PROGRESS® Corticon®

What's New in Corticon

PROGRESS

1

What's new and changed in Corticon 5.3.4

This chapter summarizes the new, enhanced, and changed features in Progress® Corticon® 5.3.4. Service Pack 4 includes the changes that were released in Service Packs 1, 2 and 3. When you update from 5.3.0 or an earlier Service pack, all sets of Service Pack changes will be applied.

This Corticon release coordinates with other Progress Software releases:

- Progress OpenEdge is available as a database connection. You can read from and write to an OpenEdge database from Corticon Decision Services. This feature is distinct from the integration of Business Rules from Corticon with an OpenEdge application, introduced in Corticon 5.3.2. OpenEdge is a separately licensed Progress Software product.
- Progress DataDirect Cloud (DDC) enables simple, fast connections to cloud data regardless of source. DataDirect Cloud is a separately licensed Progress Software product.

For details, see the following topics:

- Enhancements to rule development and OpenEdge integration in 5.3.4
- Enhancements to rule testing and tracing in 5.3.4
- Enhancements to logging in 5.3.4
- Enhancements to compiling Decision Services in 5.3.4
- Enhancements to handling of requests in disparate locales, languages, and timezones
- Changes to display of Time and DateTime across timezones in 5.3.4

Enhancements to rule development and OpenEdge integration in 5.3.4

Several improvements were made to Corticon Studio and its integration to OpenEdge. Significant items include:

Option to lock and unlock a Vocabulary

You can now choose to lock or unlock a Vocabulary in Studio by selecting the menu option **Vocabulary > Set to Read Only**.

Once locked, all the icons in the Vocabulary display a padlock symbol.

In a locked Vocabulary, all functions in the Vocabulary editor are display-only, including Custom Data Types and Database Access. **Undo** (and **Redo**) options will toggle the mode until the Vocabulary file is saved.

A locked Vocabulary can be unlocked by selecting the menu option **Vocabulary > Set to Read/Write**.

This feature gives OpenEdge users that implement BusinessRules through an imported `.brvd` file relief from the previous limitation of a locked down Vocabulary, so that they can create extended transients, enumeration lists, and constraint expressions for the imported Vocabulary.

See *Locking and unlocking Corticon Vocabularies* in the *Rule Modeling Guide* for more information.

Improved logic when updating a Vocabulary created from an OpenEdge file

Until this release, a Vocabulary that you created by importing a Business Rules Vocabulary Definition (BRVD) file created in OpenEdge could not be modified. A user importing a revised BRVD file (re-importing) had assurance that the originally imported Vocabulary was unchanged. Now that the Vocabulary can be unlocked, the advantages of extending and enhancing the original Vocabulary definitions require more precise logic of how the Vocabulary changes will apply.

The user is provided a description of the anticipated impact of the applied changes so that the changes can be documented and so that there is an opportunity to not apply the changes.

The status and intended actions on Vocabulary entities, attributes, and associations during a re-import are, in summary:

- **Match** - No action if the origin and characteristics of the element are unchanged.
- **UserDefined** - No action if the existing element was created in Corticon.
- **Add** - Adds elements in the BRVD that are unmatched in the Vocabulary
- **Remove** - Deletes an element that was created from an import of a BRVD element, but is not in the BRVD file being assessed for re-importation.
- **Remove/Re-add** - Removes the existing element and then recreates it as defined in the import file under certain conditions.
- **Merge** - Revises an existing element marked as created locally that is now a BRVD element in the import file.

See the topic *"Applying an updated BRVD to a Vocabulary"* in the *Rule Modeling Guide* for more information on this feature and the intended actions for each element type.

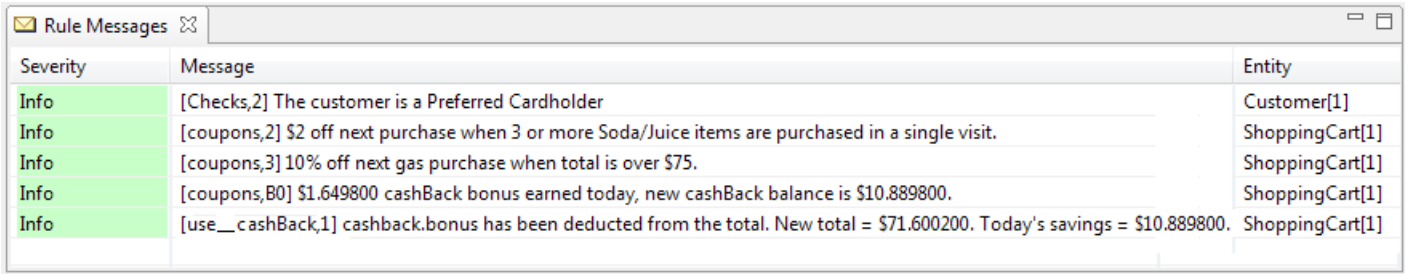
Also reference the Progress OpenEdge documentation for details about a complete end-to-end workflow involved in an integrated OpenEdge Business Rules environment.

Enhancements to rule testing and tracing in 5.3.4

Several improvements were made to the testing and tracing functions for Studio test servers and deployed servers. Significant items include:

Rule tracing

You can gain greater visibility into the rules that fired in the processing of a work document by enabling rule tracing. This will record the name of the rulesheet and the id of the rule that triggered each rule message in the response document. This is particularly useful when additional diagnostics are needed for a large application with many rulesheets and ruleflows. As an example here are the Rule Messages from the Advanced Tutorial when rule tracing is enabled:



Line 1 indicates: "In Rulesheet `Checks.ers`, rule 2 generated this statement", and line 4 indicates "In Rulesheet `coupons.ers`, line B's non-conditional (column 0) action generated this statement."

To enable this function, rule statement metadata, set the `CcServer` property as `com.corticon.reactor.rulestatement.metadata=true`

Note: It is recommended that you create or update the standard last-loaded properties file `brms.properties` to list override properties such as this for Corticon Studios and Servers. See the introductory topics in *"Configuring Corticon properties and settings" in the Server Integration and Deployment Guide* for information where to locate this properties file.

Note: The additional diagnostics are inserted into the code of the decision services generated by Corticon. Therefore, in Corticon Server, you must regenerate and recompile deployed deployment descriptors and decision services to activate this feature.

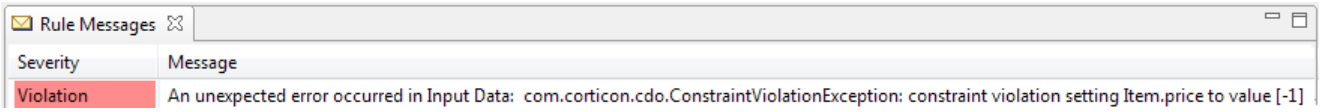
For more information, see *Running a Ruletest in Corticon Studio* in the *Rule Modeling Guide* and *Tracing rule execution* in the *Rule Modeling Guide*.

Option to relax enforcement of Custom Data Types

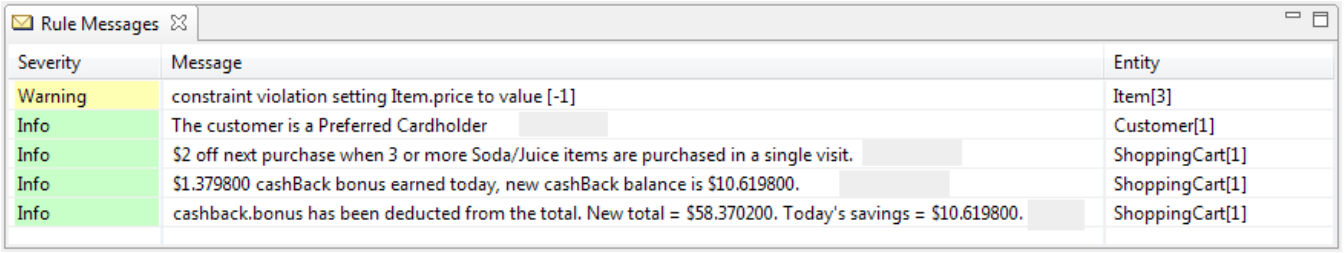
Using Custom Data Types lets you define general limitations of an attribute's values that are enforced on all Rulesheets and Ruletests in the project and its decision services. While they are valuable in focusing on what is valid in rule designs, violations of the constraints cause rule processing -- Ruletests in Studio; Decision Services in Servers -- to halt at the first constraint violation. Such exceptions indicate that values in attributes are not within numeric constraint ranges or not included in enumerated lists that have been set in the Vocabulary's Custom Data Types.

For Ruleflows, a rule that throws an exception in an early Rulesheet disables processing of the request in subsequent Rulesheets.

On the following Rule Messages tab, a constraint error - a negative value - stops subsequent rules from firing:



Relaxing the enforcement of Custom Data Type constraints produces warnings instead of violations, as shown:



This feature enables development teams and pre-production testing teams to expedite their debugging of rules and error handling. This example might indicate that the applications that format requests should handle the data constraint before forwarding a request into the rules engine. In a production environment, you should have Custom Data Type constraints enforced.

To enable this feature, set the property as `com.corticon.vocabulary.cdt.relaxEnforcement=true` in a `brms.properties` file on the Studio or Server.

Note: The additional diagnostics are inserted into the code of the decision services generated by Corticon. Therefore, in Corticon Server, you must regenerate and recompile deployed deployment descriptors and decision services to activate this feature.

For more information, see *Relaxing enforcement of Custom Data types* in the *Rule Modeling Guide* and *Running a Ruletest in Corticon Studio* in the *Rule Modeling Guide*.

Enhanced logging content

Server Logging is enhanced in this release to record Rule and Rulesheet tracings and Warnings generated when Custom Data Type enforcement is relaxed, as shown in this log excerpt of the Ruletest's `CorticonReponse`:

```
<Messages version="0.0">
    <Message postOrder="cc00000001">
      <severity>Warning</severity>
      <text>constraint violation setting Item.price to value [-1]</text>
      <entityReference href="Item_id_3" />
    </Message>
    <Message postOrder="cc00000002">
      <severity>Info</severity>
      <text>[Checks,2] The customer is a Preferred Cardholder</text>
      <entityReference href="Customer_id_1" />
    </Message>
    ...
    <Message postOrder="cc00000004">
      <severity>Info</severity>
      <text>[coupons,B0] $1.379800 cashBack bonus earned today, new cashBack
balance is $10.619800.</text>
      <entityReference href="ShoppingCart_id_1" />
    </Message>
   ...
  </Messages>
```

Extended Transients allowed as Input in Ruletests

Rulesheets that are used in Ruleflows often have dependencies on Extended Transient attributes that would, in production, be set by Rulesheets earlier in a the Ruleflow. When testing an individual Rulesheet, you can now set Extended Transient attributes as Input on a Testsheet. This allows you to validate the behavior of the individual Rulesheets in a larger Ruleflow.

Note: This feature applies only to Ruletests. It is not intended for production. If you export tests that include extended transient attributes to XML, and then run them on a deployment Server, the extended transient attributes are ignored.

See *Defining attribute properties: mode* in the *Quick Reference Guide*, *Modeling the Vocabulary in Corticon Studio* in the *Rule Modeling Guide*, and *Initializing null attributes* in the *Troubleshooting* section of the *Rule Modeling Guide* for more information.

Enhancements to logging in 5.3.4

Several improvements were made to the logging functions for deployed Corticon Servers. Significant items include:

Corticon Server deployments can maintain multiple discrete logs

Logs provide a record of server activity at a specified level of detail. When several Decision Services and versions are running, and multiple threads are handling requests, a Server instance records all those activities into one log. As research into logs often calls for setting the log level to provide more details, setting a high-detail level to analyze a given Decision Service version generates large volumes of log data for everything running on the Server. New features in this release provide greater control over logging functions so that you can:

- Specify separate log files for Decision Services
- Specify a different log detail level for each Decision Service version
- Choose to produce a separate log file for each thread running in a Decision Service version

These options are set through `modifyDecisionServiceExecutionProperty` methods that are defined through the `ICcServer` interface and maintained in the `CcServer.properties` (or `brms.properties`) file stored in a Server's `CcConfig.jar`. You can also set these properties through an in-process or SOAP call, the Corticon Server Console, or through the following TestServer API commands 245-247 `Modify Decision Service's Execution Property [Major|Major Minor]`.

If you do not specify an option at the Decision Service level, the Server-level log settings apply. The Decision Service Execution Property names and values are:

- `PROPERTY_EXECUTION_LOG_LEVEL=[ERROR|VIOLATION|TIMING|RULETRACE|INFO]`
- `PROPERTY_EXECUTION_LOG_PATH="`*`explicit path`*`"`
- `PROPERTY_EXECUTION_LOG_PER_THREAD=[true|false]`

Note: You can also use this technique to restrict rulemessage types at the Decision Service level, a feature introduced in Corticon 5.3.3:

- `PROPERTY_EXECUTION_RESTRICT_RULEMESSAGES_INFO=[true|false]`
- `PROPERTY_EXECUTION_RESTRICT_RULEMESSAGES_WARNING=[true|false]`
- `PROPERTY_EXECUTION_RESTRICT_RULEMESSAGES_VIOLATION=[true|false]`

where the default setting at the Server level is `false`. For information on the Server settings, see *"Logging at the Corticon Server level" in the Server Integration and Deployment Guide*

All non-default settings are retained in the Server's `CcServerState.xml` file, so that when the Server is restarted, the overridden property values are re-established for the relevant Decision Services.

The Decision Service properties can be queried through the `ICcServer` interface in the Test Server API commands 214-216 `Get Decision Service property value [Major|Major Minor]` to get the current values.

See the section *"Management of Corticon Logs" in the Integration and Deployment Guide* for information on log content, and differences between Server logging and Decision Service logging.

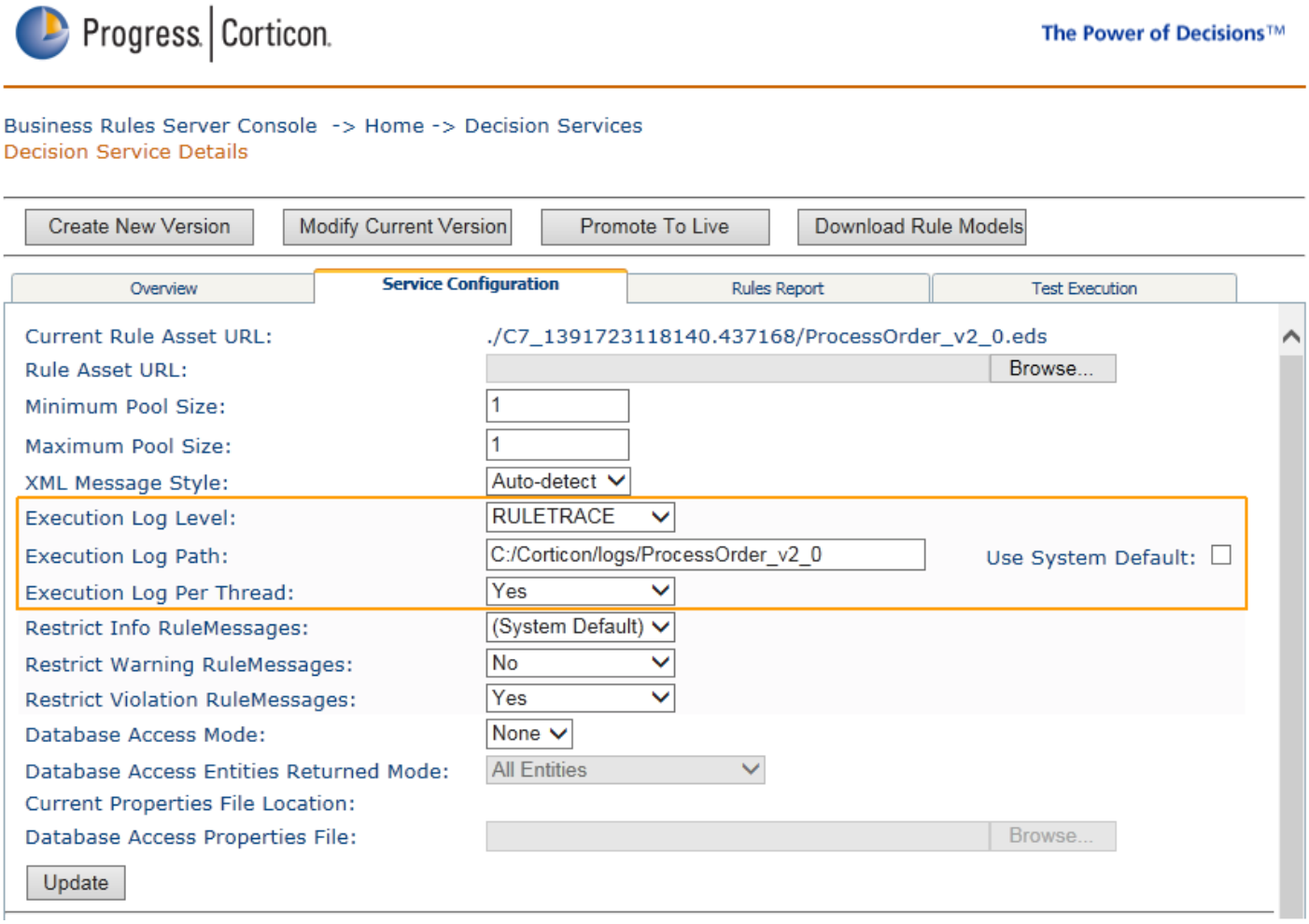
The log at the Server level can log by thread

The property that enables thread level logs at Decision Service level is supported by a corresponding property, `com.corticon.server.execution.logperthread=false`, added to `CcServer.properties` or `brms.properties` file within the Server's `CcConfig.jar`. See *"Corticon Server properties (CcServer.properties)" in the Server Integration and Deployment Guide*.

Using Java Server's Console to set logging features

For Java Servers, the Server Console enables setting and reviewing logging settings and defaults for a specified Decision Service version:

Figure 1: Example of settings for logging options in Java Server Console



Notice that, on this Java Server Console page, you can set the restrictions on rulemessage types at the Decision Service version level.

See *"Decision Service actions and information" in the guide to Deploying Web Services with Java* for information on the Server Console.

Note: Each of the Decision Service-log settings can choose the System Default value to defer to the corresponding setting at the Server-level log.

Enhancements to compiling Decision Services in 5.3.4

New in this release is the ability to easily generate Decision Service (`.eds`) files with command line utilities. This allows you to create batch scripts or other build procedures to take your rule assets, and then compile them into deployable EDS files. Previously, this needed to be done manually using the Corticon Deployment Console. The utilities for this are included with both the Corticon Server for Java and the Corticon Server Archive. If you are a .NET user you can use the utilities in the Corticon Server Archive to create EDS files which can be deployed to Corticon Server .NET.

Silent compilation of multiple Decision Services

Compiling Decision Services one by one through the manual steps of the several Corticon deployment tools can be prone to error. Using the Multiple Compilation feature, you can define parameters for several Ruleflows, and then compile them by launching a simple script with no parameters. Logs specific to these compilations provide documentation of the activities.

The essence of this feature is in the file `multipleCompilation.xml`. The following excerpts shows the required elements of a compilation object:

```
<MultipleCompilation>
 <CompilationLogDirectory>**Fully qualified path to directory where log will
be placed**</CompilationLogDirectory>
 <CompilationObjects>
  <CompilationObject>
   <DecisionServiceName>**Name of the Decision Service**</DecisionServiceName>

   <RuleflowPath>**Explicit path to the Ruleflow to compile**</RuleflowPath>
   <OutputDirectory>**Explicit path to output directory for the  .eds
file**</OutputDirectory>
   <OverrideIfExists>**true/false: Determines whether to overwrite a matching
 file in the output directory**</OverrideIfExists>
   <DatabaseAccessMode>**empty value/R/RW: Determines if and how the Rules
will be compiled for EDC compatibility**</DatabaseAccessMode>
  </CompilationObject>
  <CompilationObject>
   ...
  </CompilationObject>
 </CompilationObjects>
</MultipleCompilation>
```

Once the compilation objects are defined and the Ruleflows placed at their staging location, launching `multipleCompilation.bat` compiles each of the Ruleflows into its target Decision Service.

For more information and an example, see the topic *"Silent compilation of multiple Decision Services" in the Integration and Deployment Guide*.

Enhancements to handling of requests in disparate locales, languages, and timezones

When deploying decision services that will be consumed by users or services running in different locales, you often need to address issues with locale-dependent data formats and localized messages. New in this release is the ability to specify a "locale" when calling a decision service. When locale is specified, Corticon uses it when parsing and formatting locale-dependent data types such as Decimals and Dates. In addition Corticon will return localized Rule Messages if you defined localizations for the messages when creating the Rulesheets for your decision service. In prior releases you would have needed to deploy a decision service multiple times, once for each locale supported, to have localized Rule Messages returned. This is no longer necessary. A single deployed decision service can support multiple locales.

Note:

Localizing your rule modeling and processing environment can implement five related functions:

1. Displaying the Studio **program** in your locale of choice. This means switching the Corticon Studio user interface (menus, operators, system messages, etc.) to a new language. See *"Enabling Studio internationalization" in the Studio Installation Guide*. This is not a new feature in this release.
2. Displaying your Studio **assets** in your locale of choice. This means switching your Vocabularies, Rulesheets, Ruleflows, and Ruletests to a new language. See *"Localizing Corticon Studio" in the Rule Modeling Guide*. This is not a new feature in this release.
3. Displaying your localized Rulesheet's **rule statements** in your locale of choice. Rulesheets can specify rule statements in another language that are returned to requestors when the server is set to that language. This is not a new feature in this release. However, as of this release, when a request's execution property specifies a language that has defined appropriate rule statements, the locale-specific statements are included in the response. See *"Localizing Corticon Studio" in the Rule Modeling Guide*.
4. Enabling requests submitted to a Corticon Server to set an execution property that indicates the locale of the incoming payload so that the server can transform the payload's **locale-specific decimal and date literal values** to the decimal delimiter and month literal names of the server, run the rules, and return the output formatted for the submitter's specified locale. See the topics in *"Enabling Server localization and handling of locales" in the Rule Modeling Guide*.This feature is new in this release.
5. Enabling requests submitted to a Corticon Server to set an execution property that indicates the **timezone** of the incoming payload so that the server can transform the payload's time calculations to the timezone of the server, run the rules, and return the output formatted for the submitter's specified timezone. See the topics in *"Enabling Server localization and handling of locales" in the Rule Modeling Guide*.This feature is new in this release.

The first two functions are primarily of value to developers. The third function also helps developers yet it enhances the user experience when rule statements are returned in the server's languages, or -- as of this release -- the specified locale of the request message matches a defined language. The fourth function enables a server to handle requests from various locales at one server location, and to avoid exceptions thrown when the originating locale's decimal and date literal values are incompatible with those of the server. The fifth function enables precise handling of time functions.

Handling of decimal delimiters and literal dates across locales

When Decision Services are compiled and deployed in 5.3.4, there is a change in how rule statements are localized. Prior to this release, rule messages were localized to the locale of the server at compile and deployment time. Now, the locale is determined at rule execution time, and can be set with a message property in the service request document. When a rule statement is posted, its message locale is compared with what is defined in the Rulesheet. The process tries to match the language and country locale, otherwise it tries to match the language with no country code. If no matches are found, the process defers to the default language and country locale of the server's JVM.

When a Corticon service request document provides data formats that are unsupported by the Server, the request throws an exception. The two most common issues are:

- **Inconsistent parsing of the decimal delimiter** - For example, a message supplies a comma delimited decimal value (such as "157,1") and the Server is expecting a period ("157.1")
- **Inconsistent name of a literal month name** - For example, a message supplies a French month name (such as "avril") and the Server is expecting an English name ("April")

Now, in this release, an inbound message can provide the locale of the message payload in the form:

```
<ExecutionProperties>
 <ExecutionProperty name="PROPERTY_EXECUTION_LOCALE" value="language-country"
 />
<ExecutionProperties>
```

where *language-country* is the JVM standard identifier, such as `en-US` for English-United States. When the message's locale is specified, it is used at rule execution time regardless of the Server's default language. If the Rulesheet has a matching locale, those rule statement messages are used. Whether or not there is a match, the JVMs functionality enables it to map the input request's decimal delimiters and the literal month names to the server locale's corresponding format. When rule processing is complete, the output response maps the results to the formats of the requestor's locale, and--when rule statement messages are available for the requestor's locale--messages for that locale are included. Matching a literal month name must have the appropriate case and diacritical marks, such as août, février, décembre and März. When this property is not set on an inbound request, the Corticon Server assumes the locale of the server machine, or the language that is set as an override in the Java startup of the server. That setting will use locale settings in Corticon Rulesheets for rulestatement messages so that a server running the Rulesheet's Decision Service would get rule statements that are specified for that locale. See the topics and examples in the section *"Handling requests and replies across locales" in the Server Integration and Deployment Guide*.

Handling of timezones

When a Corticon service request document provides a timezone that is different from the server's timezone, the server processes the time as though it is the server's local time and returns results based on the server's local time.

Now, in this release, an inbound message can provide the timezone of the message payload in the form:

```
<ExecutionProperties>
 <ExecutionProperty name="PROPERTY_EXECUTION_TIMEZONE" value="TZ_string" />
<ExecutionProperties>
```

where *TZ_string* is the text in the TZ column of TZ database for the zone, such as `US/Eastern` for the Eastern timezone of the United States. When the message's timezone is specified, it is

used at rule execution time regardless of the Server's default language. When rule processing is complete, the output response maps the results to the formats of the requestor's timezone. When this property is not set on an inbound request, the Corticon Server assumes the timezone of the server machine. See the topics and examples in the section *"Example of requests that cross timezones" in the Server Integration and Deployment Guide*.

Changes to display of Time and DateTime across timezones in 5.3.4

The handling of messages across locales was improved by changing the way time is handled in the reply message when the server and the input message request are in different timezones.

The display of **Time** and **DateTime** attributes changed as follows:

- **Time** values were not showing the timezone value when printed in ISO time format. Previously, a **Time** value of `4:59:59 AM PDT` in ISO format displayed as `04:59:59.000`. That value will now display, with the timezone offset, such as `04:59:59.000-08:00`.

  Note that **DateTime** values already have the timezone offset. Their display is unchanged.

- Both **Time** and **DateTime** values display in the timezone specified in the value when possible. For example, when a request passes in a value marked as `EST`, the output return value will show `EST`. Previously, the timezone returned was the timezone of the server -- an `EST` value would display in `PST` (offset by three hours of the value) if the Server timezone was `PST`.

  If the value does not include a timezone, then the result uses the default timezone, the server's timezone unless the request included `PROPERTY_EXECUTION_TIMEZONE` set to a TZ value such as `"US/Pacific"`.

Note: Some Corticon operators always use the default timezone, such as `today`, `now`, and `toTime`.

2

What was new and changed in Corticon 5.3.3

This chapter summarizes the new, enhanced, and changed features in Progress® Corticon® 5.3.3. Service Pack 3 included the changes that were released in Service Packs 1 and 2.

The 5.3.3 release coordinated with other Progress Software releases:

- Progress OpenEdge is available as a database connection. You can read from and write to an OpenEdge database from Corticon Decision Services. This feature is distinct from the integration of Business Rules from Corticon with an OpenEdge application, introduced in Corticon 5.3.2. OpenEdge is a separately licensed Progress Software product.
- Progress DataDirect Cloud (DDC) enables simple, fast connections to cloud data regardless of source. In the Corticon 5.3.3 release, to cloud data sources such as Rollbase and SalesForce.com are available. DataDirect Cloud is a separately licensed Progress Software product.

For details, see the following topics:

- Enhancements to the Enterprise Data Connector in 5.3.3
- Enhancements to Rule Modeling in Studio 5.3.3
- Enhancements to Corticon Servers and Deployments in 5.3.3

Enhancements to the Enterprise Data Connector in 5.3.3

Several improvements were made to EDC in Corticon 5.3.3. Significant items include:

- **Connection to Progress OpenEdge** -- You can now configure Corticon to read from and write to OpenEdge database tables. Corticon's EDC connects directly to the database through the bundled Data Direct JDBC driver. You can map a Corticon vocabulary to an existing OpenEdge database schema or create/update an OpenEdge schema from Corticon to match a Corticon vocabulary. You can extract database metadata and lookup enumerations from the database. See the Progress Software web page Progress Corticon 5.3 - Supported Platforms Matrix for more information.
- **Connection to DataDirect Cloud (DDC)** -- You can now configure Corticon to use data sources that are accessed through DDC using the bundled DDC JDBC driver. DDC provides access to many data sources. In this release Corticon supports the Salesforce, and Progress Rollbase data sources.

  When connecting to a DDC data source you must have both a DDC account and an account for the targeted data source. For example if accessing Salesforce you must have a Salesforce account and have that account configured in DDC. You would then use your DDC account to access Salesforce via Corticon's EDC feature.

  When accessing a DDC datasource with Corticon's EDC you can easily read and write to database tables from within your Corticon rules as you would any other database. When using a DDC data source you define your vocabulary in Corticon, and then map it to tables and fields defined in the data source's schema.

  With all DDC data sources there are some restrictions on EDC functionality. In this release there is no support for EDC's create/update schema feature, read-only access, or transactions. The lack of transaction support in DDC means you must have `TransactionMode=ignore` in the database URL specified for connecting to DDC. Corticon will add this by default.

  See the Corticon EDC documentation for individual data sources for additional notes on using the data source.

  For more details on Corticon supported datasources, see the Progress Software web page Corticon 5.3 - Supported Platforms Matrix

For more information, see the topic *"Features in supported database brands" in the Corticon Tutorial: Using the Enterprise Data Connector (EDC).*

Enhancements to Rule Modeling in Studio 5.3.3

Several improvements were made to the user interface in Corticon Studio and Corticon Studio for Analysts. Significant items include:

- **Ruletest Scroll Lock** - On Ruletests, you can choose to have all the columns scroll together.

  To turn the feature on and off, click on the **Scroll Lock** button  in the Ruletest toolbar or on the testsheet pop-up menu. For more information, see the topic *"Context-sensitive right-click pop-up menus" and the Ruletest menu commands in the Corticon Studio: Quick Reference Guide.*
- **Unattended (silent) installs of Studio** - If you want to perform unattended installations, see the detailed instructions on creating and using response files to perform silent installations in *“Performing Silent Installations” section in the “Downloading and running the Corticon Studio installers and updaters” chapter of the Corticon Studio Installation Guide..*

Enhancements to Corticon Servers and Deployments in 5.3.3

Several improvements were made to the Java and .NET servers in Corticon 5.3.3. Significant items include:

- **Properties to restrict logging of messages** -- There are settings that restrict each of the three types of Rule Messages (info, warning, and violation) from being posted to the output of an execution. The default value for each of the properties is `false` -- that message type is not restricted. The properties can be set as follows to restrict message types:

  ```
  com.corticon.server.restrict.rulemessages.info=true
  com.corticon.server.restrict.rulemessages.warning=true
  com.corticon.server.restrict.rulemessages.violation=true
  ```

  As with other properties, Corticon Studio property settings are specified as overrides in a `brms.properties` file, while Corticon Server properties files are in the `CcServer.properties` file within the appropriate `CcConfig.jar`.

  For more information, see the topic *"Configuring Corticon properties and settings" section of the Integration and Deployment Guide.*

- **Array support for .NET object associations** -- When embedding Corticon within a .NET process, you can now use arrays with object associations. An example of usage is `private ObjectB[] myAsociationToObjectBs`. For more information, see the topic *"Using .NET Business Objects as payload for Decision Services" section of the guide Deploying Web Services with .NET.*

- **Enhanced documentation on effective timestamp and versioning** -- The documentation on this topic has been revised and updated. It now includes a discussion of differences between test Decision Service and a production (live) Decision Service. For more information, see the topics under *"Decision Service versioning and effective dating" in the Integration and Deployment Guide.*

- **Link to information on installing the server on other application servers and other platforms** -- See the Corticon KnowledgeBase entry Corticon Server 5.X sample EAR/WAR installation for different Application Servers for detailed instructions on configuring Apache Tomcat, JBoss, WebSphere, WebLogic on all supported platforms.

3

What was new and changed in Corticon 5.3.2

This chapter summarizes the new, enhanced, and changed features in Progress® Corticon® 5.3.2. For details, see the following topics:

- Introducing Open Edge Business Rules
- Enhancements to the Enterprise Data Connector in 5.3.2
- Enhancements to Rule Modeling in Studio 5.3.2
- Enhancements to Corticon Servers and Deployments in 5.3.2

Introducing Open Edge Business Rules

Progress OpenEdge® is our complete development platform for building dynamic multi-language applications for secure deployment across any platform, any mobile device, and any Cloud. This release of Corticon introduces its integration with OpenEdge, empowering you to build an integrated solution that incorporates OpenEdge BPM and Corticon BRMS. These complementary technologies in a single application development platform enable you to use ABL data structures (such as ProDataSets and temp-tables) as Corticon data structures (Vocabularies).

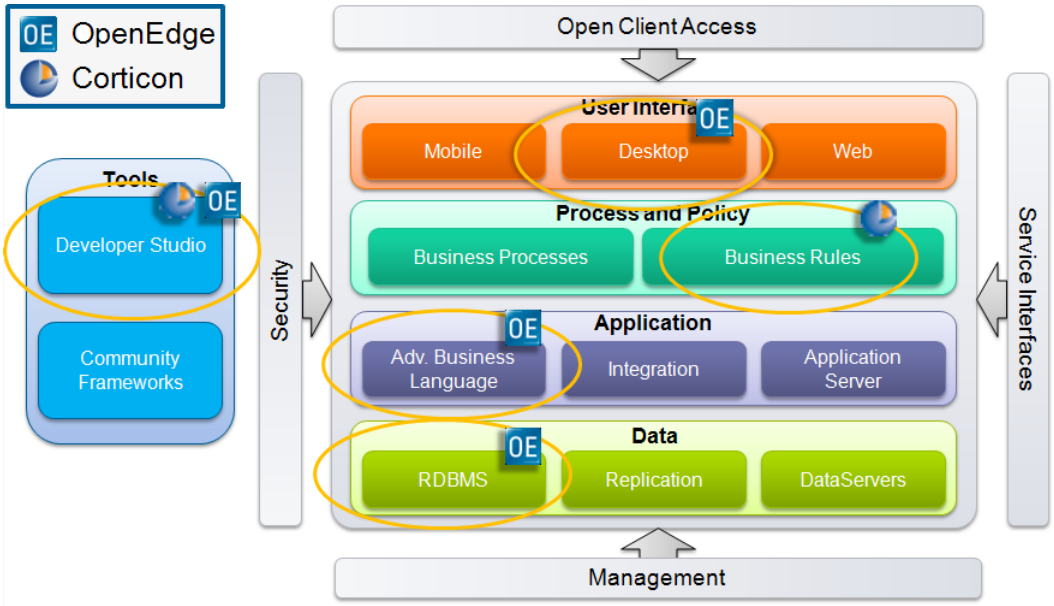
Progress Developer Studio for OpenEdge and Progress Corticon Studio integrate into a single Eclipse instance to provide tooling support for creating and updating Corticon Vocabularies, and runtime support for simple invocation of Corticon Decision Services.

The Corticon documentation in this release points out key aspects of the integration from the Corticon point of view:

- When OpenEdge creates a Business Rules Vocabulary Defintion (BRVD) file, that `.brvd` file gets imported into Corticon Studio as a completely defined Vocabulary that includes the ProDataSet or temp-table schema information. For more information, see the *Progress OpenEdge* documentation, and the Corticon topic *"Importing an OpenEdge Business Rules Vocabulary Definition (BRVD) file" in the Rule Modeling Guide.*
- The integration with OpenEdge constrains its interactions with a connected OpenEdge database to read-only functions. For more information, see the topic *"Constraints on OpenEdge BRVD Vocabularies" in the Rule Modeling Guide*.
- Default licensing of Corticon Server usage with OpenEdge constrains where you install it and how many Decision Services can be executed concurrently. For more information, see the topic *"Updating your Corticon license in Studio" in the Studio: Installation Guide.*
- Corticon Studio now packages the Corticon Server WAR file at installation. This enables OpenEdge developers (as well as other Corticon developers) to use their Tomcat or other application server to readily test their Decision Services as a remote server. For more information, see the topic *"Installed components" in the Studio: Installation Guide.*

OpenEdge/Corticon Integration Points

The following model highlights the OpenEdge and Corticon functions that facilitate the integration:



Enhancements to the Enterprise Data Connector in 5.3.2

Several improvements were made to EDC in Corticon 5.3.2. Significant items include:

- **Additional RDBMS brands are supported by the Enterprise Data Connector** -- For more information, see the Progress Software web page Progress Corticon 5.3 - Supported Platforms Matrix.

- **Database views now allow joins.** For more information, see updates to the topic *"Support for database views" in the Integration and Deployment Guide.*
- **Retrieve enumeration values from a database column** -- When you have EDC setup and connected to a database (see *"Connecting a Vocabulary to a database" in the Using EDC Guide*), you can use Custom Data Types to retrieve name/value or just value lists from specified columns in a table, as described in the following topics:
  - *"Enumerated values" in the Quick Reference Guide.*
  - *"Enumerations retrieved from a database" in the Rule Modeling Guide*
  - *"Importing an attribute's possible values from database tables" in the Using EDC Guide*
- **Special Ruletest Editor Behaviors for Remote Decision Services** -- Database access menu items are now disabled on remote decision services. For more information, see the topic *"Database Access menu commands/options not available on remote server" in the Using EDC Guide.*
- **Validation of names against reserved words** - When generating a Vocabulary to a database schema, Corticon tries to trap overloading of reserved words. You should take caution to avoid such terms. For more information, see the topic *"Vocabularies and databases" in the Using EDC Guide.*

Enhancements to Rule Modeling in Studio 5.3.2

Several improvements were made to the user interface in Corticon Studio and Corticon Studio for Analysts. Significant items include:

- **Query optimization** - Database queries where a database extended alias is used with an aggregation operator can be optimized when you provide non-conditional rules (column 0 actions) to defer the aggregate calculations to the database. When you do this, the entities used in a database aggregation are not loaded into memory. For more information, see the following topics:
  - The collection operators for aggregations in *"Aggregations that can optimize database access" in the Rule Modeling Guide.*
  - An example that demonstrates unoptimized and optimized action syntax in *"Optimizing Aggregations that Extend to Database" in the Rule Modeling Guide.*
- **Indicator when an inferred value has been overridden** - While you can see on an item-by-item basis whether a value is the one inferred from a database value (light gray text in its column property) or overridden (black gray text in its column property), you can see the overrides in the tree view by noting whether the database decoration has a black bar at its center, as illustrated for an entity:

  Similar decoration is applied to overridden attributes and associations. For more information, see updates to the topic *"Inferred property values (Best Match)" in the Integration and Deployment Guide.*

- **Sequence diagrams differentiate database filters** - When you have extended to database and defined a filter as a database query, a sequence diagram distinguishes these filters by their shape. For more information, see the updates to the topic *"Database Filters" in the Rule Modeling Guide.*
- **Studio now defaults `validate.on.activation` property to `true`** - The property `com.corticon.validate.on.activation` in the `CcCommon.properties` within `CcConfig.jar` now defaults to `true` in Studio installations. (Its default value is, and has been, `true` in Server configurations.) This is a change in behavior from prior releases to improve performance in the Studio. For more information, see the topic *"Common Properties" in the Integration and Deployment Guide.*

Enhancements to Corticon Servers and Deployments in 5.3.2

Several improvements were made to the Java and .NET servers in Corticon 5.3.2. Significant items include:

- **Server Console supports LDAP** -- Corticon Java Server Console lets you elect to use existing Lightweight Directory Access Protocol (LDAP) domains for role-based authentication, so that you can control access to Corticon Server Console and define roles in your current user management systems, such as Microsoft's Active Directory. For more information, see the topic *"Using LDAP authentication in Server Console" in the "Using the Corticon Server Console" section of the guide Deploying Web Service with Java.*
- **Logging of service name and requestor's IP address** -- Logs can now include the requestor's IP address and the service name to help in diagnosing errors with client calls to Corticon. For more information, see updates to the topic *"Logging" in the "Performance and Tuning" section of the Integration and Deployment Guide.*
- **Logging History of a Rule Execution** -- Setting the log level to RULETRACE logs details of every CorticonRequest and CorticonResponse. For more information, see updates to the topic *"Logging" in the "Performance and Tuning" section of the Integration and Deployment Guide.*
- **New test API command** -- A new command, `204 – isDecisionServiceDeployedEffectiveTimestamp` has been added to the test-API scripts and executables. For more information, see the topic *"Specifying Decision Service effective timestamp in a SOAP request message" in the Integration and Deployment Guide.*

  Note: This additional command to the test scripts was slotted into the existing sequence, thereby requiring all the higher numbered commands in the 200 series to move one number higher. For example, the **Display XML Report** commands that were numbers **223, 224, 225** are now numbers **224, 225, 226**. All other command series are unchanged.

- **Using .NET Business Objects as payload for Decision Services, and support for Nullable attributes in .NET Server** -- The techniques for using sample Corticon C# libraries to generate stubs that include nullables are described in a section of the .NET server guide. For more information, see the topic *"Using .NET Business Objects as payload for Decision Services" in the guide Deploying Web Services with .NET.*

4

What was new and changed in Corticon 5.3.1

This chapter summarizes the new, enhanced, and changed features that were in Progress® Corticon® Service Pack 1.

For details, see the following topics:

- Relational database access through the Enterprise Data Connector
- Enhancements to Filters in 5.3.1
- Improvements to Corticon Server for .NET in 5.3.1
- User Interface Improvements in 5.3.1
- Other Changes and Improvements in 5.3.1

Relational database access through the Enterprise Data Connector

Corticon 5.3.1 included a powerful feature that lets Corticon assets interact with a commercial RDBMS. This feature is named Enterprise Data Connector or "EDC", and is sometimes referred to as Direct Database Access or "DDA".

This feature was included in Corticon 4.3 and is returning now in this Service Pack release. EDC has been updated to use Hibernate, the most widely used library for Object/Relational mapping. This change to Hibernate brings with it many new and improved capabilities.

This release provides support for Microsoft SQL Server and Oracle. Bundled with the release are the Progress Data Direct database drivers necessary for accessing these databases. You no longer need to provide your own driver.

If you are a current user of EDC in Corticon 4.3 and anticipate upgrading to this implementation of EDC, contact your support representative to ensure that your requirements will be met, and then to develop your migration strategy

A new tutorial, *Using Enterprise Data Connector (EDC)*, provides a focused walkthrough of EDC setup and basic functionality.

Other documentation material provides additional information on EDC:

- *Writing Rules to access external data* chapter in the *Rule Modeling Guide* extends the tutorial into scope, validation, collections, and filters.
- *Relational database concepts in the Enterprise Data Connector (EDC)* in the *Integration and Deployment Guide* discusses identity strategies, key assignments, catalogs and schemas, database views, table names and dependencies, inferred values, and join expressions.
- *Implementing EDC* in the *Integration and Deployment Guide* discusses the mappings and validations in a Corticon connection to an RDBMS.
- *Deploying Corticon Ruleflows* in the *Integration and Deployment Guide* describes the Deployment Console parameters for Deployment Descriptors and compiled Decision Services that use EDC.
- *Vocabularies: Populating a New Vocabulary: Adding nodes to the Vocabulary tree view* in the *Quick Reference Guide* extends its subtopics to detail all the available fields for Entities, Attributes, and Associations.

Enhancements to Filters in 5.3.1

Filters were improved so that a filter that applies to multiple attributes and associations can be disabled at any selected level in the hierarchy. When all levels of a filter are enabled, that is referred to now as a **full filter,** a term that replaces *maximum filter* that was used in prior releases. Once a level is selected, and its filter disabled, the whole filter structure is referred to as a **limiting filter**, a term that expands on the notion of the *minimum filter* in prior releases.

Filters in existing assets will automatically be updated to use the improved filter capabilities.

See the "Filters & preconditions" chapter of the *Rule Modeling Guide* for a discussion and examples of the revised behaviors.

Improvements to Corticon Server for .NET in 5.3.1

The following enhancements were in the 5.3.1 release of Corticon Server for .NET:

- Mapping of Java Object Messaging (JOM) to the .NET Framework Date type `cli.System.DateTime`
- Improved documentation with detailed instructions on setup of Microsoft's .NET Framework and Internet Information Services (IIS) on several supported platforms.
- Improved samples demonstrating how to call decisions services from .NET clients.

See the *Deploying Web Services with .NET* guide for more information.

User Interface Improvements in 5.3.1

Several improvements were made to the user interface in Corticon Studio and Corticon Studio for Analysts. Significant items include:

- Rows in rule tables can wrap text by dragging a row to a larger height, and then dragging a column narrower causing the text to wrap into the assigned row height.
- The behavior of cut/copy/paste is improved in several contexts.
- In the Vocabulary, an Attribute's mode options no longer offer **Extended Persistent** as a property value.
- The Server Console's four Decision Service options are now **ON** by default.
- Added presentation-style formats of the Basic Tutorial and Advanced Tutorial to Corticon Studio for Analysts **Help** menu.

Other Changes and Improvements in 5.3.1

Other changes and improvements were included in Corticon 5.3.1

- The SOAP interface for calling decision services has been updated to address compliance issues between the generated WSDL for a Decision Service and responses sent for requests using the SOAP interface.
- The size of EDS files has been significantly reduced by the exclusion of encrypted generated source code for the Decision Service (EDS) file. The source previously had been included but in encrypted form. This information was not necessary in the EDS file.

A

Overview of Progress Corticon

Progress® Corticon® is the Business Rules Management System with the patented "no-coding" rules engine that automates sophisticated decision processes.

Progress Corticon products

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

- **Corticon Studios** are the Windows-based development environment for creating and testing business rules:
  - **Corticon Studio for Analysts.** is a standalone application, a lightweight installation that focuses exclusively on Corticon.
  - **Corticon Studio** is the *Corticon Designer* perspective in the **Progress Developer Studio** (PDS), an industry-standard Eclipse and Java development environment. The PDS enables development of applications integrated with other products, such as Progress OpenEdge.

  The functionality of the two Studios is virtually identical, and the documentation is appropriate to either product. Documentation of features that are only in the *Corticon Designer* (such as on integrated application development and Java compilation) will note that requirement. Refer to the *Corticon Studio: Installation Guide* to access, prepare, and install each of the Corticon Studio packages.

  Studio Licensing - Corticon embeds a time-delimited evaluation license that enables development of both rule modeling and Enterprise Data Connector (EDC) projects, as well as testing of the projects in an embedded Axis test server. You must obtain studio development licenses from your Progress representative.

- **Corticon Servers** implement web services for business rules defined in Corticon Studios:
  - **Corticon Server for deploying web services with Java** is supported on various application servers, and client web browsers. After installation on a supported Windows platform, that

server installation's deployment artifacts can be redeployed on various UNIX and Linux web service platforms as Corticon Decision Services. The guide *Corticon Server: Deploying web services with Java* provides details on the full set of platforms and web service software that it supports, as well as installation instructions in a tutorial format for typical usage.

- **Corticon Server for deploying web services with .NET** facilitates deployment of Corticon Decision Services on Windows .NET Framework and Microsoft Internet Information Services (IIS). The guide *Corticon Server: Deploying web services with .NET* provides details on the platforms and web service software that it supports, as well as installation instructions in a tutorial format for typical usage.

Server Licensing - Corticon embeds a time-delimited evaluation license that enables evaluation and testing of rule modeling projects on supported platform configurations. You must obtain server deployment licenses and server licenses that enable the Enterprise Data Connector (EDC) from your Progress representative.

B

Progress Corticon documentation

The following documentation, as well as a *What's New in Corticon* document, is included with this Progress Corticon release:

Corticon Tutorials	
Corticon Studio Tutorial: Basic Rule Modeling	Introduces modeling, analyzing, and testing rules and decisions in Corticon Studio. Recommended for evaluators and users getting started. *See also the PowerPoint-as-PDF version of this document that is accessed from the Studio for Analysts'* ***Help*** *menu.*
Corticon Studio Tutorial: Advanced Rule Modeling	Provides a deeper look into Corticon Studio's capabilities by defining and testing vocabularies, scope, collections, messages, filters, conditions, transient data, and calculations in multiple rulesheets that are assembled into a Ruleflow. *See also the PowerPoint-as-PDF version of this document that is accessed from the Studio for Analysts'* ***Help*** *menu.*
Corticon Tutorial: Using Enterprise Data Connector (EDC)	Introduces Corticon's direct database access with a detailed walkthrough from development in Studio to deployment on Server. Uses Microsoft SQL Server to demonstrate database read-only and read-update functions.
Corticon Studio Documentation: Defining and Modeling Business Rules	
Corticon Studio: Installation Guide	Step-by-step procedures for installing Corticon Studio and Corticon Studio for Analysts on computers running Microsoft Windows. Also shows how use other supported Eclipse installations for integrated development. Shows how to enable internationalization on Windows.

Corticon Studio: Rule Modeling Guide	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 and many *Test Yourself* exercises.
Corticon Studio: Quick Reference Guide	Reference guide to the Corticon Studio user interface and its mechanics, including descriptions of all menu options, buttons, and actions.
Corticon Studio: Rule Language Guide	Reference information for all operators available in the Corticon Studio Vocabulary. A Rulesheet example is provided for many of the operators. Includes special syntax issues, handling arithmetic and character precedence issues.
Corticon Studio: Extensions Guide	Detailed technical information about the Corticon extension framework for extended operators and service call-outs. Describes several types of operator extensions, and how to create a custom extension plug-in.
Corticon Server Documentation: Deploying Rules as Decision Services	
Corticon Server: Deploying Web Services with Java	Details installing the Corticon Server as a Web Services Server, and then deploying and exposing Decision Services as Web Services on Tomcat and other Java-based servers. Presents the features and functions of the browser-based Server Console.
Corticon Server: Deploying Web Services with .NET	Details installing the Corticon Server as a Web Services Server, and then deploying and exposing decisions as Web Services with .NET. Provides installation and configuration information for the .NET Framework and Internet Information Services (IIS) on various supported Windows platforms.
Corticon Server: Integration & Deployment Guide	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. 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 samples, server monitoring techniques, and recommendations for performance tuning.

C

Copyright

D

Third party acknowledgments

One or more products in the Progress Corticon v5.3.4 release includes third party components covered by licenses that require that the following documentation notices be provided:

Progress Corticon v5.3.4 incorporates Apache Commons Discovery v0.2 from The Apache Software Foundation. Such technology is subject to the following terms and conditions: The Apache Software License, Version 1.1 - Copyright (c) 1999-2001 The Apache Software Foundation. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The end-user documentation included with the redistribution, if any, must include the following acknowlegement: "This product includes software developed by the Apache Software Foundation (http://www.apache.org/)." Alternately, this acknowledgement may appear in the software itself, if and wherever such third-party acknowledgements normally appear.

4. The names "The Jakarta Project", "Commons", and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

5. Products derived from this software may not be called "Apache" nor may "Apache" appear in their names without prior written permission of the Apache Group.

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

Progress Corticon v5.3.4 incorporates Apache SOAP v2.3.1 from The Apache Software Foundation. Such technology is subject to the following terms and conditions: The Apache Software License, Version 1.1 Copyright (c) 1999 The Apache Software Foundation. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Apache Software Foundation (http://www.apache.org/)." Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear. 4. The names "SOAP" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org. 5. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation. THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

Progress Corticon v5.3.4 incorporates DOM4J v1.6.1. Such technology is subject to the following terms and conditions: Project License BSD style license Copyright 2001-2005 (C) MetaStuff, Ltd. All Rights Reserved.

Redistribution and use of this software and associated documentation ("Software"), with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain copyright statements and notices. Redistributions must also contain a copy of this document.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name "DOM4J" must not be used to endorse or promote products derived from this Software without prior written permission of MetaStuff, Ltd. For written permission, please contact dom4j-info@metastuff.com.

4. Products derived from this Software may not be called "DOM4J" nor may "DOM4J" appear in their names without prior written permission of MetaStuff, Ltd. DOM4J is a registered trademark of MetaStuff, Ltd.

5. Due credit should be given to the DOM4J Project - http://www.dom4j.org

THIS SOFTWARE IS PROVIDED BY METASTUFF, LTD. AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL METASTUFF, LTD. OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT,

INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Progress Corticon v5.3.4 incorporates Jaxen v1.0. Such technology is subject to the following terms and conditions: JAXEN License - $Id: LICENSE,v 1.3 2002/04/22 11:38:45 jstrachan Exp $ - Copyright (C) 2000-2002 bob mcwhirter and James Strachan. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions, and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the disclaimer that follows these conditions in the documentation and/or other materials provided with the distribution.

3. The name "Jaxen" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact license@jaxen.org.

4. Products derived from this software may not be called "Jaxen", nor may "Jaxen" appear in their name, without prior written permission from the Jaxen Project Management (pm@jaxen.org).

In addition, we request (but do not require) that you include in the end-user documentation provided with the redistribution and/or in the software itself an acknowledgement equivalent to the following: "This product includes software developed by the Jaxen Project (http://www.jaxen.org/)." Alternatively, the acknowledgment may be graphical using the logos available at http://www.jaxen.org/. THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE Jaxen AUTHORS OR THE PROJECT CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. This software consists of voluntary contributions made by many individuals on behalf of the Jaxen Project and was originally created by bob mcwhirter <bob@werken.com> and James Strachan <jstrachan@apache.org>. For more information on the Jaxen Project, please see <http://www.jaxen.org/>.

Progress Corticon v5.3.4 incorporates JDOM v1.0 GA. Such technology is subject to the following terms and conditions: $Id: LICENSE.txt,v 1.11 2004/02/06 09:32:57 jhunter Exp $ - Copyright (C) 2000-2004 Jason Hunter and Brett McLaughlin. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions, and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the disclaimer that follows these conditions in the documentation and/or other materials provided with the distribution.

3. The name "JDOM" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact <request_AT_jdom_DOT_org>.

4. Products derived from this software may not be called "JDOM", nor may "JDOM" appear in their name, without prior written permission from the JDOM Project Management <request_AT_jdom_DOT_org>.

In addition, we request (but do not require) that you include in the end-user documentation provided with the redistribution and/or in the software itself an acknowledgement equivalent to the following: "This product includes software developed by the JDOM Project (http://www.jdom.org/)."

Alternatively, the acknowledgment may be graphical using the logos available at http://www.jdom.org/images/logos. THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE JDOM AUTHORS OR THE PROJECT CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. This software consists of voluntary contributions made by many individuals on behalf of the JDOM Project and was originally created by Jason Hunter <jhunter_AT_jdom_DOT_org> and Brett McLaughlin <brett_AT_jdom_DOT_org>. For more information on the JDOM Project, please see <http://www.jdom.org/>.

Progress Corticon v5.3.4 incorporates Saxpath v1.0. Such technology is subject to the following terms and conditions: Copyright (C) 2000-2002 werken digital. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions, and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the disclaimer that follows these conditions in the documentation and/or other materials provided with the distribution.

3. The name "SAXPath" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact license@saxpath.org.

4. Products derived from this software may not be called "SAXPath", nor may "SAXPath" appear in their name, without prior written permission from the SAXPath Project Management (pm@saxpath.org).

In addition, we request (but do not require) that you include in the end-user documentation provided with the redistribution and/or in the software itself an acknowledgement equivalent to the following: "This product includes software developed by the SAXPath Project (http://www.saxpath.org/)." Alternatively, the acknowledgment may be graphical using the logos available at http://www.saxpath.org/ THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE SAXPath AUTHORS OR THE PROJECT CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. This software consists of voluntary contributions made by many individuals on behalf of the SAXPath Project and was originally created by bob mcwhirter <bob@werken.com> and James Strachan <jstrachan@apache.org>. For more information on the SAXPath Project, please see <http://www.saxpath.org/>.