Oracle® Configurator Developer

User's Guide

Release 11*i* for Windows 95/98 and Windows NT 4.0

September 2000

Part No. A73280-06

ORACLE

Oracle Configurator Developer User's Guide, Release 11*i* for Windows 95/98 and Windows NT 4.0

Part No. A73280-06

Authors: Denise Boyer, Tina Brand, Mark Sawtelle, Jan Stetson

Contents

3 Using Oracle Configurator Developer Tools

Send Us Your Comments

Oracle Configurator Developer User's Guide, Release 11*i* for Windows 95/98 and Windows NT 4.0

Part No. A73280-06

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the chapter, section, and page number (if available). You can send comments through your call to Oracle Support Services or by sending them to:

Oracle Configurator
Oracle Corporation
Documentation
21 North Avenue
Burlington, MA 01803
USA

If you would like a reply, please give your name, address, and telephone number below.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

Welcome to the Oracle Configurator Developer User's Guide. This user's guide includes the information you need to work with Oracle Configurator Developer effectively.

Intended Audience

This guide assumes you have a working knowledge of your business processes, tools, and your product configurations. It also assumes you are familiar with product configurator applications. If you have never used configurator applications, we suggest you attend one or more of the Oracle Configurator training classes available through Oracle Education.

Structure

This documentation contains...

- Introduction on page 1-1, which includes information on planning your project to ensure your success in using Oracle Configurator Developer.
- Constructing a Configurator on page 2-1, which includes information on building model structure, configuration rules, and a user interface for an Oracle Configurator window or SellingPoint application.
- Using Oracle Configurator Developer Tools on page 3-1, which includes the vocabulary for understanding instructions for using Oracle Configurator Developer.
- Glossary of Terms on page Glossary of Terms-1
- Glossary of Acronyms on page Glossary of Acronyms-1

Related Documents

For more information, see the following manuals in Release 11*i* of the Oracle Product documentation set:

- *Oracle Configurator and SellingPoint Release Notes*
- *Oracle Configurator and SellingPoint Administration Guide*
- *Oracle Configurator Developer Tutorial*
- *Oracle Configuration Interface Object (CIO) Developer's Guide*
- *Oracle Configurator Custom Web Deployment Guide*
- *Oracle SellingPoint Application Help*
- Oracle Applications Release 11*i* documentation

This document refers you to the *Oracle Configurator and SellingPoint Administration Guide* for additional information on database administration and deployment topics. This is the administration guide appropriate for the fully integrated Release 11*i* Oracle Configurator product. If you are working with Oracle SellingPoint Configurator, refer to *Oracle SellingPoint Configurator Administration Guide.*

Conventions

In examples, an implied carriage return occurs at the end of each line, unless otherwise noted. You must press the Return key at the end of a line of input. The following conventions are also used in this documentation:

Convention	Meaning
. . .	Vertical ellipsis points in an example mean that information not directly related to the example has been omitted.
. . .	Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted
boldface text	Boldface type in text indicates a specific key or screen name.
< >	Angle brackets enclose user-supplied names.
[ ]	Brackets enclose optional clauses from which you can choose one or none.
>	The left bracket alone represents the MS DOS prompt.

1

Introduction

Welcome to Oracle Configurator Developer

Oracle Configurator Developer is the application development tool in the Oracle Configurator family of products. It provides a convenient drag-and-drop interface that enables rapid deployment of your configurator.

What is a Configurator

A configurator is a tool for configuring part or all of your products and services. The configuration process can include assessing customer needs, selecting product and service components and viewing configurations. A configurator enables end users to access the parts that make up your product and the rules that govern how those parts go together. With a configurator, end users can generate any custom product configuration that the rules allow. By building a configurator, you bring the expertise of your enterprise to the point of sale, dramatically changing and improving the way your enterprise sells products and services.

The fundamental elements of a configurator built with Oracle Configurator Developer are:

- a product model that captures the structure of your product
- configuration rules that constrain the relationships between parts of your product.
- a user interface that reflects the model structure

These three elements make up a configuration model for your product. You construct and maintain configuration models in Oracle Configurator Developer. A single configuration model can be deployed to end users in these ways:

- as a stand-alone Oracle SellingPoint application, possibly in a mobile environment
- as an DHTML Oracle Configurator window in iStore, Sales Online, Order Capture, or a custom web application
- as a Java applet Oracle Configurator window in Order Management, TeleSales, or Order Capture

Building a Configurator with Oracle Configurator Developer

You can use Oracle Configurator Developer to build the Oracle Configurator Model and Configuration Rules directly from your product and business requirements. Configurator Developer automatically generates a functioning user interface based on the product model and configuration rules you provide. You can customize the default user interface to have the look and feel that you want for your enterprise.

Oracle Configurator is integrated with Oracle Applications so that an end user can configure a product based on an Oracle Bill of Materials. Oracle Configurator dynamically creates a configuration model that reflects the BOM rules: parent-child, optional or required selections, mutually exclusive selections and quantity cascade rules. If you wish to add additional model structure to the BOM Model, or add additional rules, or customize the user interface, you must do that work in Configurator Developer.

A Project is the workspace where you construct a configuration model. All the data that define your configuration model are stored in real time directly in the Oracle Configurator schema. Any Configurator Developer Project must be associated with a datasource for an Oracle Configurator schema, which is a schema of tables named with the CZ_ prefix standalone or in the Oracle Applications database.

Plan your Project

You need to plan your configurator project carefully before you open a Project and begin work in Oracle Configurator Developer. Here are some points you should consider during planning. For additional warnings and helpful hints, see the *Oracle Configurator and SellingPoint Release Notes.*

- Plan on attending a training course in using Configurator Developer.
- Plan to meet all platform requirements for Configurator Developer and your Oracle runtime configurator as presented in the *Oracle Configurator and SellingPoint ReadMe* on the Configurator Developer CD.

- Plan on setting up your Configurator Developer users on a local area network (LAN). Do not attempt to connect Configurator Developer clients to a database running on a remote server over a wide area network (WAN), T1 connection, or modem due to performance issues. A WAN configuration may be practical if the network bandwidth is high enough. However, in many situations, it will be necessary to run Configurator Developer remotely using a Citrix server collocated with the database server.
- Plan to build your entire configuration model in a single Project. There is no automated procedure for merging several smaller Projects into a single larger one.
- Plan to define your product model first, before defining rules.
- Establish standardized and meaningful naming conventions for Model nodes and Rules. Use meaningful names for Components and Options. Names like **Response 1** and **Response 2** can easily lead to confusion.
- Plan to express your requirements for valid configurations in terms of the rules that Configurator Developer provides.
- Plan to write end user Help for the Oracle runtime configurator and the configuration models you deploy there, if necessary. Help in the Oracle runtime configurator is not context-sensitive. It provides Help for the window, not for individual content items.
- Plan your user interface. If you plan to deploy your configuration model as a DHTML window, design a template to contain the Configurator frameset at runtime; the colors, banners, Done and Cancel buttons, and Help if desired. Sample HTML files for a template that you can customize are installed as part of your Oracle Configurator installation (%ORACLE_HOME%\OSP\WebUI).
- Plan to publish both test and production versions of your projects manually. In addition, note that it is difficult to update a model that is in production. See the *Oracle Configurator and SellingPoint Administration Guide* for a list of tables that must be updated from one version to another.
- Gather the requirements for needed outputs such as quotes, proposals, and order entry data, their format, and the data for populating them. Oracle Configurator provides pre-defined output for quotes, proposals, and Oracle ERP orders.
- Gather the requirements for integrating your system with other systems such as data synchronization and replication, quotes, and orders.

- Determine your requirements for deployment beyond those stated in the *Oracle Configurator and SellingPoint ReadMe* on the Configurator Developer CD, such as networked client / server PCs or mobile laptops.

Multi-User Development Strategy

If you need to have more than one developer working on a Project, consider the following suggestions.

- You can work in numerous Projects on a single Oracle Configurator schema, all sharing the same Item Master. As mentioned in the previous section, this is not a recommended strategy for developing a single configuration model, because merging of the Projects must be done manually.
- To construct your configuration model efficiently, have only one user at a time make changes to the Item Master for your Project. If more than one user is accessing a Project and making changes, they are doing so directly into a database, so last one in prevails.
- Define all of the Model structure first, before building Configuration Rules.
- When the Model structure is complete, coordinate the rule definition activities so different developers are working on different, non-overlapping parts of the structure.
- Design your Configuration Model

You also need to do the basic design work for your configuration model. You need to consider what functionality your end users require. For example, you may need to add a customer needs assessment component. You must also consider what rules you need to build into your configuration model. Ask yourself questions like, What components must be included in a valid configuration? What components are optional? What sub-components are compatible with each other? The design step may include writing a functional specification and other design documents.

Identify your Product Data

Oracle Configurator Developer requires that an Oracle Configurator schema be identified as the datasource for every Project you define. You must populate the Oracle Configurator schema Item Master with enterprise data needed for product configuration. To this end, you need to identify the source of your product data and structure. If your data comes from Oracle BOM, identify which bills to import into Configurator.

If your data comes from Oracle Inventory item data, or from an external data source, you must develop a mechanism for populating the Configurator import tables, and a plan for refreshing the import as required. Your Database Administrator (DBA) may prepare existing enterprise data for import. If property data is imported from Oracle as part of the standard BOM import, properties and their values must be represented as Catalog Descriptive Elements and Descriptive Element Values in Oracle Inventory.

For data that is not imported, the configurator project team manually populates the Oracle Configurator schema Item Master within Configurator Developer.

For more information about preparing and importing data, see the *Oracle SellingPoint Configurator Administration Guide* or your DBA. For information on setting up a datasource for Configurator Developer to connect to, see the ReadMe.pdf on the Oracle Configurator Developer CD.

For information about working with Oracle Configurator Developer Projects, see Managing an Oracle Configurator Developer Project on page 2-2.

Design your Configuration Rules

Your first step toward constructing Configuration Rules is to think about how the products and services you sell go together. Some combinations add up to something you can sell, while others do not. When you build configuration rules in Oracle Configurator Developer, you put that product knowledge into your configurator.

Here is a list of some of the issues you need to consider before building configuration rules.

- determine whether a selection affects another item or selection
- identify any options that are used as default initial selections
- define the rules governing the configuration of product families
- define the relations among product families

Quick Tour of Oracle Configurator Developer

Oracle Configurator Developer consists of the following modules, which address different aspects of constructing and testing configurator functionality.

- Model Module on page 3-16
- Configuration Rules Module on page 3-24

- User Interface Module on page 3-29
- Test/Debug Module on page 3-35

The Model, Configuration Rules, and User Interface modules present you with menus of commands and views of the configuration model you are building. The Test/Debug module launches a selected testing environment.

Getting Help with Oracle Configurator Developer

Help for Oracle Configurator Developer is available from the Help menu in the menu bar and as the *Oracle Configurator Developer User's Guide.*

Product Support

The mission of the Oracle Product Services organization is to help you resolve any issues or questions that you have regarding Oracle Configurator Developer, Oracle Configurator, and Oracle Configurator Internet Edition.

For a complete listing of available Oracle Product Services, see http://www.oracle.com/support/welcome. For support phone numbers, see http://www.oracle.com/support/contact_us.

Oracle metals subscribers should call the number for their level of support. OracleGOLD customers, please call (800)440-4653. OracleSILVER customers, please call (800)223-1711. OracleBRONZE customers, please call: East (407)240-8900, Central (719)635-8900, West (650)506-1500.

2

Constructing a Configurator

The Oracle Runtime Configurator

The Oracle runtime configurator is a generic configurator, as described in What is a Configurator on page 1-1. With Oracle Configurator Developer, you build a Model, Configuration Rules, and User Interface structure that reflect your enterprise and your end users' requirements. The Model, Configuration Rules, and User Interface structure underlying the application are stored in the Oracle Configurator schema and can be replicated to the Oracle Configurator Mobile Database.

The Active Model interprets the data in the Oracle Configurator schema and enforces valid configurations based on end user selections. The Active User Interface interprets the data in the Oracle Configurator schema and keeps the UI state current as the end user works. In other words, when the end user works in the Oracle runtime configurator, the Oracle Configurator schema, the Active Model, and the Active UI determine what is available for selection, what results from selections, and how it is displayed.

Configurator Developer provides a generic structure, and a look and feel for the Oracle runtime configurator that is enforced by the Active Model, Active UI, and Oracle Configurator schema. This generic user interface can be customized, see Customizing the Generic User Interface on page 2-62.

The application you create with Oracle Configurator Developer can be deployed as

- an Oracle SellingPoint application
- an Oracle Configurator window: Dynamic HTML in a browser
- an Oracle Configurator window: Java applet in a browser.

Further information on these deployment options is available in the Oracle Applications Help system. See *Oracle Configurator and SellingPoint Administration Guide* for more information on the mechanics of deployment.

Managing an Oracle Configurator Developer Project

You start constructing your Oracle Configurator configuration model by defining a Project in Oracle Configurator Developer. The Project is the workspace in which you build your configuration model. The Project uses data stored in the Oracle Configurator schema which you have identified as the data source for your Project. For information on how to ensure that the Oracle Configurator schema contains the data you need, see the *Oracle Configurator and SellingPoint Administration Guide* or your DBA.

Each Project corresponds to a configuration model. When you are working in a Project, all changes you make are stored immediately in the Oracle Configurator schema, so you do not need to explicitly save your work.

To save versions of a Project, select File / Save Project As... and give a unique name for each version. If the Project contains imported BOM Model structure, all refresh actions are applied to the original Project only. You must modify the Oracle Configurator schema before you can refresh the copied Project from the original BOM Model. The documentation on Refresh and Update in the *Oracle Configurator and SellingPoint Administration Guide* describes how to make the necessary modifications to refresh a copied Project.

Do not use File / Save Project As... to publish versions such as development, test, and production. Save Project As... saves the Project, not the database. See *Oracle Configurator and SellingPoint Administration Guide* for information on publishing the database.

The Overall Process

There are two approaches to working in Oracle Configurator Developer:

- self-contained mode
- integrated mode

If you are working in the self-contained mode, create the Item Master, and build your Model, Rules, and User Interface entirely within Configurator Developer. You can chose to work this way if you are building a small-scale demo or prototype system. Many real-world configuration models involve working in the integrated mode. You build a configuration model based on product structure and data from an Oracle Bill of Material. End users configure products defined in your Oracle BOM, and pass completed product configurations on to Oracle Order Management. To build such a configuration model, you begin by importing BOM data into the

Oracle Configurator schema. The *Oracle Configurator and SellingPoint Administration Guide* describes the data import process.

All the tools you need to construct the core functionality for an Oracle runtime configurator are available in Oracle Configurator Developer. The database you use during construction is the Oracle Configurator schema. This construction or development version of the Oracle Configurator schema is subsequently replicated into a testing and production version.

The BOM data import process creates an Oracle Configurator Developer Project. Open this Project to begin development. A BOM Model and Item Master are available to you in the Project. You cannot modify the imported BOM Model, but you can add structure to the Product node that is parent to the BOM Model. If you are working in the self-contained mode, you begin development by opening a new Project and creating a Model.

After you have completed your Model structure, you create Configuration Rules that define how the parts of the Model are related to each other. After you have created Configuration Rules, you use Oracle Configurator Developer to generate a User Interface that reflects the structure of your Model. This generated interface enables you to test the functionality of your Model and Configuration Rules in the Test / Debug module, without doing any additional user interface development. When you are satisfied with the performance of Model and Rules, you can customize the generated User Interface to meet your requirements.

These steps are described in more detail in the following sections:

- The Project on page 2-4
- The Item Master on page 2-5
- The Model on page 2-7
- The Configuration Rules on page 2-19
- The User Interface on page 2-57
- Test / Debug on page 2-73

Language Support in Oracle Configurator

The user interface for the Oracle Configurator Developer itself is in the English language only. Oracle Configurator Developer does support Models and User Interfaces in languages other than English, if the computer operating system is installed in the target language. For example, if you run Oracle Configurator Developer and the operating system of the computer Oracle Configurator

Developer is running on is installed in French, you can create Features, Options, Totals, and so on, with names and descriptions in French.

When you build the UI, you can make labels for fields in French, and when the end user runs the configurator, those labels appear in French. If the end user needs to input text in the configurator, the only language available is English.

The Project

When you start Oracle Configurator Developer you log in and select your data source. For details about logging in and setting up the data source, see the `ReadMe.pdf` file on the Oracle Configurator Developer CD and the *Oracle Configurator and SellingPoint Administration Guide*. Configurator Developer then opens the New Project dialog where you can create a new Project or open an existing Project. When you are working with imported BOM, the import process creates a Project which has the same name as the root node of the imported BOM Model. Select and open this Project.

Opening an Existing Project

1. Select the **Existing** tab on the **Open Project** dialog when you start Oracle Configurator Developer. If you are running Configurator Developer, select **Open Project** from the **File** menu.
2. Select the desired Project from the list of the existing Projects and their descriptions or type the name of the desired Project in the **Name** field.
3. Select **OK**.

Creating a New Project

You create a new Project if you are using Oracle Configurator Developer in self-contained mode. You use self-contained mode if you are creating a demo or prototype application or building a configuration model based on imported data in the Item Master, rather than imported BOM structure.

1. Select the **New** tab on the **New Project** dialog when you start Oracle Configurator Developer or select **New Project** from the **File** menu.
2. Type the name you want to give the Project in the **Name** field.

3. Type a description of the Project, such as which product or service the Project is set up for, in the **Description** field.
4. Select **OK**.

The Item Master

The Oracle runtime configurator uses a standard schema for configuration data referred to as the Oracle Configurator schema. The Item Master is a subschema of the Oracle Configurator schema that contains product data. The Item Master consists of Items, which are specific elements of a product, and Item Types, which are logical groupings of items. You can view the Item Master by Item Name or by Item Description, and you can choose whether or not to view items grouped by Item Type. In most development situations, Item Master data is imported.

Oracle Configurator Developer provides a mechanism called Populators to link data items in the Item Master to the Model. Populators allow you to quickly update your Model to reflect changes in product data. See Using Populators on page 2-16.

Importing Data into the Item Master

Product data is typically imported into the Oracle Configurator schema from data sources external to Oracle Configurator. Legacy product data, such as Bills of Material or Pricing information, can be imported into the Item Master. Generally, the data sources are either a specific Oracle Applications database or a non-Oracle Applications database.

Oracle Configurator imports non-Oracle Applications databases by loading ASCII text files into Oracle Configurator Import Tables. In order to load an ASCII file into the Import Tables, your legacy data must be in the correct format to populate the Database. You must create custom scripts to extract legacy data in the format required by the Import Tables. Consult the discussion of Generic Import in the *Oracle Configurator and SellingPoint Administration Guide* for more information on import of data from non-Oracle databases.

When you import data into the Item Master from Oracle Applications, Oracle Configurator provides

- scripts which create Extraction Views to populate the Oracle Configurator Import Tables
- concurrent programs for transferring the data within the Oracle Applications database in which the Oracle Configurator Database CZ_ schema exists.

Once the Oracle Configurator schema is populated with imported data, that data is available to Oracle Configurator Developer and the deployed Oracle runtime configurator. Oracle Configurator Developer automatically populates the Model Tree and the Item Master Structure with the imported BOM data. The part of the Model based on imported data cannot be altered in Oracle Configurator Developer. You can add additional Components, Resources, Totals, and so on to the Model to complete your configuration model requirements.

When you import an Oracle BOM Model, defined Catalogs are imported as Item Types. Catalog Descriptive Elements and Descriptive Element Values are imported as item Properties and Property values. You can also manually create data Items and Item Types in the Item Master for special purposes required by your configuration model. Added elements cannot be transferred back to the BOM schema to update the BOM.

For detailed information on importing data into the Item Master, see *Oracle Configurator and SellingPoint Administration Guide.*

Modifying the Item Master

You can add, edit, or delete Items, Item Types, or Properties associated with Item Types.

Adding a New Item Type

1. Select the top-level node in the Item Master tree.
2. Select **New Item Type** from the **Create** menu or use the right mouse button to select **New Item Type** from the pop-up menu.
3. Enter the Name on page 3-23, Description on page 3-23, and Properties on page 3-23 for the new Item Type.

Adding a New Item

1. Select an Item Type.
2. Select **New Item** from the **Create** menu or use the right mouse button to select **New Item** from the pop-up menu.
3. Enter the Name on page 3-23, Description on page 3-23, and Type / Properties on page 3-23 for the new Item.

Changing the Item Type of an Item

1. Select an Item.
2. Select the Change button in the Type / Properties area of the Attributes View.
3. The resulting dialog box allows you to select a new Item Type for the Item, or to create a new Item Type.
4. Repeat steps 1 through 3 for each Item whose Item Type you want to change. This is the only way to change the Item Type of an Item.

Editing an Item or Item Type

1. Select the Item or Item Type that you want to edit.
2. Edit the information you want to change in the Attributes View. The Item Master is updated with the new information.

Deleting an Item or Item Type

1. Select the Item or Item Type you want to delete.
2. Select **Delete** from the **Edit** menu or use the right mouse button to select **Delete** from the pop-up menu. You can delete an Item Type only if it has no items. If you attempt to delete an Item Type that has Items, you get an error telling you to delete the items first.
3. Select **OK** to confirm the deletion.

The Model

Constructing the Model structure is your first task in creating a configuration model. Importing Oracle BOM information creates an imported BOM Model in a Project. You can create additional structure to extend the Model to include additional aspects of your configuration problem. This added structure does not appear in the Java applet user interface, but can be accessed by launching a DHTML UI from the Java applet. See Oracle Configurator Window: Java Applet on page 2-58.

When you build a Model, you are translating your knowledge about how parts are configured into the products you sell, into terms that Oracle Configurator Developer, and ultimately the Oracle runtime configurator, can use. The Model is organized into a hierarchy called a tree. The object types that the Configurator Developer provides to build your Model are Product, Component, Feature, Option,

Resource, and Total, as well as BOM Model, BOM OptionClass, and BOM StandardItem imported from an Oracle BOM.

Model	The highest level of structure within the Project. A Model can contain Products.
Product	The highest level of configurable structure within the Model. The goal of your sales process is to configure one of these. A Product can contain Products, Components, Features, Resources, and Totals.
Component	A configurable part of a Product. A Component can contain Components, Features, Resources, and Totals.
Feature	A Feature can be a List of Options, an Integer Number, a Decimal Number, a True/False value, or a Text value. A Feature of type List of Options can contain Options.
Option	An Option is part of a List of Options in a Feature. It is often an object represented by an item in the Item Master.
Total	A Total keeps track of a quantity. A Total can have a positive or negative value. Use Numeric Rules to contribute to and consume from Totals.
Resource	A Resource also keeps track of a quantity. The value of a Resource can be positive or 0, but a Resource is violated if its value is negative. Use Numeric Rules to contribute to and consume from Resources.
BOM Model	Equivalent to a Component
BOM OptionClass	Equivalent to a Feature
BOM StandardItem	Equivalent to an Option

Properties

Model nodes can also have Properties. A Property of a node has a specific value that describes an instance of that node, but is not itself a selectable item in a configuration. Examples of common Properties are weight, diameter, and voltage.

The values of Properties are used in Configuration Rules. See Building Comparison Rules on page 2-34 and Building Property-based Compatibilities on page 2-35.

In order to import properties as part of an imported BOM Model, the properties must be defined as Descriptive Elements and Descriptive Element Values in appropriate Catalogs associated with the Oracle BOM. The database setting `ResolvePropertyDataType` controls whether Descriptive Element Values are interpreted as strings, or as numbers. See *Oracle Configurator and SellingPoint Administration Guide* for more information on this database setting.

For those portions of your model constructed directly in Oracle Configurator Developer, you can add, delete, and edit Properties directly on nodes of the Model, or you can add Properties to Items in the Item Master. When you add or delete a Property of a node in the Model, you affect only the selected node. This is in contrast this to Properties in the Item Master, where additions and deletions affect all Items of the selected Item Type. When you use an Item or Item Type in your Item Master to create a node in your Model, for example, by using a Populator, the Model node always reflects all the currently defined Properties and Property values of the originating node in the Item Master. These inherited Properties cannot be deleted or edited in your Model. Properties that you define directly in your Model do not affect your Item Master.

Properties on page 3-21 describes how to add, edit, and delete properties. Manage Properties on page 3-9 describes how to work with properties on a project-wide basis.

The Properties you define in the Item Master or the Model are User Properties. User Properties are unique attributes that you define specifically for your Project. Properties in the Item Master are associated with an Item Type. If you change the Type of an Item the Item loses its old Properties and acquires the Properties associated with the new Type.

Nodes in the Model also have System Properties. System Properties are attributes of the selected node that are specific to certain types of nodes and are provided by Oracle Configurator Developer for you to define. The System Properties that you can define are:

System Property	Node Type
Name	All nodes
Count	Options and Components
Min	Component

System Property	Node Type
Max	Component
Selection	Features, List of Options with maximum = 1, and minimum = 1

The Imported BOM Model

Each BOM Model imported from Oracle Applications corresponds to one Oracle Configurator Developer Project. The default Project name corresponds to the BOM Model name. Only Manufacturing BOM can be imported into the Oracle Configurator schema.

When you import BOM data into the Item Master from specific Oracle tables, Oracle Configurator Developer automatically populates a Model Tree and Item Master Tree with the imported data. The root node in the tree structure is a Product node that is the parent of the BOM Model. Additional Model nodes mirror the imported BOM structure. See *Oracle Configurator and SellingPoint Administration Guide* for further information about importing data from sources external to Oracle Configurator Developer.

You can use Oracle Applications Catalogs to specify Item Types for imported data. The Descriptive Elements and Descriptive Element Values defined in the Catalogs are imported as Properties and Property Values. If no Catalogs are defined, the imported BOM items are all of the Item Master Type Default Type. For search and organizational ease, you can define Item Types in Oracle Configurator Developer for imported BOM items. Be aware that if you change the Item Type of an imported Item, you lose all Property and Property Value information imported with that Item.

The information imported with each node of the BOM Model is:

- **Name** - the name of the BOM item.
- **Description** - a brief description of the BOM item.
- **Definition** - a basic definition of the BOM item including: BOM Item Type, Minimum Quantity, Maximum Quantity, Default Quantity, and a check box that is checked if optional children are mutually exclusive.
- **Properties** - defined as Descriptive Elements in an Oracle Applications Catalog.
- **Property Values** - defined as Descriptive Element Values in an Oracle Applications Catalog.

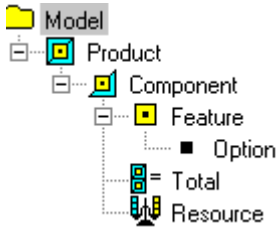
In Configurator Developer, you can add Properties to Item Types in the Item Master, and those Properties appear in the associated nodes in the BOM Model. You

cannot add Properties directly to a Model node of an imported BOM structure. Imported Properties can be modified or deleted from the Item Master, but data modified in Oracle Configurator Developer is lost when data is refreshed unless the CZ_XFR_FIELDS table is updated to exclude property values. See *Oracle Configurator and SellingPoint Administration Guide* for further information on CZ_XFR_FIELDS table.

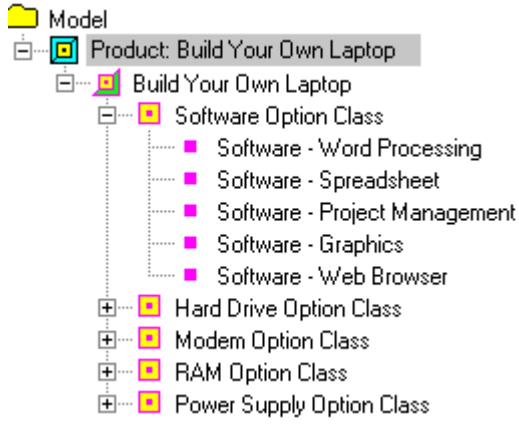
The Model nodes that represent the BOM items cannot be deleted or modified. This is because the BOM structure that is imported must be the same BOM structure that is exported to Oracle Order Management. You can add Model structure, such as an additional node for customer requirements, to the Product node that is the parent of the BOM Model.

You can easily differentiate between the BOM Model nodes and new nodes you add by the colors used in the node icons. BOM item nodes are BOM Model, BOM OptionClass, and BOM StandardItem, and correspond to entity types of the same name in Oracle BOM. In the following illustration, **Build Your Own Laptop** is a BOM Model, and is equivalent to a Component. **Software Option Class** is a BOM OptionClass and is equivalent to a Feature. **Software - Word Processing** is a BOM StandardItem, and is equivalent to an Option.

Manually Created Model



Imported BOM Model



For BOM Models, BOM OptionClasses, and BOM StandardItems the information in the Name, Description, and Definitions areas appears grayed out because these values are imported from the BOM and cannot be changed in the Oracle Configurator Developer.

The Properties action buttons are also grayed out because Properties are imported and cannot be added to the BOM Model in the Oracle Configurator Developer. Properties can be added to Item Types associated with a node.

Assemble to Order Rules

Basic Assemble To Order (ATO) Rules that are inherent in the BOM Model are imported and automatically applied in Oracle Configurator Developer. These rules include Required and Mutually Exclusive options and Quantity Cascade computations.

- **Required** rules apply to child nodes that are required with the parent node. Whenever the parent is selected, the required children are also selected.
- **Mutually Exclusive** (Mutex) rules apply to parent nodes from which you can choose only one out of all optional child nodes; that is, of all non-required children, only one can be selected.
- **Quantity Cascade** calculations determine the final quantity requirements for the selected child node.

The following figure illustrates the relationship between required, optional, and mutually exclusive nodes.

You can create and associate additional rules for all BOM items in the Model, including Resources and Totals.

Each BOM Model and BOM OptionClass corresponds to a screen in the User Interface for the Oracle runtime configurator. The UI displays BOM items in the exact order of their hierarchy in the Model.

Quantity Cascade Calculations

Quantity Cascade calculations determine the final quantity requirements for the selected child node.

When BOM items are imported, they maintain their parent/child relationships. A parent node may have multiple children; some required, some optional, and some mutually exclusive (Mutex). When a parent node is selected, all required children of that parent are selected. When any child node is selected, its parent node is also selected.

Each BOM item is imported with a Minimum Quantity, Maximum Quantity, and Default Quantity. The Minimum Quantity is the smallest number of the selected item allowed per parent. The Maximum Quantity is the largest number of the

selected item allowed per parent. The Default Quantity is the number of the selected item per parent if not modified in the selection process.

Whenever the number of a selected item is greater than zero, a Quantity Cascade calculation is performed which results in the Actual Quantity (or count) for that BOM item. The Quantity Cascade calculation is:

<parent node actual quantity> X <child node default quantity> = <child node actual quantity>

These Quantity Cascade relationships reflect the relationships between components that are built into the Oracle BOM to ensure that the BOM is filled properly. For example, consider the BOM for a car that specifies four wheels and five lug nuts for each wheel. If you select the car, that means you must have four wheels and twenty lug nuts. Similarly, if you select one wheel, that forces selection of the car, which forces the Quantity Cascade calculation, which selects four wheels and twenty lug nuts. The Numeric Rules you build in Configurator Developer respect these Quantity Cascade relationships. If you build a rule that contributes to the count of lug nuts, and the end user uses that rule to select 25 lug nuts, the following things happen. The Oracle runtime configurator determines the minimum number of cars for this number of lug nuts (2 cars) and calculates the number of wheels required (8) and the number of lug nuts needed (40).

All BOM item actual quantities are calculated this way, and are propagated from the root BOM node down through the entire BOM Model.

Building Model Structure

Follow the steps described in the following sections to add Model structure elements to an imported BOM Model, or to build a demo or prototype system. To test your Model, see Test / Debug on page 2-73.

Creating a Product

A Product is a complete object or system that an end user can configure and purchase.

1. Select the top-level Model folder in the Model View.
2. Choose **New Product** from the **Create** menu or use the right mouse button to select **New Product** from the pop-up menu.
3. Type the name you want to give the Product.

4. Enter the following attributes for the Product. Select the arrowhead to the left of each attribute to open and close each section.
   - **Name** - See details on page 3-19.
   - **Description** - See details on page 3-19.
   - **Count** - See details on page 3-21.
   - **Visibility (UI)** - See details on page 3-19.
   - **Properties** - See details on page 3-21.
   - **Populators** - See details on page 3-21.

Creating a Component

A Component is a part of a product that the end user can configure.

1. Select the Product node.
2. Choose **New Component** from the **Create** menu or use the right mouse button to select **New Component** from the pop-up menu.
3. Type the name you want to give the Component.
4. Enter the following attributes for the Component.
   - **Name** - See details on page 3-19.
   - **Description** - See details on page 3-19.
   - **Count** - See details on page 3-21.
   - **Visibility (UI)** - See details on page 3-19.
   - **Properties** - See details on page 3-21.
   - **Populators** - See details on page 3-21.

Creating a Feature

A Feature is a parameter that can be specified to configure a Component.

1. Select the Component node.
2. Choose **New Feature** from the **Create** menu or use the right mouse button to select **New Feature** from the pop-up menu.
3. Type the name you want to give the Feature.

4. Enter the following attributes for the Feature.
   - **Name** - See details on page 3-19.
   - **Description** - See details on page 3-19.
   - **Type** - See details on page 3-19.
   - **Visibility (UI)** - See details on page 3-19
   - **Properties** - See details on page 3-21.
   - **Populators** - See details on page 3-21.

Creating an Option

An Option is a possible enumerated value of a Feature.

1. Select the Feature node.
2. Choose **New Option** from the **Create** menu or use the right mouse button to select **New Option** from the pop-up menu.
3. Type the name you want to give the Option.
4. Enter the following attributes for the Option.
   - **Name** - See details on page 3-19.
   - **Description** - See details on page 3-19.
   - **Visibility (UI)** - See details on page 3-19.
   - **Properties** - See details on page 3-21.

Creating a Total or Resource

You can create Totals and Resources as child nodes of Products and Components. A Total acts as a numeric variable in your Model. A Resource is similar to a Total, but is sensitive to whether its value violates a specified minimum or maximum quantity. A Total can be used as a constant, or set when the end user makes another selection. You can use Configuration Rules to contribute quantities to or consume quantities from both Totals and Resources. See The Configuration Rules on page 2-19 for details about applying Configuration Rules to Totals and Resources.

1. Select the Product or Component node where you want to create the Total.
2. Choose **New Total** or **New Resource** from the **Create** menu or use the right mouse button to select **New Total** or **New Resource** from the pop-up menu.

3. Type the name you want to give the Total or Resource.
4. Enter the following attributes for the Total or Resource.
   - **Name** - See details on page 3-19
   - **Description** - See details on page 3-19
   - **Initial Value** - See details on page 3-22.
   - **Visibility (UI)** - See details on page 3-19.
   - **Properties** - See details on page 3-21.

Using Populators

You can use a Populator to create Products, Components, Features, and Options using Items and Item Types in the Item Master. A Populator creates the new nodes as children of the node on which you define the Populator. If you use Populators to build Model structure from Items, any properties and property values associated with the Items are incorporated into the Model. You can RePopulate the Model with current data when data in the Item Master changes by selecting Tools / RePopulate.

You can also drag and drop Items and Item Types from the Item Master Context Tree View to populate a Product node with Components. Model structure built in this way has no connection to the data in the Item Master, and cannot be updated when that data changes.

The Define Populator Dialog

You use the Define Populator dialog box to create a Populator. This dialog box provides several fields to enable you to enter the information needed to define a Populator. The **Preview** button in the lower left of the dialog box shows you the data that the Populator as defined selects from the Item Master. Use this preview to verify that the Populator is selecting the data you intend before you actually add structure to your Model.

The Create field:

The field in the upper left corner of the dialog box is labeled **Create**. Use it to specify the type of Model node the Populator creates. Your choices of nodes to create depend on the node on which you are defining the Populator. The following table summarized the available choices.

Selected Model Node	Nodes Populator Can Define
Product	Products, Components, or Features
Component	Components or Features
Feature	Options

The from field:

The field immediately to the right of the Create field is labeled **from**. Use it to specify the type of Item Master data the Populator uses to create the specified nodes. The same types of Item Master data are available for all types of nodes a Populator can create. They are:

- Item Type
- Item
- Property
- Property Value

The Where fields:

The remaining fields in the dialog box enable you to enter selection criteria to specify exactly what Item Master data the Populator uses. The selection criteria you can specify are determined by the choice of type of Item Master data you made in the **from** field. The types of criteria available to you are listed in the field labeled **Where Item Type**, **Where Item**, or **Where Property**. You specify specific criteria in the field in the lower right of the dialog box. You either type text into the field, or select from a list of available options, depending on the type of criteria you have selected.

If you select Property Value in the **from** field, you must specify criteria differently. The Define Populator Dialog box displays two fields. The first is labeled: **Where Item Type is**. Select the button to the right of the field to select from a list of Item Types. The second field is labeled: **Where Property is**. Select the button to the right of the field to select from a list of Properties.

Type of Item Master Data	Available Criteria	How you Specify Criteria
Item Type	begins with ends with contains matches	Type text into the field. Item Types are selected depending on whether the Item Type name begins with, ends with, contains, or matches the text you provide.
Item	is of Type	Select button to the right of the field, then select from list of Item Types
	is a Child of	
	begins with ends with contains matches	Type text into the field. Item Types are selected depending on whether the Item Type name begins with, ends with, contains, or matches the text you provide.
Property	is Property of	Select button to the right of the field, then select from list of Item Types
	begins with ends with contains matches	Type text into the field. Item Types are selected depending on whether the Item Type name begins with, ends with, contains, or matches the text you provide.
Property Value	Where Item Type is	Select button to the right of the field, then select from list of Item Types
	And Property is	Select button to the right of the field, then select from list of Propertiess

Creating a Populator

Follow these steps to add, edit, and delete Populators. You can cut and paste a node with populators, but if you copy a node with populators, the populators are lost in the copy.

1. Select the Product, Component, or Feature node you want to populate.

2. Select the arrowhead to the left of the Populators section to open it. This section lists existing Populators and provides buttons to **Add**, **Edit**, or **Delete** Populators.
3. Select the **Add** or **Edit** button to open the Define Populator dialog box. Add creates a new Populator, **Edit** modifies the selected Populator. Select the **Delete** button to delete the selected Populator.
4. Select the type of node you want to create from the **Create** dropdown list.
5. Select the type of Item Master data the Populator should use from the **from** dropdown list.
6. Specify the selection criteria the Populator should use. Choices available to you and the format for your selection depends on the selection you made in the **from** list.
7. Select the **Preview** button to verify that the Populator retrieves the data that you expect.
8. Select **OK** to accept and run the new or modified Populator. The Populator creates elements for the selected Product, Component, or Feature automatically using the data from the Item Master.

The Configuration Rules

One of the most critical activities in constructing your configuration model is to design and construct the rules that govern what the end user can select to make a valid configuration. You need to identify the rules that express relations among the Products, Components, Features, Options, BOM OptionClasses, and BOM StandardItems of your Model.

When you defined the requirements for your configurator, you defined the rules for a valid configuration. You now need to determine how you can most effectively and efficiently apply these rules, using the kinds of Configuration Rules provided by Oracle Configurator Developer. For each type of Configuration Rule you identify Model elements that are:

- used as general defaults.
- used as defaults when another option is selected.
- automatically selected when a user selects another option.
- permitted when a user selects another option.
- excluded when a user selects another option.

Oracle Configurator Developer provides a convenient interface that allows you to drag nodes from your model and drop them into the appropriate parts of the rule structure. Many rules can be built through this drag-and-drop interface. When you need to build more complex Configuration Rules, you can use the Advanced Expression capability. See Advanced Expressions on page 2-44 for further details. Dates can be represented in rules as Decimal Features, entering the date information in the format YYYYMMDD, for example: 19991027

Configuration Rules and Logic State

In order to enforce the rules you build, the Oracle runtime configurator maintains a logic state for Items the user can select. An Item's logic state can be true, false, or unknown. A logic state of true means that the Item is included in the configuration, a logic state of false means that the Item is excluded from the configuration, and a logic state of unknown means that no decision has been made about the Item. Within true and false, a logic state can be true because the user has explicitly selected this Item, or because the Oracle runtime configurator has set the logic state to true in response to a rule triggered by another selection the user has made. In the same way, a false logic state can result from explicit user selection or from the action of rules.

When the end user creates a new configuration, all logic states are initially unknown, unless you have defined default values. The exception is Features that have a minimum Number of Selections of one or greater, which have an initial logic state of true. The Oracle runtime configurator User Interface keeps track of all changes to an Item's logic state caused by the user's selections and the rules triggered by the user's selections.

Logical Relationships

You can express constraints among elements of your Model in terms of logical relationships. For example, one product Feature may require or imply that another Feature or Option be included in the product configuration. Oracle Configurator Developer provides the following logical relationships:

- Requires on page 2-21
- Implies on page 2-22
- Excludes on page 2-22
- Negates on page 2-23
- Defaults on page 2-23

The following five sections describe these relationships and present tables illustrating their action.

In each row of the table, the un-shaded portion indicates the logic state chosen by the end user, and a shaded portion indicates the logical state that results from the selection. The arrows indicate the direction in which the logic state is propagated. Notice that logic state can propagate both from the A side to the B side of the relation, and from the B side to the A side. Notice also that for some values and some logic relations, there is no propagation of logic state.

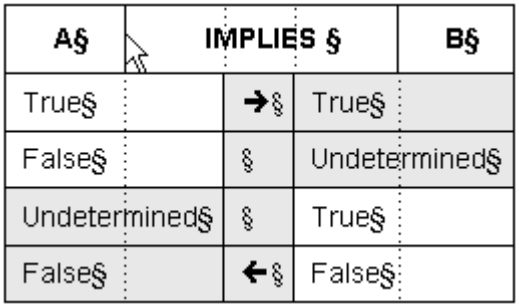
The text following the table describes the effects of applying that type of relationship.

Requires

A§	REQUIRES §	B§
True§	→§	True§
False§	→§	False§
True§	←§	True§
False§	←§	False§

The effect of the Requires relation: If you set either Item to true or false, the corresponding Item in this relation is also set to the same state (true or false). The two are forced to be the same.

Implies



A§	IMPLIES §	B§
True§	→§	True§
False§	§	Undetermined§
Undetermined§	§	True§
False§	←§	False§

The effect of the Implies relation: If you set A to true, then B is set to true as well; conversely, if you set B to false, then A is also set to false.

Compare this diagram to the corresponding one for the Requires relation. In the Implies relation, note that there are no “pushing” arrows when a result is undetermined.

Excludes

A§	EXCLUDES§	B§
True§	→§	False§
False§	§	Undetermined§
False§	←§	True§
Undetermined§	§	False§

The effect of the Excludes relation: If you set an Item to true, the corresponding Item in this relation becomes false, since it is not allowed. If you set an Item to false, it has no effect on the other Item.

Negates

A§	NEGATES§	B§
True§	→§	False§
False§	→§	True§
False§	←§	True§
True§	←§	False§

The effect of the Negates relation: If you set an Item to true or false, the corresponding Item in this relation is immediately set to the opposite. The two are forced to be opposed.

Defaults

The effect of the Defaults relation: If you set the A side of the relation to true, the B side is set to true. Unlike other logic relations, the logic state of the B side is not enforced. You can set the B side to true or false regardless of the state of the A side. The relations only action is to set B to true when A is set to true, if B is not true already. You can think of Defaults as defining additional suggested selections if the user makes a certain selection.

The Defaults relation can be used to set up an initial configuration, for example, by triggering a set of Defaults relations with a boolean Feature whose initial value is true.

Summary of Logical Relationships

Compare and contrast the effects of the logical relationships as shown in the following table:

A	REQUIRES	B
True	→	True
False	→	False
True	←	True
False	←	False

A	IMPLIES	B
True	→	True
False		Undetermined
Undetermined		True
False	←	False

A	NEGATES	B
True	→	False
False	→	True
False	←	True
True	←	False

A	EXCLUDES	B
True	→	False
False		Undetermined
False	←	True
Undetermined		False

All True and Any True

The preceding sections describes logic relations in terms of true and false values for A and B sides of the relation. You can build logic relations where the A or B side involves more than one term, for example, all the Options of a Feature. In this case, you must specify whether that side of the relation is true if ALL of the terms are true, or if that side is true if ANY of the terms are true.

- If you want the side of the relation to evaluate to true only if all terms are true, select All True in the Definition section for the rule, or use the AllTrue function in the Advanced Expression editor. This is a logical AND expression.
- If you want the side of the relation to evaluate to true if any term is true, select Any True in the Definition section for the rule, or use the AnyTrue function in the Advanced Expression editor. This is a logical OR expression.

Types of Configuration Rules

The types of Configuration Rules provided are summarized in the following table:

Rule type	Description
Logic Rule	Define logical relationships between Features, Options, BOM OptionClasses, and BOM StandardItems using a *logic relations*. The logic relations available are: Requires, Implies, Excludes, Negates, and Defaults.
Numeric Rule	Perform a numeric operation on one or more numeric Features, Option counts, or Totals placing the result in a numeric Feature, Option count, Total, or Resource. The rules available are: Contributes to and Consumes from, each with a variety of ways in which to specify the numeric operation.
Comparison Rule	Perform a comparison between the value of a Property of one or more Features and the value of another such Property, or some constant value.
Property-based Compatibility	Specify matches between the Options of one or more Features that have a common Property.
Explicit Compatibility	Specify matches between the Options of one or more Features in explicit tabular form.
Design Chart	Specify compatibility matches between one Primary Feature's Options and multiple Secondary Feature's Options in explicit tabular form.
Functional Companion	Write code to perform pre-selection or validation functions that go beyond Oracle Configurator Developer's supplied functionality.

To create Configuration Rules for an element of your Model you:

- determine the type of rule you want to create
- build the rule expression
- generate the rule logic
- test the rule. See Test / Debug on page 2-73.

Rule Folders

You can create folders in the Configuration Rules tree to organize your Configuration Rules. When you create a rule, Oracle Configurator Developer places a node for the rule in the folder you have selected. You can create as many rule folders as you need to organize the rules in a meaningful manner. Rule folders may

contain multiple types of rules. You can use standard drag-and-drop or cut, copy, and paste operations to move or copy a rule from one folder to another. The same rule cannot reside in more than one folder. If you copy a rule to a different folder, you have created a new, separate rule that can be modified independent of the original.

Enable and Disable Rules

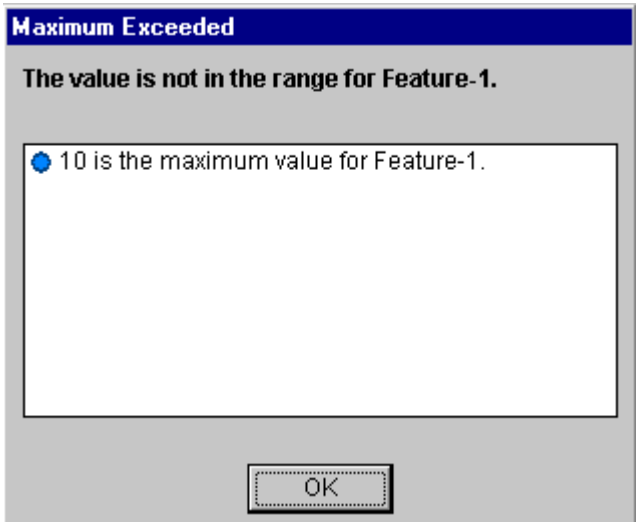
All rules except Functional Companions can be enabled or disabled. The Attributes View for each rule contains a check box labeled **disable** to the right of the rule name. Select this box to disable the rule. A check mark appears to indicate that the rule is disabled. Select the box again to enable the rule.

You can also disable and enable all the rules in a selected folder by selecting the appropriate button in the Attributes View for the folder. Disable and enable at the folder level are merely convenient methods to change the setting for a group of rules. You can still enable and disable individual rules within the folder.

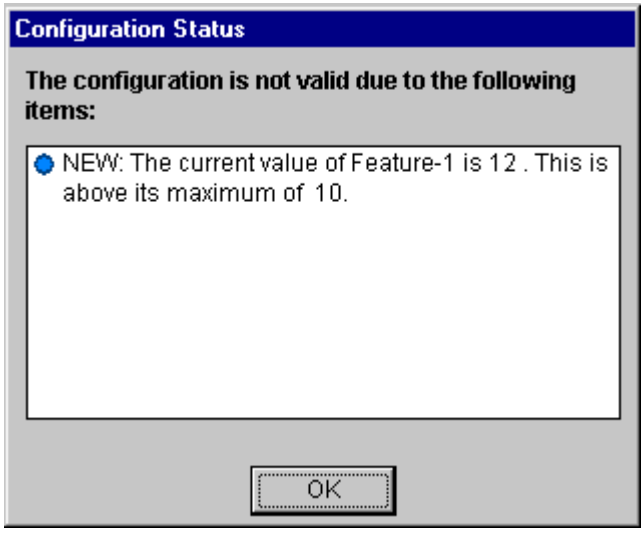
Enabling and disabling rules can be a useful tool in testing and debugging your configuration model.

Enforcing Logical Relationships

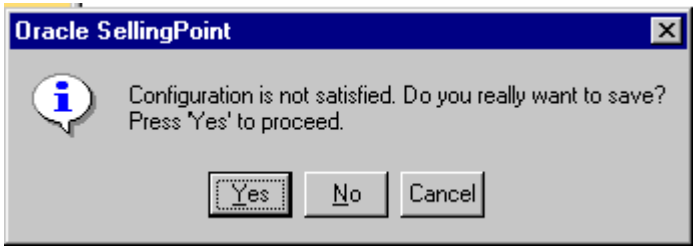
The rules you create in Oracle Configurator Developer shape choices the end user can make in the deployed configuration model. If the end user makes a selection that violates a rule, the configuration model responds with various errors and messages. For example, if the end user attempts to set a numeric Feature to a value that exceeds its defined maximum, the system responds with a message such as the following one, and the value of the Feature is not changed.



On the other hand, the numeric rules Contributes to and Consumes from are handled differently. If the user selects values that cause a numeric Feature, Option count, Total, or Resource to exceed its defined maximum or minimum values, the system responds with a message such as the following. As the text of the message indicates, the value remains invalid.

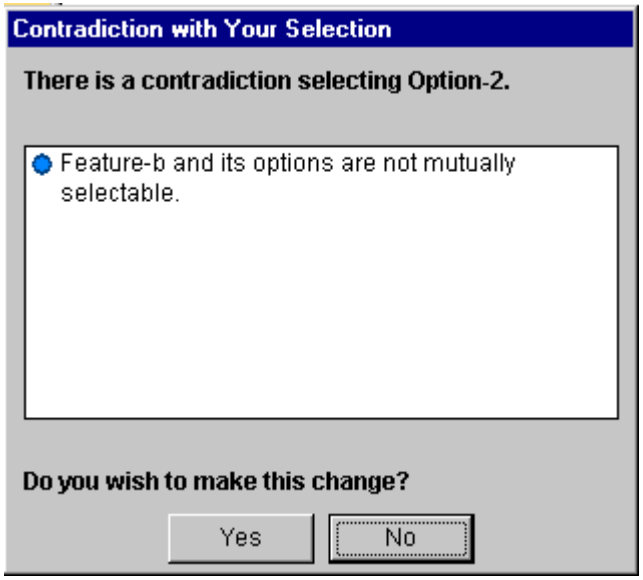


If the end user attempts to save the configuration in this state, or any other state that violates the rules you have established, the configuration model responds with a message such as the following.



The end user can save the invalid configuration.

If the end user attempts to make a change to the configuration that violates other logic rules, the systems responds with a message such as the following.



The user must either retract the change just attempted, or insist on the change. If the end user makes the change, the system changes values of other Features, Options etc. until the configuration is restored to a valid state.

Building Configuration Rules

Follow these general steps to build any of the available Configuration Rules. For detailed information on building a specific type of rule, see:

- Building Logic Rules
- Building Numeric Rules
- Building Comparison Rules
- Building Property-based Compatibilities
- Building Explicit Compatibilities
- Building Design Charts
- Building Functional Companions

1. Select the **Rules** button on the main toolbar. The Configuration Rules tree appears in the lower-left pane.

2. Select a node in the Configuration Rules tree.
3. Open the **Create** menu and select the **New** command for the type of rule you want to create. You can also highlight the Configuration Rules node in the Context Tree View, select the right mouse button, and select **New** command from the pop-up menu.
4. Type a name for the Rule.
5. Select the arrowhead to the left of the **Description** section to open it and type a short explanation of the rule.
6. The **Parameters** and **Definition** sections are specific to each type of rule. See the steps for each type of rule for details.

   Whatever rule you are building, if you position the cursor over a Feature or Option name in the rule definition, the full path to that Feature or Option appears as a tooltip. This information helps you avoid confusion, especially if you use the same Feature or Option names in different parts of your Model. The Effects list functionality is not currently implemented.
7. Select the arrowhead to the left of the **Violation Message** section to open it. Select the rule name, the rule description, or type a customized message that you want to appear in the Oracle runtime configurator User Interface if this rule is violated by a user.
8. Select **Generate Active Model** from the **Tools** menu.
9. Test your new rule. See Test / Debug on page 2-73 for details about testing your Configuration Rules.

Building Logic Rules

Logic rules establish a logical relationship between one part of your Model structure and another. You can use them to control the selections the end user makes.

To Build Logic Rules

See Building Configuration Rules for a general description of creating Configuration Rules.

1. Select the arrowhead to the left of the **Definition** section to open it.
2. Drag and drop the specific Feature, Option, BOM OptionClass, or BOM StandardItem nodes from the Model View to the rule element A and B lists.

- To apply the rule to a Feature, Option, BOM OptionClass, or BOM StandardItem, select the node with the **left** mouse button and hold down the button while you drag the node to the rule element list.
- To apply the rule to all of the Options of a Feature or BOM OptionClass, select the node with the **right** mouse button and hold down the button while you drag the node to the rule element list. Release the right mouse button. Select `OptionsOf(nodename)` from the pop-up menu.
- Indicate whether any of or all of the elements in the element lists must be true for the rule by selecting **Any True** or **All True**. See All True and Any True on page 2-24.

3. Indicate whether the result of rule element A Requires, Implies, Excludes, Negates, or Defaults rule element B.

Building Numeric Rules

Numeric Rules express constraints between elements of your Model in terms of numeric relationships. With Numeric Rules, the selection a user makes can contribute to or consume from a Resource, Total, numeric Feature, or Option count.

Contributes to

A **Contributes to** rule specifies addition of a numeric value to a specified numeric Feature, Option count, Total, or Resource. Calculation of the numeric value can involve Constants, Boolean values, numeric Features, Option counts, Option properties, or Totals.

Consumes from

A **Consumes from** rule specifies subtraction of a numeric value to a specified numeric Feature, Option count, Total, or Resource. Calculation of the numeric value can involve Constants, Boolean values, numeric Features, Option counts, Option properties, or Totals.

Negative Contributions

Negative contributions include both a Consumes from rule with a positive value, and a Contributes to rule with a negative value. If the sum of contributions is a negative value, it is ignored. If the sum of contributions is positive, the result is

contributed in the ordinary way. For example, if we have a counted option with three rules contributing 5, 4, and -3, the value contributed is 6.

Contributions to a BOM Model are handled similarly. For example, you have a BOM Model with a count of 3, and a child BOM OptionClass with a count of two, and three rules contributing 5, 4, and -3, to the BOM OptionClass. The contribution is calculated:

```
   5    +    4   +   (-3)   =      6
(rule1)    (rule2)   (rule3)    (contributions)
```

The final count of the BOM OptionClass is calculated:

```
Floor(      6     /     3     )     *     3     =     6
   (contributions) (parent count)    (parent count)
```

The sum of contributions (6) is exactly divisible by the parent count (3), so the final result is 6.

If the sum of contributions is not exactly divisible by the parent count, the result becomes the closest smaller exact multiple of the parent count. For example, if the contributions are 5, 4, and -4, the result is 3.

If the sum of contributions is smaller than the parent count, then the result becomes the default count. For example if the rule contributions are 5, 4, and -7, the result is 6 (which comes from 2 * 3).

If the sum of contributions is a negative value, it is ignored.

Unknown Values and Rule Propagation

When the end user starts a new configuration in the deployed Oracle runtime configurator, many elements in the Model structure have a logic state of unknown until the user makes a selection. There are several situations in which unknown values in rule element A of a Numeric Rule cause the rule not to propagate, so the rule has no effect.

If all of the participants in rule element A are unknown, the rule does not propagate.

If the expression in rule element A involves the operators '+' or '-', the rule propagates when the end user makes a selection so that one participant in rule element A is no longer unknown.

If the expression in rule element A involves the operators '*' or '/', the rule propagates if one participant is not unknown, and has a value of 0, or if all participants are not unknown.

To Build Numeric Rules

See Building Configuration Rules for a general description of creating Configuration Rules.

1. Select the arrowhead to the left of the **Definition** section to open it.
2. The A side of the rule consists of a comparison operator and two operands. Select an operator from a dropdown list. The operator can be one of:
   - * Multiplication
   - */ Division

   The operands can be one of:
   - **Boolean Expression:** Indicate whether any or all of the Features or Options in the list must be true for the rule by selecting **Any True** or **All True**. See All True and Any True on page 2-24. To apply the rule to a Feature, select the Feature from the Model View with the **left** mouse button and hold down the button while you drag it to the rule element list. To apply the rule to an Option, expand the Feature, select the specific Option with the **left** button while you drag it to the rule element list. To apply the rule to all of the Options of a Feature, select the Feature with the **right** mouse button and hold down the button while you drag it to the rule element list. Release the right mouse button. Select the Feature name or the Options of the Feature from the pop-up menu that appears.
   - **Option Count:** Select the specific Option with the **left** button while you drag it to the rule element list.
   - **Constant:** Type a positive numeric constant.
   - **Total or Numeric Feature:** Select the specific Total or Numeric Feature with the **left** button while you drag it to the rule element list.
3. Indicate whether the result of rule element A Contributes to or Consumes from rule element B.
4. The B side of the rule consists of a Total, Resource, Numeric Feature, Option count, or BOM StandardItem count. Drag and drop the node from the Model View to the rule element B. Only one node can be selected for numeric rule element B.

Building Comparison Rules

Use Comparison Rules to compare two numeric values to produce a logical result. Two numeric values are compared to determine if the first is greater than, greater than or equal to, less than, less than or equal to, equal to, or not equal to the second. The result of this comparison expresses constraints between elements of your Model in terms of logical relations. For example, the numeric value of one product Feature or Option compared to the numeric value of another, may require or imply that another Feature or Option be included in the product configuration.

To Build Comparison Rules

See Building Configuration Rules for a general description of creating Configuration Rules.

1. Select the arrowhead to the left of the **Definition** section to open it.
2. The A side of the rule consists of a comparison operator and two operands. Select an operator from a dropdown list. The operator can be one of:

   - <> not equal
   - = equal
   - > greater than
   - **<** less than
   - < less than or equal to
   - > greater than or equal to

   Select the type of operand from a dropdown list located above the operand field. The operands can be one of:

   - **Constant:** Type a positive numeric constant in the operand field.
   - **Total or Numeric Feature:** Select a Total or Numeric Feature with the **left** button and drag it to the operand field. Only one Total or Numeric Feature can be selected for each operand field.
   - **Option Count:** Select an Option or BOM StandardItem with the **left** button and drag it to the operand field. Only one Option Count can be selected for each operand field.
   - **Option Property:** Select an Option or BOM StandardItem with the **left** button and drag it to the operand field. Select a property from the dropdown list that appears below the operand field.

3. Indicate whether the result of rule element A Requires, Implies, Excludes, or Negates rule element B.
4. Drag and drop the specific Feature, Option, BOM OptionClass or BOM StandardItem nodes from the Model View to the rule element B.

   Indicate whether to apply **Any True** or **All True** to the Features or Options in this list. See All True and Any True on page 2-24.

Building Property-based Compatibilities

AProperty-based Compatibility rule specifies valid combinations of Options or BOM StandardItems based on property values. Many types of Model structure nodes can have Properties, but only Features of type List of Options, and BOM OptionClasses can participate in a Property-based Compatibility. Configurator Developer does not allow you to specify a Feature of a type other than List of Options as a participant. You can change the type of a participating Feature after you have built the rule, which results in an error that Configurator Developer detects when you generate the Active Model.

A Property-based Compatibility rule:

- enumerates the Features that participate in the rule
- specifies the Properties to be checked for compatible values
- specifies the type of comparison to be made between the Property values, using standard numeric and string comparison operators, such as equals, greater than, contains, and matches

A rule can include multiple comparisons among the participating Features.

To Build Property-based Compatibility Rules

See Building Configuration Rules for a general description of creating Configuration Rules.

1. Select the arrowhead to the left of the **Parameters** section to open it. From the Model View, select the Features or BOM OptionClasses that you want to include in this rule with the **left** mouse button and hold down the button while you drag it to the Participants list.
2. Select the arrowhead to the left of the **Definition** section to open it.
3. The A side of the rule consists of a comparison operator and two operands. Select an operator from a dropdown list. The operator can be one of:

- <> not equal
- = equal
- > greater than
- < less than
- < less than or equal to
- > greater than or equal to
- contains
- does not contain
- begins with
- does not begin with
- ends with
- does not end with
- matches

4. The left-hand operand is an option property. From the first dropdown list, select the Feature or BOM OptionClass for which you want to compare a Property. From the dropdown list below it, select the Property you want to compare.
5. The right-hand operand is an option property or constant value. From the first dropdown list, select the Feature or BOM OptionClass for which you want to compare a Property, or (value). In the next field, type in a constant or select a property from the dropdown list.
6. Select **And** or **Or** and follow the preceding steps to add another property comparison to the rule.

Building Explicit Compatibilities

Use an Explicit Compatibility rule to express constraints among elements of your Model involving compatibility of Options that cannot be described in terms of a shared Property.

An Explicit Compatibility rule allows you to simply specify, in tabular form, explicit matches between the Options of one or more Features.

To Build Explicit Compatibility Rules

See Building Configuration Rules for a general description of creating Configuration Rules.

1. Select the arrowhead to the left of the **Parameters** section to open it. From the Model View, select the Features that you want to include in this rule with the **left** mouse button and hold down the button while you drag it to the Participants list.

1. Select the arrowhead to the left of the **Definition** section to open it.

   The table in this section contains a column for each of the Features you selected as Participants. You can use the cells of the column to specify Options of the Feature. Select the cell to reveal the dropdown list button. Use this button to select an Option from the Options list for the Feature.

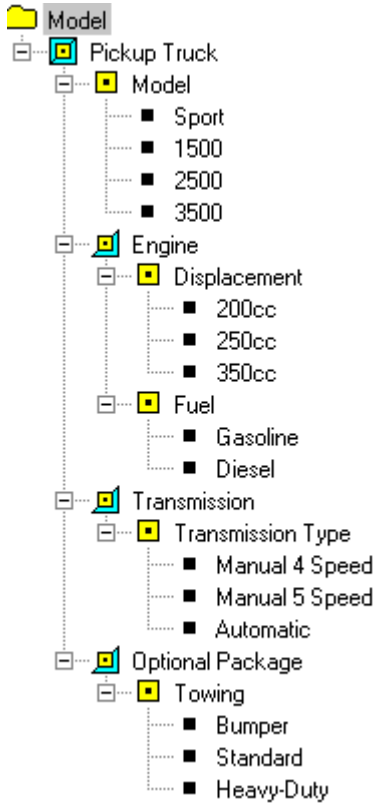
   There cannot be any blank cells in the Explicit Compatibilities table, so each column must have the same number of rows. The same Option can appear in the column as many times as necessary to build the compatibility table you need.

Building Design Charts

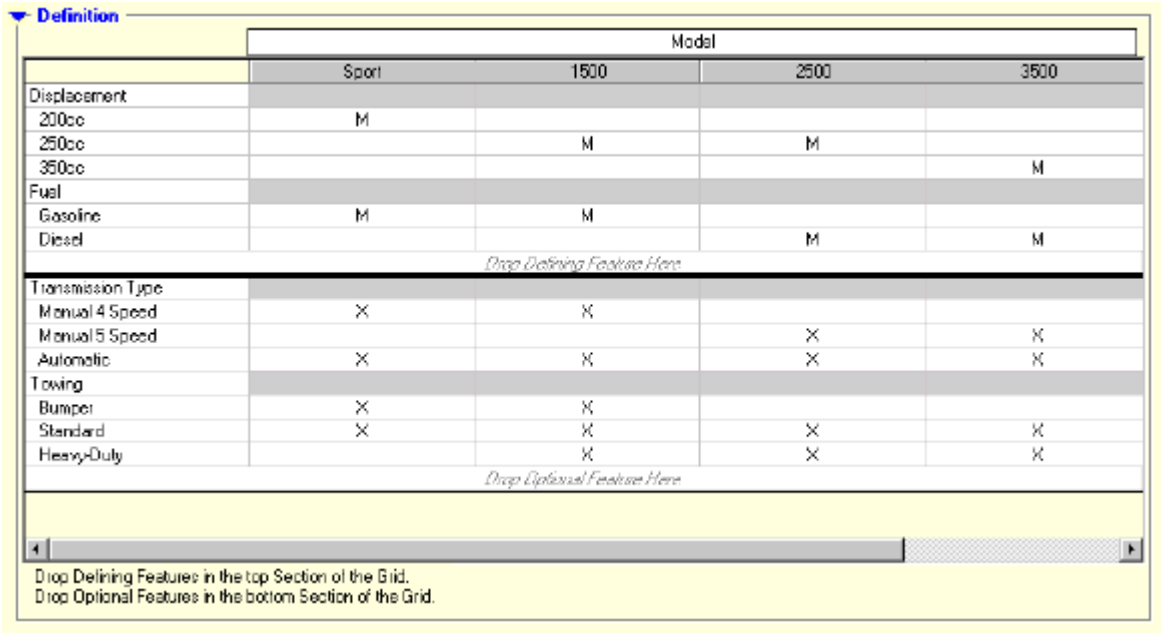
A Design Chart is a means of expressing the complex explicit compatibility relationships that are often used in industry. It enables you to graphically specify the relationship between a Model's *Primary* and *Secondary* Features.

A Primary Feature is a Feature with Options that define the variations of the product. The compatibilities defined in the Design Chart are based on this Feature. A Secondary Feature can be either a *Defining* or *Optional* Feature. A Defining Feature is a is Feature with Options that participate in the unique combinations that define the options of the Primary Feature. An Optional Feature is a Feature whose Options can be arbitrarily compatible or incompatible with the Options of the Primary Feature.

For example, a hypothetical automobile company designs and manufactures several models of sport and full-size pickup trucks. Part of the Oracle Configurator Developer Model structure looks like this:



The Design Chart looks like this:



Definition

	Model			
	Sport	1500	2500	3500
Displacement				
200cc	M			
250cc		M	M	
350cc				M
Fuel				
Gasoline	M	M		
Diesel			M	M
Drop Defining Feature Here				
Transmission Type				
Manual 4 Speed	X	X		
Manual 5 Speed			X	X
Automatic	X	X	X	X
Towing				
Bumper	X	X		
Standard	X	X	X	X
Heavy-Duty		X	X	X
Drop Optional Feature Here				

Drop Defining Features in the top Section of the Grid.
Drop Optional Features in the bottom Section of the Grid.

Model is the Primary Feature. Its options represent the available truck models. Each truck model is defined by the engine displacement and fuel used. Therefore, **Displacement** and **Fuel** are the Defining Secondary Features. Each Option of the Primary Feature is defined by a unique combination of Defining Secondary Features.

Only one Defining Feature Option can be compatible with a given Primary Feature Option. When multiple Defining Feature Options are specified, as in this example, the combination of Options must be unique for each Primary Feature Option. Each column contains a unique combination of Options, but a row can contain more than one marked cell. Note how the Options specified for Displacement and Fuel are unique for each truck model.

Various combinations of transmission type and towing package are available for each truck model, so these become Optional Secondary Features. When the user chooses a Sport model truck, it must have a 200cc gasoline engine, but it can have either a manual 4 speed or automatic transmission.

Features used in a Design Chart must be of type List of Options, and Number of Selections must be Maximum = 0 or 1, and Minimum = 1.

Defining a Design Chart in Configurator Developer sets up a network of relationships in the Oracle runtime configurator so that if the user selects any of the Options included in the Design Chart, that selection can affect the logic state of other Options in the Design Chart.

The following tables illustrate some of these effects.

Scenario 1: User selects Sport model, minimum selections = 0:

Feature	Option	Logic State
Model	Sport	User True
	1500	Logic False
	2500	Logic False
	3500	Logic False
Displacement	200cc	Available
	250cc	Logic False
	350cc	Logic False
Fuel	diesel	Logic False
	gasoline	Available
Transmission Type	Manual 4 Speed	Available
	Manual 5 Speed	Logic False
	Automatic	Available
Towing	Bumper	Available
	Standard	Available
	Heavy-Duty	Logic False

In this example, the end user selects a Sport model truck. This selection makes the logic state of the Option 'Sport' true by user selection. Because the maximum Number of Selections on this Feature is 1, the logic state of the other truck models becomes logic false. Throughout the rest of the model, those Options that are not defined as compatible by the Design Chart become logic false. Those Options that are defined as compatible remain Available. This scenario assumes the minimum

Number of Selections on all Options is 0. The next table shows the effect of a minimum Number of Selections on all Options of 1.

Scenario 1: User selects Sport model, minimum selections = 1:

Feature	Option	Logic State
Model	Sport	User True
	1500	Logic False
	2500	Logic False
	3500	Logic False
Displacement	200cc	Logic True
	250cc	Logic False
	350cc	Logic False
Fuel	diesel	Logic False
	gasoline	Logic True
Transmission Type	Manual 4 Speed	Available
	Manual 5 Speed	Logic False
	Automatic	Available
Towing	Bumper	Available
	Standard	Available
	Heavy-Duty	Logic False

In this case, if Design Chart constraints leave only one available Option, that option becomes logic true.

This example shows how selection of a Defining Secondary Feature constrains the Primary Feature. The end user selects a 250cc engine, which limits truck model choices to 1500 or 2500.

Scenario 1: User selects 250cc engine:

Feature	Option	Logic State
Model	Sport	Logic False
	1500	Available
	2500	Available
	3500	Logic False
Displacement	200cc	Logic False
	250cc	User True
	350cc	Logic False
Fuel	diesel	Available
	gasoline	Available
Transmission Type	Manual 4 Speed	Available
	Manual 5 Speed	Available
	Automatic	Available
Towing	Bumper	Available
	Standard	Available
	Heavy-Duty	Available

To Build Design Charts

See Building Configuration Rules for a general description of creating Configuration Rules.

1. Select the arrowhead to the left of the **Definition** section to open it. The table in this section displays a column for each of the Primary Feature's Options. The rows of each column are where you select the compatible Defining or Optional Secondary Feature.

   – Select the Feature or BOM OptionClass node from the Model View that is to be the **Primary Feature** and drag it to the table heading. The table is automatically populated with a column for each Option of the Primary Feature. A Primary Feature must be defined before the table is saved and you can activate the compatibility cells.

- Select a Feature node from the Model View that is to be a **Defining Feature** and drag and drop it in the Defining Feature section of the table. A table row is automatically populated for each of the Defining Feature's Options. There is no limitation on the number of Defining Features you can include.
- Select a Feature node form the Model View that is to be an **Optional Feature** and drag and drop it in the Optional Feature section of the table. A table row is automatically populated for each of the Optional Feature's Options. There is no limitation on the number of Optional Features you can include.
- If you drag a node using the left mouse button, the Feature and its Options drop wherever you position the mouse in the table. If you drag a node using the right mouse button, a menu displays for you to select the destination of the Feature and its Options.
- Once you have populated the table with all of the Defining and Optional Features, select the table cell to select the **Defining Feature Option** that is compatible with each of the Primary Feature Options. A mark appears in the cell to indicate that you have selected it. The default mark for the Defining Feature Options is an M to indicate that they are mandatory. Continue selecting appropriate table cells for all compatible Options.

  You can customize the cell activation marks to be more representative of your company by modifying the `[Design Chart]` section of your `spx.ini` file. You must specify two different cell marks, one for the Defining Feature Options and one for the Optional Feature Options. For example, if you want to set 'A' for the Defining Feature Option and 'B' for the Optional Feature Options the spx.ini file would look like this:

  ```
  [Design Chart]
  DEF=A
  SEC=B
  ```

  See *Oracle Configurator and SellingPoint Administration Guide* for more information on the `spx.ini` file.
- Select the table cell to select the **Optional Feature Option** that is compatible with each of the Primary Feature Options. A mark is placed in the cell to indicate it has been selected. The default mark for the Optional Feature Options is an X. Continue this for all compatible Options. An Optional Feature Option can be compatible with multiple Primary Feature Options and multiple Options of a given Optional Feature can be compatible with a given Primary Feature Option.

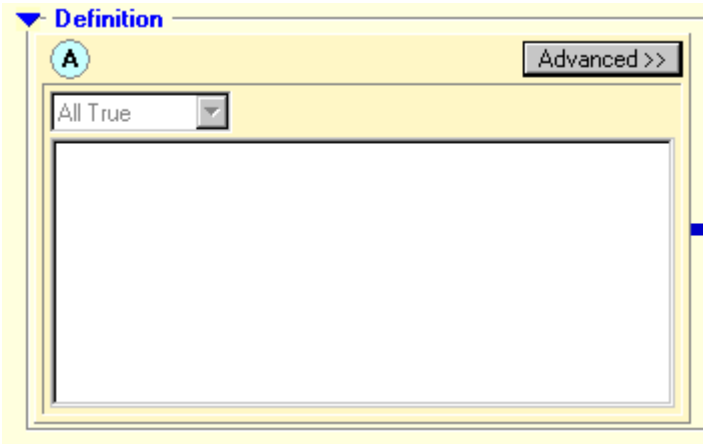
If you select a cell and want to unselect it, you can do so by using the mouse, or positioning the cursor in the cell and pressing the Spacebar, Delete key, or Backspace key.

If you want to remove a Secondary Feature (Defining or Optional) from the table, position the cursor on the Feature's name and press the Delete key or use the right mouse button to select Delete from the menu. You *cannot* delete Options from the Design Chart. When you delete a Feature, all its Options are deleted as well.

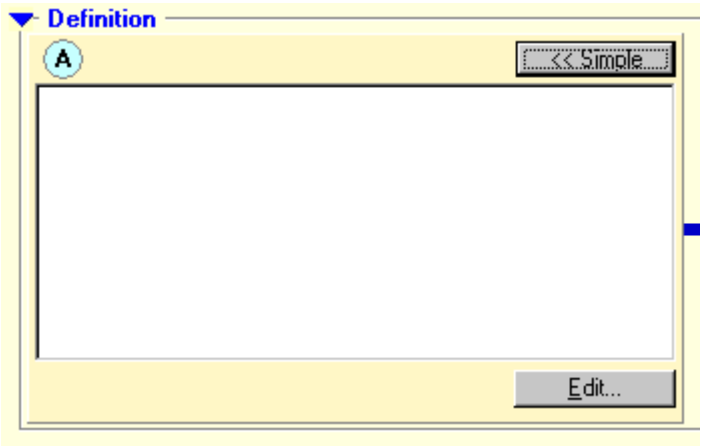
Advanced Expressions

Advanced Expressions enable you to incorporate complex expressions into your Configuration Rules. You can define Advanced Expressions for all rules except Explicit Compatibility rules, Functional Companions, and Design Charts.

You build Advanced Expressions in the Advanced Expression Editor, which you access by selecting the **Advanced** button in the Definition section of the Attributes View.

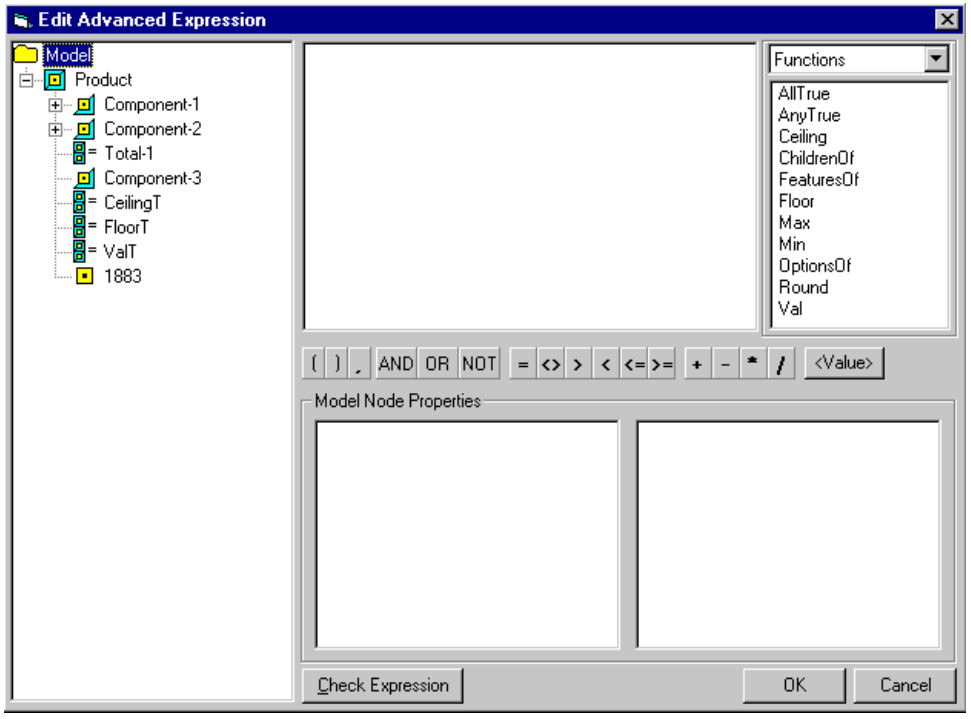


Then select the **Edit** button to open the Advanced Expression editor. Select the **Simple** button to leave Advanced Expression mode.



The Advanced Expression Editor

The Advanced Expression editor provides a drag-and-drop interface for building Advanced Expressions. The interface looks like this.

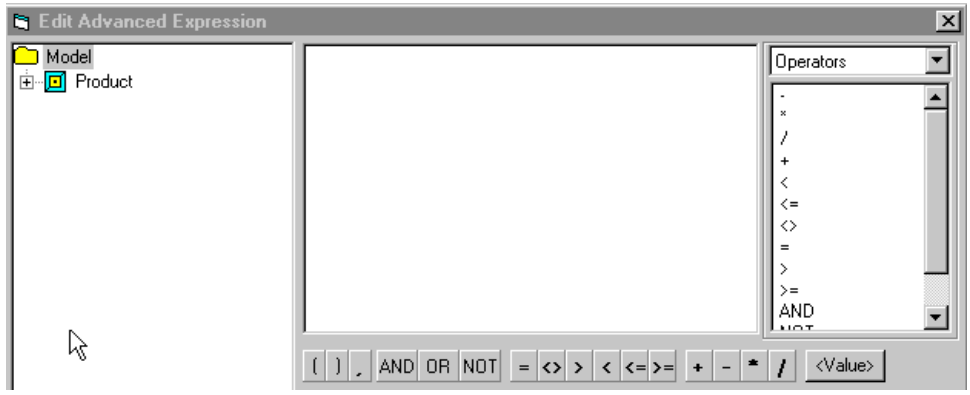


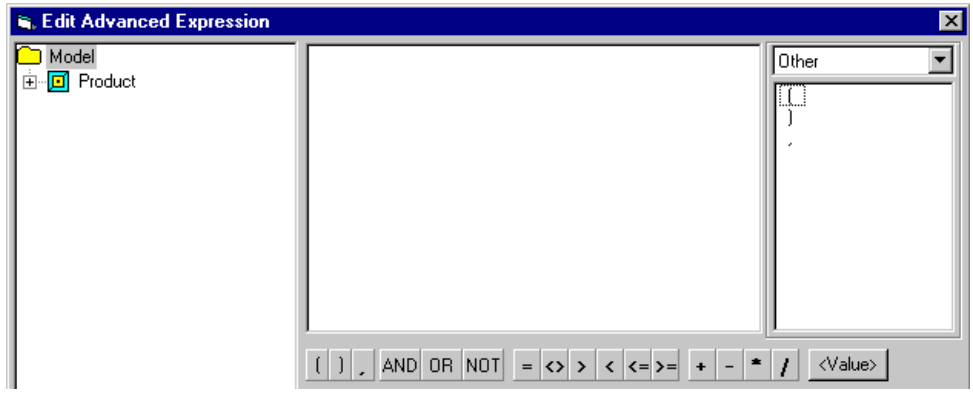
The Advanced Expression editor window is divided into several main areas.

- The Model pane in the left portion of the window.
- The Expression pane in the upper center area of the window.
- The Operators pane to the right of the Expression pane.
- The Model Node Properties area in the bottom right area of the window.

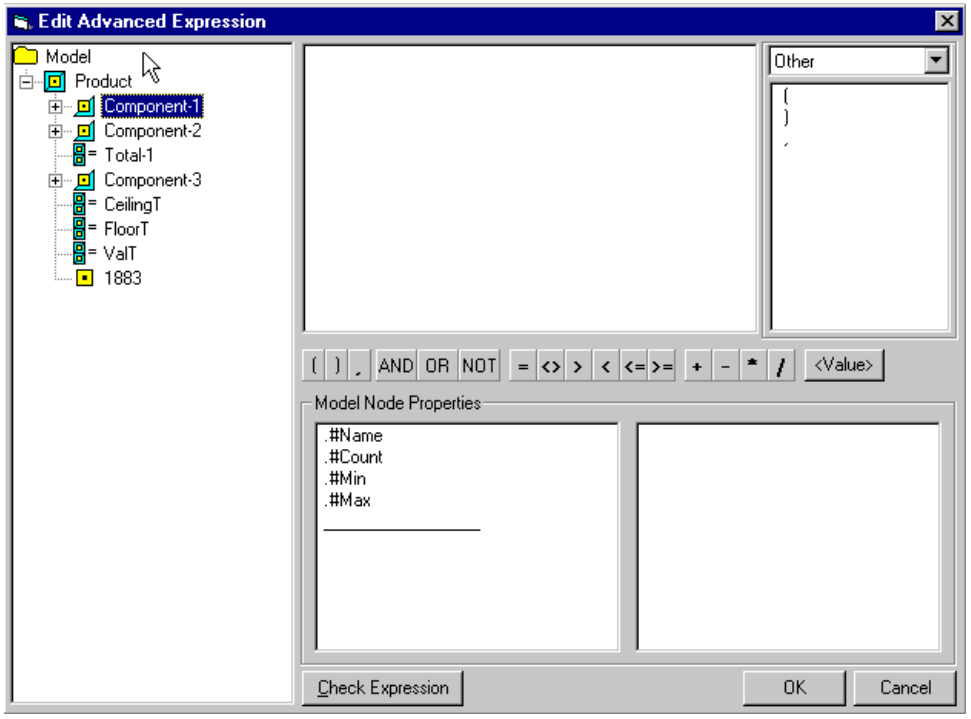
You build Advanced Expressions in the Expression pane by dragging Expression building blocks from other panes of the editing window and dropping them into the Expression pane. If you need to type a specific value into the expression, you must select the **<Value>** button, which opens a small input field in the Expression pane. You can also select and delete text from the Expression pane.

The Operators pane displays **functions**, **operators**, or **other** grouping and separator characters, depending on your selection from the dropdown list. The operators and other characters are also displayed as buttons immediately below the Expression pane. Select one of these buttons to add the character to the expression.

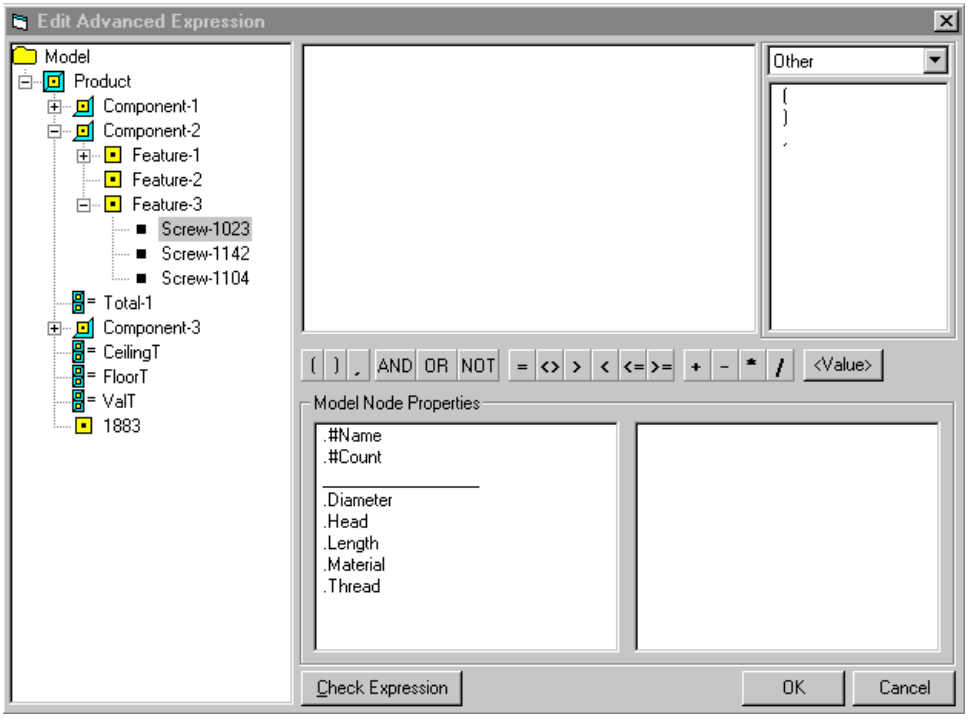




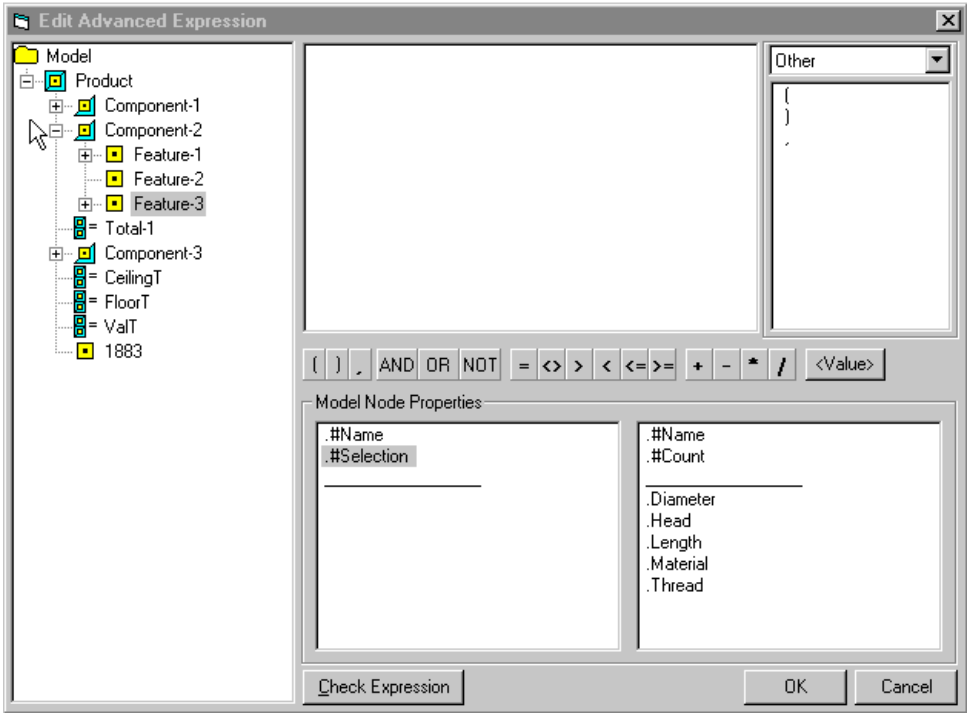
The Model pane and the Model Node Properties area together display the Model structure and Properties of nodes in the Model. The Model structure is displayed in the left-hand pane. Properties for the selected Model structure node are displayed in the center pane. System Properties, such as Name and Count, display above the dividing line with a pound sign (#) in front of them to differentiate them from User Properties. User Properties display below the line.



In this example, Component-1 has no user properties. You see only the system properties Name, Count, Min, and Max.

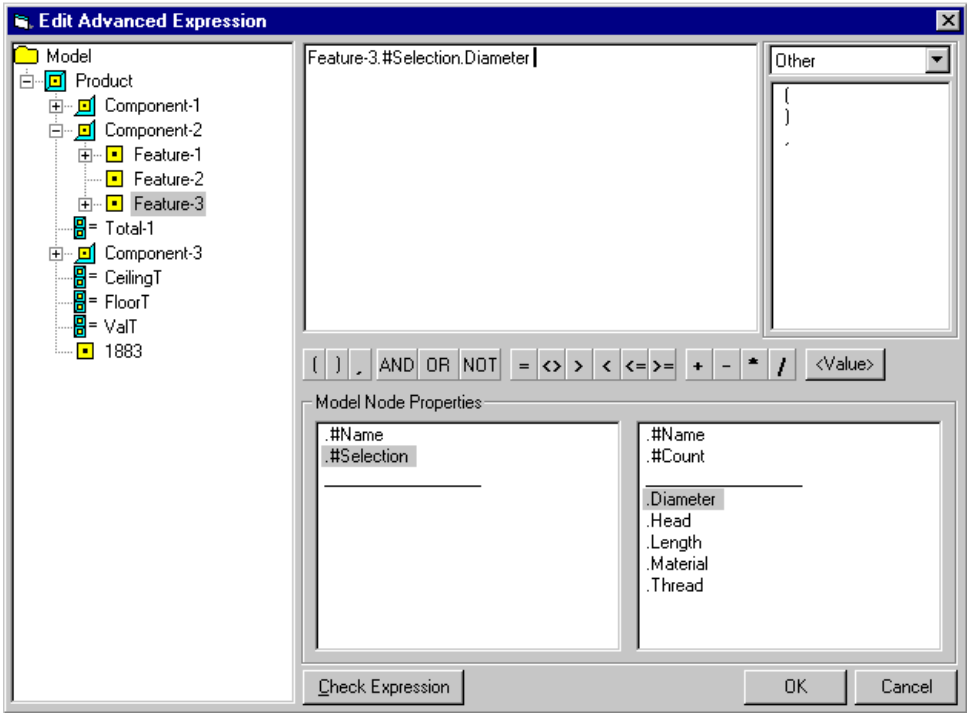


In this example, Screw1023 has the user properties Diameter, Head, Length, Material, and Thread as well as the system properties Name, Count, Min, and Max.



This example illustrates Features of type List of Options that have one required selection (maximum = 1 and minimum = 1). Such Features have a System Property called **Selection**. If you select that Property, the lower-right screen area displays Properties of options of the selected Feature.

If you select a Property from the right-hand pane and drag it into the Expression pane, it appears as illustrated in the following figure.



When the Advanced Expression is evaluated in the end-user Oracle runtime configurator, this Property evaluates to the option Property of whatever option the end user has selected for this Feature. In this example, the value is the diameter of whatever screw the end user has selected.

Building Advanced Expressions

Advanced Expressions are made up of operators and their operands, and functions and their arguments.

Operators

There are three types of Operators: logical (true/false), arithmetic, and comparison.

Operator Type	Operators	Description
Logical	AND OR NOT	AND requires two operands and returns true if both are true. OR requires two operands and returns true if either is true. NOT requires one operand and returns its opposite value: false if the operand is true, true if the operand is false.
Arithmetic	* / - +	Perform ordinary arithmetic operations on numeric operands.
Comparison	<> = > < ≤ ≥	Perform the comparison operations of not equal (<>), equal (=), less than (<), greater than (>), less than or equal (≤), greater than or equal (≥). Possible comparisons include comparisons between a numeric valued Feature and a property of a selected Option, and comparisons between a Property value and the name of an Option.
Other	( ) , . -	parenthesis '( )' are used to group sub-expressions comma ',' is used to separate function arguments dot '.' is used for referencing objects in the Model tree structure unary minus '-' is used to make positive values negative and negative values positive.

Precedence of Operators

Operators are processed in the order given in the following list. Operators with equal precedence are evaluated left to right.

1. Functions, operations within parenthesis ( ) and the dot (.) operator (for example, Function.Selection)
2. Unary minus (-10)
3. Multiplication (*) and division (/)
4. Addition (+) and subtraction (-)
5. Logical NOT
6. Equal to (=) and not equal to (<>)

7. Greater than (>), less than (<), greater than or equal to (≥), and less than or equal to (≤)
8. Logical AND, and OR

Operands

There are three types of Operands: logical (true / false), numeric, or complex. Logical Operands are sometimes referred to as Boolean Expressions. Operands can be Component, Feature, or Option nodes from your Model, or they can be literal values entered after selecting the Value button.

Logical Operands can be Features, Options, or literal values.

Numeric Operands can be Option counts, Totals, Resources, integer, or decimal Features, or constant values.

Complex Operands can be Components, or Features with Options.

Functions

Functions perform operations on their arguments and return values which are used in evaluating the entire Advanced Expression. Functions must have their arguments enclosed in parentheses and separated by commas if there is more than one argument. Function arguments can be expressions.

For example,

```
Round (13.4)
Round (Feature-1 / Feature-2)
```

are both the correct syntax for the Round function, provided that Feature-1 and Feature-2 are numeric Features.

Logical Functions:

Function	Description
AllTrue	A logical AND expression. Accepts one or more logical values and returns true of all of the arguments are true. See All True and Any True on page 2-24
AnyTrue	A logical OR expression. Accepts one or more logical values and returns true of any of the arguments are true. See All True and Any True on page 2-24

Arithmetic Functions:

Function	Description
Ceiling	Takes a single decimal number as an argument, and returns the next higher integer.
Floor	Takes a single decimal number as an argument, and returns the next lower integer.
Round	Takes a single decimal number as an argument, nearest higher or lower integer.
Min	Returns the smallest of its numeric arguments.
Max	Returns the largest of its numeric arguments.
Val	Takes a single text argument and returns it converted to a number.

Compound Functions:

Function	Description
OptionsOf	Takes a Feature as an argument and returns its Options.
FeaturesOf	Takes a Component as an argument and returns its Features.
ChildrenOf	Takes a BOM Model, BOM OptionClass, or BOM StandardItem as an argument and returns its children.

Advanced Expression Errors

You can check if your Advanced Expression is structured correctly by selecting the **Check Expression** button.

You may receive one of the following types of errors:

- Type Mismatch — the Advanced Expression does not evaluate to a datatype allowed in the portion of the rule you are building. For example, an expression that evaluates to a number is not valid in a logic rule.
- Type Mismatch — the argument of a function is the wrong data type. For example, `AllTrue(10)`.
- Type Mismatch — the operand of an operator is the wrong data type. For example, `1 AND 2`.

- Expected: Operand — the expression does not contain the correct number of operands. For example, `3 + .`
- Expected: Operator — your expression does not contain the correct number of operators. For example 3 3 is invalid because there is no Operator between the two numbers. 3 * 3 is a valid statement.

To Build Advanced Expressions for Rules

1. Select the **Advanced** button.
2. Select the **Edit** button to edit the expression.

   The **Edit Advanced Expression** editor appears. See The Advanced Expression Editor on page 2-45.
3. Drag and drop the specific operand Item(s) from the Model tree to the rule expression.

   You can also select the corresponding arithmetic, comparison, or logical Operator(s) buttons to add to the expression wherever you place the cursor. Select the **Value** button to add a numeric literal to the expression.
4. Select the **Check Expression** button to validate the expression.
5. Select **OK**.
6. Select **Generate Active Model** from the **Tools** menu.
7. Test your new rule. See Test / Debug on page 2-73 for details about testing your Configuration Rules.

Building Functional Companions

Functional Companions extend your Oracle runtime configurator by attaching custom code through established interfaces.

To enable your Functional Companion to work with your configuration model, you must associate it with a node in your Model. You create this association in Oracle Configurator Developer, as a type of Configuration Rule.

Functional Companions on your model work in any Oracle runtime configurator. In the Java applet Oracle runtime configurator, they must function without any end user action through the User Interface, because there are no visible buttons to launch Functional Companions. A DHTML UI launched from the Java applet can

use visible buttons to launch Functional Companions. See Oracle Configurator Window: Java Applet on page 2-58.

For information on building Functional Companions, see the *Oracle Configuration Interface Object (CIO) Developer's Guide.*

Rules that Relate Optional Components

There are some restrictions you must keep in mind if you construct rules that relate optional Components. To understand these restrictions, you must understand the following terms:

- An **optional Component** has a minimum count of zero and a maximum count of one or greater than one, or a minimum count of one and a maximum count greater than one.
- A **required Component** has a minimum count of one and a maximum count of one.
- The **required substructure** of a Component consists of the Component itself, all of its required children, all of their required children and so on down to, but not including, any optional Components.
- A rule **relates** two Components if it relates a Feature or Option of one Component with a Feature or Option of the other.

If you have two optional Components and you want to build a rule that relates them or their sub-Components, the following must be true:

- One Component must be a child of the other.
- If sub-Components are involved in the rule, they must be in the required substructure of the optional Component.

Here are examples of some acceptable combinations:

- A rule that relates Components within the required substructure of any Component.
- A rule that relates an optional Component with any of its children.
- A rule that relates an optional Component with a Component in the required substructure of any of its children.

Here are examples of some unacceptable combinations:

- A rule that relates two optional Components at the same level in the Model.

- A rule that relates an optional Component with a Component in the required substructure of an optional Component at the same level in the Model.
- A rule that relates an optional Component with an optional Component of any child Component.

The User Interface

When you are satisfied with any part of your Model and want to test it, you need to generate a user interface with which to access the Active Model. Oracle Configurator Developer automatically creates a user interface derived directly from the Model. This generated UI reflects the structure of your Model and provides all the UI elements needed for a running Oracle runtime configurator.

As you continue to work on your Model, you need to update the user interface to reflect changes in the Model. If your changes are not too extensive, you can select a User Interface node, then select Refresh from the Edit menu. If you make very extensive changes in the Model, the Refresh operation is not sufficient, and you must delete the existing UI and create a new one. For this reason, you do not want to do extensive customization before you have the Model complete.

Oracle Configurator Developer takes advantage of the Oracle Configurator architecture by storing the complete definition of the User Interface in the Oracle Configurator schema, where it is available to the Oracle Configurator Developer and the Active Model when you run the Oracle runtime configurator.

How the Oracle Runtime Configurator Displays the Model

Your choice of deployed application influences how your model appears in the Oracle Configurator window user interface. The choices are a stand-alone Oracle SellingPoint application, possibly in a mobile environment, or an Oracle Configurator window within Oracle Applications or in a custom web application. An Oracle Configurator window can either be a Java applet or a DHTML window. Much of how the Model is displayed in the deployed application is common to any of these deployments. See Configuring an Item in an Oracle Configurator Window on page 2-75 for the common behavior of any Oracle Configurator window. The following sections describe some of the differences.

Oracle SellingPoint Application

See the *Oracle SellingPoint Application Help* for information about configuring models in the Oracle SellingPoint application user interface.

Oracle Configurator Window: Java Applet

Only BOM nodes are visible in the Java applet. The Java applet UI cannot be modified. The Java applet ignores the Visibility toggle on items in Developer.

In the Java applet, each component is configured in the option selection pane (upper-right). The tree of configurable components is in the Model structure pane (left). End users navigate from one component to another in this Model structure pane. When an end user Selects a component, the selection options for that component's features display in the selection pane. Components whose configuration is unsatisfied or incomplete are marked by a red asterisk in the folder icon for that component.

List prices are displayed in the Java applet as soon as the options are displayed. The actual selling prices, such as discounts, are calculated and displayed only on request. This includes both total and per-item prices.

Available to Promise (ATP) dates are displayed in the Java applet on request.

It is possible to launch a DHTML Configurator window from the Java applet in Oracle Order Management. This possibility enables you to support guided selling and gathering customer requirements in a configurator launched from Order Management, which requires the Java applet. Is you add structure to gather customer requirements or to support guided selling to your Model, create a Components tree UI for the Model. If the Components tree UI was created or modified more recently than the BOM Model tree UI, the applet launches a browser running the Components tree UI. When the end user is done with the DHTML UI a small window is displayed with a button to terminate the configurator. When the end user clicks this button, the Java applet queries the database to get an appropriate termination message, passes the message on to Order Management and terminates itself.

If the BOM Model tree UI is the most recently modified, the Java applet is the UI launched from Order Management. The end user has access to either the DHTML UI or the Java applet UI, but not both.

Be aware that if you intend the end user to ever display the Java applet UI, your model must not have required selections that are not in the BOM. Because the non-BOM portion of the Model does not appear in the Java applet UI, the end user cannot make the required selections, and the configuration cannot be satisfied.

Oracle Configurator Window: DHTML

The DHTML window can be fully customized.

In anyDHTML window, each component is configured on a separate screen in the configuration window. Selection options might be presented in the form of needs assessment questions to be answered about requirements for the product. End users click the button in the selection pane for advancing to the next component to be configured.

In anyDHTML window, there is a button to toggle between configuration and summary displays. Click the Summary button at any time during the configuration session to display the summary information. Click the Configuration button to return from the summary to the selection pane.

In the DHTML window, all prices and ATP dates are displayed dynamically, with selling prices and ATP dates displayed in the summary display.

Elements of the Generic User Interface

User Interface	This is the root node of the UI.
Product Selection	This is the next level below the User Interface. From this node, you can add new UI screens.
Components Tree	This node shows the hierarchical relationships between Components and sub-Products of your Model in the UI.
Screen	This node groups the nodes that appear together on a single screen in the UI. From this node, you can add Buttons, Pictures, and Text.
Picture	This node represents a graphic image file. Add graphics to improve the appearance of the UI. You can also associate an action with an image.
Text	This node represents text in the UI. This is not the text that provides labels for Feature Controls and Value Displays
Button	This node represents a button in the UI. The end-user selects a button in order to perform some action.
Feature Control	This node represents a Feature in the UI. The end-user uses Feature controls to input values for Features or select from Options of a Feature.
Value Display	This node represents Totals and Resources in the UI. Value Display fields display values to the end-user, and are read-only.

How the Model Shapes the User Interface

The user interface you generate is based on your Model. Therefore, the structure of the Model influences the structure of the user interface in some important ways.

A user interface is based on a single Product, which can contain sub-Products. Sub-Products can have UI of their own, or be included in a UI based on their parent Product.

Products, Sub-Products, BOM Models, BOM OptionClasses, and Components are represented by screens in the UI. The Model tree is 'flattened' in the UI tree. Screens all appear at the same level in the UI tree, no matter now deeply nested the corresponding node may be in the Model tree.

Data Fields

Features, Resources, and Totals appear as fields on the screens for their parent Products and Components. For imported BOM information, BOM OptionClasses, and BOM StandardItems display as fields on the screens for their parent BOM Models and BOM OptionClasses. Totals and Resources generate Value Displays. The type of a Feature determines the default form for the field in the UI

- Features of type List of Options with maximum = 1 and minimum = 1 and no counted options generate Dropdowns.
- Features of type List of Options with counted options generate Selection Lists with a numeric input field for each Option.
- Features of type List of Options with maximum > 1 and no counted options generate Selection Lists with check boxes for each Option.
- Features of type Integer Number or Decimal Number generate Numeric Edit Controls.
- Features of type True/False generate check boxes.
- Features of type Text generate Text Edit Controls.

Data Field Labels

The data fields for Features, Resources, and Totals have automatically generated labels. The default text for these labels is the name of the Features, Resources, or Total. You cannot choose to display by either the name or description. Only display by name is possible. The display area for labels is a fixed, predetermined size. This default size may not be the best for your Oracle runtime configurator. Be sure to check the appearance of labels in your generated UI screens.If necessary, you can

alter the size of the label display area. See Customizing the Generic User Interface on page 2-62.

Generating a New User Interface

1. Select the **UI** button on the main toolbar to go to the User Interface module.
2. Choose **New User Interface** from the **Create** menu. You can also highlight the User Interfaces node in the UI Context Tree View, select the right mouse button, and select **New User Interface** from the pop-up menu.
3. A New User Interface dialog appears, prompting you for additional information to be used in generating the User Interface. That dialog contains the following controls:
   - A **User Interface Tree** that shows the Products in your Model. Select the product for which you want to generate the User Interface. The created interface reflects the structure of your Model for the selected Product node and all its children.
   - A **dropdown list** of the available UI Style options. The default value is Components Tree. This is the right selection if you are deploying your configuration model as DHTML in a browser or an Oracle SellingPoint application. If you select BOM Model tree, the resulting UI cannot be customized. You should select BOM Model tree only if your deployed configuration is the Java applet.
   - A **dropdown list** for selection of the minimum target display resolution:
   - 640 x 480 (default)
     - 800 x 600
     - 1024 x 768
     - 1600 x 1200
   - A **check box** labeled **Show all nodes**, which overrides all **Display in User Interface** settings in the Model. If the **Show all nodes** box is checked, all nodes descended from the Product or Component at the root of the User Interface are displayed in the UI regardless of whether they are marked for display in the Model. By default, this box is unchecked, which respects the visibility settings of the Model nodes.
   - **OK** and **Cancel** buttons.

4. When you select **OK**, an algorithm creates User Interface nodes corresponding to the nodes in the Model. The User Interface structure is displayed in the User Interface Tree.

   The top-level node of the new User Interface structure has a default name generated from the name of the Product node the UI is based on followed by 'User Interface.' You can edit the name and description of the UI in the usual fashion.

Customizing the Generic User Interface

The generic User Interface saves you a lot of time and routine work, but it may not satisfy all the requirements you have defined for your Oracle runtime configurator User Interface. If you are deploying your configuration model as Dynamic HTML in a browser, you can modify the generic User Interface to meet the unique needs of your configurator. User Interfaces generated for Configurator Java Applet cannot be customized.

Any customizations that you make to the generic User Interface are stored in the Oracle Configurator schema, so they are available to your running Oracle runtime configurator.

Some of the many customizations you can make to your generic User Interface are:

- Editing User Interface Objects on page 2-63
- Customizing the User Interface Default Settings on page 2-64
- Customizing the Product Selection Default Settings on page 2-66
- Customizing the Display of the Components Tree on page 2-66
- Customizing Screen Display Settings on page 2-67
- Creating new User Interface Screens on page 2-67
- Customizing Screen Design on page 2-68
- Hiding Components from User Interface Display on page 2-72

Editing User Interface Objects

Editing in the Attributes View

The Attributes View on page 3-14 is used to modify the Attributes of a selected User Interface node object. See User Interface Attributes on page 3-31.

A number of aspects of the UI can be specified at a number of points throughout the UI tree structure, so they appear repeatedly in the User Interface Attributes View They are summarized here.

Font

Specifies a text font. Select the dialog **(...)** button in the **Font** field to open the Font dialog, then select a font and its related attributes.

Picture

Specifies the pathname to a graphics file. Select the dialog **(...)** button in the **Picture** field to open the standard Windows Open dialog. Select the picture file you want. Select **Open**. The full path of the file you have selected as the background appears in the Picture field. To remove an image file, highlight the field contents and delete the path.

If you add graphic images to your User Interface, note that Oracle Configurator Developer supports only GIF (*.gif) and JPEG (*.jpg) files.

Color

Specifies a color. Select the **Use Default** option box or select the dialog **(...)** button in the **Color** field to open the Color dialog. Select a basic color or select the **Custom Color** button to define a unique color. Select **OK**.

Editing in the Preview Window

The Preview window allows editing of the Layout Data, such as position and size, for User Interface objects on a Screen. You can open the Preview Window by selecting a Screen node in the User Interface Tree and choosing **Preview** from either the **View** menu or the right-click pop-up menu.

You can open multiple Preview windows at the same time, but only one Preview window can be opened per Screen. Changes made in the Preview window are immediately reflected in the Attributes View on page 3-14 for the selected User Interface node, if applicable. Likewise, changes made in the Attributes View are

immediately reflected in any Preview window that is displaying the selected User Interface node. Moving or sizing a User Interface object in the Preview window displays the coordinate values in the Attributes View on page 3-14 as they change.

The Preview window does not accurately show all aspects of the appearance of the screen in the UI.

Customizing the User Interface Default Settings

Each User Interface has overall default settings for screen background, font, and logic state display. You can customize these settings specifically for your enterprise. For example, you can display a custom picture or color in the background.

Any customizations you make to the default settings appear throughout the User Interface on all screens that do not override the defaults. You can use the default settings to provide a unified appearance to your User Interface.

You can also set the logic state display. The logic state display enables the end user of the Oracle runtime configurator to see the logic state of objects in the configuration. You can display logic state using either icons or the color of label text or both. Logic state display using text color is not supported in the DHTML window. See Configuration Rules and Logic State on page 2-20 for further information on logic state.

Since you can define multiple User Interfaces in your configuration model, you may want different default settings for each one.

Hiding Unavailable Options

You can specify that unavailable Options should be hidden in the runtime UI. You can specify this behavior at two levels. You set the default behavior in the UI Definition, and you can set an optional override for each Option Feature control.

Computing whether an Option is available can have some performance cost. To reduce impact on performance, you can specify that Options having a logic state of false due to the action of rules (logical false or lfalse) should be interpreted as unavailable. You can specify this option in the UI Definition only, not per Feature.

Option Features that specify a maximum number of selections should probably not be set to hide their Options when using lfalse instead of availability. This would cause all unselected Options to disappear once the maximum is met, which would be particularly bad in a Feature with a maximum of 1.

Changing the User Interface Default Settings

1. Select the **UI** button on the main toolbar.
2. Select the node for the User Interface you want to modify.
3. In the Attributes View in the right pane, open the **Defaults** section if it is closed.
   - Use the **Color** field to select a background color. See Color on page 2-63.
   - Use the **Picture** field to select the image file you want as background. See Picture on page 2-63.
   - Use the **Font** field to select a Font. See Font on page 2-63.
   - Select **Icons**, **Text color**, or both to determine how the User Interface displays logic state. If you do not want to display logic state, do not select either of the logic state display options.
   - Select the **Icons** option box to display logic state using icons.

     Select **Select...** to open the Logic State Icons dialog. You can choose between check boxes, Radio Buttons or Icons by selecting the appropriate tab. The dialog displays the default graphics used in the Oracle runtime configurator to represent the possible logic states, and the name of the associated graphics file.

     You can change the graphic used by selecting a different graphics file. Select the dialog **(...)** button to open the standard Windows Open dialog and specify a new file path. The default graphics files are stored in a directory `Shared\Active Media` in the Oracle Configurator install directory. You can use your own graphics files, which can be stored wherever you like, as long as the path you supply in Oracle Configurator Developer is the same as the path needed to find the files in the deployed environment. Select **OK**.
   - Select the **Text Color** option box to display logic state using the color of label text.

     Select **Select...** to open the Logic Text Colors dialog. The dialog displays the currently selected color for each logic state.

     For each logic state field, use the **Color** field to select a color. See Color on page 2-63.
   - Select **HideUnavailableOptions** or **HideLFalseOptions**. See Hiding Unavailable Options on page 2-64.

Customizing the Product Selection Default Settings

A Product Selection node is created for each new User Interface you create. The Product Selection node contains the User Interface Tree nodes derived from your Model. You can customize the name and description for the Product Selection node of your User Interface. You can also define background color, picture, or font here instead of using the User Interface default settings.

Changing Product Selection Section Default Settings

1. Select the **UI** button on the main toolbar.
2. Expand the User Interface Tree and select the **Product Selection** section node.
3. In the Attributes View in the right pane, open the **Styles** section if it is closed.

   Use the **Background Color** field to specify a color. See Color on page 2-63.

   Use the **Background Picture** field to specify a background picture. See Picture on page 2-63.

   Use the **Font** field to specify a font. See Font on page 2-63.

Customizing the Display of the Components Tree

The Components Tree is displayed by default using simple lines. The Components Tree always appears in the Configurator Java applet and cannot be customized.

Changing the Display Style

1. Select the **UI** button on the main toolbar.
2. Expand the User Interface Tree and select the UI section node.
3. Select the **Components Tree** node.
4. In the Attributes View in the right pane, open the **Definition** section if it is closed.
   - Select a **Tree Style** from the dropdown list. See Tree Style on page 3-32.
   - Type the text you want to appear as a tip or guide to the user in the **Tooltip Text** field.
   - Select a font in the **Font** field. See Font on page 2-63.
   - Specify a background image with the **Picture** field. See Picture on page 2-63

- Specify a border for the picture with the **Borders** field. See Borders on page 3-33
- Use the **Background Color** field to specify a color. See Color on page 2-63.

Customizing Screen Display Settings

When you create a new User Interface, the User Interface tree is populated with screen nodes that directly correspond to each Product, Component, and Feature node defined in your Model. Each screen node uses the default background color, background picture, and font, unless you choose otherwise.

Changing the Screen Display Settings

1. Select the **UI** button on the main toolbar.
2. Expand the User Interface Tree and select the screen node you want to change.
3. In the Attributes View in the right pane, open the **Styles** section if it is closed.
   - Use the **Background Color** field to specify a color. See Color on page 2-63.
   - Use the **Background Picture** field to specify a background image. See Picture on page 2-63.
   - Use the **Font** field select a Font. See Font on page 2-63.

Creating new User Interface Screens

To add new screen to your User Interface:

1. Select the Produce Selection node of your User Interface
2. Select **New Screen** from the **Create** menu or use the right mouse button to access the pop-up menu and choose **New Screen**.
3. In the Attributes View in the right pane, type the name of the screen in the **Name** section.
4. Open the **Description** section if it is closed and type the new descriptive text.
5. You can further modify the screen as described in Customizing Screen Display Settings on page 2-67

Customizing Screen Design

Each screen in your User Interface contains nodes for the title bitmap and text. You can customize screens by modifying the title bitmap and text, and by adding pictures, text, and buttons, and by modifying the display of Features, BOM StandardItems, Totals and Resources.

Customizing Screen Graphics

To add or customize screen graphics, including the Title Bitmap of the screen:

1. Select the **UI** button on the main toolbar.
2. Expand the User Interface Tree and select the UI section node.
3. Create or select the screen node you want to customize.
4. To add a graphic image, use the right mouse button to select the screen node and select **New Picture** from the menu or select **New Picture** from the **Create** menu.
5. In the Attributes View in the right pane, type the name of the node in the **Name** section.
6. Open the **Description** section if it is closed and type the new descriptive text.
7. Open the **Definition** section if it is closed.
   - Type the text you want to appear as a tooltip in the **ToolTip Text** field.
   - Use the **Picture** field to select a graphics file. See Picture on page 2-63.
   - Specify a border for the picture with the **Borders** field. See Borders on page 3-33
   - Use the **Action** dropdown list to select the action you want to the Oracle runtime configurator to take when the user selects the title bitmap. See Action on page 3-33 for descriptions of available actions.
8. Open the **Layout** section if it is closed. Specify the Label layout and Tab Order. See Layout on page 3-35.

Adding Text to a Screen

To add new text to your screen:

1. Select the **UI** button on the main toolbar.
2. Expand the User Interface Tree and select the UI node.

3. Create or select the screen node you want to customize.
4. Use the right mouse button to select the screen node and select **New Text** from the menu or select **New Text** from the **Create** menu.
5. In the Attributes View in the right pane, type the name of the text node in the **Name** section.
6. Open the **Description** section if it is closed and type the new descriptive text.
7. Open the **Label** section if it is closed.
   - Type the new text as you want it to appear in the **Text** field.
   - Select the **Use Default** option box or select the dialog **(...)** button in the **Font** field to open the Font dialog. Select a font and its related attributes. Select **OK**.
   - Use the **Background Color** field to specify a color. See Color on page 2-63.
   - Select Transparent or Opaque from the **Background Style** dropdown list.
8. Open the **Layout** section if it is closed. Specify the Label layout. See Layout on page 3-35.

Adding Buttons to a Screen

You can add buttons to screens of your User Interface. You generally add a button so that the end user can perform some action by selecting the button.

Text and Graphics on Buttons

You can add graphics to a button by specifying a graphics file in the **Picture** field. See Picture on page 2-63. However, you need to know the size of graphic image and the button, because graphics are not displayed on buttons in the Preview window.

If you deploy your configurator as an Oracle SellingPoint application, you can use both text and graphics on a button.

If you deploy your end-user configurator as a Dynamic HTML in a browser, you can use either text or graphics on a buttons, but not both. If you specify both text and a graphic on a button, and deploy as Dynamic HTML in a browser, the DHTML window displays only the graphic, because the client process checks for the image source first, and displays the image if it finds one.

To Add Buttons to a Screen

1. Select the **UI** button on the main toolbar.
2. Expand the User Interface Tree and select the UI node.
3. Create or select the screen node you want to customize.
4. Use the right mouse button to select the screen node and select **New Button** from the menu or select **New Button** from the **Create** menu.
5. In the Attributes View in the right pane, type the name of the button node in the **Name** section.
6. Open the **Description** section if it is closed and type the new descriptive text.
7. Open the **Definition** section if it is closed.
   - Type the text you want to appear as a tooltip in the **ToolTip Text** field.
   - Use the **Picture** field to select the image file you want to use. See Picture on page 2-63. See Text and Graphics on Buttons on page 2-69 for more information on use of graphics on buttons.
   - Specify a border for the picture with the **Borders** field. See Borders on page 3-33.
   - Use the **Action** dropdown list to select the action you want to the Oracle runtime configurator to take when the user selects this button. See Action on page 3-33 for descriptions of available actions.
8. Open the **Label** section if it is closed.
   - Type the new button text as you want it to appear in the **Text** field. See also Text and Graphics on Buttons on page 2-69.
   - Select the **Use Default** option box or select the dialog **(...)** button in the **Font** field to open the Font dialog. Select a font and its related attributes. Select **OK**.
   - Use the **Background Color** field to specify a color. See Color on page 2-63.
   - Select Transparent or Opaque from the **Background Style** dropdown list.
9. Open the **Layout** section if it is closed. Specify the Label layout and Tab Order. See Layout on page 3-35.

Customizing Features, Totals, and Resources

A User Interface node is automatically created to correspond with each Feature node in your Model. You can make your User Interface more interactive by customizing the Feature nodes in your User Interface to display as questions for the selection of options. For example, instead of displaying a list of color options with a color label, you can display “What color would you like?” next to the selection list. You can also provide more informative text for display of Totals and Resources.

Changing Feature, Total, and Resource, Nodes

1. Select the **UI** button on the main toolbar.
2. Expand the User Interface Tree and select the UI node.
3. Expand the **Screen node** containing the object you want to customize.
4. Select the node you want to customize.
5. Open the **Description** section if it is closed and type the new descriptive text. This could describe a question or be the same information that was entered in the Name section.
6. Open the **Definition** section if it is closed.
   - Select the type of control mechanism you want displayed for this Feature (dropdown or selection list) in the **Control** field.
   - The **Format String** field for Totals and Resources is not currently implemented.
   - Type the text you want to appear as a tooltip in the **ToolTip Text** field.
   - Use the **Font** field to select a Font. See Font on page 2-63.
7. Open the **Label** section if it is closed.
   - Type the new text as you want it to appear on the field label in the **Text** field.
   - Use the **Font** field to select a Font. See Font on page 2-63.
   - Use the **Background Color** field to specify a color. See Color on page 2-63.
   - Select Transparent or Opaque from the **Background Style** dropdown list.
8. If the Feature is a List of Options, the **Option Display** section is available. Open it and Select the corresponding radio button to indicate if you want to **Label each option** or **Display pictures**. See Option Display on page 3-34.

9. Open the **Layout** section if it is closed. Specify the Control and Label layout, and the Tab Order. See Layout on page 3-35.

Hiding Components from User Interface Display

Occasionally you may not want one of the Components you defined to be visible by the user. For example, you may define a Component that is a collection of Totals used in calculations required by the configuration. While these are very useful in testing your Oracle runtime configurator, your end users do not need to see them, so you can hide the Component.

If you defined a Component that you do not want to be visible by the user, for example a Total that is used as an intermediate value in a more complex calculation, you can hide the Component.

To Hide a Component

1. Select the **UI** button on the main toolbar.
2. Expand the User Interface Tree and select the UI node.
3. Select the node that you want to hide.
4. At the top of the Attributes View, select the **Go To** button.

   The Model Tree opens, automatically selecting the Model node corresponding to the User Interface node.

   Each User Interface node that corresponds to a Model node has a Go To button that allows you to navigate directly to that node in the Model Tree.

   Although you are in the Model View, you are still in the User Interface module. The Attributes View displays a list of the User Interface nodes associated with the Model node.
5. Switch to the Model module, by selecting the **Model** button on the Toolbar.
6. Open the **Visibility** section of the Attributes View for the selected node. Select the check box labeled Display in User Interface to clear it.
7. Switch to the User Interface module, by selecting the UI button on the Toolbar.
8. In the Context Tree View for User Interfaces, select the node for the User Interface.
9. Choose **Refresh** from the **Edit** menu.

10. When you expand the User Interface Tree, the User Interface node corresponding to the hidden Model Component is no longer there.
11. When you run the Oracle runtime configurator again, the node does not appear.

Test/Debug

In order to test your Oracle runtime configurator, you must create a Model structure and User Interface, and run Generate Active Model. You can test a portion of the Model structure before you construct the rest of it. You can also test how the Model structure looks in the UI before you create any Configuration Rules.

See The Model on page 2-7, The Configuration Rules on page 2-19, and The User Interface on page 2-57 for further information on constructing these parts of your Oracle runtime configurator.

Before testing, you must also select the deployment environment for testing your Oracle runtime configurator. See Select Test Environment on page 3-10.

When you are ready to test, select **Test** from the main toolbar or **Test/Debug** from the **View** menu. This starts the Active User Interface so that you can test the Model you have created directly in the end user application that is automatically generated from the Model and Configuration Rules.

If you are testing your model in the Oracle SellingPoint application test environment:

1. Type your User Name and Password. See the *Oracle Configurator and SellingPoint Administration Guide* for this information.
2. Select the source database you want to use.
3. Select the Login button.

If you are testing your model in the Dynamic HTML in a browser test environment:

1. Configurator Developer sends an initialization message to the UI servlet, using the URL you specified in the Tools / Options dialog.
2. Your current session information (database instance, username, password, etc.) is used to connect to the Oracle Configurator schema.
3. If you have defined more than one User Interface in Configurator Developer, a popup menu lists their names. Select the one that you want to test.
4. Your default web browser opens, displaying the current Model using the selected User Interface.

you can proceed to view and test the Model structure, Configuration Rules, and User Interface of your Oracle runtime configurator. See Configuring an Item in an Oracle Configurator Window on page 2-75 for details about using the Oracle runtime configurator.

If you have modified the model since the last time you generated the User Interface, you must refresh the UI or create a new one prior to testing. If you have changed the Model or any Configuration Rules, you must regenerate the Active Model. If you are testing with the Oracle SellingPoint application, close all running test instances before testing a modified and regenerated model. Restarting the test / debug module ensures that you see all your changes.

Dynamic HTML in a browser and the Configurator Java Applet test environments do not reflect changes in your model until the servlet is restarted. For example, if you are testing your configuration model and notice an error in a button caption, you go into Developer to fix the caption. The change you just made is not reflected in the model currently running in your browser, and it is not reflected in the next model you start. You can wait for the cartridge to time out, as long as nobody else is using the cartridge, or send a `"killAndRestartServer"` message to the Web UI servlet.

You can send the `"killAndRestartServer"` message by requesting a URL. For example, if the servlet URL is:

```
http://www.mysite.com:10130/servlets/oracle.apps.cz.servlet.UiServlet
```

Request the following URL:

```
http://www.mysite.com:10130/servlets/oracle.apps.cz.servlet.UiServet?killAndRestartServer=true
```

Testing your Model

Verify that the Model structure as displayed through the User Interface functions effectively in the Oracle runtime configurator. See How the Oracle Runtime Configurator Displays the Model on page 2-57 for more information on how Model structure and User Interface are related.

View and test the structure of your Project for visibility and other Properties that you have applied. See Configuring an Item in an Oracle Configurator Window on page 2-75 for details about how the user interface works.

Testing your Configuration Rules

You test your Configuration Rules to determine they are giving you the desired results. It is good practice to test Configuration Rules incrementally. You can temporarily disable selected rules in order to determine what rule is causing problems in the Oracle runtime configurator.

Testing your User Interface

Note whether the User Interface has the look and feel that you want for your Oracle runtime configurator. Check that the screens present appropriate information to the end user in an easily usable format.

Configuring an Item in an Oracle Configurator Window

You can use the following steps for any Oracle Configurator window:

1. Oracle Configurator contains a selection pane in which you choose options. It may also contain a Model structure pane (on the left), which displays a tree of the components in the configurable product.
2. In the selection pane, begin making selections from the options that are presented.

To select this ...	Do this ...
One option	Select the check box for the desired option.
One option in a group of mutually exclusive options	Select the check box for the desired option. (This list might be presented as a set of radio buttons.)
One or more options in a group that allows multiple selections	Select the check boxes for all the desired options.
A quantity for a numeric option	Enter the desired quantity. Entering a quantity greater than zero automatically selects the option.

3. As you make selections, they are automatically validated against the rules that have been defined for the model or item that you are configuring.

   If you change one of your selections, it can automatically change the choice of valid selections for other features of the product, according to the configuration rules. You see those changes when you select an affected feature.

If you make a selection that violates a configuration rule, the configuration window displays a message describing the violation and your options for dealing with it.

4. When you are finished making selections for one component, you move on to the next components, until you configure the entire product.

   For information on navigating, see How the Oracle Runtime Configurator Displays the Model on page 2-57.

5. View the summary to display all the selections you have made, and other option information, such as quantity, price, and the total price for the configured product.

6. If an Availability button is displayed, click it to show the Available To Promise (ATP) dates for the relevant items.

7. When you have finished the entire configuration, click the Done button to continue processing your order.

   If the configuration is satisfied, you have made all the required valid selections.

   If the configuration is unsatisfied, a message tells you how to either ignore it and continue, or return to the selection pane to finish the configuration.

   The host application then closes the configuration window and continues processing your order.

Configurator Developer Messages

As you work in Configurator Developer, you occasionally receive messages that alert you to errors or warn you about potential problems in your configuration model. For example, if you create a rule but do not define it, you receive a warning message when you generate the active model. These messages, along with other execution status information, are logged by Configurator Developer in the datastore log. You can display the contents of the datastore log in the Log Messages window by selecting **Show Datastore Log** from the Tools / Options dialog, then selecting OK. The Log Messages window appears. For more information on accessing the Log Messages window, see Options... on page 3-10. For more information on using the Log Messages window, see The Log Messages Window on page 3-35.

This window provides two selections on the menu bar: **File** and **Settings**.

The File menu selection for this window allows you to select whether log messages should also be written to a file, and to choose the file. Logging execution status to a file preserves this information and may be useful if you are having problems with

Developer and need to provide information to customer support. By default, messages are logged to a file called `Developer.log` in the directory `Osp\Developer` in your Oracle install directory.

Status messages have a severity level. You can select a level on the Report Settings submenu of the Settings menu. All messages of severity equal to or higher than the level selected are logged. All messages of severity lower than the level selected are quickly ignored and permanently discarded. Setting a very low severity level can have an impact on performance, due to the large number of messages logged. The default severity level is Notification.

Available severity levels from lowest to highest are:

Tracing and informational messages

- DetailTrace
- Info

Warnings

- Empty/EOF
- Notification
- Warning

Errors

- Error
- DataDamaged
- FATAL

Error Messages

- The error, "Runtime Error ’7’ " and "out of memory" tends to be caused by too little virtual memory.
- If a "Run-time error: Automation Error" occurs when opening a new configuration in the Oracle SellingPoint application on a machine running Windows NT, the User Variables CLASSPATH conflicts with the OSPC-installed System Variables CLASSPATH. See the *Oracle Configurator and SellingPoint Release Notes* for more information.

- If you see an error "The name is not in use for a subkey or named value." check the TNS, ODBC, DSN, and spx.ini information

3

Using Oracle Configurator Developer Tools

This section provides a central vocabulary for the instructions provided in the rest of the online help.

Elements of the Oracle Configurator Developer Window

The following list illustrates the terms that are used in describing what you can do with Oracle Configurator Developer.

Element	Example(s)
Window	Oracle Configurator Developer itself
Screen	Roadmap, etc.
Banner	Oracle Configurator Developer on window
	New Project on dialog window
Menu Bar on page 3-3	Bar containing “File”, “Edit”, “View”
Toolbars on page 3-12	Bar containing buttons for modules and operations
Panes and Views on page 3-13	Upper-left, lower-left, or right-hand
View	Model Tree (upper-left pane)
	Context Tree (lower-left pane)
	Attributes (right-hand pane)
Module	The Model Module on page 3-16, Configuration Rules Module on page 3-24, and User Interface Module on page 3-29 of Oracle Configurator Developer
Dialog	Open Project

Oracle Configurator Developer Editing Tools

There are several mechanisms you can use to edit information in Oracle Configurator Developer:

Controls

You work with Oracle Configurator Developer by using a variety of controls that are provided in the Oracle Configurator Developer window.

Element	Example(s)
Control	Standard window controls
	Expand / Hide on nodes and attributes
Dropdown	List of selections under left mouse select **View** in Menu bar
Text field	Text-editable box next to **Name:** label on the **Existing** tab of Open Project dialog
Button	**OK** (e.g., in Open Project), **Rules** on Toolbar
Check Box	**Display in User Interface** under the **Visibility** attributes section of a Model node.
Radio Buttons	Selection of logic relations in a Logic Rule (e.g., Requires, Implies, etc.)
Tab	Existing (e.g., in Open Project)
Label	**Description** in any Attributes view
Attribute	Labeled area of a section in the Attributes view
Icon	Graphical marker for **Component** in Model Tree, etc.
Pop-up	Shortcut menu of commands you can access by using a Right Mouse click on various UI elements, such as text fields.

Dragging and Dropping

You can perform most of the Edit menu cut, copy, and paste commands by dragging and dropping nodes between the panes.

For instance, you can drag-and-drop Items and Item Types from the Item Master Tree to populate a Product node with Components in the Model Tree.

Keyboard Shortcuts

Cut, Copy, and Paste commands have keyboard shortcuts provided. They are:

Command	Shortcut
Cut	ctrl-x
Copy	ctrl-c
Paste	ctrl-v

These are displayed on the Edit menu, next to the command they cut short.

Menu Bar

Each menu on the Oracle Configurator Developer Menu bar contains commands for performing a set of related operations.

The menus in Configurator Developer are context-sensitive. The set of commands on a menu changes to reflect the context in which it is being used. The best example is the **Create** menu. When you change to a different Oracle Configurator Developer module, the set of things that you can create changes.

The commands on Oracle Configurator Developer menus are also available on context-sensitive pop-up menus. When you select a node in one of the Panes on the left-hand side of the Oracle Configurator Developer window, then select the right-hand mouse button, a menu of appropriate commands is displayed, reflecting the operations that you perform on the selected node.

Oracle Configurator Developer provides the following Menus:

- File Menu on page 3-4
- Edit Menu on page 3-4
- Create menu:
  - Create Menu (for Model) on page 3-6
  - Create Menu (for Item Master) on page 3-7
  - Create Menu (for Configuration Rules) on page 3-7

File Menu

Use this menu to work with Projects, and to exit Oracle Configurator Developer.

A Project is a workspace in which you create your application. Projects are stored in the Oracle Configurator schema.

You can also use the buttons on the Editing Toolbar on page 3-12 for some of these commands.

Command	Description
New Project...	Creates a new Project.
Open Project...	Opens an existing Project.
Delete Project...	Deletes an existing Project.
Project Report	Generates a report on the Project, using Microsoft Word.
Save Project As...	Copies the current Project's Model, Rules, and User Interface definitions under another Project name.
Exit	Exits Oracle Configurator Developer.

Edit Menu

Use this menu to perform editing operations on the currently selected node or value.

You can also use the buttons on the Editing Toolbar on page 3-12 for some of these commands.

Command	Description
Cut	Cuts the selected node or value. A selected node is displayed in a disabled state until you Paste it. Any type of node may be cut. You can cut and paste within a Project, but you cannot cut a node from one Project and paste it into another.

Command	Description
Copy	Copies the selected node of any type or value. You can copy a node from one Project and paste it into another.
Copy With Rules	Copies the selected product or Component node, including its Configuration Rules.
Paste	Pastes the cut or copied node or value.
Rename	Renames the selected node.
Delete	Prompts you to confirm the deletion, then deletes the selected node.
Refresh	Rebuilds the selected user interface to reflect changes you have made in the Model.
Find	Enables you to find a node in the Model or Item Master.

Cut/Copy/Paste

Cut, copy, and paste operations are mechanisms for moving or reusing Model structure or Rules. You can perform these operations on any node in the Model or Rules tree views. You can also perform these operations in any textbox in the Attributes view.

Nodes that are cut or copied can be pasted under another node in the Model tree with the following limitations:

- Option nodes can only be pasted on a Feature node.
- Feature nodes can only be pasted on a Component or Product node.
- Component nodes can only be pasted on a Component or Product node.
- Total or Resource nodes can only be pasted on a Product, Component, or Feature node.
- You can cut and paste a node with populators, but if you copy a node with populators, the populators are lost in the copy.

You can copy and paste between Projects. For example, you can copy a subtree from one Project, open a different Project, and paste the subtree into that Project. You cannot cut a node from one Project and paste it into another.

Find

The option opens a dialog box in which you can type the name of the node you wish to find. The following figure shows the dialog box.

You can search for nodes in the Model or the Item Master.

You can search for parts of node names by using wild-card characters. The wild card for a complete string is % (per cent sign). The wild card for a single character is _ (underscore). If you need to search for a name that contains one of the wild-card characters, prefix the wild-card character with the escape character / (slash).

Search target	Finds
Feature/_10%	Feature_1040, Feature_1041, Feature_10 abc
Feature/_10_	Feature/_104

Create Menu (for Model)

Use this menu to create new nodes in your Model. Whether you can create a particular type of node depends on the type of the selected node.

Command	Description
New Folder	Creates a new Folder node under the currently selected node.
New Product	Creates a new Product node under the currently selected node.
New Component	Creates a new Component node under the currently selected node.
New Feature	Creates a new Feature node under the currently selected node.
New Option	Creates a new Option node under the currently selected node.

Command	Description
New Total	Creates a new Total node under the currently selected node.
New Resource	Creates a new Resource node under the currently selected node.

Create Menu (for Item Master)

Use this menu to create new nodes in your Item Master.

Command	Description
New Item	Creates a new Item node under the currently selected node.
New Item Type	Creates a new Item Type node under the currently selected node.

Create Menu (for Configuration Rules)

Use this menu to create new Configuration Rules. Your rules are organized as nodes in the Configuration Rules tree. When you create a rule, Oracle Configurator Developer places a node for the rule in the folder you have selected. You can create as many rule folders as you need to organize them in a meaningful manner. Rule folders that you create may contain multiple types of rules.

Command	Description
New Rule Folder	Creates a new Rule folder.
New Logic Rule	Creates a new Logic Rule node.
New Numeric Rule	Creates a new Numeric Rule node.
New Comparison Rule	Creates a new Comparison Rule node.
New Property-based Compatibility	Creates a new Property-based Compatibility node.
New Explicit Compatibility	Creates a new Explicit Compatibility node.
New Functional Companion	Creates a new Functional Companion node.
New Design Chart	Create a new Design Chart node.

Create Menu (for User Interface)

Use this menu to create new elements in your User Interfaces. Your elements are organized as nodes in the User Interface tree.

Command	Description
New User Interface	Creates a new User Interface in the User Interface Context Tree View, corresponding to the Product or Component selected in the Model tree.
New Screen	Creates a new Screen under the selected User Interface node.
New Text	Creates a new Text label under the selected screen node.
New Button	Creates a new Button under the selected screen node.
New Picture	Creates a new Picture under the selected screen node.

View Menu

The View Menu is divided into upper and lower sections. The lower section selects from among the Configurator Developer modules, and corresponds to the buttons on the Module Toolbar on page 3-12. The upper section presents choices that depend on what module is currently active.

View Menu (lower)

Command	Description
Model	Switches to the Model Module on page 3-16. Changes the display of the nodes in the Model, grouping them by Type or by Description.
Configuration Rules	Switches to the Configuration Rules Module on page 3-24.
User Interface Builder	Switches to the User Interface Module on page 3-29.
Test / Debug	Launches the Oracle Configurator test environment you selected for the current Model. Use the Tools / Options dialog to select a test environment.

View Menu (upper)

Command	Description
Model	Selects whether to display the nodes of the Model by Name or by Description.
Item Master	Selects whether to display the nodes of the Item Master by Name, by Description, by Item Type and Name, or by Item Type and Description.
Rules	Selects whether to display the nodes of the Configuration Rules by Name, by Type, or by Folders.
Preview	Opens the Preview window for the selected screen node in the User Interface.

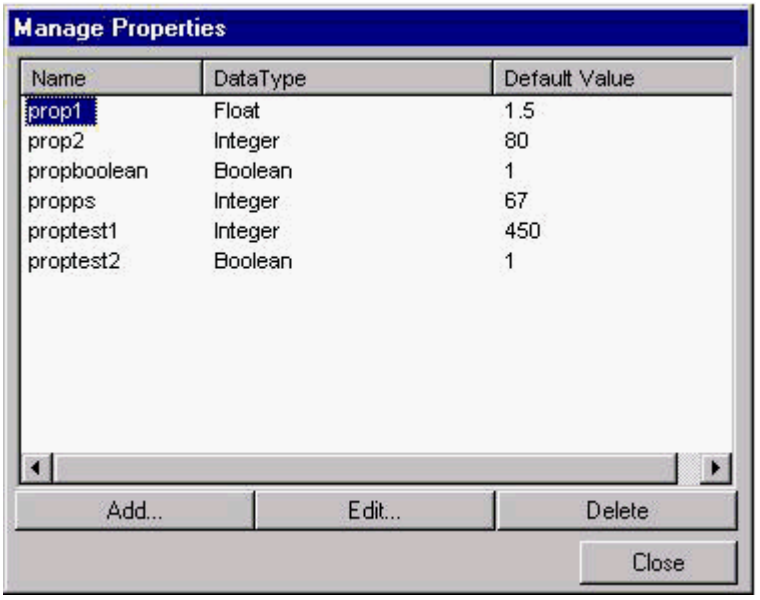
Tools Menu

Use this menu to change an Oracle Configurator Developer session option, manage properties, or to automatically build elements of your application.

Command	Description
Generate Active Model	Generates a generic active model for the currently selected Product. This includes generating logic rules.
RePopulate	Runs all Populators, refreshing your Model from the Item Master after it has been loaded with updated data, from the Import Tables.
Manage Properties	Enables you to create, edit, or delete properties. Use the property manager to completely delete a property from your project.
Options...	Opens the Options dialog for the current Oracle Configurator Developer session.

Manage Properties

The manage properties dialog enables you to add, edit, and delete properties in the list of properties that appears when you add a property to a node in your Model or Item Master.



Options...

The **Options** dialog enables you to control display of the Configurator Developer Log Messages window, and select an environment for testing and debugging.

Display Log Messages Window

Select **Show Datastore Log** on the **Log** tab of the Options dialog to display the Log Messages window. This dialog also displays the name of the file to which messages are written, if one is currently in use. For more information on the Log Messages window, see The Log Messages Window on page 3-35.

Select Test Environment

The **Test** tab of the Options dialog enables you to select the environment to use for testing your application. The available environments are:

Dynamic HTML in a browser	The Configurator Developer Test/Debug module displays the Model and User Interface in your default web browser.

SellingPoint application	The Configurator Developer Test / Debug module launches the Oracle SellingPoint application to display the Model and User Interface.

For Dynamic HTML in a browser you must supply the URL of the Servlet. The URL has the form:

```
http://host:port/servlet/oracle.apps.cz.servlet.UIServlet
```

For example:

```
http://www.mysite.com:60/myservlet/oracle.apps.cz.servlet.UiServlet
```

For more information on the Servlet and installing the Servlet on the server, see *Oracle Configurator Custom Web Deployment Guide* and *Oracle Configurator and SellingPoint Administration Guide.*

Your selection for test UI environment and the Servlet URL are stored in the `[test]` section of the `spx.ini` file. The value of the **`Launch`** parameter is `1` for SellingPoint application, `2` for Dynamic HTML in a browser. The value of **`InitServletURL`** parameter is the Servlet URL.

If you are using Dynamic HTML in a browser as your test environment, you must also specify the `JdbcUrl` parameter in the `spx.ini` file. If you are using SellingPoint application, or wish to switch between the two environments, you must specify both the `JdbcUrl` and `JdbcDriver` parameters. See *Oracle Configurator and SellingPoint Administration Guide* for more information on specifying parameters in the `spx.ini` file.

Any browser running the Oracle Configurator DHTML window must be set to display and use Javascript and Stylesheets, and must enable cookies.

Help Menu

Use this menu to get information about Oracle Configurator Developer and how to use it to construct your application.

Command	Description
Help Topics	Opens the online help for Oracle Configurator Developer.
About Oracle Configurator Developer	Displays information about this version of Oracle Configurator Developer.

Toolbars

Use the toolbars as a convenient way to choose certain commands.

Module Toolbar

Use this toolbar to choose a Module in the Oracle Configurator Developer window. You can also use the commands on the View Menu on page 3-8.

Go to the Model Module, described on page 3-16.

Go to the Configuration Rules Module, described on page 3-24.

Go to the User Interface Module, described on page 3-29.

Go to the Test/Debug Module, described on page 3-35. This module runs the specified test application.

Editing Toolbar

Use this toolbar to create and open Projects, and to perform basic editing operations on the selected node or text.

New Project

Open Project

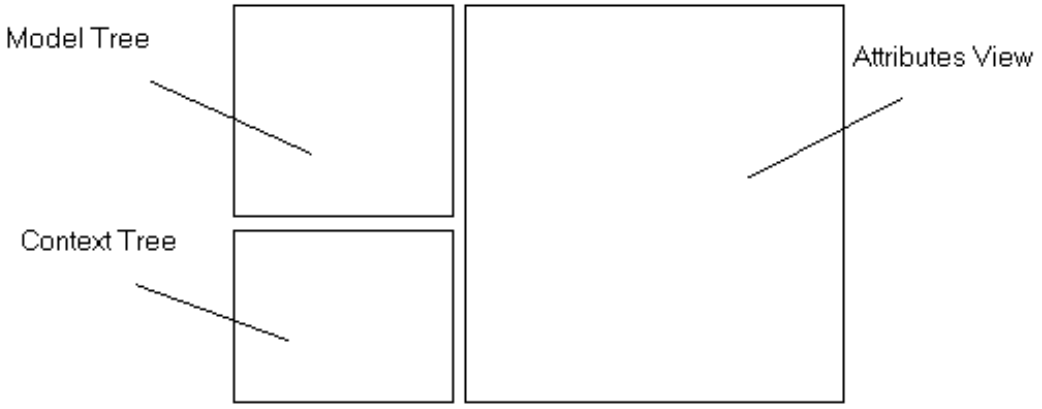
Cut

Copy

Paste

Panes and Views

The Oracle Configurator Developer window is divided into several panes. Each pane contains a view that you use for particular tasks in creating your application.



Model View

The Model View appears in the upper-left pane. It is always available, since the Model is the organizing core of your application. When you open a new project, the Model View contains only the root Model folder.

Context Tree View

The Context Tree View appears in the lower-left pane. It displays a different set of nodes, depending on which module you are currently working in.

Current Module	Context Tree View displayed
Model Module on page 3-16	The tree of Item Master Items and Item Types.
Configuration Rules Module on page 3-24	The tree of Configuration Rules for your Model, organized by rule type.
User Interface Module on page 3-29	The tree of User Interface elements for your Model, organized by element type.

See Tree Views on page 3-15 for information about a specific context tree.

Attributes View

The Attributes View appears in the right-hand pane.

An Attributes View is available on every node of every tree used in Oracle Configurator Developer. The Attributes available in an Attributes View depend on the module and node you have selected. Attributes for the Developer modules are described in the following sections:

- Model Attributes (for the Model module)
- Item Master Attributes
- Model Attributes (for the Configuration Rules module)
- Configuration Rules Attributes
- Model Attributes (for the User Interface Module)
- User Interface Attributes

Each available Attribute is presented in section. To display the contents of a section, select on the right arrow icon next to a section label. The icon changes to a down

arrow, and the sections opens. To hide the contents of a section, select the down arrow icon.

Icon	Definition
	Right arrow icon. Indicates a closed attributes view section with contents hidden.
	Down arrow icon. Indicates an open attributes view section with contents visible.

Tree Views

Oracle Configurator Developer displays the Model, Item Master, Configuration Rules, and User Interface, as tree structures called tree views. Tree views show how elements of the structure are related to each other, and which structure contain others. Branches of the tree, can be collapsed and hidden or expanded and displayed by using the (-) and (+) controls that appear on nodes that contain other nodes.

Within a tree, each element of the structure is a node. The top-level node is the *root*. When all of the tree has been collapsed or hidden, only the root node remains visible. Nodes in Oracle Configurator Developer tree views have names and graphical icons associated with them. The graphical icons provide visual cues that indicate the kind of information the node represents.

The Oracle Configurator Developer Modules

Oracle Configurator Developer organizes your work into Modules. A Module is an area in the suite of tools for accomplishing a common set of tasks such as building a Model or defining rules.

Module	Purpose
Model Module on page 3-16	Creating Model from your Item Master data.
Configuration Rules Module on page 3-24	Creating Configuration Rules against your Model.
User Interface Module on page 3-29	Creating User Interfaces based on your Model.
Test / Debug Module on page 3-35	Launching your selected test environment.

Each Module contains two Tree views:

- The Model tree for the current Model.
- A Context Tree View view that varies with the Module: Item Master, Configuration Rules, or User Interfaces.

Each Tree view displays an Attributes View containing the Attributes that you need to use in constructing the different parts of your application.

You can switch between the Modules as you work, by selecting on the Module buttons in the toolbar.

Model Module

The Model Module is where you define the Model for your application. The Configuration Rules and User Interfaces for your application are built on top of the Model.

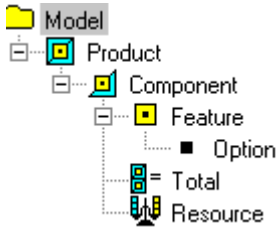
Tree Views for this Module

There are two tree views available for this module, the Model tree view and the Item Master tree view. The Attributes available in the Attributes View depend on which Tree view is selected.

Tree View Selected	Attribute View displayed
Model	Model Attributes (for the Model module) on page 3-18
Item Master	Item Master Attributes on page 3-22

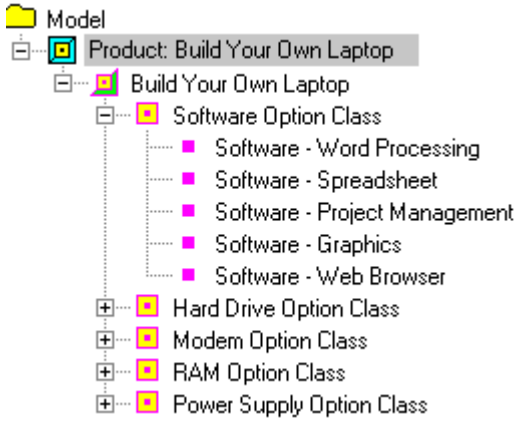
Model Tree View

You use the Model Tree to create and modify the Model nodes. The Model tree view consists of nodes that represent the related Products, Components, Features, Options, Resources, and Totals. The icons associated with each of these nodes are:



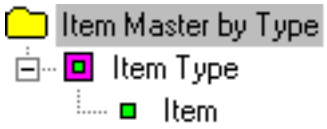
You can view the nodes in the Model either by name or by description.

The Model tree view uses colored icons to distinguish the nodes of an imported BOM Model.



Item Master Tree View

The Item Master tree view consists of nodes that represent the Items and Item Types in the Item Master. Item Types are the categories of the items. You can view items in the Item Master either by item name or by item description. You can also select whether or not to view items organized by Item Type. The icons associated with each of the nodes in the Item Master tree views are:



Model Attributes (for the Model module)

The Model view is available in every module of Oracle Configurator Developer, in the upper left pane (its default location). It provides different sets of Attributes for the selected node of your Model, depending on which Module you are working in.

The following table shows the Attributes that are available in the Attributes View when you are in the Model Module and select a node in the Model tree. Note that different Attributes are available for different types of nodes.

Attribute	Node Type					
	Products	Components	Features	Options	Totals	Resources
Name on page 3-19	X	X	X	X	X	X
Description on page 3-19	X	X	X	X	X	X
Visibility (UI) on page 3-19	X	X	X	X	X	X
Type on page 3-19			X			
Count on page 3-21	X	X				
Populators on page 3-21	X	X	X			
Properties on page 3-21	X	X		X		
Initial Value on page 3-22					X	X

Name

Provides a name for the selected node.

Oracle Configurator Developer assigns a default name based on the node's internal ID in the Oracle Configurator schema. Enter a more informative name to help you identify the node.

If you create a node with a Populator, then the name is set automatically to the name of the corresponding Item in the Item Master. (See Using Populators on page 2-16.)

You can use the same name for different nodes, since the Oracle Configurator schema's internal ID for each node is unique.

This name and ID appear in the Project Report.

Description

Provides a textual description of the selected node. By default, the Description is empty. Enter a brief explanation of the role that the node plays in your Model. The Description does not appear in the Oracle runtime configurator user interface.

Visibility (UI)

Controls whether the selected node and any children of the selected node appear in the User Interface. Controls both imported BOM nodes and nodes created in Configurator Developer. This is the only Model attribute you can select for imported BOM nodes. By default, all nodes appear in the User Interface. To exclude the selected node from your User Interface, clear the check box for **Display in User Interface**. The Java applet UI ignores this parameter.

Type

Specifies the type of data represented by the selected Feature node. Choose one of the selections in the dropdown list labeled Data Type.

List of Options	A List of Options node, displayed as children of the selected node. You can create Options "by hand", using the Create Menu (for Model) on page 3-6, or by defining and running a Populator (see Using Populators on page 2-16).
Integer Number	A whole number, such as 3 or 127.

Decimal Number	A number with a decimal part, such as 3.14159. Sometimes referred to as a floating point number.
True/False	Has two values, True and False. Sometimes referred to as boolean.
Text	A string of characters.

If you create a Feature with the type List of Options, be very careful about changing it to another type. Any Options or Populators associated with the Feature are permanently deleted. Any Configuration Rules using the deleted Options become invalid. The rule definition will indicate this fact.

Each data type requires specific additional information.

List of Options

- **Number of Selections:** Indicate the **Minimum** and **Maximum** number of this Feature that the end user can select.
  - The default value for both minimum and maximum is 1.
  - A minimum of 1 means that the Feature must be included in the configuration.
  - A maximum of 1 means the Feature can be included only once, which has the effect that selection of one Option prevents selection of any other Option.
  - A maximum of 0 means the Feature is optional. If you attempt to set a maximum value less than the minimum, the maximum value automatically reverts to the current minimum.
  - You can specify no maximum value by deleting the numeric value from the maximum field.
- **Counted Options:** The Counted Options feature is useful in situations where the end user needs to specify a number for the options of this feature. For instance, if the options of the Counted Options Feature are types of candy bars, the user can select six milk chocolate bars and six dark chocolate bars. A Counted Options Feature also allows a count of the Options to be used in a Numeric Rule (see Building Numeric Rules on page 2-31).

  If the Feature has a Maximum of 1, or if the user must not specify more than one of each option, do not check the Counted Options box.

Integer Number and Decimal Number

- **Range:** Specifies the maximum and minimum values for this Feature. An Integer Feature that has a minimum value greater than or equal to 0 is treated as a Count Feature. A Count Feature has a logic state in addition to its numeric value, and if it is set to 0 in the runtime application, its state becomes Unknown, which is displayed in the application UI as an empty box. Count Features can participate in Logic Rules, while Integer and Decimal Features cannot.
- **Initial Value:** Sets a value for this Feature has when your application starts.

True/False

- **Initial Value:** Sets a value for this Feature has when your application starts. In this case, choices are limited to True, False, and None.

Text

- Text Value: Sets the initial value for this text Feature.

Count

Controls the **minimum** and **maximum** number of instances of a Product or Component node that can be present in a valid configuration.

To make a selection mandatory in your application, set the Minimum to 1 or more. To make a selection optional, set the Minimum to 0 (Zero). The default setting is both a Minimum and Maximum of 1, which enforces configuration of exactly one instance.

Populators

Lists the Populators associated with the selected node, and allows you to add, edit, or delete them. To add a Populator, select the **Add** button. To modify the definition of a selected Populator, select the **Edit** button. To delete a selected Populator, select the **Delete** button.

See Using Populators on page 2-16 for detailed information about defining populators to populate your Model.

Properties

Enable you to define additional attributes for the node. Properties provide additional about the node, such as weight, diameter, or voltage

To add a Property, select the **Add** button. In the Add Property dialog, select an existing Property, or select New Property to create one. Provide a name and default value for the Property.

To modify the value of a Property, select a listed Property and select the **Edit** button. In the Edit Property dialog, change the value of the Property, or select the Default button to restore the Property's default value.

To delete a Property, select a listed Property and select the **Delete** button.

For a general explanation of Properties, see Properties on page 2-8.

To make project-wide modifications to Properties, use the Tools / Manage Properties dialog. See Manage Properties on page 3-9.

Initial Value

Sets a value that the selected node has before any quantities are contributed or consumed. For Resources, the default initial value is zero. You cannot set the initial value of a Resource to unknown.

For Totals, the default initial value is unknown. An unknown initial value displays as blank in the Initial Value field in Oracle Configurator Developer. An unknown initial value displays as zero in the Oracle runtime configurator. When the initial value of a Total is unknown, Configuration Rules that involve the Total do not propagate.

Item Master Attributes

The Item Master is a set of your enterprise data, structured into Items and Item Types, that is the primary source of data for your application.

The following table shows the Attributes that are available in the Attributes View when you are in the Model Module and select a node in the Item Master tree. Note that different Attributes are available for different types of nodes.

Attribute	Node Type	
	Item Types	Items
Name on page 3-23	X	X
Description on page 3-23	X	X
Type / Properties on page 3-23		X

Attribute	Node Type	
	Item Types	Items
Properties on page 3-23	X	

Name

Provides a name for the selected Item.

When you create Items in the Item Master by importing data, Oracle Configurator Developer assigns names based on the Import Tables.

When you create Items in the Item Master manually, Oracle Configurator Developer assigns a default name based on the node's internal ID in the Oracle Configurator schema. Enter a more informative name to help you identify the Items.

Description

Provides a textual description of the selected Item. By default, the Description is empty.

Enter a brief explanation of the role that the Item plays in your Item Master. The description does not appear in the UI for your finished application.

Type/Properties

Controls the Properties of Item and Item Type nodes, and the Item Type of Item nodes.

When you create Items in the Item Master by importing data, Oracle Configurator Developer creates Item Types based on the Import Tables.

To change the Item Type of the selected Item, select the **Change** button and choose an Item Type from the Item Types dialog. When you close the dialog, the Item is moved to the chosen Item Type. This is the only way to change the Item Type of an Item.

For information about changing the Properties of Item and Item Type nodes, see Properties on page 3-23.

Properties

See Properties on page 3-21.

Configuration Rules Module

The Configuration Rules Module is where you define all the Configuration Rules for your application. See The Configuration Rules on page 2-19.

Tree Views for this Module

There are two tree views available for this module, the Model tree view and the Configuration Rules tree view. The Attributes available in the Attributes View depend on which Tree view is selected.

Tree View Selected	Attributes View Displayed
Model	Model Attributes (for the Configuration Rules module) on page 3-25
Configuration Rules	Configuration Rules Attributes on page 3-26

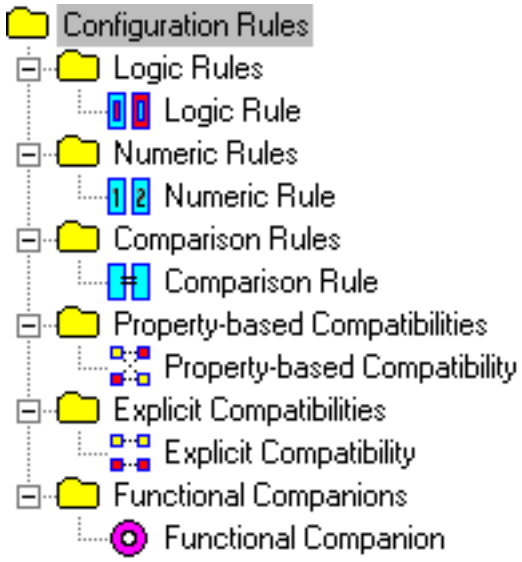
Model Tree View

You drag-and-drop nodes from the Model tree to the Configuration Rules that you are creating.

Configuration Rules Tree View

The Configuration Rules tree view consists of nodes that represent the rules, rule types and rule folders associated with the Model. Rule type nodes group rules by type. You can organize rules in any way that is useful to you by creating rule folders and putting your rules in them. You can create as many rule folders as you need to organize them in a meaningful manner. Rule folders that you create may contain multiple types of rules. You can use standard drag-and-drop or cut, copy, and paste operations to move or copy a rule from one folder to another. You can view Configuration Rules either by name, by rule type, or by folder.

The icons associated with each of the nodes in the Configuration Rules tree view are:



Model Attributes (for the Configuration Rules module)

The Model view is available in every module of Oracle Configurator Developer, in the left upper pane (its default location). It provides different sets of Attributes for the selected node of your Model, depending on which Module you are working in.

The following table shows the Attributes that are available in the Attributes View when you are in the Configuration Rules Module and select a node in the Model tree. Note that different Attributes are available for different types of nodes.

Attribute	Node Type					
	Products	Components	Features	Options	Totals	Resources
Name on page 3-19	X	X	X	X	X	X
Description on page 3-19	X	X	X	X	X	X
Count on page 3-21	X	X				

Attribute	Node Type					
	Products	Components	Features	Options	Totals	Resources
Type on page 3-19			X			
Initial Value on page 3-22					X	X
Visibility (UI) on page 3-19						
Properties on page 3-21			X			
Associated Rules on page 3-26	X	X	X	X	X	X

Associated Rules

Lists the Configuration Rules that involve the selected node.

To edit an associated rule, select it from the list and select the **GoTo** button. This selects the definition of the rule in the Context Tree View of Configuration Rules.

To create a new rule based on an existing one, select a rule from the list and select the **Copy** button. This creates a new rule, which you can edit by selecting **GoTo**.

To delete an associated rule, select it from the list and select the **Delete** button.

Configuration Rules Attributes

The following table shows the Attributes that are available in the Attributes View when you are in the Configuration Rules Module and select a node in the Configuration Rules tree. Note that different Attributes are available for different types of nodes.

Attribute	Node Type					
	Logic Rule	Numeric Rule	Comparison Rule	Property-Based Compatibility	Explicit Compatibility	Functional Companion
Name on page 3-27	X	X	X	X	X	X
Description on page 3-27	X	X	X	X	X	X
Parameters on page 3-27				X	X	
Definition on page 3-28*	X	X	X	X	X	X
Violation Message on page 3-29	X	X	X	X	X	

* The type of rule definition varies with the type of the selected node (rule type).

Name

Provides a name for the selected rule. Oracle Configurator Developer assigns a default name based on the rule node's internal ID in the Oracle Configurator schema. Enter a more informative name to help you identify the rule.

Description

Provides a textual description of the selected rule. By default, the Description is empty. Enter a description to help you understand the role that the rule plays in your Model.

Parameters

Lists the configurable elements of your Model that are used in a Property-based Compatibility or Explicit Compatibility rule.

To add a parameter, drag a node from the Model view into the **Participants** list. See Building Property-based Compatibilities on page 2-35 and Building Explicit Compatibilities on page 2-36 for details.

Definition

Used to define the selected rule.

Logic Rules To define a Logic Rule, drag nodes from the Model view into the A Side and B Side fields, then select a logic relation that describes the desired relationship between them. See Building Logic Rules on page 2-30 for details.

Numeric Rules To define a Numeric Rule, drag nodes from the Model view into the **Each Of** field in the A Side section, and select an operator and multiplier or divisor for the numeric value of the node. In the B Side section, drag a Total or Resource node from the Model view into the field, and choose the **Contributes to** or **Consumes from** logic relation. See Building Numeric Rules on page 2-31 for details.

Comparison Rules To define a Comparison Rule, drag a node from the Model view into the A Side and choose a comparison operator and a value to compare with the node's value. Drag a node from the Model view into the B Side section, and select a logic relation that relates it to the result of the comparison. See Building Comparison Rules on page 2-34 for details.

Property-based Compatibilities To define a Property-based Compatibility, drag nodes from the Model view into the Parameters on page 3-27 section, then choose the Properties of those nodes whose values are to be compared, and a comparison operation. See Building Property-based Compatibilities on page 2-35 for details.

Explicit Compatibilities To define an Explicit Compatibility, drag nodes from the Model view into the Parameters on page 3-27 section, then choose the specific values of those nodes that are explicitly compatible with each other. See Building Explicit Compatibilities on page 2-36 for details.

Design Charts To define a Design Chart, drag nodes from the Model view into the definition section, to define the Design Chart's Primary Features, Defining Features and Optional Features. Then define permitted combinations by selecting in cells of the resulting table. See Building Design Charts on page 2-37 for details.

Functional Companions To associate a Functional Companion with a Component, drag the desired Component from the Model tree into the **Base Component** field in the Attributes view. Then choose one or more roles for the Functional Companion (Validation, Auto-configuration, or Output), indicate how the Functional Companion is implemented, and type a Program String to identify the Functional Companion. See the *Oracle CIO Help* for further information on Functional Companions.

Violation Message

Controls the message that will be displayed if the Configuration Rules is violated when an end user makes an invalid selection.

User Interface Module

The User Interface Module is where you define the User Interfaces for your application. The Configuration Rules and User Interfaces for your application are built on top of the Model. See The User Interface on page 2-57.

Tree Views for this Module

There are two tree views available for this module, the Model tree view and the User Interface tree view. The Attributes available in the Attributes View depend on which Tree view is selected.

Tree view	Attributes View
Model	Model Attributes (for the User Interface Module) on page 3-30
User Interfaces	User Interface Attributes on page 3-31

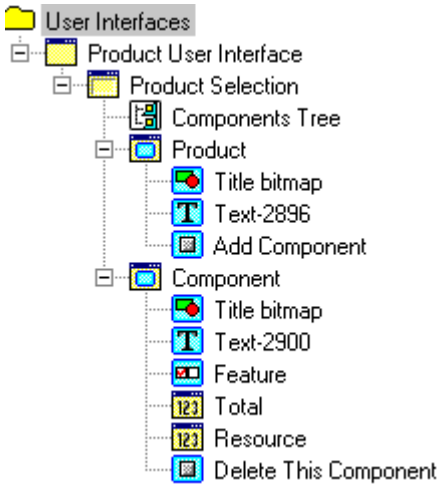
Model Tree View

You use the Model tree to create and modify the nodes that your application's User Interfaces are based on.

User Interface Tree View

You use the User Interface tree to organize your the nodes of your User Interface definitions, which are based on the nodes in the Model.

The nodes in the User Interface tree view represent the User Interface, Product Selection, Screens, Pictures, Text, Buttons, Feature Controls, and Value Displays for each UI. The icons associated with each of these nodes are:



Model Attributes (for the User Interface Module)

The Model view is available in every module of Oracle Configurator Developer, in the left upper pane (its default location). It provides different sets of Attributes for the selected node of your Model, depending on which Module you are working in.

The following table shows the Attributes that are available in the Attributes View when you are in the User Interface Module and select a node in the Model tree.

	Node Type					
Attribute	**Product**	**Component**	**Feature**	**Option**	**Total**	**Resource**
Name on page 3-19	X	X	X	X	X	X
Description on page 3-19	X	X	X	X	X	X
Associated UI Nodes on page 3-31	X	X	X	X	X	X

Associated UI Nodes

Displays a list of the User Interface nodes associated with the selected node. Select the **GoTo** button to view the corresponding node in the User Interface Context Tree View.

User Interface Attributes

The following table shows the Attributes that are available in the Attributes View when you are in the User Interface Module and select a node in the User Interface tree. Note that different Attributes are available for different types of nodes.

Attribute	Node Type								
	User Interface	Product Selection	Components Tree	Screen	Text	Picture	Button	Feature Control	Value Display
Name on page 3-32	X	X	X	X	X	X	X	X	X
Model Object on page 3-32	X			X				X	X
Description on page 3-32	X	X	X	X	X	X	X	X	X
Version on page 3-32	X								
Styles on page 3-32		X		X					
Defaults on page 3-32	X								
Definition on page 3-32			X			X	X	X	X
Label on page 3-34					X		X	X	X
Option Display on page 3-34								X	
Layout on page 3-35					X	X	X	X	X

Name

Specifies the name of the node.

Model Object

A reference to the node in the Model that a UI node is based on. Select the **Go To** button to move to the Model node corresponding to the selected UI node.

Description

Stores a description of the UI node.

Version

Provides information about the creation of the UI.

Styles

Controls **Background Color**, **Background Picture** and **Font** for the selected node.

Defaults

Specifies the default **Color**, **Font**, and **Picture**. The values specified here are used throughout the UI, unless you override them with specific choices for particular screens.

Definition

The information you specify in the Definition Attribute varies depending on the User Interface node you have selected. The following list describes the possibilities.

Tree Style Determines the way in which the Components Tree is displayed. Select one of the following:

- **None** - No Components tree structure is displayed.
- **Icons Only** - The Components tree structure displays object-specific icons next to the node name to depict hierarchy.
- **Treelines Only** - The Components tree structure displays with treelines next to the nodes to depict hierarchy.
- **Icons and Treelines** - The Components tree structure displays with icons next to the node name and treelines to further depict hierarchy.

- **Plus/Minus Only** - The Components tree structure displays only a plus sign or a minus sign next to the node name to indicate that the node can be expanded.
- **Icons and Plus/Minus** - The Components tree structure displays object-specific icons and a plus sign or a minus sign next to the node name to indicate that the node can be expanded.
- **Treelines and Plus/Minus** - The Components tree structure displays treelines to depict hierarchy and a plus sign or a minus sign next to the node name to indicate that the node can be expanded.
- **Icons, Treelines, and Plus/Minus** - The Components tree structure displays object-specific icons, treelines to depict hierarchy, and a plus sign or a minus sign next to the node name to indicate that the node can be expanded.

ToolTip Text Contains text that appears as a tooltip when the end user positions the cursor over the User Interface object in the Oracle runtime configurator that corresponds to the selected node.

Font See Font on page 2-63

Picture See Picture on page 2-63

Borders Specifies the type of border to use. Select none or fixed single from the dropdown list in the **Borders** field.

Background Color See Color on page 2-63

Action Specifies an action that takes place when the user selects a button or image in the User Interface. Select one of the following options from the **Action** dropdown list. The default action is **None**.

- **None** - No action.
- **Add Component** - Add a Component to the configuration. Specify the Component to add by selecting one from **Reference** selection list. Only Components having a maximum count greater than their minimum count appear in the list, because these Components are the only ones that can have variable numbers of instances.
- **Delete Component** - Delete a Component from the configuration. Specify the Component to delete by selecting one from **Reference** selection list. Only

Components having a maximum count greater than their minimum count appear in the list, because these Components are the only ones that can have variable numbers of instances, one of which can be deleted.

- **GoTo Screen** - Take the end user to a different screen. Specify the screen by selecting it from the **Reference** selection list.
- **Launch** - Specify a file (or URL) to be opened in a new window. Select the dialog **(...)** button for the **Reference** field to open the standard Windows Open dialog.
- **Functional Companion Output** - Select the Component the Functional Companion is attached to from the **Reference** selection list. Select the Functional Companion from the **Companion** selection list. This type of Functional Companion takes information from Oracle runtime configurator and provides post processing. For example, custom reporting.
- **Functional Companion AutoConfigure** - Select the Component the Functional Companion is attached to from the **Reference** selection list. Select the Functional Companion from the **Companion** selection list. This type of Functional Companion queries the state of specific parts of a configuration Model and determines if subsequent parts of the Model should then be changed and makes the changes to the configuration. These are sometimes used instead on logic rules.

Format String Not currently implemented.

Label

Controls the text label that identifies the node in your User Interface. Enter the **text** of the label in the **Text** field, or use the text that was generated from the corresponding Model node.

You can modify the **Font**, **Background Color**, and **Background Style** if you don't want to use the Defaults.

Option Display

Controls the label used for each selectable Option of the node in your application. Select the corresponding radio button to indicate if you want to **Label each option** or **Display pictures**.

If you choose to **Label each option**, indicate what you want to use to label the option (Name, Description, or Property). The information for this choice is the information you supplied in the Model. If you choose to use a Property, the Option

Display section changes to prompt you to select one of the Properties defined for the Feature.

If you choose to **Display pictures**, type the full path location of the image file you want to use.

Layout

Specifies the position and layout of a label or a control field in terms of the number of pixels from the **Left** and **Top** to position the upper left corner of the object, and the **Width** and **Height** in pixels. Values in these fields reflect any changes you make by editing in the Preview window. See Editing in the Preview Window on page 2-63.

In some cases, the Layout attribute enables you to specify the **Tab Order**. This is the navigation order for the object in regards to the overall tab sequence of the screen.

Test/Debug Module

This module launches the test environment specified in the Test tab of the Tools / Options dialog.

The Log Messages Window

You can activate the Log Messages window through the Options selection on the Tools menu. See Options... on page 3-10. For more information on the Log Messages window, see Configurator Developer Messages on page 2-76.

There are two selections available on the Log Messages window menu bar, **File** and **Settings**.

File Menu

Command	Description
Direct To...	Allows the user to select the disk file to which messages are written. Available only when messages are not being written to a file.
Close	Stops the recording of messages to a file. Available only when messages are being written to a file.

Command	Description
Keep File Open	By default, the log file is opened and closed with each message to ensure that the message is recorded successfully. When this option is checked, the log file is kept open, which speeds logging slightly, but it raises the risk of losing data if Developer terminates unexpectedly.
Hide	This item is positioned on the menu where 'Close' or 'Exit' is usually found because you cannot actually terminate logging, but you may not want to view the log window. This selection makes the log window invisible. Note that clicking the Close box, or pulling down the control menu and selecting Close also hides the window.

Settings Menu

Command	Description
Report Milliseconds	Ordinarily, log messages are time stamped at one-second intervals; when this option is selected, messages are time stamped to the millisecond.
Report Settings	Sets the minimum severity for reporting messages to the log window and to any open log file. The submenu lists the severity levels in ascending order. DetailTrace messages are the least severe; FATAL are most severe. The default is Notification.
Message Box Text Limit	Allows you to limit the number of characters stored in the message log window. Appending messages to the Log Window starts to become a performance issue when the window contains about 2500 characters of text. If you select Unlimited, note that there is an upper limit of 64Kb. This setting does not cause data to be lost from an open log file. All message written to the window are also written to the file and stored.

Glossary of Terms

This glossary for Oracle Configurator is followed by a Glossary of Acronyms

Active Model

The part of Oracle Configurator runtime architecture that processes model structure and rules to create configurations. Interfaces dynamically with the end user Active UI and data.

Active User Interface

The part of Oracle Configurator runtime architecture that provides the graphical views necessary to create configurations interactively. Interfaces with the Active Model and data to give users access to customer requirements gathering, product selection, and customer-centric extensions.

Application Architecture

The software structure of an application at runtime. Architecture affects how an application is used, maintained, extended, and changed.

Architecture

The software structure of a system. Architecture affects how a system is used, maintained, extended, and changed. See also Application Architecture.

Beta

An external release, delivered as an installable application, and subject to system, validation, and acceptance testing. Specially selected and prepared end users may participate in beta testing.

Bill of Material

A list of component items associated with a parent item (assembly) and information about how each item relates to the parent item.

BOM

See Bill of Material.

BOM Item

The nodes imported into the Oracle Configurator Developer Model that correspond to an Oracle BOM.

BOM Model

The imported Model node in the Oracle Configurator Developer that corresponds to Standard Model in an Oracle BOM.

BOM OptionClass

The imported Model node in the Oracle Configurator Developer that corresponds to Option Class in an Oracle BOM.

BOM StandardItem

The imported Model node in the Oracle Configurator Developer that corresponds to Standard Item in an Oracle BOM.

Boolean Expression

An element of a component in the Model that has two options: true or false.

Bug

See Defect.

Build

A specific instance of an application during its construction. A build must have an install early in the project so that application implementers can unit test their latest work in the context of the entire available application.

CIO

See Oracle Configuration Interface Object.

Client

A runtime program using a server to access functionality shared with other clients.

Comparison Rule

An Oracle Configurator Developer rule type to establish a relationship that determines the selection state of a logical item (option, boolean feature, or list-of-options feature) based on a comparison of two numeric values (numeric features, totals, resources, option counts, or numeric constants). The numeric values being compared can be computed or they can be discrete intervals in a continuous numeric input.

Compatibility Rule

An Oracle Configurator Developer rule type to establish a relationship among features in the Model that specifies the allowable combinations of options. See also, Property-based Compatibility Rule.

Compatibility Table

A type of compatibility relationship where the allowable combination of options are explicitly enumerated.

Component

Represents a configurable element in a product. An element of the Model structure, typically containing features. May correspond to one screen of selections in an Oracle runtime configurator.

Component Set

An element of the Model that contains a number of components of the same type, where each component of the set is independently configured.

Configuration

A specific set of specifications for a product, resulting from selections made in an Oracle runtime configurator.

Configuration Model

The model structure and rules-based content of an Oracle runtime configurator. The configuration model is constructed and maintained using Oracle Configurator Developer, and is interpreted at runtime by the Active Model.

Configuration Rules

The Oracle Configurator Developer logic rules and numeric rules available for defining configurations.

Configurator

The part of an application that provides custom configuration capabilities.

Constraint Rule

An Oracle Configurator Developer rule type to create a logical relationship among features and options. See also Rules.

Contributes to

An Oracle Configurator Developer numeric rule type for accumulating a total value.

Consumes from

An Oracle Configurator Developer numeric rule type for specifying the quantity of a resource used.

CRM

Customer Relationship Management. The aspect of the enterprise that involves contact with customers, from lead generation to support services.

Customer

The person or persons for whom products are configured by end users of the Oracle Configurator or other ERP and CRM applications.

Customer-centric Extensions

See Customer-centric Views.

Customer-centric Views

Optional extensions to core functionality that supplement product selection with rules for pre-selection, validation, and intelligent views. View capabilities include generative geometry, drawings, sketches and schematics, charts, performance analyses, and ROI calculations.

Customer Requirements

The needs of the customer that serve as the basis for determining the configuration of products, systems, and/or services. Also called Needs Assessment.

Data Import

Populating the Oracle Configurator schema with enterprise data from ERP or legacy systems via import tables.

Data Integration Object

Data Integration Object. A server in the runtime application that creates and manages the interface between the client (usually a user interface like the Active User Interface) and the Oracle Configurator schema.

Data Maintenance Environment

The environment in which the Oracle runtime configurator data is maintained.

Data Replication

The activity of downloading and uploading configuration, quote, and order data between the Oracle Configurator schema on the enterprise server and Oracle Configurator Mobile Database on end-user mobile laptop PCs. See also Data Synchronization.

Datasource

A programmatic reference to a database. Referred to by a datasource name, or DSN.

Data Synchronization

A process for matching the data in the Oracle Configurator schema and the data available to client processes such as the Oracle SellingPoint application. See also Data Replication.

Default

The automatic selection of an option based on the pre-selection rules or the selection of another option.

Defaults

An Oracle Configurator Developer logic rule to determine the logic state of features or options in a default relation to other features and options. For instance, if you set A to True by selecting it, B becomes true (selected) if it is available (not false) and can be set to True without contradicting a non-default rule or a previous default setting for B.

Defect

A failure in a product to satisfy the users' requirements. Defects are prioritized as critical, major, or minor, and fixes range from corrections or workarounds to enhancements. Also known as a "bug".

Defect Tracking

A system of identifying defects for managing additional tests, testing, and approval for release to users.

Deliverable

A work product that is specified for review and delivery.

Demonstration

A presentation of the tested application, showing a particular usage scenario.

Design Chart

An Oracle Configurator Developer rule type for defining advanced Explicit Compatibilities interactively in a chart view.

Design Review

A technical review that focuses on application or system design.

Developer

The tool (Oracle Configurator Developer) used to create configuration models. The person who uses Oracle Configurator Developer to create a configurator. See also Implementer

DIO

See Data Integration Object.

End User

The ultimate user of the Oracle runtime configurator. The types of end users vary by project but may include salespeople or distributors, administrative office staff, marketing personnel, order entry personnel, product engineers, or customers directly accessing the application via web or kiosk.

Enterprise

The systems and resources of a business.

Environment

The arena in which software tools are used, such as operating system, applications, and server processes.

ERP

Enterprise Resource Planning. A software system and process that provides automation for the customer's back-room operations, including order processing.

Excludes

An Oracle Configurator Developer rule type for determining the logic state of features or options in an excluding relation to other features and options. For instance, if you set A to True, B becomes false, since it is not allowed when A is true. If you set A to False, there is no effect on B, meaning it could be true, false, or unknown.

Extended Functionality

A release after delivery of core functionality that extends that core functionality with customer-centric views, more complex proposal generation, discounting, quoting, and expanded integration with ERP, CRM, and third-party software.

Feature

An element of the Model structure. A configurable parameter of a component. Features can either have a value (numeric or boolean) or enumerated options.

Functional Companion

An object associated with a component that supplies methods that can be used to initialize, validate and generate customer-centric views and outputs for the configuration.

Functional Specification

Document describing the functionality of the application based on user requirements.

Incremental Construction

The process of organizing the construction of the application into builds, where each build is designed to meet a specified portion of the overall requirements and is unit tested.

Implementation

The stage in a project between defining the problem by selecting a configuration technology vendor, such as Oracle, and deploying the completed sales configuration application. The Implementation stage includes gathering requirements, defining test cases, designing the application, constructing and testing the application, and delivering it to users.

Implementer

The person who uses Oracle Configurator Developer to build the model structure, rules, and UI customizations that make up an Oracle runtime configurator.

Implies

An Oracle Configurator Developer logic rule type that determines the logic state of features or options in an implied relation to other features and options. For instance, if you set A to True by selecting it, B becomes true, since selecting A implies that B is also selected. If you set A to False by deselecting it, there is no effect on B, meaning it could be true false or unknown based on other relations B participates in. And if you set B to True by selecting it, there is no effect on A, meaning it could be true false or unknown based on other relations A participates in. But if you set B to False by deselecting it, the relation of A implies B is preserved only by having A be false (deselected) as well.

Import Tables

Tables mirroring the Oracle Configurator schema Item Master structure, but without integrity constraints. Import Tables allow batch population of the Oracle Configurator schema Item Master. Import Tables are used in conjunction with extractions from Oracle Applications or legacy data to create, update, or delete records in the Oracle Configurator schema Item Master.

Install

A program that sets up the local machine and installs the application for testing and use.

Integration

The process of combining multiple software components and making them work together.

Integration Testing

Testing the interaction among software programs that have been integrated into an application or system.

Intelligent Views

Configuration output, such as reports, graphs, schematics, and diagrams, that help to illustrate the value proposition of what is being sold.

Item Master

A table in the Oracle Configurator schema containing data used to structure the product. Data in the item master is either entered manually or imported from Oracle Applications or legacy data.

Item Type

A table in the Oracle Configurator schema containing data used to classify the product data in the item master table.

Log File

A file containing errors, warnings and other information output by the running application.

Logic Rules

Logic rules directly or indirectly set the logical state (true, false, or unknown) of features and options in the Model.

There are four (4) primary logic rules: Implies, Requires, Excludes, and Negates. Each of these rules takes a list of features or options as operands. See also Logic, Implies, Requires, Excludes, and Negates.

Maintainability

The characteristic of a product or process to allow straightforward maintenance, alteration, and extension. Maintainability must be built into the product or process from inception.

Maintenance

The effort of keeping a system running once it has been deployed, through bug fixes, procedure changes, infrastructure adjustments, data replication schedules, etc.

Maintenance Guide

A guide for maintaining a specific application or system. The maintenance guide covers all aspects of maintenance described in the generic Maintenance Plan.

Maintenance Plan

A document that outlines what is required for successful maintenance, and who is responsible for all the actions and deliverables of carrying out maintenance on a system.

MDUI

See Model-driven UI.

Mobile Database

See Oracle Configurator Mobile Database.

Model

The entire hierarchical “tree” view of all the data required for configurations, including model structure, variables such as resources and totals, and elements in support of intermediary rules. May consist of BOM Items.

Model-driven UI

The graphical views of the model structure and rules generated by the Active UI to present end users with interactive product selection based on configuration models.

Model Structure

Hierarchical, “tree” view of data in terms of product elements (Models, Products Components, Features, Options, BOM Models, BOM OptionClasses, BOM StandardItems, Resources, and Totals). May include reusable components.

MRP

Manufacturing Resource Planning. A software system and process for monitoring and maintaining the customer's manufacturing systems.

Negates

An Oracle Configurator Developer logic rule type that determines the logic state of features or options in a negating relation to other features and options. For instance, if you set one item in the relationship to True, the other item must be false. And if you set one item to False, the other item must be true.

Node

The place in a Model occupied by a component, feature, option or variable, BOM Model, BOM OptionClass, or BOM StandardItem.

Numeric Rules

Rules that are used to set the global parameters specified in product structuring. See also, Contributes to and Consumes from.

OC

See Oracle Configurator.

Opportunity

The workspace in the Oracle SellingPoint application and Oracle Sales Online in which products, systems, and/or services are configured, quotes and proposals are generated, and orders are submitted.

Option

An element of the Model. A choice for the value of an enumerated feature.

A logical selection made by the end user when configuring a component.

Oracle Configurator

The product family consisting of development tools and runtime applications such as Oracle Configurator schema, Oracle Configurator Developer, Oracle Configurator window, and Oracle SellingPoint application. Also the Oracle runtime configurator variously packaged for use in networked, mobile, or web deployments.

Oracle Configurator Architecture

The application runtime architecture consists of the Active User Interface, the Active Model, and the Oracle Configurator schema or Oracle Configurator Mobile Database. The application development architecture consists of Oracle Configurator Developer and the Oracle Configurator schema, with test instances of an Oracle runtime configurator.

Oracle Configurator Developer

The suite of tools in the Oracle Configurator product family for constructing and maintaining configurators.

Oracle Configuration Interface Object (CIO)

A server in the runtime application that creates and manages the interface between the client (usually a user interface like the Active User Interface) and the underlying representation of model structure and rules in the Active Model.

CIO protocols support creating and navigating the Model, querying and modifying selection states, and saving and restoring configurations.

Oracle Configurator Mobile Database

The runtime version of the standard Oracle Configurator schema that manages data for the configuration model in a mobile deployment. The runtime schema includes customer, product, and pricing data as well as data created during operation of an Oracle Configurator.

Oracle Configurator Schema

The implementation version of the standard Oracle runtime configurator data-warehousing schema that manages data for the configuration model. The implementation schema includes all the data required for the runtime system as well as specific tables used during the construction of the configurator.

Oracle SellingPoint Application

The test application generated by Oracle Configurator Developer. Also a full configuration environment with opportunity management, quotes, and proposals for networked or mobile deployments.

Output

The output generated by a configurator, such as quotes, proposals, bills of material (BOM), and customer-centric views.

PDM

Product Data Management. A software system that manages the version control of product data.

Populator

An entity in the Oracle Configurator Developer that defines how to create a Model from information in the item master.

Pre-selection

The default state in a configurator that defines an initial selection of components, features, and options for configuration.

A process that is implemented to select the initial element(s) of the configuration.

Principal Design Consultant

Member of the project team responsible for architecting the design of the application.

Product

Whatever is subjected to configuration and is the output of the application.

The root element of the Model.

Product Family

A collection of products or product lines, which are organized as a group by a provider or manufacturer.

Project

The workspace in Oracle Configurator Developer in which configurators are constructed

Project Manager

A member of the project team who is responsible for directing the project during implementation.

Project Plan

A document that outlines the logistics of successfully implementing the project, including the schedule.

Property

A named value associated with an object in the Model or the item master. A set of properties may be associated with an item type.

Property-based Compatibility Rule

A kind of compatibility relationship where the allowable combinations of options are specified implicitly by relationships among property values of the options.

Prototype

A construction technique in which a preliminary version of the application, or part of the application, is built to facilitate user feedback, to prove feasibility or examine other implementation issues.

Reference

The use of a reusable component within the Model. Not implemented in Release 11*i* or before.

Regression Test

An automated test that ensures the newest build still meets previously tested requirements and functionality.

Requires

An Oracle Configurator Developer logic rule type that determines the logic state of features or options in a requirement relation to other features and options. For instance, if you set one item in the relationship to True, the other item is required to be true as well. And if you set one item to False, the other item must be false as well.

Resource

Staff or materials available or needed within an enterprise.

A variable in the Model used to maintain the balance of features not consuming more of a specific resource than has been provided by other features.

Reusable Component

A component that is referenced from multiple locations in the Model. Not implemented in Release 11*i* or before.

Reusability

The extent to and ease with which parts of a system can be put to use in other systems.

Rules

Also called business rules or configuration rules. Constraints applied among elements of the product to ensure that defined relationships are preserved during configuration. Elements of the product are components, features, and options. Rules express logic, numeric parameters, implicit compatibility, or explicit compatibility. Rules are used to provide pre-selection and validation capability in an application.

See also Logic Rules and Numeric Rules.

Runtime

The environment and context in which applications are run or used, rather than developed.

Sales Configuration

A part of the sales process to which configuration technology has been applied in order to increase sales effectiveness and decrease order errors. Commonly identifies needs assessment and product configuration.

Server

Centrally located software processes or hardware, shared by clients.

Solution

The deployed system as a response to a problem or problems.

System

The hardware and software components and infrastructure integrated to satisfy functional and performance requirements.

Test Case

A description of inputs, execution instructions, and expected results, which are created for the purpose of determining whether a specific software feature works correctly or a specific requirement has been met.

Total

A variable in the Model used to accumulate a numeric total, such as total price or total weight.

Undetermined

The logic state that is neither true nor false, but unknown at the time a logic rule is executed. This logic state is also referred to as Available, especially when considered from the point of view of the Oracle runtime configurator end user.

Unit Test

Execution of individual routines and modules by the application implementer or by an independent test consultant for the purposes of finding defects.

Update

Moving a production configurator to a new version of configuration model.

Upgrade

Moving the configurator to a new release of Oracle Configurator.

User

The person using the Oracle Configurator Developer tools and methods to build an Oracle runtime configurator. See also end user.

User Interface

The visible part of the application, including menus, dialog boxes, and other on-screen elements. The part of a system where the user interacts with the software.

User Requirements

A description of what the Oracle Configurator or Oracle SellingPoint application is expected to do from the end user's perspective.

User's Guide

Documentation on using the application or configurator to solve the intended problem.

Validation

Tests that ensure that the configured components will meet specific performance or acceptance criteria.

A type of functional companion that is implemented to ensure that the configured components will meet specific performance or acceptance criteria.

Variable

Parts of the Model that are represented by Totals, Resources, or numeric Features.

Verification

Tests that check whether the result agrees with the specification.

Glossary of Acronyms

API

Application Programming Interface

ATP

Available to Promise

BOM

Bill of Material

CIO

Configuration Interface Object

CM

Configuration Management

COM

Component Object Model

CRM

Customer Relationship Management

DBMS

Database Management System

DCOM

Distributed Component Object Modeling

DHTML

Dynamic Hypertext Markup Language

DIO

Data Integration Object

DLL

Dynamically Linked Library

DXF

Drawing Exchange Format (AutoCAD drawings)

ECO

Engineering Change Order

ERM

Enterprise Relationship Management

ERP

Enterprise Resource Planning

ESD

Electronic Software Distribution

ESP

External Service Provider

ESS

Enterprise Selling System

HT

High Tech

HTML

Hypertext Markup Language

IP

Industrial Products

IS

Information Services

ISS

Interactive Selling System

ISV

Independent Software Vendor

LAN

Local Area Network

MAPI

Messaging Application Programming Interface

MC/S

Mobile Client / Server System

MDUI

Model-Driven User Interface

MES

Marketing Encyclopedia System (Catalog)

MIS

Management Information Systems

MRP

Manufacturing Resource Planning

MS

Microsoft

OC

Oracle Configurator

OCX

Object Control File, OLE custom controls

ODBC

Open Database Connectivity

OLE

Object linking and embedding

OMS

Opportunity Management System

OOD

Object-Oriented Design

ORB

Object Request Broker

PDM

Product Data Management

PIA

Project Impact Assessment

POS

Point of Sale

QA

Quality Assurance

RAD

Rapid Application Development

RDBMS

Relational Database Management System

RFQ

Request for Quote

ROI

Return on Investment

SAS

Sales Analysis System

SCM

Supply Chain Management

SCS

Sales Configuration System

SE

Sales Engineer

SFA

Sales Force Automation

SI

System Integrator

SOT

Strategic Options Theory

SQA

Software Quality Assurance

SQL

Structured Query Language

TERM

Technology-Enabled Relationship Management

TES

Technology-Enabled Selling

UI

User Interface

VAR

Value-Added Reseller

VB

Microsoft Visual Basic

WAN

Wide Area Network

WIP

Work In Progress

Index

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W