IBM® Analytical Decision Management
Version 17 Release 0

Application Designer's Guide

IBM

Product Information

Contents

Chapter 1. Designing and configuring applications

IBM® Analytical Decision Management provides a framework for building configurable, content-driven applications that automate and prioritize decisions using models and rules, and integrate with existing IT infrastructures to deploy results in real time.

The capabilities available in each application, and the basic look and feel, are determined by the application designer who configures the application template XML file and other associated files.

This guide describes the steps for configuring and customizing an application template to meet specific requirements as defined by the business user. A basic understanding of IBM Analytical Decision Management applications is assumed before configuring your own. For more information, click the **Help** link in each application, or see the application guides which can be found in the IBM SPSS® Collaboration and Deployment Services installation directory after installation (for example, *C:\Program Files\IBM\SPSS\Deployment\7.0\documentation\DecisionManagement*).

A copy of the XML schema is provided with the product for reference purposes. It can be helpful to open the main *workspace.xsd* schema in an XML editor and familiarize yourself with it. The schema is available in the IBM SPSS Collaboration and Deployment Services installation directory (for example, *C:\Program Files\IBM\SPSS\Deployment\7.0\Server\components\decision-management\Schemas\workspace.xsd*). Note that some of the schemas and elements are for internal use only, and some have been deprecated since first being introduced. Only *workspace.xsd* should be used.

The following graphic illustrates the general IBM Analytical Decision Management infrastructure, including IBM SPSS Collaboration and Deployment Services and IBM SPSS Modeler components.
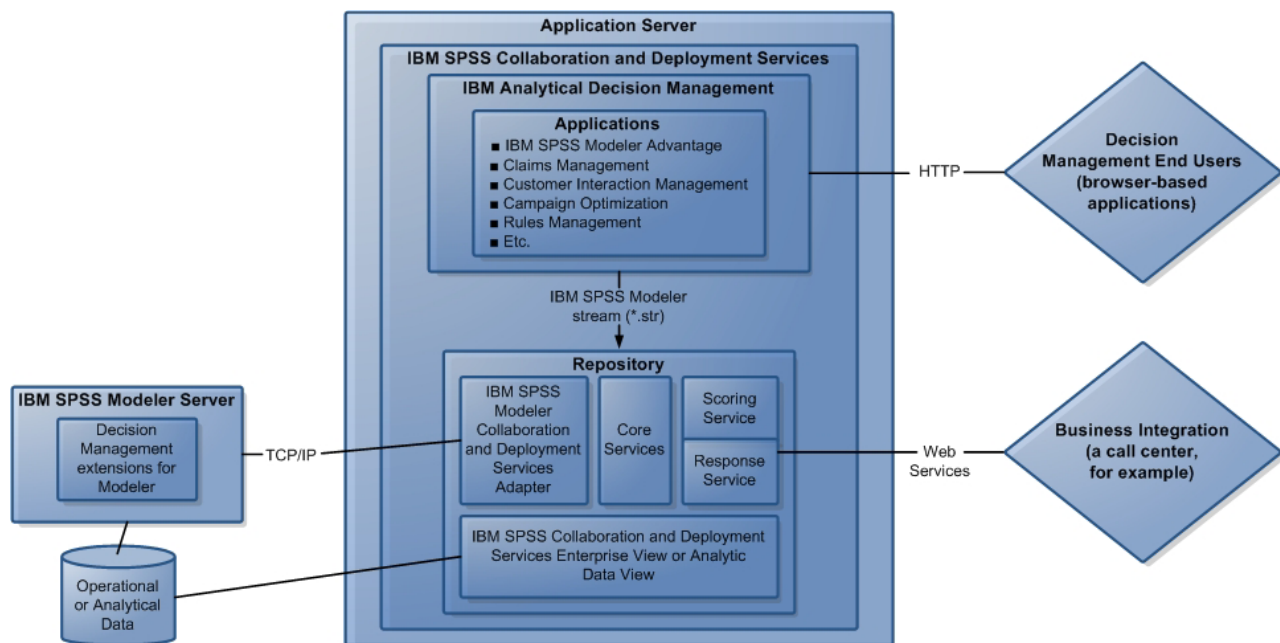
Figure 1. IBM Analytical Decision Management infrastructure

The Application Designer:

- Determines which capabilities are available to solve the business problem, such as the ability to build and score predictive models, the ability to define business rules, and the ability to combine rules and models in an end-to-end predictive application. These capabilities are configured in the application template XML.
- Defines and customizes the user interface text, style sheets, and graphics that determine the look and feel of the application.
- Specifies the dimensions that define the business problem. These dimensions provide the context in which rules and models are applied. Dimensions are configured in the application template XML. Some applications, such as IBM SPSS Rules Management or IBM SPSS Modeler Advantage, may not use dimensions.

The Application Administrator:
- Configures data and other settings for the application. For details, see the *Administering Applications* chapter of each application guide.

The Business User or Analyst:
- Builds predictive models to gain insight into your company's business problems by discovering patterns in your data.
- Defines the range of possible solutions to the business problem (such as campaigns and offers available) and specifies how records are selected and allocated for each using business rules.
- Experiments with different combinations (performs "what if" analysis) to identify the best solution.
- Deploys the application and monitors the results.

Product overview

IBM Analytical Decision Management brings the benefit of predictive analytics to real business problems, allowing you to build custom applications tailored to your customers or industry. While applications are typically configured to solve very specific problems, all are based on a common set of capabilities:

- Automate decisions using business rules.
- Add insight using predictive models.
- Use prioritization, optimization, or simulation to reach the best decision based on the above.

A number of packaged applications are available, tailored to solving specific business problems. Contact your sales representative for more information.

Available documentation

IBM Analytical Decision Management documentation is available in two formats. When you access help in the application, the online IBM Knowledge Center is launched, with the relevant help topic displayed. If you have trouble launching the help, contact your administrator in case you do not have authority to access the external Knowledge Center URL.

The Knowledge Center is an online, centralized collection of all documentation. It is always available, even when the product is not running. We recommend bookmarking it in your browser. It also contains documentation for other IBM products such as IBM SPSS Modeler, IBM SPSS Collaboration and Deployment Services, and Statistics.

At times when you are without an Internet connection, or when the Knowledge Center may be temporarily unavailable for maintenance, all documentation is also available in Portable Document Format (*.pdf*). The PDF files are available on the installation media, and are also installed into the IBM SPSS Collaboration and Deployment Services installation directory (for example, *C:\Program Files\IBM\SPSS\Deployment\7.0\Server\documentation\DecisionManagement*).

PDF documentation is also available on the Web at http://www-01.ibm.com/support/ docview.wss?uid=swg27043687, and release notes are available at http://www-01.ibm.com/support/ docview.wss?uid=swg27043686.

Chapter 2. Configuring application templates

Configuring applications

All IBM Analytical Decision Management applications are formed from a combination of seven possible basic steps, as shown in the following figure. Think of each step as a screen or tab in the user interface. Some simple applications might only include two or three steps, while more advanced applications might include all seven.

Each application is defined by an XML template that defines the capabilities and dimensions available to the business user. Each template defines a different application, as presented to business users on the *Applications launch page*. Users can then launch these applications to create their own projects. For more information, see the application guides.

XML templates

An application template includes a single `PredictiveApplication` element, which specifies the name and version of the application template used, the name of the application to use in the user interface, and other attributes. Following are some of the main child elements of `PredictiveApplication`:

- **InterfaceControl element.** Specifies the items displayed in the user interface. This element is key to any application. For example, it includes the `InterfacePages` element that controls which tabs are included in an application's user interface.
- **EntityDimension element.** Specifies the entity dimension for the application (the dimension that defines the entity of interest). Typically this is the thing being managed or allocated, such as customers, products, shipments, or claims.
- **Dimension element.** Specifies the dimensions or factors that can be used in solving the business problem, such as campaigns, channels, and offers. The dimensions define the decisions or outcomes (such as offers and actions).
- **Optimization element.** Specifies how optimization is performed, including the algorithm used and the objective function that determines what value is being optimized.
- **Deployment element.** Specifies options for deploying the application, such as the label used.

A number of additional elements are available, as defined in the public XML schema *workspace.xsd* (the non-public schemas should not be used). Many of these other elements are typically handled by the application user interface, and are not included in the application template in most cases. For more information, see the *Schema Reference Guide* provided with the product.

A copy of the XML schema is also provided with the product for reference purposes. It can be helpful to open the main *workspace.xsd* schema in an XML editor and familiarize yourself with it. The schema is available in the IBM SPSS Collaboration and Deployment Services installation directory (for example, *C:\Program Files\IBM\SPSS\Deployment\7.0\Server\components\decision-management\Schemas\ workspace.xsd*).

Note: Some schemas and elements are for internal use only, and some have been deprecated since first being introduced. Only *workspace.xsd* should be used.

Sample application template

```
<PredictiveApplication xmlns="http://com.spss.pasw.dms/workspace" templateName="ClaimsManagement"
templateVersion="1" appsVersion="17.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <InterfaceControl>
  <InterfacePages>
   <ApplicationHome stepIncluded="true" showGallery="true"/>
   <DataStep stepIncluded="true"/>
   <GlobalSelectionStep stepIncluded="true"/>
   <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="typeDecisionHierarchyDefineStep">
    <DimensionSetting name="Claim Area">
     <SelectionSection enabled="true" enableModels="true"/>
     <AggregateRuleSection enabled="true"/>
     <PredictiveModelSection enabled="true"/>
     <AllocationRuleSection enabled="false"/>
     <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
    </DimensionSetting>
   </DefineStep>
   <CombineStep stepIncluded="true" enableWhatif="true" enableTest="true">
   </CombineStep>
   <DeployScoreStep stepIncluded="true">
    <RealTimeScoring enableInteractiveQuestions="true"/>
    <ImmediateBatchScoring enableScoringOptions="true"/>
   </DeployScoreStep>
   <ReportStep stepIncluded="true"/>
  </InterfacePages>
  <InterfaceFeature id="Collaboration"/>
  <InterfaceFeature id="UploadDownload"/>
  <InterfaceFeature id="MetadataDownload"/>
 </InterfaceControl>
 <EntityDimension name="Claim"/>
 <Dimension name="Claim Area">
 </Dimension>
 <Dimension name="Action" parentDimension="Claim Area">
 </Dimension>
 <Optimization algorithm="None">
  <ObjectiveFunction/>
 </Optimization>
 <Deployment>
  <OutputAttribute referenceType="DimensionMember" name="Claim Area"
  returnValue="Claim Area.Allocation-Value">Claim Area</OutputAttribute>
  <OutputAttribute referenceType="DimensionMember" name="Action"
  returnValue="Action.Allocation-Value">Action</OutputAttribute>
 </Deployment>
</PredictiveApplication>
```

Creating an application

The process for creating your own application includes the following general steps:

- Create an application template (XML file) in the *Templates* directory.
- Create an application directory inside the *Applications* directory. Modify certain files in the directory to define the application shortcut displayed on the launchpad and to customize the look and feel of the application, if desired, by overriding the default settings.

All files are stored and edited on the machine where IBM SPSS Collaboration and Deployment Services is installed (for example, *C:\Program Files\IBM\SPSS\Deployment\7.0\components\decision-management).*

```
⊟ 📂 decision-management
   ⊟ 📂 Applications
      ⊞ 📂 CampaignOptimization
      ⊞ 📂 ClaimsManagement
      ⊞ 📂 CustomerInteractionManagement
      ⊞ 📂 DemandOptimization
      ⊞ 📂 ModelerAdvantage
      ⊞ 📂 PredictiveMaintenance
      ⊞ 📂 RulesManagement
   📂 Default
   📂 Demo
   📂 Schemas
   📂 Templates
   📂 Utilities
```
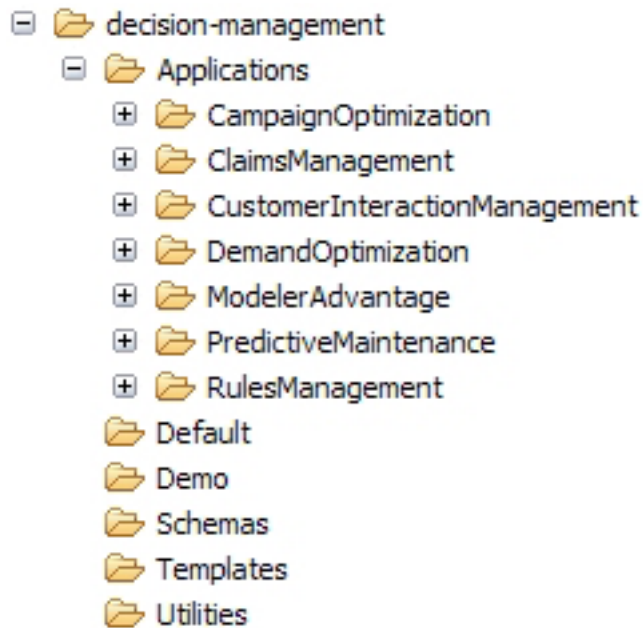
Important: Before starting, it is recommend that a backup copy of the entire *decision-management* directory be created.

To create an application

Prebuilt applications are installed with IBM Analytical Decision Management. Rather than creating an application from scratch, it's easiest to copy the existing prebuilt application that best matches your desired application and then modify it to fit your needs.

1. Copy an existing application template XML file and give it a new file name. This example uses the file name *YourApp.xml*. XML templates are stored in the *Templates* directory shown in the previous figure.

2. Open the XML template you renamed. Change the value for `templateName` to the file name you gave the new XML template in step 1 (you don't need to include the file extension):

   `templateName="YourApp"`

   Tip: Although XML files can be edited using any text editor, a number of XML editing tools are available that have options to display annotations and validate the XML markup against the *workspace.xsd* product schema.

3. Configure the rest of the XML application template to meet your needs, and save the file. See the rest of this chapter for more information.

4. Each application has a corresponding folder in the *Applications* directory. Create a new folder for your application. You may want to give it a name that matches your application template file name. In this example, the folder name is *YourApp*.

5. In the folder for an existing application that best matches your new application, copy over any files you plan to customize (for example, copy files from the *ClaimsManagement* folder into the new *YourApp* folder). You do not need to stop any servers while doing this.

   Rather than copying over only the pieces you plan to customize, you could also copy an entire folder's contents. But this is not recommended because it can make upgrading to new versions of the software more complicated.

6. Each application directory contains a file called *appGroup.xml*. The file defines the location of various resources specific to an application. Edit your application's *appGroup.xml* file. Following is an example for a custom application template (`YourApp`) that uses all the existing Claims Management stylesheets, screen text, etc.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ApplicationGroups xmlns:ns2="http://com.spss.pasw.dms/dataset" xmlns="http://com.spss.pasw.dms/appGroups">
<Group mustDisplay="true" template="YourApp">
        <CssFileSpec>/Applications/ClaimsManagement/CSS/branded.css</CssFileSpec>
        <ScreenTextFileSpec>/Applications/ClaimsManagement/ScreenText/</ScreenTextFileSpec>
        <CoachTextFileSpec>/Applications/ClaimsManagement/CoachText/</CoachTextFileSpec>
        <TermFileSpec>/Applications/ClaimsManagement/Terminology/</TermFileSpec>
        <HelpFileSpec>/Applications/ClaimsManagement/HelpLinks/HelpLinks.properties</HelpFileSpec>
</Group>
</ApplicationGroups>
```

Where:

- `mustDisplay="true"` determines whether a shortcut to the new application appears on the launch page for all users. If set to `true`, all users will see the application shortcut and will not be able to remove it. If set to `false`, users can add and remove the shortcut as they wish.

- `CssFileSpec`, `ScreenTextFileSpec`, `CoachTextFileSpec`, `TermFileSpec`, and `HelpFileSpec` can point to the location of customized user interface files, if applicable. The customized files will override the default files. These sections are only required if you customize one or more of these files. In this example, almost all files are being customized. A typical application might only customize some CSS, coach text, and screen text, in which case *appGroup.xml* would only contain those lines and the default settings would be applied for everything else. For complete details and instructions, see "File locations" on page 43 after completing the steps in this chapter.

Help specific to each prebuilt application is provided in the online Knowledge Center, along with all other IBM Analytical Decision Management documentation and all other SPSS and IBM documentation. Use the Knowledge Center's table of contents to jump to documentation for any application or product as desired.

The `HelpFileSpec` can be used to specify which section of the Knowledge Center is used in an application. You should typically have no reason to change it. To provide custom help for an application, coach text is recommended. Coach text is distinct from help, and can be fully customized. See the topic "Coach text" on page 45 for more information.

7. Each application directory contains a file called *description.xml*. The file defines the text displayed in the *Applications launch page* shortcut box.

   The file contains sections for all languages provided with the product. You can add a different language section of your choice, using a language code according to W3C standards (see the table under "Language support" on page 44 for examples). Edit your application's *description.xml* file. For example:

```
<en>
        <TitleEntry>My App for Handling Claims</TitleEntry>
        <ShortDescription>Intelligent risk management in real time</ShortDescription>
        <LongDescription>Assess the overall risk level for incoming claims and recommend
         the specific action to take.
        </LongDescription>
</en>
```

   Tip: While configuring your application(s), you may find it convenient to share the entire *decision-management* directory so you can edit the files from other machines on your network. See your operating system documentation for details about sharing directories.

8. When you finish configuring your application, enter the following URL in a supported web browser to launch IBM Analytical Decision Management and verify your work. You should see the IBM Analytical Decision Management login screen.

   http://hostname:port/DM

   Where *hostname* is the name or IP address of the machine where IBM SPSS Collaboration and Deployment Services is installed and *port* is your application server port number.

   If you have problems accessing the IBM Analytical Decision Management URL or any IBM SPSS Collaboration and Deployment Services URLs, an administrator may have customized them. Contact your administrator. More information is available in the *Application context roots* chapter of the *IBM SPSS Collaboration and Deployment Services Repository Installation and Configuration Guide*.

Configuring the user interface

The user interface for an application is defined by the `InterfaceControl` element in the XML template.

```
<InterfaceControl>
        <InterfacePages>
            <ApplicationHome stepIncluded="true" showGallery="true"/>
            <DataStep stepIncluded="true"/>
            <GlobalSelectionStep stepIncluded="true"/>
            <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="typeDecisionHierarchyDefineStep">
                <DimensionSetting name="Claim Area">
                    <SelectionSection enabled="true" enableModels="true"/>
                    <AggregateRuleSection enabled="true"/>
                    <PredictiveModelSection enabled="true"/>
                    <AllocationRuleSection enabled="false"/>
                    <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
                </DimensionSetting>
            </DefineStep>
            <CombineStep stepIncluded="true" enableWhatif="true" enableTest="true">
            </CombineStep>
            <DeployScoreStep stepIncluded="true">
                <RealTimeScoring enableInteractiveQuestions="true"/>
            </DeployScoreStep>
            <ReportStep stepIncluded="true"/>
        </InterfacePages>
        <InterfaceFeature id="Collaboration"/>
        <InterfaceFeature id="UploadDownload"/>
        <InterfaceFeature id="MetadataDownload"/>
</InterfaceControl>
```

The first-level children of `InterfacePages` represent the main tabs in the user interface. You can set them to `true` or `false` to control which tabs make up your application. See the topic "XML templates" on page 5 for more information.

The `InterfaceFeature` element defines capabilities that are available to the application, such as building and reusing models and rules, the ability to access the IBM SPSS Collaboration and Deployment Services Repository, and the ability to upload and download files.
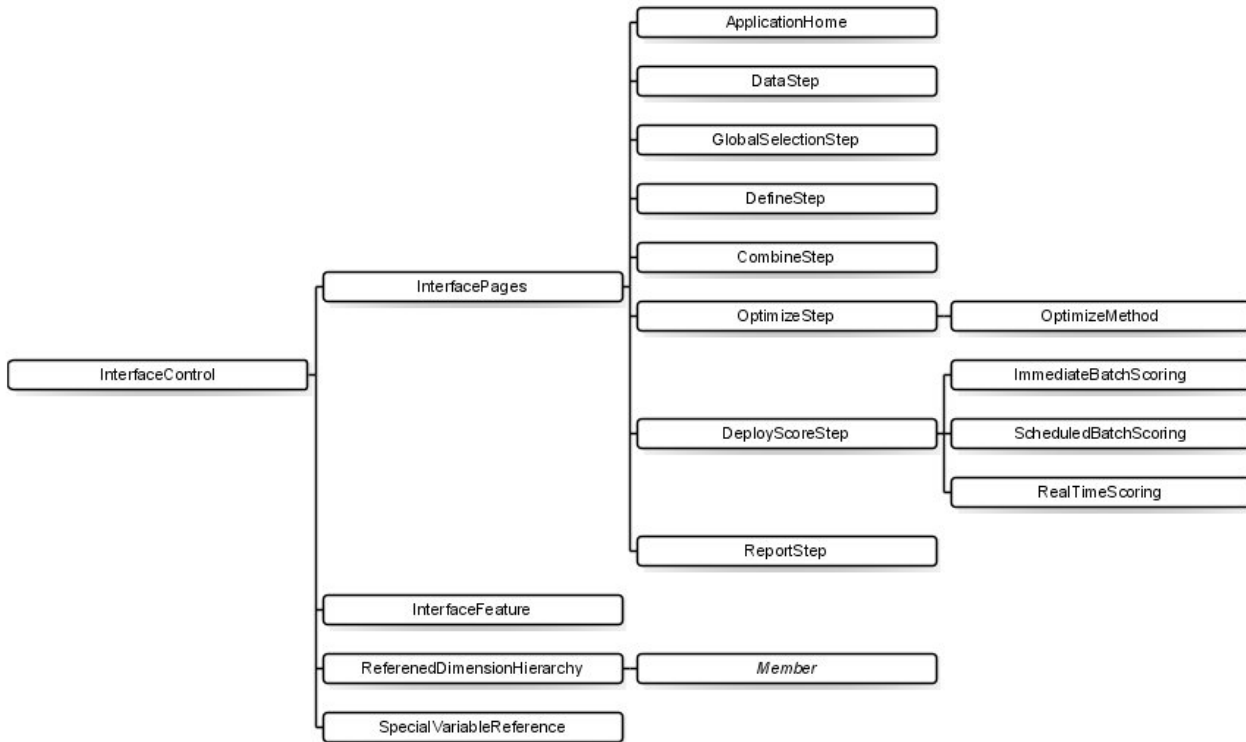
Figure 2. InterfaceControl element

When configuring applications, keep the following guidelines in mind:

- You can only have one of each tab type in an application.
- You can only have one of each "widget" within an application (for example, if your application has a Score tab it can only include one `RealTimeScoring` section).
- If more than one allocation method is enabled on the Define tab, the Combine/Prioritize/Optimize step can be used to determine how final decisions or recommendations are determined. For example if both `AggregateRuleSection` and `PredictiveModelSection` are enabled, a combine matrix can be used to reconcile cases where rules and models may return different decisions. If only one allocation method is enabled on the Define tab, the Combine/Prioritize/Optimize step is not needed.

The following table describes some of the general items that can be configured on each tab of an application.

Table 1. General configuration items.

Tab/Screen	Main configurable items
Home	• Whether screen is included in the application • Whether Gallery is included
Data	• Whether tab is included in the application • Whether user can derive new attributes • Whether the **Add/Edit additional fields and tables** section is included in the application. See the *User's Guide* for more information about this functionality.
Global Selections	• Whether tab is included in the application • Whether models are on or off (if on, can also indicate whether global selections will apply to model build operations or not)

Table 1. General configuration items (continued).

Tab/Screen	Main configurable items
Define/Modeling/Rules	- Whether tab is included in the application - Whether tab includes a decision hierarchy tree, model-only (for modeling applications), or rules-only (for rules management applications) - Whether interaction points are included on the tab - Other options are also available depending which tab-type you use. Note that `AggregationRuleSection` and `PredictiveModelSection` may not both be enabled when using more than two dimensions.
Combine/Prioritize	- Whether tab is included in the application - Whether prioritization or a combine (decision) matrix is used Note that the `MatrixCombine` method may not be used with more than two dimensions.
Optimize	- Whether tab is included in the application - Which optimization method is used
Score/Deploy	- Whether tab is included in the application - Whether immediate batch scoring (including scoring options), real-time deployment (including ability to specify interactive questions), or both, is used
Reporting	- Whether tab is included in the application - Whether results summary report is included on the Home screen

Defining dimensions

Application dimensions define the factors or outcomes that can be used in solving the business problem, such as campaigns, channels, and offers. These are configured using the `EntityDimension` and `Dimension` elements in the XML template.

```
<EntityDimension name="Claim"/>
<Dimension name="Claim Area"></Dimension>
<Dimension name="Action" parentDimension="Claim Area"></Dimension>
```

- The `EntityDimension` element defines the thing being processed by the application, such as customers, claims, or packages. In practical terms, this is what each row in the Project Data Model represents.
- The `Dimension` elements define the factors or outcomes that can be used in solving the business problem, such as campaigns, channels, and offers. In practical terms, these are the possible recommendations or decisions that can be returned for each entity, and are displayed in the dimension tree on the Define tab in the application user interface.
- Dimensions can be nested using the `parentDimension` attribute.
- If no dimensions are specified, there will not be a dimension tree on the Define tab, and the application will only be able to produce rules or models. No specific outputs or recommendations will be returned, as none have been defined.

Note that the **DefineStep** section of the XML template contains a `DimensionSetting` element as seen in the example below. This element controls which user interface sections are included on the Define tab of the application.

```
<DimensionSetting name="Claim Area">
    <SelectionSection enabled="true" enableModels="true"/>
    <AggregateRuleSection enabled="true"/>
    <PredictiveModelSection enabled="true"/>
    <AllocationRuleSection enabled="false"/>
    <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
</DimensionSetting>
```

Configuring the optimization/prioritization equation

The optimization or prioritization equation is defined in the `Optimization` element, and is displayed on the Optimize or Prioritize tab in applications that use it. For applications that don't use optimization or prioritization, such as IBM Analytical Decision Management for Claims, `Optimization algorithm` is set to `None`.

The Heuristic and CPLEX algorithms are supported. An application will have a *Prioritize* tab if it's configured to use the Heuristic algorithm, an *Optimize* tab if it's configured to use the CPLEX algorithm, and a *Combine* tab if it's configured to have multiple allocation results which can be combined to provide a single outcome using a matrix.

Example: IBM Analytical Decision Management for Customer Interactions

```
<Optimization algorithm="Heuristic">
  <ObjectiveFunction Domain="double" Functor="-" Name="Expected Profit" description="Expected_profit_using_probability_to_respond">
    <Expression xmlns="http://com.spss.pasw.dms/rules" Domain="double" Functor="*">
     <Expression Domain="double" Functor="variableReference">
      <Expression>
       <Value>value</Value>
      </Expression>
      <Expression>
       <Value>Variable</Value>
      </Expression>
      <Expression>
       <Value>Prob.to Respond</Value>
      </Expression>
      <Expression>
       <Value>Value</Value>
      </Expression>
     </Expression>
     <Expression Domain="double" Functor="variableReference">
      <Expression>
       <Value>value</Value>
      </Expression>
      <Expression>
       <Value>Variable</Value>
      </Expression>
      <Expression>
       <Value>Revenue</Value>
      </Expression>
      <Expression>
       <Value>Value</Value>
      </Expression>
     </Expression>
    </Expression>
    <Expression xmlns="http://com.spss.pasw.dms/rules" Domain="double" Functor="variableReference">
     <Expression>
      <Value>value</Value>
     </Expression>
     <Expression>
      <Value>Variable</Value>
     </Expression>
     <Expression>
      <Value>Cost</Value>
     </Expression>
     <Expression>
      <Value>Value</Value>
     </Expression>
    </Expression>
  </ObjectiveFunction>
</Optimization>
```

The resulting equation is displayed on the Prioritize tab in the IBM Analytical Decision Management for Customer Interactions user interface.

Example: IBM Analytical Decision Management for Campaign Optimization

```
<Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path="/Applications/CampaignOptimization/Optimization/
CampaignOptimization.mod">
  <ObjectiveFunction Domain="double" Name="Expected Profit" description="Expected_profit_using_probability_to_respond">
```

```
  <ExpressionFormat format="( ${Prob.to Respond} * ${Revenue} ) - ( ${OfferCost} + ${ChannelCost} )"/>
  <ExternalUsage controlVariable="use_expected_profit_function" variableType="int" enabledValue="1" disabledValue="0"/>
 </ObjectiveFunction>
 <ObjectiveFunction Domain="double" Name="ROI" description="Return_On_Investment" functionType="nonLinear">
  <ExpressionFormat format="${FN_sum} ( ( ${Prob.to Respond} * ${Revenue} ) - (${OfferCost} + ${ChannelCost}) ) / ${FN_sum}
  ( (${OfferCost} + ${ChannelCost}) )"/>
  <ExternalUsage controlVariable="use_roi_function" variableType="int" enabledValue="1" disabledValue="0"/>
 </ObjectiveFunction>
```

In this CPLEX algorithm example, a path is defined pointing to an optimization model file (*CampaignOptimization.mod*) that ships with IBM Analytical Decision Management. Do not modify this file. If you need to modify the file, contact an IBM SPSS representative.

The equations are displayed on the Optimize tab in the IBM Analytical Decision Management for Campaign Optimization user interface.

The `ExpressionFormat` element is for display purposes in the user interface. The actual implementation of the objective function is defined in IBM CPLEX Optimization Studio.

Example: IBM Analytical Decision Management for Demand Optimization

```
<Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path="/Applications/DemandOptimization/Optimization/
DemandOptimization.mod">
  <ObjectiveFunction Domain="double" Name="ExpectedRevenue" description="ExpectedRevenueDescription">
   <ExpressionFormat format="( ${UnitRevenue} * ${StockRequested} )"/>
   <ExternalUsage controlVariable="use_expected_revenue_function" variableType="int" enabledValue="1" disabledValue="0"/>
  </ObjectiveFunction>

  <ObjectiveFunction Domain="double" Name="ExpectedRevenueWithPreferred" description="ExpectedRevenueWithPreferredDescription">
   <ExpressionFormat format="( ${UnitRevenue} * ${StockRequested} * ${RequestPriority} )"/>
   <ExternalUsage controlVariable="use_prioritize_preferred_function" variableType="int" enabledValue="1" disabledValue="0"/>
  </ObjectiveFunction>
```

In this CPLEX algorithm example, a path is defined pointing to an optimization model file (*DemandOptimization.mod*) that ships with IBM Analytical Decision Management for Demand Optimization. Do not modify this file. If you need to modify the file, contact an IBM SPSS representative.

The equations are displayed on the Optimize tab in the IBM Analytical Decision Management for Demand Optimization user interface.

Example: IBM Analytical Decision Management for Operations

```
<Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path="/Applications/PredictiveMaintenance/Optimization/
PredictiveMaintenance.mod">
  <ObjectiveFunction Domain="double" Name="ExpectedSavings" description="ExpectedSavingsFunction">
   <ExpressionFormat format="( ${ProbOfFailure} * ${Revenue} ) - ( ${TimeInHours} * ${CostPerHour} )"/>
  </ObjectiveFunction>
```

In this CPLEX algorithm example, a path is defined pointing to an optimization model file (*PredictiveMaintenance.mod*) that ships with IBM Analytical Decision Management for Operations. Do not modify this file. If you need to modify the file, contact an IBM SPSS representative.

The resulting equation is displayed on the Optimize tab in the IBM Analytical Decision Management for Operations user interface.

Understanding prefix notation

Expressions in IBM Analytical Decision Management are defined in *Prefix* notation (also known as *Polish* notation). *Infix* and *Prefix* notations are two different but equivalent ways of writing expressions. The *Infix* column displays formulas people are used to, and the *Prefix* column displays the equivalent notation that would be used in IBM Analytical Decision Management expressions. All parentheses are implied in the *Prefix* column; they've been included to make the table easier to read by showing the order of evaluation.

Table 2. Infix vs. Prefix expression notation.

Infix notation	Prefix (Polish) notation
((A * B) + (C / D))	(+ (* A B) (/ C D))
((A * (B + C)) / D)	(/ (* A (+ B C)) D)
(A * (B + (C / D)))	(* A (+ B (/ C D)))

An example would be coded as follows:

```
"-"
"*"
ProbabilityToRespond
Revenue
Cost
```

The actual expression starts with the `ObjectiveFunction` element that specifies the storage data type of the result, its name, and the initial `Functor`--which is our *Prefix* notation for subtraction.

```
<ObjectiveFunction Domain="double" Functor="-" Name="Predicted Profit">
```

Unless otherwise specified, the subtraction functor will be applied to all expressions with this objective function.

The next operation specifies multiplication as the functor, overriding the default. Within this expression, two variable references are defined, inserting the values of the `Prob.to Respond` and `Revenue` variables and multiplying them together.

```
<Expression xmlns="http://com.spss.pasw.dms/rules" Domain="double" Functor="*">
      <Expression Domain="double" Functor="variableReference">
         <Expression>
            <Value>value</Value>
         </Expression>
         <Expression>
            <Value>Variable</Value>
         </Expression>
         <Expression>
            <Value>Prob.to Respond</Value>
         </Expression>
         <Expression>
            <Value>Value</Value>
         </Expression>
      </Expression>
      <Expression Domain="double" Functor="variableReference">
         <Expression>
            <Value>value</Value>
         </Expression>
         <Expression>
            <Value>Variable</Value>
         </Expression>
         <Expression>
            <Value>Revenue</Value>
         </Expression>
         <Expression>
            <Value>Value</Value>
         </Expression>
      </Expression>
</Expression>
```

In other words, the example expression is referencing a simple `value` of a `Variable` named `ProbabilityToRespond` in order to use its output field named `Value` in this spot of the equation.

A third variable reference inserts the value of the Cost variable. Because it is a child of the top-level ObjectiveFunction element (and not the multiplication expression) its value is subtracted.

```
<Expression xmlns="http://com.spss.pasw.dms/rules" Domain="double" Functor="variableReference">
      <Expression>
         <Value>value</Value>
      </Expression>
      <Expression>
         <Value>Variable</Value>
      </Expression>
```

```
    <Expression>
        <Value>Cost</Value>
    </Expression>
    <Expression>
        <Value>Value</Value>
    </Expression>
</Expression>
```

Defining equations using the expression builder

You can use the Expression Builder as a shortcut to building expressions. While the prefix notation takes a bit of time to learn, you can build expressions using the Expression Builder, and then copy the resulting code.

For example, suppose you want to define the following equation:

```
profit = quantity * (price - unit cost - promotion cost)
```

1. Create a project using any valid application template. (This can be a custom template, or a sample application such as IBM Analytical Decision Management for Customer Interactions or IBM Analytical Decision Management for Claims.)
2. On the Data tab, choose a data source that includes the fields you want to use in your equation, and choose this for the Project Data Model. (For this example you would need fields named *quantity*, *price*, *unit cost*, and *promotion cost*. You don't need any real data--you just need these fields to exist in the data model so you can choose them in the Expression Builder.)
3. From the Data tab, click **Add/Edit additional fields and tables** and then click **Add an expression**.
4. Specify the desired expression.
5. Save the project and download to a local drive, for example *profitexpression.str*.
6. Change the filename extension to **.zip* (for example, *profitexpression.zip*) and extract it. (Stream files are compressed **.zip* files and can be extracted to access the component parts, which include a primary file named *ClementineStream.xml* along with a number of **.dat* files.)
7. Open the extracted file *\data\0001.dat* in an XML or text editor, and search for an expression element such as the following:

```
<Expression Functor="*" Domain="double">
   <Expression Domain="long"><Attribute>quantity</Attribute></Expression>
   <Expression Functor="-" Domain="double">
      <Expression Functor="-" Domain="double">
        <Expression Domain="long"><Attribute>price</Attribute></Expression>
        <Expression Domain="double"><Attribute>unit cost</Attribute></Expression>
      </Expression>
      <Expression Domain="double"><Attribute>promotion cost</Attribute></Expression>
   </Expression>
</Expression>
```

8. Copy the expression code into your XML template, replace the attributes with variable references, and add variable definitions and output attributes as needed (see "Defining variables").

Defining variables

Define any variables that will be used as inputs to the optimization. Any variable referenced in the optimization function must be defined in a `Variable` element which can be a child of either `EntityDimension` or `Dimension`. In practical terms, the location of the variable definition determines the level at which the input is specified (though the user can change this in the application user interface).

For example, suppose you are prioritizing campaigns and offers based on expected profit, computed as follows:

```
expected profit = probability to respond * revenue - cost
```

To do this, the required inputs (`probability`, `revenue`, and `cost`) must be specified for each campaign or offer, as appropriate. In the user interface, you specify these inputs on the Prioritize or Optimize tab.

If you want the same value to be used for all offers within a campaign, you can specify that input at the campaign level. To do this, click **Customize table** on the Prioritize or Optimize tab. (Offer is the child of campaign, so specifying at the campaign level applies the value to all offers within that campaign.) Alternatively, you can specify different inputs for each offer.

In the XML template, the same inputs are defined as variables on the relevant dimension.

```
<Dimension name="Offer">
    <Variable name="Prob.to Respond" dataType="double" simulateName="Offers Accepted"
    simulateAction="sum"
    optimizationInputItem="true" prompt="">
        <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
        "ConstantValueSource">
            <Value>0</Value>
        </ValueSource>
    </Variable>
    <Variable name="Revenue" dataType="double" optimizationInputItem="true" prompt="">
        <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
        "ConstantValueSource">
            <Value>0</Value>
        </ValueSource>
    </Variable>
    <Variable name="Cost" dataType="double" simulateAction="sum" simulateName="Total
    Cost" optimizationInputItem="true" prompt="">
        <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
        <"ConstantValueSource">
            <Value>0</Value>
        </ValueSource>
    </Variable>
</Dimension>
```

Alternatively, if want to specify one or more inputs at the Campaign level, you can add these as variables on the Campaign dimension instead. Moving the variable definition from Offer to Campaign is the same as selecting **Campaign** in the application user interface.

```
<Dimension name="Campaign">
    <Variable name="Prob.to Respond" dataType="double" simulateName="Offers Accepted"
    simulateAction="sum"
    optimizationInputItem="true" prompt="">
        <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
        "ConstantValueSource">
            <Value>0</Value>
        </ValueSource>
    </Variable>
</Dimension>
```

Also note you can add the optional `simulateAction` as seen in the previous examples to specify that the variable should be included in simulation results. You can give it a value of `sum` (`simulateAction="sum"`) or `average` (`simulateAction="average"`), which determines how the summary value is computed for the variable. You can also specify the `simulateName` as seen in the examples above to specify the label to use for the summary value in the simulation results.

Defining constraints

Variables can also be referenced in constraints. The same general rule applies--you have to define the variable before you can use it. But in this case the variable definition and constraint may both be children of the dimension to which they apply.

Following is an example of the predefined constraints in the IBM Analytical Decision Management for Campaign Optimization application template. End users in the application do not create constraints themselves. On the Optimize tab of the user interface, they can choose to disable constraints they don't want to use. The constraints are defined in IBM CPLEX Optimization Studio.

```
<!--Constraint type="min" name="ctMinProfitConstraint">
    <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value="Min.Profit"/>
    <ExpressionFormat format="${ObjectiveValue}"/>
    <ExternalUsage controlVariable="use_min_profit_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
```

```
      </Constraint-->
  <Constraint type="max" name="ctCampaignBudgetConstraint" entityScoped="false" enabled="false" description=
  "CampaignBudgetConstraint">
   <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value="CampaignBudget"/>
   <ExpressionFormat format="${FN_sum} ( ${OfferCost} + ${ChannelCost} )"/>
   <ExternalUsage controlVariable="use_campaign_budget_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
  </Constraint>
  <Constraint type="min" name="ctMinCampaignSizeConstraint" entityScoped="false" enabled="false" description="MinCampaignOffers">
   <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
   "MinCampaignOfferCount"/>
   <ExpressionFormat format="${FN_count} ( ${Offer} )"/>
   <ExternalUsage controlVariable="use_min_campaign_offers_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
  </Constraint>
  <!--Constraint type="max" name="ctCampaignSizeConstraint" entityScoped="false" description="MaxOffersPerCampaign">
   <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value="CampaignSize"/>
   <ExpressionFormat format="${FN_count} ( ${Offer} )"/>
   <ExternalUsage controlVariable="use_campaign_size_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
  </Constraint-->
  <Constraint type="max" name="ctOffersAvailableConstraint" entityScoped="false" description="MaxOffersAvailable">
   <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value="OffersAvailable"/>
   <ExpressionFormat format="${FN_count} ( ${Offer} )"/>
   <ExternalUsage controlVariable="use_offers_available_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
  </Constraint>
  <Constraint type="max" name="ctTotalBudgetConstraint" entityScoped="false" description="TotalBudgetForAllCampaigns">
   <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value="TotalBudget"/>
   <ExpressionFormat format="${FN_sum} ( ${OfferCost} + ${ChannelCost} )"/>
   <ExternalUsage controlVariable="use_total_budget_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
  </Constraint>
  <Constraint type="max" name="ctMaxOffersConstraint" description="MaxOffersPerCustomer">
   <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value="MaxOffersNum"/>
   <ExpressionFormat format="${FN_count} ( ${Offer} / ${Customer} ) + ( ${RecentOffersNum} )"/>
   <ExternalUsage controlVariable="use_max_offers_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
  </Constraint>
  <Constraint type="max" name="ctChannelCapacityConstraint" entityScoped="false" description="MaxOffersPerChannel">
   <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value="Capacity"/>
   <ExpressionFormat format="${FN_count} ( ${Channel} )"/>
   <ExternalUsage controlVariable="use_channel_capacity_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
  </Constraint>
```

Configuring scoring output for deployment

For applications that will be deployed for batch or real-time scoring, the Deployment element in the XML template specifies the default stream outputs. These can include:

- Default options for the Deploy tab (for both Real Time Scoring and Batch Scoring). However, end users can change them on the Deploy tab. See the topic "IBM Analytical Decision Management and the Scoring Service" on page 55 for more information.
- The final allocations or decisions for each dimension, whether determined via rules, prioritization, or combine matrix.
- Outputs from the models and rules within the stream, which typically serve as inputs to those decisions.
- Any prioritization variables defined.
- Input fields, Annotations, and "Return with" fields specified for each dimension.

Each output is defined using a separate OutputAttribute element within the Deployment element of the XML template. Outputs defined in this manner can be selected for inclusion in the result set when creating the scoring configuration.

```
<Deployment>
      <OutputAttribute referenceType="DimensionMember" name="Claim Area"
         returnValue="Claim Area.Allocation-Value">Claim Area</OutputAttribute>
      <OutputAttribute referenceType="DimensionMember" name="Action"
         returnValue="Action.Allocation-Value">Action</OutputAttribute>
</Deployment>
```

Figure 3. Sample OutputAttribute specification

Returning the allocations for each dimension

The allocations for each dimension typically represent the "final answer" returned by the application, such as the offer to be presented to a customer, or the action to take on a claim. A separate allocation field is returned for each dimension. These fields can be configured as default outputs in the `Deployment` element as follows:

```
<Deployment>
     <OutputAttribute referenceType="DimensionMember" name="Claim Area"
         returnValue="Claim Area.Allocation-Value">Claim Area</OutputAttribute>
     <OutputAttribute referenceType="DimensionMember" name="Action"
         returnValue="Action.Allocation-Value">Action</OutputAttribute>
</Deployment>
```

Where:

- `referenceType` is `DimensionMember`.
- `name` is the name of the dimension being allocated.
- `returnValue` is the name of the dimension followed by `.Allocation-Value`.

Outputs from models and rules

Each model or rule used in an allocation returns one or more output fields. When combined with the final allocation for each dimension, these outputs may be useful in understanding how a particular result was determined. For example, if a claim is referred for investigation based on the total number of risk points, you might want to know which rules contributed to the total. The available fields depend on the dimensions being allocated, and the types of rules and models used, as detailed below.

Allocation using segment rules

When used in an allocation (`<AllocationRuleSection enabled="true" />`), segment rules return the following fields:

- `<<Dimension>>.Allocation-Segment`. The list of index values for all segments returning a value of *true*.
- `<<Dimension>>.Allocation-Segment Name`. The list of names for all segments returning a value of *true*, in the same order as the `Allocation-Segment` field.

These fields can be configured for scoring in the `Deployment` element, as follows:

```
<Deployment>
     <OutputAttribute referenceType="DimensionMember" name="Offer"
         returnValue="Offer.Allocation-Segment">Segment</OutputAttribute>
     <OutputAttribute referenceType="DimensionMember" name="Offer"
         returnValue="Offer.Allocation-Segment Name">Segment Name</OutputAttribute>
</Deployment>
```

Where:

- `referenceType` is `DimensionMember`.
- `name` matches the name of the dimension being allocated.
- `returnValue` matches the name of the field being returned.

Allocation using random percentages

When used in an allocation, random percentage rules (`<AllocationRuleSection enabled="true" />`) return the following field:

- `<<Dimension>>.Allocation-Segment Name`. The list of names for all segments returning a value of *true*.

This field can be configured for scoring as follows:

```
<Deployment>
     <OutputAttribute referenceType="DimensionMember" name="Offer"
          returnValue="Offer.Allocation-Segment Name">Offer Segment Name</OutputAttribute>
</Deployment>
```

Allocation using aggregated point totals

Rules that allocate decisions based on aggregated point totals (`<AggregateRuleSection enabled="true"/>`) return the following fields:

- `<<Dimension>>.Allocation.Rule-Value`. The final allocation returned by the rule.
- `<<Dimension>>.Allocation.Rule.Aggregate-Value`. The aggregated point total across all segments that returned a value of *true*. This is the sum of the values listed for `Aggregate-Segment Points`.
- `<<Dimension>>.Allocation.Rule.Aggregate-Segment`. The list of index values for all segments returning a value of *true*.
- `<<Dimension>>.Allocation.Rule.Aggregate-Segment Name`. The names of all segments returning *true*, in the same order as the `Aggregate-Segment` field.
- `<<Dimension>>.Allocation.Rule.Aggregate-Segment Points`. The "points" list for all segments returning *true*, in the same order as the `Aggregate-Segment` field.
- `<<Dimension>>.Allocation.Rule-Threshold`. The index of the segment to which the record was allocated based on the sum of points.
- `<<Dimension>>.Allocation.Rule-Threshold Segment`. The minimum number of points needed to be included in that segment.
- `<<Dimension>>.Allocation.Rule-Threshold Test Value`. The aggregate value tested. This should match the total points returned for `Aggregate-Value`.

These fields can be configured for scoring as follows:

```
<Deployment>
     <OutputAttribute referenceType="DimensionMember" name="Action"
          returnValue="Action.Allocation.Rule-Value">Rule Action</OutputAttribute>
     <OutputAttribute referenceType="DimensionMember" name="Action"
          returnValue="Action.Allocation.Rule-Aggregate-Value">Total Risk Points</OutputAttribute>
</Deployment>
```

Allocation based on model scores

When used in an allocation, predictive models (`<PredictiveModelSection enabled="true"/>`) return the following fields:

- `<<Dimension>>.Allocation.Model-Value`. The probability, propensity, or confidence value returned by the model.
- `<<Dimension>>.Allocation.Model-Threshold`. The index of the segment to which the record was allocated based on model value.
- `<<Dimension>>.Allocation.Model-Threshold Segment`. The minimum value needed to be included in that segment.
- `<<Dimension>>.Allocation.Model-Threshold Test Value`. The model value tested. This should match the value returned for `Model-Value`.

These fields can be configured for scoring as follows:

```
<Deployment>
     <OutputAttribute referenceType="DimensionMember" name="Action"
          returnValue="Action.Allocation.Model-Value">Model Action</OutputAttribute>
     <OutputAttribute referenceType="DimensionMember" name="Action"
          returnValue="Action.Allocation.Model-Threshold">Model Threshold</OutputAttribute>
</Deployment>
```

Outputs from prioritization

Outputs from prioritization include the result of the prioritization equation, such as profit, as well as any prioritization variables.

Prioritization equation

The result of the prioritization equation is output to a field named `<<objective function>>-Value`. This field can be configured for scoring as follows:

```
<Deployment>
    <OutputAttribute referenceType="Objective" name="Expected Profit"
        returnValue="Expected Profit-Value">Expected Profit</OutputAttribute>
</Deployment>
```

Where:

- `referenceType` is `Objective`.
- `name` matches the value of the `Name` attribute specified for the `ObjectiveFunction` element.
- `returnValue` matches the name of the field being returned. By convention this is the name of the `ObjectiveFunction` element, with `-Value` appended.

Prioritization variables

Variables can be configured as default outputs as follows:

```
<Deployment>
    <OutputAttribute referenceType="Variable" name="MaxOffersNum"
        returnValue="MaxOffersNum.Variable-Value">Max Offer</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="Min.Profit"
        returnValue="Min.Profit.Variable-Value">Min Profit</OutputAttribute>
</Deployment>
```

Where:

- `referenceType` is `Variable`.
- `name` matches the name of the variable.
- `returnValue` is the name of the variable with `.Variable-Value` appended.

Input fields, annotations, and "Return with" fields

Default outputs for these items are automatically configured, and do not need to be defined in the XML template (no `OutputAttribute` specification is required to use these fields).

Any "Return with" fields are automatically configured as default outputs, and do not need to be specified in the XML template. The **<entity> Key** drop-down in the user interface shows the default outputs, and you can use the Deploy tab to add others, if desired.

Example: IBM Analytical Decision Management for Customer Interactions output configuration

When scoring IBM Analytical Decision Management for Customer Interactions, available output fields include the campaigns and offers returned for each customer, along with allocation and prioritization outputs that may be useful in determining why a particular recommendation was made.

Outputs for this application can be configured in the `Deployment` element as follows:

```
<Deployment>
    <OutputAttribute referenceType="DimensionMember" name="Campaign"
      returnValue="Campaign.Allocation-Value">Campaign</OutputAttribute>

    <OutputAttribute referenceType="DimensionMember" name="Offer"
      returnValue="Offer.Allocation-Value">Offer</OutputAttribute>
```

```
      <OutputAttribute referenceType="DimensionMember" name="Offer"
        returnValue="Offer.Allocation-Segment">Segment</OutputAttribute>
      <OutputAttribute referenceType="DimensionMember" name="Offer"
        returnValue="Offer.Allocation-Segment Name">Segment Name</OutputAttribute>
      <OutputAttribute referenceType="DimensionMember" name="Offer"
        returnValue="Offer.Allocation-Annotation">Annotation</OutputAttribute>
      <OutputAttribute referenceType="DimensionMember" name="Offer"
        returnValue="Offer.$ReturnWithVariable">Age Youngest Child</OutputAttribute>

      <OutputAttribute referenceType="Objective" name="Expected Profit"
        returnValue="Expected Profit-Value">Expected Profit</OutputAttribute>
      <OutputAttribute referenceType="Variable" name="MaxOffersNum"
        returnValue="MaxOffersNum.Variable-Value">Max Offer</OutputAttribute>
      <OutputAttribute referenceType="Variable" name="Min.Profit"
        returnValue="Min.Profit.Variable-Value">Min Profit</OutputAttribute>
      <OutputAttribute referenceType="Variable" name="Prob.to Respond"
        returnValue="Prob.to Respond.Variable-Value">Prob to Respond</OutputAttribute>
      <OutputAttribute referenceType="Variable" name="Revenue"
        returnValue="Revenue.Variable-Value">Revenue</OutputAttribute>
      <OutputAttribute referenceType="Variable" name="Cost"
        returnValue="Cost.Variable-Value">Cost</OutputAttribute>
</Deployment>
```

This configuration might return the following output fields:

Table 3. Example returned output fields.

Field	Customer 1	Customer 2
Campaign	Cross Sell	Cross Sell
Offer	Personal Loan	Home Equity Loan
Segment	2	1
Segment Name	Low Debt Ratio	Homeowner
Annotation	Congratulations, you have qualified for a low interest loan	Congratulations, you have qualified for a home equity loan
Age Youngest Child		
Expected Profit	200	400
Max Offer	2	2
Min Proft	10	10
Prob to Respond	0.19	0.09
Revenue	200	400
Cost	3	3

Reviewing this output, the following can be determined:

- Customer 1 received the *Personal Loan* offer; customer 2 received the *Home Equity Loan* offer. These outcomes were determined by the *Low Debt Ratio* and *Homeowner* rules, which were the second and first segments in the allocation set, respectively.
- The annotations are returned with each offer, and are specific to that offer.
- The *Expected Profit* field lists the value returned by the prioritization equation, and is used to determine whether the offer is made. The other fields include the variables and constraints used in this calculation.

Example: IBM Analytical Decision Management for Campaign Optimization output configuration

Like IBM Analytical Decision Management for Customer Interactions, when scoring IBM Analytical Decision Management for Campaign Optimization, available output fields include the campaigns and

offers returned for each customer, along with allocation and optimization outputs that may be useful in determining why a particular recommendation was made.

Outputs for this application can be configured in the `Deployment` element as follows:

```
<Deployment>
    <OutputAttribute referenceType="DimensionMember" name="Campaign"
      returnValue="Campaign.Allocation-Value">Campaign</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Offer"
      returnValue="Offer.Allocation-Value">Offer</OutputAttribute>
    <OutputAttribute referenceType="Objective" name="ObjectiveValue"
      returnValue="ObjectiveValue-Value">Output-ObjectiveValue</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="MaxOffersNum"
      returnValue="MaxOffersNum.Variable-Value">Output-MaxOffersNum</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="RecentOffersNum"
      returnValue="RecentOffersNum.Variable-Value">Output-RecentOffersNum</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="TotalBudget"
      returnValue="TotalBudget.Variable-Value">Output-TotalBudget</OutputAttribute>
    <!-- OutputAttribute referenceType="Variable" name="CampaignSize"
      returnValue="CampaignSize.Variable-Value">Output-CampaignSize</OutputAttribute-->
    <!-- OutputAttribute referenceType="Variable" name="Min.Profit"
      returnValue="Min.Profit.Variable-Value">Output-MinProfit</OutputAttribute-->
    <OutputAttribute referenceType="Variable" name="Prob.to Respond"
      returnValue="Prob.to Respond.Variable-Value">Output-ProbToRespond</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="Revenue"
      returnValue="Revenue.Variable-Value">Output-Revenue</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="OfferCost"
      returnValue="OfferCost.Variable-Value">Output-OfferCost</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="ChannelCost"
      returnValue="ChannelCost.Variable-Value">Output-ChannelCost</OutputAttribute>
</Deployment>
```

Example: IBM Analytical Decision Management for Claims output configuration

When scoring IBM Analytical Decision Management for Claims, available output fields include the final claim area and action returned for each claim, along with rule and model outputs that may be useful in determining why a particular recommendation was made.

Outputs for this application can be configured in the `Deployment` element as follows:

```
<Deployment>
    <OutputAttribute referenceType="DimensionMember" name="Claim Area"
      returnValue="Claim Area.Allocation-Value">Claim Area</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation-Value">Action</OutputAttribute>

  <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Rule-Value">Rule Action</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Rule.Aggregate-Value">Aggregate Value</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Rule.Aggregate-Segment">Rule Segment</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Rule.Aggregate-Segment Name">Rule Segment Name</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Rule.Aggregate-Segment Points">Rule Segment Points</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Rule-Threshold">Rule Threshold</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Rule-Threshold Segment">Rule Threshold Segment</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Rule-Threshold Test Value">Rule Threshold Test Value</OutputAttribute>

    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Model-Value">Model Action</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Model-Threshold">Model Threshold</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
```

```
        returnValue="Action.Allocation.Model-Threshold Segment">Model Threshold Segment</OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Action"
      returnValue="Action.Allocation.Model-Threshold Test Value">Model Threshold Test Value</OutputAttribute>
</Deployment>
```

This configuration might return the following output fields:

Table 4. Example returned output fields.

Field	Claim 1	Claim 2
Claim Area	Automotive	Automotive
Action	Standard	Standard
Rule Action	Standard	Standard
Aggregate Value	3.0	3.0
Rule Segment	2	1|4|5
Rule Segment Name	Multiple Claims	Police Intervention|Cost Over 3K|Claim Type
Rule Segment Points	3	1|1|1
Rule Threshold	2	2
Rule Threshold Segment	2	2
Rule Threshold Test Value	3.0	3.0
Model Action	Fast Track	Standard
Model Threshold		0.3
Model Threshold Segment	0	2
Model Threshold Test Value	0.06	0.51

Reviewing this output, the following can be determined:

- The final action recommended by the application for both claims is to use standard processing.
- For both claims, the recommended action based on rules was also to use standard processing. Each claim had a total of three risks points assigned, though different rules fired in order to reach this total.
- For Claim 1, the rule segment named *Multiple Claims* fired *true*. Three risk points were assigned for this segment, as specified on the Define tab in the application.
- For Claim 2, three different segments fired *true* (*Police Intervention*, *Cost Over 3K*, *Claim Type*). These were the first, fourth, and fifth segments in the set, respectively. A single risk point was assigned for each of these segments, again as specified on the Define tab.
- The Rule Threshold value of *2* indicates the claim needed at least two risk points to be assigned this action. (Claims with fewer than two risk points are allocated to *Fast Track*.)
- The allocation based on models was *Fast Track* for Claim 1, and *Standard* for Claim 2. Claim 1 failed to cross the Model Threshold value of 0.3 for standard processing, thus it is allocated to Segment 0 and no threshold value is shown.

Example: IBM Analytical Decision Management for Operations output configuration

When scoring IBM Analytical Decision Management for Operations, available output fields include the usecase, action, and service group for each machine, along with allocation and optimization outputs that may be useful in determining why a particular recommendation was made.

Outputs for this application can be configured in the Deployment element as follows:

```
<Deployment>
  <OutputAttribute referenceType="DimensionMember" name="Usecase" returnValue="Usecase.Allocation-Value">Usecase
  </OutputAttribute>
```

```
            <OutputAttribute referenceType="DimensionMember" name="Action" returnValue="Action.Allocation-Value">Action
  </OutputAttribute>
            <OutputAttribute referenceType="DimensionMember" name="ServiceGroup" returnValue="ServiceGroup.Allocation-Value">
            ServiceGroup</OutputAttribute>
            <OutputAttribute referenceType="Variable" name="ProbOfFailure" returnValue="ProbOfFailure.Variable-Value">
            ProbOfFailure</OutputAttribute>
            <OutputAttribute referenceType="Variable" name="Revenue" returnValue="Revenue.Variable-Value">Revenue
            </OutputAttribute>
            <OutputAttribute referenceType="Variable" name="TimeInHours" returnValue="TimeInHours.Variable-Value">
            TimeInHours</OutputAttribute>
            <OutputAttribute referenceType="Variable" name="CostPerHour" returnValue="CostPerHour.Variable-Value">
            CostPerHour</OutputAttribute>
            <OutputAttribute referenceType="Variable" name="TotalCost" returnValue="TotalCost.Variable-Value">TotalCost
            </OutputAttribute>
            <OutputAttribute referenceType="Objective" name="ObjectiveValue" returnValue="ObjectiveValue-Value">ExpectedSaving
            </OutputAttribute>
 </Deployment>
```

Example: IBM Analytical Decision Management for Demand Optimization output configuration

When scoring IBM Analytical Decision Management for Operations, available output fields include the store ID and stock requested, along with allocation and optimization outputs that may be useful in determining why a particular recommendation was made.

Outputs for this application can be configured in the Deployment element as follows:

```
<Deployment>
  <!-- OutputAttribute referenceType="DimensionMember" name="Inventory" returnValue="Inventory.Allocation-Value">
  Inventory</OutputAttribute-->
  <OutputAttribute referenceType="Variable" name="StoreID" returnValue="StoreID.Variable-Value">Output-StoreID
  </OutputAttribute>
  <OutputAttribute referenceType="Variable" name="StockRequested" returnValue="StockRequested.Variable-Value">
  Output-StockRequested</OutputAttribute>
  <OutputAttribute referenceType="Objective" name="StockAllocated" returnValue="StockAllocation-Value">
  Output-StockAllocated</OutputAttribute>
  <OutputAttribute referenceType="Variable" name="UnitRevenue" returnValue="UnitRevenue.Variable-Value">
  Output-UnitRevenue</OutputAttribute>
  <OutputAttribute referenceType="Objective" name="ObjectiveValue" returnValue="ObjectiveValue-Value">
  Output-ObjectiveValue</OutputAttribute>
 </Deployment>
```

Prompting the user for scoring parameters

In cases where certain parameters may be provided at score time, the scoring configuration will prompt the user as needed. Such parameters can be defined in the XML template as follows:

```
<Variable name="MaxOffersNum" dataType="integer" optimizationInputItem="true" prompt="Max Offers">
     <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="ConstantValueSource">
         <Value>1</Value>
     </ValueSource>
</Variable>
```

In this example the scoring configuration will prompt the user to specify *Max Offers*.

Optionally, this variable can be defined as a translatable string, as follows:

```
<Variable name="MaxOffersNum" dataType="integer" optimizationInputItem="true" prompt="$$cim/Max Offers">
```

For this to work, the value $$cim/Max Offers would need to be defined as a translatable key in IBM SPSS Collaboration and Deployment Services, and values provided for the relevant languages.

Application template examples

This section provides sample XML for several different application template examples.

IBM SPSS Modeler Advantage template

IBM SPSS Modeler Advantage is an easy-to-use application that puts the power of predictive modeling in the hands of business users. Using predictive models, you can identify patterns based on what has happened in the past, and use them to predict what is likely to happen in the future. For more information, see the *IBM SPSS Modeler Advantage User's Guide* or help.

The template for IBM SPSS Modeler Advantage is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<PredictiveApplication xmlns="http://com.spss.pasw.dms/workspace" templateName="ModelerAdvantage"
templateVersion="1" appsVersion="17.0">
    <InterfaceControl>
        <InterfacePages>
            <ApplicationHome stepIncluded="true" showGallery="true"/>
            <DataStep stepIncluded="true"/>
            <GlobalSelectionStep stepIncluded="false"/>
            <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="typeModelingDefineStep"/>
            <DeployScoreStep stepIncluded="true">
                <ImmediateBatchScoring enableScoringOptions="true"/>
            </DeployScoreStep>
            <ReportStep stepIncluded="false"/>
        </InterfacePages>
  <InterfaceFeature id="Collaboration"/>
  <InterfaceFeature id="UploadDownload"/>
        <InterfaceFeature id="MetadataDownload"/>
    </InterfaceControl>
</PredictiveApplication>
```

A detailed description of the XML template that makes up a typical modeling application follows. Information about all available elements in the schema is included in the *Schema Reference Guide*.

Refer back to the preceding XML example while reading the following information.

- The `templatename` element specifies the file name of the application template. You don't need to include the file extension. This attribute is required.

  ```
  templateName="ModelerAdvantage"
  ```

- The `templateVersion` and `appsVersion` elements are for use when upgrading a custom application template. The prebuilt application templates included with IBM Analytical Decision Management are upgraded automatically each time a new product version is released. See the topic Chapter 6, "Updating custom application templates and projects," on page 73 for more information.

  ```
  templateVersion="1" appsVersion="17.0"
  ```

- The `ApplicationHome` element indicates whether the Home page and the model Gallery section will be included in the application. These attributes are optional. The default is `true`.

  ```
  ApplicationHome stepIncluded="true" showGallery="true"
  ```

- The following elements indicate which tabs will appear in the user interface. In this modeling application, only the Data, Modeling, and Score tabs will be included. Other tabs such as Global Selections, Prioritize/Optimize, and Reports will not. These attributes are all optional. The default is `true`. For a detailed description of the various tabs in the user interface, see the user's guides.

  ```
  <DataStep stepIncluded="true"/>
  <GlobalSelectionStep stepIncluded="false"/>
  <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="typeModelingDefineStep"/>
  <DeployScoreStep stepIncluded="true">
      <ImmediateBatchScoring enableScoringOptions="true"/>
  </DeployScoreStep>
  <ReportStep stepIncluded="false"/>
  ```

- As part of the `DefineStep` element, `type="typeModelingDefineStep"` indicates that the Define tab will be of the modeling type. For other application types, you might use `typeRulesManagementDefineStep` or `typeDecisionHierarchyDefineStep`.

By default, all modeling types are available in the user interface. But if you want to turn off any modeling types, you can add attributes to `typeModelingDefineStep`. For example, to exclude the Association modeling type from the user interface, include the attribute `enableAssocationModeling=false`.

* As part of the `DeployScoreStep` element, the `ImmediateBatchScoring` element indicates whether scoring options will be included in the interface. This attribute is optional.

  `ImmediateBatchScoring enableScoringOptions="true"`

* The `InterfaceFeature` element identifies major features to include in the user interface. This flexibility is valuable if you want to block users from certain functionality. At least one value is required. Possible values are described in the following table. See the user's guides for details about user interface features.

Note that inclusion of `Collaboration` and `UploadDownload` enables all other features, as is the case with this modeling application example.

Table 5. Possible values for InterfaceFeature element.

Possible values	Features enabled
`<InterfaceFeature id="ModelExport">`	Ability to save models to the IBM SPSS Collaboration and Deployment Services Repository
`<InterfaceFeature id="ModelBuild">`	Ability to build models in IBM Analytical Decision Management applications other than IBM SPSS Modeler Advantage
`<InterfaceFeature id="RuleExport">`	Ability to save rules to the IBM SPSS Collaboration and Deployment Services Repository
`<InterfaceFeature id="RuleReference">`	Ability to reference rules stored in the IBM SPSS Collaboration and Deployment Services Repository
`<InterfaceFeature id="ModelReference">`	Ability to reference models stored in the IBM SPSS Collaboration and Deployment Services Repository
`<InterfaceFeature id="UploadDownload">`	Ability to open files from or save files to the user's local file system
`<InterfaceFeature id="Collaboration">`	The ability to reference or save objects to the IBM SPSS Collaboration and Deployment Services Repository
`<InterfaceFeature id="MetadataDownload">`	The ability to download a *.ZIP* file with metadata for the current project

IBM SPSS Rules Management template

IBM SPSS Rules Management provides a central tool for creating and editing shared rules. These rules can be used throughout applications to select and process records, and to automate decisions accordingly.

The template for IBM SPSS Rules Management is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<PredictiveApplication xmlns="http://com.spss.pasw.dms/workspace" templateName="RulesManagement"
templateVersion="1" appsVersion="17.0">
    <InterfaceControl>
        <InterfacePages>
            <ApplicationHome stepIncluded="true" showGallery="true"/>
            <DataStep stepIncluded="true"/>
            <GlobalSelectionStep stepIncluded="false"/>
            <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="typeRulesManagementDefineStep"/>
            <CombineOptimizeStep stepIncluded="false">
                <CombineOptimizeMethod/>
            </CombineOptimizeStep>
            <DeployScoreStep stepIncluded="false"/>
            <ReportStep stepIncluded="false"/>
        </InterfacePages>
```

```
      <InterfaceFeature id="Collaboration"/>
          <InterfaceFeature id="MetadataDownload"/>
     </InterfaceControl>
</PredictiveApplication>
```

- In this example, the application template XML file name is *RulesManagement.xml*:

  ```
  templateName="RulesManagement"
  ```

- This application has a Home page and two tabs: Data and Rules. The only functionality included in this application is the `DefineStep` of the rules management type:

```
<DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="typeRulesManagementDefineStep"/>
```

IBM Analytical Decision Management for Customer Interactions template

IBM Analytical Decision Management for Customer Interactions decides which promotions to offer customers when they contact your organization, and delivers recommendations to a call center, web site, or store location in real time. By combining the logic of business rules with the insight gained through predictive modeling, the application identifies the most profitable decision for each customer. For more information, see the *IBM Analytical Decision Management for Customer Interactions User's Guide* or help.

The template for IBM Analytical Decision Management for Customer Interactions is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<PredictiveApplication xmlns="http://com.spss.pasw.dms/workspace" templateName="CustomerInteractionManagement" templateVersion="1"
appsVersion="17.0" priorityDimension="Offer">
 <InterfaceControl>
  <InterfacePages>
   <ApplicationHome stepIncluded="true" showGallery="true"/>
   <DataStep stepIncluded="true"/>
   <GlobalSelectionStep stepIncluded="true"/>
   <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
   "typeDecisionHierarchyDefineStep">
    <DimensionSetting name="Campaign">
     <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
     <SelectionSection enabled="true" enableModels="true"/>
     <AllocationRuleSection enabled="true"/>
     <AggregateRuleSection enabled="false"/>
     <PredictiveModelSection enabled="false"/>
    </DimensionSetting>
   </DefineStep>
   <OptimizeStep stepIncluded="true">
    <OptimizeMethod enableNumReturnsByIP="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
    "PrioritizationOptimization"/>
   </OptimizeStep>
   <DeployScoreStep stepIncluded="true">
    <RealTimeScoring enableInteractiveQuestions="true"/>
    <ImmediateBatchScoring enableScoringOptions="true"/>
   </DeployScoreStep>
   <ReportStep stepIncluded="true"/>
  </InterfacePages>
  <InterfaceFeature id="Collaboration"/>
  <InterfaceFeature id="UploadDownload"/>
  <InterfaceFeature id="MetadataDownload"/>
 </InterfaceControl>
 <EntityDimension name="Customer">
 <Variable name="MaxOffersNum" dataType="integer" optimizationInputItem="true" prompt="MaxNumOfOffers">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
    <Value>1</Value>
   </ValueSource>
  </Variable>
  <Constraint type="max" description="MaxOffersNum">
   <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
   "MaxOffersNum"/>
   <Function Domain="double" Functor="variableReference">
    <Expression xmlns="http://com.spss.pasw.dms/rules" Domain="string">
     <Value>count</Value>
    </Expression>
   </Function>
  </Constraint>
 </EntityDimension>
 <Dimension name="Campaign">
  <Variable name="Prob.to Respond" dataType="double" description="Prob.to RespondDescription" simulateName="Offers Accepted
  Projected" simulateAction="sum" optimizationInputItem="true" prompt="">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
    <Value>0</Value>
```

```
   </ValueSource>
  </Variable>
  <Variable name="Min.Profit" dataType="double" optimizationInputItem="true" prompt="">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
    <Value>0</Value>
   </ValueSource>
  </Variable>
  <Variable name="Revenue" dataType="double" optimizationInputItem="true" prompt="">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
    <Value>0</Value>
   </ValueSource>
  </Variable>
  <Variable name="Cost" dataType="double" description="TotalCostDescription" simulateAction="sum" simulateName="Total
Cost" optimizationInputItem="true" prompt="">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
    <Value>0</Value>
   </ValueSource>
  </Variable>
  <Constraint type="min" description="MinProfitConstraint">
   <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
   "Min.Profit"/>
   <Function Domain="double" Functor="variableReference">
    <Expression xmlns="http://com.spss.pasw.dms/rules" Domain="string">
     <Value>value</Value>
    </Expression>
    <Expression xmlns="http://com.spss.pasw.dms/rules" Domain="string">
     <Value>Objective</Value>
    </Expression>
    <Expression xmlns="http://com.spss.pasw.dms/rules" Domain="string">
     <Value>Expected Profit</Value>
    </Expression>
    <Expression xmlns="http://com.spss.pasw.dms/rules" Domain="string">
     <Value>Value</Value>
    </Expression>
   </Function>
  </Constraint>
 </Dimension>
 <Dimension name="Offer" parentDimension="Campaign">
 </Dimension>
 <Optimization algorithm="Heuristic">
  <ObjectiveFunction Domain="double" Functor="-" Name="Expected Profit" description=
  "Expected_profit_using_probability_to_respond">
   <Expression xmlns="http://com.spss.pasw.dms/rules" Domain="double" Functor="*">
    <Expression Domain="double" Functor="variableReference">
     <Expression>
      <Value>value</Value>
     </Expression>
     <Expression>
      <Value>Variable</Value>
     </Expression>
     <Expression>
      <Value>Prob.to Respond</Value>
     </Expression>
     <Expression>
      <Value>Value</Value>
     </Expression>
    </Expression>
    <Expression Domain="double" Functor="variableReference">
     <Expression>
      <Value>value</Value>
     </Expression>
     <Expression>
      <Value>Variable</Value>
     </Expression>
     <Expression>
      <Value>Revenue</Value>
     </Expression>
     <Expression>
      <Value>Value</Value>
     </Expression>
    </Expression>
   </Expression>
   <Expression xmlns="http://com.spss.pasw.dms/rules" Domain="double" Functor="variableReference">
    <Expression>
     <Value>value</Value>
    </Expression>
    <Expression>
     <Value>Variable</Value>
    </Expression>
    <Expression>
     <Value>Cost</Value>
    </Expression>
    <Expression>
     <Value>Value</Value>
```

```
      </Expression>
      </Expression>
    </ObjectiveFunction>
  </Optimization>
  <Deployment>
    <OutputAttribute referenceType="DimensionMember" name="Campaign" returnValue="Campaign.Allocation-Value">Campaign
    </OutputAttribute>
    <OutputAttribute referenceType="DimensionMember" name="Offer" returnValue="Offer.Allocation-Value">Offer
    </OutputAttribute>
    <OutputAttribute referenceType="Objective" name="Expected Profit" returnValue="Expected Profit-Value">
    Output-PredictedProfit</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="MaxOffersNum" returnValue="MaxOffersNum.Variable-Value">
    Output-MaxOffersNum</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="Min.Profit" returnValue="Min.Profit.Variable-Value">Output-MinProfit
    </OutputAttribute>
    <OutputAttribute referenceType="Variable" name="Prob.to Respond" returnValue="Prob.to Respond.Variable-Value">
    Output-ProbToRespond</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="Revenue" returnValue="Revenue.Variable-Value">Output-Revenue
    </OutputAttribute>
    <OutputAttribute referenceType="Variable" name="Cost" returnValue="Cost.Variable-Value">Output-Cost
    </OutputAttribute>
  </Deployment>
</PredictiveApplication>
```

- In this example, the application template XML file name is *CustomerInteractionManagement.xml*:

  ```
  templateName="CustomerInteractionManagement"
  ```

- This application has six tabs: Data, Global Selections, Define, Prioritize (`OptimizeStep`), Deploy, and Reports:

  ```
  <DataStep stepIncluded="true"/>
  <GlobalSelectionStep stepIncluded="true"/>
  <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
  "typeDecisionHierarchyDefineStep">
   <DimensionSetting name="Campaign">
    <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
    <SelectionSection enabled="true" enableModels="true"/>
    <AllocationRuleSection enabled="true"/>
    <AggregateRuleSection enabled="false"/>
    <PredictiveModelSection enabled="false"/>
   </DimensionSetting>
  </DefineStep>
  <OptimizeStep stepIncluded="true">
   <OptimizeMethod enableNumReturnsByIP="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="PrioritizationOptimization"/>
  </OptimizeStep>
  <DeployScoreStep stepIncluded="true">
   <RealTimeScoring enableInteractiveQuestions="true"/>
   <ImmediateBatchScoring enableScoringOptions="true"/>
  </DeployScoreStep>
  <ReportStep stepIncluded="true"/>
  ```

- This application's XML also defines a top-level entity dimension (`Customer`), and two main dimension members (`Campaign` and `Offer`). Note that variable names are defined for the dimensions. These are hard-coded in the XML application template rather than defined in the screen text properties files.

  Applications can only have one level of children per dimension (on the Define tab, the user interface cannot display more than one level under each dimension).

  ```
  <EntityDimension name="Customer">
  .
  .
  .
  <Dimension name="Campaign">
  .
  .
  .
  <Dimension name="Offer" parentDimension="Campaign">
  ```

- The `Optimization` element defines the equation used to "value" each potential result by the optimization algorithm as it strives to find the solution with the minimum or maximum value. Supported algorithms include `Heuristic`, which indicates that the "greedy" prioritization form of optimization will be used, or `CPLEX`, which indicates that the advanced IBM CPLEX form of optimization will be used. You can also specify `None` to disable the use of the optimization.

  ```
  <Optimization algorithm="Heuristic">
  ```

  The `Optimization` element contains the `ObjectiveFunction`. If you want to customize the objective function (the prioritization equation or optimization equation) used in an application, contact your SPSS representative if you have questions.

The prioritization equation used in the sample IBM Analytical Decision Management for Customer Interactions application is displayed on the Prioritize tab in the user interface.

IBM Analytical Decision Management for Campaign Optimization template

IBM Analytical Decision Management for Campaign Optimization is similar to the IBM Analytical Decision Management for Customer Interactions application in that it decides which promotions to offer customers when they contact your organization, and delivers recommendations to a channel such as a call center, retail web site, or store location. By combining the logic of business rules with the insight gained through predictive modeling, the application identifies the most profitable decision for each customer. IBM Analytical Decision Management for Campaign Optimization uses IBM CPLEX optimization and has the added value of maximizing the return on your campaign investment by determining the best offers for individual customers within constraints such as your budget, channel capacity, and contact policies.

For more information, see the *IBM Analytical Decision Management for Campaign Optimization User's Guide* or help.

The template for IBM Analytical Decision Management for Campaign Optimization is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<PredictiveApplication xmlns="http://com.spss.pasw.dms/workspace" templateName="CampaignOptimization" templateVersion="1"
appsVersion="17.0" priorityDimension="Offer">
 <InterfaceControl>
  <InterfacePages>
   <ApplicationHome stepIncluded="true" showGallery="true"/>
   <DataStep stepIncluded="true"/>
   <GlobalSelectionStep stepIncluded="true"/>
   <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
   "typeDecisionHierarchyDefineStep">
    <DimensionSetting name="Campaign">
     <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
     <SelectionSection enabled="true" enableModels="true"/>
     <AllocationRuleSection enabled="true"/>
     <AggregateRuleSection enabled="false"/>
     <PredictiveModelSection enabled="false"/>
    </DimensionSetting>
    <DimensionSetting name="Channel">
     <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
     <SelectionSection enabled="true" enableModels="true"/>
     <AllocationRuleSection enabled="false"/>
     <AggregateRuleSection enabled="false"/>
     <PredictiveModelSection enabled="false"/>
    </DimensionSetting>
   </DefineStep>
   <OptimizeStep stepIncluded="true">
    <OptimizeMethod enableNumReturnsByIP="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="PrioritizationOptimization"/>
   </OptimizeStep>
   <DeployScoreStep stepIncluded="true">
    <RealTimeScoring enableInteractiveQuestions="true"/>
    <ImmediateBatchScoring enableScoringOptions="true"/>
   </DeployScoreStep>
   <ReportStep stepIncluded="true"/>
  </InterfacePages>
  <InterfaceFeature id="Collaboration"/>
  <InterfaceFeature id="UploadDownload"/>
  <InterfaceFeature id="MetadataDownload"/>
 </InterfaceControl>
 <EntityDimension name="Customer">
  <Variable name="MaxOffersNum" dataType="integer" description="MaxOffersNumDescription" optimizationInputItem="true"
  prompt="MaxOffersNum">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
    <Value>1</Value>
   </ValueSource>
  </Variable>

  <Variable name="TotalBudget" dataType="double" description="TotalBudgetDescription" optimizationInputItem="true"
  prompt="TotalBudget">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
    <Value>10000</Value>
   </ValueSource>
  </Variable>
```

```
 <Variable name="RecentOffersNum" dataType="integer" description="RecentOffersNumDescription" optimizationInputItem=
"true" prompt="RecentOffersNum">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>

 <DerivedVariable name="TotalCost" dataType="double" description="BudgetSpentDescription" simulateName="Total Budget
 Spent" simulateAction="sum" >
  <VariableExpression expression="'${OfferCost}' + '${ChannelCost}'"/>
 </DerivedVariable>

</EntityDimension>
<Dimension name="Campaign" description="CampaignDescriptionText">
 <Variable name="CampaignBudget" dataType="double" description="CampaignBudgetDescription" optimizationInputItem="true"
 prompt="" inherited="false">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>

 <Variable name="MinCampaignOfferCount" dataType="integer" description="MinCampaignOfferCountDescription"
 optimizationInputItem="true" prompt="" inherited="false">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>

 <!--Variable name="CampaignSize" dataType="integer" optimizationInputItem="true" prompt="" inherited="false">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable-->
</Dimension>
<Dimension name="Offer" parentDimension="Campaign" description="OfferDescriptionText">
 <Variable name="Prob.to Respond" dataType="double" description="Prob.to RespondDescription" simulateName="Offers
 Accepted Projected" simulateAction="sum" optimizationInputItem="true" prompt="">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
 <!--Variable name="Min.Profit" dataType="double" optimizationInputItem="true" prompt="">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable-->
 <Variable name="Revenue" dataType="double" description="RevenueDescription" optimizationInputItem="true" prompt="">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
 <Variable name="OfferCost" dataType="double" description="OfferCostDescription" simulateAction="sum" simulateName="Total
 Offer Cost" optimizationInputItem="true" prompt="">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
 <Variable name="OffersAvailable" dataType="integer" description="OffersAvailableDescription" optimizationInputItem=
 "true" prompt="" inherited="false">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
</Dimension>

<Dimension name="Channel" description="ChannelDescriptionText">
 <Variable name="Capacity" dataType="integer" description="CapacityDescription" optimizationInputItem="true" prompt="">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
 <Variable name="ChannelCost" dataType="double" description="ChannelCostDescription" simulateAction="sum" simulateName=
 "Total Channel Cost" optimizationInputItem="true" prompt="">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
</Dimension>


<Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path="/Applications/CampaignOptimization/Optimization/
CampaignOptimization2.mod">
 <ObjectiveFunction Domain="double" Name="Expected Profit" description="Expected_profit_using_probability_to_respond">
  <ExpressionFormat format="( ${Prob.to Respond} * ${Revenue} ) - ( ${OfferCost} + ${ChannelCost} )"/>
```

```
 <ExternalUsage controlVariable="use_expected_profit_function" variableType="int" enabledValue="1"
 disabledValue="0"/>
</ObjectiveFunction>
<ObjectiveFunction Domain="double" Name="ROI" description="Return_On_Investment" simulateAction="average">
 <ExpressionFormat format="((${Prob.to Respond} * ${Revenue}) - (${OfferCost} + ${ChannelCost})) / (${OfferCost} +
 ${ChannelCost})"/>
 <ExternalUsage controlVariable="use_roi_function" variableType="int" enabledValue="1" disabledValue="0"/>
</ObjectiveFunction>


<!--Constraint type="min" name="ctMinProfitConstraint">
 <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
 "Min.Profit"/>
 <ExpressionFormat format="${ObjectiveValue}"/>
 <ExternalUsage controlVariable="use_min_profit_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
</Constraint-->
<Constraint type="max" name="ctCampaignBudgetConstraint" entityScoped="false" enabled="false" description=
"CampaignBudgetConstraint">
 <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
 "CampaignBudget"/>
 <ExpressionFormat format="${FN_sum} ( ${OfferCost} + ${ChannelCost} )"/>
 <ExternalUsage controlVariable="use_campaign_budget_constraint" variableType="int" enabledValue="1"
 disabledValue="0"/>
</Constraint>
<Constraint type="min" name="ctMinCampaignSizeConstraint" entityScoped="false" enabled="false" description=
"MinCampaignOffers">
 <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
 "MinCampaignOfferCount"/>
 <ExpressionFormat format="${FN_count} ( ${Offer} )"/>
 <ExternalUsage controlVariable="use_min_campaign_offers_constraint" variableType="int" enabledValue="1"
 disabledValue="0"/>
</Constraint>
<!--Constraint type="max" name="ctCampaignSizeConstraint" entityScoped="false" description="MaxOffersPerCampaign">
 <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
 "CampaignSize"/>
 <ExpressionFormat format="${FN_count} ( ${Offer} )"/>
 <ExternalUsage controlVariable="use_campaign_size_constraint" variableType="int" enabledValue="1" disabledValue=
 "0"/>
</Constraint-->
<Constraint type="max" name="ctOffersAvailableConstraint" entityScoped="false" description="MaxOffersAvailable">
 <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
 "OffersAvailable"/>
 <ExpressionFormat format="${FN_count} ( ${Offer} )"/>
 <ExternalUsage controlVariable="use_offers_available_constraint" variableType="int" enabledValue="1"
 disabledValue="0"/>
</Constraint>
<Constraint type="max" name="ctTotalBudgetConstraint" entityScoped="false" description="TotalBudgetForAllCampaigns">
 <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
 "TotalBudget"/>
 <ExpressionFormat format="${FN_sum} ( ${OfferCost} + ${ChannelCost} )"/>
 <ExternalUsage controlVariable="use_total_budget_constraint" variableType="int" enabledValue="1" disabledValue=
 "0"/>
</Constraint>
<Constraint type="max" name="ctMaxOffersConstraint" description="MaxOffersPerCustomer">
 <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
 "MaxOffersNum"/>
 <ExpressionFormat format="${FN_count} ( ${Offer} / ${Customer} ) + ( ${RecentOffersNum} )"/>
 <ExternalUsage controlVariable="use_max_offers_constraint" variableType="int" enabledValue="1" disabledValue=
 "0"/>
</Constraint>
<Constraint type="max" name="ctChannelCapacityConstraint" entityScoped="false" description="MaxOffersPerChannel">
 <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
 "Capacity"/>
 <ExpressionFormat format="${FN_count} ( ${Channel} )"/>
 <ExternalUsage controlVariable="use_channel_capacity_constraint" variableType="int" enabledValue="1"
 disabledValue="0"/>
</Constraint>
<OPLMapping tupleSetVariable="EntityAllocations" outputDecisionVariable="ObjectiveFunction">
 <EntityField referenceType="Variable" name="MaxOffersNum" dataReturn="Value" tupleField=
 "MaxOffersNum_Variable_Value" tupleFieldType="int"/>
 <EntityField referenceType="Variable" name="RecentOffersNum" dataReturn="Value" tupleField=
 "RecentOffersNum_Variable_Value" tupleFieldType="int"/>
 <EntityField referenceType="Variable" name="TotalBudget" dataReturn="Value" tupleField=
 "TotalBudget_Variable_Value" tupleFieldType="float"/>
 <EntityField referenceType="System" name="entity" tupleField="entity" tupleFieldType="int"/>
 <EntityField referenceType="DimensionMember" name="Campaign" dataReturn="Value" tupleField=
 "Campaign_Allocation_Value" tupleFieldType="string"/>
 <EntityField referenceType="Variable" name="CampaignBudget" dataReturn="Value" tupleField="Budget_Variable_Value"
 tupleFieldType="float"/>
 <EntityField referenceType="Variable" name="MinCampaignOfferCount" dataReturn="Value" tupleField=
 "MinCampaignOfferCount_Variable_Value" tupleFieldType="int"/>
 <!-- EntityField referenceType="Variable" name="CampaignSize" dataReturn="Value" tupleField=
 "CampaignSize_Variable_Value" tupleFieldType="int"/-->
 <EntityField referenceType="DimensionMember" name="Offer" dataReturn="Value" tupleField="Offer_Allocation_Value"
 tupleFieldType="string"/>
```

```
<!-- EntityField referenceType="DimensionMember" name="Offer" dataReturn="Segment" tupleField=
"Offer_Allocation_Segment" tupleFieldType="string"/-->
<!--EntityField referenceType="DimensionMember" name="Offer" dataReturn="Segment Name" tupleField=
"Offer_Allocation_Segment_Name" tupleFieldType="string"/-->
<EntityField referenceType="DimensionMember" name="Channel" dataReturn="Value" tupleField=
"Channel_Allocation_Value" tupleFieldType="string"/>
<EntityField referenceType="Variable" name="Prob.to Respond" dataReturn="Value" tupleField=
"Prob_to_Respond_Variable_Value" tupleFieldType="float"/>
<!-- EntityField referenceType="Variable" name="Min.Profit" dataReturn="Value" tupleField=
"Min_Profit_Variable_Value" tupleFieldType="float"/-->
<EntityField referenceType="Variable" name="Revenue" dataReturn="Value" tupleField="Revenue_Variable_Value"
tupleFieldType="float"/>
<EntityField referenceType="Variable" name="OfferCost" dataReturn="Value" tupleField="OfferCost_Variable_Value"
tupleFieldType="float"/>
<EntityField referenceType="Variable" name="OffersAvailable" dataReturn="Value" tupleField=
"OffersAvailable_Variable_Value" tupleFieldType="int"/>
<EntityField referenceType="Variable" name="Capacity" dataReturn="Value" tupleField="Capacity_Variable_Value"
tupleFieldType="int"/>
<EntityField referenceType="Variable" name="ChannelCost" dataReturn="Value" tupleField=
"ChannelCost_Variable_Value" tupleFieldType="float"/>
<EntityField referenceType="System" name="priority" tupleField="priority" tupleFieldType="int"/>
<EntityField referenceType="System" name="entity_allocation_id" tupleField="entity_allocation_id" tupleFieldType=
"int"/>
<!-- EntityField referenceType="Objective" name="ObjectiveValue" dataReturn="Value" tupleField=
"Expected_Profit_Value" tupleFieldType="float"/-->

<OptimizationOutput name="IsOptimal-Value" valueVariable="OptimalAllocations" variableType="int" thresholdType=
"equal" thresholdValue="1"/>
<OptimizationOutput name="ObjectiveValue-Value" valueVariable="Contribution" variableType="float" role=
"objectiveValue" selectionOnly="false"/>
</OPLMapping>
</Optimization>
<Deployment>
<OutputAttribute referenceType="DimensionMember" name="Campaign" returnValue="Campaign.Allocation-Value">Campaign
</OutputAttribute>
<OutputAttribute referenceType="DimensionMember" name="Offer" returnValue="Offer.Allocation-Value">Offer
</OutputAttribute>
<OutputAttribute referenceType="Objective" name="ObjectiveValue" returnValue="ObjectiveValue-Value">
Output-ObjectiveValue</OutputAttribute>
<OutputAttribute referenceType="Variable" name="MaxOffersNum" returnValue="MaxOffersNum.Variable-Value">
Output-MaxOffersNum</OutputAttribute>
<OutputAttribute referenceType="Variable" name="RecentOffersNum" returnValue="RecentOffersNum.Variable-Value">
Output-RecentOffersNum</OutputAttribute>
<OutputAttribute referenceType="Variable" name="TotalBudget" returnValue="TotalBudget.Variable-Value">Output-TotalBudget
</OutputAttribute>
<!-- OutputAttribute referenceType="Variable" name="CampaignSize" returnValue="CampaignSize.Variable-Value">
Output-CampaignSize</OutputAttribute-->
<!-- OutputAttribute referenceType="Variable" name="Min.Profit" returnValue="Min.Profit.Variable-Value">Output-MinProfit
</OutputAttribute-->
<OutputAttribute referenceType="Variable" name="Prob.to Respond" returnValue="Prob.to Respond.Variable-Value">
Output-ProbToRespond</OutputAttribute>
<OutputAttribute referenceType="Variable" name="Revenue" returnValue="Revenue.Variable-Value">Output-Revenue
</OutputAttribute>
<OutputAttribute referenceType="Variable" name="OfferCost" returnValue="OfferCost.Variable-Value">Output-OfferCost
</OutputAttribute>
<OutputAttribute referenceType="Variable" name="ChannelCost" returnValue="ChannelCost.Variable-Value">Output-ChannelCost
</OutputAttribute>
</Deployment>
</PredictiveApplication>
```

- In this example, the application template XML file name is *CampaignOptimization.xml*:

```
templateName="CampaignOptimization"
```

- This application has six tabs: Data, Global Selections, Define, Optimize, Deploy, and Reports:

```
<DataStep stepIncluded="true"/>
<GlobalSelectionStep stepIncluded="true"/>
<DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
"typeDecisionHierarchyDefineStep">
    <DimensionSetting name="Campaign">
        <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
        <SelectionSection enabled="true" enableModels="true"/>
        <AllocationRuleSection enabled="true"/>
        <AggregateRuleSection enabled="false"/>
        <PredictiveModelSection enabled="false"/>
    </DimensionSetting>
    <DimensionSetting name="Channel">
        <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
        <SelectionSection enabled="true" enableModels="true"/>
        <AllocationRuleSection enabled="false"/>
        <AggregateRuleSection enabled="false"/>
        <PredictiveModelSection enabled="false"/>
    </DimensionSetting>
</DefineStep>
```

```
<OptimizeStep stepIncluded="true" enableTest="false">
    <OptimizeMethod enableNumReturnsByIP="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
     "PrioritizationOptimization"/>
</OptimizeStep>
<DeployScoreStep stepIncluded="true">
    <RealTimeScoring enableInteractiveQuestions="false"/>
    <ImmediateBatchScoring enableScoringOptions="true"/>
</DeployScoreStep>
<ReportStep stepIncluded="true"/>
```

- As with the IBM Analytical Decision Management for Customer Interactions application, this application's XML also defines a `Customer` entity dimension and a top-level `Campaign` dimension with an `Offer` child. However, a second top-level dimension called `Channel` is also defined. The two top-level dimensions (Campaign and Channel) appear as tabs on the Define tab of the application user interface. When users create new IBM Analytical Decision Management for Campaign Optimization projects, they will be prompted to select whether to display one or both dimensions and in which order to display the tabs. Both tabs are displayed by default, with the Campaign tab appearing first.

  Note that variable names are defined for the dimensions. These are hard-coded in the XML application template rather than defined in the screen text properties files.

  Applications can only have one level of children per dimension (on the Define tab, the user interface cannot display more than one level under each dimension).

```
<EntityDimension name="Customer">
.
.
.
<Dimension name="Campaign">
.
.
.
<Dimension name="Offer" parentDimension="Campaign">
.
.
.
<Dimension name="Channel">
```

- The `Optimization` element defines the equation used to "value" each potential result by the optimization algorithm as it strives to find the solution with the minimum or maximum value. Supported algorithms include `CPLEX`, which indicates that the advanced IBM CPLEX form of optimization will be used, or `Heuristic`, which indicates that the "greedy" prioritization form of optimization will be used. You can also specify `None` to disable the use of the optimization.

  The IBM Analytical Decision Management for Campaign Optimization application uses CPLEX optimization, and points to the optimization model *CampaignOptimization.mod* that ships with IBM Analytical Decision Management. Do not modify this file. If you need to modify the file, contact an IBM SPSS representative.

```
<Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path=
 "/Applications/CampaignOptimization/Optimization/CampaignOptimization.mod">
```

  The `Optimization` element contains the `ObjectiveFunction`. If you want to customize the objective function (the optimization equation or prioritization equation) used in an application, contact your SPSS representative if you have questions.

  The two optimization equations used in the sample IBM Analytical Decision Management for Campaign Optimization application are displayed on the Optimize tab in the user interface. You can choose between an *Total projected profit based on current project settings* equation or a *Return on investment* equation. For more information, see the *IBM Analytical Decision Management for Campaign Optimization User's Guide* or help.

  The Optimization element also contains several constraints (`CampaignBudgetConstraint`, `MinCampaignOffers`, `MaxOffersPerCampaign`, `MaxOffersAvailable`, `TotalBudgetForAllCampaigns`, `MaxOffersPerCustomer`, and `MaxOffersPerChannel`. These constraints are displayed on the Optimize tab of the application, allowing users to select which constraints apply to their current project.

IBM Analytical Decision Management for Claims template

Using IBM Analytical Decision Management for Claims, organizations can harness the power of predictive analytics to process incoming claims in real time. For example, claims can be set on a "fast track" for immediate payment, processed in the normal manner, or referred to the special investigations unit. For more information, see the *IBM Analytical Decision Management for Claims User's Guide* or help.

The template for IBM Analytical Decision Management for Claims is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PredictiveApplication xmlns="http://com.spss.pasw.dms/workspace" templateName="ClaimsManagement"
templateVersion="1" appsVersion="17.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <InterfaceControl>
  <InterfacePages>
   <ApplicationHome stepIncluded="true" showGallery="true"/>
   <DataStep stepIncluded="true"/>
   <GlobalSelectionStep stepIncluded="true"/>
   <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="typeDecisionHierarchyDefineStep">
    <DimensionSetting name="Claim Area">
     <SelectionSection enabled="true" enableModels="true"/>
     <AggregateRuleSection enabled="true"/>
     <PredictiveModelSection enabled="true"/>
     <AllocationRuleSection enabled="false"/>
     <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
    </DimensionSetting>
   </DefineStep>
   <CombineStep stepIncluded="true" enableWhatif="true" enableTest="true">
   </CombineStep>
   <DeployScoreStep stepIncluded="true">
    <RealTimeScoring enableInteractiveQuestions="true"/>
    <ImmediateBatchScoring enableScoringOptions="true"/>
   </DeployScoreStep>
   <ReportStep stepIncluded="true"/>
  </InterfacePages>
  <InterfaceFeature id="Collaboration"/>
  <InterfaceFeature id="UploadDownload"/>
  <InterfaceFeature id="MetadataDownload"/>
 </InterfaceControl>
 <EntityDimension name="Claim"/>
 <Dimension name="Claim Area">
 </Dimension>
 <Dimension name="Action" parentDimension="Claim Area">
 </Dimension>
 <Optimization algorithm="None">
  <ObjectiveFunction/>
 </Optimization>
 <Deployment>
  <OutputAttribute referenceType="DimensionMember" name="Claim Area"
  returnValue="Claim Area.Allocation-Value">Claim Area</OutputAttribute>
  <OutputAttribute referenceType="DimensionMember" name="Action"
  returnValue="Action.Allocation-Value">Action</OutputAttribute>
 </Deployment>
</PredictiveApplication>
```

- In this example, the application template XML file name is *ClaimsManagement.xml*:

  ```xml
  templateName="ClaimsManagement"
  ```

- This application has six tabs: Data, Global Selections, Define, Combine, Deploy, and Reports:

  ```xml
  <DataStep stepIncluded="true"/>
  <GlobalSelectionStep stepIncluded="true"/>
  <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="typeDecisionHierarchyDefineStep">
     <DimensionSetting name="Claim Area">
        <SelectionSection enabled="true" enableModels="true"/>
        <AggregateRuleSection enabled="true"/>
        <PredictiveModelSection enabled="true"/>
        <AllocationRuleSection enabled="false"/>
        <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
     </DimensionSetting>
  </DefineStep>
  <CombineStep stepIncluded="true" enableWhatif="true" enableTest="true">
  </CombineStep>
  <DeployScoreStep stepIncluded="true">
     <RealTimeScoring enableInteractiveQuestions="true"/>
     <ImmediateBatchScoring enableScoringOptions="true"/>
  </DeployScoreStep>
  <ReportStep stepIncluded="true"/>
  ```

- This application's XML also defines a top-level entity dimension (`Claim`) and two main dimension members (`Claim Area` and `Action`) and their dimension members. These dimensions will be displayed on the Define tab in the user interface.

  Applications can only have one level of children per dimension (on the Define tab, the user interface cannot display more than one level under each dimension).

```
<EntityDimension name="Claim"/>
<Dimension name="Claim Area">
.
.
.
<Dimension name="Action" parentDimension="Claim Area">
```

IBM Analytical Decision Management for Operations template

IBM Analytical Decision Management for Operations uses optimization to avoid costly machine downtime and unnecessary engineer visits by predicting future machine failures. You might anticipate which parts are likely to fail first and replace them at the appropriate time. This helps to maximize profits and avoid losses due to machines failing and being out of action by predicting when they will fail.

For more information, see the *IBM Analytical Decision Management for Operations User's Guide* or help.

The template for IBM Analytical Decision Management for Operations is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<PredictiveApplication xmlns="http://com.spss.pasw.dms/workspace" templateName="PredictiveMaintenance" templateVersion="1"
appsVersion="17.0">
 <InterfaceControl>
  <InterfacePages>
   <ApplicationHome stepIncluded="true" showGallery="true"/>
   <DataStep stepIncluded="true"/>
   <GlobalSelectionStep stepIncluded="true"/>
   <DefineStep stepIncluded="true" enableInteractionPoints="false" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:type="typeDecisionHierarchyDefineStep">
    <DimensionSetting name="Usecase">
                   <SelectionSection enabled="true" enableModels="true"/>
                   <AggregateRuleSection enabled="false"/>
                   <PredictiveModelSection enabled="false"/>
                   <AllocationRuleSection enabled="true"/>
                   <PlanningSection enableInteractionPoints="false" enableStartEndDates="false"/>
    </DimensionSetting>
   <DimensionSetting name="ServiceGroup">
    <SelectionSection enabled="true" enableModels="true"/>
                   <AggregateRuleSection enabled="false"/>
                   <AllocationRuleSection enabled="false"/>
    <PredictiveModelSection enabled="false"/>
                   <PlanningSection enableInteractionPoints="false" enableStartEndDates="false"/>
               </DimensionSetting>
   </DefineStep>
   <OptimizeStep stepIncluded="true" enableTest="true">
    <OptimizeMethod enableNumReturnsByIP="false" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="PrioritizationOptimization"/>
   </OptimizeStep>
   <DeployScoreStep stepIncluded="true">
               <ImmediateBatchScoring enableScoringOptions="true"/>
               <RealTimeScoring enableInteractiveQuestions="false"/>
   </DeployScoreStep>
   <ReportStep stepIncluded="false"/>
  </InterfacePages>
  <InterfaceFeature id="Collaboration"/>
  <InterfaceFeature id="UploadDownload"/>
  <InterfaceFeature id="MetadataDownload"/>
 </InterfaceControl>
 <EntityDimension name="Machine">
  <Variable name="TotalBudget" dataType="double" description="TotalBudgetDescription" optimizationInputItem="true"
  prompt="">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
    <Value>10000</Value>
   </ValueSource>
  </Variable>
  <Variable name="HoursInPeriod" dataType="double" description="HoursInPeriodDescription" optimizationInputItem="true"
  prompt="">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
    <Value>8</Value>
   </ValueSource>
  </Variable>
```

```xml
    <DerivedVariable name="TotalCost" dataType="double" description="TotalCostDescription" simulateName="BudgetSpent"
    simulateAction="sum" >
     <VariableExpression expression="'${TimeInHours}' * '${CostPerHour}'"/>
    </DerivedVariable>
  </EntityDimension>
  <Dimension name="Usecase" description="UsecaseDescription">
  </Dimension>
  <Dimension name="Action" parentDimension="Usecase" description="ActionDescription">
   <Variable name="ProbOfFailure" dataType="double" description="ProbOfFailureDescription" simulateName="FailuresPrevented"
   simulateAction="sum" optimizationInputItem="true" prompt="">
    <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
     <Value>0</Value>
    </ValueSource>
   </Variable>
   <Variable name="Revenue" dataType="double" description="RevenueDescription" optimizationInputItem="true" prompt="">
    <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
     <Value>0</Value>
    </ValueSource>
   </Variable>
   <Variable name="TimeInHours" dataType="double" description="TimeInHoursDescription" simulateAction="sum" simulateName=
   "TotalTimeInHours" optimizationInputItem="true" prompt="">
    <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
     <Value>0</Value>
    </ValueSource>
   </Variable>
  </Dimension>
  <Dimension name="ServiceGroup" description="ServiceGroupDescription">
   <Variable name="NumberOfStaff" dataType="double" description="NumberOfStaffDescription" optimizationInputItem="true"
   prompt="">
    <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
     <Value>0</Value>
    </ValueSource>
   </Variable>
   <Variable name="CostPerHour" dataType="double" description="CostPerHourDescription" optimizationInputItem="true"
   prompt="">
    <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
     <Value>0</Value>
    </ValueSource>
   </Variable>
  </Dimension>

  <Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path="/Applications/PredictiveMaintenance/Optimization/
  PredictiveMaintenance.mod">
   <ObjectiveFunction Domain="double" Name="ExpectedSavings" description="ExpectedSavingsFunction">
    <ExpressionFormat format="( ${ProbOfFailure} * ${Revenue} ) - ( ${TimeInHours} * ${CostPerHour} )"/>
   </ObjectiveFunction>

   <Constraint type="max" name="ctTotalBudgetConstraint" entityScoped="false" description="TotalBudgetConstraint">
    <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
    "TotalBudget"/>
            <ExternalUsage controlVariable="use_total_budget_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
            <ExpressionFormat format="${FN_sum} ( ${TimeInHours} * ${CostPerHour} )"/>
   </Constraint>
   <Constraint type="max" name="ctNumberOfStaffConstraint" entityScoped="false" description="NumberOfStaffConstraint">
    <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
    "NumberOfStaff"/>
    <ExternalUsage controlVariable="use_number_of_staff_constraint" variableType="int" enabledValue="1"
    disabledValue="0"/>
    <ExpressionFormat format="${FN_count} ( ${FN_sum} ( ${TimeInHours} ) / ${HoursInPeriod} )"/>
   </Constraint>
   <OPLMapping tupleSetVariable="EntityAllocations" outputDecisionVariable="ObjectiveFunction">
            <EntityField referenceType="System" name="entity" tupleField="entity" tupleFieldType="int"/>
            <EntityField referenceType="DimensionMember" name="Usecase" dataReturn="Value" tupleField="Usecase_Allocation_Value"
            tupleFieldType="string"/>
            <EntityField referenceType="DimensionMember" name="Action" dataReturn="Value" tupleField="Action_Allocation_Value"
            tupleFieldType="string"/>
            <EntityField referenceType="DimensionMember" name="ServiceGroup" dataReturn="Value" tupleField=
            "ServiceGroup_Allocation_Value" tupleFieldType="string"/>
            <EntityField referenceType="Variable" name="TotalBudget" dataReturn="Value" tupleField="TotalBudget_Variable_Value"
            tupleFieldType="float"/>
    <EntityField referenceType="Variable" name="ProbOfFailure" dataReturn="Value" tupleField=
    "ProbOfFailure_Variable_Value" tupleFieldType="float"/>
    <EntityField referenceType="Variable" name="Revenue" dataReturn="Value" tupleField="Revenue_Variable_Value"
    tupleFieldType="float"/>
    <EntityField referenceType="Variable" name="TimeInHours" dataReturn="Value" tupleField=
    "TimeInHours_Variable_Value" tupleFieldType="float"/>
    <EntityField referenceType="Variable" name="CostPerHour" dataReturn="Value" tupleField=
    "CostPerHour_Variable_Value" tupleFieldType="float"/>
    <EntityField referenceType="Variable" name="NumberOfStaff" dataReturn="Value" tupleField=
    "NumberOfStaff_Variable_Value" tupleFieldType="float"/>
    <EntityField referenceType="Variable" name="HoursInPeriod" dataReturn="Value" tupleField=
    "HoursInPeriod_Variable_Value" tupleFieldType="float"/>
    <EntityField referenceType="System" name="entity_allocation_id" tupleField="entity_allocation_id" tupleFieldType=
    "int"/>
```

```
            <OptimizationOutput name="IsOptimal-Value" valueVariable="OptimalAllocations" variableType="int" thresholdType="equal"
            thresholdValue="1"/>
            <OptimizationOutput name="ObjectiveValue-Value" valueVariable="Contribution" variableType="float" role="objectiveValue"
            selectionOnly="false"/>
        </OPLMapping>
  </Optimization>
  <Deployment>
        <OutputAttribute referenceType="DimensionMember" name="Usecase" returnValue="Usecase.Allocation-Value">Usecase
        </OutputAttribute>
        <OutputAttribute referenceType="DimensionMember" name="Action" returnValue="Action.Allocation-Value">Action</OutputAttribute>
        <OutputAttribute referenceType="DimensionMember" name="ServiceGroup" returnValue="ServiceGroup.Allocation-Value">ServiceGroup
        </OutputAttribute>
        <OutputAttribute referenceType="Variable" name="ProbOfFailure" returnValue="ProbOfFailure.Variable-Value">ProbOfFailure
        </OutputAttribute>
        <OutputAttribute referenceType="Variable" name="Revenue" returnValue="Revenue.Variable-Value">Revenue</OutputAttribute>
        <OutputAttribute referenceType="Variable" name="TimeInHours" returnValue="TimeInHours.Variable-Value">TimeInHours
        </OutputAttribute>
        <OutputAttribute referenceType="Variable" name="CostPerHour" returnValue="CostPerHour.Variable-Value">CostPerHour
        </OutputAttribute>
        <OutputAttribute referenceType="Variable" name="TotalCost" returnValue="TotalCost.Variable-Value">TotalCost
        </OutputAttribute>
        <OutputAttribute referenceType="Objective" name="ObjectiveValue" returnValue="ObjectiveValue-Value">ExpectedSaving
        </OutputAttribute>
  </Deployment>
</PredictiveApplication>
```

- In this example, the application template XML file name is *PredictiveMaintenance.xml*:

  ```
  templateName="PredictiveMaintenance"
  ```

- This application has five tabs: Data, Global Selections, Define, Optimize, and Deploy:

  ```
  <DataStep stepIncluded="true"/>
  <GlobalSelectionStep stepIncluded="true"/>
  <DefineStep stepIncluded="true" enableInteractionPoints="false" xmlns:xsi="http://www.w3.org/2001/
  XMLSchema-instance" xsi:type="typeDecisionHierarchyDefineStep">
   <DimensionSetting name="Usecase">
                <SelectionSection enabled="true" enableModels="true"/>
                <AggregateRuleSection enabled="false"/>
                <PredictiveModelSection enabled="false"/>
                <AllocationRuleSection enabled="true"/>
                <PlanningSection enableInteractionPoints="false" enableStartEndDates="false"/>
   </DimensionSetting>
   <DimensionSetting name="ServiceGroup">
    <SelectionSection enabled="true" enableModels="true"/>
                <AggregateRuleSection enabled="false"/>
                <AllocationRuleSection enabled="false"/>
    <PredictiveModelSection enabled="false"/>
                <PlanningSection enableInteractionPoints="false" enableStartEndDates="false"/>
            </DimensionSetting>
  </DefineStep>
  <OptimizeStep stepIncluded="true" enableTest="true">
   <OptimizeMethod enableNumReturnsByIP="false" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="PrioritizationOptimization"/>
  </OptimizeStep>
  <DeployScoreStep stepIncluded="true">
             <ImmediateBatchScoring enableScoringOptions="true"/>
             <RealTimeScoring enableInteractiveQuestions="false"/>
  </DeployScoreStep>
  <ReportStep stepIncluded="false"/>
  ```

- This application's XML defines a Machine entity dimension and a top-level Usecase dimension with an
  Action child. However, a second top-level dimension called Service Group is also defined. The two
  top-level dimensions (Usecase and Service Group) appear as tabs on the Define tab of the application
  user interface. When users create new IBM Analytical Decision Management for Operations projects,
  they will be prompted to select whether to display one or both dimensions and in which order to
  display the tabs. Both tabs are displayed by default, with the Usecase tab appearing first

  Note that variable names are defined for the dimensions. These are hard-coded in the XML application
  template rather than defined in the screen text properties files.

  Applications can only have one level of children per dimension (on the Define tab, the user interface
  cannot display more than one level under each dimension).

  ```
  <EntityDimension name="Machine">
  .
  .
  .
  <Dimension name="Usecase">
  .
  .
  ```

```
.
<Dimension name="Action">
.
.
.
<Dimension name="ServiceGroup">
```

- The `Optimization` element defines the equation used to "value" each potential result by the optimization algorithm as it strives to find the solution with the minimum or maximum value. Supported algorithms include `CPLEX`, which indicates that the advanced IBM CPLEX form of optimization will be used, or `Heuristic`, which indicates that the "greedy" prioritization form of optimization will be used. You can also specify `None` to disable the use of optimization.

  The IBM Analytical Decision Management for Operations application uses CPLEX optimization, and points to the optimization model *PredictiveMaintenance.mod* that ships with IBM Analytical Decision Management. Do not modify this file. If you need to modify the file, contact an IBM SPSS representative.

  ```
  <Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path=
   "/Applications/PredictiveMaintenance/Optimization/PredictiveMaintenance.mod">
  ```

  The `Optimization` element contains the `ObjectiveFunction` element. If you want to customize the objective function (the optimization equation) in an application, contact your SPSS representative if you have questions.

  The optimization equation used in the sample IBM Analytical Decision Management for Operations application is displayed on the Optimize tab in the user interface. For more information, see the *IBM Analytical Decision Management for Operations User's Guide* or help.

  The Optimization element also contains two constraints (`ctTotalBudgetConstraint` and `ctNumberOfStaffConstraint`. These constraints are displayed on the Optimize tab of the application, allowing users to select which constraints apply to their current project.

IBM Analytical Decision Management for Demand Optimization template

IBM Analytical Decision Management for Demand Optimization is similar to the IBM Analytical Decision Management for Campaign Optimization application in that it helps decide which product to allocate to which store location, and delivers recommendations to store locations. By combining the logic of business rules with the insight gained through predictive modeling, the application identifies the most profitable decision for each store.

IBM Analytical Decision Management for Demand Optimization uses IBM CPLEX optimization and has the added value of maximizing profits and avoiding losses due to inventory shortfalls and restocking fees by allocating the right amount of inventory to each store, while keeping within constraints such as the available stock, minimum stock level to maintain, and minimum order size that can be sent to a store.

For more information, see the *IBM Analytical Decision Management for Demand Optimization User's Guide* or help.

The template for IBM Analytical Decision Management for Demand Optimization is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<PredictiveApplication xmlns="http://com.spss.pasw.dms/workspace" templateName="DemandOptimization" templateVersion="1"
appsVersion="17.0">
 <InterfaceControl>
  <InterfacePages>
   <ApplicationHome stepIncluded="true" showGallery="true"/>
   <DataStep stepIncluded="true"/>
   <GlobalSelectionStep stepIncluded="true"/>

   <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
   "typeDecisionHierarchyDefineStep">
    <DimensionSetting name="Inventory">
     <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
     <SelectionSection enabled="true" enableModels="true"/>
     <AllocationRuleSection enabled="false"/>
     <AggregateRuleSection enabled="false"/>
     <PredictiveModelSection enabled="false"/>
```

```
   </DimensionSetting>
  </DefineStep>
  <OptimizeStep stepIncluded="true">
   <OptimizeMethod enableNumReturnsByIP="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:type="PrioritizationOptimization"/>
  </OptimizeStep>
  <DeployScoreStep stepIncluded="true">
   <RealTimeScoring enableInteractiveQuestions="false"/>
   <ImmediateBatchScoring enableScoringOptions="true"/>
  </DeployScoreStep>
  <ReportStep stepIncluded="true"/>
 </InterfacePages>
 <InterfaceFeature id="Collaboration"/>
 <InterfaceFeature id="UploadDownload"/>
 <InterfaceFeature id="MetadataDownload"/>
</InterfaceControl>
<EntityDimension name="Order" description="OrderDescriptionText">
 <Variable name="StockRequested" dataType="integer" description="StockRequestedDescription" optimizationInputItem="true"
 prompt="StockRequested" simulateName="TotalStockRequested" simulateAction="sum">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
 <!-- Variable name="ExpectedSales" dataType="double" description="ExpectedSalesDescription" optimizationInputItem="true"
 prompt="ExpectedSales">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>1.0</Value>
  </ValueSource>
 </Variable -->
 <Variable name="RequestPriority" dataType="double" description="RequestPriorityDescription" optimizationInputItem="true"
 prompt="RequestPriority">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>1.0</Value>
  </ValueSource>
 </Variable>
 <Variable name="StoreID" dataType="string" description="StoreIDDescription" optimizationInputItem="true" prompt=
 "StoreID">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value></Value>
  </ValueSource>
 </Variable>
 <Variable name="MinStoreOrder" dataType="integer" description="MinStoreOrderDescription" optimizationInputItem="true"
 prompt="MinStoreOrder">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
</EntityDimension>

<Dimension name="Inventory" description="InventoryDescriptionText">
 <Variable name="UnitRevenue" dataType="double" description="UnitRevenueDescription" optimizationInputItem="true" prompt=
 "UnitRevenue">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
 <Variable name="StockAvailable" dataType="integer" description="StockAvailableDescription" optimizationInputItem="true"
 prompt="StockAvailable">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
 <Variable name="MinStockLevel" dataType="integer" description="MinStockLevelDescription" optimizationInputItem="true"
 prompt="MinStockLevel">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ConstantValueSource">
   <Value>0</Value>
  </ValueSource>
 </Variable>
</Dimension>

<Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path="/Applications/DemandOptimization/Optimization/
DemandOptimization.mod">
 <ObjectiveFunction Domain="double" Name="ExpectedRevenue" description="ExpectedRevenueDescription">
  <ExpressionFormat format="( ${UnitRevenue} * ${StockRequested} )"/>
  <ExternalUsage controlVariable="use_expected_revenue_function" variableType="int" enabledValue="1"
  disabledValue="0"/>
 </ObjectiveFunction>

 <ObjectiveFunction Domain="double" Name="ExpectedRevenueWithPreferred" description=
 "ExpectedRevenueWithPreferredDescription">
  <ExpressionFormat format="( ${UnitRevenue} * ${StockRequested} * ${RequestPriority} )"/>
  <ExternalUsage controlVariable="use_prioritize_preferred_function" variableType="int" enabledValue="1"
  disabledValue="0"/>
 </ObjectiveFunction>
```

```
    <Constraint type="greaterThanEqual" name="StockAvailableConstraint" entityScoped="false" description=
    "StockAvailableConstraintDescription">
     <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
     "StockAvailable"/>
     <ExpressionFormat format="${FN_sum} ( ${StockRequested} )"/>
     <ExternalUsage controlVariable="use_stock_available_constraint" variableType="int" enabledValue="1"
     disabledValue="0"/>
    </Constraint>

    <Constraint type="lessThanEqual" name="MinStockConstraint" entityScoped="false" description=
    "MinStockConstraintDescription">
     <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
     "MinStockLevel"/>
     <ExpressionFormat format="${StockAvailable} - ${FN_sum} ( ${StockRequested} )"/>
     <ExternalUsage controlVariable="use_min_stock_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
    </Constraint>

    <Constraint type="lessThanEqual" name="MinOrderSizeConstraint" entityScoped="false" description=
    "MinOrderSizeDescription">
     <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="VariableReferenceBoundary" value=
     "MinStoreOrder"/>
     <ExpressionFormat format="${FN_sum} ( ${StockRequested} ) / ${StoreID}"/>
     <ExternalUsage controlVariable="use_min_order_constraint" variableType="int" enabledValue="1" disabledValue="0"/>
    </Constraint>

    <OPLMapping tupleSetVariable="EntityAllocations" outputDecisionVariable="ObjectiveFunction">
     <EntityField referenceType="Variable" name="StockRequested" dataReturn="Value" tupleField=
     "StockRequested_Variable_Value" tupleFieldType="int"/>
     <!-- EntityField referenceType="Variable" name="ExpectedSales" dataReturn="Value" tupleField=
     "ExpectedSales_Variable_Value" tupleFieldType="float"/-->
     <EntityField referenceType="Variable" name="RequestPriority" dataReturn="Value" tupleField=
     "RequestPriority_Variable_Value" tupleFieldType="float"/>
     <EntityField referenceType="Variable" name="StoreID" dataReturn="Value" tupleField="StoreID_Variable_Value"
     tupleFieldType="string"/>
     <EntityField referenceType="Variable" name="MinStoreOrder" dataReturn="Value" tupleField=
     "MinStoreOrder_Variable_Value" tupleFieldType="int"/>

     <EntityField referenceType="System" name="entity" tupleField="entity" tupleFieldType="int"/>
     <EntityField referenceType="DimensionMember" name="Inventory" dataReturn="Value" tupleField=
     "Inventory_Allocation_Value" tupleFieldType="string"/>
     <EntityField referenceType="Variable" name="UnitRevenue" dataReturn="Value" tupleField=
     "UnitRevenue_Variable_Value" tupleFieldType="float"/>
     <EntityField referenceType="Variable" name="StockAvailable" dataReturn="Value" tupleField=
     "StockAvailable_Variable_Value" tupleFieldType="int"/>
     <EntityField referenceType="Variable" name="MinStockLevel" dataReturn="Value" tupleField=
     "MinStockLevel_Variable_Value" tupleFieldType="int"/>
     <EntityField referenceType="System" name="entity_allocation_id" tupleField="entity_allocation_id" tupleFieldType=
     "int"/>
     <!-- EntityField referenceType="System" name="priority" tupleField="priority" tupleFieldType="int"/-->
     <!-- EntityField referenceType="Objective" name="ObjectiveValue" dataReturn="Value" tupleField=
     "Expected_Profit_Value" tupleFieldType="float"/-->

     <OptimizationOutput name="StockAllocation-Value" valueVariable="StockAllocations" role="optimizedValue"
     variableType="int" selectionOnly="false" thresholdType="greaterThanEqual" thresholdValue="0"/>
     <OptimizationOutput name="ObjectiveValue-Value" valueVariable="Contribution" role="objectiveValue" variableType=
     "float" selectionOnly="false" />
    </OPLMapping>
   </Optimization>
   <Deployment>
    <!-- OutputAttribute referenceType="DimensionMember" name="Inventory" returnValue="Inventory.Allocation-Value">Inventory
    </OutputAttribute-->
    <OutputAttribute referenceType="Variable" name="StoreID" returnValue="StoreID.Variable-Value">Output-StoreID
    </OutputAttribute>
    <OutputAttribute referenceType="Variable" name="StockRequested" returnValue="StockRequested.Variable-Value">
    Output-StockRequested</OutputAttribute>
    <OutputAttribute referenceType="Objective" name="StockAllocated" returnValue="StockAllocation-Value">
    Output-StockAllocated</OutputAttribute>
    <OutputAttribute referenceType="Variable" name="UnitRevenue" returnValue="UnitRevenue.Variable-Value">
    Output-UnitRevenue</OutputAttribute>
    <OutputAttribute referenceType="Objective" name="ObjectiveValue" returnValue="ObjectiveValue-Value">
    Output-ObjectiveValue</OutputAttribute>
   </Deployment>
</PredictiveApplication>
```

- In this example, the application template XML file name is *DemandOptimization.xml*:

```
   templateName="DemandOptimization"
```

- This application has six tabs: Data, Global Selections, Define, Optimize, Deploy, and Reports:

```
       <DataStep stepIncluded="true"/>
       <GlobalSelectionStep stepIncluded="true"/>

       <DefineStep stepIncluded="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="typeDecisionHierarchyDefineStep">
```

```
 <DimensionSetting name="Inventory">
  <PlanningSection enableInteractionPoints="true" enableStartEndDates="true"/>
  <SelectionSection enabled="true" enableModels="true"/>
  <AllocationRuleSection enabled="false"/>
  <AggregateRuleSection enabled="false"/>
  <PredictiveModelSection enabled="false"/>
 </DimensionSetting>
</DefineStep>
<OptimizeStep stepIncluded="true">
 <OptimizeMethod enableNumReturnsByIP="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="PrioritizationOptimization"/>
</OptimizeStep>
<DeployScoreStep stepIncluded="true">
 <RealTimeScoring enableInteractiveQuestions="false"/>
 <ImmediateBatchScoring enableScoringOptions="true"/>
</DeployScoreStep>
<ReportStep stepIncluded="true"/>
```

- This application's XML defines an `Order` entity dimension and a single `Inventory` dimension with no children.

  Note that variable names are defined for the dimensions. These are hard-coded in the XML application template rather than defined in the screen text properties files.

  Applications can only have one level of children per dimension (on the Define tab, the user interface cannot display more than one level under each dimension).

```
<EntityDimension name="Order" description="OrderDescriptionText">
.
.
.
<Dimension name="Inventory" description="InventoryDescriptionText">
```

- The `Optimization` element defines the equation used to "value" each potential result by the optimization algorithm as it strives to find the solution with the minimum or maximum value. Supported algorithms include `CPLEX`, which indicates that the advanced IBM CPLEX form of optimization will be used, or `Heuristic`, which indicates that the "greedy" prioritization form of optimization will be used. You can also specify `None` to disable the use of optimization.

  The IBM Analytical Decision Management for Demand Optimization application uses CPLEX optimization, and points to the optimization model *DemandOptimization.mod* that ships with IBM Analytical Decision Management. Do not modify this file. If you need to modify the file, contact an IBM SPSS representative.

```
<Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path=
 "/Applications/DemandOptimization/Optimization/DemandOptimization.mod">
```

  The `Optimization` element contains the `ObjectiveFunction` elements. If you want to customize the objective functions (the optimization equation) available for use in an application, contact your SPSS representative if you have questions.

  The two optimization equations used in the sample IBM Analytical Decision Management for Demand Optimization application are displayed on the Optimize tab in the user interface. You can choose between an *Expected revenue per item * items supplied* equation or a *Expected revenue per item * items supplied * the bias for the distributor* equation. For more information, see the *IBM Analytical Decision Management for Demand Optimization User's Guide* or help.

  The Optimization element also contains three constraints (`StockAvailableConstraint`, `MinStockConstraint`, and `MinOrderSizeConstraint`. These constraints are displayed on the Optimize tab of the application, allowing users to select which constraints apply to their current project.

Chapter 3. Customizing the user interface

File locations

IBM Analytical Decision Management provides a framework for customizing the appearance of your applications. You can modify various files to customize the look and feel and the text displayed in the user interface. The process for modifying these files is described in this chapter. We recommend using a simple text editor such as Microsoft Notepad to edit all *.css* and *.properties* files.

The default user interface files for IBM Analytical Decision Management are installed in the IBM SPSS Collaboration and Deployment Services installation directory, for example *C:\Program Files\IBM\SPSS\Deployment\7.0\components\decision-management\Default*. To customize all applications, modify files in the *Default* directory. To customize one application and leave all others with the default settings, copy **only** the required directories and files from *Default* to a new application directory. For example, if you only want to customize CSS and coach text for a certain application, you only need to copy those folders – and you only need to copy the specific files and settings you want to customize. Settings in your application directory override those in *Default*.

Important: Before you begin, we recommend making a backup copy of the entire *decision-management* directory.

General steps for customizing an application

After creating an application template as described in the previous chapter, follow these general steps to customize the look and feel of your application. The remaining sections in this chapter provide complete details for these general steps.

Note that the default installed files may be overwritten as subsequent Fix Packs and other installations are applied. As a precaution, we recommend backing up any custom application templates and customized application files.

1. In the *Applications* directory, create a new folder for your application (for example, *YourApp)* as discussed in the previous chapter.
2. Copy folders and only the files you plan to customize from an existing application and paste into your new application's directory. Choose an existing application that most closely resembles the application you want to create. For example, if you installed the prebuilt IBM Analytical Decision Management for Claims application, you can copy files from the *ClaimsManagement* directory into your new *YourApp* directory.
3. Open the *Default* directory and copy any other elements you want to customize, and paste them into your application's directory. For example, if you want to create custom terminology, copy the *Terminology* directory.

   Your application folder only needs to contain the files you want to customize. All other files will be read from *Default*. Even within each file, you only need to include the portions you want to customize. For example, if customizing screen text, *ScreenText_en.properties* only needs to contain the text you are customizing. All other text will be read from *Default*. In other words, any item or file not explicitly defined or contained within your application directory will be handled by the *Default* directory.
4. Copy the *appGroup.xml* file and *description.xml* file from an existing application and paste it into your new application's folder (for example, into *YourApp*).

   Edit the *appGroup.xml* file to reference the folders containing any files you plan to customize in your application's folder (see "Configuring applications" on page 5 for details). Edit the *description.xml* file to specify the text used in your application's shortcut box on the *Applications launch page* (see "Configuring applications" on page 5 for details).

5. Use the instructions in this chapter to customize your application's user interface.

Tip: When customizing files on the machine where IBM SPSS Collaboration and Deployment Services is installed, you may be able to share the *decision-management* directory to edit the files from any other machine on your network. Refer to your operating system documentation or see your network administrator for details about sharing directories and files.

Note that you cannot customize the Login screen.

User interface text

You can customize all text displayed in your application to tailor the product for your company's needs. Such text includes:

Coach text: Coach text is brief, helpful, easily customizable text that appears throughout the user interface when you click the small coach text icon:

Coach text is separate from the full, browsable help system. The full help system is available by clicking **Help** at the top of the screen or clicking the help icon in dialogs:

The help system contains general information about the user interface and is not customizable, whereas the coach text can be personalized to be very specific to your application, industry, or company. Coach text is stored in *CoachText_en.properties* files, where *en* is the language.

Messages: Message text consists of all messages in the application, including errors, warnings, and information messages. Messages are stored in *Messages_en.properties* files, where *en* is the language.

Screen text: Screen text is text used on user interface elements such as tabs, buttons, dialog titles, and field labels. Screen text is stored in *ScreenText_en.properties* files, where *en* is the language. Note that the text used in shortcut boxes on the *Applications launch page* is handled in a special way with a *description.xml* file (see "Configuring applications" on page 5 for details).

Terminology: Terminology refers to a custom list of terms used in your application. You can use terminology to replace certain terms such as *model* or *score* with terms more familiar to your users. Each term has a unique ID that can be referenced from other areas of the application such as coach text, message text, and screen text. Terminology definitions are stored in *Terminology_en.properties* files, where *en* is the language.

Language support

User interface text is stored in *.properties* files in the IBM SPSS Collaboration and Deployment Services installation directory (for example, *C:\Program Files\IBM\SPSS\Deployment\7.0\components\decision-management\Default\ScreenText*).

The *.properties* files shipped with IBM Analytical Decision Management are provided in multiple languages. Each file name includes the language code according to W3C definition standards. If you need another language, you can create your own *.properties* files with the proper language code and translate the content from the shipped files. Make sure you save the files in ASCII format. The following table lists the languages currently provided with IBM Analytical Decision Management. Note that this list is subject to change from release to release, and the documentation may not be available in these languages.

Each user can easily change the language of his or her application without having to restart any servers. For example, German-speaking users, English-speaking users, and Japanese-speaking users can all be using the same application in their own language. From the Tools menu in your browser, go to Internet Options or Options, switch to the desired language, and then refresh the screen.

Table 6. Example language codes.

Language	Language code	Example file name
English	en	*CoachText_en.properties*
German	de	*CoachText_de.properties*
Spanish	es	*CoachText_es.properties*
French	fr	*CoachText_fr.properties*
Italian	it	*CoachText_it.properties*
Japanese	ja	*CoachText_ja.properties*
Korean	ko	*CoachText_ko.properties*
Portuguese (Brazilian)	pt_BR	*CoachText_pt_BR.properties*
Chinese (China/simplified)	zh_CN	*CoachText_zh_CN.properties*
Chinese (Taiwan/traditional)	zh_TW	*CoachText_zh_TW.properties*

The following sections include the syntax of each *.properties* file you can customize, including examples. As a precaution, save a copy of all original files before proceeding (we recommend saving a copy of the entire *decision-management* directory). This allows you to revert to the original files in the future, if necessary.

Note that each application directory includes a *description.xml* file that defines the text used in shortcut boxes on the *Applications launch page*. This file works differently than properties files, in that content for all languages is included within the file. For example, the English section for the IBM Analytical Decision Management for Claims is defined as follows:

```
<en>
    <TitleEntry>IBM® Analytical Decision Management for Claims</TitleEntry>
    <ShortDescription>Intelligent risk management in real time</ShortDescription>
    <LongDescription>Assess the overall risk level for incoming claims and recommend the specific action
     to take.</LongDescription>
</en>
```

Sections are included for several different languages. To add another language, simply create a new section (for example, <nl> for Dutch) and customize the text to meet your needs.

Coach text

Certain areas of the user interface provide coach text. Each area has a unique name identifying it in the coach text properties file. In general, entries in the coach text properties file are listed in the order they appear in the user interface. Follow the instructions in this section to customize the coach text in all applications or in one specific application. See "File locations" on page 43 for details about file locations.

Important: Do not modify any IDs in the coach text properties files. Only modify text after the equals sign (=). For example, in `HOME_DATASTEP_TITLE=Data`, `HOME_DATASTEP_TITLE` is the ID and should never be modified.

To customize coach text across all applications

1. Open the file *Default\CoachText\CoachText_en.properties*, where *en* is the desired language.
2. Make the desired changes, save, and close. Note that any customizations made to coach text in an application override the default coach text.

To customize coach text in one application

1. Open the file *Applications\YourApp\CoachText\CoachText_en.properties*, where *YourApp* is the name of your custom application's folder and *en* is the desired language. If this file does not yet exist, copy it from *Default/CoachText* and remove all sections from it except for those you plan to customize.

2. Make the desired changes, save, and close. The file should only contain the sections you customize.

Examples

1. To change the special "hover" style coach text used for the Data step icon on the Home page, modify the following lines in the coach text properties file. Modify the text after `HOME_DATASTEP_TITLE=` to change the title of the coach text window, or modify the text after `HOME_DATASTEP_CONTENT=` to change the main text displayed in the coach text window.

```
HOME_DATASTEP_TITLE=Data
HOME_DATASTEP_CONTENT=Select data for modeling, simulation, or operational
use.<br/><br/>This may include information about current customers and results
from previous claims.
```

2. You can include simple HTML elements in coach text such as links, bold and italic text, etc. For example, to include an e-mail link so users can easily contact someone in your organization, include something like the following in the coach text properties file. The link in this example opens a new, blank e-mail message when the user clicks it from coach text on the Data tab.

```
DATA_SOURCE_MY_TITLE=My Data Source
DATA_SOURCE_MY_CONTENT=Contains any data sources that you have created for your
own use, either in this or other projects.<br/><br/>If you have questions about
data, contact <a href="mailto:bobjones@yourcompany.com">Bob Jones</a>.
```
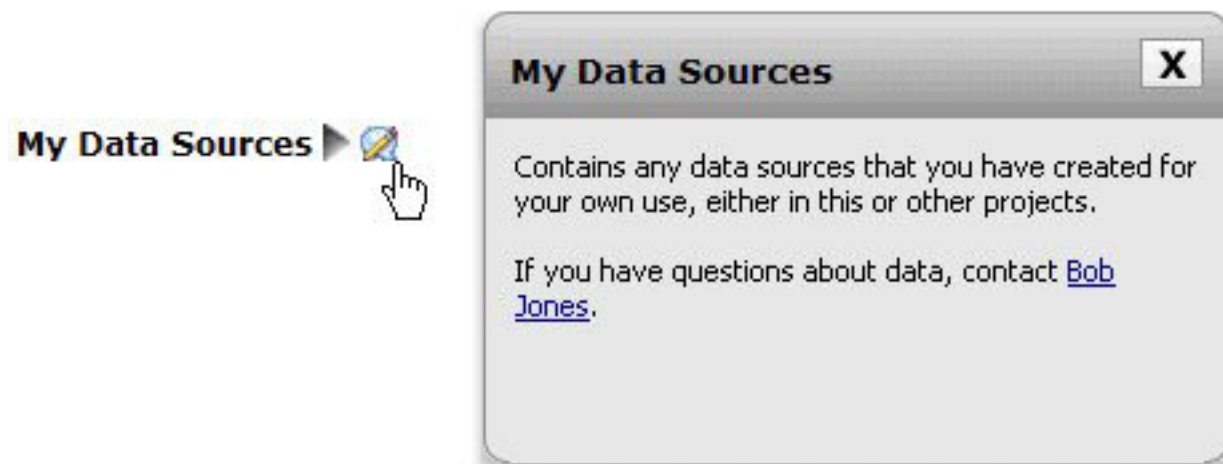

Message text

You can customize the text used in warning, error, and informational messages. Follow the instructions in this section to customize the message text in all applications or in one specific application. See "File locations" on page 43 for details about file locations.

Important: Do not modify any IDs in the message properties files. Only modify text after the equals sign (=). For example, in `LOGIN_FAILED=Login unsuccessful`, `LOGIN_FAILED` is the ID and should never be modified.

To customize messages across all applications

1. Open the file *Default\Message\Message_en.properties*, where *en* is the desired language.

2. Make the desired changes, save, and close. Note that any customizations made to messages in an application override the default messages.

To customize messages in one application

1. Open the file *Applications\YourApp\Message\Message_en.properties*, where *YourApp* is the name of your custom application's directory and *en* is the desired language. If this file does not yet exist, copy it from *Default/Message* and remove all sections from it except for those you plan to customize.
2. Make the desired changes, save, and close.

Example

To change the error message displayed when a user fails to log in to IBM Analytical Decision Management, modify the following line in the appropriate message properties file. Only modify the text after `LOGIN_FAILED=`.

For example, you may want to include the name of an administrator users should contact at your company for certain issues.

```
DELETE_RULE=Are you sure you want to delete this rule?<br/><br/>If you have questions about rules,
see our rules administrator <a href="mailto:janedoe@yourcompany.com">Jane Doe</a> in office L318.
```


Screen text

Text such as dialog headings, field labels, tab labels, and button labels can be customized. Follow the instructions in this section to customize screen text in all applications or in one specific application. See "File locations" on page 43 for details about file locations.

Important: Do not modify any IDs in the screen text properties files. Only modify text after the equals sign (=). For example, in `TOOLTIP_NEW_MODEL=New model`, `TOOLTIP_NEW_MODEL` is the ID and should never be modified.

To customize screen text across all applications

1. Open the file *Default\ScreenText\ScreenText_en.properties*, where *en* is the desired language.
2. Make the desired changes, save, and close. Note that any customizations made to screen text in an application override the default screen text.

To customize screen text in one application

1. Open the file *Applications\YourApp\ScreenText\ScreenText_en.properties*, where *YourApp* is the name of your custom application's directory and *en* is the desired language. If this file does not yet exist, copy it from *Default/ScreenText* and remove all sections from it except for those you plan to customize.
2. Make the desired changes, save, and close.

Example

To change the name of the main section headings on the Data tab, modify the following two values in the appropriate screen text properties file.

```
##################################################
# Screen text used on Data tab
##################################################
Project_Data_Model=Project Data Model
Data_Sources=Data Sources
```

Terminology

You can customize terminology used in your applications to fit your business needs. You can also add new terms. Each term has a unique ID that can be referenced in other areas of the application such as coach text windows, messages, and screen text. Follow the instructions in this section to create custom terms in all applications or one specific application. See "File locations" on page 43 for details about file locations.

To add custom terminology across all applications

1. Open the file *Default\Terminology\Terminology_en.properties*, where *en* is the desired language.
2. Modify or add one or more terminology definitions. Each term must have a unique ID.
3. Insert the appropriate terminology tag into the desired coach text, screen text, or messages *.properties* files. Use the syntax `<#:DataStep>`, where `DataStep` is the ID from the terminology properties file.

To add custom terminology to one application

1. Open the file *Applications\YourApp\Terminology\Terminology_en.properties*, where *YourApp* is the name of your custom application's directory and *en* is the desired language. If this file does not yet exist, copy it from *Default/Terminology* and remove all sections from it except for those you plan to customize.
2. Modify or add one or more terminology definitions.
3. Insert the appropriate terminology tag into the desired coach text, screen text, or messages *.properties* files located in your application's directory (for example, *Applications\YourApp*).

Example

The prebuilt applications included with the product use terminology variables for all tab names in the user interface. For example, the IBM Analytical Decision Management for Claims application defines the following terminology variables in its *Terminology_en.properties* file:

```
##################################################
#  Variables for names of tabs in UI
##################################################
ApplicationHome=Home
DataStep=Data
GlobalSelectionStep=Global Selections
DefineStep=Define
CombineStep=Combine
DeployScoreStep=Deploy
ReportStep=Reports
```

And then the IBM Analytical Decision Management for Claims application uses those terminology variables in its *ScreenText_en.properties* file:

```
##################################################
# Name of each step/tab in the UI. These may differ per application type
##################################################
ApplicationHome=<#:ApplicationHome>
DataStep=<#:DataStep>
GlobalSelectionStep=<#:GlobalSelectionStep>
DefineStep=<#:DefineStep>
CombineOptimizeStep=<#:CombineOptimizeStep>
DeployScoreStep=<#:DeployScoreStep>
ReportStep=<#:ReportStep>
```

To change the name of the *Combine* tab to something like *Prioritize*, for example, simply change `CombineStep=Combine` to `CombineStep=Prioritize` in the *Terminology_en.properties* file. The new tab name will then be used in every spot where the terminology variable is inserted (for example, it is also used in the coach text for the IBM Analytical Decision Management for Claims application, as shown below).

```
#################################################
# Following entries are for hover-style coach text on Home screen
#################################################

HOME_DATASTEP_TITLE=<#:DataStep>
HOME_DATASTEP_CONTENT=Select data for modeling, simulation, or operational
use.<br/><br/>This may include information about current customers and
records from previous claims.

HOME_GLOBALSELECTIONSTEP_TITLE=<#:GlobalSelectionStep>
HOME_GLOBALSELECTIONSTEP_CONTENT=Choose claims you want to include or exclude
from the application.<br/><br/>For example, all claims relating to floods or
windshield damage may be excluded due to special handling requirements.

HOME_DEFINESTEP_TITLE=<#:DefineStep>
HOME_DEFINESTEP_CONTENT=Define the types of claims and possible actions along
with the rules and models for determining the likelihood of fraud.

HOME_COMBINEOPTIMIZESTEP_TITLE=<#:CombineStep>
HOME_COMBINEOPTIMIZESTEP_CONTENT=Specify how rules and models are combined to
determine the recommended action for each claim.

HOME_DEPLOYSCORESTEP_TITLE=<#:DeployScoreStep>
HOME_DEPLOYSCORESTEP_CONTENT=Validate your current configuration and mark it
ready to be deployed.
```

Look and feel

You can change the appearance of your applications by modifying graphics files and cascading style sheets (*.css*). Experience with graphics and style sheets is recommended for modifying elements such as:

- Colors
- Borders
- Background properties
- Size and position of elements
- Margins and padding
- Fonts and text properties

Note that styles cascade. The default styles and images are applied unless any are customized in your application, in which case they override the default. For example, if the only portion of your application you want to customize is the font, you can either change the values in the following section of the default CSS (*layout.css*) to change the font for all applications, or copy the section from *layout.css* to the custom CSS for your application (*branded.css*) to only change the font there. In the latter scenario, the *branded.css* file for your application would only need to contain the following section, and all other styles and settings will be applied from the default styles.

```
body,table td,select,pre,.gwt-Button {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    font-size: 12px;
}
```

The following figure illustrates the process of customizing a single application, while leaving the default alone. This is the recommended method. To customize graphics, copy the graphics you want to customize from the default images folders to the images folder in your application directory, then modify them (or create new images files in your application directory of the correct file name, size, and folder location). To customize *.css* settings, open any or all of the default *.css* files and copy the sections you want to customize into your application's *branded.css* file.
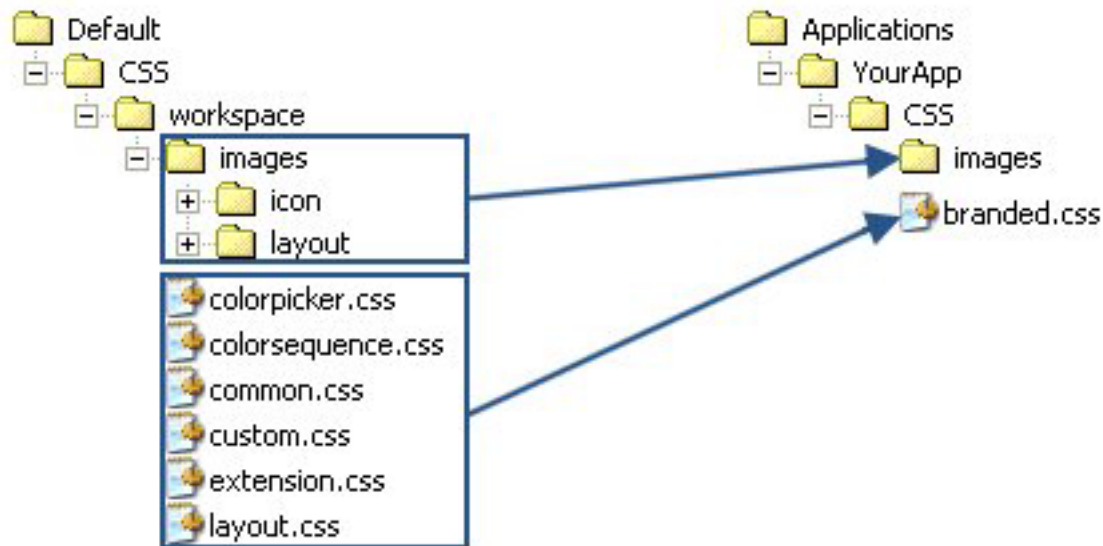
Figure 4. Copying graphics and CSS settings from default

The same general process should be used for customizing user interface text.

Customizing style sheets and graphics

Graphics and style sheets are stored in the IBM SPSS Collaboration and Deployment Services directory (for example, *C:\Program Files\IBM\SPSS\Deployment\7.0\components\decision-management\Default\CSS*). Use the following steps to modify the style sheets and/or graphics for your custom application.

To customize CSS and graphics across all applications

1. In the *Default\CSS* folder, modify settings in one or more CSS files.
2. In the *Default\CSS\images* folder, edit any graphics files (*.gif*) in graphics software of your choice to make any desired changes, or replace the graphics with your own. Note that if you change the dimensions of a graphic, you also need to search the CSS for the graphic file name and update the defined pixel dimensions in order for the graphic to be displayed properly in the application user interface.

To customize CSS and graphics in one application

1. In the *CSS* folder of the application you want to customize (for example, *Applications\YourApp\CSS*), open the file *branded.css*.
2. Copy any sections you want to change from the default CSS files (*common.css, custom.css, extension.css, layout.css*) into *branded.css*.

   Note that you can only customize *colorpicker.css* across all applications (you cannot have unique settings per application). The file defines which colors are available for labels in the user interface.
3. Modify sections in *branded.css* as desired, save, and close.
4. Open your application's *appGroup.xml* file and set the `<CssFileSpec>` value to your application's *branded.css* stylesheet. Without this setting, your application would apply all default styles. For example:

   `<CssFileSpec>/Applications/YourApp/CSS/branded.css</CssFileSpec>`

Note: Changes aren't always reflected in the user interface immediately. You may need to wait a few minutes and refresh your web browser.

Examples
To change the graphics used on the applications launch page

On the *Applications launch page,* a graphic is used in the shortcut box for each application and a small icon version is used beside each application name in list view.

The process for customizing these graphics is a bit different than for other graphics. You must modify settings in the default *custom.css* file (*Default\CSS\workspace\custom.css*). The shortcut graphics for each prebuilt application are defined in the CSS as follows:

```
.launcher_customerinteractionmanagement_icon {
 width: 82px;
 height: 100px;
 background: url("images/icon/LAUNCHER_customer_interactions.png")
  no-repeat scroll 0 0;
}

.launcher_claimsmanagement_icon {
 width: 82px;
 height: 100px;
 background: url("images/icon/LAUNCHER_claims_processing.png") no-repeat
  scroll 0 0;
}

.launcher_rulesmanagement_icon {
 width: 82px;
 height: 100px;
 background: url("images/icon/LAUNCHER_rules_management.png") no-repeat
  scroll 0 0;
}

.launcher_modeleradvantage_icon {
 width: 82px;
 height: 100px;
 background: url("images/icon/LAUNCHER_modeler_advantage.png") no-repeat
  scroll 0 0;
}

.launcher_campaignoptimization_icon {
 width: 82px;
 height: 100px;
 background: url("images/icon/LAUNCHER_campaign_optimization.png")
  no-repeat scroll 0 0;
}

.launcher_predictivemaintenance_icon {
 width: 82px;
 height: 100px;
 background: url("images/icon/LAUNCHER_predictive_maintainance.png")
  no-repeat scroll 0 0;
}

.launcher_demandoptimization_icon {
 width: 82px;
 height: 100px;
 background: url("images/icon/LAUNCHER_campaign_optimization.png")
  no-repeat scroll 0 0;
}

.launcher_YourApp_icon {
    width: 82px;
```

```
     height: 100px;
     background: url("images/icon/LAUNCHER_YourApp.gif") no-repeat
         scroll 0 0;
}
```

And the small graphics for the list view are defined as follows:

```
.launcher_list_customerinteractionmanagement_icon {
 width: 18px;
 height: 15px;
 background: transparent
  url("images/icon/LAUNCHER_list_customer_interactions.gif") no-repeat;
}

.launcher_list_claimsmanagement_icon {
 width: 18px;
 height: 18px;
 background: transparent
  url("images/icon/LAUNCHER_list_claims_processing.gif") no-repeat;
}

.launcher_list_rulesmanagement_icon {
 width: 15px;
 height: 16px;
 background: transparent
  url("images/icon/LAUNCHER_list_rules_management.gif") no-repeat;
}

.launcher_list_modeleradvantage_icon {
 width: 16px;
 height: 16px;
 background: transparent
  url("images/icon/LAUNCHER_list_modeler_advantage.gif") no-repeat;
}

.launcher_list_campaignoptimization_icon {
 width: 18px;
 height: 15px;
 background: transparent
  url("images/icon/LAUNCHER_list_campaign_optimization.gif") no-repeat;
}

.launcher_list_demandoptimization_icon {
 width: 19px;
 height: 16px;
 background: transparent
  url("images/icon/LAUNCHER_list_campaign_optimization.gif") no-repeat;
}

.launcher_list_predictivemaintenance_icon {
 width: 16px;
 height: 16px;
 background: transparent
  url("images/icon/predictivemaintenance16x16.png") no-repeat;
}

.launcher_list_YourApp_icon {
     width: 18px;
     height: 18px;
     background: transparent
         url("images/icon/LAUNCHER_list_YourApp.gif") no-repeat;
}
```

- If you want to customize the graphic for an existing prebuilt application, simply modify the graphics defined in the CSS (for example, to customize the IBM SPSS Modeler Advantage graphics, edit *LAUNCHER_modeler_advantage.gif* and *LAUNCHER_list_modeler_advantage.gif*).
- If you want to create new graphics for a new application, add your graphics to the default icon folder and then add new CSS settings. For example, if your XML application template file name is *YourApp.xml*, you would add the appropriate CSS settings to *custom.css* as shown above. Be sure the width and height definitions are correct for your new graphics.

To change the colors available in the color picker

To customize the colors available in the color picker, modify the file *Default\CSS\workspace\colorpicker.css*. The color picker is used for label colors on the Deploy tab and the matrix on the Combine tab of certain applications. You can change existing color values or add new colors as you like. Note that you can only customize these colors in the default CSS for all applications – you cannot customize them per application.

You can use the name of a color or the hex value (for example, red is #FF0000).

Figure 5. Color picker

```
.colorpicker0
{
  background-color: White;
}
.colorpicker1
{
  background-color: red;
}
.colorpicker2
{
  background-color: Pink;
}
.colorpicker3
{
  background-color: Silver;
}
.colorpicker4
{
  background-color: Orange;
}
.colorpicker5
{
  background-color: Yellow;
}
.

.

.
```

Chapter 4. Scoring Service configuration

Scoring includes the process of generating real-time values by supplying predictive models with input data. In general, to use a model for generating scores:

1. Select a model from the IBM SPSS Collaboration and Deployment Services Repository to use for scoring.
2. In IBM SPSS Collaboration and Deployment Services Deployment Manager, define a scoring configuration for the model.
3. Supply the configured model with data and generate scores.

Scoring is a component of IBM SPSS Collaboration and Deployment Services. This chapter provides brief information specific to IBM Analytical Decision Management regarding the IBM SPSS Collaboration and Deployment Services Scoring Service.

For complete information about scoring, see the *Scoring* chapter of the *IBM SPSS Collaboration and Deployment Services Deployment Manager User's Guide*. The guide is included on the IBM SPSS Collaboration and Deployment Services installation disc and installed with IBM SPSS Collaboration and Deployment Services.

IBM Analytical Decision Management and the Scoring Service

The general process for scoring IBM Analytical Decision Management applications is as follows:

- An IBM SPSS Modeler stream (*.str* file) is automatically created in the repository when a user saves a IBM Analytical Decision Management   project.
- The IBM SPSS Modeler stream can then be used with the Scoring Service. In IBM SPSS Collaboration and Deployment Services Deployment Manager, create a scoring configuration. When creating the scoring configuration, some IBM Analytical Decision Management-specific dialogs must be completed to enable interactive scoring, select interaction points (if configured), and set global variables (such as *Max Offers*, for example).

Creating a Scoring Service Configuration

You must use IBM SPSS Collaboration and Deployment Services Deployment Manager to create a scoring configuration. For complete instructions, see the help in IBM SPSS Collaboration and Deployment Services Deployment Manager. Keep the following IBM Analytical Decision Management-specific points in mind when creating scoring configurations using the IBM SPSS Collaboration and Deployment Services Deployment Manager *Configure Scoring Model* dialog box. Each section corresponds to a section in the dialog box.

Model Specific Settings

Enable Interactive Scoring. If your application (stream) supports interactive scoring, you can select this option to choose whether interactive scoring is enabled for the scoring configuration. If enabled, and if the Scoring Service does not have all the inputs needed, a *MissingDataException* will be returned that identifies the missing data (field), and the interactive question that can be used to prompt for the needed values. The caller can then prompt for the missing data and call the Scoring Service again (passing all data). Interactive scoring is configured on the Deploy tab in IBM Analytical Decision Management applications.

- While *MissingDataException* can identify multiple pieces of missing data, it does not necessarily identify all missing data. *MissingDataException* will communicate which data is missing at the current stage of processing.

- If **Enable Interactive Scoring** is not enabled, you get a *MissingDataException* without any interactive questions.

Select Interaction Point. If your application uses multiple interaction points, you can select which interaction point the configuration should use in the *Model Specific Settings* dialog box. Interaction points specify where an item such as a campaign or offer applies. Options might include a call center, web site, ATM, or in-store location. Administrators can predefine the interaction points available for selection by business users. The interaction points defined are displayed to business users in the IBM Analytical Decision Management applications. You can create multiple scoring configurations in IBM SPSS Collaboration and Deployment Services Deployment Manager--one for each interaction point.

Max Offer. Note that *Max Offer* is an example of a field that might be displayed in the IBM SPSS Collaboration and Deployment Services Deployment Manager dialog. Variables and constant values that are defined on the entity dimension can be overridden for a specific score configuration.

Configuring Model Specific Settings in IBM Analytical Decision Management

The model specific settings displayed in the scoring configuration are defined in IBM Analytical Decision Management. Interactive scoring is performed on a per-field basis, and is configured on the Deploy tab in the IBM Analytical Decision Management application.

The available interaction points, if any, are configured for each application by an IBM Analytical Decision Management administrator.

Choosing Model Outputs

In the scoring configuration, you can choose which model outputs to include with the results. The fields that can be output are those specified on the Deploy tab in the IBM Analytical Decision Management user interface.

Advanced Settings

Under Advanced Settings, you can specify options for batch scoring, caching, and logging. See the IBM SPSS Collaboration and Deployment Services documentation for more information.

Response Service

The Response Service supplements the Scoring Service. It's a web service allowing client applications such as call center interfaces to send responses to the service to be logged. For example, a bank might have a call center interface that presents specific offers to the call center agent. The agent can then make the appropriate offer to the bank customer, and the customer's answer (response) is sent to the Response Service and logged. The following figure presents the flow of a complete example.
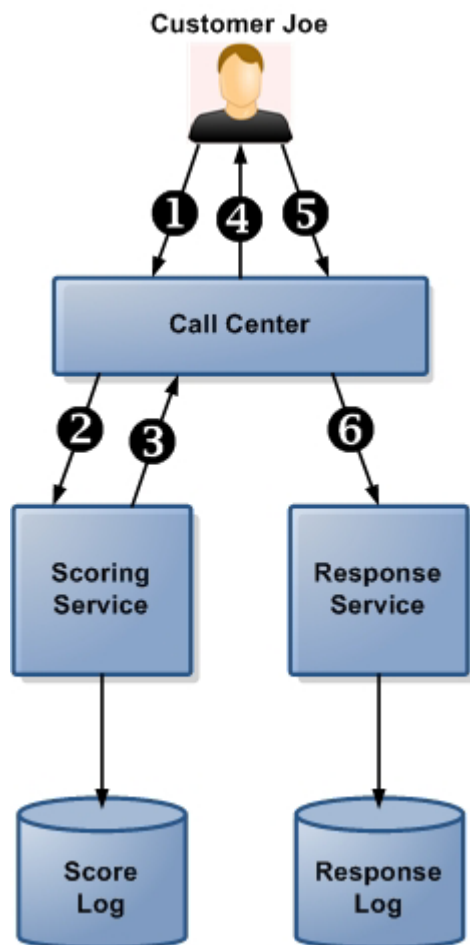
Figure 6. Example use of Scoring Service and Response Service

Table 7. Scoring Service and Response Service example.

Figure label	Description
❶	Customer Joe calls.
❷	The call center sends Joe's customer ID to the Scoring Service. If logging is turned on in the scoring configuration (optional), this information is sent to the score log. Note that score logging is distinct from response logging.
❸	The Scoring Service determines the best offer for customer Joe (*Gold Card*, for example) and sends the offer back to the call center. This information is also written to the score log (if enabled); Views and Queries can be written against the score log.
❹	The call center operator presents the *Gold Card* offer to customer Joe.
❺	Joe says yes to the offer.
❻	The call center sends Joe's "yes" response to the Response Service and this response is logged. Queries can be written against the response log, or against both logs.

For more information about the Response Service and hooking it up to your front-office application, contact your SPSS representative.

Chapter 5. Using rules from IBM Operational Decision Management

Rules created in a Business Rule Management System such as IBM Operational Decision Management can be referenced and used in IBM Analytical Decision Management applications as follows:

1. The administrator must download a *.ZIP* file containing the metadata pertinent for rule creation. This metadata can be downloaded from a particular project in IBM Analytical Decision Management, or from a particular analytic data view's logical model in IBM SPSS Collaboration and Deployment Services Deployment Manager. The downloaded *.ZIP* file contains an XML schema that describes the data model.

2. An expert user edits the XML schema(s) so the request and response types are tailored for the rules they wish to create.

3. An expert user reads the XML schema(s) into an external tool such as IBM Operational Decision Management Rule Designer, uses it to create rules for use with the IBM Analytical Decision Management project, and deploys each rule as a Web service.

4. In IBM SPSS Collaboration and Deployment Services Deployment Manager, the administrator creates an external rule reference for each rule to be used in IBM Analytical Decision Management. Each rule reference is saved as an object in IBM SPSS Collaboration and Deployment Services Repository.

5. Referenced rules are accessible in the IBM Analytical Decision Management application, where they can be browsed and used in the same manner as other models and rules. For more information, see the *User's Guide* for the IBM Analytical Decision Management application.

When users choose to reference external rules in IBM Analytical Decision Management, IBM Analytical Decision Management will check whether the external rule's input parameter has a flat or hierarchical data model. If the external rule's input parameter data model is flat, then IBM Analytical Decision Management will check whether it is compatible with the Project Data Model. External rules with hierarchical input parameters may only be referenced in IBM Analytical Decision Management if the Project Data Model is a Data View source and if the use of external rules based on hierarchical data is allowed in the project.

Downloading metadata

Rules created in a Business Rules Management System such as IBM Operational Decision Management can be referenced and used in IBM Analytical Decision Management applications, provided they have been developed to support the same data model used in the current IBM Analytical Decision Management project, and can be deployed as a Web service for use by the application. To accomplish this, an IBM Analytical Decision Management administrator must download the relevant metadata. There are two ways to download the metadata, and the complexity of the data model and the potential resultant rule complexity differ accordingly:

Downloading project metadata from IBM Analytical Decision Management

An administrator can click the **Download metadata** icon in IBM Analytical Decision Management to download a *.ZIP* file containing metadata for the current project. The downloaded file is saved locally on the system used by the administrator to trigger the download.

Figure 7. Download metadata icon

The metadata downloaded this way supports the creation of rules based on the project data model. The project data model will be flat (not hierarchical), so any rules authored based on this metadata can only utilize a flat data model. A rule authored based on this flat data model may be referenced in an IBM Analytical Decision Management project no matter the type of the project data model data source.

The following files are included in the *.ZIP* file:
- **XML schema definition (*.XSD).** Contains definitions of the field types in the data model for the current IBM Analytical Decision Management project. This file can be imported into an external development tool such as IBM Operational Decision Management Rule Designer in order to develop rules for use with the current project.
- **Web Service Description Language template (*.WSDL).** Included to assist in development of Web services that use the data model. The *.WSDL* file is provided as a template that can be used to build Web services responsible for mediating messages between the IBM Analytical Decision Management application and an external rule service. Details on the mediating Web service can be manually supplied by the integrator.
- **Include dimension metadata.** If a dimension tree has been defined for the project on the Define tab, dimension tree metadata can optionally be included.

Downloading data model metadata for an IBM SPSS Collaboration and Deployment Services analytic data view

In IBM SPSS Collaboration and Deployment Services Deployment Manager, you can export metadata for a particular analytic data view.

Metadata downloaded in this way supports the creation of rules based on the logical data model of an analytic data view. An analytic data view's logical data model may be hierarchical. An IBM Analytical Decision Management Data View source supports generation of an XML representation of the hierarchical data for each record, which can then be passed to external rules during execution. Rules authored based on this metadata may be complex and utilize the hierarchical data model.

In IBM Analytical Decision Management, an external rule based on a hierarchical data model may only be referenced in a project where the project data model uses a Data View source. To ensure compatibility, the Data View source and the external rule must be based on the same analytic data view's logical data model. Only the Data View source can generate the hierarchical data based on the analytic data view instance it references. In other words, to use external rules, the project data model in the IBM Analytical Decision Management project needs to match that used to create the rule, and the analytic data view makes that possible.

To download a *.ZIP* file with metadata for the analytic data view's logical data model:
1. In IBM SPSS Collaboration and Deployment Services Deployment Manager, open an analytic data view from the repository and go to the **Logical** tab.
2. Right-click a table and export the XOM metadata.
3. Select a location to save the file to and click **Save**.

The following files are included in the *.ZIP* file:

XML schema definitions (*.XSD). Two or more XML schema definitions are included. One contains a definition which may be useful for a rule response type. The other schema(s) contain the definitions of the tables, including hierarchical relationship detail and field types in the logical data model of the analytic data view.

You can import these schema files into an external development tool such as IBM Operational Decision Management Rule Designer in order to develop rules for use with a project whose project data model uses a Data View source based on the same analytic data view instance.

To improve performance, administrators are given the option to restrict use of these complex external rules. For details, see the topic "Locking data options" in the online help for the application.

Editing the XML schema definition

The *.XSD* file downloaded from IBM Analytical Decision Management contains two complex data types: one representing the IBM Analytical Decision Management project data model, which should be used for the input parameter of IBM Operational Decision Management rules projects; and one representing a rules return type, which should be used for the output parameter of IBM Operational Decision Management rules projects.

The IBM SPSS Collaboration and Deployment Services Deployment Manager analytic data view metadata contains two or more *.XSD* files: one relating to the logical data model of an analytic data view, which should be used for the input parameter of IBM Operational Decision Management rules projects; and one or more containing a complex type representing a rules return type, which should be used for the output parameter of IBM Operational Decision Management rules projects.

Expert users may choose to edit the generated *.XSD* schema definition to tailor them for the specific rule or rules they wish to create. There are circumstances when the user wants to edit the input and/or the response related types defined in the related *.XSD* schema definition prior to creating their rules. For instance, the expert user may need to edit the input parameter related schema to permit their authored rules to handle the potential of input fields with null values. The expert user can do this by indicating via the *.XSD* schema definition which input parameter fields/elements are required and which are optional.

IBM Analytical Decision Management input parameter *.XSD*

If project metadata is downloaded from IBM Analytical Decision Management, then complex data type in the *.XSD* which represents the input parameter will contain details for all fields in the IBM Analytical Decision Management project data model.

The input type's `dmname` element, as seen in the following example, matches the project data model name. It also includes `xsd:restriction` elements for the input types (i.e. min and max values for numeric types, and enumerations for categorical types). This type metadata can be used when imported into an external development tool such as IBM Operational Decision Management Rule Designer:

```
<xsd:complexType name="drugs">
 <xsd:annotation>
  <xsd:appinfo>
   <dmname>drugs</dmname>
  </xsd:appinfo>
 </xsd:annotation>
<xsd:sequence>
  <xsd:element name="Age">
   <xsd:annotation>
    <xsd:appinfo>
     <dmname>Age</dmname>
    </xsd:appinfo>
   </xsd:annotation>
   <xsd:simpleType>
    <xsd:restriction base="xsd:long">
     <xsd:minInclusive value="15"/>
     <xsd:maxInclusive value="74"/>
    </xsd:restriction>
   </xsd:simpleType>
  </xsd:element>
  <xsd:element name="Sex">
   <xsd:annotation>
    <xsd:appinfo>
     <dmname>Sex</dmname>
    </xsd:appinfo>
   </xsd:annotation>
   <xsd:simpleType>
    <xsd:restriction base="xsd:string">
     <xsd:enumeration value="F"/>
     <xsd:enumeration value="M"/>
```

```
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  ...
 </xsd:sequence>
</xsd:complexType>
```

IBM SPSS Collaboration and Deployment Services analytic data view *.XSD*

If metadata relating to an analytic data view is downloaded from IBM SPSS Collaboration and Deployment Services Deployment Manager, then the input parameter *.XSD* schema(s) may contain multiple complex data types representing details of the analytic data view logical data model. Each complex data type will relate to a table defined in the logical data model of the analytic data view instance.

The following example includes two complex types: `Customer` and `Order`. The primary complex type is `Customer`, which has a defined relationship with `Order` (i.e. a customer can have many orders):

```
<xsd:complexType name="Customer">
 <xsd:sequence>
  <xsd:element name="FirstName" type="xsd:string"/>
  <xsd:element name="Sirname" type="xsd:string"/>
  <xsd:element name="Age" type="xsd:long"/>
  <xsd:element name="Sex" type="xsd:string"/>
  <xsd:element name="CustID" type="xsd:string"/>
  <xsd:element maxOccurs="unbounded" minOccurs="0" name="Orders" type="xom:Order"/>
 </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Order">
 <xsd:sequence>
  <xsd:element name="CustID" type="xsd:string"/>
  <xsd:element name="NoOfItems" type="xsd:long"/>
  <xsd:element name="TotalCost" type="xsd:long"/>
  <xsd:element name="Date" type="xsd:date"/>
 </xsd:sequence>
</xsd:complexType>
```

The input parameter *.XSD* exported from an analytic data view does not include `xsd:restriction` elements for the input types (i.e. min and max values for numeric type, and enumerations for categorical type).

Editing the input parameter type

The number of possible occurrences of each field-related element is not explicitly set in the *.XSD* schema (i.e., the attributes `minOccurs` and `maxOccurs` are not being set, and the default value for both is 1). As such, if the schema is not edited before using it to generate IBM Operational Decision Management rules, all fields in the input parameter type will be required by the generated rules.

Editing the project data model related input parameter type

If the schema defining the input parameter type was generated by downloading an IBM Analytical Decision Management project's metadata, the input parameter's data model will be flat. In this case, IBM Analytical Decision Management will directly create the payload required by the rule service on execution. This gives the expert user more freedom when editing the input parameter type and still maintaining compatibility (i.e., the expert user can remove fields/elements they will not require for rule authoring, or indicate fields/elements are optional).

Editing the analytic data view logical data model related input parameter type

If the schema defining the input parameter type was generated by exporting an analytic data view's logical data model from IBM SPSS Collaboration and Deployment Services Deployment Manager client, the input parameter's data model may be hierarchical. If it is hierarchical, then IBM Analytical Decision Management will *not* directly create the payload required by the rule service on execution. The main part of the payload will be generated by the Analytic Data View source. This limits how the expert user may

edit the input parameter type and maintain compatibility (for example, the expert user can only indicate fields/elements are optional).

General information

Prior to invoking a rule created in a Business Rules Management System such as IBM Operational Decision Management, IBM Analytical Decision Management asks the rule's service whether its input parameter data model is hierarchical or flat. If the input parameter is flat, it also checks which inputs are required by the rule's service.

If the input parameter's data model is flat, and the record to score contains a null value for any of the rule service's required fields, IBM Analytical Decision Management will not invoke the rule, and the output/response fields of the rule are populated with default values. The value depends on the type of the output/response field as follows:

Table 8. Values for response field types.

Response field type	Value
xsd:string	""
xsd:boolean	""
xsd:int	0
xsd:long	0
xsd:byte	0
xsd:double	0.0
xsd:float	0.0
xsd:date	0.0
xsd:time	0.0
xsd:datetime	0.0

This can impact the results of scoring records via a rule service, and care should be taken in editing the input parameter in the *.XSD* schema.

If the input parameter's data model is hierarchical, and the record to be scored contains a null value for any of the rule service's required fields, IBM Analytical Decision Management will invoke the rule, but the field or fields with a null value will be omitted from the payload created by the Data View source node. If the omitted field was a required field, this will cause the rule invocation to fail, so again care should be taken when editing the input parameter in the *.XSD* schema.

Editing the schema requires knowledge of the rule services that will be created. If a particular input field is not required by the rule, and the rule's input parameter data model is **flat**, then the field-related element may either be **removed** from the input parameter in the schema or **marked as optional**. If a particular field is not required by the rule, and the rule's input parameter is **hierarchical**, the field-related element **cannot be removed**; but it may be **marked as optional**. This is because IBM Analytical Decision Management does not directly generate the payload for rules based on a hierarchical data model.

When you remove an input parameter field, the field value is ignored when the rule is called and the rule cannot refer to the field. When you make an input parameter field optional, a null value is allowed and the rule must be written defensively to check for that. For example, if the rule's input parameter data model is flat and you know the field Age will not be used by the rule, you can remove the related element to indicate it is not a required field. In the following *.XSD* schema example, if the field Age is not required, the entire section <xsd:element name="Age"> can be removed from the schema:

```
<xsd:complexType name="drugs">
<xsd:annotation>
 <xsd:appinfo>
  <dmname>drugs</dmname>
 </xsd:appinfo>
 </xsd:annotation>
<xsd:sequence>
 <xsd:element name="Age">
  <xsd:annotation>
   <xsd:appinfo>
    <dmname>Age</dmname>
   </xsd:appinfo>
  </xsd:annotation>
  <xsd:simpleType>
   <xsd:restriction base="xsd:long">
    <xsd:minInclusive value="15"/>
    <xsd:maxInclusive value="74"/>
   </xsd:restriction>
  </xsd:simpleType>
 </xsd:element>
 <xsd:element name="Sex">
  <xsd:annotation>
   <xsd:appinfo>
    <dmname>Sex</dmname>
   </xsd:appinfo>
  </xsd:annotation>
  <xsd:simpleType>
   <xsd:restriction base="xsd:string">
    <xsd:enumeration value="F"/>
    <xsd:enumeration value="M"/>
   </xsd:restriction>
  </xsd:simpleType>
 </xsd:element>
 ...
 </xsd:sequence>
</xsd:complexType>
```

Or if the rule's input parameter data model is flat or hierarchical, you might want to make an input field element optional if it's not required by the rule(s)--by setting the element's minOccurs attribute value to 0:

```
<xsd:complexType name="drugs">
 <xsd:annotation>
  <xsd:appinfo>
   <dmname>drugs</dmname>
  </xsd:appinfo>
 </xsd:annotation>
<xsd:sequence>
  <xsd:element name="Age" minOccurs="0">
   <xsd:annotation>
    <xsd:appinfo>
     <dmname>Age</dmname>
    </xsd:appinfo>
   </xsd:annotation>
   <xsd:simpleType>
    <xsd:restriction base="xsd:long">
     <xsd:minInclusive value="15"/>
     <xsd:maxInclusive value="74"/>
    </xsd:restriction>
   </xsd:simpleType>
  </xsd:element>
  <xsd:element name="Sex">
   <xsd:annotation>
    <xsd:appinfo>
     <dmname>Sex</dmname>
    </xsd:appinfo>
   </xsd:annotation>
   <xsd:simpleType>
    <xsd:restriction base="xsd:string">
     <xsd:enumeration value="F"/>
     <xsd:enumeration value="M"/>
    </xsd:restriction>
```

```
    </xsd:simpleType>
   </xsd:element>
   ...
  </xsd:sequence>
</xsd:complexType>
```

Hierarchical data model example:

```
<xsd:complexType name="Customer">
 <xsd:sequence>
  <xsd:element name="FirstName" type="xsd:string" minOccurs="0"/>
  <xsd:element name="Sirname" type="xsd:string" minOccurs="0"/>
  <xsd:element name="Age" type="xsd:long"/>
  <xsd:element name="Sex" type="xsd:string"/>
  <xsd:element name="CustID" type="xsd:string"/>
  <xsd:element maxOccurs="unbounded" minOccurs="0" name="Orders" type="xom:Order"/>
 </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Order">
 <xsd:sequence>
  <xsd:element name="CustID" type="xsd:string"/>
  <xsd:element name="NoOfItems" type="xsd:long"/>
  <xsd:element name="TotalCost" type="xsd:long" minOccurs="0"/>
  <xsd:element name="Date" type="xsd:date"/>
 </xsd:sequence>
</xsd:complexType>
```

The rules can also be written defensively in order to handle null values, in which case the field can be marked as optional in the schema. Following is an example of an IBM Operational Decision Management rule called *currentOfferIsReplacePhone* that has been written to deal with null values:

```
if
    the CURRENT OFFER of 'the customer' is not null
    and the CURRENT OFFER of 'the customer' is "Replace Phone"
then
    make it true that 'the response' is value ;
else
    make it false that 'the response' is value ;
```

Editing the response parameter type

Note that it is possible to create and utilize IBM Operational Decision Management rules that return a variety of types to IBM Analytical Decision Management (*boolean* for use in Selections, *numeric* for use in aggregate rules for example, and *string*). However, to create anything other than a *string* data response rule, the XML schema definition (*.XSD*) which contains the response type must be manually edited.

Also, a rule project can contain multiple rules, each returning different types. It is necessary to maintain multiple versions of the *.XSD* to support this (for example, a version where the data response variable has been set to numeric, another for string, another for boolean, etc.).

The *.XSD* file downloaded from IBM Analytical Decision Management contains two complex data types: one representing the IBM Analytical Decision Management project data model which should be used for the input parameter of IBM Operational Decision Management rules projects, and one representing a rules return type which should be used for the output parameter of IBM Operational Decision Management rules projects.

The response-related *.XSD* file downloaded from IBM SPSS Collaboration and Deployment Services Deployment Manager contains the response type definition and will only contain the one complex data type.

In both of the above cases, the return type (whose name will always have a "response" suffix) defines a value element which defaults to a *string*. If you want to create a rules project with a different kind of return value, you must manually edit the downloaded *.XSD* before importing the IBM Operational Decision Management rules project so that the return type's value element is defined as desired. Following are a few examples:

```
<xsd:complexType name="claim_dataresponse">
    <xsd:annotation>
        <xsd:appinfo>
            <dmname>claim_dataresponse</dmname>
        </xsd:appinfo>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:element name="value" type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>



<xsd:complexType name="claim_dataresponse">
    <xsd:annotation>
        <xsd:appinfo>
            <dmname>claim_dataresponse</dmname>
        </xsd:appinfo>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:element name="value" type="xsd:boolean"/>
    </xsd:sequence>
</xsd:complexType>



<xsd:complexType name="claim_dataresponse">
    <xsd:annotation>
        <xsd:appinfo>
            <dmname>claim_dataresponse</dmname>
        </xsd:appinfo>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:element name="value" type="xsd:long"/>
    </xsd:sequence>
</xsd:complexType>
```

Creating external rule references

In IBM SPSS Collaboration and Deployment Services Deployment Manager, the Add New External Rule dialog box defines a reference to an external rule accessed through a web service, or a reference to a local rule. These rules can then be used in IBM Analytical Decision Management applications. For example, this makes it possible to support IBM Operational Decision Management rules in IBM Analytical Decision Management.

1. To create a rule reference, in IBM SPSS Collaboration and Deployment Services Deployment Manager, select a folder in the repository and from the menus choose:

   File > **New** > **External Rule...**

   The Add New External Rule dialog will be displayed.

2. Specify a name for the external rule reference.

3. In the **Execution Mode** section, select **Web Service** or **Local (IBM Operational Decision Management rules only)**.

   For the **Web Service** option, the rule is executed via an external web service. For the **Local** option, the rule is executed via the IBM SPSS Modeler Server (the same server defined in IBM SPSS Collaboration and Deployment Services Deployment Manager for use by IBM Analytical Decision Management). Local IBM Operational Decision Management rules are deployed to a Rule Execution Server embedded in the same Java Virtual Machine as the IBM SPSS Modeler Server JVM or the application server JVM.

   Important: Before using the **Local (IBM Operational Decision Management rules only)** option, note that some installation and configuration steps are required. See "Setting up the local Rule Execution Server" on page 67.

4. If you selected **Web Service** in the previous step, enter the URL of the web service where the rule is deployed (for example, *http://myRuleServer:8080/theRestOfTheURL*). The URL will be validated

automatically to ensure it meets the appropriate naming conventions. If you selected **Local (IBM Operational Decision Management rules only)** in the previous step, enter the path to the rule in the local IBM Operational Decision Management repository. You can use the *reslist* script to obtain this path, as described later in this document.

Once defined in IBM SPSS Collaboration and Deployment Services Deployment Manager, external rules can be browsed and used in IBM Analytical Decision Management applications in much the same manner as other models and rules. For more information, see the *User's Guide* for the IBM Analytical Decision Management application.

Setting up the local Rule Execution Server

If you plan to reference local rules from IBM Operational Decision Management, some configuration steps are required for setting up and deploying your local rules.

Step 1: Copy required files from IBM Operational Decision Management V8.6.0.0

1. Copy the file *rescopy.xml* from the following folder on the IBM SPSS Collaboration and Deployment Services server machine onto the machine where IBM Operational Decision Management V8.6.0.0 is installed.

   C:\Program Files\IBM\SPSS\Deployment\7.0\Server\components\decision-management \Utilities\RESCopy\

   The *rescopy.xml* script will aid in copying certain required files from the IBM Operational Decision Management system into the IBM Analytical Decision Management installation. The script requires two parameters:

   `-Dwodm` identifies the IBM Operational Decision Management *executionserver* directory.

   `-Dtarget` identifies the target directory where files will be copied and a *.zip* file will be built.

2. Run the *rescopy.xml* script to copy the required files. For example, type the following in the Windows command prompt and press Enter:

   ```
   CALL "%ANT_HOME%\bin\ant" -buildfile rescopy.xml -Dwodm="C:\Program Files\IBM\ODM85\
   ODM\executionserver" -Dtarget=C:\CopiedFiles
   ```

   This will run the script to produce the following folders and then combine everything into a *DM-rescopy.zip* file:

   ```
   C:\CopiedFiles\rescopy\lib\
           asm-3.1.jar
           asm-analysis3.1.jar
           asm-commons-3.1.jar
           asm-tree-3.1.jar
           asm-util.3.1.jar
           bcel-5.1.jar
           dom4j-1.6.1.jar
           j2ee_connector-1_5-fr.jar
           jdom-1.1.2.jar
           jrules-engine.jar
           jrules-res-execution.jar
           log4j-1.2.8.jar
           mina-core-2.0.1-IBM.jar
           openxml4j-beta.jar
           slf4j-api-1.5.8.jar
           slf4j-simple-1.5.8.jar
           sam.jar
           ra.xml

   C:\CopiedFiles\rescopy\tools\
           jrules-res-tools.jar

   C:\CopiedFiles\rescopy\
           DM-rescopy.zip
   ```

Step 2: Install the required files onto all machines with the pasw.externalrule component

The files copied in the previous section must be installed on the IBM SPSS Collaboration and Deployment Services server, and also onto all IBM SPSS Modeler products that will be used with IBM Analytical Decision Management or with IBM SPSS Modeler Advantage to execute local IBM Operational Decision Management rules. The *pasw.externalrule* folder described in this section is installed with the extensions for IBM SPSS Modeler, which is part of the IBM Analytical Decision Management installation process.

- Extract the *DM-rescopy.zip* file created in the previous section into each */ext/bin/pasw.externalrule* folder. Example locations are provided below. After extraction, several *.jar* files and the *ra.xml* file will be installed to the *lib* folder, and a *tools* folder will be created containing a single *.jar* file.

  IBM SPSS Collaboration and Deployment Services: *C:\Program Files\IBM\SPSS\Deployment\7.0\ Server\components\modeler\ext\bin \pasw.externalrule\.*

  IBM SPSS Modeler Server: *C:\Program Files\IBM\SPSS\ModelerServer\17\ext\bin\pasw.externalrule\.*

  IBM SPSS Modeler (clients): *C:\Program Files\IBM\SPSS|Modeler\17\ext\bin\pasw.externalrule\.*

Step 3: Modify the resdeploy, reslist, and resundeploy scripts

The following three scripts are provided with IBM Analytical Decision Management:

- The **resdeploy** script deploys IBM Operational Decision Management rules for local execution.
- The **reslist** script lists the IBM Operational Decision Management rules deployed for local execution.
- The **resundeploy** script undeploys IBM Operational Decision Management rules for local execution.

Windows (*.bat*) and UNIX (*.sh*) versions of these scripts are installed to the *pasw.externalrule\scripts* folder in the IBM SPSS Collaboration and Deployment Services installation directory (for example, *C:\Program Files\IBM\SPSS\Deployment\7.0\Server\components\modeler\ext\bin\pasw.externalrule\scripts*). Complete the following steps to prepare the scripts for use.

1. Using a simple text editor, edit the value for CADS_HOME in the three scripts so it points to your IBM SPSS Collaboration and Deployment Services installation directory. If on Windows, edit the *.bat* versions. If on UNIX, edit the *.sh* versions. For example:

   ```
   set CADS_HOME=C:\Program Files\IBM\SPSS\Deployment\7.0\Server
   ```

2. If the machine's operating system is Solaris, also change the first line of the scripts from #!/bin/sh to #!/bin/ksh.

Step 4: Modify the local Rule Execution Server configuration file

Local IBM Operational Decision Management rules are deployed to an IBM Operational Decision Management Rule Execution Server embedded in the same Java Virtual Machine (JVM) as the IBM SPSS Modeler Server JVM or the application server JVM. Each local IBM Operational Decision Management Rule Execution Server must be configured prior to use by editing the ra.xml file with any required changes.

The objectives of configuring the local Rule Execution Servers are to:

- Permit "hot" deployment of local IBM Operational Decision Management rules
- Specify the Rule Execution Server's persistence type and location. The persistence type may be file or database.

Using a simple text editor, edit each copy of the ra.xml file as follows. The file is located in the pasw.externalrule\lib folder of the IBM SPSS Collaboration and Deployment Services installation directory, IBM SPSS Modeler Server installation directory, and IBM SPSS Modeler client installation directory, as described previously.

Note that after modifying the ra.xml file, for the changes to take effect you must restart the product the ra.xml file was installed to.

Enabling hot deployment of local rules

As stated, local IBM Operation Decision Management rules are deployed to a J2SE type Rule Execution Server embedded in the same Java Virtual Machine (JVM) as the calling application. In the context of IBM Analytical Decision Management execution, this will be either the IBM SPSS Modeler Server JVM or the application server JVM.

By default, a J2SE Rule Execution Server is configured to be stand-alone and unmanaged. Once an IBM Operation Decision Management rule is loaded, the standalone J2SE Rule Execution Server will not re-load the rule until the JVM is restarted. This is particularly important when an IBM Analytical Decision Management project will be run in the application server JVM, because you do not want to start and stop the application server just to pick up a rule change.

The local/J2SE Rule Execution Server can be configured to use a shared database and to be managed by a Rule Execution Server console over TCP/IP. The Rule Execution Server console will inform the local/J2SE Rule Execution Servers when a rule is changed in the shared database, causing the local/J2SE RES to reload the rule without the need for the JVM to be restarted.

The IBM Operational Decision Management Rule Execution Server console is a web-based graphical interface that provides access to most Rule Execution Server features. It is a component of IBM Operational Decision Manager's IBM Decision Server, and will have been installed as part of the IBM Decision Manager installation (which was used to copy the files from in step 1).

Note: The following must be adhered to when configuring the Rule Execution Server console to support local Rule Execution Server hot deployment

- The Rule Execution Server console must use a shared network database for persistence -- so the local J2SE Rule Execution Servers can use the same rule store as the Rule Execution Server console that manages them.
- The Rule Execution Server console must have TCP/IP notification enabled to permit the console to inform local/J2SE Rule Execution Servers of updates

See https://www-01.ibm.com/support/knowledgecenter/SSQP76_8.6.0/ com.ibm.odm.dserver.rules.res.managing/topics/tsk_res_rescons_config_tcpip.html for more information.

To permit the local Rule Execution Server to be managed by the Rule Execution Server console, you must edit the `ra.xml` file to enable Execution Unit management over TCP/IP.

1. Modify `ra.xml` to specify the Rule Execution Server console host and notification port. For example:

```
<config-property>
<config-property-name>plugins</config-property-name>
<config-property-type>java.lang.String</config-property-type>
<config-property-value>
    (pluginClass=com.ibm.rules.res.xu.management.XUManagementPlugin,
        xuName=default,
        protocol=tcpip,
        tcpip.port=1883,
        tcpip.host=9.20.103.80,
        tcpip.retryInterval=10000,
</config-property-value>
</config-property>
```

   For details, see https://www-01.ibm.com/support/knowledgecenter/SSQP76_8.6.0/ com.ibm.odm.dserver.rules.res.managing/topics/tsk_res_config_xu_tcpip.html.

Setting persistence

1. By default, the `persistenceType` setting has a value of `file`, which signifies that the Rule Execution Server will use `file` system persistence. For example:

```
<config-property-name>persistenceType</config-property-name>
<config-property-type>java.lang.String</config-property-type>
<config-property-value>file</config-property-value>
</config-property>
```

If `database` system persistence is desired instead, skip this step.

Note: If the local Rule Execution Server is to be managed by a shared Rule Execution Server console, then `database` persistence is expected. Each local `ra.xml` file should be configured to use the same shared network database as the Rule Execution Server console.

Change the value of the `persistenceProperties` setting to `DIRECTORY=<RES path>`, where `<RES path>` is the full path to the Rule Execution Server data folder that will be used to store the deployed IBM Operational Decision Management rules. Note that according to Java standards, forward slashes should be used in paths (regardless of the operating system used). For example:

```
<config-property-name>persistenceProperties</config-property-name>
<config-property-type>java.lang.String</config-property-type>
<config-property-value>DIRECTORY=C:/RES</config-property-value>
```

Notes:

- By default, the `persistenceProperties` setting may contain the `XOM_PERSISTENCE_TYPE`, `XOM_PERSISTENCE_DIRECTORY`, and `DIRECTORY` options. For example:

```
<config-property-name>persistenceProperties</config-property-name>
<config-property-type>java.lang.String</config-property-type>
<config-property-value>
DIRECTORY=res_data,XOM_PERSISTENCE_TYPE=file,XOM_PERSISTENCE_DIRECTORY=res_xom
</config-property-value>
```

  In IBM Analytical Decision Management, only the `DIRECTORY` option is supported, so the `XOM_PERSISTENCE_TYPE` and `XOM_PERSISTENCE_DIRECTORY` options should be removed, leaving only the `DIRECTORY` option.

- If file persistence is used and IBM SPSS Collaboration and Deployment Services and IBM SPSS Modeler Server (or client) are installed on different machines, the directory specified here must be accessible from all the machines (perhaps shared over the network). The same can be said if using database persistence. The database needs to be accessible from all machines. The `ra.xml` file installed on each SPSS product machine should point to the same local Rule Execution Server repository to avoid problems with keeping repositories in sync.

2. If you want to use `database` persistence instead of `file` persistence, change the `persistenceType` setting from `file` to `jdbc` as follows.

```
<config-property-name>persistenceType</config-property-name>
<config-property-type>java.lang.String</config-property-type>
<config-property-value>jdbc</config-property-value>
```

Then change the `persistenceProperties` setting as follows. You must define all information necessary for the Rule Execution Server JVM to successfully access the configured database server. This includes the database URL, the fully qualified name of the JDBC driver class, and any optional security credentials. The JDBC persistence example below is for a case where the remote database is of the type DB2 and resides on the fictitious server `mydb2server.mycompany.ibm.com`.

```
<config-property-name>persistenceProperties</config-property-name>
<config-property-type>java.lang.String</config-property-type>
<config-property-value>
    DRIVER_CLASS_NAME=com.ibm.db2.jcc.DB2Driver,
    URL=jdbc:db2://mydb2server.mycompany.com:50000/ILOGRESDB,
    USER=dbuser1,
    PASSWORD=dbpass
</config-property-value>
```

Notes:

- As mentioned previously, the `XOM_PERSISTENCE_TYPE` and `XOM_PERSISTENCE_DIRECTORY` options are not supported with IBM Analytical Decision Management and should not be included in the `persistenceProperties` section.

- If a Java class name is supplied in the `DRIVER_CLASS_NAME` of the `persistenceProperties` setting, it is also necessary to add the appropriate Java client .jar files(s) to the classpath of the Rule Execution Server JVM. To do this, simply copy the required .jar file(s) into all the `pasw.externalrule\lib` folders (the same folders where `ra.xml` exists, as described previously). For example, if the remote database is DB2 for Windows, the client .jar file `db2jcc.jar` should be copied from `<DB2_HOME>/java` to the `pasw.externalrule\lib` folder of the IBM SPSS Collaboration and Deployment Services installation directory, IBM SPSS Modeler Server installation directory, and IBM SPSS Modeler client installation directory.

Step 5: Using scripts to deploy, list, and undeploy local IBM Operational Decision Management rules

The three scripts described below are provided with IBM Analytical Decision Management to list, deploy, and undeploy local IBM Operational Decision Management rules. Before using the scripts, follow the steps in "Step 3: Modify the resdeploy, reslist, and resundeploy scripts" on page 68. Scripts are provided for Windows (*.bat*) and UNIX (*.sh*) platforms.

Note that the `persistenceType` defined previously has no impact on the behavior of these scripts. However, if *database* persistence is used instead of *file* persistence, it may be necessary to use the optional `-cp` argument when running the scripts to add the necessary database driver *.jar* file(s) to the runtime classpath. However, if the *.jar* file(s) were already copied as described in "Step 4: Modify the local Rule Execution Server configuration file" on page 68, the `-cp` argument should not be needed.

To list the contents of a local Rule Execution Server

- At a command prompt, change to the directory containing the *reslist* script. Then run the *reslist* script as follows. The script requires a `-config` argument followed by the full path to the *ra.xml* file on the IBM SPSS Collaboration and Deployment Services server. On Windows, for example:

```
reslist.bat -config "C:\Program Files\IBM\SPSS\Deployment\7.0\Server\components\
modeler\ext\bin\pasw.externalrule\lib\ra.xml"
```

To deploy IBM Operational Decision Management RuleApp archives to a local Rule Execution Server

1. Download the IBM Operational Decision Management RuleApp archive (*.jar* file) to a folder accessible from your IBM SPSS Collaboration and Deployment Services server machine. You can download the archive from the Rule Execution Server hosting the web service, or export it from IBM Operational Decision Management Rule Designer. This is a temporary copy, so you can place it anywhere.

2. At a command prompt, change to the directory containing the *resdeploy* script. Then run the *resdeploy* script as follows. The script requires a `-config` argument followed by the full path to the *ra.xml* file on the IBM SPSS Collaboration and Deployment Services server, and it requires an `-archive` argument followed by the full path to the RuleApp *.jar* file to be deployed. On Windows, for example:

```
resdeploy.bat -config "C:\Program Files\IBM\SPSS\Deployment\7.0\Server\components\
modeler\ext\bin\pasw.externalrule\lib\ra.xml" -archive "C:\RES\My_ODM_RuleApp_Name.jar"
```

The *resdeploy* script also provides optional arguments for controlling versions of deployed rulesets. If the Rule Execution Server contains an earlier version of the RuleApp (or one of the rulesets it contains) that you want to deploy, you may want to use some of the following optional arguments when running the script to avoid naming conflicts, for example.

Table 9. resdeploy script optional arguments

Optional argument	Possible values
-merging	• `add.ruleapp` keeps the existing RuleApp in the Rule Execution Server and deploys the new RuleApp with a version number that increments the original (default behavior) • `replace.ruleapp` replaces the original RuleApp with the new RuleApp so there is no RuleApp version number change • `add.ruleset` keeps the existing ruleset in the Rule Execution Server and deploys the new ruleset with a version number that increments the original • `replace.ruleset` replaces the original ruleset with the new ruleset so there is no ruleset version number change
-versioning	• `major` applies any increments to the major parts of version numbers (default behavior) • `minor` applies any increments to the minor parts of the version numbers

To view usage information for any of the scripts, run them without any command line arguments.

To remove IBM Operational Decision Management RuleApps (and their rulesets) from a local Rule Execution Server

• At a command prompt, change to the directory containing the *resundeploy* script. Then run the *resundeploy* script as follows. The script requires a `-config` argument followed by the full path to the *ra.xml* file on the IBM SPSS Collaboration and Deployment Services server, and it requires a `-ruleapp` argument followed by the name of the RuleApp to be removed. On Windows, for example:

```
resundeploy.bat -config "C:\Program Files\IBM\SPSS\Deployment\7.0\Server\components\modeler\
ext\bin\pasw.externalrule\lib\ra.xml" -ruleapp "My_ODM_RuleApp_Name"
```

By default, the *resundeploy* script will try to remove version 1.0 of the RuleApp. When necessary, different versions can be removed by using an optional `-version` argument followed by the version number to remove.

Before running the *resundeploy* script, you can use the *reslist* script to verify the name of the RuleApp to remove and also to verify the RuleApp's removal after running the *resundeploy* script.

Chapter 6. Updating custom application templates and projects

You may choose to update your IBM Analytical Decision Management applications for various reasons:

- Updating of custom application templates following installation of a new version of the software, or migration to a new server environment running the same or a newer version. Note that when you upgrade to a new version of the software, you only need to update custom application templates and projects based on those templates if you want to take advantage of new features in the new version of the software. Otherwise, projects based on custom application templates will continue to function as they did before after installing the new software version.

- Updating of existing projects to apply changes to the XML application template on which the project was based. This can happen in combination with migration to a new release, or any time changes have been made to an XML application template for any reason.

Updating custom application templates

The basic steps for updating a custom application template XML file are detailed below. It is important to note that when you install a new version of IBM Analytical Decision Management, the previous installation of the product will automatically be upgraded. No manual uninstallation or migration is required. Any custom application templates you have will be preserved, and projects based on them will work as they did previously. Standard IBM Analytical Decision Management prebuilt applications are subject to modification as upgrades are installed, and changes will be picked up when new projects are created in the user interface.

If you want your custom applications to take advantage of any new features added to the new version of the product, you must update your XML application template and then use the *Template Utility* to update projects that are based on that template.

If you want to add support for new features added to the software, the custom application template XML files, any stylesheet customizations, and any other customized application files must be updated:

1. Make any desired changes or additions to the application template. For example, a new type of user interface control may require additional attributes under the `InterfaceControl` element.

2. Customized CSS styles, images, and *.properties* files may need to be updated, though these depend on the extent of customizations you have made, and should be minimal if care was taken to minimize changes to application stylesheets and leverage defaults when appropriate. (The default stylesheets under *\Server\components\decision-management\Default* are automatically updated when the new software version is installed, along with subfolders for packaged applications such as *\Server\components\decision-management\Applications\ClaimsManagement*. All applications that leverage these default folders will automatically pick up changes, however custom stylesheets and other files for custom applications that are not part of the default files must be manually updated.)

3. Because default stylesheets are automatically updated with the software, some changes may be immediately visible in the user interface. But in some cases, additional application template changes may be required for all features to work as expected.

4. In cases where template changes are required, existing projects based on those templates will also need to be updated using the *Template Utility* as described in the section "IBM Analytical Decision Management Template Utility" on page 74.

Note: Migrating to an earlier version of IBM Analytical Decision Management than the one used to create the application is not supported.

Updating projects

Projects are the application workspaces (IBM SPSS Modeler streams) used by end users in the interface, and are based on application templates. Projects saved in an earlier version of the software will continue to work in the new version. No change to the project is required for this to work.

- However, to add support for changes made to the custom application template, existing projects will need to be updated using the *Template Utility*. Such changes made to a custom application template may be to pick up new features added to the new release of the software, or simply to fine-tune the functionality in that application. In either case, the projects created using the original application template will need to be updated using the *Template Utility* before they will reflect the changes made to the underlying template.
- Prior to being updated, projects will continue to function as they did previously.
- If no changes to application templates are needed, use of the *Template Utility* is not required.

IBM Analytical Decision Management Template Utility

The *Template Utility* can be used to update existing projects following changes to the XML application template on which the projects are based.

A typical use case is as follows:

- A custom XML application template has been created and saved in the standard location under the IBM SPSS Collaboration and Deployment Services installation folder (for example, *C:\Program Files\IBM\SPSS\Deployment\7.0\Server\components\ decision-management\Templates\ MyCustomClaimsTemplate.xml*).
- End users have created and saved one or more projects in IBM Analytical Decision Management based on that template.
- Subsequent changes to the XML application template have been introduced.
- To pick up these changes, existing projects now need to be updated to work with the new version of the application template.

Note: It is strongly recommended that the updated application template be reviewed and tested thoroughly to ensure that required changes are complete and functioning as expected before existing projects are updated.

Before using the Template Utility

1. In your custom application template XML file, increment the `templateVersion` attribute. For example, if the previous value was 1, change it to 2. This attribute is used in combination with the `templateName` attribute to identify projects to update. Projects based on template versions earlier than the specified value will be updated.

   ```
   <PredictiveApplication xmlns="http://com.spss.pasw.dms/workspace" templateName="MyCustomClaimsTemplate"
   templateVersion="2" appsVersion="17.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   ```

2. Create an XML "patch" file containing the updated sections of the application template you want to push out to end users' projects.

   The XML file requires the following elements.

- A `patch` element identifying the `templateName`, `templateVersion`, and `appliesToVersions`. For example:

   ```
   <patch templateName="MyCustomClaimsTemplate" templateVersion="8" appliesToVersions="4,5,6,7">
   ```

- Optionally, you can also include an `instructions` element for informational purposes. The `instructions` text you include will be displayed in the **Patch Instructions** section of the *Template Utility*. For example:

```
<instructions>
        This patch deletes the Offer dimension from the projects. After applying the patch,
        the projects based on the template will be updated the next time they are opened
        and resaved in Decision Management.
</instructions>
```

- An `actions` element.
- Inside the `actions` element, one or more `action` elements. The `action` elements define the type of `patch` action to perform (`addElement`, `replaceElement`, `deleteElement`, or `setAttribute`) and the `path` to the section of the application template you want to push out to end users' projects. For example:

```
<actions>
        <action type="deleteElement" path="/PredictiveApplication/Dimension[@name='Offer']">
        </action>
</actions>
```

- Inside each `action` element, one `NewValue` element is included if you are adding or replacing sections of the template. Each `NewValue` element defines the new values to use. For example:

```
<actions>
        <action type="addElement" path="/PredictiveApplication">
            <NewValue>
                <Dimension name="Channel" description="ChannelDescriptionText">
                    <Variable name="Capacity" dataType="integer"
                        optimizationInputItem="true" prompt="">
                        <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                            xsi:type="ConstantValueSource">
                            <Value>0</Value>
                        </ValueSource>
                    </Variable>
                </Dimension>
            </NewValue>
        </action>
</actions>
```

You can add, replace, delete, or modify many parts of projects via the `path` (xpath) in XML patch files. Following are common examples complete XML patch files.

addElement examples

The `addElement` patch action type is used for adding sections to projects. The `path` must point to the parent location in the XML under which you want to add the item. `addElement` is one of the most common updates made to projects. The following three patch file examples are shipped with the *Template Utility* inside a *samples* folder for your reference.

patch_add_new_dim_hierarchy.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<patch templateName="MyClaimsManagement" templateVersion="2"
    appliesToVersions="1">
    <instructions>This patch adds a new dimension hierarchy with two
        dimensions (Agent and Agent Type)</instructions>
    <actions>
        <action type="addElement"
            path="/PredictiveApplication/InterfaceControl/InterfacePages/DefineStep">
            <NewValue>
                <DimensionSetting name="Agent Type">
                    <SelectionSection enabled="true" enableModels="false" />
                    <AggregateRuleSection enabled="false" />
                    <PredictiveModelSection enabled="false" />
                    <AllocationRuleSection enabled="true" />
                    <PlanningSection enableInteractionPoints="true"
                            enableStartEndDates="true" />
                </DimensionSetting>
            </NewValue>
        </action>
        <action type="addElement" path="/PredictiveApplication">
            <NewValue>
                <Dimension name="Agent Type">
                    <Property>Name</Property>
                    <Property>Category</Property>
```

```
                    <Property>Organization</Property>
                    <Property>Group</Property>
                    <Property>Description</Property>
                </Dimension>
            </NewValue>
        </action>
        <action type="addElement" path="/PredictiveApplication">
            <NewValue>
                <Dimension name="Agent" parentDimension="Agent Type">
                    <Property>Name</Property>
                    <Property>Category</Property>
                    <Property>Organization</Property>
                    <Property>Group</Property>
                    <Property>Description</Property>
                </Dimension>
            </NewValue>
        </action>
        <action type="addElement" path="/PredictiveApplication/Deployment">
            <NewValue>
                <OutputAttribute referenceType="DimensionMember"
                    name="AgentType" returnValue="AgentType.Allocation-Value">Agent Type</OutputAttribute>
            </NewValue>
        </action>
        <action type="addElement" path="/PredictiveApplication/Deployment">
            <NewValue>
                <OutputAttribute referenceType="DimensionMember"
                    name="Agent" returnValue="Agent.Allocation-Value">Agent</OutputAttribute>
            </NewValue>
        </action>
    </actions>
</patch>
```

patch_add_output_attribute.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<patch templateName="MyCampaignOptimization" templateVersion="2"
    appliesToVersions="1">
    <instructions>This patch adds 'Capacity' as an output attribute</instructions>
    <actions>
        <action type="addElement" path="/PredictiveApplication/Deployment">
            <NewValue>
                <OutputAttribute referenceType="Variable"
                    name="Capacity" returnValue="Variable.Variable-Value">Capacity</OutputAttribute>
            </NewValue>
        </action>
    </actions>
</patch>
```

patch_enable_batch_scoring.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<patch templateName="MyCustomerInteractionManagement"
    templateVersion="2" appliesToVersions="1">
    <instructions>This patch enables batch scoring on the Deploy tab</instructions>
    <actions>
        <action type="addElement"
            path="/PredictiveApplication/InterfaceControl/InterfacePages/DeployScoreStep">
            <NewValue>
                <ImmediateBatchScoring enableScoringOptions="true" />
            </NewValue>
        </action>
    </actions>
</patch>
```

replaceElement example

The replaceElement patch action type is used for replacing sections of projects. The path must point to the specific XML item you want to replace.

```
<patch templateName="MyCustomCampaignApp" templateVersion="8"appliesToVersions="5,6,7">
    <instructions>This patch replaces the Offer dimension with a Channel dimension in projects based on the
    MyCustomCampaignApp.xml template. After applying the patch, projects based on the template will be
```

```
        updated the next time they are opened and resaved in Decision Management.</instructions>
        <actions>
            <action type="replaceElement" path="/PredictiveApplication/Dimension[@name='Offer']">
                <NewValue>
                    <Dimension name="Channel" description="ChannelDescriptionText">
                        <Variable name="Capacity" dataType="integer"
                            optimizationInputItem="true" prompt="">
                            <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                                xsi:type="ConstantValueSource">
                                <Value>0</Value>
                            </ValueSource>
                        </Variable>
                        <Variable name="ChannelCost" dataType="double"
                            simulateAction="sum" simulateName="Total Channel Cost"
                            optimizationInputItem="true" prompt="">
                            <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                                xsi:type="ConstantValueSource">
                                <Value>0</Value>
                            </ValueSource>
                        </Variable>
                    </Dimension>
                </NewValue>
            </action>
        </actions>
</patch>
```

deleteElement example

The `deleteElement` patch action type is used for deleting sections of projects. The `path` must point to the specific XML item you want to delete.

```
<patch templateName="MyCustomCampaignApp" templateVersion="8" appliesToVersions="4,5,6,7">
    <instructions>This patch deletes the Offer dimension from projects based on the MyCustomCampaignApp.xml
    template. After applying the patch, projects based on the template will be updated the next time they are
    opened and resaved in Decision Management</instructions>
    <actions>
        <action type="deleteElement" path="/PredictiveApplication/Dimension[@name='Offer']">
        </action>
    </actions>
</patch>
```

setAttribute example

The `setAttribute` patch action type is used for modifying projects. The `path` must point to the specific XML item you want to modify.

```
<patch templateName="MyCustomCampaignApp" templateVersion="8" appliesToVersions="6,7">
    <instructions>This patch changes the data type of CampaignSize to double in projects based on the
    MyCustomCampaignApp.xml template. After applying the patch, projects based on the template will be
    updated the next time they are opened and resaved in Decision Management.</instructions>
    <actions>
        <action type="setAttribute" path="/PredictiveApplication/Dimension[@name='Campaign']/Variable[@name=
        'CampaignSize']" attributeName = "dataType" attributeValue="double">
        </action>
    </actions>
</patch>
```

Important notes

- Use caution when creating XML patch files and applying them. The patch files are extremely flexible and allow you to change virtually anything. You should be very familiar with your XML application template before creating a patch file for it. For example, if you add a new step/tab, make sure you also add the associated user interface elements. Or if you add a dimension, make sure you also add the user interface element for it.
- Disabling a section of the user interface is not supported. It does not remove the objects created using the disabled user interface previously. For example, disabling a section of the user interface may leave behind objects such as rules, which may not be visible anywhere in the user interface, but are still executed at run time.

- If your patch file includes a `replaceElement` action type, be sure the path points to the XML item you want to replace. For example, the incorrect syntax below points to the parent `DimensionSetting` item instead of the `AllocationRuleSection` item to replace:

Incorrect:

```
<action type="ReplaceElement" path="/PredictiveApplication/InterfaceControl/InterfacePages/
DefineStep/DimensionSetting[@name='Channel']">
        <NewValue>
                <AllocationRuleSection enabled="true"/>
        </NewValue>
</action>
```

To replace the `AggregateRuleSection` item properly in this example (to change it from `false` to `true`), you must include the full path to the specific item as seen below:

Correct:

```
<action type="ReplaceElement" path="/PredictiveApplication/InterfaceControl/InterfacePages/DefineStep/
DimensionSetting[@name='Channel']/AllocationRuleSection">
        <NewValue>
                <AllocationRuleSection enabled="true"/>
        </NewValue>
</action>
```

Using the Template Utility

After updating the custom application template XML file and creating an XML patch file as described in the previous section "Before using the Template Utility" on page 74, perform the following steps to run the *Template Utility* and update projects.

1. On the machine where IBM SPSS Collaboration and Deployment Services Repository and IBM Analytical Decision Management is installed, copy the following *.zip* file onto the Windows machine you want to run the utility from.

   C:\Program Files\IBM\SPSS\Deployment\7.0\Server\components\ decision-management\Utilities\ TemplateUtility\SPSS_Decision_Mgmt_17_TemplateUtility.zip

2. Extract the *.zip* file you copied in the previous step. This will result in a folder called *SPSS_Decision_Mgmt_17_TemplateUtility*.

3. To run the *Template Utility*, double-click the file *TemplateUtility.bat* in the folder you extracted in the previous step.

   If you have problems running the utility, verify you have Java 1.6 installed and have a JAVA_HOME system environment variable pointing to your Java bin directory, or that your Path system environment variable points to your Java bin directory. The utility is only supported on Windows. In some cases, you may also need to log on to the Windows machine with administrator authority before running the utility.

4. When prompted, provide your IBM SPSS Collaboration and Deployment Services Repository connection details. This is the repository containing the projects you want to update. You must have Decision Management Administrator authority to use the *Template Utility*. The following options are available, if desired.

   - **Set Credentials.** Leave this box unchecked if you want to enable the single sign-on feature, which attempts to log you in using your local computer username and password details. If single sign-on is not possible, or if you check this box to disable single sign-on, a further screen is displayed for you to enter your credentials.

   - **Ensure secure connection.** Specifies whether a Secure Sockets Layer (SSL) connection should be used. SSL is a commonly used protocol for securing data sent over a network. To use this feature, SSL must be enabled on the server hosting the repository. If necessary, contact your local administrator for details.

5. Click **Browse** and select the XML patch file you created in the previous section "Before using the Template Utility" on page 74. The **Application Name**, **Versions to Upgrade**, **Upgrade to Version**, and **Patch Instructions** sections of the *Template Utility* will display the values defined in your selected XML patch file.

6. Select which end user projects (stream files) to update by moving them from the **Available Projects** section to the **Selected Projects** section.

7. Click **Apply** to apply the updates to all selected projects based on the specified application template. The updates will not be applied until the next time the projects are opened and resaved in IBM Analytical Decision Management.

Note that the *Template Utility* does not apply changes to any projects in the Gallery.

If you ever need to revert to a previous version of a project, you can use IBM SPSS Collaboration and Deployment Services Deployment Manager to do so. Deployment Manager provides full control over versions and labels of files in the repository.

Example scenarios

Example 1: Standard IBM Analytical Decision Management for Claims application

Jane creates a project in release 7 based on the standard, prebuilt IBM Analytical Decision Management for Claims application template packaged with the software, and deploys the project for scoring in her production environment. She does not make any changes to the *ClaimsManagement.xml* application template or stylesheets. After upgrading her environment to the new version 8 of IBM Analytical Decision Management, the following observations can be made:

- The deployed project scores exactly as it did before.
- Jane can open the project in the new version of IBM Analytical Decision Management and run the Test and Simulation features as she did in the previous version, and close the project without saving.
- If the new release did not include any changes to the *ClaimsManagement.xml* application template, she can also modify and save changes to her project.
- If the new release includes changes to the *ClaimsManagement.xml* application template to add support for new features, Jane may choose to update her project using the *Template Utility* to start using those new features in her project.

Example 2: Custom application

Bill creates a project in release 7 based on a custom application template he has created, and deploys the project for scoring in a production environment. He creates his application by copying the standard *ClaimsManagement.xml* application template under a different filename, and modifying the copy. After installing the new version 8 of IBM Analytical Decision Management, his custom application template is preserved and the following observations can be made:

- The deployed project scores exactly as it did before.
- Bill can open the project in the new version of IBM Analytical Decision Management and run the Test and Simulation features as he did in the previous version.
- Because his project uses a custom application template, it was not automatically updated when the new version of IBM Analytical Decision Management was installed. As a result, he can continue to modify and save changes to his project exactly as before.
- If the new software version includes new features Bill wants to use, he will need to manually update his custom template to make use of these features. After doing this, he will then need to update his project using the *Template Utility*. This will also allow him to make use of the new features.

Example 3: Updated application template

Ted creates and deploys a project based on a custom application template he created, and then has subsequently made changes to that custom application template.

After updating the custom template, the deployed project scores exactly as it did before.

Ted can also open the existing project in IBM Analytical Decision Management and run the Test and Simulation features as he did previously. However, if he wants his existing project to pick up the new changes he made to his template, he will need to update his existing project using the *Template Utility*. Note that this is only required for the existing project. Any new projects Ted creates based on the updated template will pick up the changes.

Appendix A. Optimization example

The 6.x versions of IBM Analytical Decision Management supported **prioritization**. With prioritization, the best offer for a given customer can be selected. This process is performed in the context of a single customer, one customer at a time, and without regard to other customers. Starting with IBM Analytical Decision Management version 7, you also have the option of using **optimization**. With optimization, you can calculate the best combination of offers to give to selected customers. This process is performed in the context of all customers at one time.

Optimization is performed using complex mathematical optimization (referred to as *CPLEX* or *IBM ILOG CPLEX* in this appendix). CPLEX offers state-of-the-art performance and a robust optimization engine for solving problems expressed as mathematical programming models.

The goal of optimization is to identify the solution that best meets a specific goal, such as maximizing the revenue from a marketing campaign or minimizing the risk of fraud or churn. The result of optimization is a solution that answers this question across all possible combinations within your population, in the manner that best maximizes the value you want to optimize (such as profit).

Optimization maximizes or minimizes a desired outcome based on a mathematical equation. The **optimization equation** is displayed on the **Optimize** tab in applications that use it. An application will have an Optimize tab if it's configured to use the CPLEX optimization algorithm. The optimization equation describes what the optimization is trying to maximize or minimize. **Constraints** are also used to set limits on variables used in the optimization equation. For example, a constraint called MAX_BUDGET might be used to specify the maximum amount of money that can be spent on a campaign.

Purpose

This appendix provides an example configuration of a custom optimization application. The steps outlined in this appendix are specific to this custom example and are not required for the standard IBM Analytical Decision Management for Campaign Optimization application shipped with the product.

The standard Campaign Optimization application uses a preconfigured Optimization Programming Language (OPL) model file called *CampaignOptimization.mod*. In some cases, the standard Campaign Optimization application and its OPL model might not suit the purposes of the business problem at hand. This appendix walks through an example of a custom OPL model built using IBM ILOG CPLEX Optimization Studio, and a custom IBM Analytical Decision Management optimization application, for a fictional company called **ABC Electric**.

IBM ILOG CPLEX Optimization Studio

IBM ILOG CPLEX Optimization Studio (referred to as *Optimization Studio* in this appendix) provides the fastest way to build efficient optimization models and state-of-the-art applications for the full range of planning and scheduling problems. With its integrated development environment (IDE), descriptive modeling language, and built-in tools, it supports the entire model development process.

Optimization Studio uses a model file as its input. A model file is a text file containing Optimization Programming Language (OPL). OPL provides a natural mathematical description of optimization models. Using high-level syntax for mathematical models produces substantially simpler and shorter code than general-purpose programming languages, reducing the effort and improving the reliability of application development, upgrades, and maintenance.

Some of the examples in this appendix use Optimization Studio to edit, run, and verify CPLEX models independently of IBM Analytical Decision Management.

Initial business problem

For the example used in this appendix, we have a fictitious business called **ABC Electric**. ABC Electric has a fixed number of spools of wire in stock. This wire is needed to complete various wire orders that have been queued up. Each order requires a specified length of wire. ABC Electric knows it does not have enough wire to complete all orders in the queue. Each spool of wire has a fixed length. All spools contain the same type of wire and can be used interchangeably.

They will create a custom IBM Analytical Decision Management application called *Order Optimization* that uses optimization to maximize the number of orders it can complete.

Exercise overview

This appendix discusses the following example tasks:

1. ABC Electric launches IBM Analytical Decision Management to run their custom *OrderOptimization.str* project. This project serves as a starting point and will optimize the maximum number of orders filled.
2. They launch Optimization Studio and familiarize themselves with the IDE.
3. They use Optimization Studio to familiarize themselves with their existing custom model file *OrderOptimization.mod*.
4. They use a text editor to familiarize themselves with their custom *OrderOptimization.xml* application template.
5. They consider a new business requirement that adds a priority to each spool.
6. They update their custom model file to use the spool priority.
7. They update their custom IBM Analytical Decision Management application template XML file to use the new spool priority.
8. They create and run a new project using the new template and note the results.

This appendix assumes the reader is already familiar with the process for configuring applications in IBM Analytical Decision Management and familiar with general Decision Management concepts such as dimensions and child dimensions.

Familiarizing yourself with the Order Optimization application

An ABC Electric employee logs on to IBM Analytical Decision Management in his browser and completes the following tasks:

1. On the applications launch page, from the **Order Optimization** application shortcut box, browse for and open the **OrderOptimization** project from the following location in the repository: */Optimization/OrderOptimization.str*.
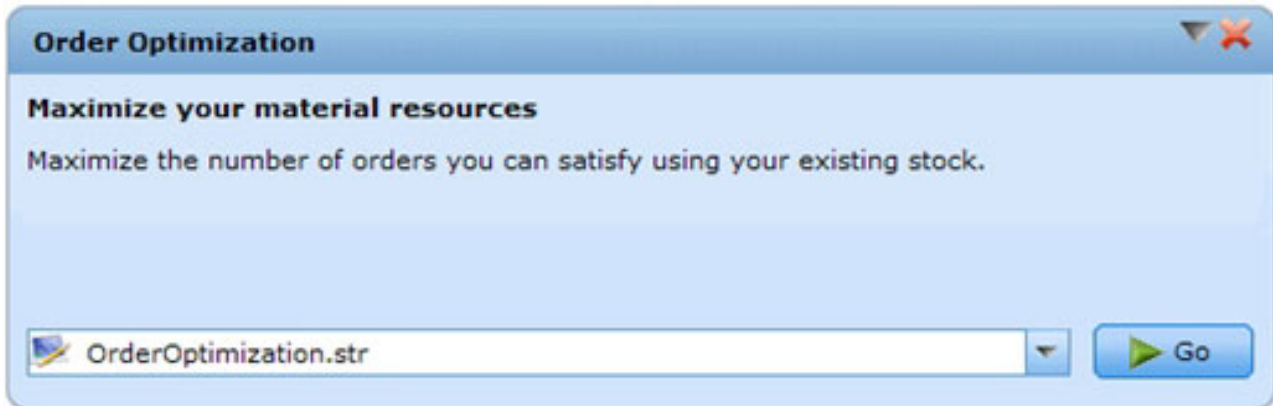
Figure 8. Example custom application

2. On the Data tab, preview the data. There is one row per Order, and a field indicating the length of wire required to fulfill the order.

OrderID	Length
ID00001	113
ID00002	189
ID00003	19
ID00004	76
ID00005	42
ID00006	139
ID00007	68
ID00008	217
ID00009	153
ID00010	82

Figure 9. Data preview

3. On the Define tab:
   - This is a simple application with only a single dimension called *Spool*. There are currently **3** spools defined. There are no allocation rules in this project as there are no child dimensions. Selection

rules for spools are enabled, but there are currently no rules defined. With these settings, all orders can be fulfilled by all spools. In fact, based on just the settings in Define, all orders will be fulfilled by each spool. An order for 10 units of wire will be fulfilled 3 times – resulting in a total of 30 units being allocated.

Figure 10. Spool dimension with three spools

- Run a simulation. This confirms that based on the Define settings, each order is allocated 3 times – once per spool.
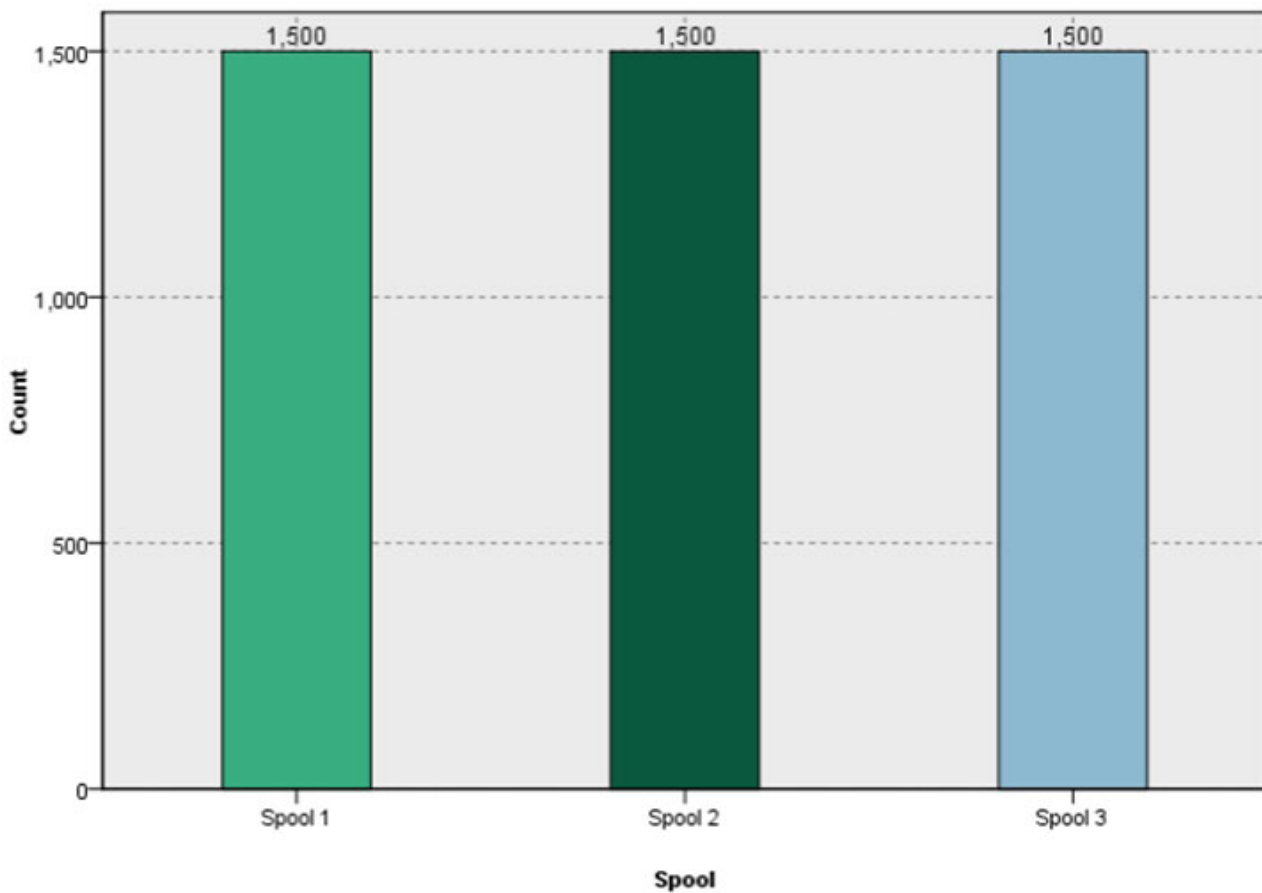
Figure 11. Simulation

4. Close the scenario results and go to the Optimize tab:
   - Note the Order Length variable. This is the length of each order and is set to the length attribute of the Project Data Model.

- Note that each spool has a size. By default, each spool will contain 1000 units of material.
- Expand the optimization equation and note the value. The project maximizes the number of fulfilled orders.
- Expand the constraints. This tells the optimization that the amount of material is limited. The spools have a finite length that can be allocated.
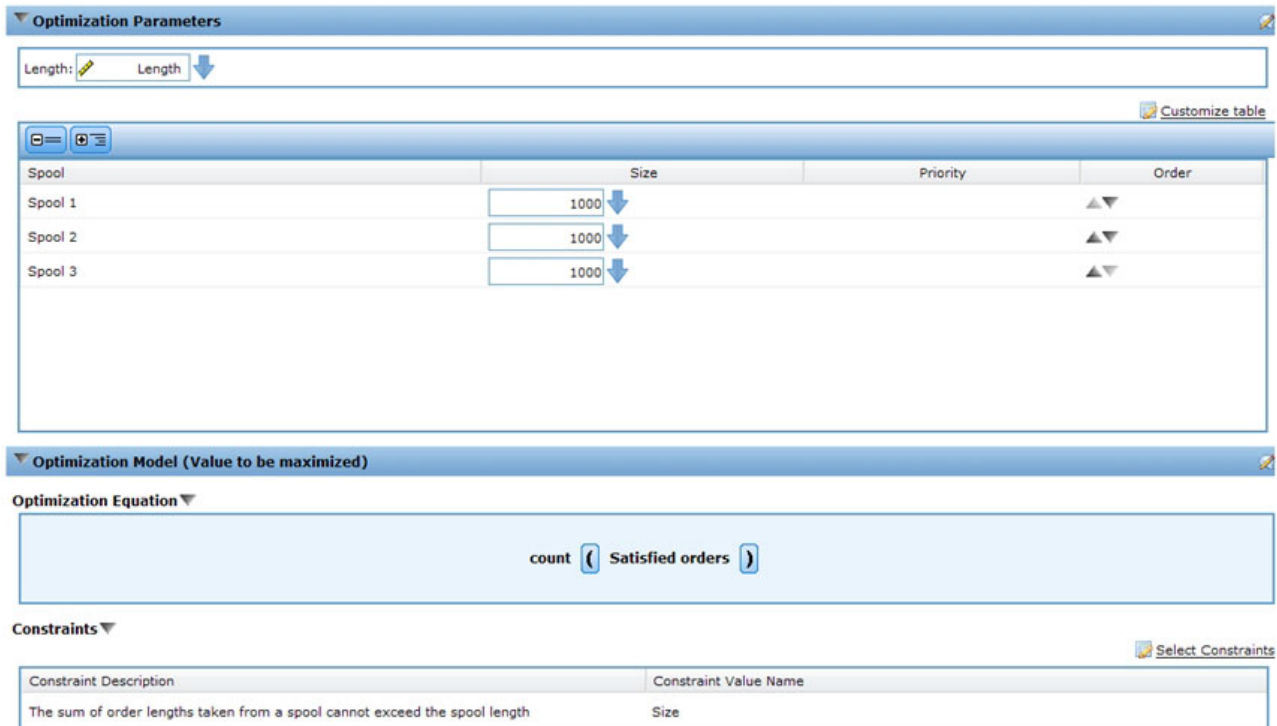
Figure 12. Optimization tab

- Run another simulation. Note that 189 orders out of the 1500 are fulfilled. A total of 2990 units are allocated from the 3 spools.

5. Go to the Deploy tab:
   - In the **Batch Scoring** section, specify **File** as the destination, **Flat File** as the file type, and specify a temporary file (*c:\tempout.txt*) for the **File Location**.
   - In **Specify Output fields**, also select the **Order ID** and **Length** variables so they will be included in the output for reference.
   - Click **Score Now**.
   - Using a text editor, open the score output file that was created (*c:\tempout.txt*). Notice the OrderID of the first record (for example, ID00003). Search the file to see if you can find another record with the same OrderID. Note that additional records with this same OrderID are not found. This means each order is fulfilled just once.

Using IBM ILOG CPLEX Optimization Studio

An expert user at ABC Electric starts Optimization Studio by going to **Start** > **IBM ILOG CPLEX Studio IDE** (or, alternatively, **Start** > **All Programs** > **IBM ILOG** > **CPLEX Optimization Studio** > **CPLEX Studio IDE**). Then he performs the following tasks.

1. He notes there are many sample OPL projects available and navigates through them to find the project called **OrderOptimization**:
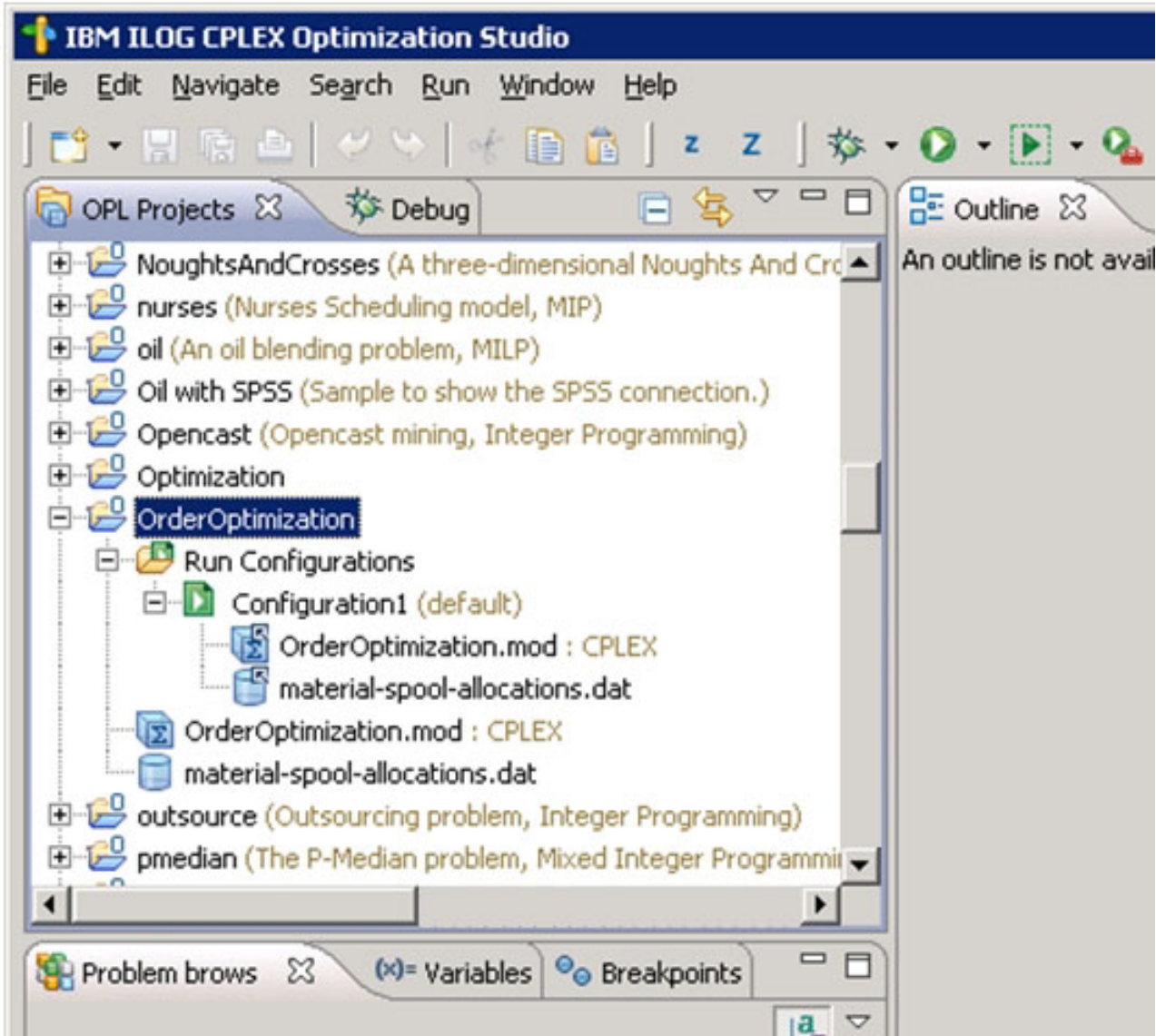
Figure 13. OrderOptimization project in Optimization Studio

- The *OrderOptimization* project contains a model (*.mod*) and a data file (*.dat*). The details of each will be explored later in this document.
- *OrderOptimization.mod* is an OPL model file. This is an exact copy of the OPL model file that ABC Electric uses for IBM Analytical Decision Management in their *OrderOptimization* project.
- *Material-spool-allocations.dat* is a data file. Note that it is similar to (but not the same as) the contents of the input file *material-orders.csv* used in the IBM Analytical Decision Management project.
- *Configuration1* is a run configuration. This allows the OPL project to run within the IDE. The *.mod* and *.dat* files are also listed under the run configuration.

2. Right-click **Configuration1** and select **Run This**. In the lower left **Problem browser** tab, you see the **Data** that was input, the **Decision variables** that were used, the **Decision expressions** (the objective function), and the **Constraints**.
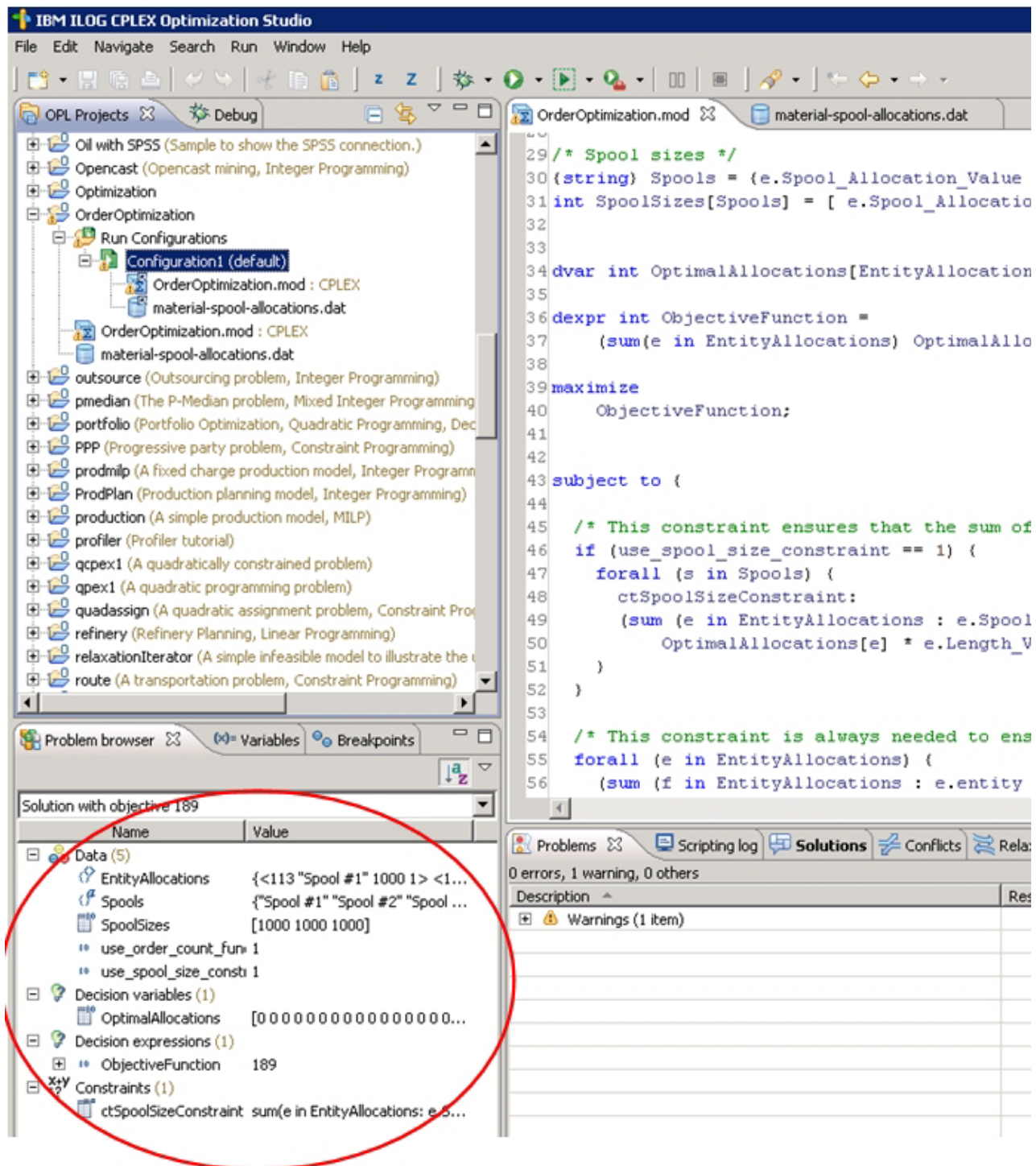
Figure 14. Run Configuration1

3. In the Problem browser, select **OptimalAllocations**. Note that an icon appears with the text **Show data view...**. Click the icon. This opens the Entity Allocations in the main pane. Entity Allocations is the input data derived from the *.dat* file. However, notice there is an extra column called **Value**. The Value column represents the *OptimalAllocation* value. **0** indicates that the record was not part of the optimal solution. **1** indicates that it is part of the optimal solution. The ABC Electric expert scrolls through the output and notices that some records have the value **1**.
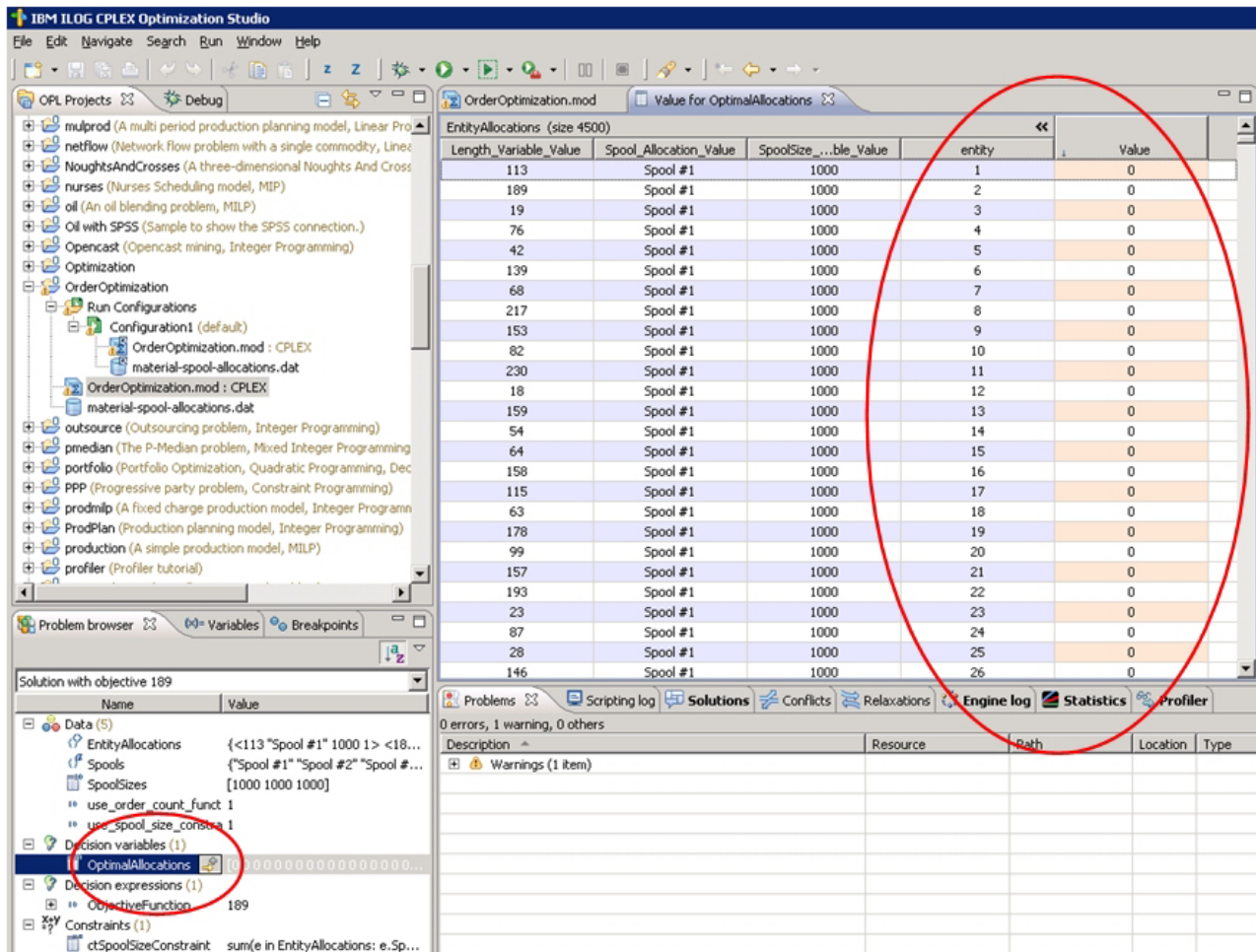
Figure 15. Value column

OrderOptimization.mod

In Optimization Studio, the ABC Electric employee double-clicks *OrderOptimization.mod* to open it. At line 9 he finds the following declarations:

```
tuple entityallocation {
    int Length_Variable_Value;
    string Spool_Allocation_Value;
    int SpoolSize_Variable_Value;

    /* Typically always include the Decision Management-generated entity field */
    int entity;
    /* Only the Decision Management-generated priority field when necessary */
    /* int priority; */
}

{entityallocation} EntityAllocations = ...;
```

In this section, `EntityAllocations` is declared as an input to the optimization. It has a type of `entityallocation` which is declared as a `tuple`. A tuple consists of an ordered list of attributes along with their data types. Taken together, these attributes define a single row of input.

- He opens the *material-spool-allocations.dat* file. On line 5, the values for `EntityAllocations` are being input. The values are assigned in the order they are listed.
- Switching back to *OrderOptimizations.mod* shows two more input variables at lines 23 and 26:

```
int use_spool_size_constraint = ...;
int use_order_count_function = ...;
```

- Unlike the tuple, they consist of a single value. Selecting the *material-spool-allocations.dat* tab again shows the values being set at lines 1 and 3, respectively.
- In *OrderOptimizations.mod*, notice lines 30 and 31:

```
{string} Spools = {e.Spool_Allocation_Value | e in EntityAllocations};
int SpoolSizes[Spools] = [ e.Spool_Allocation_Value : e.SpoolSize_Variable_Value | e in EntityAllocations];
```

  `Spools` is declared as a set of strings that will contain the distinct data values passed in for `Spool_Allocation_Value`. With the supplied *.dat* file as input, it will have the set of values "Spool #1", "Spool #2", "Spool #3".

  `SpoolSizes` is declared as an array that is indexed using the values of `Spools`. This array will contain the `SpoolSize_Variable_Value` of the corresponding spool; 1000 for "Spool #1", etc.

- In *OrderOptimizations.mod*, notice line 34:

```
dvar int OptimalAllocations[EntityAllocations] in 0..1;
```

  `OptimalAllocations` is declared as a decision variable array based on the number of tuples (records) in `EntityAllocations`. The purpose of an OPL model in optimization problems is to find values for the decision variables that satisfy all constraints and optimize a specific optimization equation. In this case, the values will be assigned as 0 or 1.

- Notice line 36-40:

```
dexpr int ObjectiveFunction =
    (sum(e in EntityAllocations) OptimalAllocations[e]);

maximize
    ObjectiveFunction;
```

  This section declares the optimization equation that should be maximized. For each tuple in `EntityAllocations`, CPLEX will assign a value (0 or 1) in `OptimalAllocations` such that the sum is maximized.

- Notice lines 43-59:

```
subject to {

/* This constraint ensures that the sum of lengths cut from a spool
   cannot exceed the spool's size */
   if (use_spool_size_constraint == 1) {
     forall (s in Spools) {
       ctSpoolSizeConstraint:
       (sum (e in EntityAllocations : e.Spool_Allocation_Value == s)
          OptimalAllocations[e] * e.Length_Variable_Value) <= SpoolSizes[s];
     }
   }

 /* This constraint is always needed to ensure that an order is only
      allocated to at most one spool. */
   forall (e in EntityAllocations) {
     (sum (f in EntityAllocations : e.entity == f.entity)
   OptimalAllocations[f]) <= 1;
```

Appendix A. Optimization example   **89**

```
    }
};
```

These are the constraints. The first constraint limits the wire that can be used from each spool based on the spool size. The second constraint prevents the same order from being allocated from multiple spools. Recall earlier, we noticed that Define was allocating all orders to all spools. It is this constraint that causes an order to be filled just once (from one spool). This second constraint is coded in the OPL and is not influenced by any parameters passed in by IBM Analytical Decision Management.

• Notice lines 62-63:

```
float Contribution[EntityAllocations] =
    [ e : OptimalAllocations[e] * 1.0 | e in EntityAllocations];
```

For each tuple in `EntityAllocations`, this array will receive a 1 or 0 depending on whether the tuple is included in the optimal solution or not. When the configuration ran, a warning was given saying this variable was not referenced. It is only used when IBM Analytical Decision Management is calling CPLEX (more on that later).

OrderOptimization.xml

Using a text editor, the ABC Electric employee opens his custom IBM Analytical Decision Management application template *OrderOptimization.xml* located in the installation directory (for example, *C:\IBM\SPSS\Deployment\6.0\Server\components\decision-management\Templates\OrderOptimization.xml*).

• Notice lines 41-42 (`EntityDimension`):

```
<EntityDimension name="Order" description="OrderDescription">
 <Variable name="Length" dataType="integer" description="LengthDescription"
 optimizationInputItem="true" prompt="Order Length" simulateAction="sum"
 simulateName="TotalLengthUsed">
```

Here the entity dimension is declared (as usual). `Length` is declared as an entity variable. The length of the order can be set for each order. All of the attributes (`optimizationInputItem`, `simulateAction`, and `simulateName`) also existed in IBM Analytical Decision Management 6.x and 7.x.

• Notice lines 49-61 (Spool dimension):

```
<Dimension name="Spool" description="SpoolDescription">
 <Property&gt;Name</Property&gt;
 <Property&gt;Category</Property&gt;
 <Property&gt;Organization</Property&gt;
 <Property&gt;Group</Property&gt;
 <Property&gt;Description</Property&gt;

 <Variable name="SpoolSize" dataType="integer" description="SpoolSizeDescription"
 optimizationInputItem="true" prompt="" inherited="false">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:type="ConstantValueSource">
   <Value&gt;1000</Value>
  </ValueSource>
 </Variable>
</Dimension>
```

The dimension `Spool` along with the variable `SpoolSize` is defined. The size of the spool can be set for each spool. Again, this capability also existed in version 6.x and 7.x.

• Notice line 63:

```
<Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path=
"/Applications/OrderOptimization/Optimization/OrderOptimization.mod">
```

The `Optimization` element existed in version 6.x and 7.x. However, some new options were added in version 7:

- The `algorithm` attribute supports a `CPLEX` value. This tells IBM Analytical Decision Management to use CPLEX optimization as opposed to Heuristic optimization.

- The `objectiveValueName` attribute supports an `ObjectiveValue` value. This is effectively used to declare a special variable that will receive the value of the contribution variable that is output from CPLEX. Reference the `contributionVariable` attribute on the `OPLMapping` element described later in this document. On line 88 the objective value (contribution) will be used on an `OutputAttribute` to output the value.

- The `Path` attribute tells IBM Analytical Decision Management where the CPLEX model file is located. The path is relative to the *components\decision-management* directory on the IBM SPSS Collaboration and Deployment Services server.

• Notice lines 64-67:

```
<ObjectiveFunction Domain="integer" Name="OrderCount" description=
"OrderCountDescription">
 <ExpressionFormat format="${FN_count} ( ${SatisfiedOrders} )"/>
 <ExternalUsage controlVariable="use_order_count_function"
 variableType="int" enabledValue="1" disabledValue="0"/>
</ObjectiveFunction>
```

Here the `ObjectiveFunction` is declared in the XML. This element existed in IBM Analytical Decision Management 6.x and 7.x. However, in this application, the actual function is declared in the CPLEX model file. Instead of using the `Expression` child element that is used with the Heuristic algorithm, this application utilizes the `ExpressionFormat` and `ExternalUsage` child elements.

The `ExpressionFormat` element is there mainly for display purposes. Recall that previously the optimization equation was expanded in the IBM Analytical Decision Management user interface. The equation displayed is derived from this element. It is a best practice to make this meaningful and in sync with the CPLEX model file. In addition, the user interface uses this as a cue to indicate that a variable is used in this project and should be displayed on the Optimize tab.

The `ExternalUsage` attribute is used to set control variable values in CPLEX. The value of the `controlVariable` attribute must match a variable as declared in the CPLEX model file. Likewise, the value for `variableType` (int) must match how the variable is declared in the model file.

Optimization Studio can be used to show how they match by finding `use_order_count_function` in the *.mod* file.

The XML allows multiple objective functions to be defined. However, only one may be in use at a time, as selected by the user in IBM Analytical Decision Management. If the user has enabled a particular objective function, then the control variables listed under that function in the XML will be set to the value given for the `enabledValue` attribute. Otherwise, it will get the value for the `disabledValue` attribute.

• Notice lines 69-73:

```
<Constraint type="max" name="ctSpoolSizeConstraint" entityScoped="false"
enabled="true" description="SpoolSizeConstraint">
 <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
 "VariableReferenceBoundary" value="SpoolSize"/>
 <ExpressionFormat format="${FN_sum} ( ${Length} )"/>
 <ExternalUsage controlVariable="use_spool_size_constraint" variableType=
 "int" enabledValue="1" disabledValue="0"/>
<Constraint>
```

The Constraint element existed in versions 6.x and 7.x. Similar to the objective function, the real constraint logic is not in this XML file, but in the model file. The ExpressionFormat is used to display the constraint logic in the IBM Analytical Decision Management user interface. Likewise, the ExternalUsage element behaves as it does under the ObjectiveFunction element.

- Notice line 75-83:

```
<OPLMapping tupleSetVariable="EntityAllocations" contributionVariable=
"Contribution" outputDecisionVariable="ObjectiveFunction">
 <EntityField referenceType="Variable" name="Length" dataReturn="Value"
 tupleField="Length_Variable_Value" tupleFieldType="int"/>
 <EntityField referenceType="DimensionMember" name="Spool" dataReturn=
 "Value" tupleField="Spool_Allocation_Value" tupleFieldType="string"/>
 <EntityField referenceType="Variable" name="SpoolSize" dataReturn="Value"
 tupleField="SpoolSize_Variable_Value" tupleFieldType="int"/>
 <EntityField referenceType="System" name="entity" tupleField="entity"
 tupleFieldType="int"/>

 <!-- Must tell Decision Management how the optimal flag is being returned -->
 <OptimizationOutput name="IsOptimal-Value" valueVariable=
 "OptimalAllocations" variableType="int" thresholdType="equal"
 thresholdValue="1"/>
</OPLMapping>
```

The OPLMapping element is specific to the CPLEX algorithm. It describes how IBM Analytical Decision Management should pass data into and out of the CPLEX engine at run time.

The tupleSetVariable attribute identifies the name of a tuple in the model file. EntityAllocations in the model file shows how it matches.

The contributionVariable attribute describes how each entity record (allocation or tuple) contributes to the optimal solution. For example, in campaign optimization, it describes how much this customer offer is expected to contribute to the profit. The value of the contributionVariable must match the name of a variable (array) in the model file.

The Contribution variable in the model file shows how it matches.

The outputDecisionVariable attribute identifies the objective value as declared in the model file.

The ObjectiveFunction in the model file shows how it matches. The value from this variable is assigned to the objectiveValueName specified on the Optimization element.

The EntityField sub element is used to define the attributes making up the tuple that is in the model file.

The referenceType attribute tells where the value comes from in IBM Analytical Decision Management. It may come from a Variable, DimensionMember, or System. System is used to pass values that are managed internally by IBM Analytical Decision Management.

The name attribute describes which referenceType is passed. If referenceType is set to Variable, it describes the name of the variable. If referenceType is set to DimensionMember, it describes the name of the dimension. If referenceType is set to System, it describes which internal system value to use. The system value entity is used by IBM Analytical Decision Management to uniquely identify each entity.

The tupleField attribute is used to map to the tuple field in the model file.

The tupleFieldType attribute must match the type of the tuple field as defined in the model file. EntityAllocations can be viewed in the model file to see how it matches.

The OptimizationOutput sub element describes how the model returns whether or not the entity record is part of the optimal solution.

The valueVariable and variableType attribute values map to a variable declared in the model.

The thresholdType and thresholdValue attributes are used to create a simple expression against the variable. In this example, the entity is included in the optimal solution if the OptimalAllocations array element is set to 1.

New business requirement

ABC Electric's original business problem was to simply maximize the number of customers. But their business has recently changed slightly. Some of the wire spools hold older wire that will not meet the electrical code requirements of the future. ABC Electric wants to make sure they sell the wire on these old spools first, whenever possible. Their inventory of spools has also changed. Each spool now contains 75000 units. They now have enough to meet all orders.

- Spool #1 contains the newest wire. They want to sell this last.
- Spool #2 contains moderately old wire.
- Spool #3 contains the oldest wire. They want to sell this first.

To satisfy this requirement, they will modify the model by adding a `Priority` to each spool. They will run the new model in Optimization Studio to test and verify it is working, update the IBM Analytical Decision Management XML file to work with the new model, and then use IBM Analytical Decision Management to optimize the orders.

Updating OrderOptimization.mod

1. In Optimization Studio, the expert user at ABC Electric creates a copy of the **OrderOptimization** project in case he makes a mistake later on and wants to revert back:
   - Right-click the project and select **Copy**.
   - Right-click the project again and select **Paste**.
   - Name the new copy *OrderOptimization_original* and click **OK**.

   The expert user then performs the following tasks.

2. From the OrderOptimization project, double-click **material-spool-allocations.dat** to open it.
   - IBM Analytical Decision Management's only way to input data is through the single tuple definition. While OPL supports multiple sources, the integration with IBM Analytical Decision Management only supports a single input tuple variable. Since the ultimate goal is to create a model that will work with IBM Analytical Decision Management, the new priority field must be added to the *.dat* file.
   - The goal is to assign Spool #1 a priority of 1, Spool #2 a priority of 2, and Spool #3 a priority of 3. Using the Find/Replace functions from the **Edit** menu, add a new field between the `Spool_Allocation_Value` and the `SpoolSize_Variable_Value`. Do this by replacing:

     `"Spool #1",` with `"Spool #1",1,`

     `"Spool #2",` with `"Spool #2",2,`

     `"Spool #3",` with `"Spool #3",3,`

     for all records in the file. Your records should now match this format:

     `<158,"Spool #3",3,1000,1500>`

   - Edit the spool size to be 75000 instead of 1000 by replacing `,1000,` with `,75000,`
   - Save the changes.

3. Next, update the tuple to include the new priority that is being input. Open the *OptimizationModel.mod* file from the OrderOptimization project. Add the new field `SpoolSize_Variable_Value` to the `entityallocation` tuple by inserting a line as follows:

```
tuple entityallocation {
    int Length_Variable_Value;
    string Spool_Allocation_Value;
```

```
int SpoolPriority_Variable_Value;   // Higher priority spools get used first
int SpoolSize_Variable_Value;
```

4. Save the model changes and run it by right-clicking **Configuration1** and selecting **Run This**. You are verifying the changes made so far, but the model is not using the new priority field yet.

5. Update the objective function to use the new priority field:

   - In the model file, locate the objective function (the optimization equation):

   ```
   dexpr int ObjectiveFunction =
        (sum(e in EntityAllocations) OptimalAllocations[e]);
   ```

   - Edit the function to include the new priority as follows and then save the model.

   ```
   dexpr int ObjectiveFunction =
        (sum(e in EntityAllocations) (OptimalAllocations[e] *
         e.SpoolPriority_Variable_Value));
   ```

   Multiplying by the priority gives entities with a higher priority a larger value. Since CPLEX is maximizing the value returned by `ObjectiveFunction`, it will try to allocate from the higher priority spools first.

6. Run the configuration again and check the results. After running the configuration, click **Show data view** as you did previously. Scroll through the records. Note there are no allocations from spool 1 since it has the lowest priority.

Updating OrderOptimization.xml

The model file is now working fine in Optimization Studio. The ABC Electric employee will now complete the following tasks to update the IBM Analytical Decision Management application template to work with the new model that uses the spool priority:

1. In case a mistake is made later, save a copy of the original XML file (for example, *C:\IBM\SPSS\Deployment\6.0\Server\components\decision-management\Templates\ OrderOptimization.xml*).

2. Edit *OrderOptimization.xml* to add the priority by adding a new variable to the `Spool` dimension. Insert the following highlighted lines.

```
<Dimension name="Spool" description="SpoolDescription">
 <Property>Name</Property>
 <Property>Category</Property>
 <Property>Organization</Property>
 <Property>Group</Property>
 <Property>Description</Property>

 <Variable name="SpoolSize" dataType="integer" description="SpoolSizeDescription"
 optimizationInputItem="true" prompt="" inherited="false">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
  "ConstantValueSource">
   <Value>1000</Value>
  </ValueSource>

 </Variable>
 <Variable name="SpoolPriority" dataType="integer" description=
 "SpoolPriorityDescription" optimizationInputItem="true" prompt="" inherited="false">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
  "ConstantValueSource>
   <Value>1</Value>
  </ValueSource>
 </Variable>
</Dimension>
```

3. Update `ExpressionFormat` in the `ObjectiveFunction` element. If this isn't done, the IBM Analytical Decision Management user interface will think the variable is not used and will not display it.

```
<ExpressionFormat format="${FN_count} ( ${SatisfiedOrders} * ${SpoolPriority})"/>
```

4. Add the new `EntityField` to the `OPLMapping` by inserting the following highlighted section and then saving the changes:

```
<OPLMapping tupleSetVariable="EntityAllocations" contributionVariable="Contribution"
outputDecisionVariable="ObjectiveFunction">
    <EntityField referenceType="Variable" name="Length" dataReturn="Value" tupleField=
    "Length_Variable_Value" tupleFieldType="int"/>
    <EntityField referenceType="DimensionMember" name="Spool" dataReturn="Value"
    tupleField="Spool_Allocation_Value" tupleFieldType="string"/>
    <EntityField referenceType="Variable" name="SpoolPriority" dataReturn="Value"
    tupleField="SpoolPriority_Variable_Value" tupleFieldType="int"/>
    <EntityField referenceType="Variable" name="SpoolSize" dataReturn="Value"
    tupleField="SpoolSize_Variable_Value" tupleFieldType="int"/>
    <EntityField referenceType="System" name="entity" tupleField="entity"
    tupleFieldType="int"/>
```

5. Finally, the original *OrderOptimization.mod* being used by the *OrderOptimization* application must be replaced with the updated one. In Optimization Studio, locate the updated *OrderOptimization.mod* file. Right-click the *.mod* file and view its properties to locate the path to where the model is currently stored, and copy the *.mod* file.

6. Paste it into *C:\IBM\SPSS\Deployment\6.0\Server\components\decision-management\Applications\ OrderOptimization\Optimization*, overwriting the older version.

   Currently, the XML application template is set to look for the model in the directory *Applications/OrderOptimization/Optimization*:

```
<Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path=
"/Applications/OrderOptimization/Optimization/OrderOptimization.mod">
```

Launching IBM Analytical Decision Management and trying it

1. The ABC Electric employee opens IBM Analytical Decision Management and creates a new project within the Order Optimization application (this will prompt IBM Analytical Decision Management to pick up the changes to the application template XML).
   - On the Data tab, he adds the same CSV input file (*material-orders.csv*). This file doesn't need to be edited because the variable values will come from the variable values set on the Optimize tab.
   - On the Define tab, he adds the three spools: **Spool1**, **Spool2**, and **Spool3**.
   - On the Optimize tab, he sets the **Order Length** variable to be supplied by the **Length** attribute.
   - He sets the size of each spool to **75000**.
   - He sets the priority of each spool: **Spool1** is priority **1**, **Spool2** is priority **2**, and **Spool3** is priority **3**.
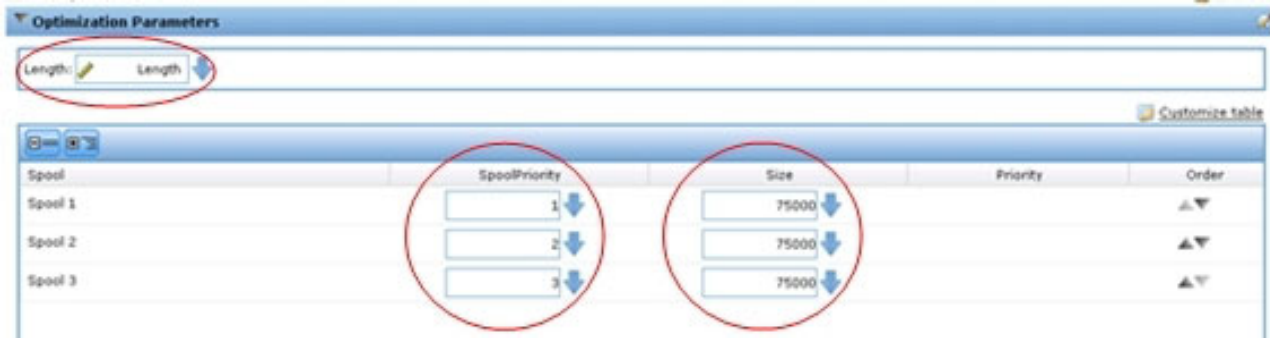
Figure 16. Spool priority

2. He clicks the **Create a new scenario based on current project settings** icon to run a new scenario, and then inspects the results. Only spools 2 and 3 will be used to fill orders.

3. On the Deploy tab, he scores to a temporary file and inspects the results. No orders were filled from spool 1, as intended.

OrderID	Length	Spool	Output-ObjectiveValue	Output-SpoolPriority
ID00002	189.0	Spool2	2.0	2.0
ID00006	139.0	Spool2	2.0	2.0
ID00008	217.0	Spool2	2.0	2.0
ID00009	153.0	Spool2	2.0	2.0
ID00011	230.0	Spool2	2.0	2.0
ID00013	159.0	Spool2	2.0	2.0
ID00016	158.0	Spool2	2.0	2.0
ID00019	178.0	Spool2	2.0	2.0
ID00021	157.0	Spool2	2.0	2.0
ID00022	193.0	Spool2	2.0	2.0

Figure 17. Score results

XML and OPL reference

For reference purposes, this section provides the complete contents of the final custom application template XML file and OPL model file described in the examples throughout this appendix.

OrderOptimization.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
 Licensed Materials - Property of IBM
 IBM SPSS Products: Decision Management
 (C) Copyright IBM Corp. 2010, 2013
 US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP
 Schedule Contract with IBM Corp.
-->

<PredictiveApplication xmlns="http://com.spss.pasw.dms/workspace" templateName=
"OrderOptimizationWithSpoolPriority" templateVersion="1" appsVersion="7.0">

 <InterfaceControl>
  <InterfacePages>
   <ApplicationHome stepIncluded="true" showGallery="true"/>
```

```
  <DataStep stepIncluded="true"/>
  <GlobalSelectionStep stepIncluded="true"/>
  <DefineStep stepIncluded="true" xmlns:xsi= "http://www.w3.org/2001/XMLSchema-instance"
  xsi:type="typeDecisionHierarchyDefineStep">
   <DimensionSetting name="Spool">
         <PlanningSection enableInteractionPoints="false" enableStartEndDates="true"/>
    <SelectionSection enabled="true" enableModels="true"/>
         <AllocationRuleSection enabled="false"/>
         <AggregateRuleSection enabled="false"/>
         <PredictiveModelSection enabled="false"/>
   </DimensionSetting>
  </DefineStep>
  <OptimizeStep stepIncluded="true" enableTest="false">
   <OptimizeMethod enableNumReturnsByIP="true" xmlns:xsi=
   "http://www.w3.org/2001/XMLSchema-instance" xsi:type="PrioritizationOptimization"/>
  </OptimizeStep>
  <DeployScoreStep stepIncluded="true">
   <RealTimeScoring enableInteractiveQuestions="false"/>
   <ImmediateBatchScoring enableScoringOptions="true"/>
  </DeployScoreStep>
  <ReportStep stepIncluded="true"/>
 </InterfacePages>
 <InterfaceFeature id="Collaboration"/>
 <InterfaceFeature id="UploadDownload"/>
 <InterfaceFeature id="MetadataDownload"/>
</InterfaceControl>

<EntityDimension name="Order" description="OrderDescription">
 <Variable name="Length" dataType="integer" description="LengthDescription"
 optimizationInputItem="true" prompt="Order Length" simulateAction="sum" simulateName=
 "TotalLengthUsed">
  <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
  "ConstantValueSource">
   <Value>1</Value>
  </ValueSource>
 </Variable>
</EntityDimension>

<Dimension name="Spool" description="SpoolDescription">
    <Property>Name</Property>
 <Property>Category</Property>
 <Property>Organization</Property>
 <Property>Group</Property>
 <Property>Description</Property>
 <Variable name="SpoolPriority" dataType="integer" description= "SpoolPriorityDescription"
 optimizationInputItem="true" prompt="" inherited="false">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="ConstantValueSource">
    <Value>1</Value>
   </ValueSource>
 </Variable>
 <Variable name="SpoolSize" dataType="integer" description="SpoolSizeDescription"
 optimizationInputItem="true" prompt="" inherited="false">
   <ValueSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
   "ConstantValueSource">
    <Value>1000</Value>
   </ValueSource>
 </Variable>
</Dimension>

<Optimization algorithm="CPLEX" objectiveValueName="ObjectiveValue" path="/Applications/
OrderOptimizationWithSpoolPriority/Optimization/OrderOptimization.mod">
 <ObjectiveFunction Domain="integer" Name="OrderCount" description="OrderCountDescription">
  <ExpressionFormat format="${FN_count} ( ${SatisfiedOrders} * ${SpoolPriority})"/>
  <ExternalUsage controlVariable="use_order_count_function" variableType="int"
  enabledValue="1" disabledValue="0"/>
 </ObjectiveFunction>
```

```
    <Constraint type="max" name="ctSpoolSizeConstraint" entityScoped="false" enabled="true"
    description="SpoolSizeConstraint">
      <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
      "VariableReferenceBoundary" value="SpoolSize"/>
      <ExpressionFormat format="${FN_sum} ( ${Length} )"/>
      <ExternalUsage controlVariable="use_spool_size_constraint" variableType="int"
      enabledValue="1" disabledValue="0"/>
    </Constraint>
<!--
    <Constraint type="max" name="ctSpoolPriorityConstraint" entityScoped="false" enabled="true"
    description="SpoolPriorityConstraint">
      <Boundary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
      "VariableReferenceBoundary" value="SpoolPriority"/>
      <ExpressionFormat format="${FN_sum} ( ${Length} )"/>
      <ExternalUsage controlVariable="use_spool_priority_constraint" variableType="int"
      enabledValue="1" disabledValue="0"/>
    </Constraint>
-->
    <OPLMapping tupleSetVariable="EntityAllocations" contributionVariable="Contribution"
    outputDecisionVariable="ObjectiveFunction">
      <EntityField referenceType="Variable" name="Length" dataReturn="Value" tupleField=
      "Length_Variable_Value" tupleFieldType="int"/>
      <EntityField referenceType="DimensionMember" name="Spool" dataReturn="Value"
      tupleField="Spool_Allocation_Value" tupleFieldType="string"/>
      <EntityField referenceType="Variable" name="SpoolPriority" dataReturn="Value"
      tupleField="SpoolPriority_Variable_Value" tupleFieldType="int"/>
      <EntityField referenceType="Variable" name="SpoolSize" dataReturn="Value"
      tupleField="SpoolSize_Variable_Value" tupleFieldType="int"/>
      <EntityField referenceType="System" name="entity" tupleField="entity"
      tupleFieldType="int"/>

      <!-- Must tell Decision Management how the optimal flag is being returned -->

      <OptimizationOutput name="IsOptimal-Value" valueVariable="OptimalAllocations"
      variableType="int" thresholdType="equal" thresholdValue="1"/>
    </OPLMapping>
  </Optimization>

  <Deployment>
    <OutputAttribute referenceType="DimensionMember" name="Spool" returnValue=
    "Spool.Allocation-Value">Spool</OutputAttribute>
    <OutputAttribute referenceType="Objective" name="ObjectiveValue" returnValue=
    "ObjectiveValue-Value">Output-ObjectiveValue
    </OutputAttribute>
    <OutputAttribute referenceType="Variable" name="SpoolPriority" returnValue=
    "SpoolPriority.Variable-Value">Output-SpoolPriority
    </OutputAttribute>

  </Deployment>
</PredictiveApplication>
```

OrderOptimization.mod

```
/*********************************************
 * OPL 12.4 Order Optimization Model
 * Author: jdoe
 * Creation Date: 02 Oct 2012 at 15:44:18
 *********************************************/
/*** Tuple definition.
     Only a subset of the fields generated by Decision Management need to be
   supplied e.g. the entity, dimension variable name, dimension variable values,
   priority, objective function value. */
tuple entityallocation {
       int Length_Variable_Value;
       string Spool_Allocation_Value;
       int SpoolPriority_Variable_Value;   // Higher priority spools get used first
       int SpoolSize_Variable_Value;
```

```
        /* Typically always include the Decision Management-generated entity field */
        int entity;
        /* Only the Decision Management-generated priority field when necessary */
        /* int priority; */
}
{entityallocation} EntityAllocations = ...;
/* Control variables for constraints */
int use_spool_size_constraint = ...;
/* Control variables for objective functions */
int use_order_count_function = ...;
int use_spool_priority_constraint = ...;
/* Spool sizes */
{string} Spools = {e.Spool_Allocation_Value | e in EntityAllocations};
int SpoolSizes[Spools] = [ e.Spool_Allocation_Value : e.SpoolSize_Variable_Value | e in
EntityAllocations];

dvar int OptimalAllocations[EntityAllocations] in 0..1;
dexpr int ObjectiveFunction =
        (sum(e in EntityAllocations) (OptimalAllocations[e] *
         e.SpoolPriority_Variable_Value));
maximize
        ObjectiveFunction;

subject to {
    /* This constraint ensures that the sum of lengths cut from a spool cannot exceed
   the spool's size */
    if (use_spool_size_constraint == 1) {
        forall (s in Spools) {
            ctSpoolSizeConstraint:
            (sum (e in EntityAllocations : e.Spool_Allocation_Value == s)
                OptimalAllocations[e] * e.Length_Variable_Value) <= SpoolSizes[s];
        }
    }
    /* This constraint is always needed to ensure that an order is only allocated to
   at most one spool. */
    forall (e in EntityAllocations) {
        (sum (f in EntityAllocations : e.entity == f.entity)
      OptimalAllocations[f]) <= 1;
    }
};
/* The Contribution provides a value for each allocation. */
float Contribution[EntityAllocations] =
    [ e : (OptimalAllocations[e] * e.SpoolPriority_Variable_Value) | e in
    EntityAllocations];
```

Appendix B. URL syntax

You can construct your own custom URLs to launch specific IBM Analytical Decision Management projects and bypass the Applications launch page. For example, you may want to open a specific tab of a certain version of an IBM SPSS Modeler Advantage project. The following sections provide details for the URL syntax that can be used.

This may be useful for custom integrations with IBM Analytical Decision Management.

The username, password, pass, and provider parameters

Together, these parameters specify authentication information so the IBM Analytical Decision Management login dialog will not be displayed.

The `username` parameter specifies the user name with which to log in to IBM Analytical Decision Management.

Syntax:
```
username=<user id>
```

The `password` parameter specifies the password.

Syntax:
```
password=<password>
```

The `pass` parameter is an alternative to the `password` parameter, with the difference being that the `pass` parameter uses base 64 encoding for security purposes. 64-bit encoding helps hide the password from plain sight. It is not encrypted. To fully secure the password, you must use SSL.

Syntax:
```
pass=<base 64 encoded password>
```

The `provider` parameter specifies the security provider to use for validating credentials. A value is required for `provider` if the `username` and `password` parameters are used.

Syntax:
```
provider=<provider>
```

Where `<provider>` is one of the following values:
- `Native` for the built-in security provider
- `AD_<name>/<domain>` for Active Directory, where `<name>` is the security provider name within the system and `<domain>` is the DNS namespace
- `ADL_<name>/<domain>` for Active Directory with local override, where `<name>` is the security provider name within the system and `<domain>` is the DNS namespace
- `ldap_<name>` for OpenLDAP, where `<name>` is the security provider name within the system

Special characters, such as spaces, must be escaped.

Examples:
```
http://yourserver:8080/DM/?username=admin&password=mypassword&provider=native
```

The id parameter and version parameter

Together, the `id` parameter and the `version` parameter can identify a specific project (stream file) to open from the repository.

The `id` parameter specifies the repository *object id* of the project to open. See your administrator for help locating the object id of a project. The object id can be obtained by using the various APIs or by using IBM SPSS Collaboration and Deployment Services Deployment Manager client to right-click the file and view its properties.

The `version` parameter specifies the repository *version* of the project to open by using the version marker or the version label. Special characters, such as spaces, must be escaped. Omit this parameter to open the LATEST version of the project.

Syntax:

```
id=<object id>
version=m.<version marker>
version=l.<label>
```

Examples:

```
http://yourserver:8080/DM/?id=091e53590d73db3e0000013a5aea9840bf53&version=m.1:2006-12-04%2020:39:17.995
```

```
http://yourserver:8080/DM/?id=091e53590d73db3e0000013a5aea9840bf53&version=l.firstVersion
```

The tab fragment

The `#T_n` fragment can be used to specify the tab (page) of the IBM Analytical Decision Management application to open. `0` indicates the first tab, `1` the second, etc. `-1` is used to open the Home page. This fragment is optional. If specified, it must be at the end of the URL.

For example, to launch IBM SPSS Modeler Advantage to the Data tab, specify `#T_0`. To launch IBM SPSS Modeler Advantage to the Modeling tab, specify `#T_1`. To launch to the Home page, use `#T_-1`.

Example:

```
http://yourserver:8080/DM/?id=091e53590d73db3e0000013a5aea9840bf53?username=admin&password=mypassword
&provider=native#T_1
```

The datasourcename and datasourcefile parameters

Together, the `datasourcename` parameter and the `datasourcefile` parameter instruct IBM Analytical Decision Management to create a data source with the specified name and source file. Default settings are used for the data source. The `datasourcefile` parameter must specify the full path of the file located on the IBM SPSS Modeler Server machine. When used in conjunction with the `#T_` fragment for launching a specific tab, the data source will become the default data source on the specified tab. For example, `#T_1` would select the data source by default on the second tab of the project and `#T_2` would select the data source by default on the third tab. Note that the project will be opened with the specified data source already created. IBM Analytical Decision Management will create the data source if it does not exist. If it does exist, IBM Analytical Decision Management uses the file specified (replaces what was there previously).

These settings are not saved unless the user explicitly saves the project.

Syntax:

```
&datasourcefile=<full path to data on Modeler Server>
&datasourcename=<name to give data source in UI>
```

Examples:

```
http://yourserver:8080/DM/?id=091e53590d73db3e0000013a5aea9840bf53&datasourcefile= C:\Program Files\IBM\SPSS\
ModelerServer\17\Demos\DRUG1n&datasourcename=DSDrug#T_2
```

The template parameter

This parameter is used to launch IBM Analytical Decision Management and create a new project based on the specified application template. The user is responsible for saving the new project, if desired, with a name and location of their choosing. For example, to launch IBM Analytical Decision Management with a new IBM SPSS Modeler Advantage project:

```
http://yourserver:8080/DM/?template=ModelerAdvantage
```

Template names for the other applications are as follows:

```
template=CampaignOptimization
template=ClaimsManagement
template=CustomerInteractionManagement
template=DemandOptimization
template=ModelerAdvantage
template=PredictiveMaintenance
template=RulesManagement
```

The logout parameter

This parameter is used to force a log out from IBM Analytical Decision Management

Example:

```
http://yourserver:8080/DM/logout
```

Sending login information using the POST method

When sent on the URL as described previously, the username and password will be seen in the URL address bar of the browser. To avoid this being visible in plain sight, the username and password can instead be sent as part of contents in a "post" message.

Following is example HTML that uses JavaScript to pass the username, password, and security provider as part of the POST data. This way, the information is encoded using base 64 encoding. To security encrypt the password, you must use SSL.

A HTML example like this with an associated JavaScript (*.js*) file might be used to add a button to a web page on your company's Intranet that launches a certain IBM Analytical Decision Management project.

```
<html>
<head>
<script type="text/javascript" language="javascript" src="base64.js"></script>
<script type="text/javascript">

function post_to_url(path, parameter, method, user, pass, provider) {
    method = method || "post"; // Set method to post by default, if not specified.

    // The rest of this code assumes you are not using a library.
    // It can be made less wordy if you use one.
    var form = document.createElement("form");
    form.setAttribute("method", method);
    form.setAttribute("action", path + parameter);

    if (user != null && pass != null)
    {
        var userName = Base64.encode(encodeURI(user));
        var password = Base64.encode(encodeURI(pass));

        var userField = document.createElement("input");
        userField.setAttribute("type", "hidden");
        userField.setAttribute("name", "username");
        userField.setAttribute("value", userName);

        form.appendChild(userField);

        var passField = document.createElement("input");
```

```
            passField.setAttribute("type", "hidden");
            passField.setAttribute("name", "pass");
            passField.setAttribute("value", password);

            form.appendChild(passField);
        }

        if (provider != null)
        {
            var providerField = document.createElement("input");
            providerField.setAttribute("type", "hidden");
            providerField.setAttribute("name", "provider");
            providerField.setAttribute("value", provider);

            form.appendChild(providerField);
        }

        document.body.appendChild(form);
        form.submit();

}
</script>


</head>
    <table>
        <tr>
            <td><button onclick="post_to_url('http://localhost:9084/DM/', '', 'post', 'admin', 'spss', 'Native');">
            Test</button> </td>
            <td>Post User Info</td>
            <td>gethost() + ''</td>
        </tr>
    </table>
</html>
```

Appendix C. Accessibility

Accessibility features help users who have a disability, such as restricted mobility or limited vision, to use information technology products successfully. IBM strives to provide products with usable access for everyone, regardless of age or ability. This product uses standard web browser navigation keys.

Because IBM Analytical Decision Management is a browser-based application, accessibility options for disabled users, such as those who are visually impaired, are controlled from your web browser settings. For details about user interface actions, including keyboard shortcuts, screen readers, and so on, refer to your web browser's documentation.

Vendor software

IBM Analytical Decision Management may require use of some vendor software that is not covered under the IBM license agreement. IBM makes no representation about the accessibility features of these products. Contact the vendor for the accessibility information about its products.

IBM and accessibility

See the IBM Human Ability and Accessibility Center for more information about the commitment IBM has to accessibility.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who want to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Software Group
ATTN: Licensing
200 W. Madison St.
Chicago, IL; 60606
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other product and service names might be trademarks of IBM or other companies.

Index