

## **What's New in Corticon**



---

# Notices

---

© 2016 Progress Software Corporation and/or one of its subsidiaries or affiliates. All rights reserved.

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

Business Making Progress, Corticon, DataDirect (and design), DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, Deliver More Than Expected, Icenium, Kendo UI, Making Software Work Together, NativeScript, OpenEdge, Powered by Progress, Progress, Progress Software Business Making Progress, Progress Software Developers Network, Rollbase, RulesCloud, RulesWorld, SequeLink, Sitefinity (and Design), SpeedScript, Stylus Studio, TeamPulse, Telerik, Telerik (and Design), Test Studio, and WebSpeed are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries. AccelEvent, AppsAlive, AppServer, BravePoint, BusinessEdge, DataDirect Spy, DataDirect SupportLink, Future Proof, High Performance Integration, OpenAccess, ProDataSet, Progress Arcade, Progress Profiles, Progress Results, Progress RFID, Progress Software, ProVision, PSE Pro, SectorAlliance, Sitefinity, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, WebClient, and Who Makes Progress are trademarks or service marks of Progress Software Corporation and/or its subsidiaries or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

Please refer to the Release Notes applicable to the particular Progress product release for any third-party acknowledgements required to be provided in the documentation associated with the Progress product.



---

# Overview of Progress Corticon

---

Progress® Corticon® is the Business Rules Management System with the patented rules engine that enables you to automate sophisticated decision processes—without having to write code.

## Progress Corticon products

Progress Corticon distinguishes its development toolsets from its server deployment environments.

- **Corticon Studio** is the Windows-based development environment for creating and testing business rules:
  - When installed as a standalone application, Corticon Studio provides a complete Eclipse development environment for Corticon in the **Corticon Designer** perspective. You can use this Eclipse installation as the basis for adding other Eclipse tools.
  - When installed into an existing Eclipse environment such as the **Progress Developer Studio**, Corticon integrates with Progress OpenEdge which enables you to develop Corticon applications in the **Corticon Designer**.

---

**Note:** Refer to the *Corticon Installation Guide* for details about integrating Corticon Studio into an existing Eclipse environment.

---

- **Corticon Servers** implement web services and in-process servers for deploying business rules defined in Corticon Studio:
  - **Corticon Server for Java** is supported on various application servers, and client web browsers. After you install it on a supported Windows platform, its deployment artifacts can be redeployed on various UNIX and Linux web service platforms as Corticon Decision Services.
  - **Corticon Server for .NET** facilitates deployment of Corticon Decision Services on Windows .NET Framework and Microsoft Internet Information Services (IIS).

- **Corticon Web Console** enables administration of multiple remote Corticon Servers. A Web Console server is deployed into a Progress Application Server, and then is accessed by users through authenticated web browser connections.

### Use with other Progress Software products

Corticon releases coordinate with other Progress Software releases:

- [Progress OpenEdge](#) is available as a database. You can read from and write to an OpenEdge database from Corticon Decision Services. When Progress Developer Studio for OpenEdge and Progress Corticon Studio are integrated into a single Eclipse instance, you can use the capabilities of integrated Corticon business rules in Progress OpenEdge. See the OpenEdge document [OpenEdge Business Rules](#) for more information. OpenEdge is a separately licensed Progress Software product.
- [Progress DataDirect Cloud](#) enables simple, and fast connections to cloud data regardless of source. DataDirect Cloud is a separately licensed Progress Software product.
- [Progress RollBase](#) is a low-code platform for rapid development of business applications for cloud and on premise deployment. When combined with Corticon, you integrate the power of business rules with your business applications. Rollbase is a separately licensed Progress Software product.

## What's new and changed in Corticon 5.6

---

This section summarizes the new, enhanced, and changed features in Progress® Corticon® 5.6.

For details, see the following topics:

- [Improved process for custom extended operators and service callouts](#)
- [Enhancements to Corticon Studio Tester](#)
- [Find precise location of problems in editors](#)
- [Finding entity, attribute and association references](#)
- [Create graphs of attribute and logical dependencies](#)
- [Use Natural Language expressions on filters](#)
- [Start designing rules using Natural Language](#)
- [Add comments to Rulesheets](#)
- [Create flexible Corticon Studio reports](#)
- [REST API to return a Decision Service's Vocabulary metadata](#)
- [REST API to pass a Decision Service as a URL parameter](#)
- [Improved Web Console](#)
- [Settings to use Rule Execution Recording on Decision Services](#)
- [Deployment security: Authentication and encryption](#)
- [Automatic building and validation of projects](#)

- [Simplified installation](#)
- [Miscellany](#)

## Improved process for custom extended operators and service callouts

When you are creating business rules, you sometimes need to perform operations that are not built natively into Corticon. For example you may need a complex mathematical formula or to retrieve data from an external web service. Corticon provides the ability to add custom extensions for just such purposes. This has been present in previous releases but has been made easier to use in Corticon 5.6.

Extensions are written as custom Java code that you package into one or more JAR files. In earlier releases, you had to modify Corticon's Java classpath and perform other configuration steps to use an extension in your Decision Services. In Corticon 5.6, you simply add extension jar files to your rule project and Corticon bundles them into the EDS file for your Decision Service.

This ease of adding extensions makes it easier to develop extensions yourself or to use open source extensions that you download from the Corticon community.

By bundling extensions with EDS files, the EDS file becomes *self-contained*. You can deploy it to a Corticon Server without modifying the server's classpath. It also allows you to have different Decision Services, or versions of Decisions Services, running that use different versions of an extension.

When developing an extension, it needs to implement one or more Java interfaces that Corticon has defined. These interfaces have not changed in Corticon 5.6. What is new is the addition of Java annotations to describe the extension. The use of annotations eliminates the need for additional configuration files.

When developing a project in Corticon Studio you can add extensions to your project through the project's **Properties** dialog box, so that they are available for development and running rule tests. The **Package and Deploy** wizard in Corticon Studio will include any extensions used by the project into the EDS file it generates or deploys. If you want to script the building of your EDS files, you can also use the Corticon ANT scripts to package extensions into EDS files.

For compatibility with previous releases you can still place extension JAR files on the Corticon classpath so that they are available to all Decision Services.

Extensions can be created in the Java development environment included in Corticon Studio, or you can use another IDE.

There are two types of Corticon extensions:

- **Extended operators** - Operators are used when defining conditions and actions in a Rulesheet. While Corticon has a large built-in set of operators, you can expand this set by adding custom operators. Operators can operate on individual attributes, collections or sequences. Examples include:
  - Financial functions, such as net present value, and loan amortization
  - Statistical functions, such as standard deviation, and permutations
  - Engineering functions, such as pi, sine, and cosine
- **Service callouts** - Callouts can be used in a ruleflow to retrieve, modify, or store data that is being processed by the rules. The most common use is to access data in a database or external web service. For example, if your Ruleflow needs to look up an applicant's credit rating, the service callout can have a step in the Ruleflow processing that calls out to a trusted realtime ratings provider, and then adds the response back into the decision processing.

For details on creating extended operators and service callouts, see *"Getting started creating custom extensions" topic of the Guide to Creating Corticon Extensions*. You can also examine the sample extensions bundled with Corticon Studio in the **Advanced** section of the **Help > Samples** page:

**Getting Started**

**Intermediate**

**Advanced**

**Extended Operators** - This sample demonstrates how to extend Corticon's capabilities with custom rule operators.

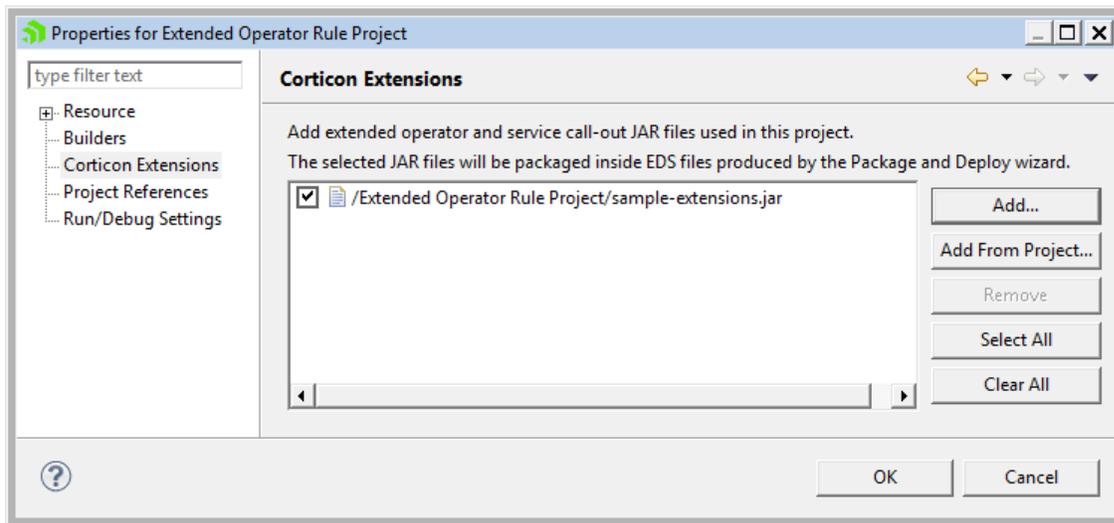
**Service Callouts** - This sample demonstrates how to extend Corticon's capabilities with custom service callouts.

Once your extended operator and service callout `.java` files are ready, you compile them into classes, package them into JARs, and then stage the JARs for access by projects.

## A look at using extensions in a project

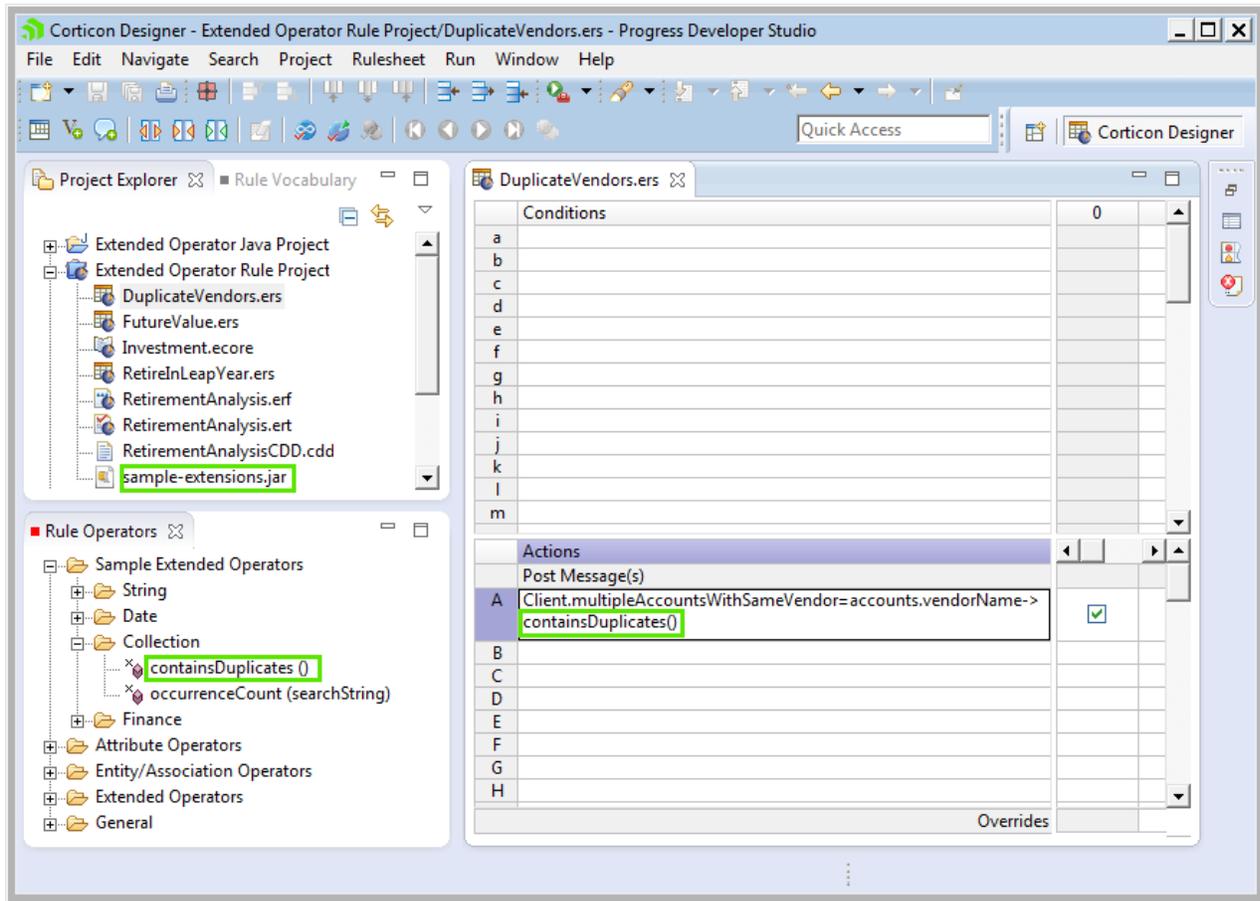
### Using extended operators in a Rulesheet

To add an extension JAR to a project, open the project, select **Project > Properties**, and then click **Corticon Extensions**. Click **Add** to navigate to the JAR you want, and then click **Open** to add it into the list, as shown:



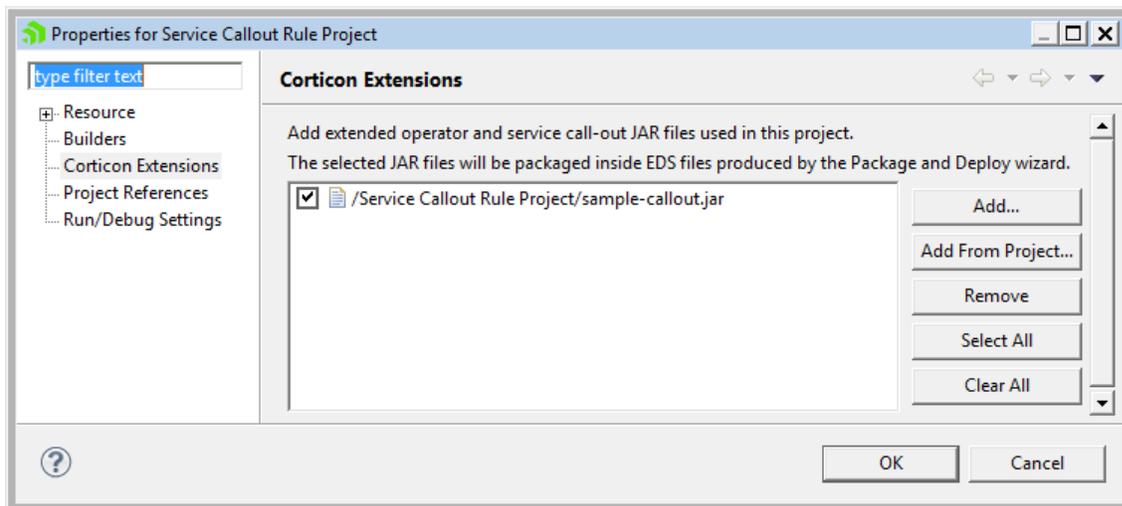
Click **OK**. The JAR is added to the project's folder and listed in the **Project Explorer**.

When a Rulesheet in this project is opened, any custom operator extensions in the JAR are added to the **Rule Operators** section, so that you can drag and drop a custom operator to the Rulesheet, as shown:

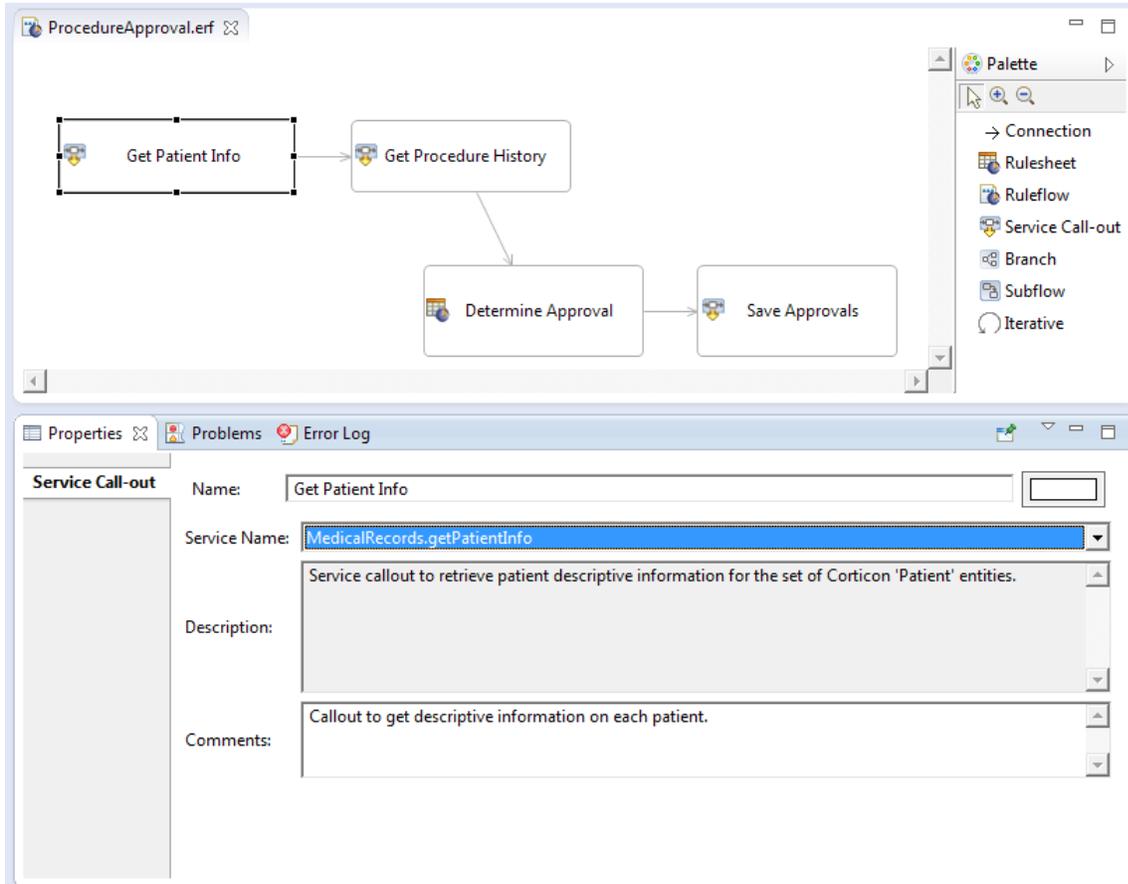


### Using a Service Callout in a Ruleflow

Add a service callout JAR file to a project using the same procedure as an extension JAR -- open the project, select **Project > Properties**, and then click **Corticon Extensions**. Click **Add** to navigate to the JAR you want, and then click **Open** to add it into the list, as shown:



Because the JAR file contains a service callout extension, the callout is available when creating a Ruleflow, as shown:

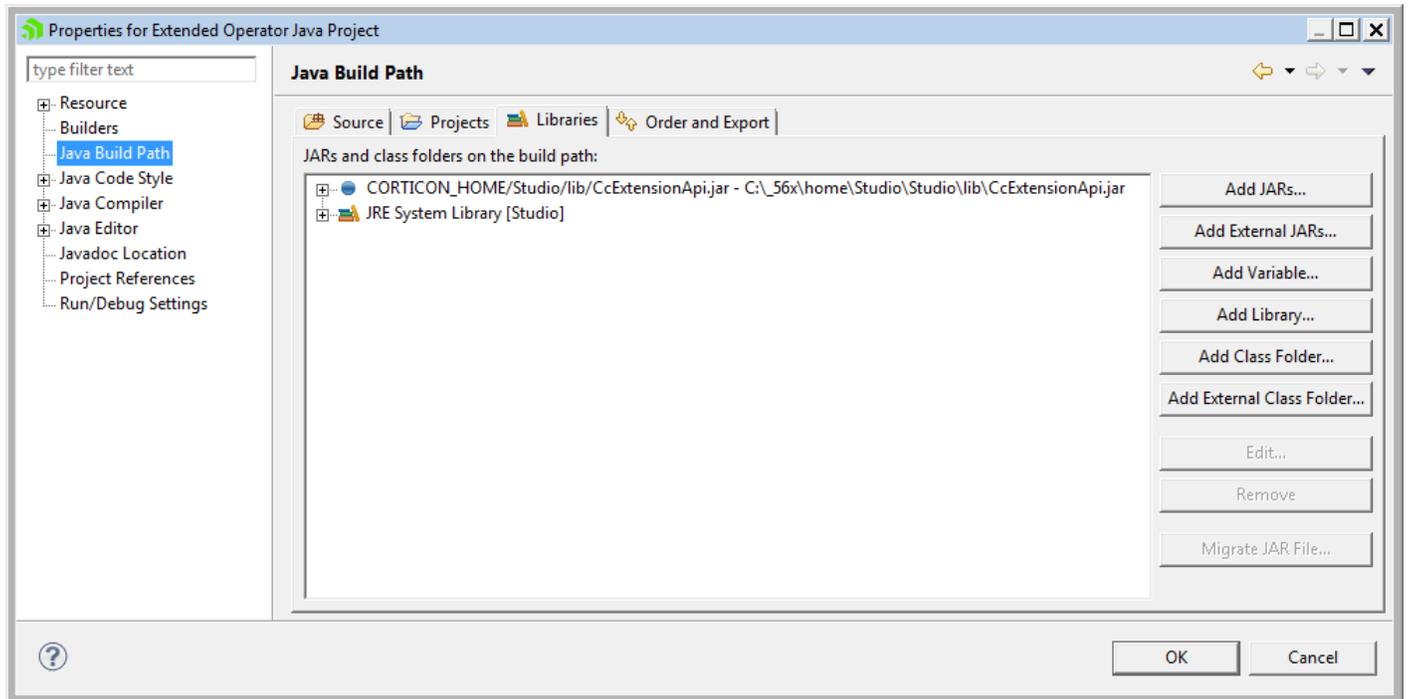


The description and comments that you add to a service callout's source files provide modelers with key information about the service callout.

## A look at creating extensions in Corticon Studio

Both the Extended Operator and Service Callout samples contain Java projects demonstrating how to create a Java extension. These are standard Java projects. Each has the CorticonCorticon JAR file that defines the API for Corticon extensions, `CcExtensionApi.jar` on its build path.

When developing extensions in Corticon Studio, add this JAR to the **Java Build Path** using the predefined Eclipse variable `CORTICON_HOME`. For example:



See the complete set of topics in *the Guide to Creating Corticon Extensions*.

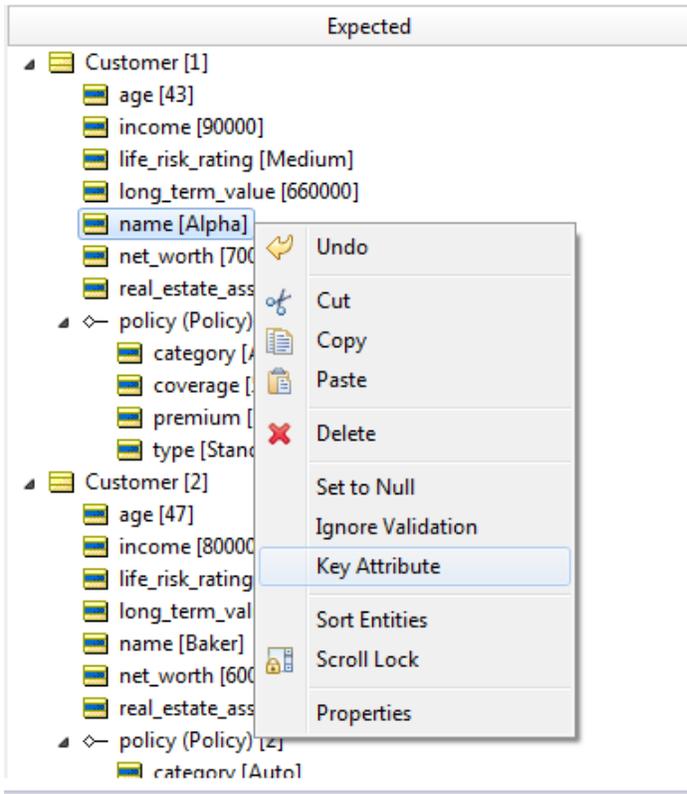
## Enhancements to Corticon Studio Tester

In this release, the Ruletest feature has improvements in navigation, and difference detection as described in this section.

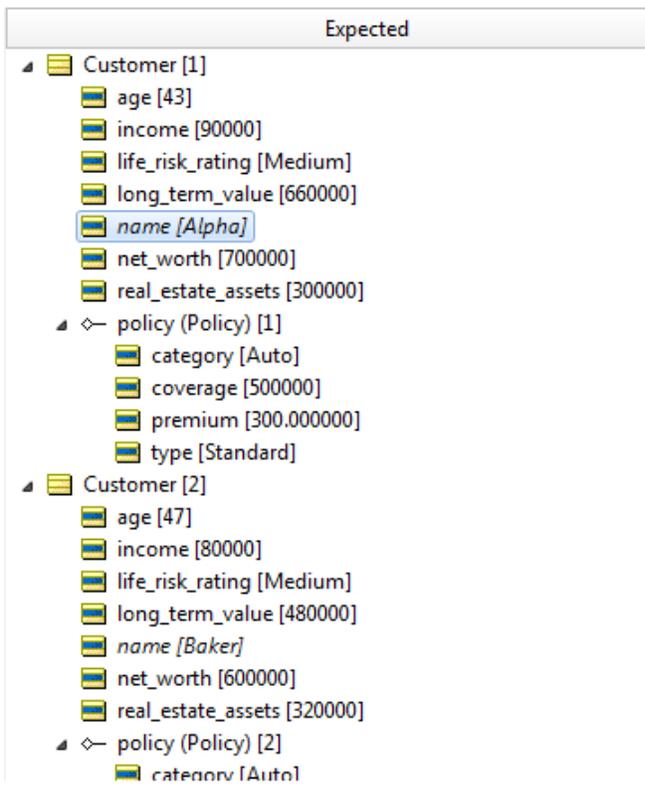
### Improved difference detection in Ruletests

The execution of Ruletests can, in some cases, erroneously detect differences between the Output and Expected results. This typically occurs in Rulesheets that add new entities to collections. The unsorted nature of collections makes it difficult to match the collections in the Output and Expected results with complete accuracy. An optional feature is now available to help when you encounter problems with test failures due to the randomness of entity ordering. To avoid this problem, you can specify certain attributes as *key attributes* that will assist the comparison algorithm, so that the validation linking entities in both panels are chosen based on the key values. For example an ID attribute that will not change across test runs could be used as a key attribute.

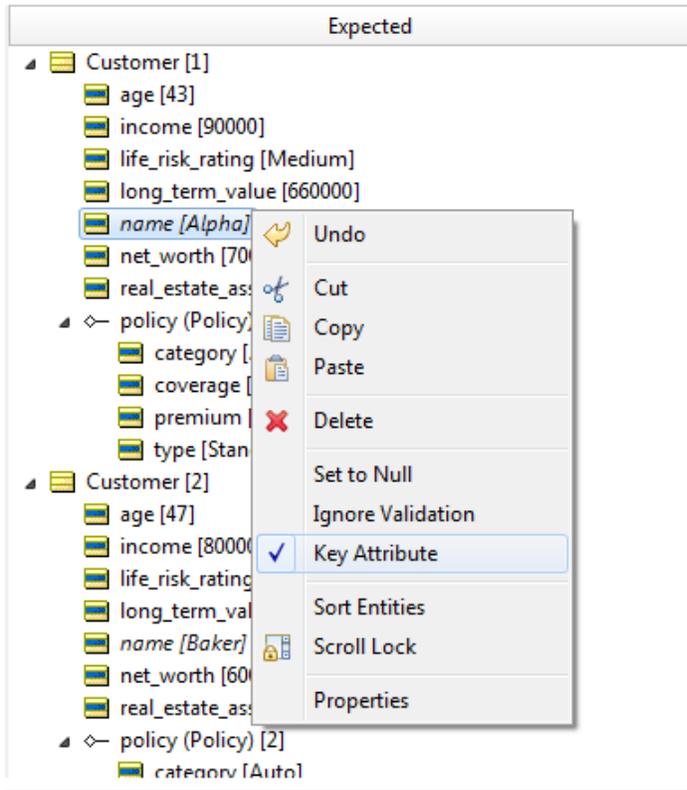
To set a key attribute, right-click on it in the Expected panel, and then select **Key Attribute**, as shown:



Key attributes are shown in italics in the current entity as well as in all other corresponding entities in the Expected panel, as shown:



To remove a key attribute, right-click on it in the Expected panel, and then select **Key Attribute** to clear the setting, as shown:



Setting multiple key attributes will attempt to match the full set.

This topic was added as *"Using key attributes to improve difference detection in Ruletests"* in the *Quick Reference Guide*.

## Improved Ruletest scrolling and navigation to differences

When reviewing the results of a test run, two navigation features make it easier to find differences:

- Synchronized scrolling** - When you slide the scroll tab in the Ruletest panels, the three columns do not by default move together, making alignment of data points difficult. You can set (or unset) synchronized scrolling of columns by either right-clicking any of the Ruletest panels and then selecting **Scroll Lock**, or by clicking  in the Corticon Studio toolbar. Once set to synchronize, all panels synchronize their scrolling, even advancing across collapsed entities and associations to stay synchronized on the first displayed line.
- Navigation to differences** - The Ruletest window now shows the number of discovered differences and controls to navigate across items. For example, in the upper right of the Ruletest window, the following image shows that the test results have identified six differences:

Differences: 6 

The four buttons take you to the first, previous, next, and last discovered difference respectively.

This material was added as the topic *"Navigating in Ruletest Expected comparison results"* in the *Rule Modeling Guide*.

## Property to disable the trimming of String values

In the Ruletest feature that lets you set expected values for tests, sometimes leading or trailing blanks on String values (called *whitespace*) cause imprecise matching to the output from rules. While the default behavior of trimming the whitespace is typically preferred, a new property, when set to `false`, allows you to tell Corticon Studio to not perform trimming, and thus reduce these validation mismatches.

The default behavior is apparent when copying the output to the Expected column, as that action strips whitespaces, and often reveals apparent mismatches immediately. The default value is `true`.

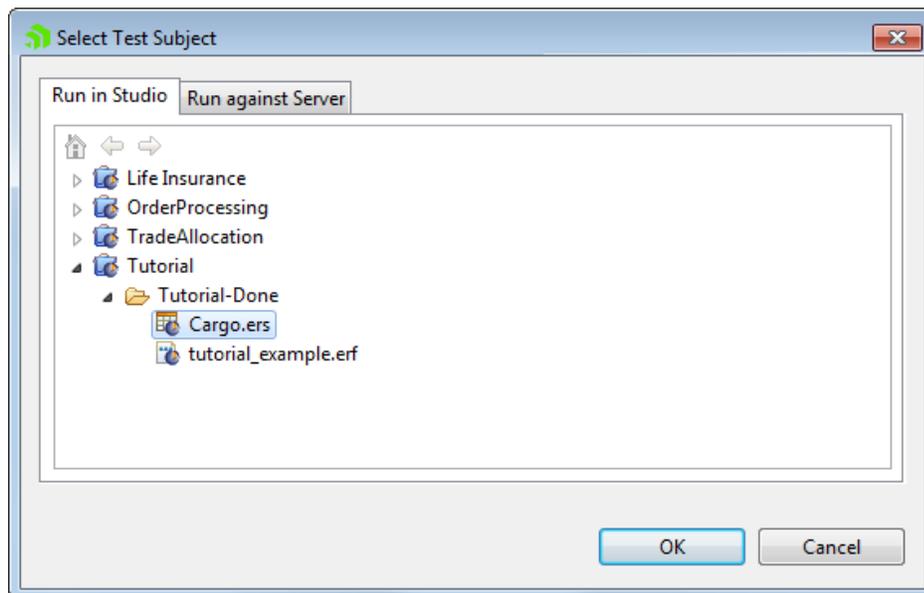
```
com.corticon.testers.trimstringvalues=true
```

This information was added to the *Corticon Studio Properties* in the *Integration and Deployment Guide*, and referenced in the topic "Testing rule scenarios in the Ruletest Expected panel" in the *Rule Modeling Guide*

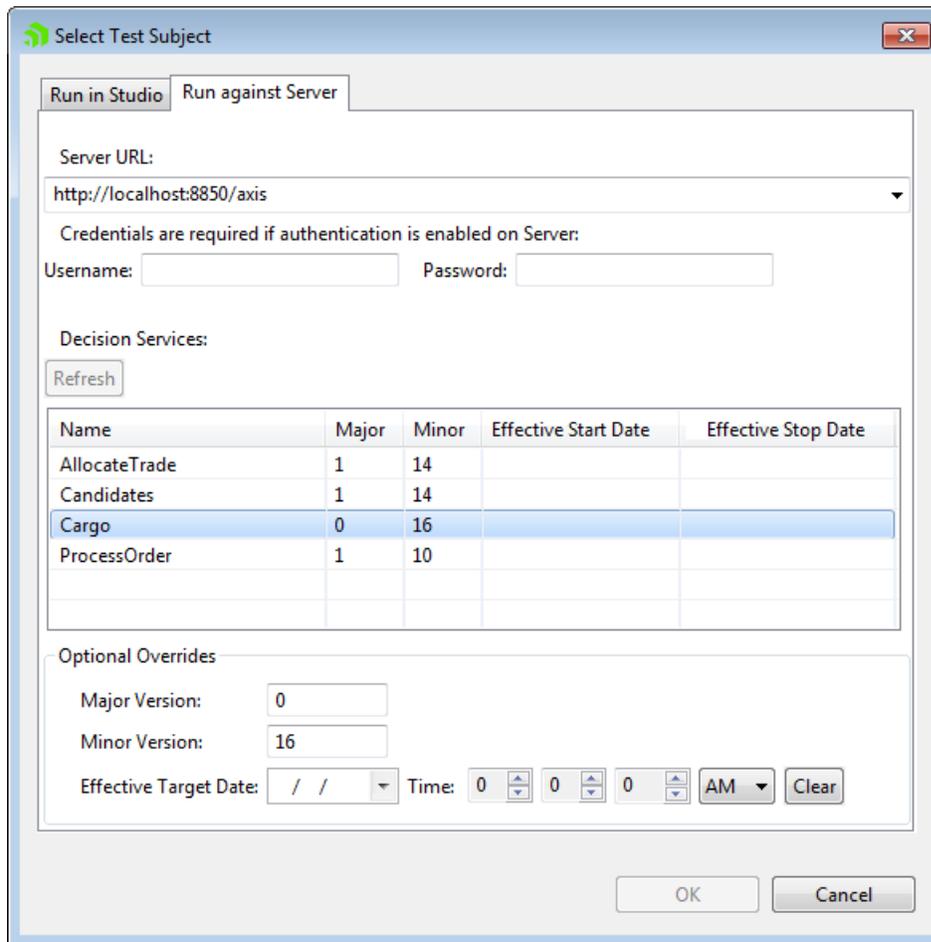
## Improvements to the selection of a test subject

The dialog box for changing a Ruletest's test subject has been streamlined. The features that were available in Corticon Studio 5.5.2 are still present yet the layout was modified to show the local and remote server specifications on separate panels.

When the dialog box opens -- whether within the **New Ruletest** procedure, or when changing the test subject of an existing testsheet -- the default tab is **Run in Studio**, as shown:



Clicking the **Run against Server** tab lets you enter or choose a Server, then reveal its deployed Decision Services, and choose optional overrides, as shown:



**Note:** You can directly edit or paste the test subject path on the testsheet. This feature, typically reserved for advanced users, rejects an invalid file name but might not catch all the subtleties of validation that is applied in the **Select Test Subject** dialog box.

This information was updated in related deployment guides and the *Quick Reference* guide.

## Find precise location of problems in editors

Corticon editors in previous releases annotated errors in the Studio's **Problems** view that let you double-click on a problem line to open the corresponding file in its editor. That works for modest sized files that have only a few components in the display area. But when files become large, the navigation to the error location can be a difficult.

This feature lets you double-click on a problem line to open the appropriate file, and then bring the specific location into view as well as give it the focus.

In the following illustration, the problem location is Rulesheet cell [b3598] of the 2DIM Rulesheet. Double-clicking the problem line opened the file to that precise location, as shown:

The screenshot displays the Corticon editor interface. The top window shows a Rulesheet with columns for cell indices (3597-3603) and rows for conditions (a, b, c) and actions (A, B). The cell at row 'b', column 3598 contains the value '76b' and is highlighted with a red border. Below the Rulesheet, the 'Problems' window is open, showing a list of errors and warnings. The error 'Invalid number format (possible overflow)' is highlighted, with its location specified as 'Rulesheet cell [b3598]'.

Conditions	3597	3598	3599	3600	3601	3602	3603
a Policy.beneficiaryAge	33	34	35	36	37	38	39
b Policy.applicantAge	76	76b	76	76	76	76	76
c							

Actions							
Post Message(s)							
A Policy.factor	0.4561	0.4585	0.461	0.4636	0.4664	0.4694	0.4725
B							

Description	Resource	Path	Location	Type
Errors (3 items)				
dPositionHiYiel is not a valid call on [bx].	AccountDerivations.ers	/TradeAllocation/Rules/Allocation	Action row [C]	Validation Message Marker
Invalid number format (possible overflow)	2DIM.ers	/ExcelMatrixImport	Rulesheet cell [b3598]	Problem
One or more referenced Rulesheet contain errors.	Allocation.eref	/TradeAllocation/Rules/Allocation	Unknown	Validation Message Marker
Warnings (1 item)				
One or more referenced Rulesheet contain errors.	Allocation.eref	/TradeAllocation/Rules/Allocation	Unknown	Validation Message Marker

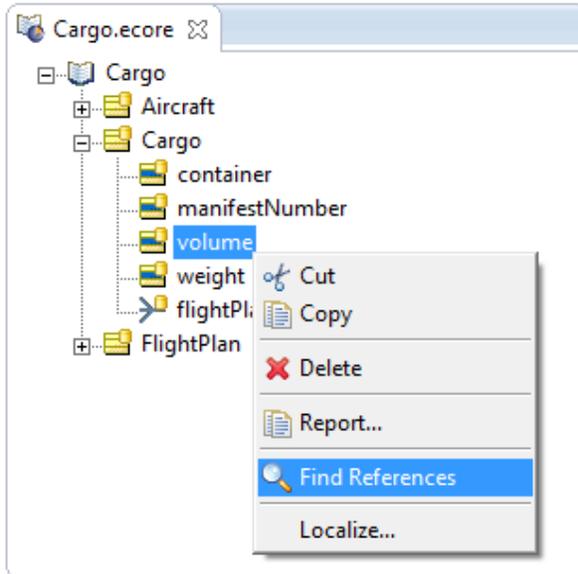
This functionality applies to Vocabularies, Rulesheets, Ruleflows, and Ruletests.

**Note:** When migrating projects from earlier releases, the marker metadata has not been captured. When you clear the existing problem list, and then perform a full build of the project, the location metadata that enables this feature is established.

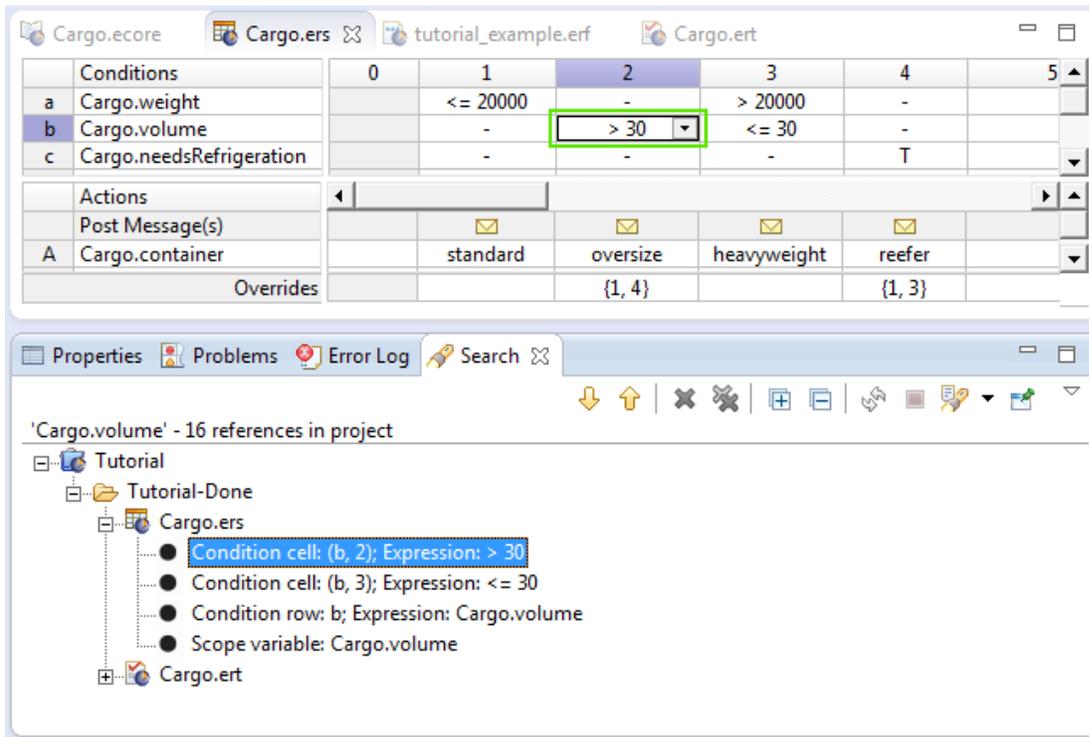
This material was added as the topic *"Precise location of problem markers in editors"* in the *Rule Modeling Guide*.

## Finding entity, attribute and association references

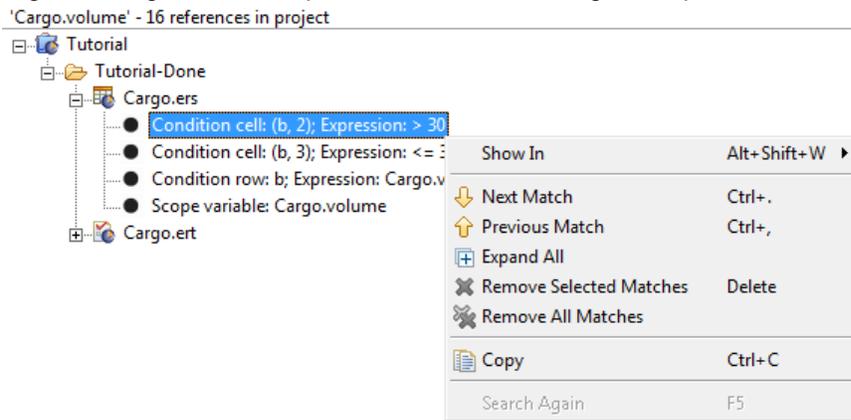
When working on large projects with many assets, it is a real convenience to find everywhere an entity, attribute, or association is referenced in the project. This allows you to do impact analysis before modifying a vocabulary. This feature reveals the vocabulary item's usage and impact, and lets you easily navigate to each instance. Now, just right-click on an entity, attribute, or association name in the Vocabulary editor, and then select **Find References** to initiate a search, as shown:



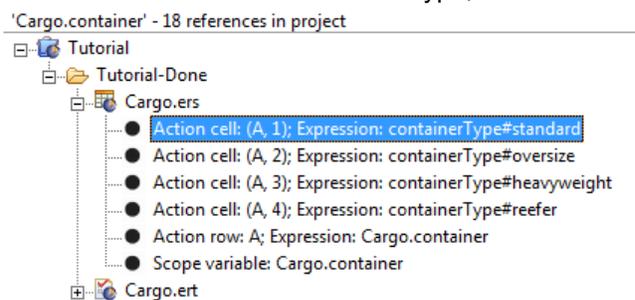
The search examines all Rulesheets, Ruleflows, and Ruletests in the project, then lists each match within each asset. Double-clicking on a match opens the asset and puts the focus at that location, as shown:



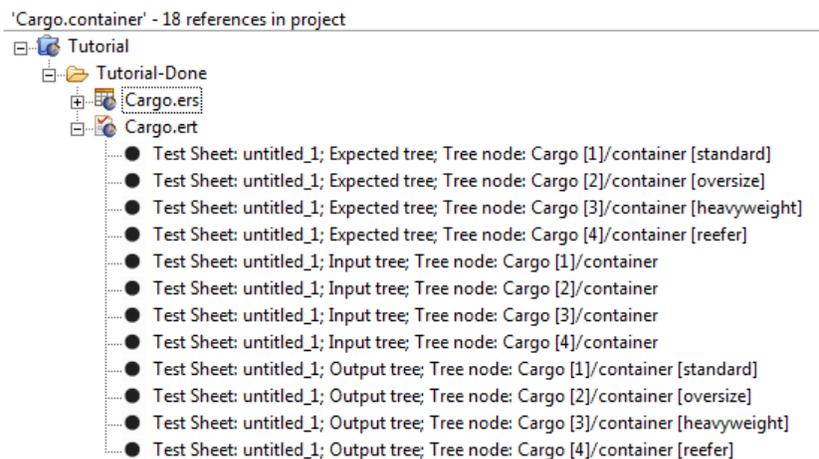
Right-clicking on a match presents a menu of navigation options and their keyboard shortcuts, as shown:



When the attribute is a custom data type, the search results indicate the Data Type Name and value, as shown:



When results are in Ruletests, the results detail the testsheet, tree, and node, as shown:

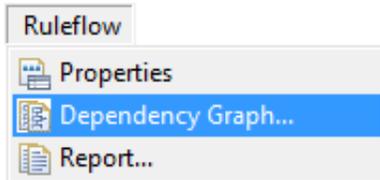


This material was added as a new topic, *"Finding references to an entity or attribute in a project"* in the *Rule Modeling Guide*.

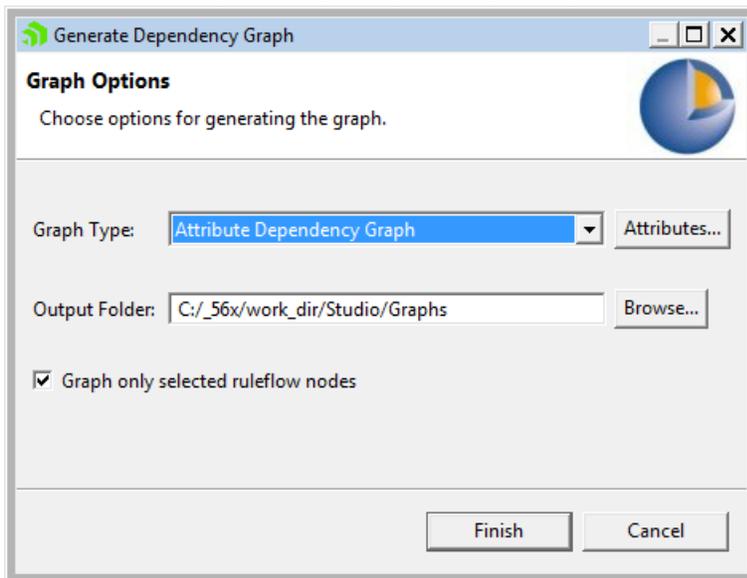
## Create graphs of attribute and logical dependencies

When working on large Ruleflows you often want to know the dependencies between the nodes in the Ruleflow. This can help you determine how best to order the nodes or detect unanticipated dependencies. Dependencies are identified by the attributes that are set or referenced in the nodes of a Ruleflow. You also often want to know how one or more attributes are used in a Ruleflow. Ruleflow graphing lets you see the dependencies and where attributes are used. This is useful for understanding a Ruleflow, debugging problems, and performing impact analysis when changing a vocabulary.

With the Ruleflow you want to graph open in its Studio editor, select the **Ruleflow** menu command **Dependency Graph**, as shown:



The **Generate Dependency Graph** dialog box opens:



Choose the type of graph you want, and the output folder. You can focus the analysis on just nodes you selected before opening the dialog, or all nodes on the Ruleflow canvas.

---

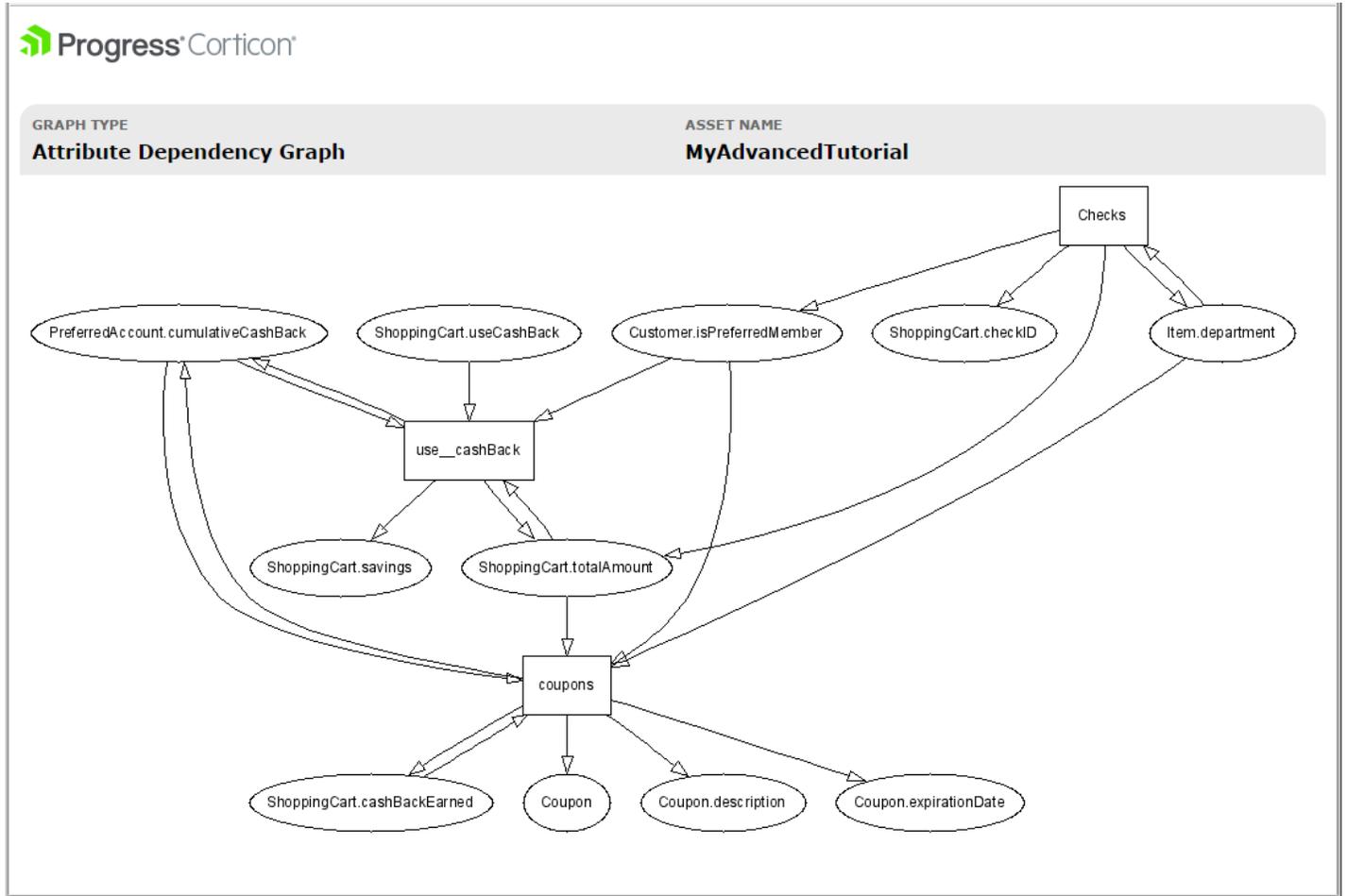
**Note:** When no objects on the Ruleflow canvas are preselected, the option to graph only selected nodes has no effect.

---

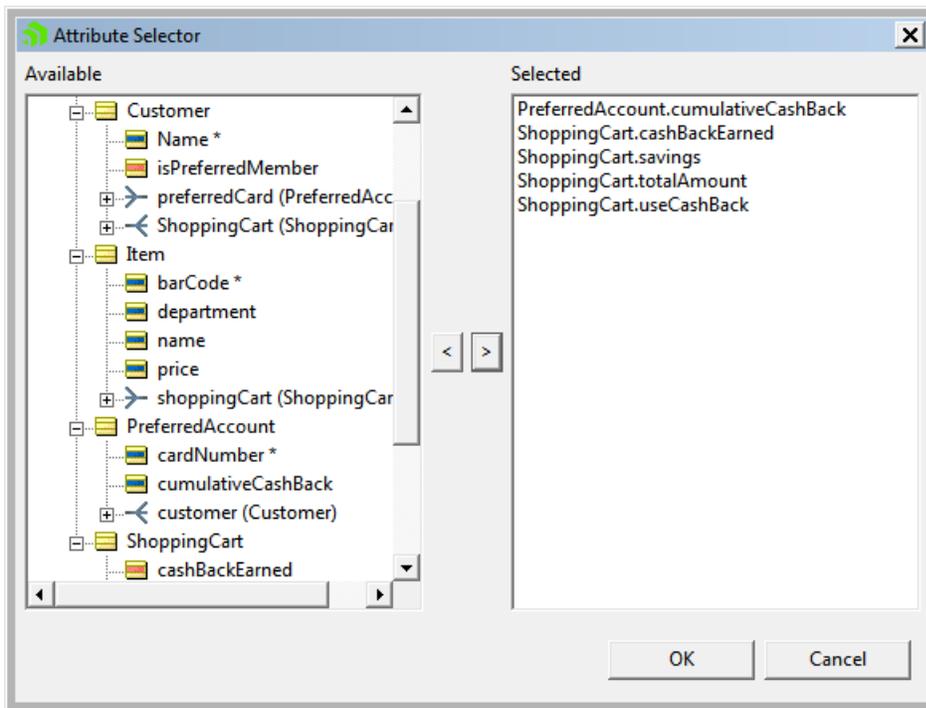
### Attribute Dependency Graph

An *attribute dependency graph* shows the attributes that establish dependencies – that is, when a Rulesheet uses an attribute set by another Rulesheet, the former has a dependency on the latter.

When you just generate a graph right away, all the attributes are included, as in this graph of the advanced tutorial's Ruleflow:

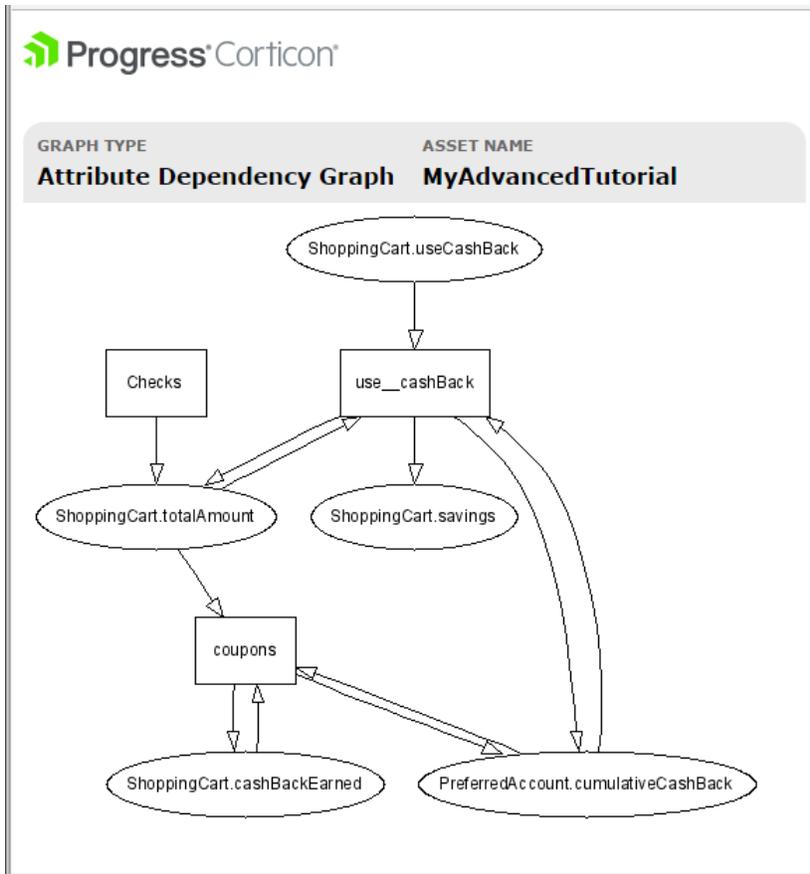


For large projects, graphs with all the attributes and dependencies can be difficult to work with. You can specify that only selected attributes are to be analyzed. Click **Attributes** to open the **Attribute Selector** dialog box, as shown:



In this illustration, five attributes were selected, so clicking **OK** returns to the graph options where clicking **Finish** generates the graph.

The graph opens in your default browser, as illustrated:

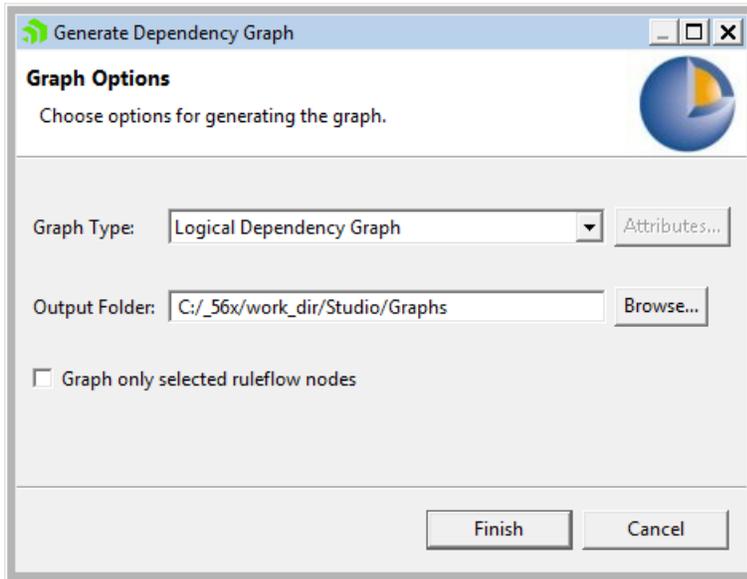


The graph image and its supporting files are saved in the output folder.

**Note:** When you next generate an attribute graph from the same Ruleflow, it overwrites the existing file unless you relocate generated files or specify unique output folders.

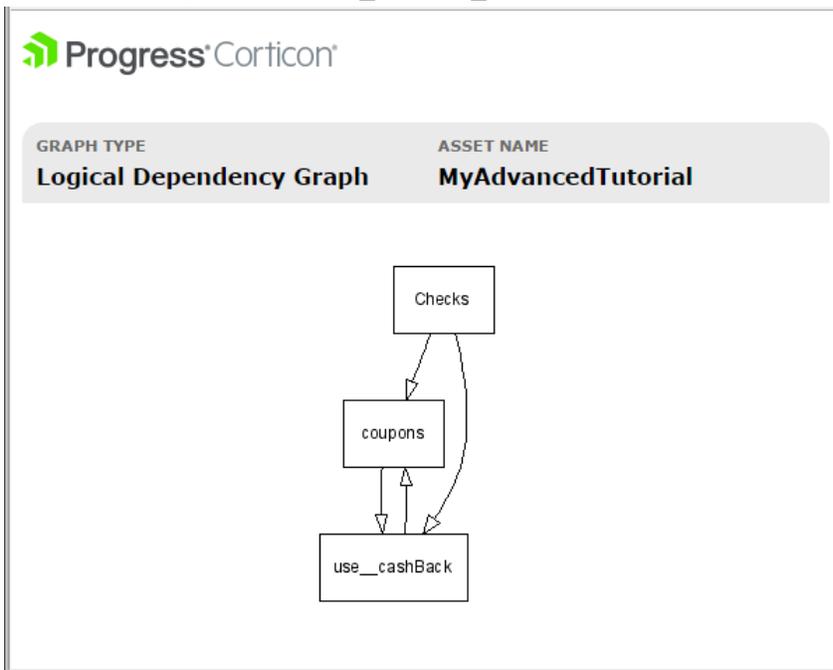
### Logical Dependency Graph

A *logical dependency graph* shows the dependency between the Rulesheets in a Ruleflow. Change the graph type to **Logical Dependency Graph**, as shown:



You can set the output folder to your preference and if Ruleflow nodes were selected before opening the dialog box, the analysis is limited to those nodes. The option to specify attributes is not relevant and not available.

Clicking **Finish** generates the graph. The following illustration is the logical dependency graph for the **Life Insurance** sample's `iSample_policy_pricing.erf`:



The graph image and its supporting files are saved in the output folder.

---

**Note:** When you again generate a dependency graph from the same Ruleflow, it overwrites the existing file unless you relocate generated files or specify unique output folders.

---

This material was added as the topic *"Generating Ruleflow dependency graphs"* in the *Rule Modeling Guide*.

# Use Natural Language expressions on filters

The use of Natural Language in Rulesheets has been extended to include filters, as shown:

The screenshot displays two windows from the Corticon Rulesheet editor. The top window, titled "Natural Language", shows a table of expressions for a filter, condition, and action. The bottom window, titled "\*Cargo.ers", shows a table of conditions and actions for a rule, with a filter highlighted in the left sidebar.

Filter Expression	English (United States)
1 Cargo.weight < Aircraft.maxCargoWeight	Reject any package that exceeds the assigned aircraft capacity
2	

Condition Expression	English (United States)
a Cargo.weight	What is the weight (in kilograms) of the package?
b Cargo.volume	What is the volume (LxWxH in cubic meters) of the package?
c	

Action Expression	English (United States)
A Cargo.container	Then use this type of container...
B	

Scope	Conditions	0	1	2	3
Aircraft	a What is the weight (in kilograms) of the package?		<= 20000	-	> 20000
Cargo	b What is the volume (LxWxH in cubic meters) of the package?		-	> 30	<= 30
	c				

Filters	Actions	Post Message(s)	0	1	2	3
1 Reject any package that exceeds the assigned aircraft capacity	A Then use this type of container...			standard	oversize	heavyweight
2	B					
3	C					
4						
5						
	Overrides				1	

This material was added to the topic *"Working with rules and filters in natural language"* in the *Rule Modeling Guide*.

## Start designing rules using Natural Language

As an aid to Rulesheet design, you can now create Natural Language phrases for the conditions, actions, and filters *before* defining those expressions.

The screenshot displays two windows from the Corticon interface. The top window, titled "Natural Language", shows a list of phrases for filters, conditions, and actions. The bottom window, titled "\*Cargo.ers", shows a Rulesheet with a table of conditions and actions.

Filter Expression		English (United States)
1	Cargo.weight < Aircraft.maxCargoWeight	Reject any package that exceeds the assigned aircraft weight capacity
2		Reject any package that exceeds the assigned aircraft volume capacity
3		

Condition Expression		English (United States)
a	Cargo.weight	What is the weight (in kilograms) of the package?
b	Cargo.volume	What is the volume (LxWxH in cubic meters) of the package?
c		

Action Expression		English (United States)
A	Cargo.container	Then use this type of container...
B		

Conditions		0	1	2	3
a	What is the weight (in kilograms) of the package?		<= 20000	-	> 20000
b	What is the volume (LxWxH in cubic meters) of the package?		-	> 30	<= 30
c					

Actions		Post Message(s)		
A	Then use this type of container...	standard	oversize	heavyweight
B				
C				
Overrides			1	

Adding a Natural Language phrase makes the next line available for additional entries. Then, in the Rulesheet, you can define the expression that satisfies the natural language phrase, as shown:

The screenshot shows two windows from a rule modeling application. The top window, titled "Natural Language", displays a table of filter, condition, and action expressions for the English (United States) locale. The bottom window, titled "\*Cargo.ers", shows a rule sheet with a scope tree on the left, a table of conditions, a table of actions, and an overrides section.

Filter Expression		English (United States)
1	Cargo.weight < Aircraft.maxCargoWeight	Reject any package that exceeds the assigned aircraft weight capacity
2	Cargo.volume < Aircraft.maxCargoVolume	Reject any package that exceeds the assigned aircraft volume capacity
3		

Condition Expression		English (United States)
a	Cargo.weight	What is the weight (in kilograms) of the package?
b	Cargo.volume	What is the volume (LxWxH in cubic meters) of the package?
c		

Action Expression		English (United States)
A	Cargo.container	Then use this type of container...
B		

Conditions	0	1	2	3
a Cargo.weight		<= 20000	-	> 20000
b Cargo.volume		-	> 30	<= 30
c				

Actions	0	1	2	3
Post Message(s)		✉	✉	✉
A Cargo.container		standard	oversize	heavyweig
B				
C				

Overrides	0	1	2	3
			1	

This material was added to the topic *"Working with rules and filters in natural language"* in the *Rule Modeling Guide*.

## Add comments to Rulesheets

While Ruletests and Ruleflows have a **Comments** tab on each file editor's **Properties** tab, Rulesheets now offer that feature as well. The comments can, for example, describe the Rulesheet's purpose, change history, or contact information.

The screenshot shows the Properties window for a Rulesheet. The "Comments" tab is selected, displaying the following text:

Rules that evaluate the applicant's age and the ratio of their net worth to income to determine adjusted long term value.  
 Revised 2017-08-05 to adjust age range in rules 1 and 2.  
 Contact: Cleveland Office's actuarial team.

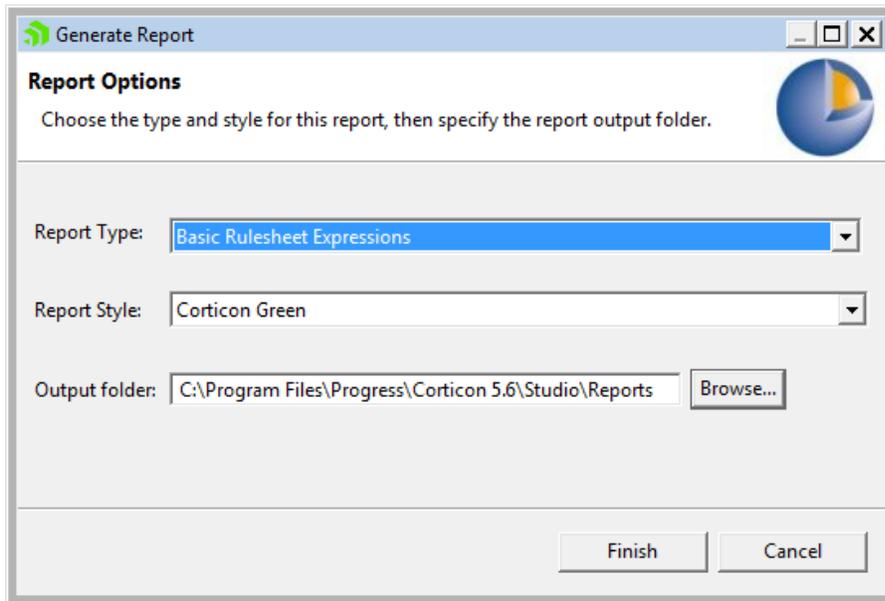
Comments show as Help when you hover the mouse over them when the Rulesheet is on a Ruleflow canvas. They also show in the header section of the Rulesheet's report.

This material was added to *"Rulesheet properties"* in the *Quick Reference Guide*.

## Create flexible Corticon Studio reports

Corticon Studio lets you create reports on each of the assets in a project: Vocabulary, Rulesheets, Ruleflows, and Ruletests. While the reporting mechanisms were available in prior releases, you were limited to the bundled reports.

Now, you can generate reports from a dialog box that lets you choose from standard types and styles of reports, and then set the output folder, as illustrated here for a Rulesheet:



The standard reports provide summaries at different levels of detail for the different asset types.

For example:

**Progress Corticon**

**REPORT TYPE**  
Basic Rulesheet Natural Language

**PROJECT NAME**  
Tutorial

**REPORT DATE**  
2016-09-14

**ASSET FILE NAME**  
.../Tutorial/Cargo.ers

**COMMENTS**  
The basic rules for determining the appropriate container for a package.

**SCOPE**

**FILTERS**  
Reject any package that exceeds the assigned aircraft weight capacity

**RULES**

#	Conditions (incl. Values)	Actions (incl. Values)	Rule Statements	Alias	Overrides
1	What is the weight (in kilograms) of the package? <= 20000	Then use this type of container... <b>standard</b>	INFO: Cargo weighing <= 20,000 kilos must be packaged in a standard container.	Cargo	
2	What is the volume (LxWxH in cubic meters) of the package? > 30	Then use this type of container... <b>oversize</b>	INFO: Cargo with volume > 30 cubic meters must be packaged in an oversize container.	Cargo	1
3	What is the weight (in kilograms) of the package? > 20000 What is the volume (LxWxH in cubic meters) of the package? <= 30	Then use this type of container... <b>heavyweight</b>	INFO: Cargo weighing > 20,000 kilos, with volume <= 30 cubic meters, must be packaged in a heavyweight container.	Cargo	

The standard report types are:

**Report Type** one of the XSLT files for the asset type:

- **Vocabulary**
  - Basic Vocabulary
  - Detailed Vocabulary
- **Rulesheets**
  - Basic Rulesheet Expressions
  - Basic Rulesheet Natural Language
  - Detailed Rulesheet Expressions
  - Detailed Rulesheet Natural Language
- **Ruleflows**
  - Basic Ruleflow Expressions
  - Basic Ruleflow Natural Language
  - Detailed Ruleflow Expressions
  - Detailed Ruleflow Natural Language
- **Ruletests**
  - Basic Ruletest

The type files are located at `[CORTICON_WORK_DIR]\Studio\Reports\XSLT\` in folders according to the asset types. You can copy the files to use as templates or change them to create report types that are then offered in the **Report Type** dropdown menu for the asset type.

**Report Style** is the CSS stylesheet to use for the report. The basic stylesheets are:

- Corticon Blue
- Corticon Green

The style files are located at `[CORTICON_WORK_DIR]\Studio\Reports\CSS\`. You can copy a stylesheet file to use as a template to create custom report styles that are then offered in the **Report Style** dropdown menu.

**Output Folder** is the location where the report will be stored on disk. The default location is `[CORTICON_WORK_DIR]/Studio/Reports`. You can create a root location such as `C:\CorticonStudioReports` and then append subfolder names to sort out your projects, tasks, clients, or versions.

This material was added to the topic *"The Corticon Studio reporting framework" in the Rule Modeling Guide*.

See also:

- *"Creating a Vocabulary report" section of the Quick Reference Guide*
- *"Creating a Rulesheet report" section of the Quick Reference Guide*
- *"Creating a Ruleflow report" section of the Quick Reference Guide*
- *"Creating a Ruletest report" section of the Quick Reference Guide*

## REST API to return a Decision Service's Vocabulary metadata

A REST API for retrieving vocabulary metadata from a deployed Decision Service is now available. This is useful for integrating Corticon with other applications that have to format REST or SOAP calls to a Decision Service.

### Structure of a request

To retrieve vocabulary metadata, make an HTTP GET request to the `getVocabularyMetadata` endpoint specifying the Decision Service name and version as the following URL parameters:

- `name=Decision Service name`
- `majorVersion= Major version of the Decision Service`
- `minorVersion= Minor version of the Decision Service`

For example:

```
http://localhost:8850/axis/corticon/decisionService/getVocabularyMetadata
?name=ProcessOrder&majorVersion=1&minorVersion=1
```

The following JSON-formatted document is an example of a response:

```
{
  "majorVersion": 1,
  "name": "MetadataTest",
  "minorVersion": 0,
  "entities": [
```

```

{
  "name": "Entity_1",
  "associations": [
    {
      "name": "associationObject1",
      "targetEntity": "AssociationObject1",
      "inContext": true,

      "mandatory": false
      "cardinality": "1"
    },
    {
      "name": "associationObjectOverride",
      "targetEntity": "AssociationObject2",
      "inContext": false,
      "mandatory": true
      "cardinality": "*"
    }
  ],
  "attributes": [
    {
      "dataType": "Boolean",
      "name": "boolean1",
      "inContext": true,
      "type": "Base",
      "mandatory": true
    },
    {
      "dataType": "Date",
      "name": "date1",
      "inContext": false,
      "type": "Transient",
      "mandatory": false
    },
    {
      "dataType": "DateTime",
      "name": "datetime1",
      "inContext": true,
      "type": "Base",
      "mandatory": true
    },
    {
      "cdtConstraintExpr": "value < 100.0", <== Constraint expression associated
with this Attribute.
      "dataType": "Decimal",
      "name": "decimal1",
      "inContext": false,
      "type": "Transient",
      "mandatory": false
    },
    {
      "dataType": "Integer",
      "name": "int1",
      "cdtEnumeration": [ <== A CDT that is values only (no labels).
        {"value": "1"},
        {"value": "2"},
        {"value": "3"},
        {"value": "4"}
      ],
      "inContext": true
      "type": "Base",
      "mandatory": true
    },
    {
      "dataType": "String",
      "name": "string1",
      "cdtEnumeration": [ <== A CDT that has labels and values.
        {
          "value": "s"
        }
      ]
    }
  ]
}

```

```

        "label": "Small"
      },
      {
        "value": "m"
        "label": "Medium"
      },
      {
        "value": "l"
        "label": "Large"
      }
    ],
    "inContext": false,
    "type": "Transient",
    "mandatory": false
  },
  {
    "dataType": "Time",
    "name": "time1",
    "inContext": true,
    "type": "Base",
    "mandatory": true
  },
]
},
{
  "name": "AssociationObject1",
  "associations": [],
  "attributes": [
    {
      "dataType": "String",
      "name": "string1",
      "inContext": true
      "mandatory": true
    },
  ],
}
},
{
  "name": "AssociationObject2",
  "associations": [],
  "attributes": [
    {
      "dataType": "String",
      "name": "string1",
      "inContext": false
      "mandatory": false
    },
  ],
}
]
}
}

```

The metadata API is available for Corticon Decision Services deployed to either Java or .NET servers.

For details about the response content and a complete example, see the new topic *"Accessing the Vocabulary metadata of a Decision Service" in the Rule Modeling Guide*.

## REST API to pass a Decision Service as a URL parameter

The REST API has been enhanced to allow passing a Decision Service's name and version as parameters in GET requests, and in the payload for POST requests. This is the recommended way to pass this information.

For example, a request that would get the properties for a Decision Service `ProcessOrder` version 1.16, with the `GET` parameters shown in bold:

```
http://<serverHost>:8850/axis/corticon/decisionService/getProperties
      ?name=ProcessOrder&majorVersion=1&minorVersion=16
```

Evaluation of a request looks first for the URL parameter, and if that is not found, it looks for it as an HTTP header.

`POST` requests can include fields inside the JSON object to specify the Decision Service instead. The following example for the same Decision Service version sets these properties in the fields shown in bold:

```
http://<serverHost>:8850/axis/corticon/decisionService/setProperties
```

```
{
  "name" : "ProcessOrder",
  "majorVersion" : "1",
  "minorVersion" : "16",
  "msgStyle": "FLAT",
  "autoReload", "true"
}
```

---

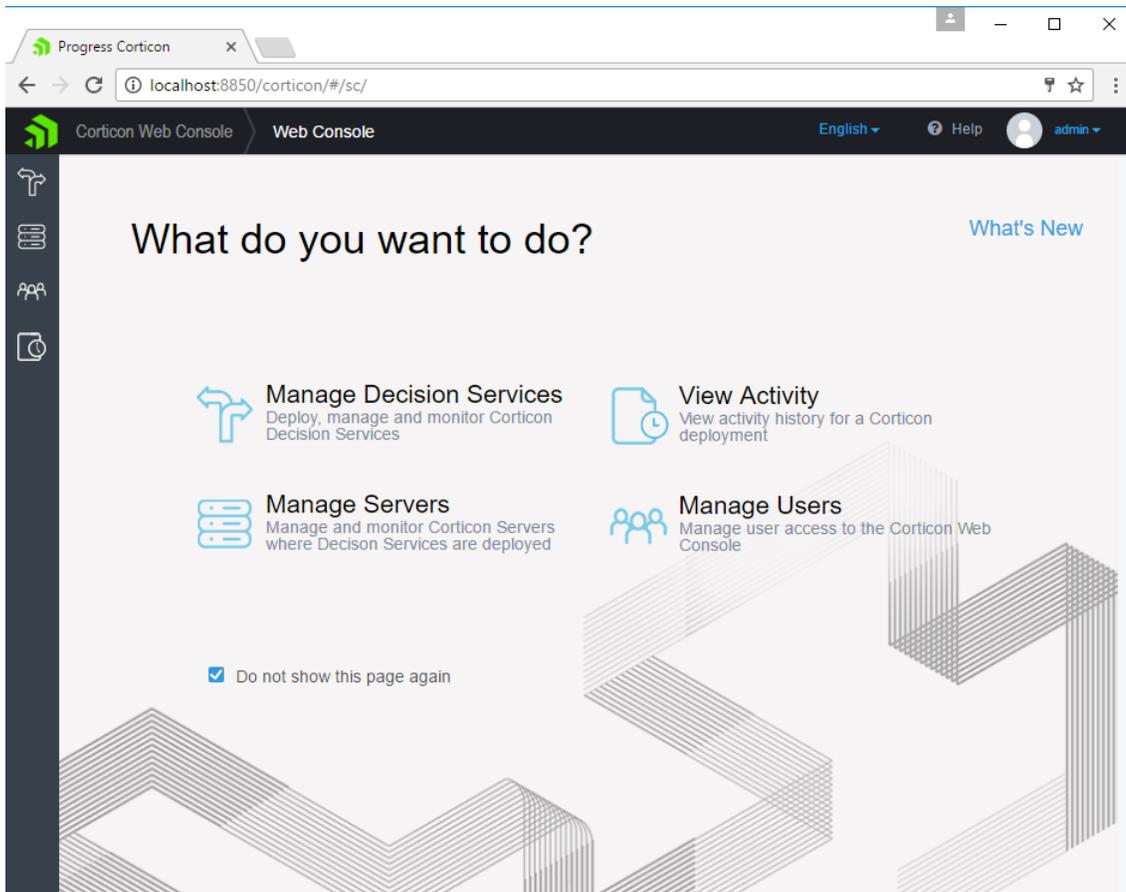
**Note:** The previous method of passing the information in HTTP headers is still supported.

---

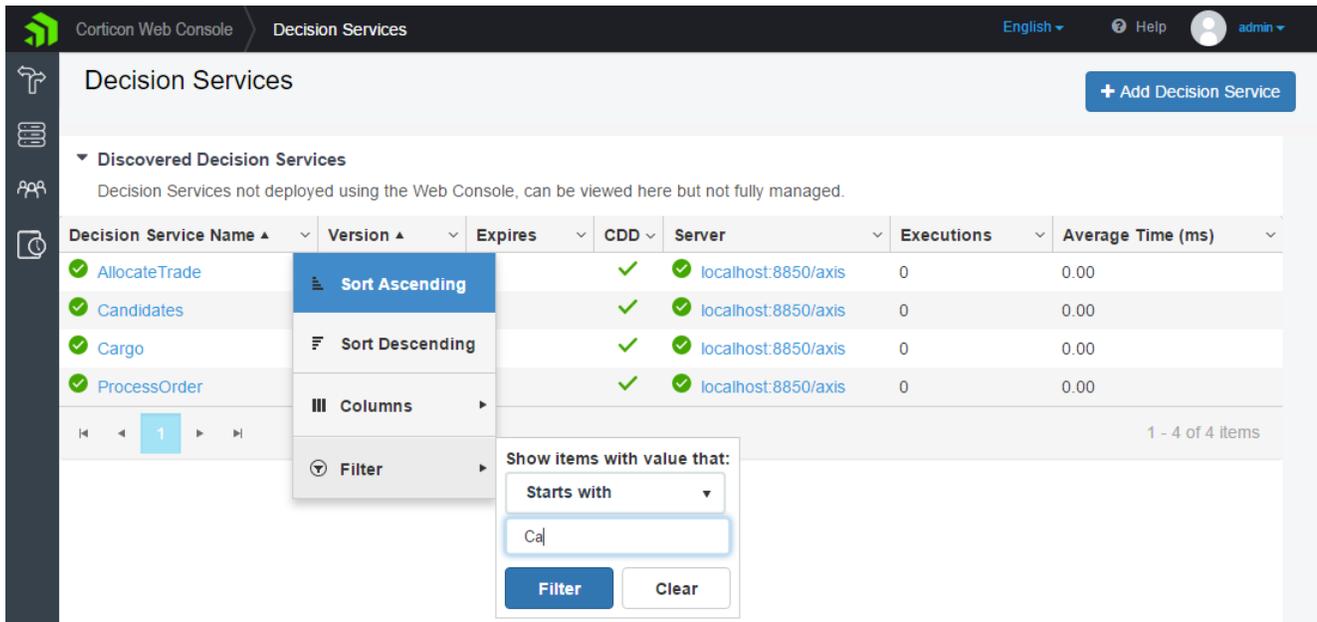
The REST calls in the following topics were updated with this option: *"REST call"*, and *"Sample JSON request and response messages"* in the *Integration and Deployment Guide*.

## Improved Web Console

Using the Corticon Web Console is easier than ever, starting right at its top-level page where you can quickly access its functions.



The improved tables and grids not only have dynamic resizing, they also let you sort the rows, choose columns that you want to see, and filter the rows listed.



Dialog boxes are now compact, those with a lot of data have tabs, as shown:

### Add Decision Service ✕

Decision Service [Database](#) [Advanced](#) [Monitored Attributes](#)

When adding a Decision Service you must specify a name, select a server and provide the EDS file of the Decision Service. Other properties are optional. To add the Decision Service to an existing Application select "**Add to an Existing Application**"

**Name**

**Description**

**EDS File** [Choose File...](#)

**Servers**

**Add to an Existing Application**

The Decision Services page is comprehensive. It now distinguishes the managed Decision Services that are in applications and those that are independent. Then, it lists the discovered Decision Services on standalone servers and servers in server groups.

Corticon Web Console Decision Services English Help admin

### Decision Services + Add Decision Service

**Managed Decision Services**  
Decision Services deployed using the Web Console, can be fully managed here.

Application Name	Servers	Executions	Average Time (ms)
Insurance	local server	0	0.00

Decision Service Name	Version	Executions	Average Time (ms)
Generate Policy	0.41	0	0.00
Price Policy	0.41	0	0.00

1 - 1 of 1 items

Decision Service Name	Version	Servers	Executions	Average Time (ms)
Cargo	1.1	QA Group	0	0.00

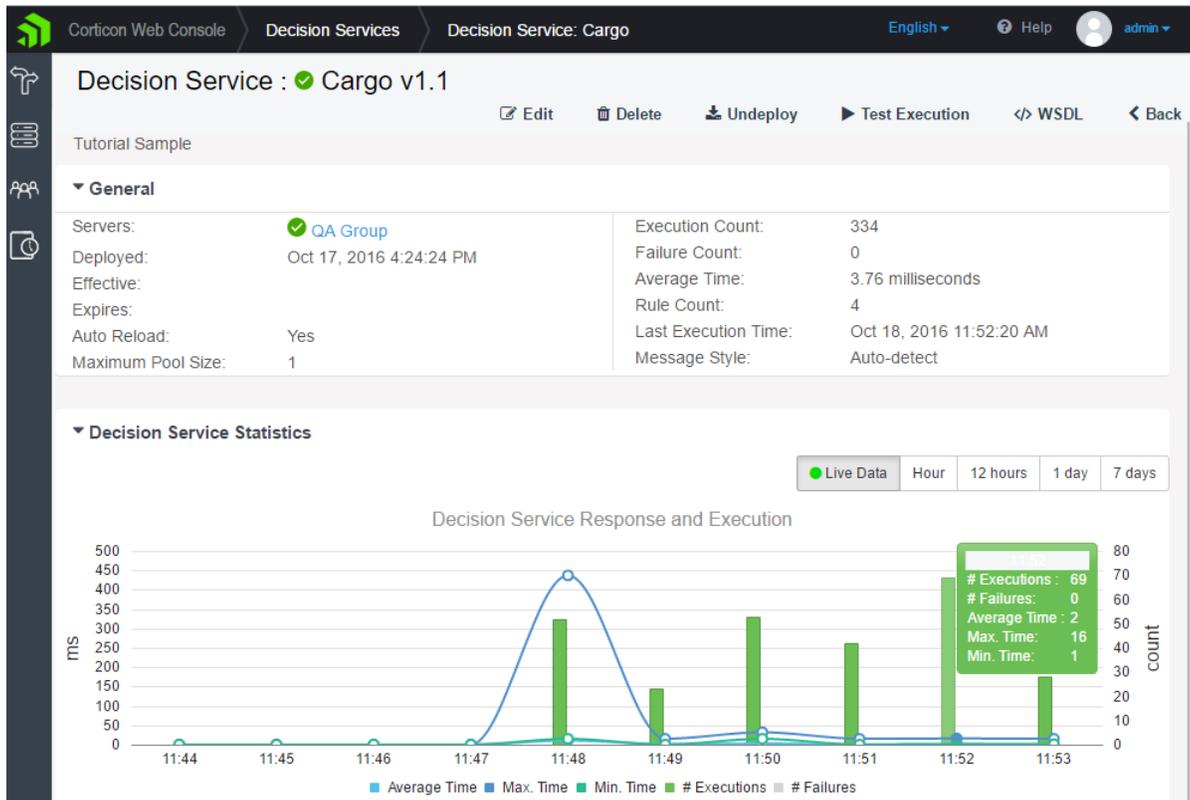
1 - 1 of 1 items

**Discovered Decision Services**  
Decision Services not deployed using the Web Console, can be viewed here but not fully managed.

Decision Service Name	Version	Effective	Expires	CDD	Server	Executions	Average Time (ms)
AllocateTrade	1.14	✓		✓	localhost:8850/axis	0	0.00
AllocateTrade	1.14	✓		✓	NBBEDGSAINTMA5:8850/axis	0	0.00
AllocateTrade	1.14	✓		✓	NBBEDGSAINTMA1:8850/axis	0	0.00
Candidates	1.14	✓		✓	NBBEDGSAINTMA5:8850/axis	0	0.00
Candidates	1.14	✓		✓	localhost:8850/axis	0	0.00
Candidates	1.14	✓		✓	NBBEDGSAINTMA1:8850/axis	0	0.00
Cargo	0.16	✓		✓	NBBEDGSAINTMA1:8850/axis	0	0.00
Cargo	0.16	✓		✓	NBBEDGSAINTMA5:8850/axis	0	0.00
Cargo	0.16	✓		✓	localhost:8850/axis	0	0.00
ProcessOrder	1.10	✓		✓	NBBEDGSAINTMA5:8850/axis	0	0.00

1 - 10 of 12 items

Information is presented in clear, easy to read pages.



These features are described in detail in the Corticon 5.6 [Web Console Guide](#)

## Visualize monitored attributes in a Decision Service

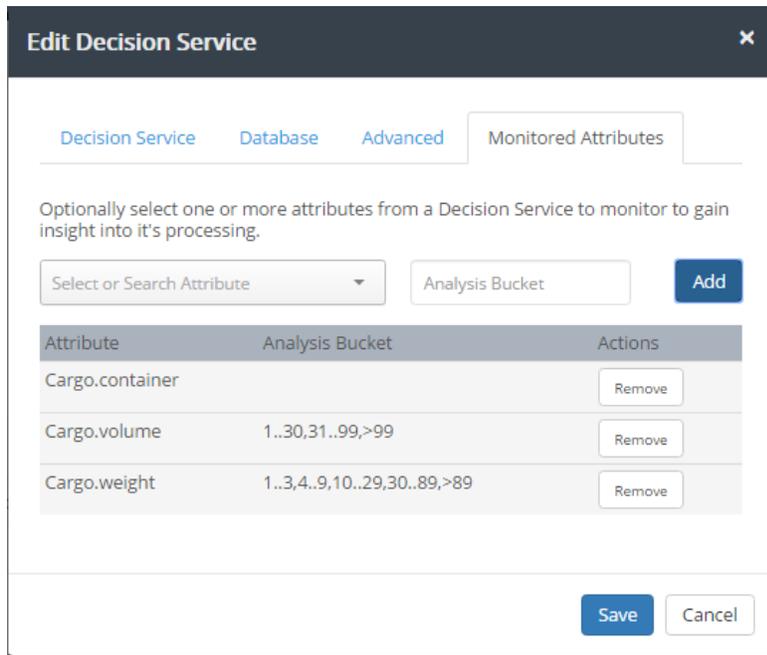
The Web Console lets you monitor the value distribution of one or more attributes in a deployed Decision Service. By choosing attributes to monitor, you can view the statistical breakdown of attribute values over the course of many Decision Service executions.

For example, the Ruleflow created in the [Tutorial: Basic Rule Modeling in Corticon Studio](#) reads integer values for `Cargo.volume` and `Cargo.weight` in the request, and assigns a text value to the attribute `Cargo.container`. To monitor these attributes, select the name in the **Monitored Attribute** dialog, enter comma-delimited values or value ranges in the **Analysis Buckets** entry area, and then click **Add**.

When you set *bucket* ranges of values, you can analyze categories of data. Bucketing is a useful when a wide range of numeric or date data is possible. For this example, the three buckets for `Cargo.volume` are 1 to 30 kilos, 31 to 99 kilos, and greater than 99.

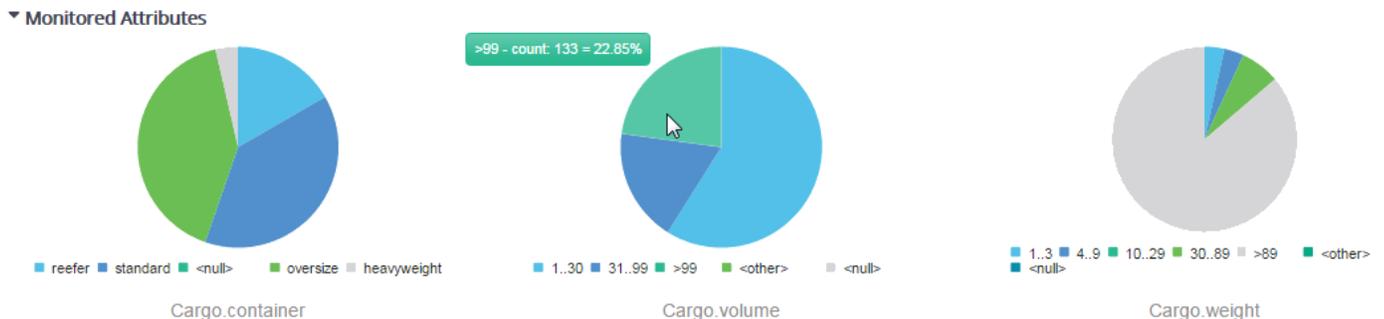
Entering no values can be useful for string values, especially when there is a small set of values defined in a Custom Data Type (such as `Cargo.container` in this example), or there is small set of known values, such as risk ratings `high`, `medium`, `low`.

The monitored attributes in this example are listed as shown:



Click **Save** to enable your selections.

In this example, the integer values are examined across narrower ranges than the rules, perhaps as a study to see whether new container categories should be considered. The results of attribute monitoring are visualized as follows:



This feature is presented in the Corticon 5.6 [Web Console Guide](#)

## Settings to use Rule Execution Recording on Decision Services

The Rule Execution Recording feature introduced in the Corticon 5.5.2 release provided a way to record each execution's request and response as well as all rule messages into a specially designated database. (See the topic *"Implementing Rule Execution Recording in a database"* in the *Integration and Deployment Guide*.

The feature described its implementation and how to configure Corticon Deployment Descriptor (CDD) files to use the service in deployment. The mechanisms for using the recording service through other deployment packagings and toolsets are now available.

After you configure and enable a server for Rule Execution Recording, you can dynamically enable or disable use of the Execution Recording Service for individual Decision Services not deployed through CDDs in:

- **Corticon Web Console** - See the *"Applications and Decision Services"* section of the *Web Console Guide*. (Decision Services deployed through CDDs must still set this property in their .cdd file)
- **SOAP API** - You can set `PROPERTY_EXECUTION_RECORDING_SERVICE_ENABLED` to `true` or `false`
- **Server test scripts** - The scripts accept the property name `PROPERTY_EXECUTION_RECORDING_SERVICE_ENABLED` followed by the value `true` or `false` in transactions 248, 249, and 250)
  - In-process testing: `testServer.bat` (Java) and `Corticon-Api-Inprocess-Test.exe` (.NET)
  - SOAP API testing: `testServerAxis.bat` (Java) and `Corticon-Api-Remote-Test.exe` (.NET)

To create Decision Services in test scripts, use the following transactions:

```
101 - Add a Decision Service (3 parameters)
102 - Add a Decision Service (6 parameters)
103 - Add a Decision Service (9 parameters)
```

Then, to enable rule execution recording on the Decision Service, use...

```
248 - Set Decision Service's Property Value
249 - Set Decision Service's Property Value (by specific Decision Service Major Version)
250 - Set Decision Service's Property Value (by specific Decision Service Major and
Minor Version)
```

... to set the property as shown:

```
Enter transaction number: 248
Input Service Name: Order
Input Decision Service Property Name: PROPERTY_EXECUTION_RECORDING_SERVICE_ENABLED
Input Decision Service Property Value: true
```

---

**Note:** Publish deployment mechanisms from Studio cannot set this property on a server.

---

This material has been added to related topics in the documentation.

## Deployment security: Authentication and encryption

When planning how you will deploy and manage Corticon Decision Services, you need to consider how to secure the deployments. When deploying Corticon, you can use the basic authentication and encrypted communication on your host application server to secure your deployment.

- **Authentication** requires anyone accessing a server to authenticate by supplying their username and password credentials. Implementing authentication lets you control which users can access a server and what actions they can perform. For example, a user might be able call a Decision Service but not to deploy one.
- **Encryption** enables confidential communication between a server and a client. Enabling HTTPS requires that you add your signed CA certificate to each server, and a CA client certificate on each of the clients that will access it.

---

**Note:** For detailed information on configuration and use of Corticon security features, see the topics under *"Secure servers with authentication and encryption"* in the *Integration and Deployment Guide*.

---

## Secure deployment of Decision Services

The center of a Corticon deployment are the **Corticon Servers** where you deploy Decision Services. Configuring basic authentication on the Corticon Servers controls which users and external processes can call Decision Services or perform administrative operations. You can extend authentication to use LDAP directory services.

When administrators use the **Corticon Web Console** to manage and monitor any Corticon Servers that are configured for basic authentication, each server definition must include the username and password that the Web Console can present to authenticate on the managed server. When accessing the Web Console from a browser, you must supply username/password credentials.

When developers deploy Decision Services from **Corticon Studio**, they can either deploy directly to a Corticon Server or to a Web Console which will then deploy the Decision Service to one or more Corticon Servers on your behalf. If using the Web Console to manage your deployment, you should deploy through it. In either case, you need to provide credentials and have administrative rights to perform a deployment.

When system integrators use **Corticon REST and SOAP APIs** to access a secured server, the connection information must have valid credentials before any calls to a Decision Service are allowed.

---

**Note:** The Corticon Server installer bundles an instance of the Progress® Application Server (PAS) for the Corticon Java Server and for Corticon Web Console. PAS is an instance of Tomcat 8 that has been hardened to remove known vulnerabilities. You can choose to deploy with this application server or a preferred, supported application server. While the documentation assumes PAS deployment, the principles presented can be applied to other application servers.

---

### Configuring Corticon Server authentication

Configuring basic authentication for a Corticon Server is done through the `web.xml` file in Corticon Server's `axis.war` file. This file is the deployment descriptor for the Corticon Server when deployed to an application server. Within it, you configure authentication for accessing the Corticon Server.

By default, basic authentication is not enabled and Corticon is using the user definitions in the `tomcat-users.xml` file, a plain text file located in the application server's `conf` folder. For production deployments, it is recommended that you enable basic authentication and define security constraints for accessing the Corticon Server. The specifics of this can vary based on your requirements and your authentication service.

The bundled `web.xml` file contains a commented-out section showing common settings. Typically you want to define security constraints to limit the ability to call a Decision Service or perform administrative actions to authenticated users with specific defined roles.

---

**Note:** You are able to configure the role required to call a Decision Service but you cannot refine this to specific Decision Services. If a user can call any Decision Service, they can call them all.

---

Uncommenting the sample basic authentication configuration in `web.xml` and restarting the Corticon server enables basic authentication for ALL calls to the Corticon Server. You can restrict defined user roles to specified URLs, as described in *"Securing Server endpoints" in the Integration and Deployment Guide*.

### Configuring Web Console authentication

The Web Console requires that you provide a valid username/password to access it. By default, the Web Console stores authorized usernames and passwords in its bundled database.

**Note: About LDAP** - In a production deployment you typically want to configure the Corticon Web Console to use an LDAP service such as Microsoft's Active Directory. See the topic *"Using LDAP for Web Console authentication"* in the *Integration and Deployment Guide* for details on configuring LDAP.

### Configuring authentication in server test scripts

Corticon includes the `testServerRest` and `testServerAxis` command line utilities for testing aspects of the REST and SOAP APIs. If a Corticon Server has basic authentication enabled you will need to provide username and password credentials by entering transaction number 0 (zero). This allows you specify credentials for establishing a session with a Corticon server. For example:

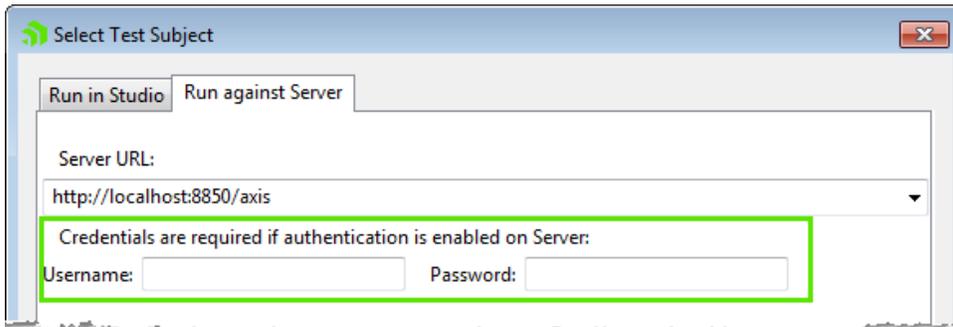
```
Enter transaction number: 0
Input new URL to J2EE Web Server: http://localhost:8850
Input new Web Application name: axis
(optional) Username if using a secured server: admin
(optional) Password if using a secured server: admin
Transaction completed.
```

For more about how and when to use option 0 (zero) in test scripts, see *"Using authentication in server test scripts"* in the *Integration and Deployment Guide*.

### Accessing a Server from Studio

When using Corticon Studio to access a Corticon Server that has basic authentication enabled, you need to provide username and password credentials for that server. This occurs when publishing or downloading a Decision Service or running rule tests against a deployed Decision Service. For example:

- **Ruletests on Servers** - When a Corticon Studio Ruletest wants to use a deployed server to test its rules, authentication will require credentials permit both Admin and Execution operations to enable connection to the server. To support this functionality the **Select Test Subject** dialog's **Run against Server** tab, as shown:



The credentials are saved within each Testsheet for automatic re-use later.

## Encrypt communication between Corticon components

Configuring HTTPS enables encrypted traffic between the components of your Corticon deployment:

- On a **Corticon Server**, configuring HTTPS encrypts all calls to execute a Corticon Decision Service and encrypts the response that is returned. This is particularly important if you have an unsecure network and are passing sensitive data to or from a Decision Service. Configuring HTTPS on the Corticon Server also encrypts all administrative traffic with the Corticon Server.

- On a **Web Console**, configuring HTTPS will encrypt all communication between the web browser used by a Corticon administrator and the Web Console. This is important if you have an unsecure network and want to prevent snooping on administrative traffic.
- **Clients using HTTPS** to access a Corticon Server must have client certificate installed at their end. This is true for clients calling the Corticon Server to execute a decision service, administrators using their browser to access the Web Console, and the Web Console Server for accessing managed Corticon Servers.

### Setting up encrypted communications

Configuring HTTPS requires a signed CA certificate to be installed on each Corticon Server and Web Console you want to enable for encryption. When the Corticon Server and Web Console are hosted on the same application server, a single certificate can be shared. Any client that wants to use HTTPS to enable secure communications with a server requires a client certificate so that they can handshake and negotiate the encryption algorithm that will be applied.

To enable HTTPS on Corticon Server for Java, obtain a private key and a signed Web server digital certificate, and then install the Web server digital certificate in the Java keystore using the Java Keytool utility.

When certificates have been added and the server restarted, HTTPS is enabled on its default port, 8851.

### Using HTTPS in Corticon Studio

You must obtain and install public key certificates for the Corticon Studio. The public certificate then needs to be imported to the Java keystore for the Corticon Studio.

You can use HTTPS from Studio to a Server for:

- **Packaging operations** - Whether deploying Decision Services directly to a Corticon Server or to a Corticon Server managed by the WebConsole, you can choose the `https://` protocol and provide the Server's HTTPS port, 8851.
- **Running Ruletests on Servers** - When you choose a remote server that has enabled HTTPS, you can choose the `https://` protocol and provide the Server's HTTPS port, 8851.

## Automatic building and validation of projects

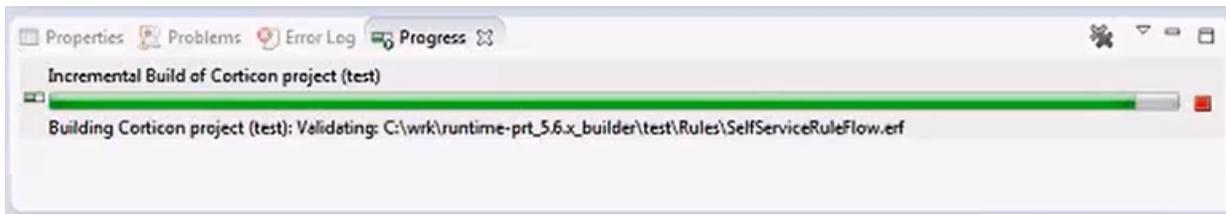
Corticon Studio now uses the Eclipse builder mechanism to validate Corticon assets. An Eclipse builder thread runs in the background to validate changed assets and their dependencies.

Corticon Studio performs the validation in the background when any asset is saved to see if other assets are still valid. This validation does not block any interaction with Corticon Studio.

### About validation...

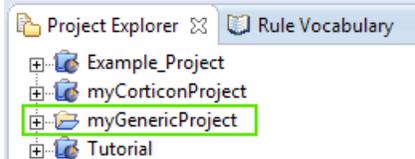
The project validation process has an impact on available memory, and uses background CPU cycles. Here are some tips:

- If your workspace has several large projects, try to keep inactive ones closed to optimize resources. This is particularly important when you restart Studio, as it validates all open projects on startup. Closing projects in your workspace that you are not working on limits the amount of validation done on startup.
- When performing tasks such as re-arranging assets in a large project, you might want to shut off the option to **Build Automatically**. Once you have completed your changes, choose **Clean**, and then turn **Build Automatically** on again.
- When building large projects, follow the build progress in the **Progress** view (exposed from the menu item **Window > Show View > Other** then **General: Progress**).

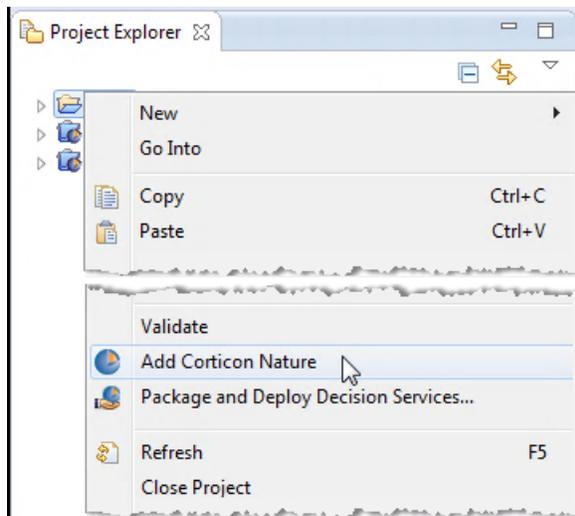


## Corticon Nature

Corticon builds and extensions rely on your project having a *Corticon Nature*. The following illustration contrasts a Corticon project and a generic project:



To convert a generic project to give it a Corticon Nature, right-click on the project's folder, and then select **Add Corticon Nature**, as illustrated:



This information was added as the topic *"Ensure that all projects build as Corticon projects"* in the *Installation Guide*.

## Simplified installation

In prior releases, Corticon service packs and hotfixes were distinct downloads. You had to install the major.minor version, and then overlay the latest service pack or hotfix.

Starting with this release, both the Corticon Studio installer and the Corticon Server installer can perform any installation or upgrade that is required. If a Corticon 5.6.1 service pack or 5.6.0.3 hotfix becomes available, it is included when performing a new install, so you do not need to first install 5.6.0.

User modified files are backed up when the installer is updating an earlier Corticon 5.6 install, and differences are reconciled when the install completes.

The *Corticon Installation Guide* has been revised to describe the smart service pack and hotfix installers.

## Miscellany

- Two new server properties that adjust performance in batch processing are now exposed :

```
com.corticon.services.registerNewSCOEntities  
com.corticon.reactor.engine.registerNewEntities
```

For details, see *"Server properties" in the Integration and Deployment Guide*.

- The Corticon Studio View **Rule Project Explorer** has been dropped. Its functionality was identical to the **Project Explorer**.
- The legacy Java Server Console and its associated documentation have been dropped from the server installer. The Corticon Web Console should be used for administering Corticon Servers through a Web UI. The legacy Java Server Console and its documentation are available as a separate download. It is packaged as `PROGRESS_CORTICON_5.6_SERVER_CONSOLE.zip` within the `PROGRESS_CORTICON_5.6_SERVER.zip` download file available on the Progress download site.
- Progress Corticon Studio downloads and installations are now only 64-bit. The 32-bit installer is no longer available.
- Management of transient memory used by Studio editor has been optimized.
- Documentation of Custom Data Types has been enhanced to clarify where labels and values are accessed and exposed.
- Documentation for *"Creating custom context URLs on a web server" in the Installation Guide* has been enhanced to describe how to specify the added context's sandbox, cdd, and logs directories.
- Documentation for the remove operator was changed to describe its behavior when applied to a collection. When you use **.remove** to delete elements of a collection, lower-level associated entities are removed. See *"Remove element" in the Rule Language Guide*.
- Documentation of qualifications for database query filters was enhanced to note that a filter query that has relational operators with Boolean operands does not qualify.
- Methods previously deprecated in the Foundation API have now been dropped.

---

# A

---

## Progress Corticon documentation - Where and What

---

Corticon provides the following tutorials and documentation, available in installed components or online, as indicated.

Corticon Tutorials available at the <a href="#">Corticon Learning Center</a>	
<a href="#">Tutorial: Basic Rule Modeling in Corticon Studio</a>	An introduction to the Corticon Business Rules Modeling Studio. Learn how to capture rules from business specifications, model the rules, analyze them for logical errors, and test the execution of your rules -- all without any programming. <i>Online only. Uses samples packaged in the Corticon Studio.</i>
<a href="#">Tutorial: Advanced Rule Modeling in Corticon Studio</a>	Learn about the concepts underlying some of Studio's more complex and powerful functions such as ruleflows, scope and defining aliases in rules, understanding collections, using String/DateTime/Collection operators, modeling formulas and equations in rules, and using filters. <i>Online only.</i>
<a href="#">Modeling Progress Corticon Rules to Access a Database using EDC</a>	Shows rule modelers how to model and test rules that read/write to a relational database. <i>Online only. Uses samples packaged in the Corticon Studio, and Microsoft SQL Server 2014 as its EDC database.</i>

<a href="#">Connecting a Progress Corticon Decision Service to a Database using EDC</a>	Shows integration developers how to set up the Vocabulary to connect and map to a database, configure EDC settings in the Web Console, and other deployment-related tasks. <i>Online only. Uses samples packaged in the Corticon Studio, and Microsoft SQL Server 2014 as its EDC database.</i>
<a href="#">Deploying a Progress Corticon Decision Service as a Web Service for Java</a>	This tutorial is for server administrators who are responsible for deploying Corticon Decision Services. Learn how to prepare and deploy a Decision Service on Progress Corticon Server as a REST or SOAP Web Service for Java. This tutorial is also available for download as PDF.
<a href="#">Deploying a Progress Corticon Decision Service in process for Java</a>	This tutorial is for Java developers who want to access a Corticon Decision Service in process from a Java client application. This tutorial is also available for download as PDF.
<a href="#">Using Corticon Business Rules in a Progress OpenEdge Application</a>	This tutorial is for experienced OpenEdge ABL developers. Learn what you need to do on the OpenEdge side to access and use a Corticon Decision Service from an OpenEdge ABL application.
<b>Corticon Online Documentation</b>	
<a href="#">Progress Corticon User Assistance</a>	Updated online help for the current release.
<a href="#">Introducing the Progress® Application Server</a>	The Progress Application Server (PAS) is the Web application server based on Apache Tomcat installed as the default Corticon Server. TCMAN, the command-line utility, manages and administers the Progress Application Server.
<a href="#">Progress Corticon Documentation site</a>	Access to all guides in the Corticon documentation set in PDF format and JavaDocs.
<b>Corticon Documentation on the <a href="#">Progress download site</a></b>	
Documentation	Package of all guides in PDF format.
What's New Guide	PDF format.
Installation Guide	PDF format.
Corticon Studio Installers	Include Eclipse help for all guides except Web Console.

**Components of the Corticon documentation set**

The components of the Progress Corticon documentation set are the following guides:

<b>Release and Installation Information</b>
---

<i>What's New in Corticon</i>	Describes the enhancements and changes to the product since its last point release.
<i>Corticon Installation Guide</i>	Step-by-step procedures for installing all Corticon products in this release.
<b>Corticon Studio Documentation: Defining and Modeling Business Rules</b>	
<i>Corticon Studio: Rule Modeling Guide</i>	Presents the concepts and purposes the Corticon Vocabulary, then shows how to work with it in Rulesheets by using scope, filters, conditions, collections, and calculations. Discusses chaining, looping, dependencies, filters and preconditions in rules. Presents the Enterprise Data Connector from a rules viewpoint, and then shows how database queries work. Provides information on versioning, natural language, reporting, and localizing. Provides troubleshooting of Rulesheets and Ruleflows. Includes <i>Test Yourself</i> exercises and answers.
<i>Corticon Studio: Quick Reference Guide</i>	Reference guide to the Corticon Studio user interface and its mechanics, including descriptions of all menu options, buttons, and actions.
<i>Corticon Studio: Rule Language Guide</i>	Reference information for all operators available in the Corticon Studio Vocabulary. Rulesheet and Ruletest examples are provided for many of the operators.
<i>Corticon Studio: Extensions Guide</i>	Detailed technical information about the Corticon extension framework for extended operators and service callouts. Describes several types of operator extensions, and how to create custom extension plug-ins.
<b>Corticon Server Documentation: Deploying Rules as Decision Services</b>	
<i>Corticon Server: Integration and Deployment Guide</i>	An in-depth, technical description of Corticon Server deployment methods, including preparation and deployment of Decision Services and Service Contracts through the Deployment Console tool. Describes JSON request syntax and REST calls. Discusses relational database concepts and implementation of the Enterprise Data Connector. Goes deep into the server to discuss state, persistence, and invocations by version or effective date. Includes troubleshooting servers through logs, server monitoring techniques, performance diagnostics, and recommendations for performance tuning.
<i>Corticon Server: Deploying Web Services with Java</i>	Details setting up an installed Corticon Server as a Web Services Server, and then deploying and exposing Decision Services as Web Services on the <a href="#">Progress Application Server (PAS)</a> and other Java-based servers. Includes samples of XML and JSON requests.

<i>Corticon Server: Deploying Web Services with .NET</i>	Details setting up an installed Corticon Server as a Web Services Server, and then deploying and exposing decisions as Web Services with .NET. Includes samples of XML and JSON requests.
<a href="#">Corticon Server: Web Console Guide</a>	Presents the features and functions of browser connection to a Web Console installation to manage Java and .NET servers in groups, manage Decision Services as applications, and monitor performance metrics of managed servers.