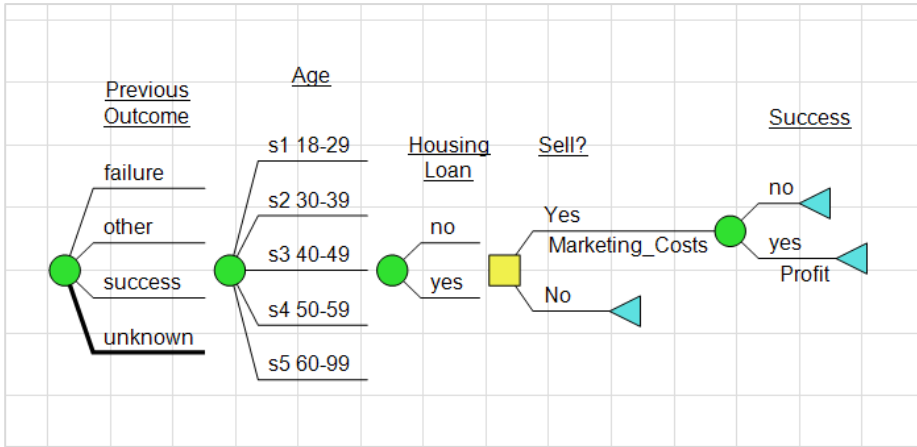
Notice that the decision comes after both the Age and Housing Loan nodes in the tree. This is because these are two variables that are observed (on the CSR's screen) for each customer contacted prior to deciding whether or not you should attempt to sell them the product.

- ⇒ With the Sell? decision selected, check the Asymmetric box within Decision Tree | Instance.
- ⇒ Re-connect the Success node to the endpoint for the Yes alternative of the Sell? decision.
- ⇒ Select the branches of the Success chance node and press the Delete key to remove the get/pay expression.
- ⇒ With the branches for Success still selected, check the Asymmetric box within Decision Tree | Instance.

Now you'll need to modify the get/pay expressions in the tree to indicate at what point the Marketing Costs and Profit values are paid and received.

- ⇒ Select the Yes branch of the Sell? decision, drop-down the Edit Get/Pay box within Decision Tree | Get/Pay group, and select Marketing\_Costs.
- ⇒ Select the yes branch of the Success chance node, drop-down the Edit Get/Pay box within Decision Tree | Get/Pay group, and select Profit.

Your Decision Tree should now match what is shown in Figure 20-16.



**Figure 20-16. Decision Tree with Sell? Decision Added**

This Decision Tree can be thought of as a representation of a single customer/CSR interaction within the campaign. While the "Sell?" decision is a downstream decision because it's preceded by several uncertainties, those uncertainties represent characteristics of customer accounts that can be readily observed rather than events that will play out in the future. The goal of the decision analysis is to reveal a decision policy (that is, a set of decision rules) that can be applied to all customers contacted. You'll run one last Decision Analysis to see what the optimal decision policy is.

⇒ Click Home | Run | Decision Analysis. The Policy Tree is displayed (Figure 20-17).

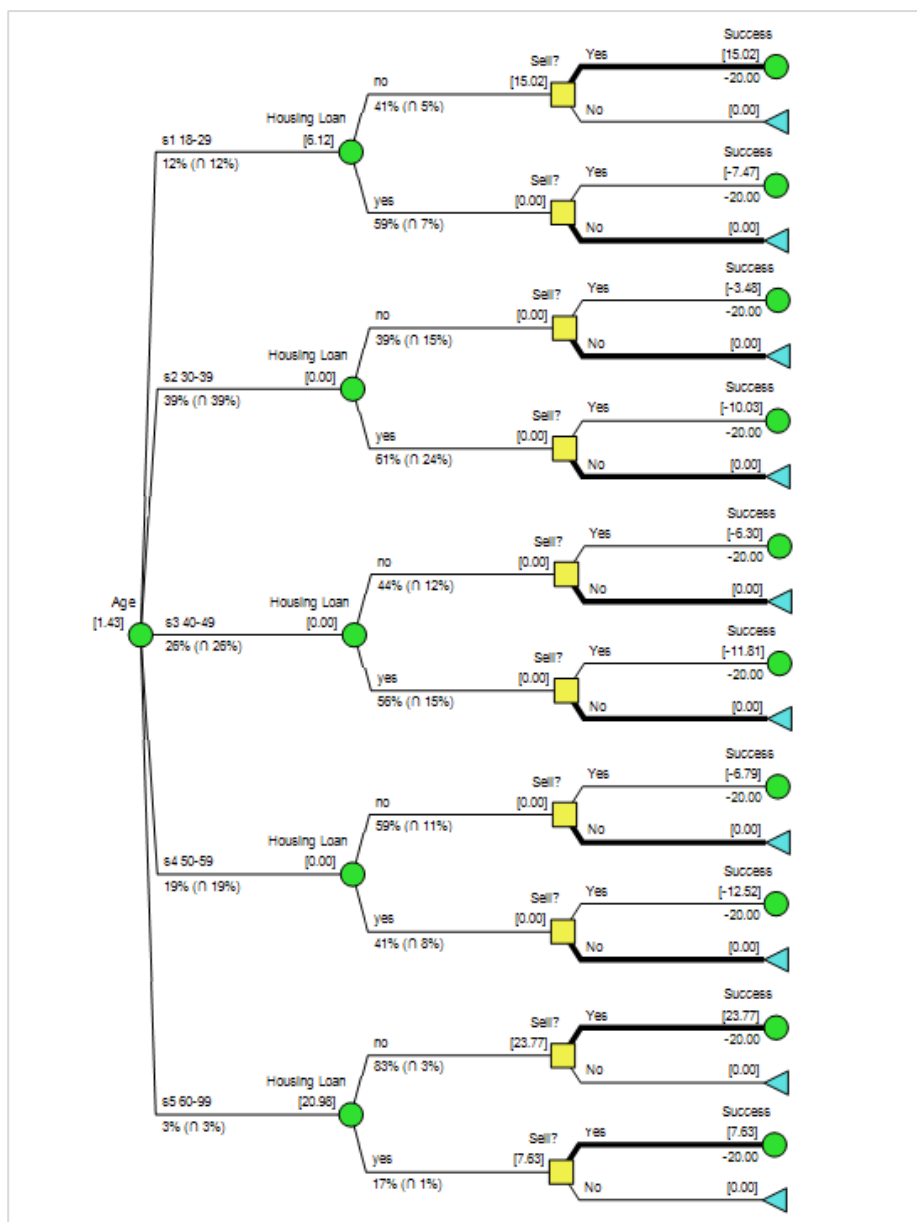


Figure 20-17. Policy Tree™ with Sell? Decision Added

As you can see from the Policy Tree output there are a couple of scenarios in which your customer service representatives should attempt to sell the term deposit. More specifically, young customers (ages 18-29) who don't have a housing loan and older customers (ages 60-99) regardless of housing loan status are most likely to subscribe to the product. With this information, the bank can implement this decision policy by flagging the subset of customers that fit these particular criteria and reaching out to them to offer the term deposit product.

# 21. Database Linking in DPL™ (Enterprise only)

## 21.1 Overview

---

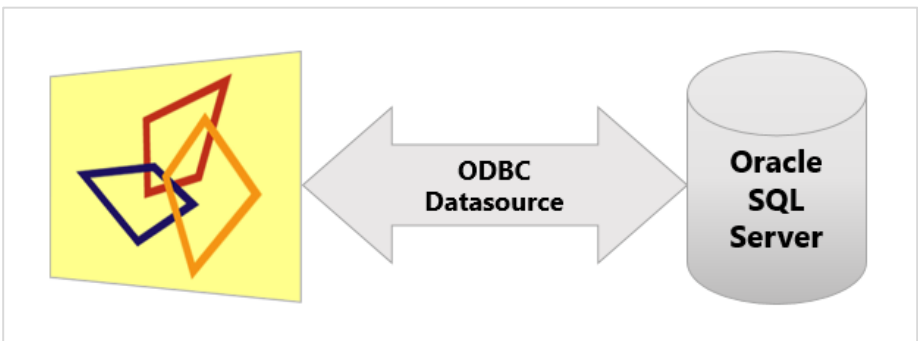
With DPL Enterprise you can use data stored in a database to initialize nodes in much the same way you can store data in Excel and use Excel initialization links. In situations where some of the data you need for your DPL model is already stored in a database and is subject to revision, using database initialization links will ensure you have the most recent data and will reduce errors due to data re-entry. In situations where multiple people need access to the data and may be revising it, you can configure the database to keep track of revisions.

This chapter discusses how to use database linking in DPL Enterprise.

## 21.2 ODBC Data Sources

---

DPL communicates with a database via the Windows' ODBC (Open Database Connectivity) mechanism. See Figure 21-1.



**Figure 21-1. DPL/Database Communication uses ODBC**

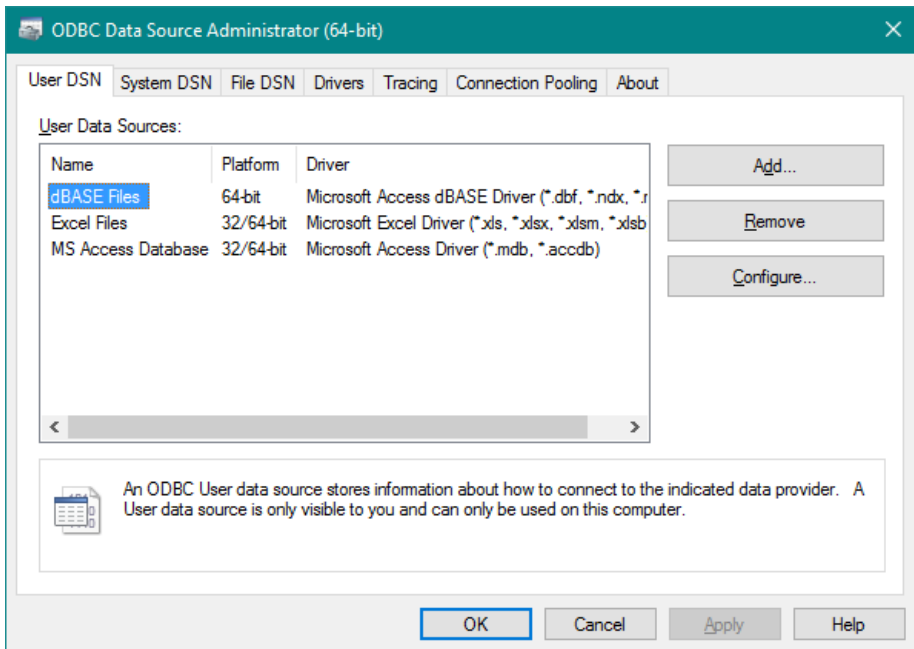
By using ODBC, DPL gives you the flexibility to choose your database management system or even change it as requirements evolve.

### 21.2.1 Setting up an ODBC Data Source for a Desktop Database

Before using database links in DPL, you must set up an ODBC data source for the database with which you wish to communicate. You will do this now for the example database delivered with DPL Enterprise. Depending on the version of Windows you are running, the following steps for setting up the data source may vary.

The following instructions assume you are running 64-bit versions of Windows, DPL, and Microsoft Office. Most recent computers have 64-bit Windows. If you have a 32-bit version of Microsoft Office installed, contact support for guidance. Note: DPL 9 is a 64-bit executable, but prior versions of DPL were 32-bit, so if you are upgrading and have a database linked model you will need to create a new, 64-bit data source.

- ⇒ Go to Administrative Tools on your machine.
- ⇒ Open ODBC Data Sources (64-bit). The ODBC Data Source Administrator dialog appears. See Figure 21-2.



**Figure 21-2. ODBC Data Source Administrator Dialog**

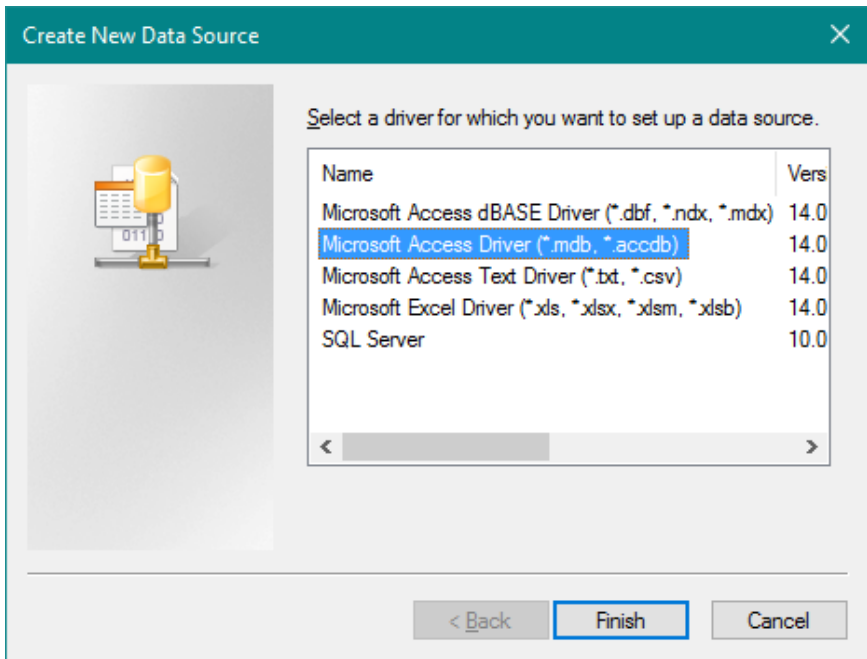
Among other things, the ODBC Data Source Administrator dialog displays all the data sources currently configured on your computer. On the User DSN tab, data sources that are available only to the logged in user are



displayed. On the System DSN tab, data sources available to all users of the machine plus services are displayed.

The ODBC Data Source Administrator dialog allows you to add new data sources and delete or configure existing ones. You will now add a data source.

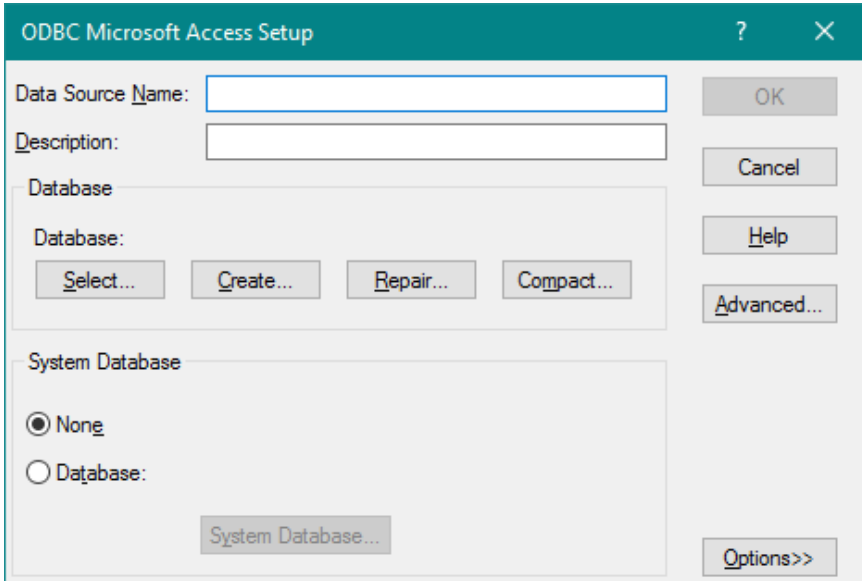
- ⇒ Select the User DSN tab if it isn't already.
- ⇒ Click the Add button. The Create New Data Source dialog appears. See Figure 21-3.



**Figure 21-3. Create New Data Source Dialog**

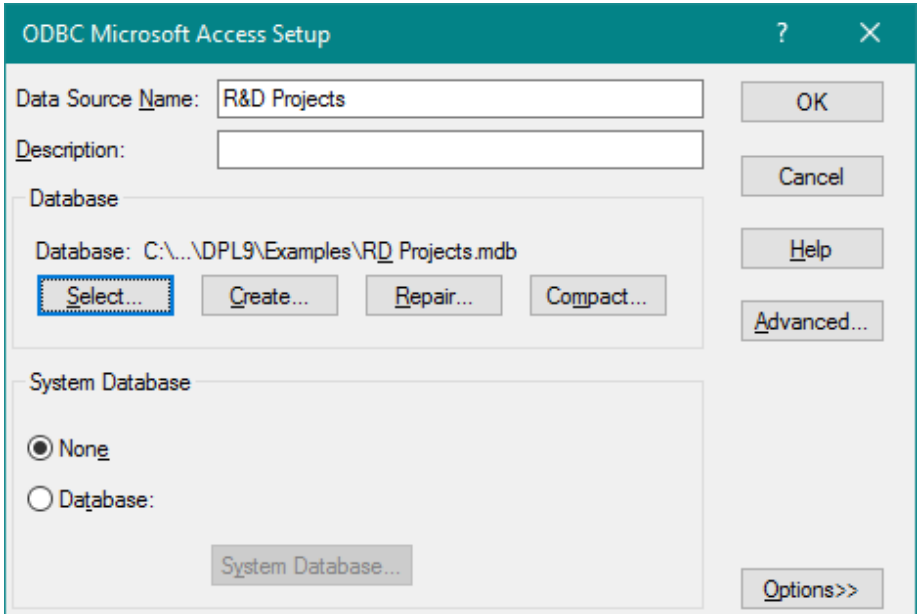
The sample database delivered with DPL Enterprise is a Microsoft Access database.

- ⇒ Select Microsoft Access Driver (\*.mdb) from the list.
- ⇒ Click Finish. The ODBC Microsoft Access Setup dialog appears. See Figure 21-4.



**Figure 21-4. ODBC Microsoft Access Setup Dialog**

- ⇒ For the Data Source Name, enter "R&D Projects".
- ⇒ You may optionally give the data source a description.
- ⇒ In the Database section, click the Select... button.
- ⇒ Use the Select Database dialog to browse to the Examples folder below the DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples.
- ⇒ Select R&D Projects.mdb for the Database Name.
- ⇒ Click OK. The ODBC Microsoft Access Setup dialog should now look like Figure 21-5.



**Figure 21-5. Completed ODBC Microsoft Access Setup Dialog**

- ⇒ Click OK to close the ODBC Microsoft Access Setup dialog. The new data source should appear in the User Data Sources list.
- ⇒ Click OK to close the ODBC Data Source Administrator.

Note: the information required to set up an ODBC Data Source varies depending on the database to which the data source refers. For Access, the only thing you had to specify was a file name and location. Access is a desktop database. To configure a data source for a server-based database such as Oracle or SQL Server, you will likely need to get information from your IT department or database administrator.

## 21.3 Database Tables in DPL™

The data needed for a DPL model will likely not all be scalar data. E.g., the probabilities for a chance node might be in one- or two-dimensional array. Tables that store non-scalar data to be used in DPL models must be structured in a particular way. If the database is being developed primarily to store data that DPL will use, then you can configure the tables in the database directly so that they meet the requirements of DPL. If the database is pre-existing and/or will store data for purposes other than for

use with DPL, the tables themselves do not need to be structured in this particular way; rather queries and/or views can be written to provide the necessary structure for DPL. Put another way, the underlying tables can be structured in whatever way it is deemed appropriate as long as they contain the necessary information so that a query or view of the table can be developed to provide the structure that DPL requires for non-scalar data.

For simplicity, the term tables is used in the discussion below when talking about the structure that DPL requires for non-scalar data, but remember that it may be a query or view of the table that provides the needed structure.

### 21.3.1 Required Fields

Table 21-1 summarizes the fields that are required in a table that stores non-scalar data for a DPL model.

Field	Purpose	Type	Default Name
Model ID	Identifies which model the record belongs to (optional).	Integer	ModelID
Project ID	Identifies which project the record belongs to. Required in any table that stores project specific information.	Integer	ProjectID
Node ID	Identifies the specific data item to DPL. It is used in the Data tab of the Node Definition dialog.	String	NodeID
Dimensions	Tells DPL whether the data item stored in the record is scalar (0), a one-dimensional array (1) or a two-dimensional array (2).	Integer	Dims
Rows	Tells DPL the number of rows for the data item.	Integer	Rows
Columns	Tells DPL the number of columns for the data item.	Integer	Cols

**Table 21-1. Required Fields for a Node ID-Structured Table**

A combination of the Model ID and/or Project ID fields is used to identify which model and/or project the data belongs to. Depending on the overall design of the decision and data management system you are developing,

you may want to use both. Any table that stores project-specific information must have a Project ID field in it to identify the data with the project. If you are storing data in a database for multiple different DPL models, you may want to also identify which model the data in each record belongs to by using the Model ID field. If you have a situation where some data is common across all projects for a particular model (i.e., not project-specific), you might have a table that has a Model ID field but not a Project ID field.

A Node ID structured table needs to meet one more requirement. The fields in the table that will store the data for each item must all begin with the same prefix and must be numbered sequentially from 001 on. The default numeric data fields prefix is "Data\_". A record that stores numeric information should store the data for the data item in Data\_001, Data\_002, etc. You may change the data fields prefix but the fields storing the data must end with 001, 002, etc. For example, if your data fields prefix is "fld", then the fields storing the data must be fld001, fld002, etc.

DPL can also access string data stored in a database. If a table stores string data, then you must specify the String fields prefix. The default prefix is "Str\_". A record that stores string information should store the data for the data item in Str\_001, Str\_002, etc.

Node IDs in the table must adhere to DPL's variable naming requirements. I.e., they cannot contain any punctuation or spaces. They may contain letters or numbers but they must start with a letter. Node IDs are case sensitive, i.e., "cost" is different from "Cost".

The order of the fields in each table does not have to match the order shown in Table 2-1 or Figure 21-6. However, the order in which the data is stored within the data fields is important. If a record contains a Node ID for a one-dimensional array with 3 elements, then the data for the first element needs to be stored in the first data field (e.g., Data\_001), the data for the second element needs to be stored in the second data field (e.g., Data\_002), etc. Similarly if a record contains a Node ID that is storing probabilities for a chance node, then the probability for the first branch needs to be stored in the first data field, and so forth.

### 21.3.2 Required Fields for Revision Tracking

If you wish to set up a database that tracks revisions of data, then you need to have two additional fields in tables storing data. Table 21-2 summarizes these fields.

Field	Purpose	Type	Default Name
Revision ID	Identifies which revision this record is.	Integer	RevisionID
Revision Date	The date the revision was made.	Date	RevisionDate

**Table 21-2. Required Revision Tracking Fields**

The Revision ID field must be an integer greater than or equal to zero, where a higher number indicates a more recent revision. For each table in the database storing project-specific data, the combination of Project ID, Node ID, Revision ID must be a unique key, i.e., there cannot be two records with the same values for Project ID, Node ID, and Revision ID. If the table has a Model ID, then Model ID, Node ID, Revision ID must be a unique key. If a table has both a Model ID and a Project ID, then all four must be a unique key.

Figure 21-6 shows the Access design view for a Node ID structured table called Project\_Tbl.

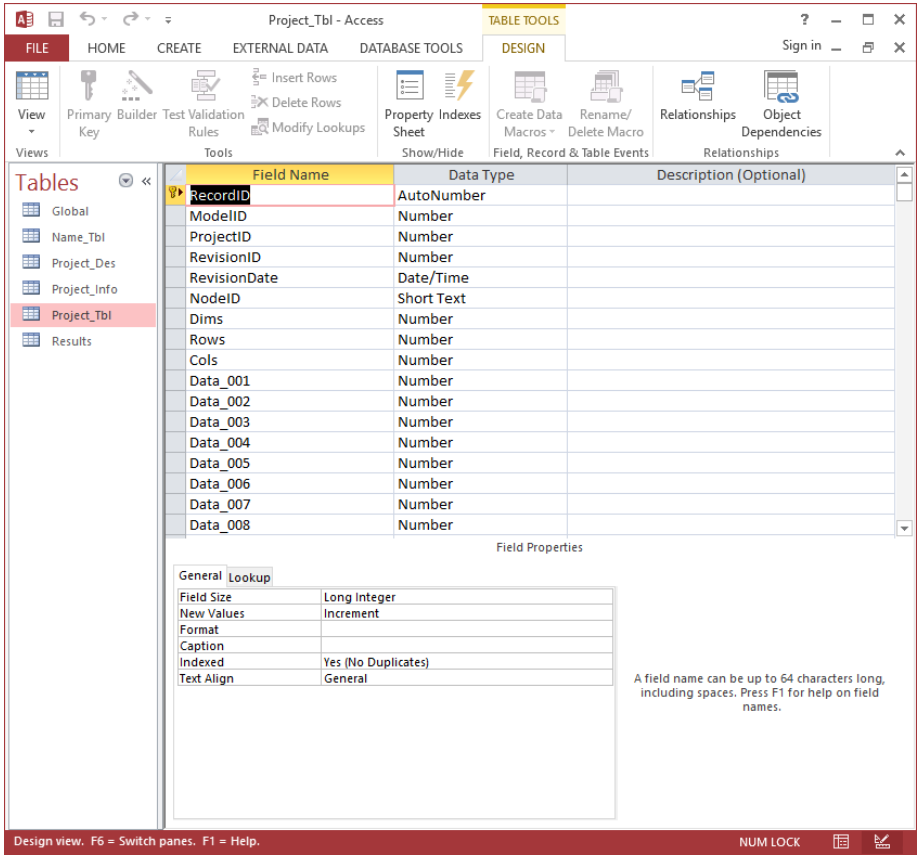


Figure 21-6. Access Design View of Project\_Tbl

Figure 21-7 shows the Access datasheet view for Project\_Tbl.

Record	Model	Project	Revisor	RevisionDate	NodeID	Dims	Rows	Cols	Data_001	Data_002
438	1	1	0	12/21/2012	probs_phase_1	1	0	2	0.71	0.29
434	1	1	1	12/28/2012	probs_phase_1	1	0	2	0.71	0.29
435	1	1	2	2/1/2013 12:30:00 PM	probs_phase_1	1	0	2	0.71	0.29
436	1	1	3	2/2/2013 12:29:00 PM	probs_phase_1	1	0	2	0.71	0.29
437	1	1	4	2/4/2013 4:22:00 PM	probs_phase_1	1	0	2	0.71	0.29
438	1	1	0	12/21/2012	probs_phase_2	1	0	2	0.41	0.59
439	1	1	1	12/28/2012	probs_phase_2	1	0	2	0.41	0.59
440	1	1	2	2/1/2013 12:30:00 PM	probs_phase_2	1	0	2	0.41	0.59
441	1	1	3	2/2/2013 12:29:00 PM	probs_phase_2	1	0	2	0.41	0.59
442	1	1	4	2/4/2013 4:22:00 PM	probs_phase_2	1	0	2	0.41	0.59
443	1	1	0	12/21/2012	probs_phase_3	1	0	2	0.81	0.19
444	1	1	1	12/28/2012	probs_phase_3	1	0	2	0.81	0.19
445	1	1	2	2/1/2013 12:30:00 PM	probs_phase_3	1	0	2	0.81	0.19
446	1	1	3	2/2/2013 12:29:00 PM	probs_phase_3	1	0	2	0.81	0.19
447	1	1	4	2/4/2013 4:22:00 PM	probs_phase_3	1	0	2	0.81	0.19
448	1	1	0	12/21/2012	probs_regulatory_approval	1	0	2	0.91	0.09
449	1	1	1	12/28/2012	probs_regulatory_approval	1	0	2	0.91	0.09
450	1	1	2	2/1/2013 12:30:00 PM	probs_regulatory_approval	1	0	2	0.91	0.09
451	1	1	3	2/2/2013 12:29:00 PM	probs_regulatory_approval	1	0	2	0.91	0.09
452	1	1	4	2/4/2013 4:22:00 PM	probs_regulatory_approval	1	0	2	0.91	0.09
453	1	1	0	12/21/2012	probs_strength_of_competiti	1	0	3	0.1	0.55
454	1	1	1	12/28/2012	probs_strength_of_competiti	1	0	3	0.1	0.55
455	1	1	2	2/1/2013 12:30:00 PM	probs_strength_of_competiti	1	0	3	0.1	0.55
456	1	1	3	2/2/2013 12:29:00 PM	probs_strength_of_competiti	1	0	3	0.1	0.55
457	1	1	4	2/4/2013 4:22:00 PM	probs_strength_of_competiti	1	0	3	0.1	0.55
458	1	1	0	12/21/2012	probs_market_share	2	3	3	0	0.4
459	1	1	1	12/28/2012	probs_market_share	2	3	3	0	0.4
460	1	1	2	2/1/2013 12:30:00 PM	probs_market_share	2	3	3	0	0.4
461	1	1	3	2/2/2013 12:29:00 PM	probs_market_share	2	3	3	0	0.4
462	1	1	4	2/4/2013 4:22:00 PM	probs_market_share	2	3	3	0	0.4

Figure 21-7. Access Datasheet View of Project\_Tbl



Figure 21-8 shows the Access design view for a Node ID structured table containing string information called Project\_Des.

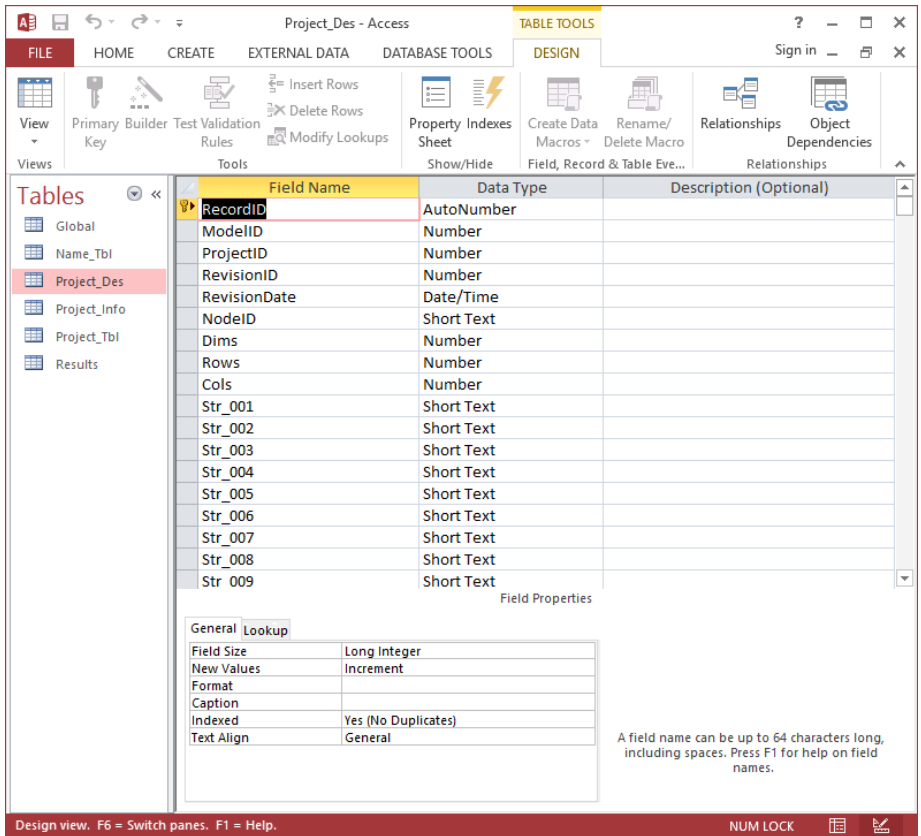


Figure 21-8. Access Design View of Project\_Des

Figure 21-9 shows the Access datasheet view for a Node ID structured table containing string information called Project\_Des.

RecordID	ModelID	ProjectID	RevisionID	RevisionDat	NodeID	Dims	Rows
1	1	1	0	2/20/2013	Project_Description	0	0
2	1	2	0	2/20/2013	Project_Description	0	0
3	1	3	0	2/20/2013	Project_Description	0	0
4	1	4	0	2/20/2013	Project_Description	0	0
5	1	5	0	2/20/2013	Project_Description	0	0
6	1	6	0	2/20/2013	Project_Description	0	0
7	1	1	0	2/20/2013	Project_Assumptions	1	1
8	1	2	0	2/20/2013	Project_Assumptions	1	1
9	1	3	0	2/20/2013	Project_Assumptions	1	1
10	1	4	0	2/20/2013	Project_Assumptions	1	1
11	1	5	0	2/20/2013	Project_Assumptions	1	1
12	1	6	0	2/20/2013	Project_Assumptions	1	1
13	1	1	0	2/20/2013	Project_Name	0	0
14	1	2	0	2/20/2013	Project_Name	0	0
15	1	3	0	2/20/2013	Project_Name	0	0
16	1	4	0	2/20/2013	Project_Name	0	0
17	1	5	0	2/20/2013	Project_Name	0	0
18	1	6	0	2/20/2013	Project_Name	0	0
19	1	1	0	2/20/2013	Cost_Assumptions	2	2
20	1	2	0	2/20/2013	Cost_Assumptions	2	2
21	1	3	0	2/20/2013	Cost_Assumptions	2	2
22	1	4	0	2/20/2013	Cost_Assumptions	2	2
23	1	5	0	2/20/2013	Cost_Assumptions	2	2
24	1	6	0	2/20/2013	Cost_Assumptions	2	2

Figure 21-9. Access Datasheet View of Project\_Des

### 21.3.3 Table Names/Field Names

Table names and field names in a database table accessed by DPL database cannot contain any punctuation or spaces. They can contain letters or numbers but they must start with a letter. Database management systems may vary regarding whether table names and field names are case sensitive. To be consistent with other identifiers in DPL, table names and field names are case sensitive in DPL. Because a particular database management system may or may not be case sensitive with regard to table and field names, you cannot have table(field) names that differ only in case, e.g., both "field1" and "Field1" are not allowed. The specific database management system you are using may have other restrictions on table names and field names. Please check with your database documentation.

### 21.3.4 Scalar/Simple String Data

If you have a large number of scalar values and/or simple string values (i.e., not arrays of strings) that DPL needs to access, these data can be stored in a table that is not Node ID structured (i.e., having Node ID, Dims, Rows, Cols, Data\_001, etc.) as described above. The data may be stored in tables in which the field/column names in the table identify the data, i.e., a more standard database table structure. Tables that are not Node ID structured must still have Model ID and/or Project ID fields as appropriate and revision tracking fields as appropriate. Only scalar values and simple string values can be stored in a table not using the Node ID structure.

## 21.4 Configuring Database Access within DPL™

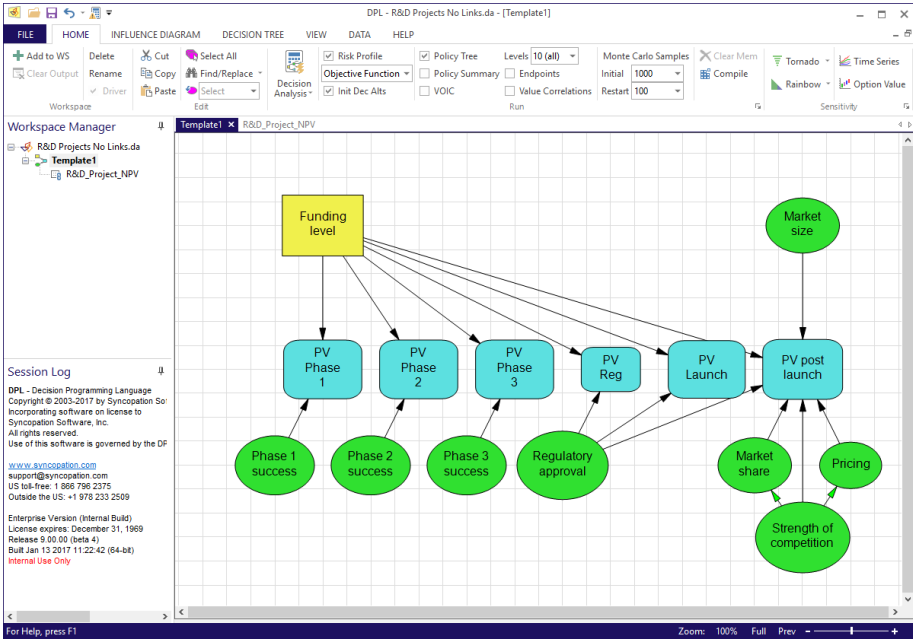
---

Once you have set up the database that you wish DPL to access, you need to give DPL some database configuration information. You do this via the Database Specification dialog. You must give this configuration information to DPL before you can set up any database links within a model in DPL.

You will do this now.

- ⇒ Start DPL.
- ⇒ Select File | Open.
- ⇒ Navigate to the Examples folder underneath the DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples.
- ⇒ Select R&D Project No Links.da and click Open.

DPL opens the Workspace as shown in Figure 21-10.



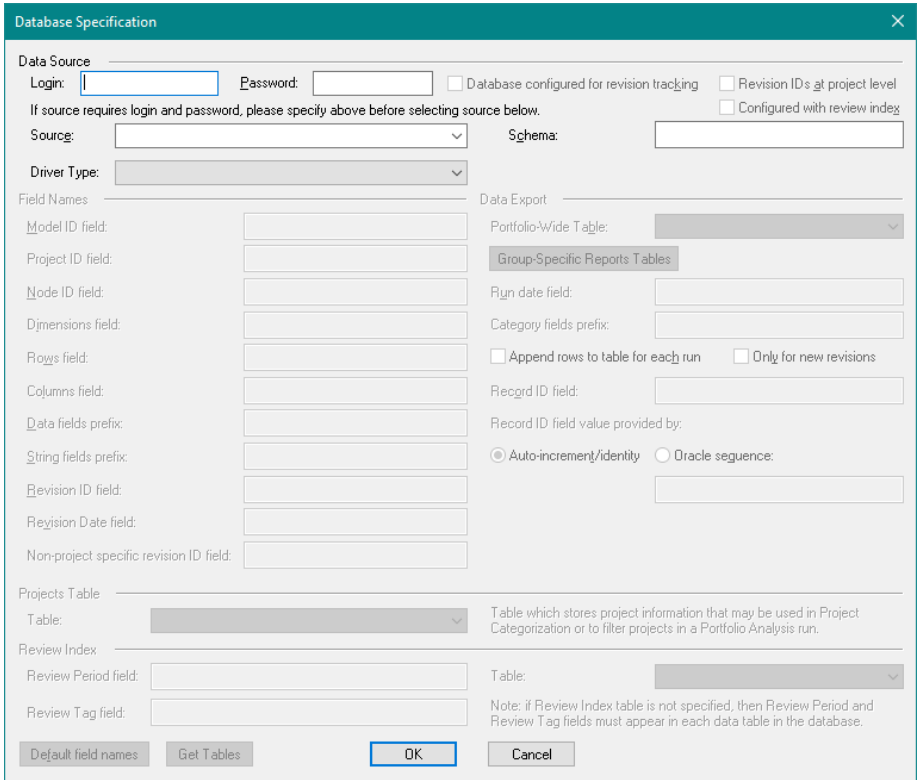
**Figure 21-10. R&D Projects No Links Workspace**

This Workspace contains a template model called Template1 for a pharmaceutical development example. The data required for the model is stored in the database R&D Projects.mdb for which you set up an ODBC Data Source in Section 21.2.1. If you did not complete those steps, you will need to do so before proceeding with the tutorial that follows.

Further, the model uses a converted Excel spreadsheet in a DPL program (R&D\_Project\_NPV) to perform the cash flow calculations. The chance and decision nodes in the Influence Diagram are calculation linked as driver nodes to the program. As DPL analyzes the model, the driver nodes send data to the program. The value nodes in the Influence Diagram are linked as metric nodes to the program; as DPL calculates get/pay expressions throughout the Decision Tree, calculated results are sent back from the program.

⇒ Click Data | Database | Set Up

The Database Specification dialog appears as shown in Figure 21-11.



**Figure 21-11. Database Specification Dialog**

- ⇒ In the *Source* drop-down list, select R&D Projects from the list.
- ⇒ For the *Driver Type* drop-down list, select Access.

Note: Often a Data Source for a corporate database or other multi-user database will require a login and password. R&D Projects does not. If the Data Source requires a login and password, you should specify those before selecting the Data Source from the drop-down list. When you select a Data Source from the drop-down list, DPL attempts to connect to the database for the Data Source. If the Data Source requires a login and password, the connection will fail unless these have been provided first.

The R&D Projects database contained in R&D Projects.mdb uses the default field names for the fields that DPL requires to access the tables. DPL provides a quick way to set up the field names in this situation.

- ⇒ Click the Default field names button at the bottom left-hand corner of the dialog. DPL fills in the required field names. See Figure 21-12.

**Figure 21-12. Database Specification Dialog with Field Names**

Note: if your database does not use the default field names, then you may edit the field names in the edit boxes on the dialog.

⇒ Click OK to close the Database Specification dialog.

If you connected successfully, you might have briefly seen a Loading database information dialog. If you receive an error, you may need to contact your IT department.

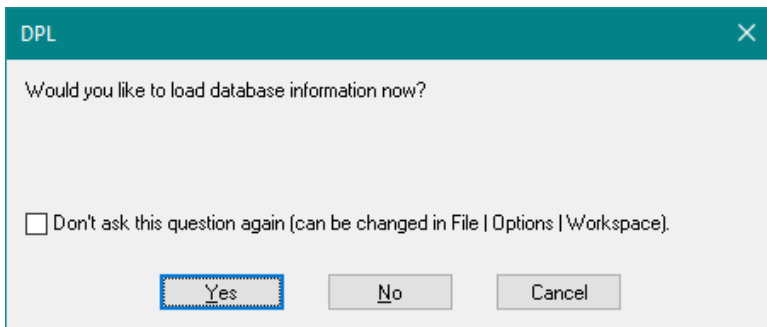
You have now told DPL what it needs to know in order to access the data stored in the R&D Projects database. As mentioned above when you change the Data Source using the Database Specification dialog, DPL loads information about the database for the data source when you click OK to close the dialog. See the next section for more information.

⇒ Save your Workspace file under a new name.

## 21.5 Loading Database Schema

When you change the Data Source in the Database Specification dialog, DPL loads the database schema for the Data Source. Specifically, DPL gathers the table names, field names within each table and other information about the database. For a database that is on a remote server or is sizeable, it may take a few minutes to load this information and you may notice a delay. DPL uses this information in a number of places. This information only needs to be loaded once (and only needs to be re-loaded if the structure of the database has changed, e.g., if new tables have been added, or new fields to tables). The schema information is saved with the DPL Workspace file when you save it.

If you know the structure of the database has changed, you may ask DPL to load database schema by going to Data | Database | Load Schema. Further, when you first set up a DPL Workspace file with a Data Source specified, DPL will prompt you as to whether you wish to load database information each time the Workspace is opened. See Figure 21-13.



**Figure 21-13. Load Database Information Prompt**

During the development phase of the database, the structure may change and you may wish to answer Yes to this question and check the "Don't ask this question again" checkbox. DPL will then automatically load the database schema each time the Workspace is opened. Alternatively, if you don't check the checkbox, you will be asked each time the Workspace is loaded. Leaving the checkbox unchecked may be wise for large and/or remote databases, particularly if you work offline and might not have access to the database. Once you have finalized the design of the database, you no longer need to regularly load the schema. At this point, it makes sense to change the setting to "Don't load" via File | Options | Workspace.

Lastly, as indicated above, the information that DPL gathers when loading the database schema is structural in nature. DPL is not loading the actual data stored in the tables when it loads the schema information. DPL does the data extraction from the database when you run a database-linked model or when you create a program from a database-linked model.

## 21.6 Creating Database-Linked Models

---

In order to create a database-linked model, you must first set up the Data Source using the Database Specification dialog via Data | Database | Set Up. This is explained in Section 21.4. If you have not already completed the tutorial in that section, please do so now.

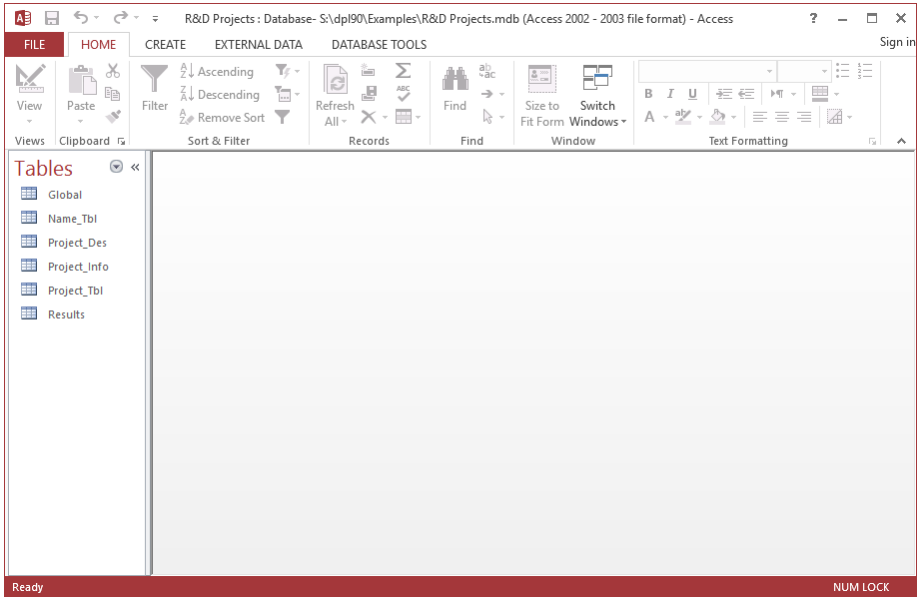
- ⇒ If it is not already open, browse to find the file that you saved from Section 21.4 and open it now.
- ⇒ If the Load Database Information prompt comes up, select No and check "Don't ask this again".

### 21.6.1 Linking Existing Nodes to a Database

The file that you started with in Section 21.4 contains a number of nodes that are missing node data and that are ready to be linked to the database. This section will show you how to add database links to an existing node. Before you do this, you will explore the database R&D Projects database. If you do not have Access on your computer, you will not be able to explore the database but you may wish to read through this section and explore the figures.

- ⇒ If you have Access on your computer, use Windows Explorer to Navigate to the Examples folder underneath the DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples.
- ⇒ Double-click on R&D Projects.mdb to open it in Access. See Figure 21-14.





**Figure 21-14. R&D Projects Database Open in Access**

⇒ In the Navigation pane on the left, double-click on Project\_Tbl to open it in datasheet view. See Figure 21-15.

The screenshot shows the Microsoft Access interface with the 'Project\_Tbl' table open in Datasheet View. The table has the following columns: Record, Model, Project, Revisor, RevisionDate, NodeID, Dir, Row, Co, and Data\_0C. The data is as follows:

Record	Model	Project	Revisor	RevisionDate	NodeID	Dir	Row	Co	Data_0C
433	1	1	0	12/21/2012	probs_phase_1	1	0	2	0.7
434	1	1	1	12/28/2012	probs_phase_1	1	0	2	0.7
435	1	1	2	2/1/2013 12:30:00 PM	probs_phase_1	1	0	2	0.7
436	1	1	3	2/2/2013 12:29:00 PM	probs_phase_1	1	0	2	0.7
437	1	1	4	2/4/2013 4:22:00 PM	probs_phase_1	1	0	2	0.7
438	1	1	0	12/21/2012	probs_phase_2	1	0	2	0.4
439	1	1	1	12/28/2012	probs_phase_2	1	0	2	0.4
440	1	1	2	2/1/2013 12:30:00 PM	probs_phase_2	1	0	2	0.4
441	1	1	3	2/2/2013 12:29:00 PM	probs_phase_2	1	0	2	0.4
442	1	1	4	2/4/2013 4:22:00 PM	probs_phase_2	1	0	2	0.4
443	1	1	0	12/21/2012	probs_phase_3	1	0	2	0.8
444	1	1	1	12/28/2012	probs_phase_3	1	0	2	0.8
445	1	1	2	2/1/2013 12:30:00 PM	probs_phase_3	1	0	2	0.8
446	1	1	3	2/2/2013 12:29:00 PM	probs_phase_3	1	0	2	0.8
447	1	1	4	2/4/2013 4:22:00 PM	probs_phase_3	1	0	2	0.8

**Figure 21-15. Project\_Tbl Open in Datasheet View**

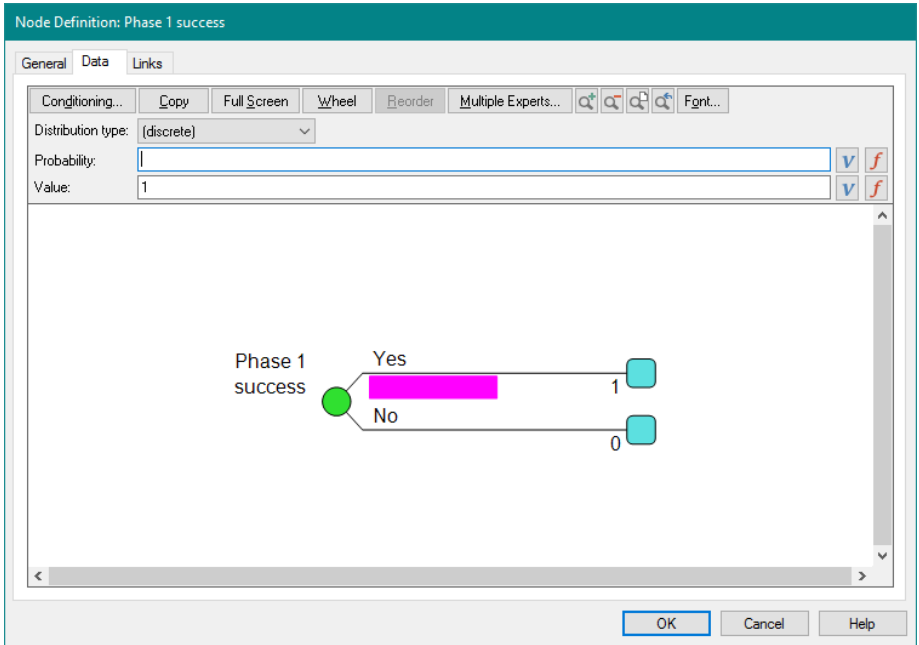
Note that there are a number of datasets in the table. Specifically there are records with Node IDs for six projects which are identified by Model ID = 1 (all of them have the same Model ID) and Project IDs 1 through 6. This database is also configured for revision tracking (this will be covered in Section 21.7), so each Model ID and Project ID combination has a number of revisions for each Node ID. For example, the first five rows of the table all store different revisions for the Node ID probs\_phase\_1. Note: the data may be the same for each of the revisions.

- ⇒ Explore the table some more if you'd like.
- ⇒ Do not make any edits to the data.

If you are unfamiliar with Access, you should know that any edits you make are instantly saved. You are not prompted to save changes when you close Access; the edits are automatically saved as you make them.

Note in particular that there is a project with Model ID = 1 and Project ID = 1. You will use this in DPL.

- ⇒ Close Access.
- ⇒ Switch back to DPL.
- ⇒ Double-click on the Phase 1 success node. Note that the node has no probability data. See Figure 21-16.



**Figure 21-16. Node Definition Dialog for Phase 1 Success**

As you may have noticed while exploring the database, the probability data for this node as well as the other chance nodes in the model are stored in the database in a table called Project\_Tbl.

- ⇒ Switch to the Links tab.
- ⇒ In the *Initialization links* section, select Database.

Note: database links are always initialization links. I.e., DPL will get the data for the node from the database once at the beginning of a run. The data is then used in the subsequent analysis.

When setting up a database link for a node, you must tell DPL the Model ID and/or Project ID of the record you wish to link to. This is also done in the *Initialization links* section, and DPL displays the Model ID and Project ID edit boxes on the Links tab when Database is selected as the Initialization Link type.

- ⇒ Type 1 in the Model ID edit box.
- ⇒ Type 1 in the Project ID edit box. Note: this is the project that you saw when you browsed the database table. The Links tab should now look like Figure 21-17.

Node Definition: Phase 1 success

General Data Links

Calculation links

These links are recalculated throughout the run. They may be either inputs used in the linked spreadsheet's calculations (e.g., unit sales) or calculated results from the spreadsheet (e.g., NPV).

None (local)
  DPL Program
  Microsoft Excel

Workbook:

Value:

This is a driver node. Its value will be sent to the DPL Calculation program each time it changes during the run.

---


Initialization links

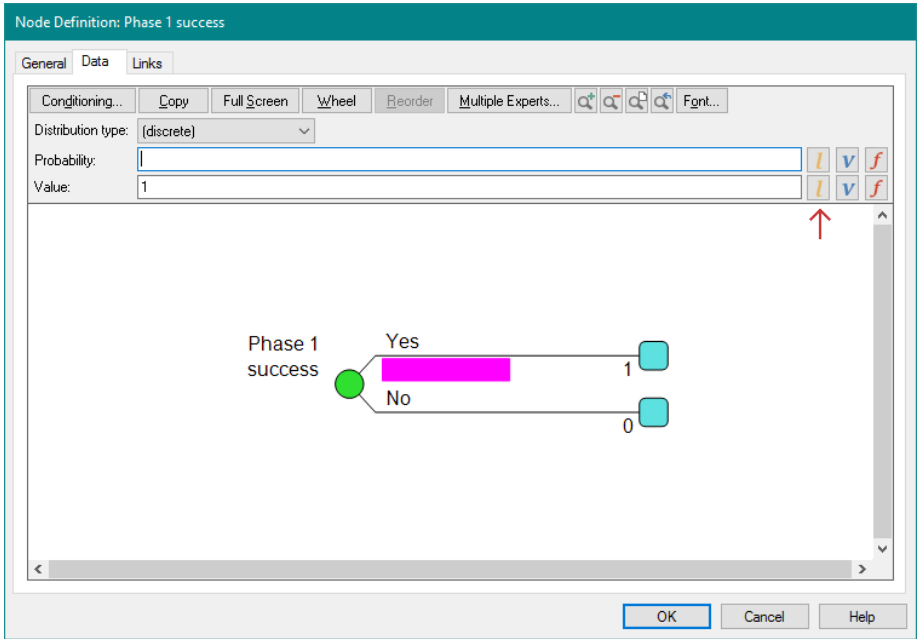
These links are for connecting to a spreadsheet, database or program that contains data (values and/or probabilities) which will be used in the Data tab to initialize nodes. These links are refreshed once at the start of each run.

None (local or DPL program)
  Microsoft Excel
  Database

Model ID:  Project ID:


**Figure 21-17. Completed Links Tab for Phase 1 Success**

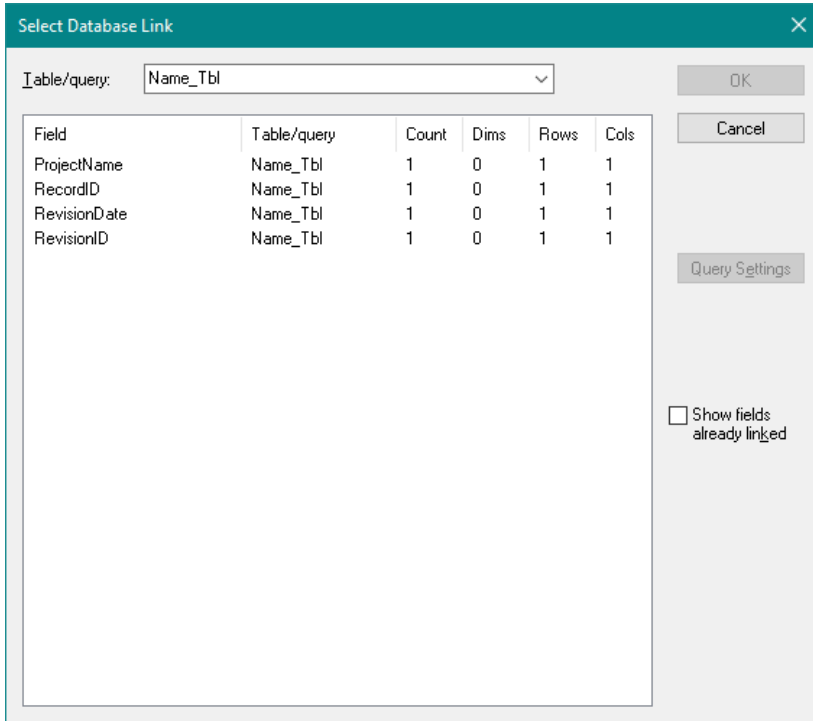
⇒ Switch back to the Data tab. Note: that there is now a Link button (  ) next to the probability and value edit boxes. See Figure 21-18.



**Figure 21-18. Data Tab with Link Button**

The Link button allows you to tell DPL the table and Node ID that the node is linked to via the Select Database Link dialog.

- ⇒ Click the Link button (  ) next to the probability edit box. See Figure 21-19.



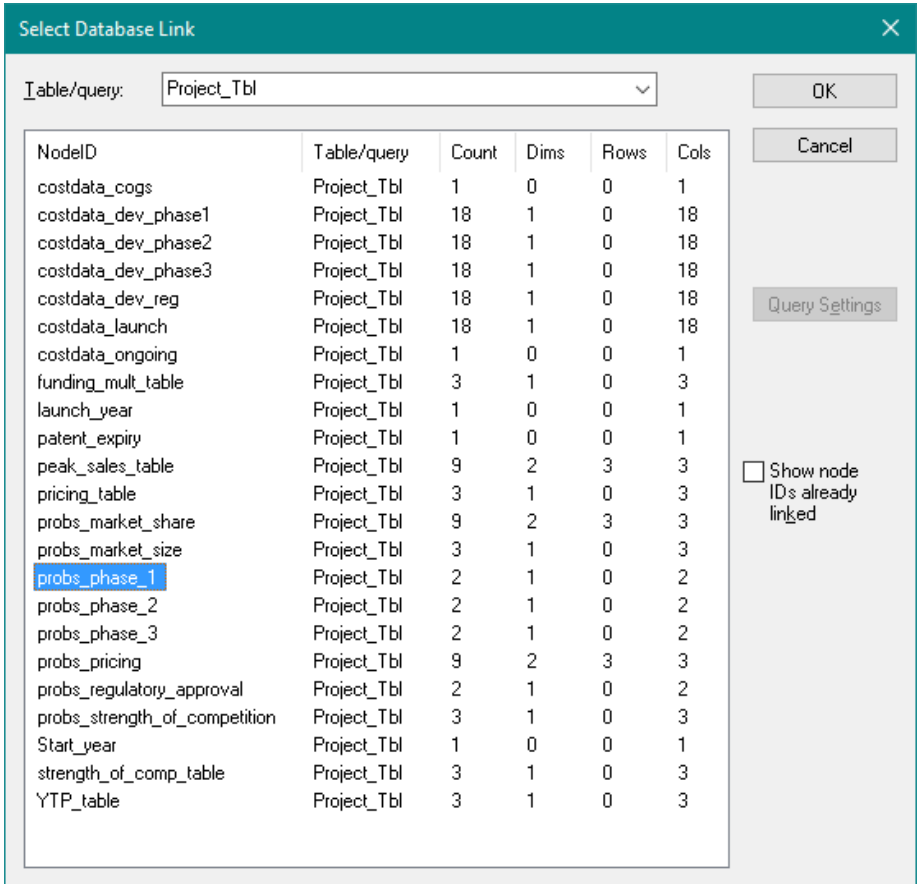
**Figure 21-19. Select Database Link Dialog**

The Select Database Link dialog provides a combo box at the top for you to select the table in the database for the link. When you select a table, the list below the combo box is populated with the Node IDs within that table. Currently, the table selected is Name\_Tbl (this is a descriptive table with string data in it; you will not be using this table).

⇒ Use the drop-down list to select Project\_Tbl.

The list of Node IDs also contains information about the data for each Node ID. The list tells you the total count of data elements for the item, the dimensions, rows and columns.

⇒ Within the list, select probs\_phase\_1 as the Node ID. See Figure 21-20.

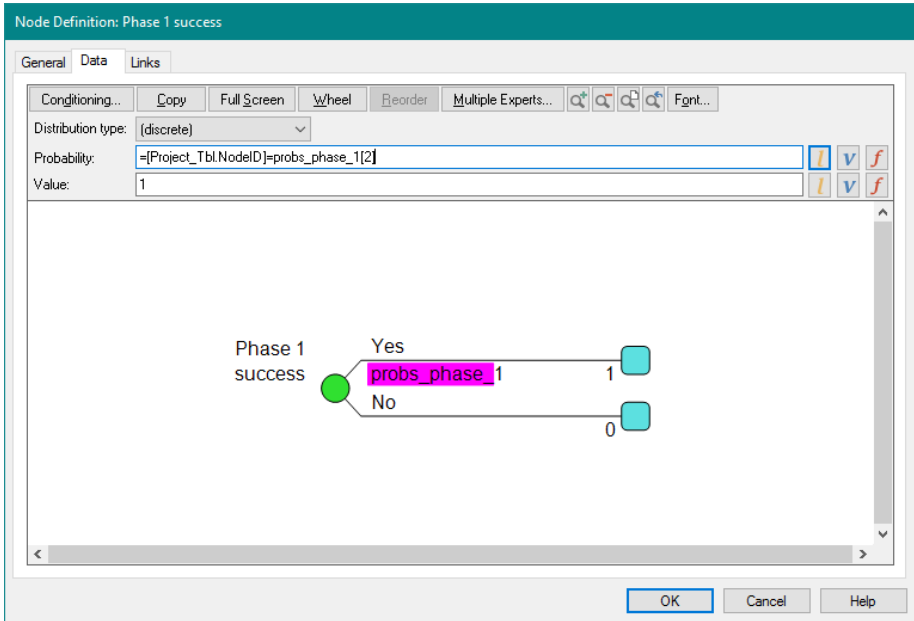


**Figure 21-20. Probs\_phase\_1 Selected in Select Database Link**

Note that Phase 1 Success is a two-outcome chance event and that you are selecting a Node ID with two data elements for its probabilities, so the Count column reads "2".

Also note that the "Show Node IDs already linked" checkbox is unchecked by default. As you link additional Node IDs to your model later in this section, you will want to leave this checkbox unchecked so that you will only be selecting from Node IDs that are not yet linked.

- ⇒ Click OK to close the Select Database Link dialog. DPL fills in the database link for the probability node data. See Figure 21-21.



**Figure 21-21. Database Link Node Data for Phase 1 Success**

The syntax for the database link node data is as follows.

```
= [Project_Tbl.NodeID]=probs_phase_1 [2]
```

The node data must start with "=" to indicate a database link. The information contained within the square brackets is of the form `table_name.NodeID` which indicates to DPL which table the Node ID is in, e.g., `Project_Tbl`. Immediately following the closed square bracket is another equal sign. Following this second equal sign is the Node ID that the node is linked to, e.g., `probs_phase_1`. The information following the Node ID tells DPL the dimensionality of the data, i.e., this is a two column row array. For more information on arrays within DPL, see Chapter 9.

As with Excel initialization links, when you use a database initialization link, the link only appears on the first branch of the node (for Phase 1 Success this is the Yes branch). In this example, you are using a database initialization link for the probability data. Therefore, the remaining probability data for the node must be blank. The same applies for value data. The initialization link appears on the first branch and all remaining value data must be blank.

DPL will only allow you to put an initialization link on the first branch.



- ⇒ Press the down arrow key twice to move the selection to the probability data for the No branch.


Note that the Link buttons for both the probability data and the value data are now disabled.

- ⇒ Click OK to close the Node Definition dialog.

You have now specified a database link for the Phase 1 Success node. Note: the value data for the node (1, 0) is not stored in the database.

- ⇒ Double-click Phase 2 success to edit its definition.
- ⇒ Switch to the Links tab.
- ⇒ Select Database in the *Initialization links* section.

Note that DPL fills in the Model ID and Project ID that you used previously. You can have database links to multiple Model ID/Project ID records within a model, though in most cases this won't be necessary. DPL assumes you want to use the same Model ID/Project ID as the existing link(s).

- ⇒ Switch to the Data tab.
- ⇒ Click the Link () button. The Select Database Link dialog appears. This time it has Project\_Tbl already selected since that is the last table you used.
- ⇒ Select probs\_phase\_2 in the list.
- ⇒ Click OK to close the Select Database Link dialog. Again, DPL fills in the database link for the Yes branch of the node and leaves the No branch blank.
- ⇒ Click OK to close the Node Definition dialog.

Repeat the above procedure to create database initialization links for the probabilities of the remaining chance nodes in the model using the Node ID for each node as indicated in Table 21-3. As mentioned earlier, leave the Show Node IDs already linked checkbox unchecked, so that after you link a Node ID it will not appear in the list the next time you use the dialog. Note some of the chance nodes have separate conditioning. To specify the Node ID for these chance nodes, use the Probabilities tab.

Node	Node ID
Phase 3 success	probs_phase_3
Regulatory approval	probs_regulatory_approval
Market size	probs_market_size
Market share	probs_market_share
Pricing	probs_pricing
Strength of competition	probs_strength_of_competition

**Table 21-3. Node IDs for Probability Database Initialization Links**

⇒ Save your file.

When you created the database initialization link for the Market share node, you may have noted that the syntax for the database initialization link for it is:

```
=[Project_Tbl.NodeID]=probs_market_share[3][3]
```

The Node ID for the market share probabilities is a two-dimensional array. Market share is conditioned by Strength of competition. Both Market share and Strength of competition are three-outcome chance nodes. Therefore, nine probabilities are needed for Market share and these are stored in a 3 by 3 array. DPL indicates this dimensionality in the database initialization link by adding "[3][3]" following the Node ID.

The Pricing node is also conditioned and also uses a two-dimensional (3 by 3) array link.

The model is now ready to run.

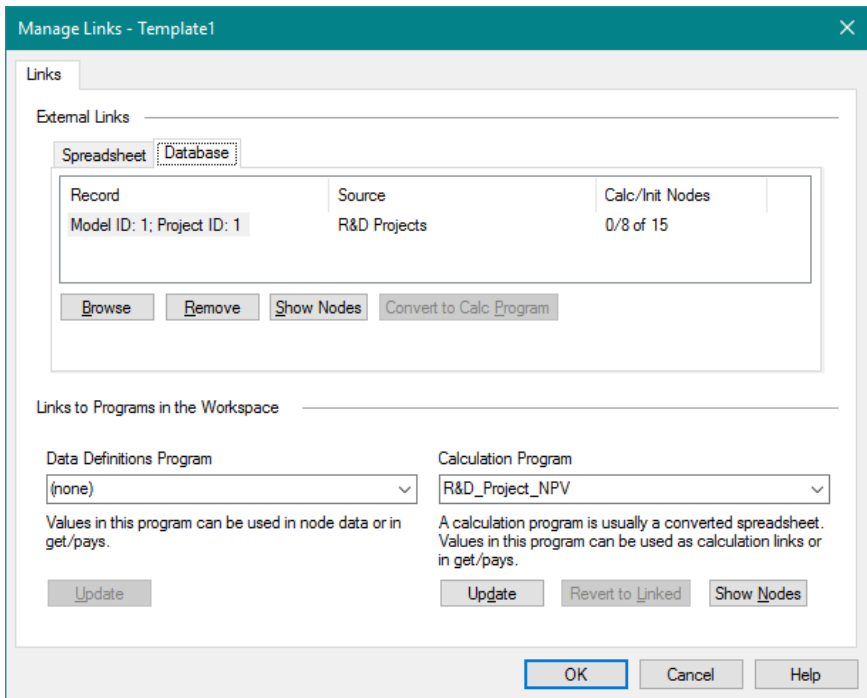
⇒ Press F10 to run a decision analysis.

DPL extracts the data for the probabilities for each chance node from the database and produces the requested results.

### 21.6.2 Changing Records for Database Initialization Links

The Manage Links dialog displays all the records (Model ID, Project ID combinations) that the model is linked to. In this dialog, you can also change records that the model is linked to. You will do this now in order to see results for a different project in the R&D Projects database.

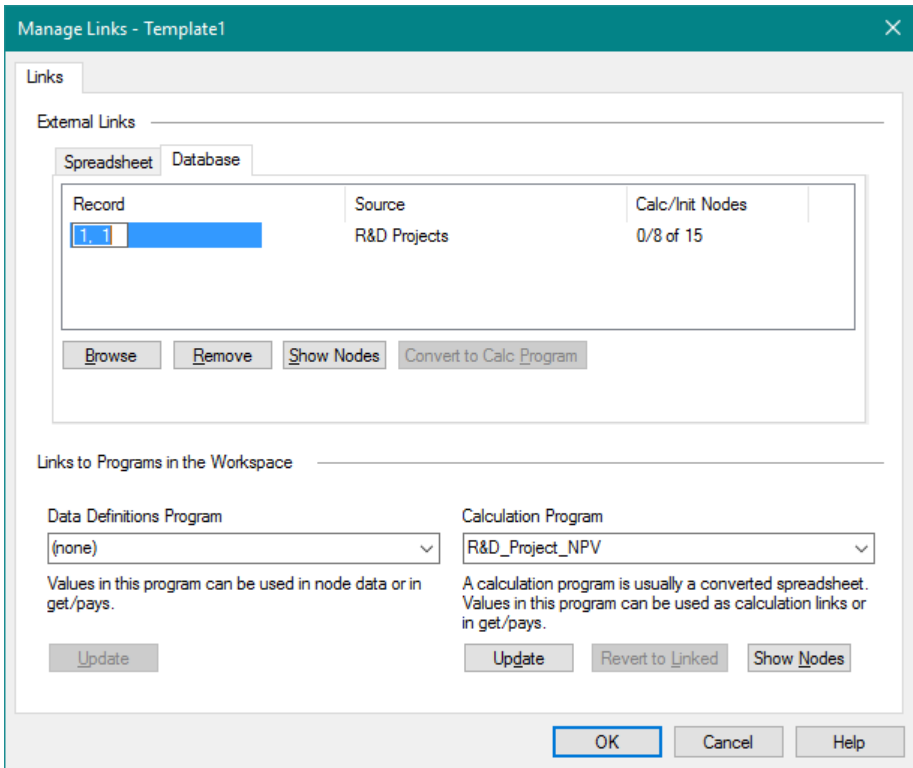
- ⇒ Activate the Template1 model.
- ⇒ Click Influence Diagram | Links | Manage. The Manage Links dialog appears as shown in Figure 21-22.



**Figure 21-22. Manage Links Dialog**

This model is currently linked to one record (Model ID/Project ID combination), namely Model ID = 1 and Project ID = 1.

- ⇒ Select the record in the first row of the Database Links list box.
- ⇒ Press F2 to edit the record. Note that the record changes to two comma separated numbers. See Figure 21-23.



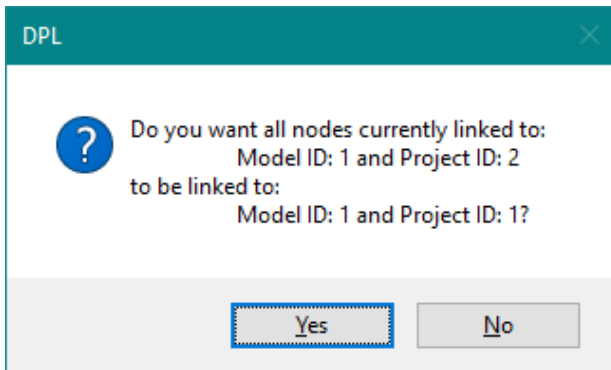
**Figure 21-23. Editing a Record in the Manage Links Dialog**

- ⇒ Type "1, 2" for the record. Note: you must separate the two numbers with a comma.
- ⇒ Press Enter.
- ⇒ Click OK to close the Manage Links dialog.
- ⇒ Double-click Phase 1 success to edit its definition.
- ⇒ Switch to the Links tab. Note that within the *Initialization links* section the Project ID is now 2.
- ⇒ Click Cancel.
- ⇒ Press F10 to run a Decision Analysis. Note that the results are substantially different from what you got for Model ID = 1, Project ID = 1.

Note: when you change the record that a model is linked to, none of the tables it is linked to change. If you look at the node data for Phase 1 success, you can see it is still linked to Project\_Tbl.

As mentioned previously, you can link a model to multiple records (either multiple Project IDs or multiple Model ID/Project ID combinations). If you wish to link a node to a new record, you do this via the Links tab of the Node Definition dialog. You will try this now.

- ⇒ Activate the Template1 model.
- ⇒ Double-click Phase 1 success to edit its definition.
- ⇒ Switch to the Links tab.
- ⇒ Set Project ID to be 1.
- ⇒ Click OK. You will get the prompt shown in Figure 21-24.



**Figure 21-24. Change Record Prompt**

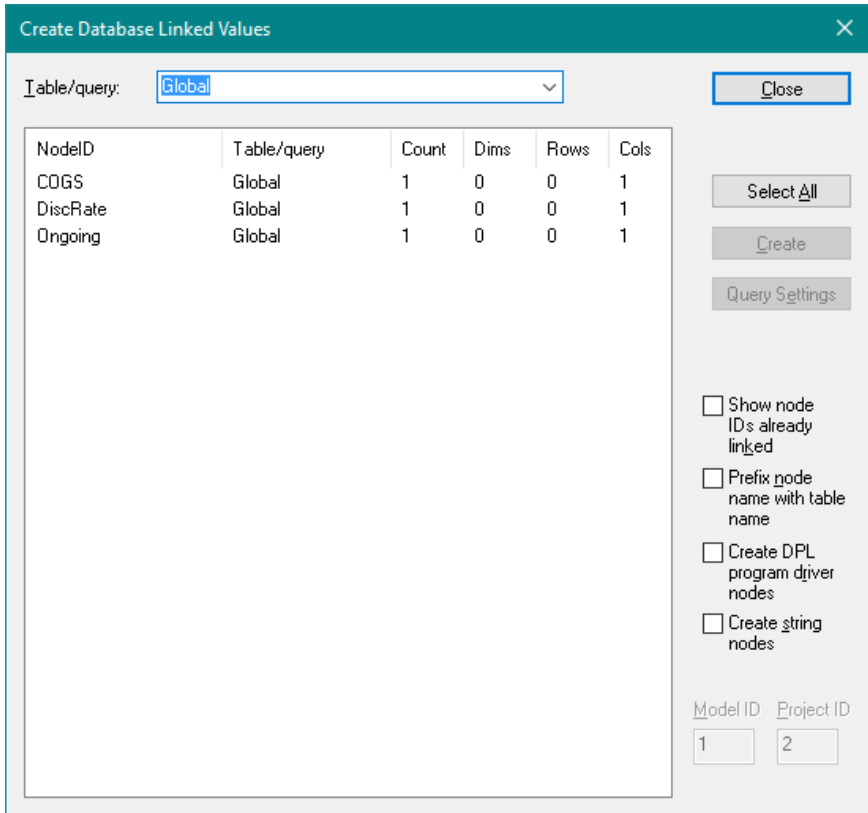
- ⇒ Answer No to the prompt.

By answering no, you are telling DPL not to change any other nodes linked to record Model ID = 1, Project ID = 2. Phase 1 success will now be linked to Model ID = 1, Project ID = 1, while the remaining chance nodes are linked to Model ID = 1, Project ID = 2. Therefore, the probability of success for phase 1 is from project 1 in the database, while the rest of the probabilities are from project 2.

- ⇒ Open the Manage Links dialog to see this (Influence Diagram | Links | Manage)
- ⇒ Click Cancel to close the dialog.
- ⇒ Press F10 to run a decision analysis. The results are different yet again.
- ⇒ Activate the model and open the Manage Links dialog again.
- ⇒ Click Undo to restore your model to only be linked to Model ID = 1, Project ID = 2



- ⇒ Double-click Template1 in the Workspace Manager to activate it.
- ⇒ Drop-down the Influence Diagram | Node | Linked Node split button and choose Database Initialization-Linked... from the list. The Create Database Linked Values dialog comes up as shown in Figure 21-26.

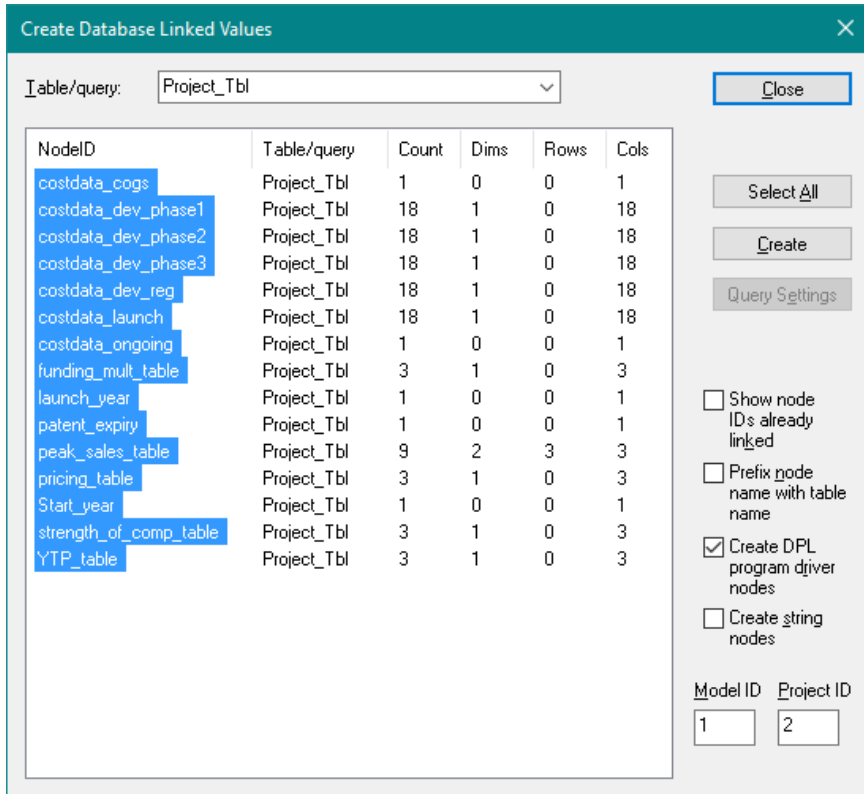


**Figure 21-26. Create Database Linked Values Dialog**

This dialog allows you to select one or more Node IDs from one or more tables in the database that the DPL Workspace is connected to, and create database-linked value nodes for each. It displays similar information to the information in the Select Database Link dialog. In addition, you can tell DPL whether to prefix node names with the table name, create DPL program driver nodes, or create string nodes.

- ⇒ If Project\_Tbl is not selected in the *Table/query* combo box, select it.
- ⇒ Make sure the Show Node IDs already linked checkbox is still unchecked.

- ⇒ Check the Create DPL program driver nodes checkbox.
- ⇒ Click the Select All button. The dialog should look like Figure 21-27. You have selected all the Node IDs in the Project\_Tbl table that were not already linked to your model.
- ⇒ Click Create.

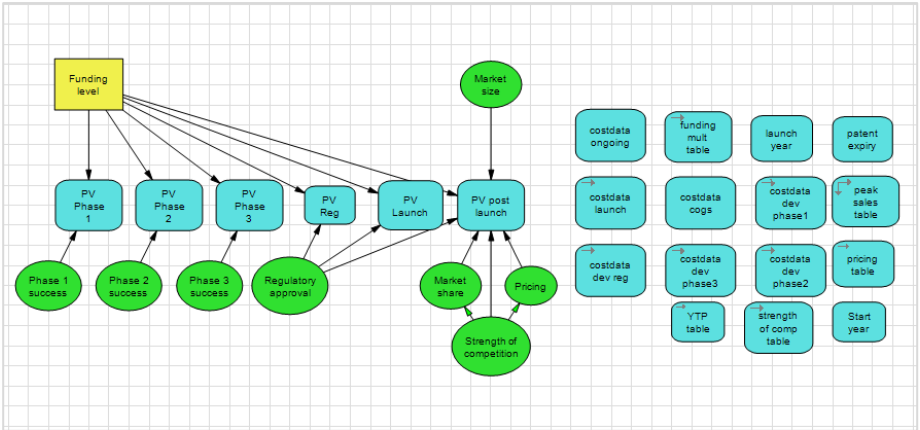


**Figure 21-27. Selected Node IDs**

If you needed to create database-linked nodes for Node IDs from another table, you could select that table now, select the Node IDs and click Create again. In this example, you do not need to do that.

- ⇒ Click Close to close the Create Database Linked Values.
- ⇒ Click the Full button on the status bar or press Ctrl+L to Zoom Full. The newly created nodes are off to the right of the previously existing nodes. See Figure 21-28.





**Figure 21-28. Influence Diagram with New Database-Linked Nodes**

Notice that DPL automatically created the appropriate array values based on the dimension of the data.

- ⇒ Double-click the `costdata ongoing` node. You can see that it is linked to the Node ID `costdata_ongoing`.
- ⇒ Switch to the Links tab. The node is database initialization linked to Model ID = 1, Project ID = 2. Also the node is calculation linked to the value `costdata_ongoing` in the DPL program `R&D_Project_NPV`.

When you created these nodes, you told DPL to create DPL program driver nodes. Now when you run the model, DPL will extract the data from the database for these new nodes and send it to the DPL program (i.e., the data extracted from the database will override the data for these values/arrays in the DPL program). Note for the driver nodes to work correctly the items being overridden in the DPL program must have the same names as the Node IDs in the database.

- ⇒ Double-click the `peak sales table` node. You can see that DPL created a two-dimensional array with 3 rows and 3 columns and that the array is linked to the 3 by 3 Node ID `peak_sales_table`.
- ⇒ Run a Decision Analysis.

The results are different from when you first changed the model link record to Model ID = 1, Project ID = 2 because the costs and other assumptions DPL is using are now extracted from the database for project 2, whereas previously DPL was using the assumptions in the DPL program which differ from what is in the database for project 2.

### 21.6.4 Node Data Syntax for Tables Not Using Node ID Structure

As mentioned in Section 21.3.4, you may store scalar and simple string data in a table that does not use the Node ID structure. The syntax for the database link node data is slightly different in this case. The syntax is as follows

```
=[Project_Info.InPortfolio]
```

As with nodes linked to tables containing Node IDs, the node data must start with "=[\" to indicate a database link. In this case though, the information contained within the square brackets is of the form `table_name.field_name` which indicates to DPL which table the data is in, e.g., `Project_Info` and which field the data is in, e.g., `InPortfolio`. Nothing follows the closed square bracket. No dimensionality information is needed since the node must be a scalar or simple string.

You may use same the methods described in Sections 21.6.1 and 21.6.3 to link existing nodes and create new database linked nodes to non-Node ID – structured tables for scalars and simple strings.

## 21.7 Databases Configured for Revision Tracking

---

One reason to store data in a database is to keep track of the various revisions that the data may go through while the project or asset is being analyzed.

If you have set up your database so that it tracks revisions, you must tell DPL this and provide it with some more information about the field names in the database. You do this in the Database Specification dialog. For databases that are configured for revision tracking, both a Revision ID and Revision Date field must exist in each table storing project data.

- ⇒ Click Data | Database | Set Up.
- ⇒ Check the *Database configured for revision tracking* checkbox. The Revision ID and Revision Date field edit boxes are enabled.
- ⇒ Click Default field names to set the field names for the two revision fields.
- ⇒ Click OK.

The R&D Projects database was already configured for revision tracking, although you have not been using it as such up to this point. Therefore, you did not have to change the Source or change the database in any way. Normally you would probably set the *Database is configured for revision tracking* setting at the same time as initially specifying the data source. When you change the revision tracking setting, DPL also automatically loads the database schema information.

- ⇒ Save the Workspace.
- ⇒ Run a Decision Analysis.

Note that the results are slightly different. Previously when DPL did not know the database was configured for revision tracking, it was using the first data set it found for Model ID = 1, Project ID = 2. Now that DPL knows the database is configured for revision tracking, it uses the most recent revision which is slightly different.

## 22. DPL™ Developer API (Enterprise only)

### 22.1 Overview

---

DPL Enterprise includes an Application Programming Interface (API) which is an interface for controlling DPL from other programs. DPL's API can be used to automate repetitive tasks such as updating and running several models. It can also be used to leverage DPL's decision analysis engine in programs meant for use by persons not familiar with DPL or even decision analysis.

Although using the DPL API requires some programming ability, common tasks can be accomplished with only a few commands and do not require specialist software development expertise. If you have written Excel macros in Visual Basic for Applications (VBA), then you should have no trouble learning to use the DPL API.

Most uses of the API involve creating one or more template Workspace files as part of the development process. At runtime, the client application controlling DPL supplies specific data and runs analyses to produce results. These results can then be displayed to the user either by DPL or by the client application.

The DPL API uses Automation (also called OLE Automation) as the underlying technology for exposing capabilities to client programs. Automation makes it easy to control DPL from current versions of Microsoft Office. Future versions of the DPL API may employ other technologies.

The DPL API can be used with any language that supports Automation, including the .NET family of languages. Code examples in this chapter are written in VBA.

Several objects, properties and methods in the DPL API correspond to features that are only available in the Portfolio version of DPL. These are labelled "Portfolio only" in this section. For specific information on DPL Portfolio version features, see the DPL 9 Portfolio User Guide.

## 22.2 Controlling DPL™ from Visual Basic

---

In this section, you will create a Visual Basic program that opens a DPL Workspace file and runs a Decision Analysis. The tutorial that follows assumes the client is Visual Basic for Applications (VBA) in Microsoft Excel 2013 but the process is similar in any VB or VBA client environment.

### 22.2.1 Running a Decision Analysis

Before you can use DPL objects in a program, you need to register DPL as a server. You will need administrator privileges to register DPL on your machine. If you have registered previous versions of DPL on the same machine where you are about to register DPL 9, you will need to unregister those first.

If you need to unregister an earlier version of DPL, do the following.

- ⇒ Run a Command prompt as Administrator.
- ⇒ Change directories to where the version of DPL that you wish to unregister is installed. E.g., DPL8.exe is usually the installed in:  
C:\Program Files (x86)\Syncopation\DPL8.
- ⇒ Type "DPL8 /Unregister" at the command prompt (without the quotes).
- ⇒ Hit Enter.

DPL's splash screen will be visible for a second or so as DPL unregisters itself with the system.

To register DPL 9, do the following.

- ⇒ Run a Command prompt as Administrator.
- ⇒ Change directories to the DPL 9 installation directory, usually  
C:\Program Files\Syncopation\DPL9.
- ⇒ Type "DPL9 /Register" at the command prompt (without the quotes).
- ⇒ Hit Enter.

DPL's splash screen will be visible for a second or so as DPL registers itself with the system. You are now ready to begin using the DPL API.

You will start with a blank Excel workbook.

- ⇒ Start Microsoft Excel.
- ⇒ Click Developer | Code | Visual Basic.

⇒ Select Insert | Module.

⇒ Type the following code into the editor window.

```
Sub RunWildcat()  
    Dim oDPLApp As Object, oDPLWS As Object  
    Set oDPLApp = CreateObject("DPL.Application")  
    oDPLApp.Show = 1  
    Call oDPLApp.OpenWorkspace("C:\Wildcat.da")  
    Set oDPLWS = oDPLApp.Workspace  
    oDPLWS.RunDecisionAnalysis  
End Sub
```

You will need to change "C:\Wildcat.da" to the actual location of that example file on your computer (e.g., "C:\Program Files \Syncopation\DPL9\Examples\Wildcat.da").

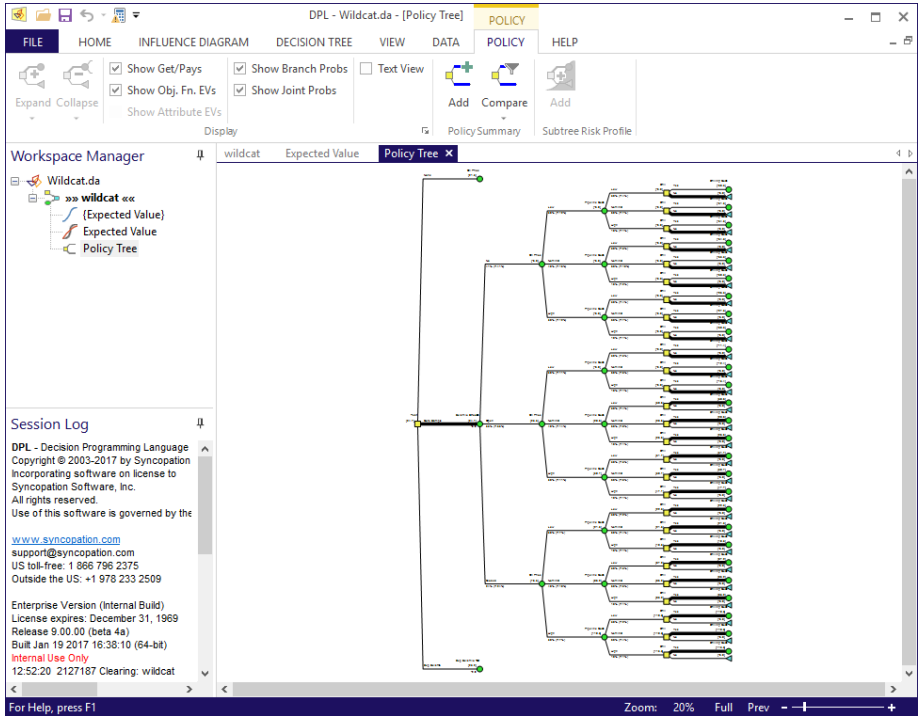
⇒ Start DPL.

⇒ Return to the Visual Basic Editor.

⇒ Make sure the cursor is in the Sub RunWildcat.

⇒ Press F5 to run your Sub.

See Figure 22-1 for the result of the run.



**Figure 22-1. Wildcat Policy Tree™**

Note: if you run your macro before starting DPL, the CreateObject function will create an invisible instance of DPL. You can make such an instance visible using the Show property of DPLApplication (in the example above, the line "oDPLApp.Show = 1" was included just in case).

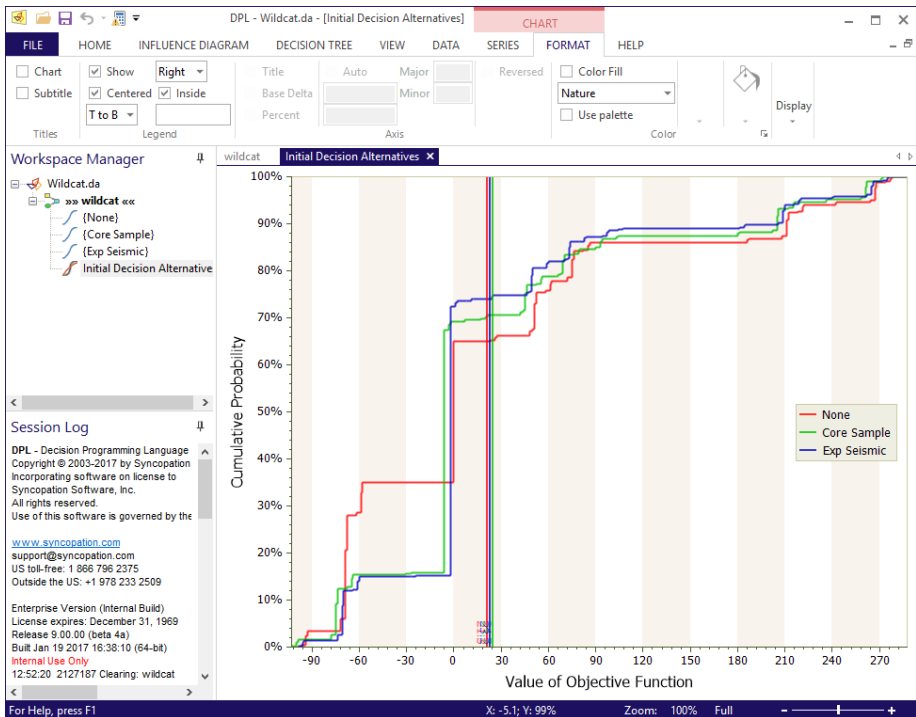
In the above, you ran a Decision Analysis without specifying any options. In the DPL API, run options are properties of the DPLWorkspace object (oDPLWS).

⇒ Insert the following two lines before the RunDecisionAnalysis line.

```
oDPLWS.PolicyTree = 0
oDPLWS.InitialDecisionAlternatives = 1
```

⇒ Press F5 to run the Sub again.

See Figure 22-2 for the result of the new run.



**Figure 22-2. Wildcat Risk Profile**

Lastly, you'll use the API to make a "what-if" change to the model to look at a non-optimal alternative of the Test decision. To do this you'll use branch control on the Test decision in the Decision Tree.

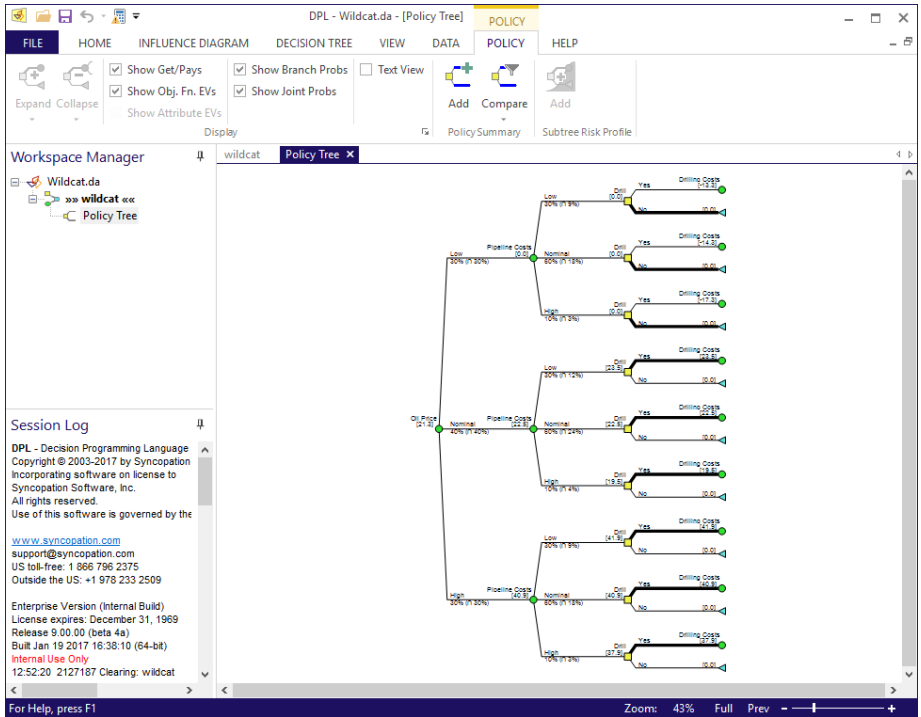
⇒ Add these lines to your Sub, replacing the two you previously added.

```
oDPLWS.PolicyTree = 1
Dim oDPLModel as Object
Set oDPLModel = oDPLWS.MainModel
Dim oDPLNode as Object
Set oDPLNode = oDPLModel.Nodes("Test")
Dim oDPLTreeNode as Object
Set oDPLTreeNode = oDPLNode.TreeNodes(1)
oDPLTreeNode.BranchControl = 0
oDPLModel.ClearMemory
```

⇒ Press F5 to run the Sub again.



See Figure 22-3 for the result of the branch control run.



**Figure 22-3. Wildcat Policy Tree™ with Test Controlled**

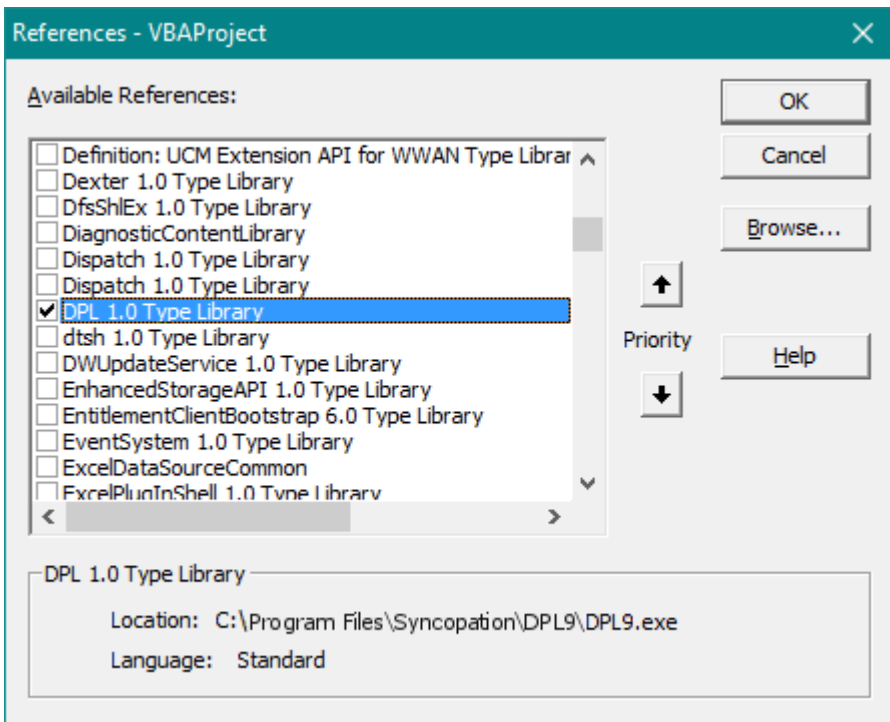
### 22.2.2 Adding a Type Library Reference

If you would like DPL's object types (DPLApplication, DPLWorkspace, etc.) to be known to your development environment, you can add a reference to the DPL API type library. Doing so has the benefit that your development environment can check syntax and provide lists of properties/methods. Adding a reference is not required, and the examples in this chapter do not assume that it has been done (in particular, they declare variables as type Object rather than as a specific DPL object type).

The type information is contained in the main DPL executable (DPL9.exe). To add a reference from Excel VBA, go to the Visual Basic Editor and follow these instructions:

- ⇒ Select Tools | References....
- ⇒ Click Browse...

- ⇒ Change Files of Type to be Executable Files.
- ⇒ Find your DPL executable. It is usually in C:\Program Files\Syncopation\DPL9\DPL9.exe.
- ⇒ Click Open.
- ⇒ Scroll down in the Available Reference list to find "DPL 1.0 Type Library"
- ⇒ Check the checkbox next to it if it isn't already.
- ⇒ Click OK.



**Figure 22-4. Type Library References**

With the type library reference, you can use DPL types in your code, as in the following example. Note that you still need to Dim the application as Object (not as DPLApplication).

```
Sub RunWildcatTypes()  
    Dim DPLApp As Object  
    Set DPLApp = CreateObject("DPL.Application")  
    DPLApp.OpenWorkspace ("C:\Wildcat.da")  
  
    Dim WS As DPLWorkspace  
    Set WS = DPLApp.Workspace  
  
    Dim Model As DPLModel  
    Set Model = WS.MainModel  
  
    Dim Node As DPLNode  
    Set Node = Model.Nodes("Test")  
  
    Dim TreeNode As DPLTreeNode  
    Set TreeNode = Node.TreeNodes(1)  
    TreeNode.BranchControl = 0  
  
    Model.ClearMemory  
    WS.RunDecisionAnalysis  
End Sub
```

Test the Type Library now.

- ⇒ Enter the new Sub into Module1.
- ⇒ Change the path from "C:\Wildcat.da" to where the file is on your computer.
- ⇒ Press F5 to run it.
- ⇒ Note: if you place your cursor next to the WS on the last line of the Sub and type ".", Excel provides you with a list of Methods and Properties of a DPLWorkspace.

## 22.3 API Objects and Types

---

DPL's API provides several objects which allow you to manipulate the application, its documents and their data. Figure 22-5 shows the hierarchical relationships of the DPL API objects.

```
DPLApplication (1 per running instance of DPL)
  →DPLWorkspace (1)
    →DPLModel (0 or more)
      →DPLNode (0 or more)
        →DPLTreeNode (0 or more)
      →DPLResult (0 or more)
```

**Figure 22-5. DPL API Objects**

### 22.3.1 The DPLApplication Object

The Application object represents a running instance of DPL. It is the starting point for any conversation with the DPL API.

The methods of the Application object correspond to activities that effect the whole application and not just individual windows (e.g., Models, Risk Profile Charts, etc). Most methods in the Application object correspond to commands in the File menu.

The following code obtains a reference to a DPLApplication object.

```
Dim oDPLApp As Object
Set oDPLApp = CreateObject("DPL.Application")
```

### 22.3.2 The DPLWorkspace Object

The DPLWorkspace object corresponds to a DPL Workspace (.DA) file. Only one Workspace file can be loaded at a time. DPL creates a blank Workspace when it starts. The currently loaded Workspace can be accessed using the Workspace property of the DPLApplication object.

```
Dim oDPLWS As Object
Set oDPLWS = oDPLApp.Workspace
```

### 22.3.3 DPLModel Objects

DPLModel objects correspond to components of a DPL Workspace that can be run or evaluated, including Models, Programs and Commands.

```
Dim oDPLModel as Object, oDPLModel2 as Object
Set oDPLModel = oDPLWS.MainModel
Set oDPLModel2 = oDPLWS.Models("Another Model")
```

A DPLModel object can be any of the types shown in Table 22-1 (see the Type property).

ID	Hex value (decimal value)	Description
ID_DOC_TYPE_IDDT	0x0001 (1)	Model (influence diagram and decision tree)
ID_DOC_TYPE_PROGRAM	0x0002 (2)	Program not run directly
ID_DOC_TYPE_DRIVER_PROGRAM	0x0004 (4)	Runnable program
ID_DOC_TYPE_COMMAND	0x0005 (5)	Command

**Table 22-1. DPLModel Types**

**22.3.4 DPLNode Objects**

DPLNode objects correspond to nodes in a DPL Influence Diagram. A DPLNode object is returned by the Nodes property (collection) of a DPLModel object.

```
Dim oDPLNode1 as Object
Set oDPLNode1 = oDPLModel.Nodes("Revenue")
```

**22.3.5 DPLTreeNode Objects**

DPLTreeNode objects correspond to nodes in a DPL Decision Tree. Each Decision Tree node is an instance of an Influence Diagram node of type Decision, Chance or Controlled. A DPLTreeNode object is returned by the TreeNodes property (collection) of a DPLNode object.

```
Dim oDPLTreeNode1 as Object
Set oDPLTreeNode1 = oDPLNode1.TreeNodes(1)
```

### 22.3.6 DPLResult Objects

DPLResult objects correspond to components of a DPL Workspace that are the result of analyses, such as Policy Trees, Risk Profile Charts, Rainbow Diagrams, etc.

```
Dim oDPLResult as Object
Set oDPLResult = oDPLWS.Results("Expected Value")
```

A DPLResult object can be any of the types shown in Table 22-2 (see the Type property).

ID	Hex value (decimal value)	Description
ID_DOC_TYPE_ENDPOINTS	0x0010 (16)	Endpoints
ID_DOC_TYPE_RISK_PROFILE	0x0020 (32)	Risk Profile chart
ID_DOC_TYPE_DECPOLICY	0x0021 (33)	Policy Tree
ID_DOC_TYPE_POLICYSUM	0x0022 (34)	Policy Summary
ID_DOC_TYPE_SENS	0x0023 (35)	Rainbow diagram (one-way)
ID_DOC_TYPE_TWOWAY_SENS	0x0024 (36)	Rainbow diagram (two-way)
ID_DOC_TYPE_VAL_COMP	0x0025 (37)	Value tornado diagram
ID_DOC_TYPE_NOM_COMP	0x0026 (38)	Base case tornado diagram
ID_DOC_TYPE_EVENT_COMP	0x0027 (39)	Event tornado diagram
ID_DOC_TYPE_VOIC	0x0028 (40)	Value of info/control chart
ID_DOC_TYPE_CORR	0x0029 (41)	Value correlations chart
ID_DOC_TYPE_OPTION_VALUE	0x002A (42)	Option value chart

**Table 22-2. DPLResult Types**

### 22.3.7 DPLPortfolioGroup Object (Portfolio only)

DPLPortfolioGroup objects correspond to groups in the DPL portfolio stored in the current workspace. A DPL portfolio always contains at least one group. DPLPortfolioGroups are accessed via the PortfolioGroup property of the DPLWorkspace object.

```
Dim oDPLGroup1 as Object
Set oDPLGroup1 = oDPLWS.PortfolioGroup("Group 1")
```

### 22.3.8 DPLPortfolioElement Object (Portfolio only)

DPLPortfolioElement objects correspond to elements in the DPL portfolio. Each group in the DPL portfolio contains at least one element. DPLPortfolioElements are accessed via the PortfolioElement property of a DPLPortfolioGroup object.

```
Dim oDPLElem1 as Object
Set oDPLElem1 = _
    oDPLGroup1.PortfolioElement("Element 1")
```

### 22.3.9 Parameter Data Types

The DPL API uses the five data types described below. Examples of how to declare them in VB and C/C++ are given after each description. Note that in VB one can also use Variants for any type. The Variant subtypes (VT\_\*) are shown in the comments.

**Object:** An OLE Automation IDispatch pointer.

```
Dim o as Object 'VB VT_DISPATCH
LPDISPATCH lpDispatch; // C/C++
```

**Double:** A double precision floating point number, which can take values in the range of 2.2250738585072014e-308 to 1.7976931348623158e+308.

```
Dim d as Double 'VB VT_R8
double d; // C/C++
```

**Long:** A 32-bit signed integer, which can take values in the range of -2147483648 to 2147483647.

```
Dim i as Long 'VB VT_I4
int i; // C/C++
```

**Bool:** A Long value interpreted to be boolean (zero for false, nonzero for true).

```
Dim b as Long 'VB VT_I4
BOOL b; // C/C++
```

**String:** An OLE Automation BSTR string.

```
Dim s as String    'VB VT_BSTR
  _bstr_t s;      // C/C++
```

**Variant:** A Variant is a variable that can take on one of many data types, and may be either a scalar or an array. In the DPL API, Variants are used to return arrays consisting of strings and doubles, such as the data in a tabular report displayed in a grid control.

```
Dim v as Variant  'VB
VARIANT v;       // C/C++
```

## 22.4 API Reference

---

This section documents the properties and methods of each type of object in the DPL API.

The syntax for properties is:

property\_name : type[DPL Type Library type].

The property declaration is followed by a parenthetic comment indicating whether the property is read-only or read/write.

The syntax for methods is:

method\_name(param1 : param1 type, param2 : param2 type, ...)

### 22.4.1 DPLApplication Properties

DisplayRunStatus : Bool (read/write)

Controls whether or not DPL displays the run status dialog when running an analysis. The default is false if DPL was started by Automation, true otherwise.

ErrorsToLog : Long (read/write)

Controls the display of error and warning messages that would normally be displayed in message boxes. Must be one of the following:

- 0 Display interactively with message boxes
- 1 Print in session log
- 2 Both



LastError : String (read-only)

A string describing the last error (or warning) from DPL, whether or not that error was displayed. In most cases, the string in LastError will be the same as the description property of the VB Err object.

```
On Error GoTo OnErr
... 'code that results in an error
OnErr:
MsgBox Err.Description
MsgBox oDPLApp.LastError 'same as above
```

LogFile : String (read/write)

The path of a text file for the session log. Log messages will be printed both to the Session Log window and the specified file.

The file will be opened immediately and will stay open until the DPL Application instance is released. Any existing file of the same name will be overwritten. To close the file explicitly, set LogFile to "" (an empty string).

Options(Name : String) : Long (read/write)

Reserved for future use. There are no DPLApplication named options at this time.

PostRunExcelMacro : String (read/write)

Contains the name of an Excel macro that DPL should run at the end of an analysis. Using this property with the RunAsynchronous property, you can have your client program start a DPL analysis and return to an idle state, then you can have DPL pass control back to the client after the analysis. With this technique, you can run a DPL analysis from a VBA macro in an Excel workbook which is itself linked to DPL and will be used in the analysis. See the DrugDevelopmentRunButton example files.

ReleaseNumber : Long (read-only)

The DPL release number as an integer. For example, DPL release 8.01.02 would return the integer 80102.

RunAsynchronous : Bool (read/write)

When this property is true, calls to DPL "Run" methods will return immediately. When false (the default), they will return when the analysis is

complete. You can use this method to "kick off" a run in a situation where the results will be examined interactively by the user, or you can use it in conjunction with the PostRunExcelMacro to take back control after the analysis (if the client is Excel).

#### Show : Long (read/write)

The current state of the DPL main window. Set this property to maximize, minimize, show or hide DPL's main window. The appropriate values are those of the Windows API ShowWindow command, a few of which are listed below. Consult the Windows API documentation for details.

```
SW_HIDE      0
SW_SHOWNORMAL 1
SW_SHOWMINIMIZED 2
SW_SHOWMAXIMIZED 3
```

If DPL is started by Automation (e.g., by using CreateObject("DPL.Application") in VBA) when no instance of DPL is running, DPL's main window will initially be hidden.

#### Workspace : Object[DPLWorkspace] (read-only)

This method returns a DPLWorkspace object corresponding to the currently loaded DPL Workspace file. If no Workspace is loaded, DPL will throw an exception.

### **22.4.2 DPLApplication Methods**

#### CloseWorkspace()

Closes the current Workspace file without prompting.

#### Exit(Code : Long)

Terminates the DPL Application returning Code to the system (e.g., use Exit(0) for "normal" termination). This value will be available to the program or script that started DPL. Note that the call is asynchronous, so DPL may still be running when the method returns. DPL returns 1 in the case of an activation error or license problem, so use values greater than 1 if you need custom return codes.

NewWorkspace()

Creates a new, blank Workspace file. If a Workspace was previously loaded, it is closed.

OpenWorkspace(Path : String)

Opens the Workspace file indicated by Path.

RegisterROT()

Register the DPL Application in the Windows Running Object Table (ROT).

RevokeROT()

Revoke the DPL Application's registration from the Windows Running Object Table (ROT), if the application was previous registered by a call to the RegisterROT() method.

SaveWorkspace()

Saves the Workspace file using the current file name and path. The Workspace file must be named otherwise DPL will throw an exception.

SaveWorkspaceAs(Path : String)

Saves the Workspace file to Path.

Warning(message : String)

Causes DPL to display a message box containing message.

WriteToLog(message : String), WriteLnToLog(message : String)

Writes message to the session log. WriteLnToLog appends a new line, so that the next log message will start in the first column of the following line. To send a multi-line message to the session log, include carriage return / linefeed pairs ("`\r\n`" in C/C++, `Chr$(13)+Chr$(10)` in VB).

### 22.4.3 DPLWorkspace Properties

#### Distribution : Long (read/write)

The type of risk profile(s) DPL should generate in the next run. Must be one of the following:

- 0 No distribution
- 1 Objective function
- 2 Objective function and all attributes
- 3 All attributes
- 4 One attribute

If the value is 4, DistributionIndex should be set to the index of the desired attribute.

#### DistributionGraphFormat : Long (read/write)

Controls how the risk profile generated in a Monte Carlo simulation run is displayed.

- 0 Cumulative
- 1 Histogram

This has no effect on risk profiles generated in Decision Analysis runs, which are always displayed in Cumulative form.

#### DistributionIndex : Long (read/write)

The index of the attribute for which a risk profile should be generated in the next run. Ignored unless Distribution is set to 4.

#### DistributionIntervals : Long (read/write)

The number of distribution intervals DPL should use for the risk profiles generated in the next run. Must be between 0 and 10000. Default is 500.

For Monte Carlo simulation runs, this property can be set to -1, indicating that DPL should store all samples and do no aggregation.

#### EvaluationMethod : Long (read/write)

The evaluation method DPL should use in the next run. Must be one of the following:

- 0 Fast sequence evaluation
- 1 Full enumeration
- 2 Discrete Tree Simulation

FatPolicy : Bool (read/write)

True if DPL should produce a Policy Tree which includes rolled-back expected values for all attributes in the next run. Ignored if there is only one attribute.

InitialDecisionAlternatives : Bool (read/write)

True if DPL should generate a risk profile chart showing initial decision alternatives in the next run. Ignored if the tree doesn't start with a decision or if it starts with a decision having more than eight alternatives.

MainModel : Object[DPLModel] (read-only)

An object reference to the main model of the currently loaded Workspace file. Returns Null if no Workspace is loaded or the current Workspace has no main model.

```
Dim oDPLModel As Object
Set oDPLModel = oDPLApp.MainModel
```

Note that this property is read-only. To change the main model, use the Models property to obtain an object reference to the model you want to make main, then call its MakeMain property.

Models(Name : String) : Object[DPLModel] (read-only)

An object reference to the model with title Name. Returns Null if there is no model (DPL Model or Program window) with title Name in the Workspace. The example that follows displays the type of a model named "Bob".

```
Dim oDPLModel As Object
Set oDPLModel = oDPLApp.Model("Bob")
If oDPLModel Is Nothing Then
    MsgBox "Sorry, no Bob in the Workspace"
Else
    MsgBox "Bob is type " + Str(oDPLModel.Type)
End If
```

MonteSeed : Long (read/write)

This property can be used to save/restore the seed for the pseudo-random number generator used for Monte Carlo runs, such as when successive runs with the same "random draw" are desired.

Options(Name : String) : Long (read/write)

Used to set various options not having their own properties. Currently the following options are recognized:

- DecimalPlaces
- OutputScaling
- ZeroEquivalent
- ProbsAsPercents (Bool)
- DecimalPlacesProbs

These control display as per the similarly named options in File | Options | Outputs.

PolicyLevels : Long (read/write)

The number of policy levels DPL should include in the Policy Tree in the next run. Default is all levels. A value of 1 is treated as 2, since the Policy Tree requires two or more levels.

PolicySummary : Bool (read/write)

True if DPL should generate a Policy Summary in the next run.

PolicyTree : Bool (read/write)

True if DPL should generate a Policy Tree in the next run.

PortfolioAttributes : Variant (read-only) (Portfolio only)

Returns the portfolio attributes as a Variant array. In the returned array each attribute is a string, and they are ordered as they appear in the Portfolio Attributes dialog.

PortfolioGroup(Name : String) : Object[DPLPortfolioGroup] (read-only) (Portfolio only)

Returns an object reference to the DPLPortfolioGroup of the given name.

PortfolioOptions(Name : String) : Long (read/write) (Portfolio only)

This property is used to get or set various portfolio analysis / portfolio data report options. These options generally correspond to the checkboxes in the Portfolio Analysis Options / Data Report Options dialogs. Currently the following options are recognized:

Options for portfolio analysis runs:

- PortfolioDistributionIndex
- IndividualDistributions
- IndividualDistributionsIndex
- PortfolioTimeSeries
- IndividualTimeSeries
- TimeSeriesInitialPeriod
- TimeSeriesFromIndex
- TimeSeriesToIndex
- RiskReward
- PortfolioTornado
- ProjectValueRange
- ExpectedPotentialValue
- ProductivityRatio
- InitialDecisionProductivityRatio
- ExtractProjectDataOnRun

Options for portfolio data reports:

- CombineAllGroups
- GroupProbabilities
- GroupEventValues
- GroupScalarValues
- GroupArrayValues
- AggregateProjectSets
- DefaultStatesOnly
- DisplayProjectCategories
- DisplayProjectIDs
- DisplayColumnTotals
- ColumnsAreYears
- FirstYear
- MultipleRowsPerProject

RainbowLow : Double (read/write)

RainbowHigh : Double (read/write)

RainbowStepSize : Double (read/write)

RainbowValue : String (read/write)

Set these properties before running a rainbow diagram. RainbowValue is the name of the value for sensitivity, RainbowLow and RainbowHigh are the ends of the range, and RainbowStepSize is the value increment.

RecordEndpoints : Bool (read/write)

True if DPL should record endpoints in the next run. Requires that the evaluation method be full enumeration otherwise it is ignored.

Results(Name : String) : Object[DPLResult] (read-only)

An object reference to the result with title Name. Returns Null if no Workspace is loaded or there is no result with the title Name in the Workspace. See the Model property for an example.

SamplesInitial : Long (read/write)

The initial number of samples for a Monte Carlo simulation run or a Decision Analysis run using discrete tree simulation evaluation method.

SamplesRestart : Long (read/write)

The minimum number of samples following each decision in a Monte Carlo simulation run or a Decision Analysis run using discrete tree simulation evaluation method. Has no effect if the model does not contain decisions.

TimeSeriesFromAtt : Long (read/write)

The index of the first attribute to be displayed in the next Time Series Percentiles run. Note that the first attribute has index 1 (not 0).

TimeSeriesInitialPeriod : Long (read/write)

The number displayed for the first attribute on the x-axis in the Time Series Percentiles graph. For example, if each attribute represents cash flow in a given year, and the first attribute corresponds to 2010, you would set this property to 2010.



TimeSeriesNumPeriods : Long (read/write)

The number of time periods (attributes) to be displayed in the next Time Series Percentiles run. For example, if there are five attributes corresponding to years 2014-2018, you would use:

```
oDPLWS.TimeSeriesFromAtt = 1
oDPLWS.TimeSeriesNumPeriods = 5
oDPLWS.TimeSeriesInitialPeriod = 2014
oDPLWS.RunTimeSeries
```

Title : String (read-only)

The file name of the Workspace (e.g., "Wildcat.da"). This property is read-only; use the DPLApplication method SaveWorkspaceAs() to rename the Workspace.

ValueCorrelations : Long (read/write)

True if DPL should generate a Value Correlations chart in the next Decision Analysis or Monte Carlo run. For a Decision Analysis, Value Correlations require that the evaluation method be full enumeration.

ValueOfInfoControl : Bool (read/write)

True if DPL should generate an Expected Value of Perfect Information/Control chart in the next Decision Analysis run. Ignored if the number of PolicyLevels is not the maximum for the Decision Tree.

**22.4.4 DPLWorkspace Methods**CreateProgram(Data : String, Name : String, Driver : Bool)

Creates a DPL Program in the current Workspace with the DPL code contained in Data and the title Name. Driver should be True if the program can be run directly, or False if the program is intended as an include file to be referenced by another program or a Model. The example that follows creates and runs a simple program.

```
Dim Data As String
Data = "chance c.{lo,hi} = {0.5} = 1,2;" _
      & "sequence: " _
      & "gamble on c and get c"
oDPLApp.CreateProgram(Data, "C", 1)
Dim oDPLModel as Object
```

```

Set oDPLModel = oDPLApp.Model("C")
oDPLModel.MakeMain
oDPLWS.RunDecisionAnalysis

```

This method is convenient for small programs, such as those supplying data used to initialize nodes in a template Influence Diagram. However for large programs, writing a .dpl file to disk and calling ImportProgram is more efficient.

DBExportData(Table : String, Fields : String, Values : String, Append : Bool, ForceUpdate : Bool, SequenceName : String) : Bool (Portfolio only)

This method allows the controlling program to export data directly to the database without a portfolio analysis run. The parameters correspond to equivalent sections of an SQL UPDATE statement. A nonzero return value indicates success.

DBSelectData(Table : String, Fields : String, Where : String) : Variant (Portfolio only)

This method allows the controlling program to query the database directly. The parameters are as in an SQL SELECT statement. The results of the query are returned as a Variant array.

DBDeleteData(Table : String, Where : String) : Bool (Portfolio only)

This method allows the controlling program to delete records from the database directly. The parameters are as in an SQL DELETE statement. A nonzero return value indicates success.

DeleteProjectData() (Portfolio only)

Deletes any extracted project data from the workspace.

ExtractProjectData(ProjectIDs : String) : Long (Portfolio only)

Extract project data for all of the projects whos ID's are listed in the ProjectIDs parameter. ProjectIDs is a comma separated list of integers. The return value is the number of projects for which data was successfully extracted.

ImportProgram(Path : String, Name : String, Driver : Bool)

Imports the program (.dpl source file) specified by Path into the current Workspace and gives it the title Name. Driver should be true if the program can be run directly, or false if the program is intended as an include file to be referenced by another Program or a Model.

ResetMonteSeed()

Resets the seed of the pseudo-random number generator used for Monte Carlo runs. The seed is generated from low order bits of the system time.

RunDataReport() (Portfolio only)

Runs a portfolio data report using the current settings.

RunDecisionAnalysis()

Runs a Decision Analysis. To specify run options, use the following properties: EvaluationMethod, Distribution, DistributionIndex, InitialDecisionAlternatives, ValueCorrelations, PolicyTree, PolicyLevels, FatPolicy, PolicySummary, ValueOfInfoControl. These properties have the same meanings and defaults as in the Home | Run group.

The example that follows runs a Decision Analysis generating a Risk Profile but no Policy Tree.

```
oDPLWS.Distribution = 1
oDPLWS.PolicyTree = 0
oDPLWS.RunDecisionAnalysis
```

RunMonteCarloSimulation()

Runs a Monte Carlo simulation. To specify run options, use the following properties: SamplesInitial, SamplesRestart, Distribution, InitialDecisionAlternatives, DistributionIntervals, DistributionGraphFormat, ValueCorrelations, PolicySummary, PolicyLevels, PolicyTree. These properties have the same meanings and defaults as in the Home | Run group.

RunOptionValue()

Runs an Option Value analysis. DPL will use the default decision alternatives as specified in Node Definition General. Or you may specify defaults by using the DPLNode DefaultState property before the run.

RunPortfolioAnalysis() (Portfolio only)

Runs a portfolio analysis using the current settings.

RunRainbow()

Runs a Rainbow Diagram sensitivity analysis. The properties RainbowLow, RainbowHigh, RainbowStepSize and RainbowValue must be set before running a rainbow.

RunTimeSeries()

Runs a Time Series analysis. See the TimeSeriesNumPeriods property for an example.

RunTornado(type : Long, option : Long)

Runs a tornado diagram. The type of tornado is determined by the type and option parameters. The valid combinations are shown in Table 22-3.

type	opt	Tornado Type
37	0	Value
38	0	Base Case
38	1	Initial Decision Alternatives
39	0	Event (Deterministic)
39	1	Event (Probabilistic)

**Table 22-3. Tornado Types**

Note that 37, 38 and 39 correspond to the DPLResult Type values of the charts produced by the run. The example that follows runs a Base Case Tornado.

```
oDPLWS.RunTornado(38, 0)
```

SetProjectIds(ProjectIDs : String) : Bool (Portfolio only)

This method can be used to set all the project IDs for all of the elements in the portfolio. The ProjectIDs string consists of a comma separated list of integers for each portfolio element, with the lists for different elements

separated by vertical bars, for example "1,3|2,4|5,6" for a portfolio with three elements, each with two projects.

UpdateAllResults()

This method refreshes all output (DPLResult) windows. This may be necessary after changing options that effect display.

```
oDPLWS.Options("DecimalPlaces") = 1
oDPLWS.UpdateAllResults
```

**22.4.5 DPLModel Properties**

EvalResultEV : Double (read-only)

The expected value (EV) results of the most recent Decision Analysis run.

EvalResultCE : Double (read-only)

The certain equivalent (CE) results of the most recent Decision Analysis run.

EvalResultsValid : Long (read-only)

Indicates whether evaluation results (e.g., EvalResultEV) exist. The value will be 0-2 with the following meanings.

- 0 Results not valid
- 1 Results valid from a Decision Analysis run
- 2 Results valid from a Monte Carlo simulation run

IncludeEnd : String (read/write)

IncludeStart : String (read/write)

Used to set the name of the includes (DPL Programs) referenced by the Model. IncludeStart corresponds to the "Data Definitions Program" in the Model Links dialog; IncludeEnd corresponds to "Calculation Program".

Nodes(Name : String) : Object[DPLNode] (read-only)

Used to obtain an object reference to a node (DPLNode) in the model with title or variable name Name. See the DPLNode property VariableName for an explanation of node titles and variable names. Returns Null if no such node exists in the DPLModel.

```
Dim oModel As Object
Dim oNode As Object
Set oModel = WS.MainModel
Set oNode = oModel.Nodes("Test")
```

Options(Name : String) : Long (read/write)

Reserved for future use. There are no DPLModel named options at this time.

Title : String (read/write)

The title of this DPLModel object, as displayed in the Workspace Manager.

Type : Long (read-only)

The type of this DPLModel object. See Table 1 for a list of DPLModel types.

**22.4.6 DPLModel Methods**Delete()

Deletes the associated document from the Workspace.

MakeMain()

Makes this document the main model, that is, the document that will be run in the next analysis. This DPLModel must be of type Model (Influence Diagram / Decision Tree) or a driver program. If the model is not of one of these types, DPL throws an exception.

ClearMemory()

Deletes compile structures and removes any volatile (unrenamed) outputs associated with this model. Same as Home | Run | Clear Mem.

If you run an analysis and then modify the model, you will need to issue a ClearMemory command to force the model to be recompiled.

### 22.4.7 DPLNode Properties

#### DefaultState : Long (read/write)

The default state of this node as set in the General tab of the Node Definition dialog and as used by Base Case tornadoes and Option Value charts. The first state has index 0. The value -1 indicates there is no default.

```
Dim oDPLNode as Object
Set oDPLNode = oDPLModel.Nodes("Drill")
oDPLNode.DefaultState = 1 'No
```

#### Name : String (read/write)

The title or name of the node. Note that this string will include carriage return/linefeed pairs if the node name is on more than one line. If there are instances of this node in the Decision Tree, their names will be changed as well. A node name can contain characters not valid in a DPL variable name, such as punctuation and whitespace. Use the VariableName property to get the name as sanitized for use in expressions.

#### Probabilities(Index : Long) : String (read/write)

The probabilities of the node. The individual probabilities are indexed as they appear in the Data tab (or Probabilities tab if separately conditioned) of the Node Definition dialog, with the topmost branch being index 0. The probabilities are stored as strings even if they are constant numbers.

#### SensHigh : Double (read/write)

See SensLow.

SensLow : Double (read/write)

SensLow and SensHigh are the Low and High values used for each bar in Value tornado diagrams. They can only be set for unconditioned, constant value nodes.

```
Dim oDPLNode1 as Object, oDPLNode2 as Object
Set oDPLNode1 = oDPLModel.Nodes("Sales")
Set oDPLNode2 = oDPLModel.Nodes("Costs")
oDPLNode1.SensLow = 1.3
oDPLNode1.SensHigh = 1.9
oDPLNode2.SensLow = 0.7
oDPLNode2.SensHigh = 1.1
oDPLWS.RunTornado(37, 0)
```

TreeNode(Index : Long) : Object[DPLTreeNode]

Used to obtain an object reference to a Decision Tree node (DPLTreeNode) in the model which is an instance of this Influence Diagram node (DPLNode). If there is only one instance of this node in the Decision Tree, it will have index 1. If there are several instances, the instance number is shown in the title bar of the Node Definition dialog when you double-click on a node in the Decision Tree.

Type : Long (read-only)

The type of this node. The type is one of the following:

- 0 Decision
- 1 Chance
- 2 Controlled
- 3 Value

Values(Index : Long) : String (read/write)

The values of the node. The individual values are indexed as they appear in the Data tab (or Values tab if separately conditioned) of the Node Definition dialog, with the topmost branch being index 0. The values are stored as strings even if they are constant numbers.

```
Dim oDPLNode1 as Object
Set oDPLNode1 = oDPLModel.Nodes("Costs")
'Costs is a three-state chance node
oDPLNode1.Values(0) = "10"
oDPLNode1.Values(1) = "12"
oDPLNode1.Values(2) = "15"
```



VariableName : String (read-only)

The variable name of the node for use in expressions. If the node name is a valid identifier (that is, it begins with a letter and consists of letters, numbers and underscores) then the variable name is simply the node name. If not, the variable name is the node name with invalid characters replaced with underscores ('\_'). For example, if the node name is "R&D Costs", the variable name is "R\_D\_Costs".

**22.4.8 DPLTreeNode Properties**BranchBlock : String (read/write)

Used to block (temporarily remove from consideration) certain branches of a decision node. The format of the string is a comma separated list of ones and zeroes, one for each decision alternative. A zero indicates that that alternative is blocked.

```
Dim oDPLNode1 as Object
Dim oDPLTreeNode1 as Object
Set oDPLNode1 = oDPLModel.Nodes("Invest")
Set oDPLTreeNode1 = oDPLNode1.TreeNodes(1)
'Block the second of three alternatives
oDPLTreeNode1.BranchBlock = "1,0,1"
```

Set BranchBlock to an empty string ("") to unblock all branches.

Note that changing the BranchBlock state of a tree node will overwrite the BranchControl state and vice versa. If you use this method on a TreeNode that is not a decision node, DPL throws an exception.

BranchControl : Long (read/write)

Used to temporarily control a decision or chance node to a given state. Zero corresponds to the first state. The value -1 indicates that the tree node is not controlled.

```
'Control the node to its first state
oDPLTreeNode1.BranchControl = 0
```

Note that changing the BranchControl state of a tree node will overwrite the BranchBlock state and vice versa. Set BranchBlock to -1 to clear control.

Name : String (read/write)

The name of the tree node. Note that this string will include carriage return/linefeed pairs if the name is on more than one line. Changing the name of a tree node does not change the name of the influence diagram node of which it is an instance.

Type : Long (read-only)

The type of this tree node. The types are the same as those of influence diagram nodes (see Type in DPLNode). Only decision and chance nodes appear in the Decision Tree.

**22.4.9 DPLResult Properties**Data(Sheet : Long) : Variant (read-only)

If this DPLResult object corresponds to a result window in the form of a grid (for example, a set of endpoints, type ID\_DOC\_TYPE\_ENDPOINTS), this property returns all the data on a given sheet of the report as a Variant array.

DisplaySize(Index : Long) : Long (read/write)

This property represents the desired image size of images (pictures of the result window) returned by the CopyPicture() method. Index 0 is the size in the x direction, and index 1 is the size in the y direction.

NumSheets() : Long (read-only)

If this DPLResult object corresponds to a grid, this property returns the number of sheets in the grid.

NumRows(Sheet : Long) : Long (read-only)NumFixedRows(Sheet : Long) : Long (read-only)NumColumns(Sheet : Long) : Long (read-only)NumFixedColumns(Sheet : Long) : Long (read-only)

If this DPLResult object corresponds to a grid, these properties report the number of rows and columns, regular and fixed, of the specified sheet. Sheet indices range from 0 to NumSheets() - 1. Regular rows and columns are the ones shown with a white background color which scroll. Fixed rows

and columns are the gray background areas at the top and left of the grid used to display headings, project names, etc.

Options(Name : String) : Long (read/write)

Reserved for future use. There are no DPLResult named options at this time.

Title : String (read/write)

The title of this DPLResult object, excluding the type prefix. For example, a Risk Profile Chart might have the title "Expected Value".

Type : Long (read-only)

The type of this DPLResult object. See Table 22-2 for a list of DPLResult types.

#### 22.4.10 DPLResult Methods

CopyPicture(Long : format)

Copies a picture of the associated window to the Windows clipboard in both bitmap and metafile format. The format parameter is reserved for future functionality. You must send a format but it is ignored.

Delete()

Deletes the associated document from the Workspace.

DisplayCategories(CatTypes : String, Cats : String, Show : Bool) (Portfolio only)

This method can be used to control which projects are displayed in a report or graphical result window, using the categories defined for the portfolio. The CatTypes parameter is a comma separated list of category types, the Cat parameter is a comma separated list of category codes, and the Show parameter is a boolean flag indicating if the projects matching the category specification should be shown (1) or hidden (0).

DisplayProjects(ProjectstoShow : String, Show : Bool) (Portfolio only)

This method can be used to control which projects are displayed in a report or graphical result window. The ProjectstoShow string is a comma separated list of project ID's (e.g., "1, 3, 5"), and the Show parameter is 1 if the listed projects should be shown, 0 if they should be hidden.

Export(Path : String)

Exports the associated document to the comma separated value (CSV) or XML file in Path. The format is determined by the file extension of Path, which must be either ".CSV" or ".XML".

ReduceDist(Index : Long, Name : String, NumStates : Long, Format : Long)

Reduces a distribution to a chance node. This method is only valid for Risk Profile charts. Index indicates which of the risk profile datasets displayed in the chart should be reduced, the first one being index 0.

The node will be called Name and will have NumStates outcomes. The output is determined by the Format parameter:

- 1 DPL session log
- 2 Windows clipboard
- 3 Both

SavePicture(Path : String)

Save a picture of the result window to a file in the location given by the Path parameter. The picture can be saved either as a Windows bitmap or a Windows metafile, as indicated by the file extension, which must be .BMP or .EMF.

ShowDistStatistics()

Causes DPL to print the distribution statistics to the session log.

ShowLegend(Show : Bool)

Turns the display of the chart legend on (1) or off (0).

UpdateReport(ExportData : Bool, ForceUpdate : Bool) (Portfolio only)

Updates the report, rerunning the portfolio if necessary. If ExportData is 1 (true), results will be exported to the database (if the portfolio workspace is set up for export). If ForceUpdate is 1 (true), the portfolio will be reran even if project data has not changed.

**22.4.11 DLPPortfolioGroup Properties (Portfolio only)**Name : String (read-write) (Portfolio only)

The name of the portfolio group.

ModelID : Long (read-write) (Portfolio only)

The model ID for this group.

PortfolioElement(Name : String) : Object[DLPPortfolioElement] (read-only) (Portfolio only)

Returns an object reference to the DLPPortfolioGroup of the given name.

**22.4.12 DLPPortfolioElement Properties (Portfolio only)**Name : String (read-write) (Portfolio only)

The name of the portfolio element.

ProjectIDs : String (read-write) (Portfolio only)

The project IDs for this element as a comma separated list of integers, e.g., "1,3,5".

## 23. DPL™ User Function Libraries (Enterprise only)

### 23.1 Overview

---

One way to extend the functionality of DPL is to create a Windows DLL with functions to be called from your DPL models. A Windows DLL that is designed to be called by DPL is called a DPL user function library, or just a DPL DLL. A DPL DLL can serve as a way to interface DPL with other programs, such as software packages performing detailed economic calculations for specific industry verticals. Calling a function in a DPL DLL involves very little computational overhead, so a DLL can be attractive in situations where runtime performance is critical.

Implementing a DPL DLL requires technical expertise and significant effort, so before starting such a project you should consider whether more high-level mechanisms such as running Excel macros, database initialization links or the DPL Developer API would better meet your needs.

### 23.2 Technical Considerations

---

A DPL user function library must have a file extension of .DLL. Each library may contain one or more user functions in addition to the standard functions required by the Windows DLL mechanism. There is no limit to the number of libraries that may be in use during a DPL session. A library is loaded the first time it is encountered during the compilation of a DPL model, program or command procedure. A library is released by Windows only when all DPL sessions that have referenced the library are terminated. At the time a library is loaded, it must reside in the current directory or in a directory included in the PATH environment variable.

Though a library can behave as a full Windows application, it is recommended that a library constrain itself to providing computational functions suitable for execution during a DPL analysis.

User functions are divided into two categories: functions called implicitly by DPL, and functions called explicitly by DPL programs and command procedures. Implicitly called functions are optional. When a library is

loaded, DPL checks for the presence of certain predefined function entry points. At various times during a DPL session, these entry points are called, such as when an analysis is starting or ending. Explicitly called functions are referenced by expressions in DPL models, which are then executed when the expressions are evaluated.

User functions may be written in any language that complies with the requirements of a native Windows DLL. The examples used in this specification (including the sample code) have been prepared for use with the Microsoft Visual Studio C++ compiler.

All functions receive at least one parameter which is the number of additional parameters passed to the function. This will allow for future expansion of the parameter list for each function. A function can check that the number of parameters received are correct and terminate via the error call-back function if it is not (parameter mismatches will generally result in a crash).

User functions may call certain DPL functions as part of their execution. When a library is loaded, it is passed a list of function entry points in DPL. These functions provide access to DPL features, such as writing to the DPL Log or accumulating distribution data.

All definitions required to interface with DPL are provided in an include file called `dpuserf.h`. This file should be included in the DLL source after `windows.h` and any required C language include files. A Zip file containing the sample code and these include files can be obtained from Syncopation support.

## 23.3 Implicit Functions

---

### 23.3.1 Load Library

This function is called the first time a DPL session compiles a program or command procedure that references a DLL. Note that each DPL session that references a DLL will call the load function. If a DLL cannot operate with more than one session at a time (e.g., because of its use of static storage), it must reject the load attempt by calling the DPL error function.

This function is passed a list of entry points within DPL that may be used to perform various services during subsequent library function calls. A function definition macro for the load function is provided in the `dpuserf.h` include file. Another include file, `load.h`, is also provided for validating and saving the entry point list (see the sample code).

Typical functions performed by the load function include:

- Saving the list of DPL function entry points
- Writing a sign-on/copyright message to the DPL Log
- Verifying that the library is not already active
- Opening files that will be used during subsequent processing
- Allocating global memory and other Windows objects
- Creating a window to provide a custom interface to the user
- Controlling the use of fast sequence evaluation during subsequent analyses

Name:

load

Parameters:

1. Integer number of parameters (= 3)
2. Pointer to an array of pointers to functions
3. Integer number of elements in the array of parameter 2
4. Integer DPL version number (e.g., 0x00090000 => Release 9.0)

Returns:

- 0 : Fast sequence evaluation is not allowed in this library
- 1 : Fast sequence evaluation is allowed

Fast sequence evaluation (the fastest exact evaluation method) should not be allowed if the library requires that all Decision Tree paths be executed during a decision analysis; e.g., if the library matches each call to data obtained from a disk file. This function should not return if the load is to be aborted (call the error function instead as defined following sub-sections).

### 23.3.2 Unload Library

This function is called when a DPL session that previously called the library's load function is terminated. The library is not released until all DPL sessions that have referenced the library are terminated. Libraries that need to perform termination processing before being released by Windows should maintain a count of the active sessions connected to the library and perform final processing when the count reaches zero.

Typical functions performed by the unload function include:

- Closing open files
- Deallocating global memory and other Windows objects
- Destroying any windows created by the library



Name:

unload

Parameters:

1. Integer number of parameters (= 0)

Returns:

(None)

**23.3.3 Begin Analysis**

This function is called prior to beginning an analysis of a compiled DPL program. Analyses result from executing any of the Run menu functions. Performing a tornado diagram or rainbow diagram will result in multiple analyses.

The function should prepare the library for subsequent function calls during an analysis. Typical functions performed by the begin analysis function include:

- Initializing variables used during an analysis
- Opening or repositioning files for endpoint input or output
- Allocating temporary memory and other Windows objects

Name:

begin\_analysis

Parameters:

1. Integer number of parameters (= 0)

Returns:

(None)

**23.3.4 End Analysis**

This function is called after an analysis has completed.

Typical functions performed by the end analysis function include:

- Closing open files
- Deallocating temporary memory and other Windows objects

**Name:**

end\_analysis

**Parameters:**

1. Integer number of parameters (= 0)

**Returns:**

(None)

**23.3.5 Reset (error recovery)**

This function is called during DPL error processing. The cause of the error may or may not relate directly to the library and may or may not have occurred during an analysis. The library should perform whatever recovery processing is required to guarantee that it is prepared to receive subsequent function calls.

Typical functions performed by the reset function include:

- Calling the end\_analysis function, if appropriate
- Deleting output files

**Name:**

reset

**Parameters:**

1. Integer number of parameters (= 0)

**Returns:**

(None)

**23.3.6 Window Functions**

This function is called whenever DPL needs to disable all DPL windows or enable all DPL windows. This function must be implemented by libraries that create their own windows.

**Name:**

window\_functions

**Parameters:**

1. Integer number of parameters (= 2)
2. Integer indicating the required window function
3. Window handle

**Returns:**

An integer (BOOL) indicating whether the window handle parameter matches a window owned by the library

Consider the follow code fragment:

```
#include "dpluserf.h"
BOOL CALLBACK WINDOW_FUNCTIONS( int function, HWND hWnd )
{
    switch( function ) {
    case WF_ENABLE:
        if( hLibWnd )
            EnableWindow( hLibWnd, TRUE );
        break;
    case WF_DISABLE:
        if( hLibWnd )
            EnableWindow( hLibWnd, FALSE );
        break;
    case WF_ROTATE:
        return( hWnd == hLibWnd );
    }
    return( YES );
}
```

Here, it is assumed that the library has created a window and saved its handle in the variable, `hLibWnd`. For the enable and disable functions, the window is enabled or disabled if the window handle is valid. Note that the window handle parameter is ignored and the function always returns 1, defined as YES. For the rotate function, the window handle parameter is compared with the library window handle and the result of the comparison is returned.

## 23.4 Explicit Functions

---

Explicit functions are the functions you write and call from your DPL models.

Explicit functions behave in much the same way as DPL built-in functions: they are coded in DPL expressions, receive numbers and strings as parameters, and return a double precision floating point number.

An explicit function is referenced in a DPL expression by coding the name of the library, optionally followed by a period and the function entry point

name, then a parenthesized argument list. The library name may not include a path specification. At the time the library is loaded, it must exist either in the current directory or in a directory in the current path.

If a period and function entry point name do not follow the library name, the default entry name, `calculate`, will be used. The argument list consists of from 0 to 127 expressions or strings separated by commas. The following are all valid explicit function references:

```
mylib()
mylib.calculate( 1, x+y )
mylib.filename( "c:\\dpl\\test.dat" )
```

All functions should begin with the following line:

```
extern "C" double CALLBACK name( int num_parms, double
argument[], int num_args )
```

where `name` is the name of the function. This provides the following required definitions:

- The routine observes the appropriate calling convention
- The function returns a double precision floating point number
- The function accepts three parameters:
  1. The number of parameters (= 2)
  2. A pointer to an array of double precision floating point numbers
  3. The integer number of elements in the array specified by parameter #2.

Although an explicit function reference is coded in DPL with separate arguments, the library function always receives three parameters. The parameters passed by the explicit function are placed into the array accessed by the second parameter above. For example:

```
mylib(1,2,3,4)
```

would result in three parameters being passed to the library:

1. The number of additional parameters (always 2)
2. A pointer to an array of four elements containing the numbers 1, 2, 3, and 4
3. The number 4 (the number of elements in the array)

An empty explicit function reference argument list, such as `mylib()` above, would still result in three parameters; however, the second parameter would be a null pointer and the third parameter would be zero.

String pointers are passed to the library as specially encoded floating point numbers. An attempt to use one of these numbers in ordinary arithmetic will result in a floating point math error. The include file `dpluserf.h` contains two macros for dealing with these numbers:

```
IS_STRING_ARG( p )
```

This macro tests a double precision floating point number to see if it is a string argument pointer. It evaluates to one (TRUE) if the number is a string pointer and a zero (FALSE) if it is not.

```
STRING_ARG( p )
```

This macro returns the string argument pointer encoded in the double precision floating point number.

The follow code fragment illustrates the use of the string macros.

```
if (IS_STRING_ARG(argument[0])) {  
    int len = strlen(STRING_ARG(argument[0]));  
    ...  
} else  
    (*DPL_functions.error)( "Not a string!" );
```

The parameter count, the argument count, and the `IS_STRING_ARG` macro may be used to perform parameter validation by a library function.

The values in the array parameter are passed by value. This means that any changes made to them within the function will not affect values in the DPL model (i.e., the array parameter entries may be used as local variables). Strings pointed to by string argument pointers should not be modified. Doing so may cause unpredictable results.

All explicit functions must return a valid double precision floating point number. This number will be used to complete expression evaluation at the point where the function reference occurred within the DPL model.

If all explicit function reference arguments are constants, a call will be performed to the function only once at compile-time. Otherwise, a call will be performed each time the containing expression is evaluated and one or more arguments have changed in value unless the library's load function indicated that no optimizations should be performed. In this case, a call will be made each time the containing expression is evaluated whether or not any arguments have changed in value.

## 23.5 DPL™ Callback Functions

---

When a library is loaded, the load function will be passed a list of call-back function pointers from within DPL, should the load function be defined by the library. This list should be saved for use by the library during subsequent function calls. The include file `dpluserf.h` defines this list and allocates storage for it. This file also contains a macro to assist in defining the load function. Another include file, `load.h`, may be included at the beginning of the load function to validate and copy the call-back function pointers. The function pointers may then be used to call DPL to perform various services. For example:

```
#include "dpluserf.h"

LOAD_FUNCTION
{
#include "load.h"
(*DPL_functions.write_to_log)( "Sample User Library\n" );
return( YES );
}
```

Here, `LOAD_FUNCTION` is a macro from the `load.h` include file that defines the load function entry point and declares its parameter list.

Call back functions provide access to DPL functions (e.g., writing to the DPL Log) and to functions that are difficult to perform in a DLL (e.g., loading a resource or performing formatted input and output). Call-back functions are invoked using the following syntax:

```
(*DPL_functions.name)( argument_list )
```

where `name` represents the name of the call\_back function and `argument_list` represents the list of required arguments.

### 23.5.1 Error

This function writes the message argument to the DPL Log and terminates the calling function. The calling function does not receive control again after issuing the call. As part of error processing, DPL will call the library's reset function.

Name:

Error

Arguments:

1. Pointer to char (message)

Returns:

(Does not return)

**23.5.2 Move from left-to-right (with count)**

This function moves the source argument to the destination argument starting with the left-most bytes of the source and destination arguments. The number of bytes moved is provided by the count argument.

Name:

movlrc

Arguments:

1. Pointer to char (the destination)
2. Pointer to char (the source)
3. Integer byte count

Returns:

(None)

**23.5.3 Move from right-to-left (with count)**

This function moves the source argument to the destination argument starting with the right-most bytes of the source and destination arguments. The number of bytes moved is provided by the count argument. This function is useful when moving a string to the right to provide space on the left.

Name:

movrlc

Arguments:

1. Pointer to char (the destination)
2. Pointer to char (the source)
3. Integer byte count

Returns:

(None)

### 23.5.4 Compare from left-to-right (with count)

This function compares the source argument to the destination argument starting with the left-most bytes of the source and destination arguments. The number of bytes compared is provided by the count argument.

Name:

cmplrc

Arguments:

1. Pointer to char (the destination)
2. Pointer to char (the source)
3. Integer byte count

Returns:

- 0 : Arguments are not equal
- 1 : Arguments are equal

### 23.5.5 Write to DPL™ Log

This function writes the message argument to the DPL Log.

Name:

write\_to\_log

Arguments:

1. Pointer to char (message)

Returns:

(None)



### 23.5.6 Capture DPL™ Distribution Accumulation

This function provides DPL with the address of an alternative function for accumulating distributions during an analysis. The function must accept two double precision floating point arguments: a probability (y-axis value) and an outcome value (x-axis value). This function must respond to the call by calling the `accum_dist` call-back function, as appropriate.

Name:

`capture_dist`

Arguments:

1. Pointer to a function accepting two double precision floating point numbers

Returns:

(None)

### 23.5.7 Accumulate DPL Distribution

This function should be called by the function argument of the `capture_dist` function after it is called by DPL to accumulate distribution data. This function is generally used to accumulate distributions on other than the normal outcome (tree endpoint) probability/value pairs of a decision analysis.

For example, assume the library embodies an endpoint value model that involves cashflows for a number of periods. Assume further that each time the library is called to calculate an endpoint value, the cashflows for each period are computed and saved in static storage. If the library captures the distribution accumulation, it will receive a call to accumulate distributions for each endpoint. At that time, it may call the `accum_dist` call-back function once for each time period passing two arguments:

- The current probability times the cash flow for the period (y-axis)
- The number of the period (x-axis)

For each call, the cashflow increment will be added to the total for the period. In this way, a distribution will be accumulated of expected cashflow for each period.

Name:

accum\_dist

Arguments:

1. Double precision floating point number (y-axis value)
2. Double precision floating point number (x-axis value)

Returns:

(None)

### 23.5.8 Get Analysis Outcome

This function may be called after completing an analysis to obtain the expected value and certain equivalent of the run. The function returns a structure containing room for both values; however, the field for certain equivalent will be meaningful only if the model contained a utility function specification. The structure is defined in the include file `dpluserf.h`.

Name:

get\_outcome

Arguments:

(None)

Returns:

An outcome structure containing the expected value and certain equivalent of a DPL analysis. (see `dpluserf.h`)

## 23.6 Code Examples

The sample problem involves a decision between two loan alternatives. The first alternative has possibly higher interest rates but lower loan costs than the second. The object is to minimize the amount paid on the load according to the following formula:

$$\text{amount paid} = \text{costs} + \text{principal} * (1 + \text{rate}) ^ \text{time}$$

where costs is the cost of the loan, principal is the amount of the loan, rate is the interest rate per period, and time is the number of periods. The principal and time values are constant throughout a decision analysis (although the user may perform a sensitivity analysis on them). The costs will depend on the loan decision while the rate will depend both on the loan decision and on a general uncertainty on interest rates.

While the sample is implemented as a DPL program for conciseness, the procedure is the same for a graphical model.

The DPL program file: SAMPLE.DPL

```

decision loan.{a,b};
chance rate.{high, med, low} | loan =
    {.10, .85, .05}, // loan.a
    {.20, .79, .01}; // loan.b
value loan_costs | loan = 1000, 500;
value rate | rate = .12, .09, .08;
value principal = 50000;
value time = 30;

sequence:

get sample.init( principal, time ) then
decide about loan then
gamble on rate and
pay sample( loan_costs, rate )
    
```

The value function described above will be implemented in the user library sample.dll. The library contains two explicit entry points:

- Init – used to set the principal and time values for an entire analysis
- Calculate – used to form an endpoint value (called implicitly)

Since the values for principal and time do not change during a run, it would be inefficient to pass them to the value function at each endpoint. In this particular example, there are only two "constant" values. However, some

problems may involve hundreds of similar terms. Consequently, these values are passed to the library at the beginning of each run (the library will also receive a `begin_analysis` call at this time but this call cannot pass any arguments to the library). The values are passed at the beginning of the sequence section by the line

```
get sample.init( principal, time ) then
```

Because the `init` function returns a value of zero, this line has no effect on the results of the analysis. Also, because the values are set in the sequence section, sensitivity analyses may be performed on principal and time.

The two values that depend on the sequence of events, `loan_costs` and `rate`, are passed to the value model by the line

```
pay sample( loan_costs, rate )
```

The library's `calculate` function will then calculate the value function and return the result.

Since DPL is unaware of the relationships among events and values in a value model supplied by a user library, care must be exercised if DPL is allowed to run analyses with the fast sequence evaluation method. The sample program above will operate correctly with fast sequence evaluation since all value model variables are passed at tree endpoints. The following code fragment would also operate correctly:

```
decide about loan and
pay loan_costs then
gamble on rate and
pay sample( rate )
```

Here, it is assumed that the library's `calculate` function no longer includes the loan costs. In general, it is good practice to promote portions of the value function as far up the tree as possible to assist DPL's optimizations. Because `loan_costs` could be separated from the value function (it appeared as an expression term; i.e., connected to the rest of the function by plus or minus), it could be promoted without problem.

Assume, however, that the loan time was uncertain and the sequence section was as follows:

```
decide about loan and
pay loan_costs then
gamble on time then
gamble on rate and
pay sample( rate, time )
```

The following code fragment would be incorrect:

```

decide about loan and
pay loan_costs then
gamble on time and
get sample.time( time ) then
gamble on rate and
pay sample( rate )

```

Here, the library function `time` would save the value of `time` for the subsequent `calculate` call and return a value of zero. In this case, `time` is not separable and should not be promoted (it is not a separate term of the value function). This sequence would be evaluated correctly, however, if the library's load function suppressed fast sequence evaluation.

When implementing a value function in a user library, it is safest to suppress fast sequence evaluation at the library's load call. If fast sequence evaluation is desired, it must be verified that the policy produced employing that method is the same as that produced with full tree enumeration.

The following sample file can be used with the `sample.dpl` file defined earlier:

Library source file: `SAMPLE.CPP`

```

////////////////////////////////////
// SAMPLE DPL USER FUNCTION LIBRARY
////////////////////////////////////

#include "windows.h"
#include "math.h"
#include "dpluserf.h"

static struct constants {
    double principal, time;
} c;

////////////////////////////////////
// WINDOWS LIBRARY INITIALIZATION
extern "C" BOOL CALLBACK LibMain(HANDLE hInstance,
    WORD wDataSeg, WORD cbHeap, LPSTR lpszCmdLine)
{
    return YES;
}

////////////////////////////////////

```

```

// WINDOWS LIBRARY TERMINATION
extern "C" VOID CALLBACK WEP( int nParameter ) {}

////////////////////////////////////
// LOAD-TIME INITIALIZATION
extern "C" LOAD_FUNCTION
{
    #include "load.h"

    (*DPL_functions.write_to_log)( "Sample DPL User Function
Library\n" );

    // enable fast sequence evaluation
    return( YES );
}

////////////////////////////////////
// GET THE CONSTANT PARAMETERS
extern "C" double CALLBACK INIT( int num_parms,
    double argument[], int num_args )
{
    if ( num_args != 2 )
        (*DPL_functions.error)( "Function \"init\" requires two
arguments" );

    // save a copy of the constants
    c = *(struct constants *)argument;

    return(0.0);
}

////////////////////////////////////
// CALCULATE THE VALUE MODEL

#define loan_costs      argument[0]
#define rate            argument[1]

extern "C" double CALLBACK CALCULATE( int num_parms,
    double argument[], int num_args )
{
    if ( num_args != 2 )
        (*DPL_functions.error)( "Function \"CALCULATE\" requires
two arguments" );
}

```

```
        return loan_costs + c.principal * pow( 1.0 + rate, c.time
    );
}
```

The above code is a simple example of what a library can add to an analysis. The input is validated in both `init` and `calculate`. The `init` function stores the time and principle for later use by the `calculate` function, which returns the total loan payment to DPL.

The functions `LibMain` and `WEP` are required to support Windows DLL initiation and termination (see the Windows SDK documentation for a detailed description of the process of constructing a DLL).

The degree of parameter validation performed by a user library is left to the discretion of the library developer. The sample code checks only that the number of arguments passed to the functions `init` and `calculate` are correct. The library could also have verified that the total number of parameters and the type of each argument were correct. Full parameter validation for the `init` function might be:

```
if( num_parms != 2 )
    (*DPL_functions.error)( "Incorrect number of parameters" );
if( num_args != 2 )
    (*DPL_functions.error)( "Function \"init\" requires two
arguments" );
if( IS_STRING_ARG( loan_costs ) || loan_costs < 0.0
    || loan_costs > c.principal )
    (*DPL_functions.error)( "Invalid \"loan_costs\"" );
if( IS_STRING_ARG( rate ) || rate <= 0.0 || rate > 1.0 )
    (*DPL_functions.error)( "Invalid \"rate\"" );
```


Since the sample problem contains only four paths, this degree of validation would have no effect on run time. For larger trees, the impact on run time could be excessive. If the library developer is also the author of the models which will use the library, minimal validation may be sufficient.


# A Overview of Spreadsheet Linking

DPL provides a complete set of features for linking to Excel spreadsheets. Most DPL analyses involve spreadsheet linking at some point. This appendix summarizes DPL's main spreadsheet linking features and gives suggestions on when and how to use each feature.

## A.1 Types of Spreadsheet Links in a DPL™ Model

---

Calculation Links (  ) are appropriate when the value measures being used for the analysis are calculated in a spreadsheet model, usually a financial model producing NPV, IRR, etc. Notice that the icons have a 2-headed arrow, indicating that data is being exchanged to/from Excel during the entire course of a run. More specifically DPL sends values to Excel (drivers), Excel calculates the output metrics, and sends those back to DPL for outputs. Most DPL analyses employ calculation links and there are several ways to set them up.

Initialization Links (  ) are used when it is more convenient to keep the numbers initializing DPL nodes (probabilities and values) in a spreadsheet rather than within DPL node data. Notice that the icons have an arrow with a single head, indicating that data is sent from Excel to DPL only once at the beginning of a run to initialize node data. With initialization links, you can design your DPL model as a template to be used with several input data spreadsheets.

A DPL model can have either or both types of links.

DPL's linking features are intended to work with driver cells and output/metric cells that are named ranges in Excel. In general when developing an Excel spreadsheet that you intend to link to DPL, you should make a habit of creating range names for any cells or ranges that may later become calculation- or initialization-linked to nodes in DPL. You may also find it helpful to adopt standard conventions for your range names, although DPL does not require this.



When a DPL model is linked to a spreadsheet, DPL uses Excel to recalculate output metrics (e.g., NPV) in each scenario. This approach is easy and simple, and works well with spreadsheets of moderate complexity and size. Very large or complex spreadsheets may require a long time to recalculate. For large spreadsheets of advanced complexity, it may make more sense to use spreadsheet conversion as described in Chapter 15.

## A.2 Calculation Links

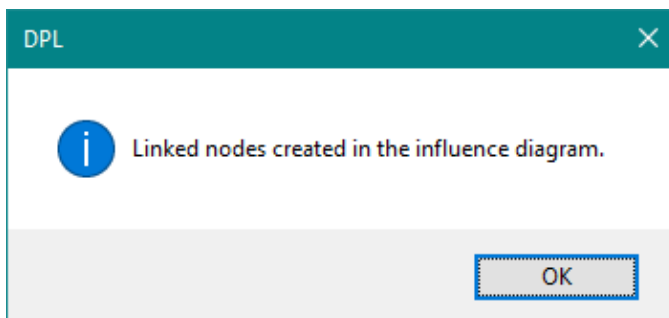
There are three ways to establish calculation links between a DPL model and an Excel spreadsheet. A table summarizing the three methods is given at the end of this section.

### A.2.1 Method 1: Home | Add to WS | Excel Linked Model..

When you are starting with a more or less complete Excel spreadsheet but have not yet developed a DPL model for the spreadsheet, you may want to use the command:

Home | Add to WS | Excel Linked Model..

This command creates a DPL Influence Diagram containing only value nodes from the named ranges in an Excel spreadsheet (all or a specified selection of them). If you're working in Decision Tree focused modeling mode (i.e. the Influence Diagram pane is minimized), you'll be notified that the linked nodes have been added to the Influence Diagram (Figure A-1). Value nodes do not appear directly in the Decision Tree.

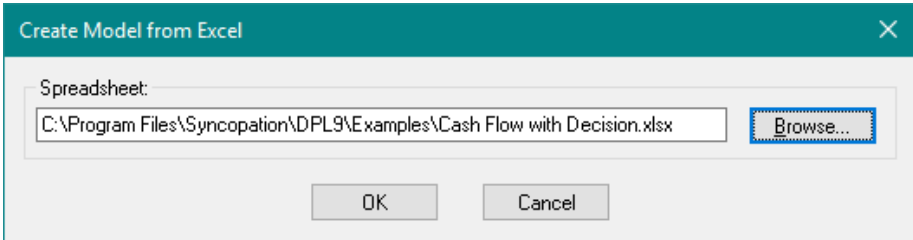


**Figure A-1. Linked node created prompt when in Decision Tree-focused Modeling Mode**

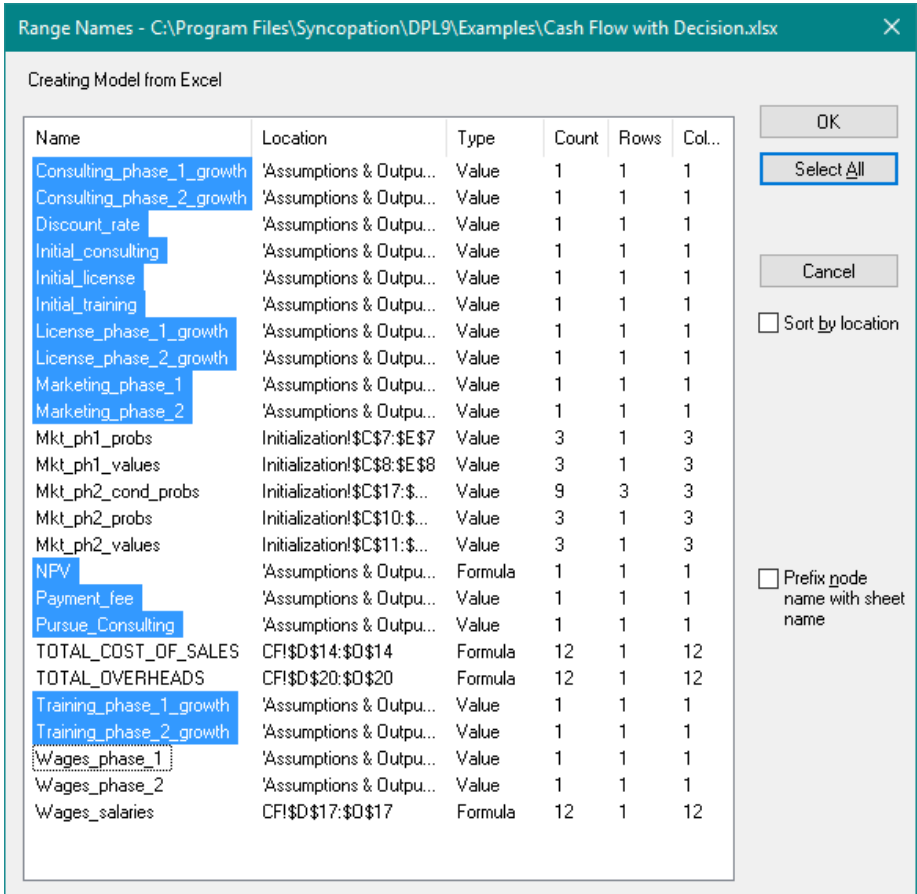
For the selected named ranges/cells, DPL creates value nodes, adds them to the Influence Diagram, and creates influence arcs between driver nodes and metric nodes. When you do this, you will typically select several driver values and at least one metric value (formula) from a list of range names in your Excel spreadsheet. The basic steps are as follows:

- ⇒ Click Home | Add to WS | Excel Linked Model.... Browse to locate the spreadsheet you wish to use.
- ⇒ Select the names of the ranges you wish to use.
- ⇒ Click OK.

DPL creates an Influence Diagram with arcs between driver nodes and metric nodes. This provides you a starting point for your linked model. See Figure A-2 and Figure A-3.



**Figure A-2. Create Model from Excel Dialog**



**Figure A-3. Range Names Dialog for Creating Model from Excel**

When you use this method you will see that DPL includes all of the named ranges in the Range Names dialog, so that you can choose to use any or all of the cells in your linked model.

When creating the nodes, DPL examines the spreadsheet to see what each range contains. For single cell ranges that do not contain a formula, DPL creates driver nodes initialized with data from the spreadsheet. For single cell ranges that do contain a formula, DPL creates metric nodes with no data. For multiple cell, one-dimensional ranges that do not contain any formulae, DPL creates driver array nodes initialized with data from the spreadsheet. For multiple cell, one-dimensional ranges that do contain a formula, DPL creates metric series nodes with no data. For multiple-cell, two-dimensional arrays that do not contain any formulae, DPL creates driver two-dimensional array nodes initialized with data from the

spreadsheet. For multiple-cell, two-dimensional arrays that contain a formula, DPL creates metric two-dimensional array nodes with no data.

Also see Section 4.2 of this manual for an example of Method 1.

### **A.2.2 Method 2: Adding Calculation-linked Nodes to an Existing Model**

Depending on the situation, you may build a model in DPL and a separate spreadsheet in parallel. You might then link the two at some point in the process by adding new nodes to DPL that are linked to the spreadsheet. In this situation, you may wish to use the command:

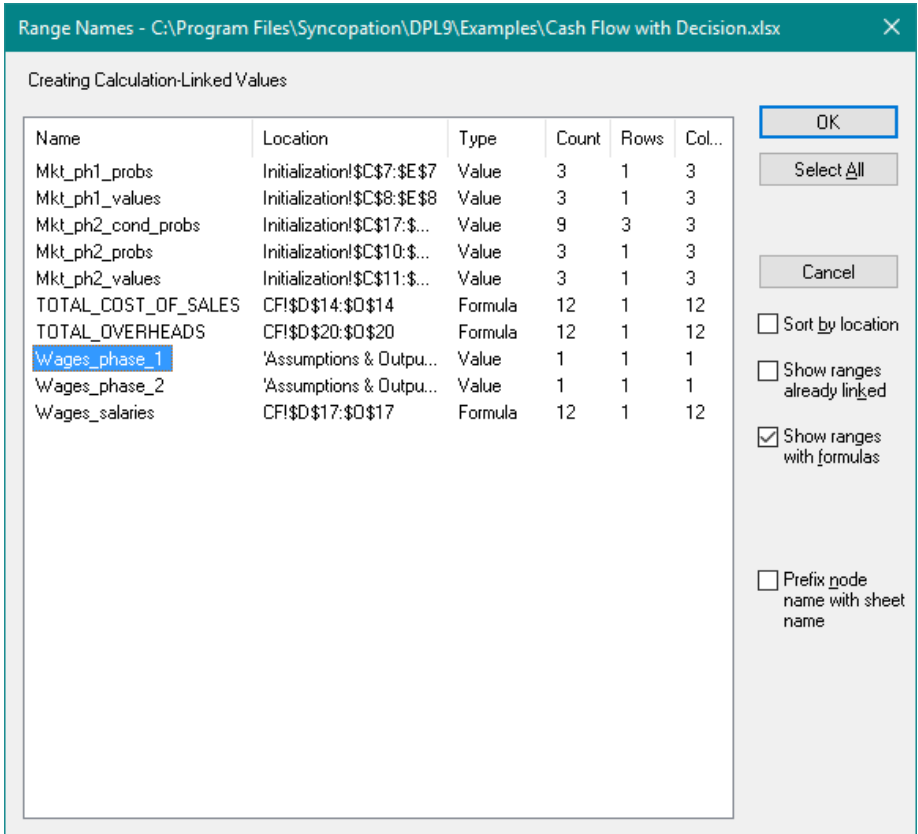
Decision Tree/Influence Diagram | Node | Linked Node | Excel Calculation-linked...

This command adds linked values to the Influence Diagram. The nodes are created as values but (as with Method 1) they can be changed later to decision nodes or chance nodes as desired.

The basic steps for this method are as follows:

- ⇒ Depending on your modeling mode choose Decision Tree or Influence Diagram | Node | Linked Node | Excel Calculation-linked...
- ⇒ If you have not created any Excel linked nodes, browse to locate the spreadsheet you wish to use otherwise DPL uses the currently linked spreadsheet.
- ⇒ Select the names of the cells for which you wish to add linked values.
- ⇒ Click OK.

See Figure A-4.



**Figure A-4. Range Names Dialog for Creating Linked Values from Excel**

DPL adds the new value(s) to your Influence Diagram as value nodes, but does not add influence arcs. You may need to add influence arcs as well as changing the node type(s) to decision or chance nodes. See Section 4.1 for an example of this method.

When you use this method you will notice that DPL lets you check whether to show ranges that are already linked (the default is no) and whether to show ranges with formulas (the default is yes). DPL assumes that since you are adding a new linked node, you probably do not want to choose a range that is already linked to another node, and you probably do want to consider cells with formulas, since they could become new metric nodes. You can check or uncheck these options if needed and the contents of the Range Names list will be updated.

As with Method 1, DPL inspects the spreadsheet and creates nodes of the appropriate type (driver, metric) and dimension.

The Linked Nodes method can be used in conjunction with the Add to WS | Excel Linked Model method. You can start by using the Excel Linked Model command and subsequently add nodes linked to existing or newly named ranges within Excel.

### A.2.3 Method 3: Establishing Calculation Links for an Existing Node

You may already have a node created in your DPL model that needs to be linked to your spreadsheet. In this situation, edit the node as usual and switch to the Links tab of the Node Definition dialog.

The Links tab of the Node Definition dialog allows you to link an existing node to an Excel range.

From the Links tab, within the *Calculation links* section:

- ⇒ Specify *Microsoft Excel* as the Calculation Link type;
- ⇒ DPL automatically sets the spreadsheet to the current linked spreadsheet. If there isn't one, use the Browse button to select the spreadsheet;
- ⇒ Click Range Names...;
- ⇒ Select the desired range from the list;
- ⇒ Click OK.

Alternatively, once you have specified Excel as the Calculation link type, you can copy the cell in Excel (using the Excel command Home | Clipboard | Copy or Ctrl+C), switch back to DPL and click the Paste Link button on the Links tab of Node Definition dialog.

See Figure A-5 and Figure A-6.

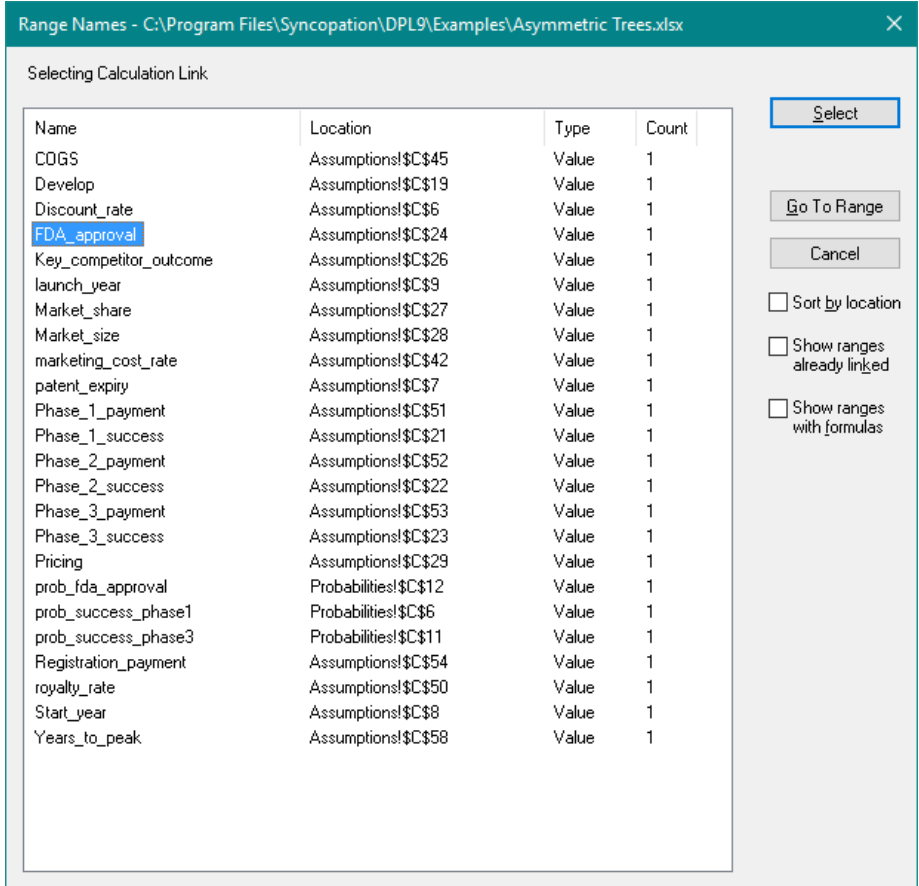
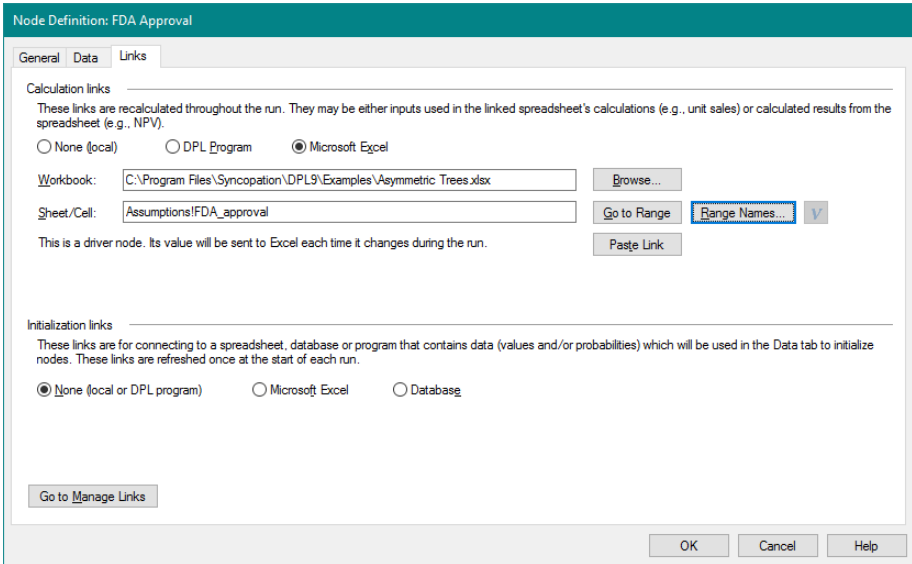


Figure A-5. Range Names Dialog when Linking an Existing Node



**Figure A-6. Range Names Dialog when Linking an Existing Node**

With this method, DPL will also adjust which range names appear in the dialog depending on whether the node you are linking has data in it. If it has data, you will see that "Show cells with formulas" is not checked, because DPL assumes it will be a driver node. Conversely, if the node has no data, by default DPL checks "Show cells with formulas" because DPL assumes the node will be a metric node which is typically linked to a cell with a formula in it. You can check or uncheck these options to see the full set or restricted set of range names if you need to.



**A.2.4 Summary of Calculation Links Methods**

Table A-1 summarizes when each method is appropriate.

<b>Situation</b>	<b>Method</b>
Building a spreadsheet-linked DPL model from scratch	Method 1
Adding a node to the DPL model for a new driver cell in the spreadsheet	Method 2
Linking an existing or newly created node to a range in the spreadsheet	Method 3

**Table A-1. Summary of Methods for Establishing Calculation Links**

**A.3 Initialization Links**

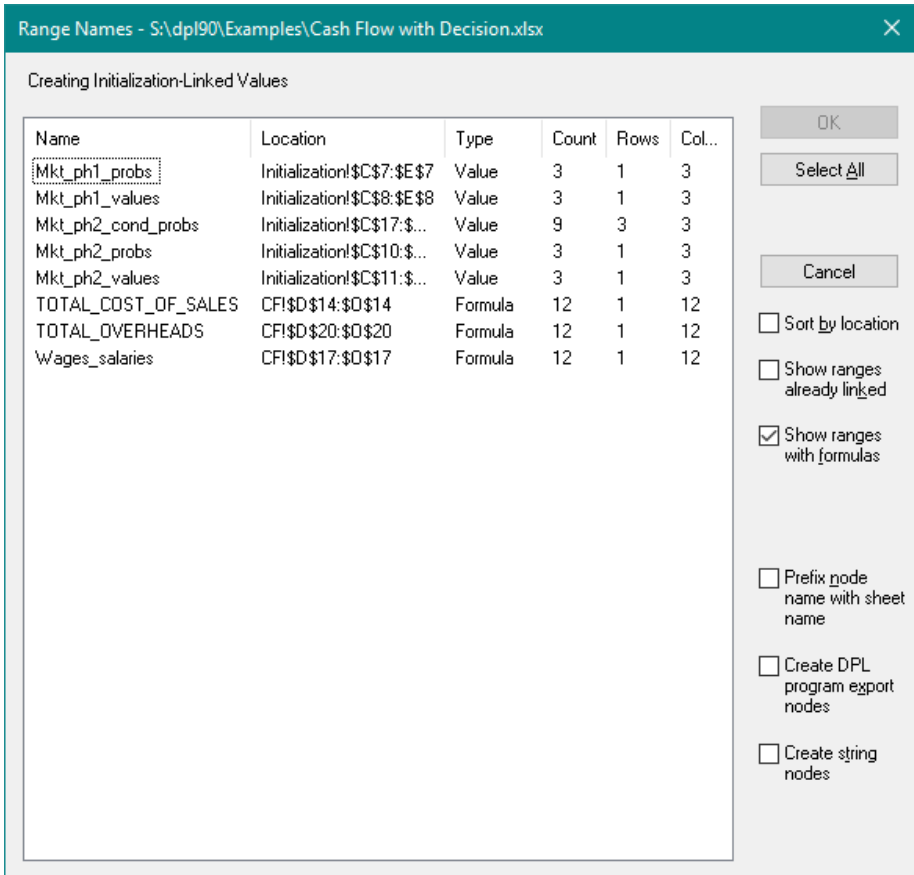
Initialization links are appropriate when you wish to store probabilities and/or values for DPL node data in a spreadsheet. DPL uses the initialization links to set the probabilities and/or values for the linked node(s). Initialization links are different from calculation links in that DPL gets the data from the spreadsheet once at the beginning of a run to initialize the node's values and/or probabilities whereas calculation links exchange data to/from Excel during the entire course of the run.

To use initialization links, first define a range in Excel that contains the probability or value data you wish to link to the node in DPL. The way you define the range will vary depending on the node and its conditioning. For a simple 3-state chance node, you define a 3-cell range containing the probabilities and/or another 3-cell range containing the values. Probabilities and values are initialized separately in chance nodes and must be stored in separate named ranges in Excel.

For a chance node that is conditioned by another 3-outcome event, you would define a two-dimensional cell range (e.g., 3 by 3) in which the rows correspond to the states of the conditioning node, and the columns correspond to the probabilities or values for the states of the conditioned node. An example of a conditioned 3-state chance node is provided.

There are two methods to create initialization-linked nodes.

1) If you have not created the node yet, use Influence Diagram | Node | Linked Node | Excel Initialization-Linked... See Figure A-7.



**Figure A-7. Range Names Dialog for Creating Initialization-Linked Values from Excel**

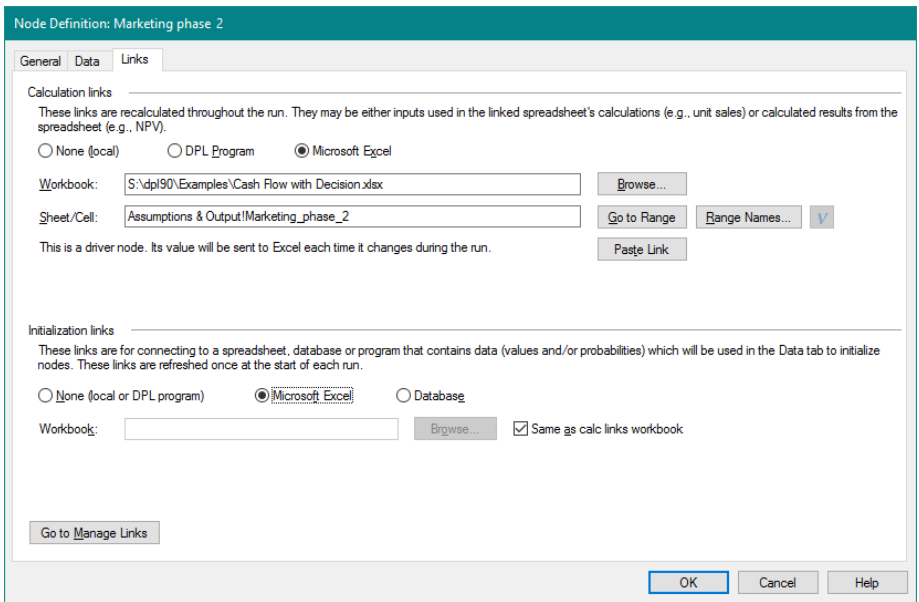
Using this method, DPL creates value nodes for the selected ranges. DPL will automatically create one- or two-dimensional arrays to match the range size.

2) To establish initialization links for an existing node, do the following.

- ⇒ Launch the Node Definition dialog for the node that you wish to link. In this example, the node is named Marketing phase 2.
- ⇒ Switch to the Links tab.

- ⇒ Turn on initialization links by clicking Microsoft Excel within the *Initialization Links* section.
- ⇒ If needed, browse for the spreadsheet workbook you wish to use for the link.

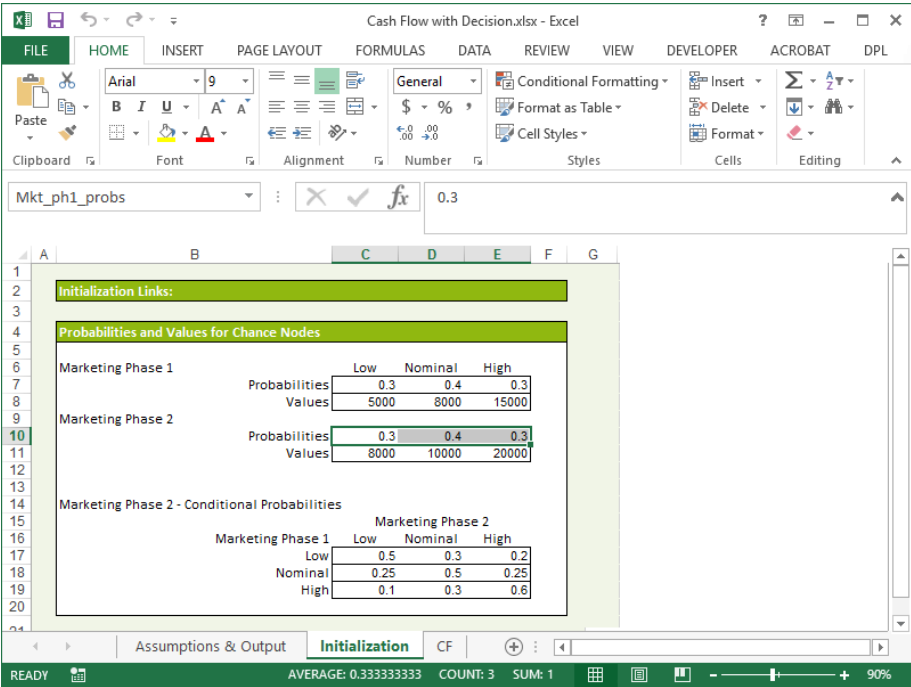
Note that in the dialog shown in Figure A-8, the initialization link is established with a node that already has calculation links. The *Same as calc links workbook* option is checked, so you are linking this node to the same workbook. However, if the node did not already have calculation links or if you want to link it to a different workbook for initialization, you would uncheck the Same as calc links option and use the Browse button to find the new workbook.




**Figure A-8. Node Definition: Initialization Links**

- ⇒ In the Excel spreadsheet, make sure that you have named a range containing the data you wish to initialize the node.

In Figure A-9 the range Mkt\_ph2\_probs has been defined and it contains the probabilities 0.3, 0.4, 0.3. For unconditioned nodes, the range can be a row or a column.

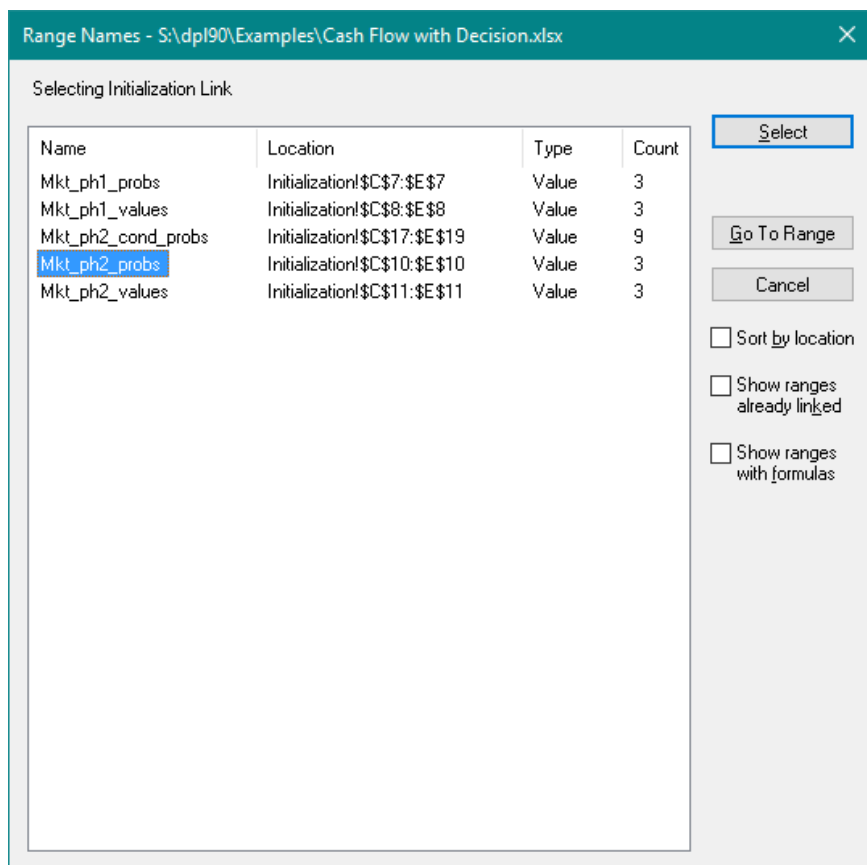


**Figure A-9. Excel Range with 3 Probabilities**

- ⇒ In DPL, switch to the Data tab of the Node Definition dialog.
- ⇒ Select the probability input for the first branch of the node.
- ⇒ Click the Links button (  ).

DPL displays a list of the range names in the spreadsheet that are suitable for linking.

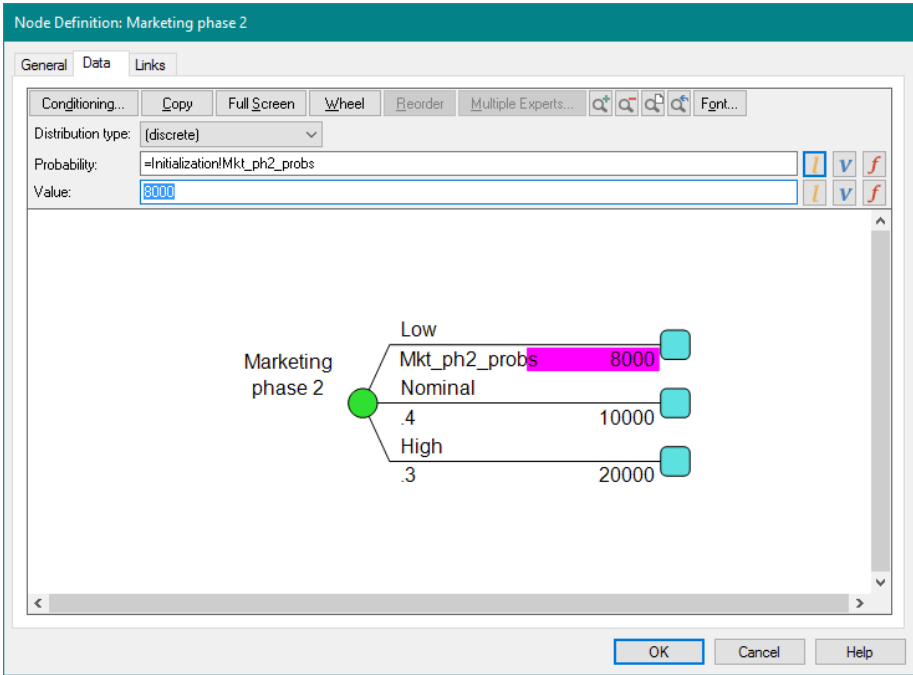
- ⇒ Select the appropriate range, in this example Mkt\_ph2\_probs is selected (Figure A-10)



**Figure A-10. Selecting Range for Initializing Probabilities of Marketing Phase 1 Node**

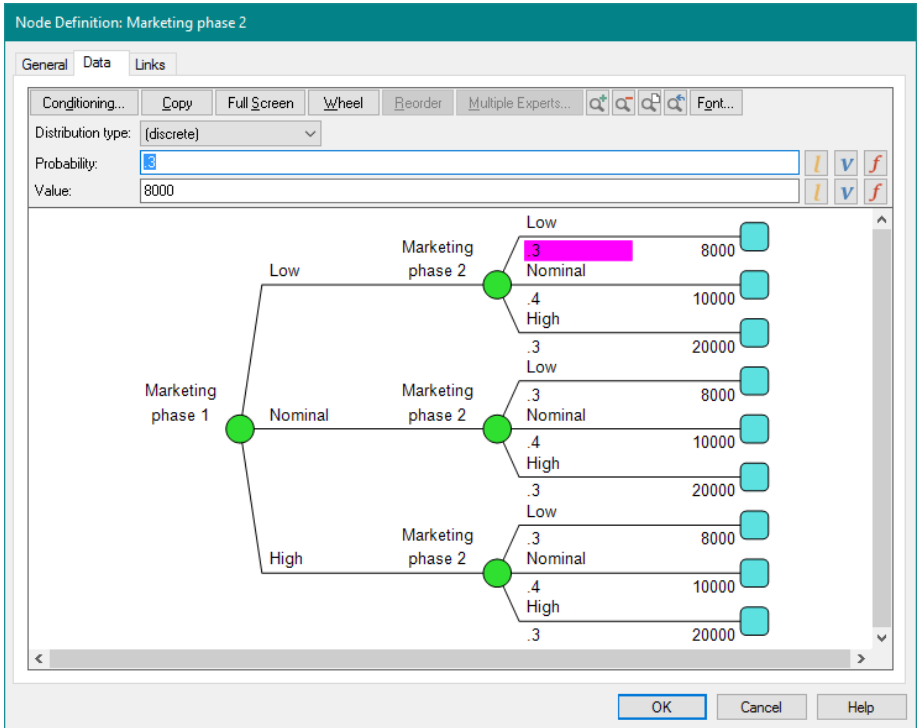
⇒ Click OK.

See Figure A-11. The range appears in the probability edit box for the Low outcome as a reference in "=Sheet!Cell" notation. The remaining probabilities are cleared.



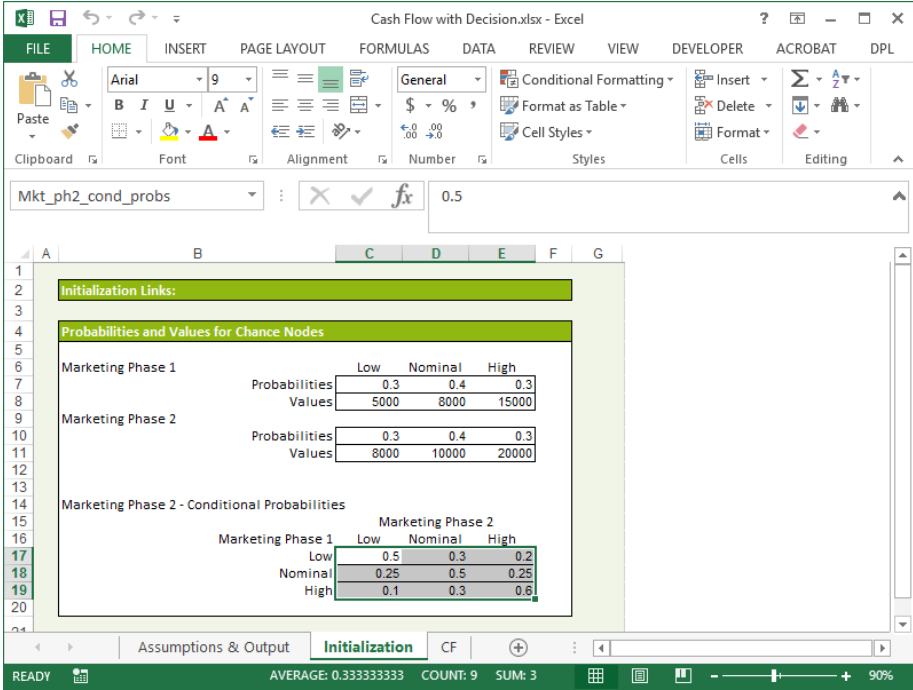
**Figure A-11. Initialization Link for Three-State Chance Node**

Suppose in the example above that the node Marketing phase 2 is conditioned by the 3-state chance node, Marketing phase 1. In this case, you need to specify 9 probabilities and 9 values for the Marketing phase 2 node since it is conditioned. The Data tab for the Marketing phase 2 node would initially look like Figure A-12.



**Figure A-12. Data Tab for Conditioned Chance Node**

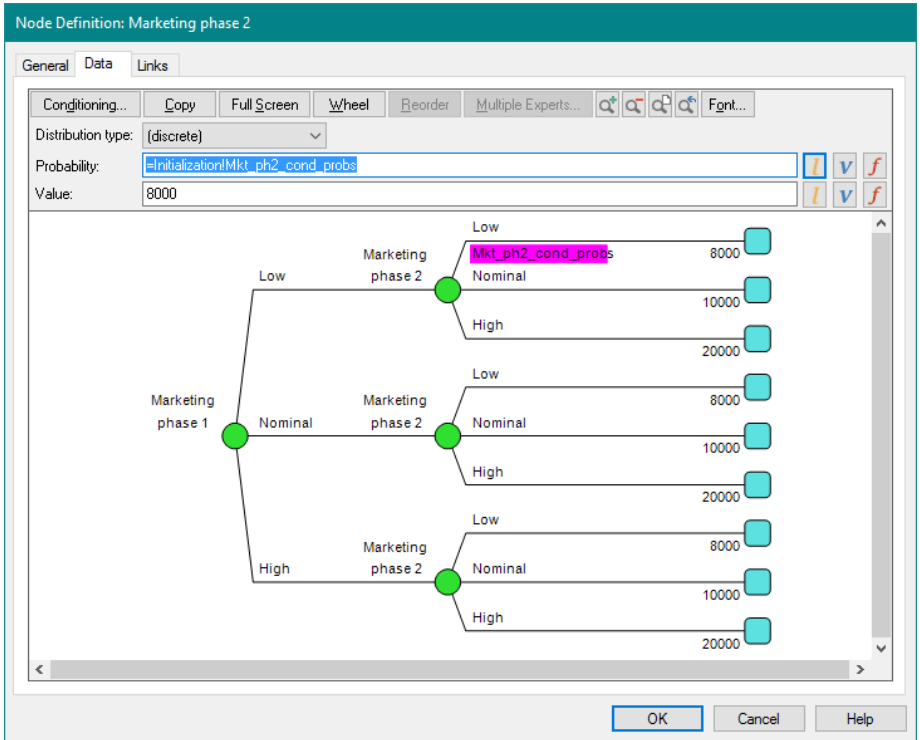
In Figure A-13 the range Mkt\_ph2\_cond\_probs has been defined and it contains the probabilities for each state of Marketing Phase 2 conditioned on each state of Marketing Phase 1. Note that the probabilities sum to 1.0 across the rows.



**Figure A-13. Excel Range with 9 Probabilities for Conditioned Node**

To use this named range to initialize the Marketing phase 2 node, you follow the same steps as before, linking the probability input for the first (top) branch of the node to the range `Mkt_ph2_cond_probs`. The Data tab of the Node Definition dialog will look like Figure A-14 once the initialization link has been established.





**Figure A-14. Initialization Link for Conditioned Chance Node**

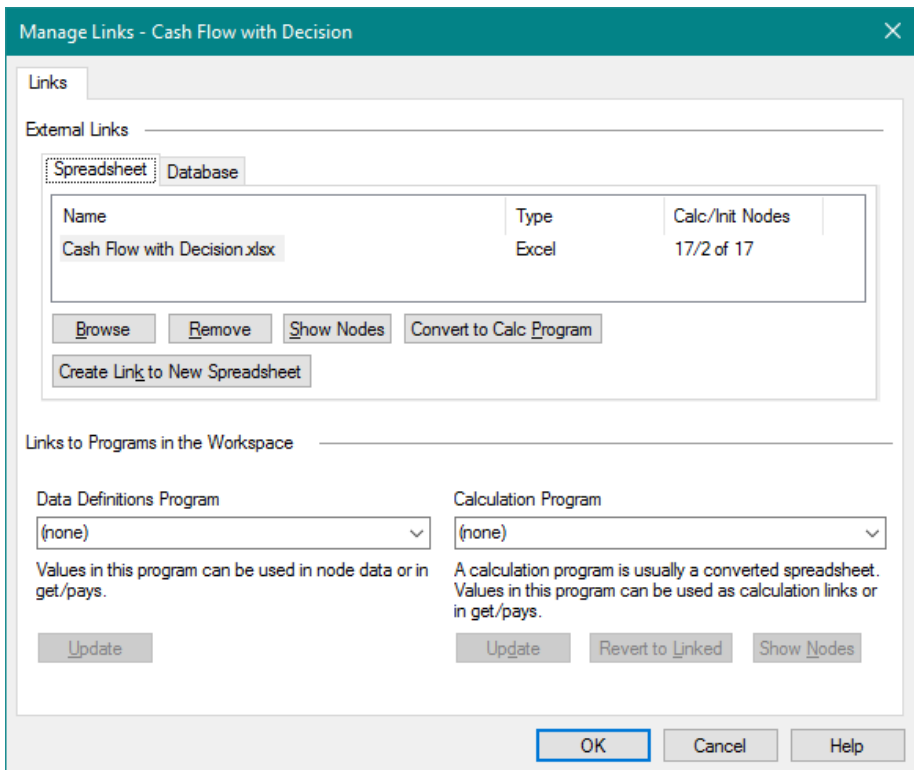
As noted above, in a discrete chance node definition, probabilities and values are initialized separately, so if both will be initialized using links you need to define two range names in the spreadsheet: one for probabilities and one for values. In the examples above, only the probabilities are linked. Values can be linked in the same way; simply use the Links button next to the Value edit box to fill in the Value data in the Node Definition dialog.

Initialization links can be very useful if you are developing large models, especially if you are using your DPL model as an "engine" for multiple Excel templates.

## A.4 Managing Spreadsheet Links

Once you've established links for your individual nodes, you can make changes to all the links as a group using the Manage Links dialog by selecting Influence Diagram/Decision Tree | Links | Manage. See Figure A-15. From this dialog you can:

- Change the spreadsheet name or location.
- Remove all links to the spreadsheet.
- Browse the nodes linked to the spreadsheet.
- Create a link to a new spreadsheet.
- Convert the spreadsheet to a DPL Calculation Program.



**Figure A-15. Manage Links Dialog**

The Manage Links dialog also tells you how many nodes are linked to the spreadsheet. The number of nodes with calculation links is displayed first in the Calc/Init Nodes column of the Linked Spreadsheets list. The number of

nodes with initialization links is displayed second, and the total number of nodes is also given. The linked spreadsheet in Figure A-15 has 17 calculation links nodes and 2 initialization links nodes out of 17 nodes in the model. A couple of the nodes in this model have both calculation links and initialization links.

After you've set up the links in your model, you can use Show Nodes to check that all the nodes are linked to the correct cells; see Figure A-16. This is particularly useful if you used Method 3 to establish the links, since you could have chosen the wrong range by mistake. The dialog shows each linked node, the sheet and range name to which it is linked, the link type and node type. A node is shown twice if it is both calculation-linked and initialization-linked.

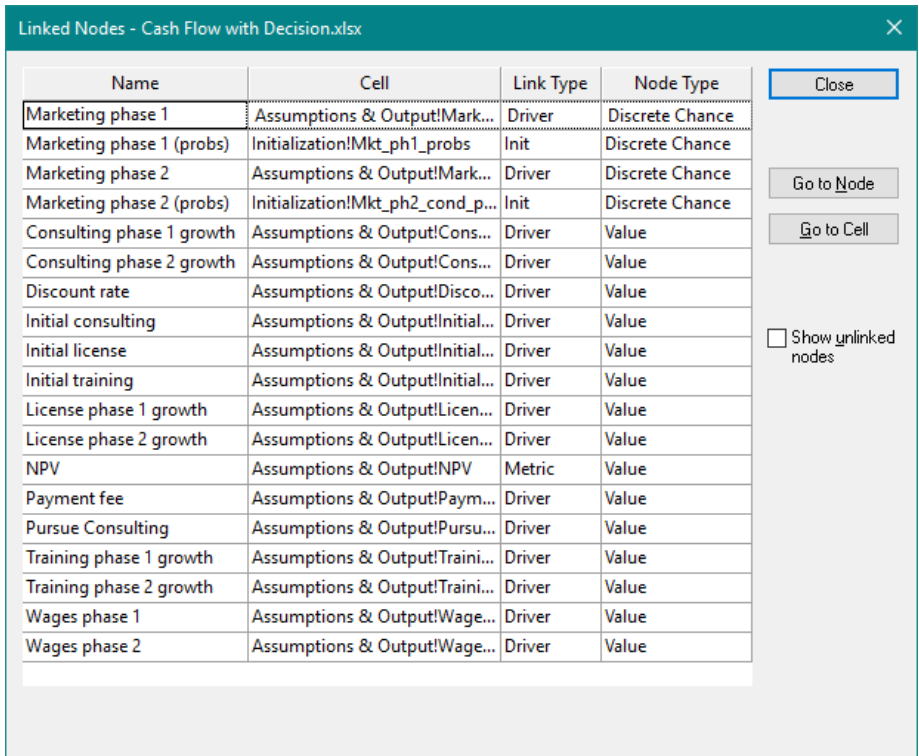


Figure A-16. Show Linked Nodes Dialog

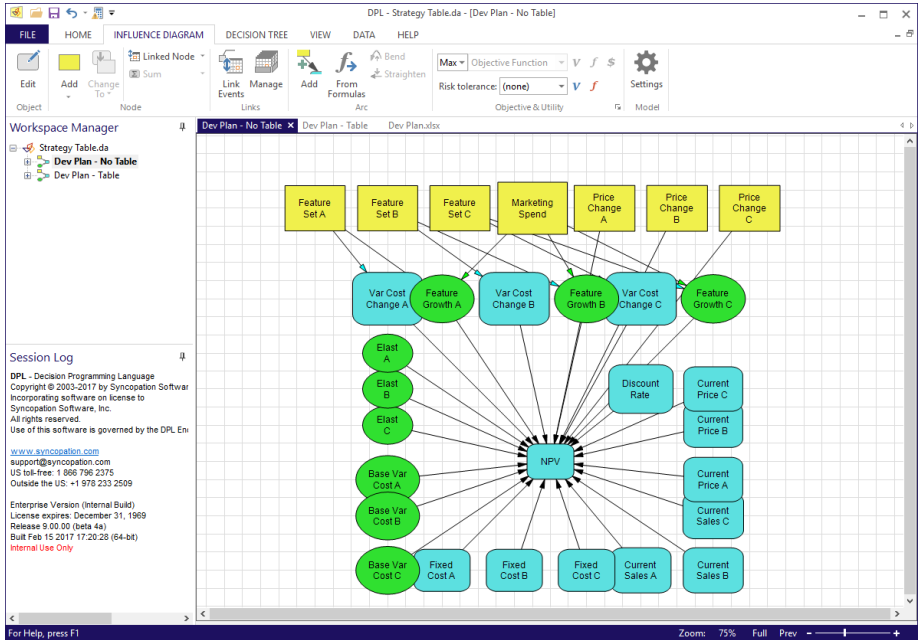
## B Using Strategy Tables

When building a decision model you may find that there are a number of decisions that should be included in the model that must be made together but that not all of the combinations of the decision alternatives make sense or are viable. Initially, it may be easier to think about the decisions individually when framing the problem and developing the model. However, when it comes to analyzing the model, using a Decision Tree with a large number of separate decisions may not be the best way to analyze the problem. In situations such as this, you can use a strategy table node to "aggregate" your these decisions into a manageable number of strategies. In this context, a strategy means a specific combination of decision alternatives for the decisions included in the strategy table.

This chapter assumes you have already been through some of the tutorials in the previous chapters or that you are already familiar with DPL. To learn more about how to use a strategy table node, consider the following example.

- ⇒ Select File | Open.
- ⇒ Navigate to the Examples folder underneath the DPL installation directory, usually C:\Program Files\Syncopation\DPL9\Examples.
- ⇒ Select Strategy Table.da.

DPL opens the Workspace as shown in Figure B-1.



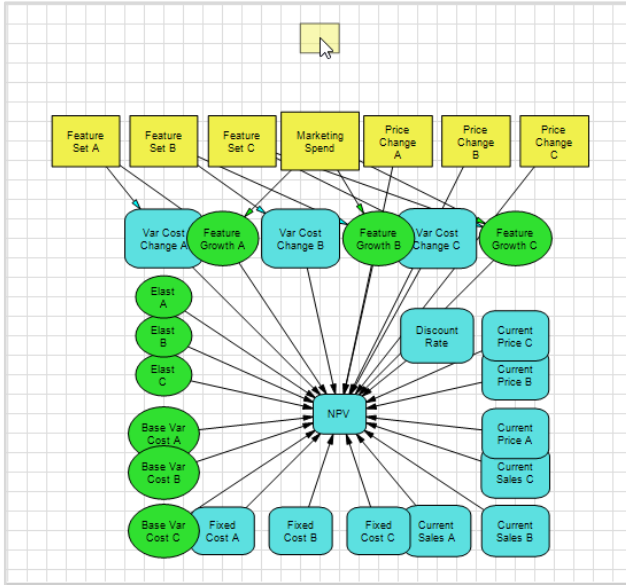
**Figure B-1. Strategy Table Model**

The sample model is for a company with three product lines (A, B, and C). The company must make product development decisions (called Feature Set A, Feature Set B, etc.) for each product line, pricing decisions for each product line, and a decision regarding how much to spend on marketing. The company is faced with uncertainties in the demand for its products as well as uncertainties regarding how much it will cost to produce the updated product lines.

The model in Figure B-1 has over 43 million paths. It is likely that not all of these paths make sense. For example, the company may not want to add a number of new features to all of their products and cut prices on all of them.

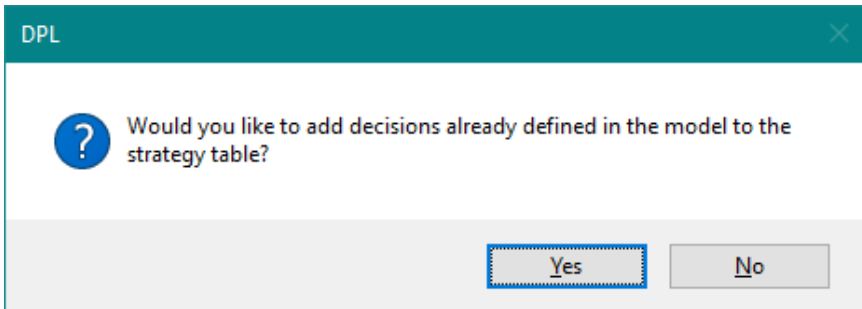
Assume the company is considering a number of alternative strategies for its future product development. You will create a strategy table to model these development strategies.

- ⇒ Click the Zoom Out button on the status bar to make the Influence Diagram smaller and create space above the decision nodes.
- ⇒ Click Influence Diagram | Node | Add | Strategy Table on the ribbon.
- ⇒ Click above the decisions to place the Strategy node as shown in Figure B-2.



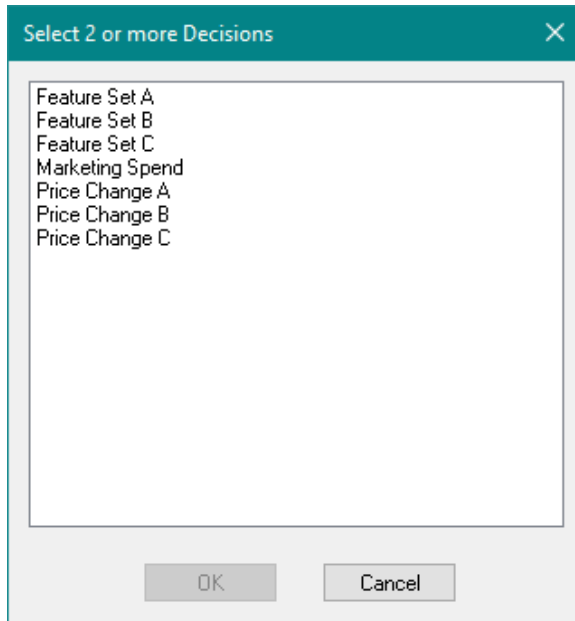
**Figure B-2. Cursor for Placing Strategy Table Node**

You will be asked whether you want to add decisions to the strategy table as shown in Figure B-3. If you answer yes, DPL will create the strategy table with the decisions you select. If you answer no, DPL will create the strategy table with two new default decisions.



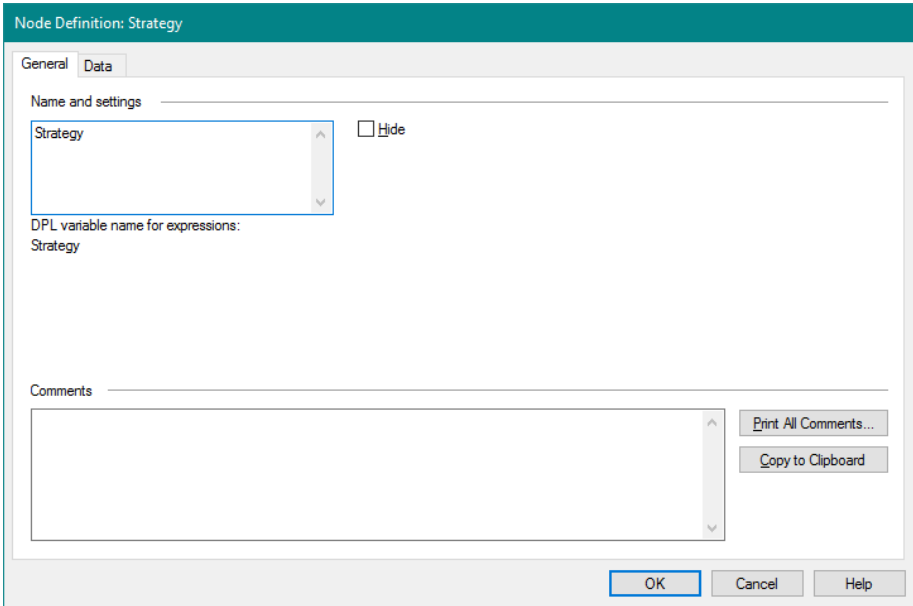
**Figure B-3. Add Existing Decisions to Strategy Table Prompt**

⇒ Click Yes. The Select Decisions dialog appears as shown in Figure B-4.



**Figure B-4. Select Decisions Dialog**

- ⇒ Select all the decisions in the list.
- ⇒ Click OK. The Node Definition dialog appears.



**Figure B-5. General Tab of Node Definition Dialog for Strategy Table**

- ⇒ Change the name of the strategy table to "Strategy" as shown in Figure B-5.
- ⇒ Select the Data tab.

As shown in Figure B-6, the Data tab displays the Strategy Table Window.



Node Definition: Strategy

General Data

Commands Print... Page Setup... Copy Format... Full Screen

	Feature Set A	Feature Set B	Feature Set C	Marketing Spend	Price Change A	Price Change B	Price Change C
Strategy1	Add features	Add features	Add features	Increase	Increase	Increase	Increase
Strategy2	Keep same	Keep same	Keep same	Keep Same	Keep same	Keep same	Keep same
	Minimal	Minimal	Minimal	Decrease	Decrease	Decrease	Decrease

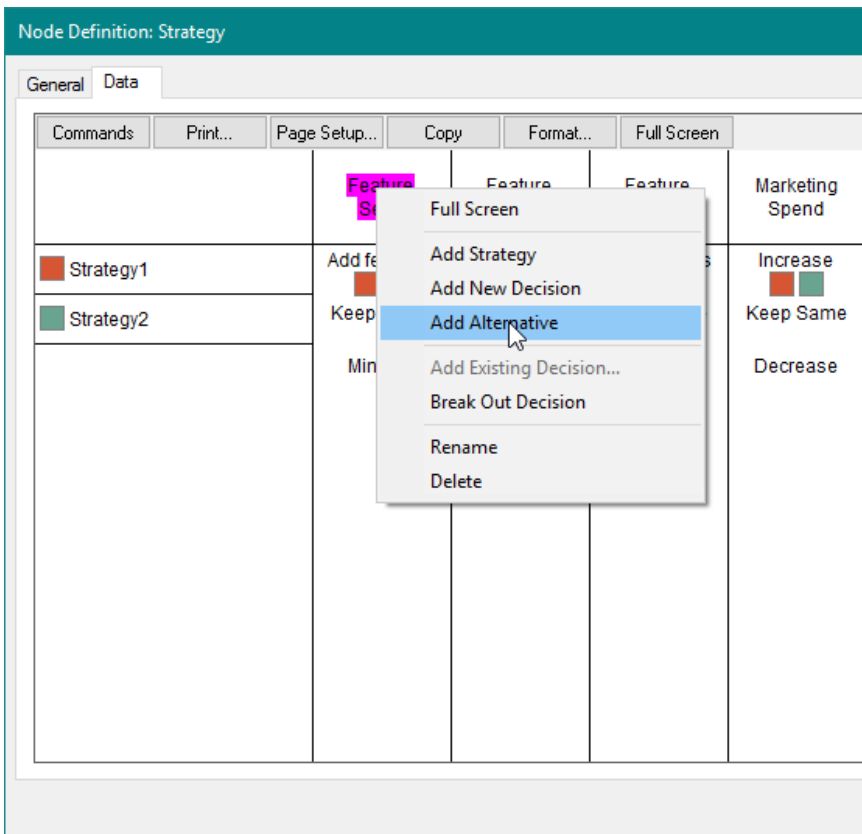
OK Cancel Help

**Figure B-6. Data Tab for Strategy Table**

In the leftmost column of the strategy table, DPL displays the strategies defined. By default, DPL creates two strategies named Strategy1 and Strategy2. DPL uses colored icons to represent each strategy. The first strategy in the strategy table is always represented by a red square icon. In each of the following columns, DPL displays the decisions included in the strategy table. DPL has included the decisions you selected previously. If you had not selected decisions, DPL would have created two default decisions called Decision1 and Decision2, which would be displayed in the columns to the right of the strategy column.

In each decision column, DPL displays the decision alternatives for the decision that heads the column. Under each decision alternative in each decision column, DPL displays the strategy icon that applies to that alternative.

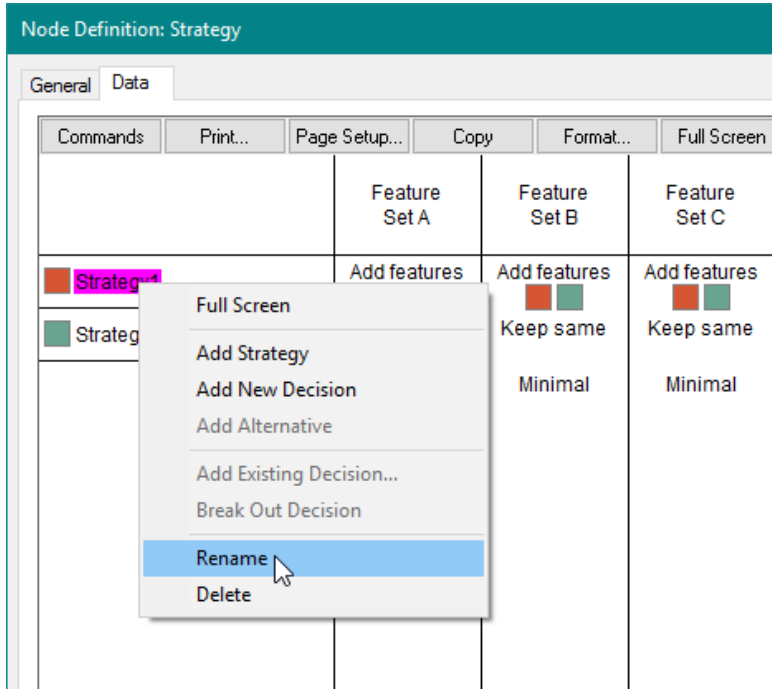
As indicated by Figure B-6 both Strategy 1 and Strategy2 are defined by the first decision alternative of all the decisions in the strategy table. Therefore Strategy1 and Strategy2 are initially identical. The strategy table has a number of buttons across the top. Frequently used strategy table commands can be accessed via the Commands button or by right-clicking within an area of the strategy table to bring up the context menu as show in Figure B-7. The commands enabled within the context menu or drop down list will depend on what you have selected or clicked in the strategy table.



**Figure B-7. Context Menu for Strategy Table with Decision Selected**

You will now define some distinct strategies. To start, you will rename the existing strategies.

- ⇒ Right-click within the Strategy1 cell to bring up the context menu and select Rename (Figure B-8). DPL enters text edit mode.



**Figure B-8. Context Menu for Decision in Strategy Table**

- ⇒ Rename the strategy to be "Status Quo".
- ⇒ Press Enter. The decision column of the strategy table should look like Figure B-9.

The screenshot shows a dialog box titled "Node Definition: Strategy" with two tabs: "General" and "Data". Below the tabs is a menu bar with "Commands", "Print...", "Page Setup...", "Copy", "Format...", and "Full Screen". The main area contains a table with three columns: "Feature Set A", "Feature Set B", and "Feature Set C". The first row is labeled "Status Quo" and the second row is labeled "Strategy2". Each cell in the table contains a label and two small colored squares (red and green).

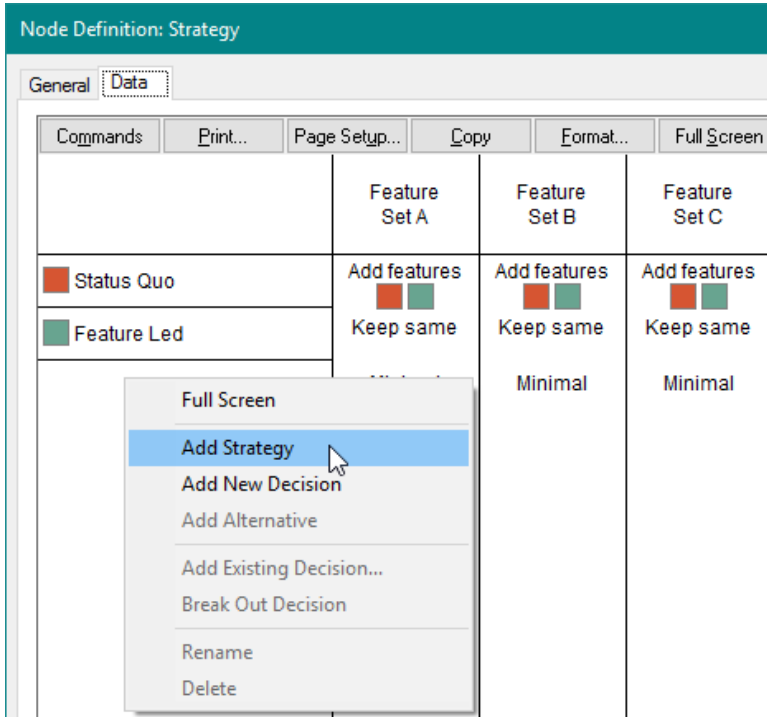
	Feature Set A	Feature Set B	Feature Set C
Status Quo	Add features ■ ■	Add features ■ ■	Add features ■ ■
Strategy2	Keep same	Keep same	Keep same
	Minimal	Minimal	Minimal

**Figure B-9. Renamed Status Quo Strategy**

⇒ Double-click Strategy2 to put it into edit mode and rename it Feature Led.

You will now create two new strategies.

⇒ Right-click anywhere within the strategy table and select Add Strategy from the context menu as shown in Figure B-10.

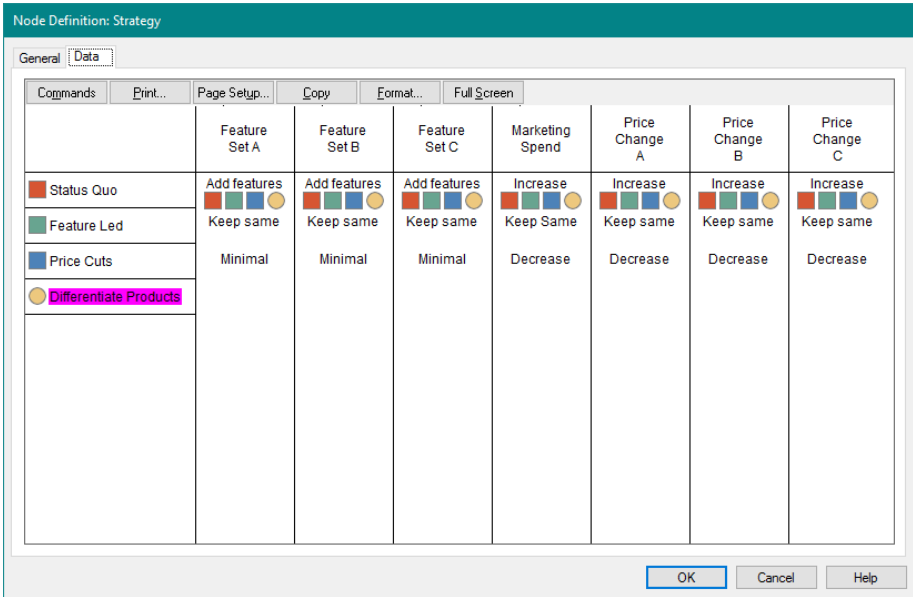


**Figure B-10. Selecting Add Strategy from Context Menu**

DPL creates a new strategy called Strategy3, color-coded blue, and puts you into text edit mode.

- ⇒ Rename the new strategy to be "Price Cuts".
- ⇒ Repeat the create strategy process to create another strategy called "Differentiate Products". This will be color-coded yellow.

Your strategy table should now look like Figure B-11.

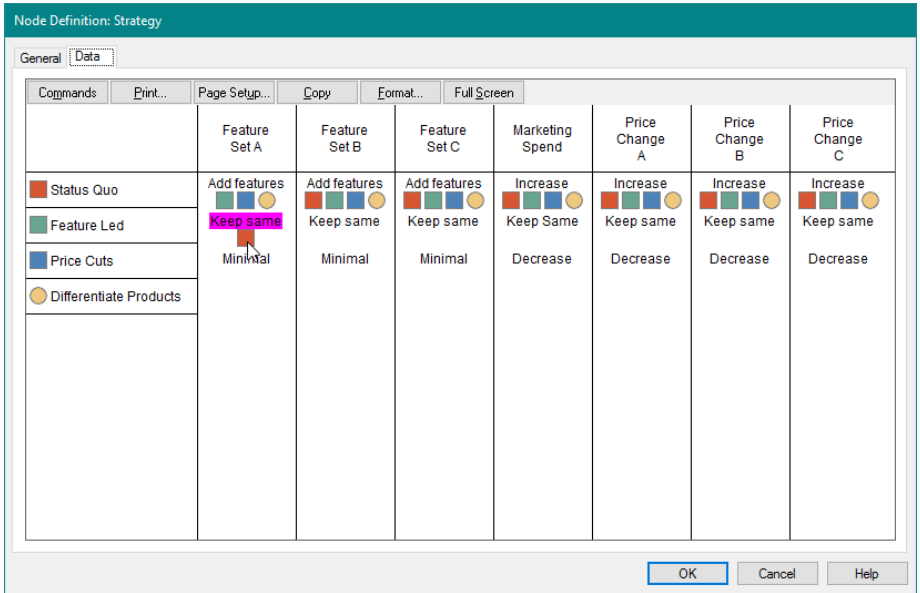


**Figure B-11. Strategy Table with Four Strategies**

DPL allows you to define up to 12 strategies in a strategy table. For this example, four strategies will suffice. You will now make the four strategies distinct from one another by defining them in terms of the decision alternatives that each represents.

To define strategies in terms of a decision alternative, you drag the icon to for a given strategy to below the decision alternative that defines the strategy for each decision in the strategy table. You can either drag the strategy icons from the strategy column on the left to the decision alternative columns or within the decision columns from one alternative to the other.

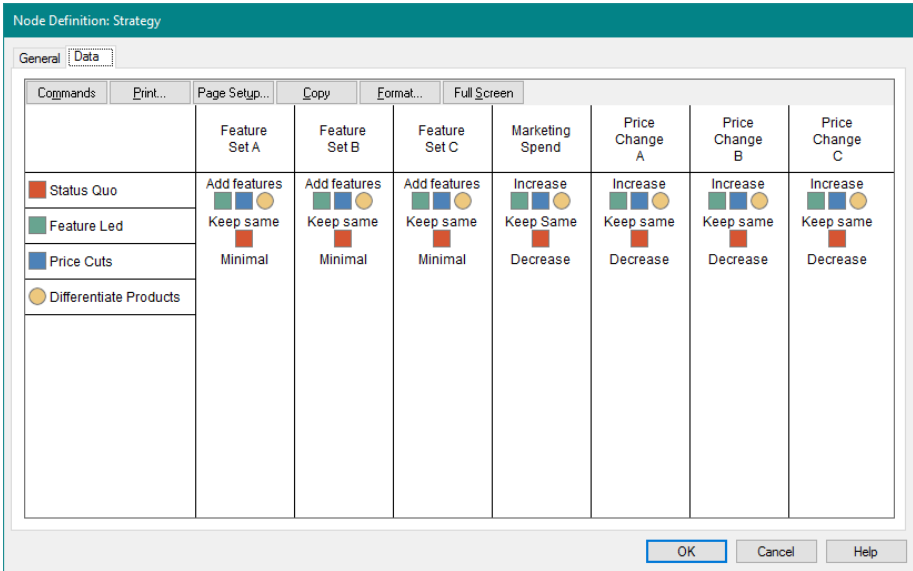
- ⇒ Drag the red square icon representing Status Quo to below the Keep same alternative under Feature Set A. See Figure B-12.



**Figure B-12. Dragging Strategy Icons to Define Strategies**

⇒ Repeat this procedure to drag the icon for Status Quo to the Keep same alternative of the remaining decisions.

The Status Quo strategy should now be defined as indicated in Figure B-13.



**Figure B-13. Status Quo Strategy Defined by Decision Alternatives**

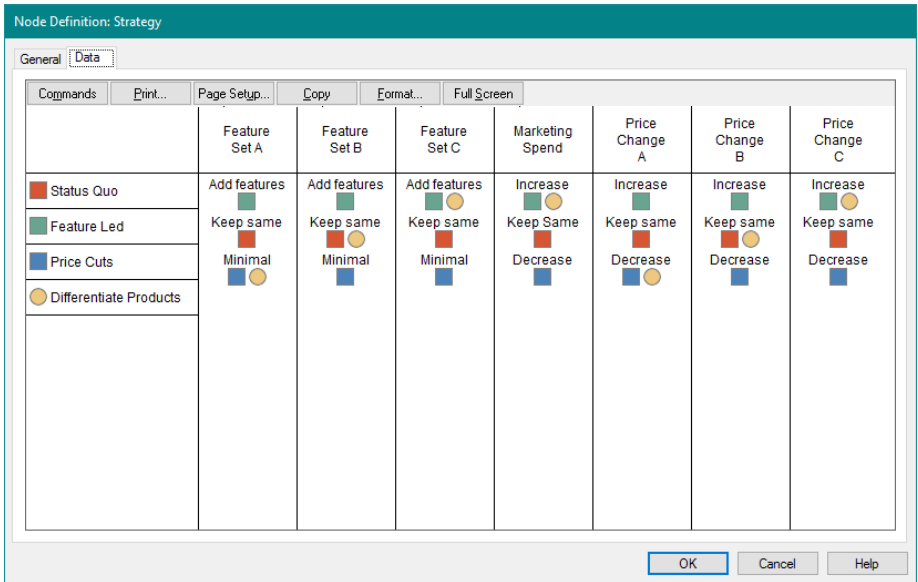
⇒ Define the remaining three strategies as indicated in Table B-1. In the table, Add = Add Features, Same = Keep same, Min = Minimal, Inc = Increase, and Dec = Decrease.

	Feat Set A	Feat Set B	Feat Set C	Market Spend	Price Change A	Price Change B	Price Change C
Feature Led	Add	Add	Add	Same	Inc	Inc	Inc
Price Cuts	Min	Min	Min	Dec	Dec	Dec	Dec
Differentiate Products	Min	Same	Add	Inc	Dec	Same	Inc

**Table B-1. Strategy Definitions**



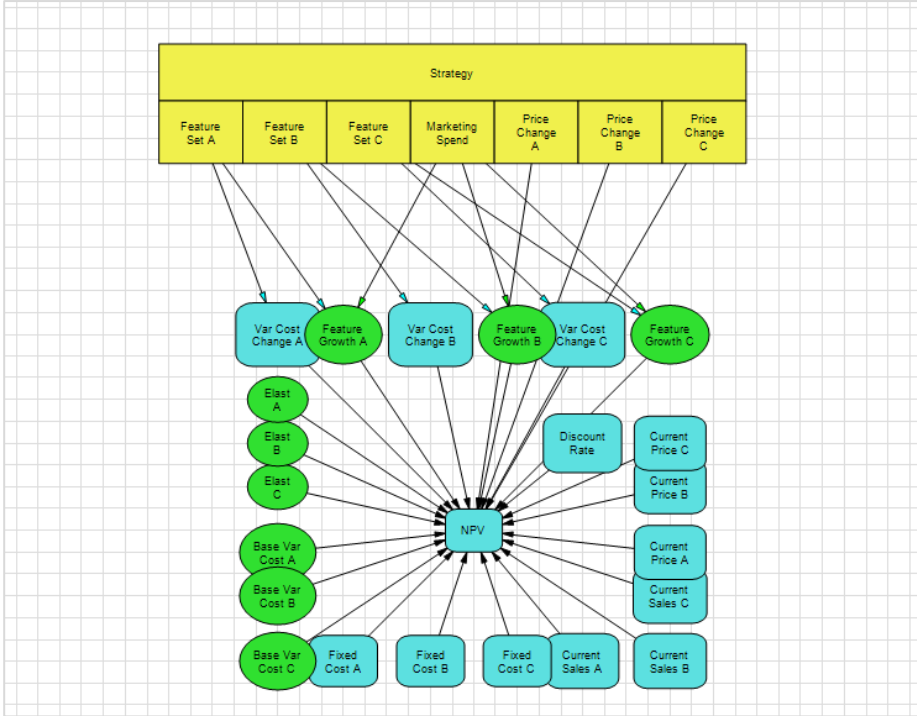
After completing the strategy definitions, your strategy table should look like Figure B-14. Note that each strategy has to be defined by an alternative of each decision in the strategy table (i.e., the icon representing the strategy has to appear in each decision column). However, not every decision alternative needs to be used to define a strategy, i.e., an alternative may have no strategy icons beneath it, indicating that the alternative is not considered in any strategy. In this example, every decision alternative is used at least once in some strategy.



**Figure B-14. Strategy Table with Completed Strategy Definitions**

- ⇒ Click OK to close the Node Definition dialog for the strategy table.
- ⇒ Click on whitespace or press ESC to deselect the strategy node.

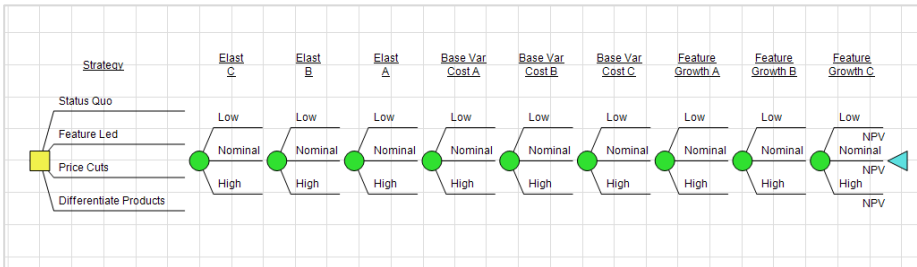
Your Influence Diagram should now look like Figure B-15. Note that DPL has grouped the decisions added to the strategy table beneath the strategy node.



**Figure B-15. Influence Diagram with Strategy Node Added**

⇒ Press the Tab key to switch to the Decision Tree pane.

DPL has also updated the Default Tree within the Decision Tree pane (Figure B-16). It has removed the individual decisions and replaced them with the strategy table and the table's four alternatives (strategies). The Decision Tree now has 78,732 paths.



**Figure B-16. Decision Tree with Strategy Node Added**

You can now run the model to find the optimal product strategy.

In this example, you saw how to create a strategy table, add a number of existing decisions to it, create strategies and define the strategies in terms of the decision alternatives. The strategy table interface also allows you to create decisions directly within the strategy table, delete decisions, remove decisions (but maintain them in the Influence Diagram), reorder the decisions in the strategy table (by dragging and dropping the decisions in the decision columns) and create and delete decision alternatives.

## C System Requirements and Compatibility with Older Releases

DPL 9 is a 64-bit Microsoft Windows application. It is compatible with Windows Vista, 7, 8 (not RT), 8.1 and 10. Windows 7 or later is recommended for best performance.

DPL can optionally link to Microsoft Office Excel versions 2007, 2010, 2013, and 2016.

A minimum of 25 MB of free disk space is required. For a complete installation, with all documentation, approximately 45 MB is required.

A display with resolution 1280 by 800 or higher is required. A larger, higher resolution screen provides a better modeling experience.

DPL 9 is compatible with previous releases of DPL. Decision Analysis Workspace/Project files (.da) from DPL 4.0, 5.0, 6.0, 7, and 8 can be loaded into DPL 9. Note that .da files are called Decision Analysis Workspace files but were called Decision Analysis Project files in some earlier versions.

You can import DPL Influence Diagram files (.inf) from DPL 3.x into a DPL 9 Workspace using File | Import. You can also import DPL Program files (.dpl) in the same manner.

You can "back save" a DPL 9 Workspace file to either a DPL 7 or 8 Workspace file using File | Save As. Some information for new features in DPL 9 may be lost.

# D Keyboard Shortcuts

Action	Keyboard Shortcut
Change Conditioning	Ctrl+Y
Change node type to most recently changed type	Ctrl+T
Clear Memory (delete unsaved Endpoint Database, Policy Tree, Risk Profile, etc.)	F9
Clear All Output associated with Model	Ctrl+Shift+F9
Copy	Ctrl+C; Ctrl+Insert
Cut	Ctrl+X; Shift+Delete
Find	Ctrl+F
Go to most recently active model	Ctrl+F12
Go to Node	F5
Help	F1
New Workspace	Ctrl+N
Open Workspace	Ctrl+O
Paste	Ctrl+V; Shift+Insert
Play Endpoints	Alt+F10
Print	Ctrl+P
Redraw	Ctrl+R
Repeat Find	F3
Replace	Ctrl+H
Run Decision Analysis/Monte Carlo simulation using most recent evaluation method and produce outputs as specified on the ribbon	F10
Run Portfolio Analysis and produce most recently requested portfolio outputs	Shift+F10
Run most recently requested Tornado	F8
Save Workspace	Ctrl+S
Select all	Ctrl+A
Snap to Grid	Ctrl+G
Switch between Influence Diagram/Decision Tree pane	Tab
Toggle between Graphics/Text Mode in Policy Tree and Policy Summary	Tab
Undo	Ctrl+Z; Alt+Backspace
View Workspace Manager	Alt+F12
View Properties	Alt+Enter
View Session Log	Shift+F12
Zoom Full	Ctrl+L; Ctrl+Shift+R
Zoom In	Ctrl+>
Zoom Out	Ctrl+<
Zoom Previous	Ctrl+E

## E Glossary of DPL™ and Decision Analysis Terms

### A

---

**Alternative:** A state of a decision node. DPL chooses among alternatives during tree rollback to maximize (or minimize) the objective function. The set of alternatives for a decision is the range of possible actions to take for that decision.

**Always Gamble:** Tells DPL to always gamble on the outcome of a chance event overriding the "Don't Gamble" specification, which is used during the creation of an Event Tornado Diagram. Events with "Always Gamble" specified are never replaced by their expected values. *See "Don't Gamble."*

**Arc:** *See Influence Arc.*

**Array:** A one- or two-dimensional set of numbers. Arrays can be used to store related numbers, much as a look-up table is used in a spreadsheet. The syntax for a one-dimensional array is `arrayname[column_subscript]`. The syntax for a two dimensional array is `arrayname[row_subscript][column_subscript]`. Row and column subscripts start at zero, i.e., the first element of a two-dimensional array is `arrayname[0][0]`.

**Array Formulas:** Operations in DPL ordinarily take scalars (single values) as operands and return scalars as results. When an operand that is ordinarily required to be a scalar is replaced with an array, the containing formula is said to be an array formula. An array formula must be enclosed in the special symbols { = and } (the notation used by Microsoft Excel).

**Array Subscripts:** Numbers or formulas used to reference the elements of an array. For example, `arrayname[1][3]` has subscripts of 1 and 3. Array subscripts can also be formulas.

**Arrowheads:** In the Influence Diagram, the color of an arc's arrowhead indicates the type of conditioning the arc implies. Black arrowheads indicate timing only; blue arrowheads indicate values only; green arrowheads indicate probabilities only; light orange arrowheads indicate both values and probabilities.

**Asymmetric Node:** A Decision Tree node with branches that lead to different nodes.

**Attribute:** A measure of value (Profit, Health Effects, Environmental Effects) tracked for each path of a Decision Tree during an evaluation. Attributes are usually combined at each endpoint by an objective function, generating a single measure of value that is used during tree rollback. Probability distributions for each attribute and the objective function may be graphed separately.

## B

---

**Base Case Tornado Diagram:** Used with probabilistic models; a diagram generated by evaluating the outcome of a model when each chance event is individually set to a high and a low state while all other chance events are set to a nominal state. The Base Case that the results are compared to is the result when all chance events are set to a nominal state.

**Base Result:** The expected value (and certain equivalent, if risk tolerance is used) of the objective function for the initial run of the model in a Value Tornado before any of the sensitivity variables are tested, i.e., with the model as currently defined in the Model Window. The Base Result also establishes the optimal policy for the model.

**Bayes' Rule:** A mathematical method for reversing the order of conditioning in chance events (e.g., used to assign probability when you have the probability of A given B, but need the probability of B given A). The formula for Bayes' Rule is:  $P(A_i|B) = \frac{P(A_i) \times P(B|A_i)}{P(A_1) \times P(B|A_1) + P(A_2) \times P(B|A_2) + \dots + P(A_n) \times P(B|A_n)}$

**Beta Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Binomial Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Branch:** An element in a Decision Tree representing one state of an event. Decision node branches represent alternative choices and discrete chance node branches represent possible outcomes.

**Branch Block:** A modeling option that temporarily removes one or more alternatives from consideration in a given decision node instance.

**Branch Control:** A modeling option that forces the outcome (or branch) of an event to a particular state in a Decision Tree.

---

## C

---

**Certain Equivalent (CE):** The certain amount equivalent to the expected utility of the model, taking risk attitude into account. Specifically, the minimum guaranteed amount a decision-maker would accept in place of an uncertain lottery. When a utility function is specified, the certain equivalent is provided in addition to the expected value.

**Chance Node:** A node in the Influence Diagram or Decision Tree that represents an event whose outcomes are uncertain. Chance nodes can be either discrete or continuous.

**Chi Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Chi-Square Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Code:** A model, or part of a model, written in the DPL programming language. Can be viewed in a Program Window.

**Command Window:** A testing and debugging environment in which you can display or temporarily change any value or calculation in a model.

**Compilation:** Before running an analysis on a model, DPL compiles the model. During compilation, DPL checks for syntax and other errors and prepares the model for evaluation by the DPL engine.

**Conditional Decision Policy:** A policy containing decisions which will be made after initial uncertainties are resolved. The optimal alternative may depend on the state of the uncertainties, and may therefore be different in different parts of the Decision Tree.

**Conditioned:** When a value or event has separate numbers or formulas for either its values or probabilities or both depending upon the states of one or more other events. For example, if  $\text{Costs} = 0$  if Develop Drug is No and  $\text{Costs} = \text{Development Costs} + \text{Launch Costs}$  if Develop Drug is Yes, then Costs is conditioned by Develop Drug. See also Dependency.

**Constraint Function:** An "if-then-else" expression that tests a condition and specifies different objective functions for true or false outcomes or prunes the tree, eliminating any further branches.



**Continuous Chance Node:** A node in a model that represents an uncertain event with more than a finite number of outcomes. A continuous chance node does not have a fixed number of states. To generate a distribution of outcomes for a continuous chance node, samples are continuously drawn from a named distribution. The graphical symbols for a continuous chance node are dark green ovals in the Influence Diagram and dark green circles in the Decision Tree. A model which contains continuous chance nodes is always evaluated with Monte Carlo simulation.

**Control, Branch:** *See Branch Control.*

**Controlled Node:** A node in the Influence Diagram representing an event whose states are always controlled in the Decision Tree. The graphical symbols for a controlled node are a white rectangle in the Influence Diagram and a white square in the Decision Tree.

**Conversion:** DPL can convert spreadsheet models to blocks of DPL code. The resulting code can be included in DPL models or edited and run as a stand-alone DPL program, so that DPL acts as a "spreadsheet compiler".

**Cumulative Distribution:** a graph generated by DPL whose X-coordinates represent outcome values and whose Y-coordinates represent the sum of the probabilities of all possible outcomes less than or equal to the associated outcome value. In the Home | Run group, you can choose to have DPL generate and display cumulative distributions for the optimal policy or all initial decision alternatives. Also, if you have defined more than one attribute for the model, a separate cumulative distribution for each of the attributes can be generated. The cumulative distributions are displayed in a Risk Profile Chart.

**Current Model:** The model that has been most recently run or compiled. The current model's item in the Workspace Manager is indicated by bold face type.

**Cycle:** A set of nodes and influence arcs that create a loop in which a node depends directly or indirectly on itself. An Influence Diagram may not include cycles.

---

## D

---

**Decision Alternative:** *See Alternative.*

**Decision Node:** A node in the Influence Diagram or Decision Tree representing an event that the decision-maker has control over, i.e., an event with alternative choices for the decision-maker to choose among. During evaluation, the alternative that maximizes (or minimizes) the objective function is chosen. The graphical symbols for a decision node are a yellow rectangle in the Influence Diagram and a yellow square in the Decision Tree.

**Decision Sensitive:** A variable in a model such that when its value is changed, a different decision policy is optimal from that which is optimal for the variable at its current setting.

**Decision Tree:** A graphical representation of a decision model that displays the sequence of events including the order of decisions and uncertainties and when get/pay expressions occur.

**Decision Tree pane:** A pane of the Model Window that provides a graphical interface to manipulate the Decision Tree in a model.

**Default State:** The state of an event that is user-defined as the "default" setting for purposes of linking to an Excel spreadsheet. If the state of an event is unknown for a particular path on an asymmetric tree, DPL assumes the event is in its default state.

**Default Tree:** The Decision Tree that DPL builds automatically in the Decision Tree as you develop a model in the Influence Diagram. DPL builds the Default Tree based on the decision nodes, chance nodes, value nodes and influence arcs you define in the Influence Diagram. The Default Tree is always symmetric. DPL will continue to build the Default Tree until you edit the Decision Tree yourself directly in the Decision Tree.

**Definition Section:** A section of a DPL program which specifies and initializes the elements of a decision model — decisions, uncertainties, variables, values, series, arrays, etc. The Influence Diagram is a graphical representation of a definition section.

**Dependency:** Node B depends on node A if one of the data expressions for B includes A, but B does not have a separate data expression for each state of A. For example, if  $Revenues = Units * Price$ , then Revenues depends on Units and Price. *See also Conditioned.*

**Deterministic Model:** A model that only has deterministic relationships — there are no uncertainties. The model contains only value and decision

variables, and yields a single output for a single set of inputs or setting of each variable. A spreadsheet is an example of a deterministic model.

**Discrete Chance Node:** A node in the Influence Diagram or Decision Tree that represents an event whose outcomes are uncertain. A discrete chance node models the uncertainty of the event with a discrete number of states (or outcomes). Probabilities are specified for each state or a named distribution is given. The graphical symbols for a discrete chance node are bright green ovals in the Influence Diagram and bright green circles in the Decision Tree.

**Discrete Tree Simulation:** Simulation method of evaluating models utilizing random sampling of discrete or discretized probability distributions (chance nodes) with a finite number of outcomes. *See also Monte Carlo.*

**Display Function:** A function that enables you to write text or formatted numbers to the Session Log during an analysis, allowing the writing of custom reports.

**Distributed Sampling:** A method of evaluating models based on Discrete Tree Simulation. In Distributed Sampling, the probabilities of nodes in the tree are represented exactly until the number of samples remaining at a node is small. At this point, normal Discrete Tree Simulation (selecting chance node outcomes randomly) is used for the rest of the tree.

**Distribution:** There are two types of probability distributions in DPL: 1. Chance node data specified as set of discrete states and associated probabilities. 2. Chance node data specified as a named distribution. *See Named Distributions.*

**DLL:** *See Dynamic Link Library.*

**Don't Gamble:** A property assigned to a chance node that effectively reduces it to one branch. DPL replaces a chance event with its expected value if the "Don't Gamble" specification has been set. Subsequent variables depending on the state of this event will be assigned values calculated by taking the expected value over the event, where feasible.

**Downstream Decision:** A decision which occurs in a Decision Tree after initial uncertainties are resolved. Downstream decisions are sometimes called real options.

**DPL Program:** A description of a decision model in DPL's own language (i.e. DPL Code). It contains all the structure and data necessary to analyze a model and can be used in place of an Influence Diagram and Decision Tree, or as documentation for a graphic model.

**Driver Node / Driver Variable:** A node linked to a specific cell in a spreadsheet to provide input for the spreadsheet model. DPL exports data to the cell. Data exported by DPL will overwrite any information contained in the cell.

**Dummy Node:** A value node that represents an intermediate calculation in a model. It contains no data and is ignored during analysis. Dummy nodes are included for clarity or communication only, or to consolidate influence arcs.

**Dynamic Link Library:** A file containing functions, instructions or programs for use by DPL during analysis.

---

## E

---

**Endpoint:** The terminal point of each path through a Decision Tree. The number of endpoints (or leaf nodes) of a Decision Tree equals the number of paths through the tree. Endpoints are represented as blue triangles:

1. Each blue triangle in a Policy Tree is an endpoint.
2. In the Decision Tree because the Decision Tree is schematic, blue triangles can represent more than one endpoint. In the Decision Tree, an endpoint triangle serves as the point of connection when you add a node to the tree.

**Endpoint Database™:** The complete set of endpoints that result from running a decision model. The Risk Profile for the best alternative of the initial decision of a model is created from the subset of endpoints in the Endpoint Database that are part of the optimal policy. Certain additional outputs can be created from the Endpoint Database without the need to re-run the entire model.

**Endpoint Database™, Displaying:** The process of displaying a recorded Endpoint Database in a spreadsheet-like tabular report. Displaying of the Endpoint Database can take a significant amount of memory and may not be possible with very large models. Occasionally with large models, there is enough computer memory available to record, but not to display the Endpoint Database.

**Endpoint Database™, Recording:** The process of saving the Endpoint Database from a model run in order to re-use it later. Recording of the Endpoint Database can take a significant amount of memory and may not be possible on some computers with very large models. Recording the Endpoint Database is most appropriate and useful with models that have complex value models, and therefore have relatively long runtimes per endpoint calculation.

**Erlang Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Event:** A node with a set of possible states (outcomes or alternatives). In DPL, decision nodes and chance nodes are events.

**Event Tornado Diagram:** A sensitivity analysis that compares the range of uncertainty of each chance event in a model.

**Expected Value (EV):** The probability-weighted average of all possible outcomes.

**Exponential Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Export Node / Export Variable:** *See Driver Node / Driver Variable.*

## F

---

**Fast Sequence Evaluation:** The default method of calculation DPL uses for an analysis. This is the fastest evaluation method that calculates the exact expected value. Certain model structures are not suitable for Fast Sequence Evaluation and should instead be analyzed using Enumerate Full Tree.

**Frequency Histogram:** A view option in the Risk Profile Chart. It plots the probability of occurrence for intervals of outcome values as a vertical bar.

**Full Tree Enumeration:** An evaluation method that evaluates each path of the tree completely. This is in contrast to Fast Sequence Evaluation.

---

## G

---

**Gamble:** An evaluation action taken by DPL for a chance event when evaluating a Decision Tree. When DPL gambles on a chance event, it creates the alternative outcomes of the chance event, assigns them a probability and determines the value of the objective function for each outcome.

**Gamma Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Gaussian Distribution:** (Also called Normal Distribution.) A named distribution supported by DPL. For a detailed description see On-Line Help.

**Geometric Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Get/Pay Expressions:** Expressions specified on branches of the Decision Tree to associate values with branches and endpoints. "Get" expressions are added to the value function and "Pay" expressions are subtracted. If a model has multiple attributes, the get/pay expression must contain expressions separated by commas for each attribute.

---

## H

---

**Hide Intermediates:** An option when creating a model from Excel that prevents intermediate cells from being displayed in the Range Names dialog used to select which spreadsheet cells to include in the model. If intermediate are included, the linked nodes will be Dummy Nodes.

**Hyperexponential Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

---

## I

---

**Import Node / Import Variable:** *See Metric Node / Metric Variable.*

**Influence Arc:** An arc drawn from one node to another in an Influence Diagram. An influence arc indicates timing or conditional dependence of nodes. An arc from node A to node B means A influences B.

**Influence Diagram:** a graphical representation of the components of a decision problem — decisions, uncertainties, and values — and the relationships among them. Comprised of nodes and influence arcs.

**Influence Diagram pane:** A pane of the Model Window that provides a graphical interface to manipulate the Influence Diagram in a model.

**Initial Decision:** A decision in a Decision Tree that occurs before any chance events.

**Initial Decision Alternatives Tornado:** A diagram that displays a Base Case Tornado for each alternative of the first decision node in the Decision Tree.

**Initial Uncertainties:** Chance events in a Decision Tree that occur before a downstream decision.

**Interval:** A subset within a Series in which (one or more) elements are given by the same expression. For example, a series defining the number of days in each year from 2013-2016 would have an Interval from 2013–2015 in which each element would have a value of 365. The Interval for 2016 would be a single element Interval with a value of 366.

## J

---

**Joint Probability:** The probability associated with a particular set of outcomes for multiple separate events. The joint probability is calculated by taking the marginal probability for the selected outcome of each event and multiplying them together.

## L

---

**Laplace Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Links:** Connections to spreadsheets, databases or DPL programs that allow DPL to perform calculations or obtain data.

**Local Variables:** Nodes that are not linked to a cell or cells in a spreadsheet; to a record in a database; nor to a DPL program.

**Logistic Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Lognormal Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Lottery:** An event whose outcome depends on chance.

---

## M

---

**Marginal Probability:** The probability that a given outcome of a chance event will occur, absent any other related information. For example, if there is an even chance of sun or rain, the Marginal Probability of rain is 0.5.

**Maximize:** Instructs DPL to choose the decision policy that returns the maximum expected value for the Objective Function.

**Maxwell Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Metric Node / Metric Variable:** A node linked to a specific cell in a spreadsheet that retrieves the output of spreadsheet model calculations for use in DPL analysis. A DPL Metric Node does not contain any data in DPL.

**Minimize:** Instructs DPL to choose the decision policy that returns the minimum expected value for the Objective Function.

**Model Window:** Graphical interface for designing decision models and running analyses. Contains both the Influence Diagram pane and the Decision Tree pane.

**Modified Monte Carlo Simulation:** method of evaluating models based on Discrete Tree Simulation. In Modified Monte Carlo, samples are first allocated to branches of chance events using the branch probabilities rather than completely random sampling. *See also Distributed Sampling.*

**Monte Carlo Simulation:** Method of evaluating models utilizing random sampling of continuous probability distributions (chance nodes) with a set sample size. *See also Discrete Tree Simulation.*

**Multi-attribute Utility Analysis (abbreviated MUA or MAU):** A method for making decisions where the value function depends on more than one factor, such as cost and schedule. DPL allows you to track multiple attributes using independent value expressions and combine them in a single utility function.

**Multidimensional Value Nodes:** Value nodes that refer to one- or two-dimensional arrays or a series in a DPL model.



---

## N

---

**Name:** A string of characters used to identify a node, string or constant. Nodes will be referenced by their names when used as variables.

**Named Distributions:** Pre-defined probability distributions which can be assigned to chance events in DPL based on input for one or more parameters. DPL supports 21 different Named Distributions, such as the Normal Distribution and the Beta Distribution.

**Negative Binomial Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Node:** A graphical representation of a decision, value, uncertainty, strategy table, or controlled event.

**Normal Distribution:** (Also called Gaussian Distribution.) A named distribution supported by DPL. For a detailed description see On-Line Help.

---

## O

---

**Objective Function:** An expression defining the quantity to be optimized during the analysis. Generally used to express the relationship between multiple attributes.

**Optimal Policy:** A set of decision alternatives that optimize the Objective Function.

**Optimization:** Allows DPL to take advantage of special structural properties of the model to reduce computation time.

**Option Value Chart™:** A chart that displays the "option value", or incremental value added by flexibility in each downstream decision in the tree.

**Outcome:** A state of a chance node or chance event that DPL evaluates during rollback by combining together the probability associated with the outcome and the value associated with the outcome to calculate its expected value.

---

## P

---

**Pascal Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Perform Subtree:** A technique allowing you to repeat a section of a Decision Tree in more than one place in the Decision Tree. Allows the subtree to be drawn once and performed several times.

**Poisson Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Policy Summary™:** An output generated by DPL which displays each event and its associated distribution under the optimal policy resulting from the analysis. You can also use the Policy Summary to compare the optimal policy distributions to distributions associated with a range of the Risk Profile or to when a decision or chance is in a particular state.

**Policy Tree™:** A graphical representation of the optimal policy for every decision in a model given the outcomes of all the uncertainties in the model. Shows all possible paths through a Decision Tree and indicates the value of all expressions in the model, the probabilities associated with each chance event outcome, the rollback values (EV or CE) for each node, and the optimal policy choices for each decision.

**Prob Function:** A function that returns the probability of an event at any point in an analysis at which the state of the event is unknown. This function is useful in decision problems in which the probability of an event is itself a major consideration in making decisions.

**Probabilistic Base Case Tornado Diagram:** Used with probabilistic models; a diagram generated by evaluating the outcome of a model when each chance event is individually set to a high and a low state while all other chance events are allowed to vary across their states. The base run that the results are compared to will equal the expected value of the model as in a Risk Profile or a Policy Tree.

**Probabilistic Model:** A model in which some inputs are described with probability distributions and which generates probability distributions as outputs. A probabilistic model in DPL has at least one chance event. If the model has been developed graphically in the Model Window then a probabilistic model will have either continuous or discrete chance nodes or both.

**Program Window:** Provides an interface for creating, editing and compiling DPL program files. Converted spreadsheets can be viewed in this window.

**Promoting Terms:** Moving elements of get/pay expressions to nodes which are encountered earlier in the evaluation. The advantage to this approach is that terms can be calculated in the tree as soon as all conditioning events are in known states, rather than waiting until the end of the path to calculate all the terms at once. In general, promoting terms reduces run time.

---

## R

---

**Rainbow Diagram:** An analysis tool which varies a single value over a user specified range and displays the impact on the expected value or certain equivalent and changes to the optimal decision policy.

**Rayleigh Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Real Option:** *See Downstream Decision.*

**Reduction:** A DPL tool used to reduce an output probability distribution (Risk Profile) to a single chance event. The reduced event will be described by a discrete probability distribution with a user specified number of states. The output in the form of DPL code is written to the Session Log or pasted as a discrete chance node to a specified model.

**Risk Attitude:** Decision-maker's willingness to gamble relative to the expected value. One who is not willing to gamble in the face of a positive expected value is "risk averse," while one who is willing to gamble in the face of a negative expected value is "risk seeking." One who makes decisions based on expected value is "risk neutral."

**Risk Profile:** The probability distribution for a specified policy comprised of the entire range of outcomes that are possible given the policy.

**Risk Profile Chart:** A window displaying a probability distribution for a specified policy. A Risk Profile Chart displaying a cumulative probability distribution of the optimal policy is a typical output of a DPL decision analysis.

**Risk Profile Dataset:** The underlying probability distribution data for a specified policy, which is typically displayed in a Risk Profile Chart. Risk Profile Datasets may be saved in a Workspace without being displayed in a Risk Profile Chart.

**Risk Tolerance Coefficient:** Measure of the decision-maker's attitude towards risk. Common definition for risk tolerance  $r$  is the highest value for which you would accept a gamble in which you could win  $r$  or lose  $r/2$  with equal probability. In DPL, the Risk Tolerance Coefficient describes the incorporation of risk when the built-in exponential utility function is used.

**Roll Forward:** The first phase of a DPL decision analysis run, in which DPL calculates an outcome value and joint probability for each path in the Decision Tree. As each path is traversed, get or pay expressions are evaluated and combined to provide the path outcome value. If the model uses multiple attributes, each endpoint will have multiple outcome values.

**Roll Back:** The third phase of a DPL decision analysis run, in which DPL uses the outcome values and joint probabilities of each endpoint to calculate expected values or expected utilities at each node in the Decision Tree. At each chance node in the Rollback procedure, DPL determines the expected utility/value by calculating the probability-weighted average of outcome values. At each decision node in the Rollback, DPL determines the alternative providing the maximum or minimum value, as appropriate. The aggregate of these optimal decision alternatives comprises the optimal decision policy.

**Rolled-back Expected Value:** The value of the objective function at a node in the Policy Tree given the states of all preceding events on the path leading into the node, and the expected value of all subsequent outcomes of chance events and the optimal selection of alternatives for all subsequent decision events.

**Root Node:** The node on the Decision Tree that does not have any branches leading into it (the first node on the far left of the Decision Tree). Every tree contains a root node.

## S

---

**Sampling:** Approximating a full range of outcomes by evaluating a subset number of paths.

**Scenario:** A path through the tree, from the root to an endpoint, that is a combination of specific decision alternatives and chance event outcomes. This path, a Scenario, represents a single possible state of the world.

**Schematic Diagram:** A type of Decision Tree diagram that does not show a full representation of the tree. Most Decision Trees appear as schematic diagrams in the Model Window.

**Sensitivity Analysis:** Allows you to investigate the impact of a node or group of nodes on the Expected Value and/or Optimal Policy of a model. There are many types of Sensitivity Analyses. *See Rainbow Diagram, Two-Way Rainbow Diagram, Value Tornado Diagram, Base Case Tornado Diagram, Probabilistic Base Case Tornado Diagram, Initial Decision Alternatives Tornado Diagram, and Event Tornado Diagram.*

**Sequence Section:** The DPL code version of the Decision Tree. The Sequence Section tells DPL in what order to evaluate nodes and get/pay expressions.

**Series:** A type of variable that allows you to define a one-dimensional set of values/formula using intervals. Subsequent elements of a series can depend upon earlier elements. Similar to a set of formulas in a row in Excel where column  $n + 1$  depends upon column  $n$ .

**Session Log:** A pane located within the Workspace Window which maintains a record of the DPL session. Error messages and DPL commands performed in the Model Window will be automatically written to the Session Log. Information from an analysis can also be written to the Session Log. *See Display Function.*

**Spreadsheet Model:** A spreadsheet which contains a set of calculations which evaluates one scenario of a decision analysis. DPL links to the spreadsheet model in order to rapidly evaluate a wide range of scenarios based on specified levels of uncertainty.

**Spreadsheet Conversion:** *See Conversion.*

**State:** One of a discrete number of possible settings for an event. The states of a decision event are referred to as alternatives. The states of chance event are referred to as outcomes. Each branch of an event in the Decision Tree represents a state.

**State Function:** A function that returns a numerical value representing the state of an event. It can be used to process and test any event's state.

**Statename Function:** Returns the name of a state of an event as a string ("High", "Low", etc.)

**Strategy:** A single set of alternatives for all of the decisions contained in a Strategy Node.

**Strategy Node:** A node which represents a set of decisions to be made at one point in time. The different alternatives in a strategy node are called strategies. The complete set of strategies in a strategy node is referred to as a strategy table. Each strategy is defined by an alternative for each decision included in the strategy node. The graphical symbol for a strategy node is a yellow rectangle in the Influence Diagram with attached yellow rectangles for each decision node that is contained in the strategy node. In the Decision Tree, the graphical symbol is a yellow square. During evaluation a strategy node is evaluated in the same manner as a decision node, i.e., one alternative is chosen that maximizes (or minimizes) the objective function.

**Strategy Table:** A collection of decision nodes and a set of defined strategies in a strategy node.

**Subscript:** Used to reference particular elements of an array or series. For a series a subscript ranges between the lower and upper boundaries of the series. *For using subscripts with arrays, see Array Subscripts.*

**Subtree:** A section of Decision Tree from a given node in the tree forward (i.e., to the right).

**Subtree Risk Profile™:** A Risk Profile in which the scenarios included in the Risk Profile are restricted to a particular subset of the optimal scenarios, e.g., a particular combination of states of one or more events. A Subtree Risk Profile can be specified from either the Policy Tree or via a dialog.

**Symmetric Node:** A node in a Decision Tree whose branches all lead to the same subsequent event.

**Symmetric Tree:** A Decision Tree in which every node is symmetric.

---

## T

---

**Time Series Percentiles:** A graph that displays the range of outcomes over several time periods (e.g., yearly cash flows over a 10 year period).

**Tolerance:** *See Risk Tolerance Coefficient.*

**Tornado Diagram:** A sensitivity analysis that displays the value and policy impacts of varying input values. *See Value Tornado Diagram, Base Case Tornado Diagram, Probabilistic Base Case Tornado Diagram, Initial Decision Alternatives Tornado Diagram, and Event Tornado Diagram.*

**Triangular Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Two-Way Rainbow Diagram:** An analysis tool which varies two values over user-specified ranges and displays the impact on the expected value, or certain equivalent, and changes to the optimal decision policy.

---

## U

---

**Uncertainty:** *See Chance Nodes.*

**Uniform Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**User Library:** A Dynamic Link Library (DLL) file which contains a set of functions or a program (to be run during a DPL analysis) and language that allows communication with DPL. User Libraries are called from commands in the Decision Tree or program file.

**User Library Function:** An external set of commands which perform a routine or calculation during a DPL analysis. User Library Functions are stored in a User Library, which has a file extension of .DLL.

**Utiles:** The unit of measurement for the transformation performed by the Utility Function. In a model incorporating risk tolerance, DPL selects the policy which maximizes the expected utility (i.e. greatest utile value). The inverse of the utility function converts utiles to certain equivalents.

**Utility Function:** A mathematical expression which captures the decision-maker's Risk Attitude. The Utility Function translates the outcomes of a value model into utiles. In DPL, the built-in exponential utility function is defined as:  $u(x) = -e(-x/r)$ , where  $x$  is the value to be converted to utiles and  $r$  is the risk tolerance coefficient. Alternatively, you may specify a user-defined utility function.

---

## V

---

**Value Correlations Chart:** A chart that displays the correlation coefficient between each value in the model (including values associated with decision and chance events) and the objective function. Value Correlations Charts help you determine which variables may be driving your objective function in either a positive or negative direction.

**Value Model:** A model that defines a particular value of interest (e.g., costs, profits, social welfare) from other data. A DCF (discounted cashflow) valuation spreadsheet is an example of a value model.

**Value Node:** A node in the Influence Diagram representing a number or an expression. Its graphical symbol is a blue rounded rectangle.

**Value of Control:** The difference between the expected value of the model and the expected value of the model when a particular uncertainty is changed into a decision.

**Value of Perfect Information:** The difference between the expected value of the model and the expected value of the model when the outcome of a particular uncertainty is known before a decision is made.

**Value Tornado Diagram:** Evaluates the outcome of a model when a selected set of variables are each set individually to high and low values while all other variables are treated in the way in which they are defined in the base model. If the model is deterministic than all other variables are held at their current value. If the model is probabilistic, then all other variables are allowed to take on their full range. The low and high results for each variable are compared to the results of the full model run (called the Base Result). When used on a deterministic model, the diagram is used to help determine which variables have the greatest effect on the objective function and/or are decision sensitive, if applicable. Those variables with the greatest impact or which are decision sensitive may subsequently be modeled as uncertainties. When used with probabilistic models, Value Tornado Diagrams can only be run on individual states of events or conditioned values. For probabilistic models, Value Tornado Diagrams may not be as useful as Event Tornado Diagrams or Base Case Tornado Diagrams.

## W

---

**Weibull Distribution:** A named distribution supported by DPL. For a detailed description see On-Line Help.

**Workspace Manager:** A pane in DPL, located within the Workspace Window, which lists all documents and stored data associated with the current Workspace. The Workspace Manager can be used to rename or delete items, access saved data or switch to another window.



# Index

Allow event already active ..... 442

Analysis complete dialog..... 137

API ..... 580

- Automation ..... 580
- register DPL as server ..... 581
- type library reference ..... 585
- VBA ..... 581

API data types ..... 591

API methods..... 594

- DPLModel..... 606
- DPLResult ..... 611, 613
- DPLWorkspace..... 601

API objects

- DPLApplication..... 588
- DPLModel..... 588
- DPLNode..... 589
- DPLResult ..... 590
- DPLTreeNode ..... 589
- DPLWorkspace..... 588

API properties

- DPLApplication..... 592
- DPLModel..... 605
- DPLNode..... 607
- DPLResult ..... 610
- DPLTreeNode ..... 609
- DPLWorkspace..... 596

arrays ..... 291

- one-dimensional ..... 291
- two-dimensional ..... 307

asymmetric

- Decision Tree ..... 46
- nodes ..... 46

Asymmetric trees ..... 47

- pruned sequential ..... 441

Attributes, multiple..... See multiple attributes

Base Case Tornado ..... 167

- setup ..... 168

Bayesian revision ..... 236, 243

Branch control ..... 344

Branch definition dialog ..... 265

- blocking branches via ..... 343
- risk tolerance tab ..... 424

Calculation links ..... See spreadsheet links:calculation

case sensitivity ..... 466, 549, 554

certain equivalents

- view toggle ..... 422

Certain equivalents ..... 422

change node types .... See node types, changing

Clear Mem ..... 91

Combine Expert Opinions dialog... 513

Command ribbon ..... 29

- Contextual tabs ..... 38
- Data tab ..... 37
- File tab ..... 30
- Help tab ..... 38
- Home tab ..... 30
- Influence Diagram/Decision tree tabs ..... 30
- split buttons within ..... 32
- tooltips ..... 29
- View tab ..... 37

Command window ..... 496

Compatibility ..... 668

Compile ..... 91

Conditioning

- dialog ..... 69
- probabilistic ..... 68
- value-wise ..... 180
- via arc type..... 146
- via Node Definition ..... 142

constraint function..... 273

continuous chance nodes ..... 213

- named probability distribution.. 213

converting

- conversion options dialog ..... 455
- models to programs..... 466
- revert to linked..... 460
- spreadsheets to programs ..... 452

Create Database Linked Values dialog ..... 575

create model from excel..... 135

data input tree ..... 52

- selection in ..... 142

- separation of ..... 68
- database links ..... 543
  - establishing ..... 560
  - ODBC data source setup ..... 544
  - program driver node ..... 577
  - selecting source for ..... 557
  - syntax in node data ..... 568
  - table configuration ..... 547
- Database Specification dialog ..... 556
- database tables
  - loading schema ..... 559
  - Model ID field ..... 563
  - Node ID structured ..... 550
  - Project ID field ..... 563
  - required fields for ..... 548
  - table/field names ..... 554
- decision analysis
  - output options ..... 87
  - running preliminary ..... 76
- decision nodes ..... 50
  - alternatives ..... 51
  - defining objective for ..... 288
- Decision Tree ..... 43
  - display options ..... 268
  - modeling mode ..... 43
  - reordering within ..... 81
- Decision Tree pane ..... 25
  - indicator for ..... 25
- default state ..... 85
- Default Tree ..... 124, 140, 186
- discrete chance nodes ..... 43
  - default outcome names ..... 48
  - default probabilities ..... 162
  - default values ..... 162
  - distribution type ..... 55
- Discrete tree simulation ..... 88
- Document Navigator ..... 26
- Downstream decision
  - adding ..... 81, 173
  - in Policy Summary ..... 111
- DPL
  - activating ..... 12
  - command ribbon ..... 29
  - installing ..... 12
  - new features ..... 41
  - registering ..... 15
  - support ..... 19
  - training ..... 21
- DPL Code ..... See DPL Program
- DPL Program ..... 463
  - advanced techniques within ..... 493
  - arrays ..... 480
  - chance keyword ..... 472
  - comments ..... 465
  - const keyword ..... 482
  - controlled keyword ..... 475
  - decide keyword ..... 484
  - decision keyword ..... 472
  - default...case ..... 477
  - defining events in ..... 484
  - definition section ..... 464
  - definition section components .. 471
  - dont keyword ..... 489
  - excel clause ..... 472
  - gamble keyword ..... 485
  - get keyword ..... 489
  - given keyword ..... 474
  - if/then/else directive ..... 495
  - include directive ..... 494
  - integer keyword ..... 483
  - options directive ..... 495
  - pay keyword ..... 489
  - perform keyword ..... 488
  - quit keyword ..... 488
  - sequence section ..... 465
  - series ..... 478
  - set keyword ..... 489
  - stop keyword ..... 488
  - string keyword ..... 482
  - use tolerance/utility ..... 490
  - value keyword ..... 475
- DPL variable names ..... 143
- DPL Workspace ..... 22
- driver nodes ..... 73, 128
- duplicating models ..... 160
- Endpoint Database™ ..... 353
  - display ..... 369
  - exporting ..... 378
  - filtering ..... 358
  - importing ..... 382
  - merging ..... 385
  - playing ..... 363
  - reconnecting ..... 374
  - recording ..... 354
  - recording in parallel ..... 388
  - saving ..... 369

sorting .....	362	initialization links	
state names/numbers.....	378	for conditioned nodes .....	646
Estimate Probabilities dialog.....	525	Initialization links.....	See spreadsheet
estimating probabilities.....	522	links:initialization	
evaluation methods.....	87	keyboard shortcuts .....	669
event tornadoes.....	434	learning .....	236, 366
always/don't gamble setting for	435	Macros	
deterministic.....	435	Excel .....	497
probabilistic.....	435	setting up macro node .....	506
Example Navigator .....	17	VBA.....	497
Excel Macros .....	See Macros:Excel	Manage Links dialog .....	222
Expected value of perfect information		changing records in .....	571
/ control.....	115	converting spreadsheet via.....	454
Fast sequence evaluation.....	87, 92	metric nodes .....	73, 128
File Options dialog .....	39	Model Settings dialog.....	32
editing default state names in ....	47	allow event already active setting	
produce simplified metafile setting		.....	442
.....	121	change default outcome grouping	
updating modeling mode in.....	125	within .....	49
fill down.....	322	ignore conditioning setting .....	84
flag values.....	53	Model Window .....	25
Full tree enumeration .....	87	Modeling level options.....	38
Full tree enumeration from endpoints		modeling modes.....	25
.....	365	Monte Carlo simulation.....	199
Get/Pay expressions.....	73	downstream decision .....	231
copy and paste .....	78	initial samples .....	215
defined in Branch Definition ....	265	running .....	214
defined via Get/Pay group .....	83	up-front decision .....	221
display options.....	267	MUA.....	See multiple attributes
using multidimensional values node		multidimensional value nodes .....	291
within .....	348	use in Get/Pay expressions.....	348
imperfect information .....	236	multiple attributes .....	256
influence arcs .....	133	adding attributes .....	260
arrowhead colors .....	254	automatic incrementing of.....	318
bending .....	146	in Get/Pay expressions.....	265
changing types .....	241	Policy Tree™ with .....	269
create via shortcut .....	141	multiple experts .....	512
from formulas.....	134	experience index .....	514
predecessor.....	183	overlap factor.....	514
successor .....	183	ranking.....	515
summary.....	254	weights .....	515
to indicate timing .....	187	multiple objective functions .....	276
Influence Diagram .....	124	naming .....	285
modeling mode.....	124	naming models.....	61, 132
Influence Diagram pane .....	25	Node Definition dialog	
indicator for.....	25	data tab.....	52
Initial Decision Alternatives tornado		general tab .....	51
.....	171	links tab .....	131

- probabilities tab..... 69
- tree instance tab ..... 59
- values tab..... 69
- node types
  - changing ..... 161
- Objective & Utility dialog ..... 263
  - defining multiple objectives in.. 283
- objective function ..... 256
  - defining ..... 262
  - minimizing ..... 278
- objective functions, multiple ..... See multiple objective functions
- Online Help ..... 18
- Option Value Chart™ ..... 413
  - default states in ..... 414
- outcome grouping ..... 49
  - mixed ..... 59
- output metric ..... 44
- Outputs
  - copying to presentation..... 121
  - policy options ..... 91
  - probability distribution options... 89
  - saving within workspace ..... 113
- Parallel Endpoint Recording ..... 388
- perform subtree ..... 65, 245
  - and continue..... 250
  - perform reference ..... 66, 245
  - perform target ..... 66, 245
- policy dependent probability ..... 112
- Policy Summary™ ..... 111, 112
  - advanced features of ..... 404
  - comparison ..... 407
- Policy Tree™ ..... 76, 92
  - advanced features of ..... 393
  - attribute display options in ..... 335
  - attribute expected values in.... 262
  - data elements ..... 99
  - display options ..... 101
  - drilling down within ..... 96
  - expand commands ..... 97
  - filtering..... 396
  - introduction ..... 94
  - multiple objective functions in... 289
- Probabilistic Base Case Tornado... 196
- probability assessment..... 512
- Rainbow Diagrams
  - introduction ..... 153
  - on a probability ..... 193
- one-way..... 189
- policy changes within ..... 191, 432
- policy tips..... 192, 433
- two-way..... 426
- Range Names dialog..... 126
- Real option..... 81
- References
  - Type Library ..... 585
- reserved word ..... 466
- revision tracking ..... 579
  - required fields for..... 550
- Risk Profile ..... 88
  - charts ..... 105
  - comparing multiple..... 217
  - cumulative ..... 107
  - datasets ..... 105
  - for attributes ..... 337
  - formatting ..... 110
  - frequency histogram ..... 108
  - initial decision alternatives 228, 249
  - number of intervals ..... 107
  - reading cumulative..... 107
  - select quantity for ..... 88
  - statistics..... 109, 216
- risk tolerance..... 417
  - by branch..... 425
  - coefficient ..... 421
  - sensitivity analysis on ..... 422
- Run Settings dialog ..... 91
  - editing number of intervals in.. 107, 328
  - editing number of samples in ... 230
- Saving workspace ..... 61
- Select Attribute button ..... 262
- Select Database Link dialog..... 566
- Select Link button ..... 565
- Select Variable button ..... 142
- Selection
  - branches ..... 63
  - color indicator..... 52
  - node ..... 63
- sensitivity analysis
  - comparison of types ..... 438
  - decision sensitive variables ..... 159
  - introduction ..... 153
  - types ..... 426
- series..... 291
  - in DPL..... 299

using relative subscripts within .	304
Session Log .....	23
spreadsheet links	
calculation .....	632
establishing calculation .....	633
establishing initialization .....	641
initialization .....	632
link excel metric command.....	74
link model events dialog .....	71
managing .....	650
overview .....	632
types .....	632
Strategy table .....	652
adding decisions to .....	654
creating strategies .....	660
defining strategies .....	662
subtree .....	65
Subtree Risk Profiles™ .....	400
symmetric	
Decision Tree .....	45
node .....	45
System requirements.....	668
template model.....	556
time series percentiles .....	313
defining attributes for .....	315
defining Get/Pay Expressions for .....	320
defining range bar for.....	331
error bar display .....	330
for initial decision alternatives..	340
setup.....	326
Tornado diagrams	
introduction .....	153
undo .....	365
user function libraries .....	614
callback functions .....	622
code examples .....	627
explicit functions .....	619
implicit functions .....	615
utility function .....	424
Value Correlations .....	119
value nodes .....	124
creating linked .....	126
dimensionality .....	129
in the Decision Tree.....	74
Value of perfect information / control .....	115
Value Tornado.....	153
color changes in .....	157
formatting .....	158
setup.....	154
Workspace Manager .....	22
non-window items .....	106
Workspace Window .....	22
Workspace-level options.....	38
Zoom Controls.....	27