# Table of Contents

**Introduction** 17

**Chapter 1: What's new?** 19  
   New features in version 10.1.0 19  
   Run links 19  
   Changed features in version 10.1.0 19

**Chapter 2: Understanding the IBM Cognos Business Viewpoint software development kit** 21  
   Business Viewpoint software development kit components 21  
   Business Viewpoint components 23  
   Supporting technologies 23  
   Business Viewpoint architecture 25  
   Service-Oriented Architecture 25

**Chapter 3: Using the IBM Cognos Business Viewpoint web service** 29  
   Using the Business Viewpoint web service API 29  
   Access and authentication 30  
   Enabling HTTP sessions in an Axis toolkit 30  
   Sending a request to the Business Viewpoint web service 31  
   Connect to the Business Viewpoint web service 31  
   Log on to the Business Viewpoint web service 32  
   Submit a request to the Business Viewpoint web service 33  
   Process the response 33  
   Log off from the Business Viewpoint web service 33  
   Example: Sending a request to the Business Viewpoint service in Java 34

**Chapter 4: Using the IBM Cognos Software Development Kit with Business Viewpoint** 37  
   Sending a request to the IBM Cognos Dimension Management Service 37  
   Connect to IBM Cognos 37  
   Log on to IBM Cognos 38  
   Submit a request to the Dimension Management Service 39  
   Process the response 41  
   Log off from IBM Cognos 42  
   IBM Cognos Software Development Kit reference information 42  
   Services 42  
   Methods 44  
   Classes 47  
   Example: Using the runSpecification method with the Dimension Management Service in Java 52

**Chapter 5: Business Viewpoint class reference** 55  
   Dimension 55  
   Hierarchy 55  
   Level 55  
   MemberList 56

Licensed Materials – Property of IBM  
Chapter 6: Web service methods reference 59

API methods 59
- execute(xmlApiRequest) 59
- logon(credential) 60
- logoff() 61
- ping() 62
- trustedLogon() 63
Handling exceptions 63
- Java exceptions (Axis toolkit) 63

Part 1: XML API request reference

Chapter 7: XML API request reference 65

action 66
- Content Model 67
- Parent Elements 67

Chapter 8: Application actions reference 69

applicationRoles 70
- Content Model 70
- Parent Elements 70
- What's New 70
class 70
- Content Model 70
- Parent Elements 71
- What's New 71
cognos8Package 71
- Content Model 71
- Parent Elements 71
cognos8Package 71
- Content Model 71
- Parent Elements 71
create 71
- Content Model 71
- Parent Elements 71
- What's New 72
createUser 72
- Content Model 72
- What's New 72
createVersion 72
- Content Model 72
csvFile 72
Chapter 9: Repository actions reference 89

class 90
    Content Model 90
    Parent Elements 90
constraint 90
    Content Model 90
    Parent Elements 90
constraints 91
    Content Model 91
    Parent Elements 91
create 91
    Content Model 91
delete 91
    Content Model 91
find 91
    Content Model 91
get 91
    Content Model 91
hint 91
    Content Model 92
    Parent Elements 92
hints 92
    Content Model 92
    Parent Elements 92
modify 92
    Content Model 92
name 92
    Content Model 92
    Parent Elements 93
objectID 94
    Content Model 94
    Parent Elements 94
property 94
    Content Model 94
    Parent Elements 94
requestedProperties 94
    Content Model 94
Content Model 87
Parent Elements 87
What's New 88
versionName 88
    Content Model 88
    Parent Elements 88
versionType 88
    Content Model 88
    Parent Elements 88
Parent Elements 87
versionDescription 87
    Content Model 87
    Parent Elements 87
Content Model 87
Parent Elements 87
Part 2: Import specification reference

Chapter 10: Import specification

annotation 96
Attributes 96
Content Model 96
Parent Elements 96
What's New 96
crossReference 97
Attributes 97
Content Model 97
Parent Elements 97
dimension 97
Attributes 98
Content Model 98
Parent Elements 98
dynamicHierarchy 98
Attributes 98
Content Model 99
Parent Elements 99
hierarchy 99
Attributes 99
Content Model 99
Parent Elements 99
importSpecification 100
Content Model 100
level 100
Attributes 100
Content Model 100
Parent Elements 100
list 100
Attributes 100
Content Model 100
Parent Elements 101
memberApex 101
Attributes 101
Content Model 101
Parent Elements 101
members 101
Attributes 101
Content Model 101
Parent Elements 102
role 102
Attributes 102
Content Model 103
## Chapter 11: Using import specification

Logging and reusing import actions 105

Typical usage scenario 106

## Chapter 12: Export specification

<table>
<thead>
<tr>
<th>Element</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>dimensionVersion</td>
<td>108</td>
</tr>
<tr>
<td>Attributes</td>
<td>108</td>
</tr>
<tr>
<td>Content Model</td>
<td>108</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>108</td>
</tr>
<tr>
<td>exportSpecification</td>
<td>108</td>
</tr>
<tr>
<td>Attributes</td>
<td>108</td>
</tr>
<tr>
<td>Content Model</td>
<td>109</td>
</tr>
<tr>
<td>hierarchy</td>
<td>109</td>
</tr>
<tr>
<td>Attributes</td>
<td>109</td>
</tr>
<tr>
<td>Content Model</td>
<td>109</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>109</td>
</tr>
<tr>
<td>level</td>
<td>109</td>
</tr>
<tr>
<td>Attributes</td>
<td>109</td>
</tr>
<tr>
<td>Content Model</td>
<td>109</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>110</td>
</tr>
<tr>
<td>list</td>
<td>110</td>
</tr>
<tr>
<td>Attributes</td>
<td>110</td>
</tr>
<tr>
<td>Content Model</td>
<td>110</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>110</td>
</tr>
<tr>
<td>memberSet</td>
<td>110</td>
</tr>
<tr>
<td>Attributes</td>
<td>110</td>
</tr>
<tr>
<td>Content Model</td>
<td>110</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>110</td>
</tr>
<tr>
<td>numLatestVersions</td>
<td>110</td>
</tr>
<tr>
<td>Content Model</td>
<td>110</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>111</td>
</tr>
<tr>
<td>securityAttribute</td>
<td>111</td>
</tr>
<tr>
<td>Attributes</td>
<td>111</td>
</tr>
<tr>
<td>Content Model</td>
<td>112</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>112</td>
</tr>
<tr>
<td>securityMapElement</td>
<td>112</td>
</tr>
<tr>
<td>Attributes</td>
<td>112</td>
</tr>
<tr>
<td>Content Model</td>
<td>112</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>112</td>
</tr>
<tr>
<td>securityRule</td>
<td>112</td>
</tr>
<tr>
<td>Content Model</td>
<td>113</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>113</td>
</tr>
<tr>
<td>securityRulesExport</td>
<td>113</td>
</tr>
<tr>
<td>Content Model</td>
<td>113</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>113</td>
</tr>
<tr>
<td>versionId</td>
<td>113</td>
</tr>
</tbody>
</table>
Part 4: XML API response reference

Chapter 13: XML API response reference

actionResponse 116
  Content Model 116
  association 116
    Attributes 116
    Content Model 116
    Parent Elements 116
  associationCount 116
    Content Model 116
    Parent Elements 117
  associationObjectID 117
    Content Model 117
    Parent Elements 117
  associations 117
    Content Model 117
    Parent Elements 117
  class 117
    Content Model 117
    Parent Elements 117
  classType 117
    Content Model 117
    Parent Elements 118
  create 118
    Content Model 118
    Parent Elements 118
    What's New 118
  createResult 118
    Content Model 118
    Parent Elements 118
  createUserResult 118
    Content Model 118
    Parent Elements 118
  createVersionResult 119
    Content Model 119
    Parent Elements 119
  dataType 119
    Content Model 119
    Parent Elements 119
  delete 119
    Content Model 119
    Parent Elements 119
    What's New 119
  deleteResult 119
    Content Model 120
    Parent Elements 120
deleteUserResult 120
  Content Model 120
  Parent Elements 120
  What's New 120
depthFromRoot 120
  Content Model 120
  Parent Elements 120
description 120
  Content Model 120
  Parent Elements 120
exportDataResult 121
  Content Model 121
  Parent Elements 121
exportSummary 121
  Content Model 121
  Parent Elements 121
exportTarget 121
  Content Model 121
  Parent Elements 121
exportTargets 121
  Content Model 121
  Parent Elements 122
external 122
  Content Model 122
  Parent Elements 122
  What's New 122
findResult 122
  Content Model 122
  Parent Elements 122
getResult 122
  Content Model 122
  Parent Elements 122
importDataResult 123
  Content Model 123
  Parent Elements 123
importSummary 123
  Content Model 123
  Parent Elements 123
isCore 123
  Content Model 123
  Parent Elements 123
isHighlighted 123
  Content Model 123
  Parent Elements 123
isInternal 124
  Content Model 124
  Parent Elements 124
key 124
  Content Model 124
<table>
<thead>
<tr>
<th>Property</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>properties</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>129</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>130</td>
</tr>
<tr>
<td>property</td>
<td>130</td>
</tr>
<tr>
<td>Attributes</td>
<td>130</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>130</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>130</td>
</tr>
<tr>
<td>read</td>
<td>130</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>130</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>130</td>
</tr>
<tr>
<td>What's New</td>
<td>130</td>
</tr>
<tr>
<td>runLinkResult</td>
<td>131</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>131</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>131</td>
</tr>
<tr>
<td>What's New</td>
<td>131</td>
</tr>
<tr>
<td>runPublicationResult</td>
<td>131</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>131</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>131</td>
</tr>
<tr>
<td>sampleObject</td>
<td>131</td>
</tr>
<tr>
<td>Attributes</td>
<td>131</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>132</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>132</td>
</tr>
<tr>
<td>sampleObjects</td>
<td>132</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>132</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>132</td>
</tr>
<tr>
<td>securityIdentity</td>
<td>132</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>132</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>132</td>
</tr>
<tr>
<td>What's New</td>
<td>132</td>
</tr>
<tr>
<td>setObjectPermissionsResult</td>
<td>132</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>132</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>133</td>
</tr>
<tr>
<td>statusDetails</td>
<td>133</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>133</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>133</td>
</tr>
<tr>
<td>What's New</td>
<td>133</td>
</tr>
<tr>
<td>statusMessage</td>
<td>133</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>133</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>133</td>
</tr>
<tr>
<td>surrogateKeyId</td>
<td>133</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>133</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>134</td>
</tr>
<tr>
<td>systemName</td>
<td>134</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>134</td>
</tr>
<tr>
<td>Parent Elements</td>
<td>134</td>
</tr>
<tr>
<td>systemVersion</td>
<td>134</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Model</td>
<td>134</td>
</tr>
</tbody>
</table>
Introduction

IBM® Cognos® Business Viewpoint helps to provide you with one version of the truth for dimensions used in an enterprise's performance management processes. With Business Viewpoint, you have a controlled, collaborative, workflow-oriented business process to manage both manual and automated changes to all data related to how enterprises analyze and manage their business. Business users are given the responsibility and authority to manage dimensions in their areas of responsibility.

This document is intended for use with the IBM Cognos Business Viewpoint software development kit, which allows you to extend your capabilities in modeling and managing your master dimensions. Implementation details using the IBM Cognos Business Viewpoint Web Service and using the IBM Cognos Software Development Kit with Business Viewpoint are described.

Audience

To use the Developer Guide effectively, you should be familiar with the following:

- Web Services Description Language (WSDL) 1.1
- Simple Object Access Protocol (SOAP) 1.1
- XML programming techniques
- IBM Cognos Business Intelligence
- IBM Cognos Framework Manager

Finding information

To find IBM Cognos product documentation on the Web, including all translated documentation, access one of the IBM Cognos Information Centers at http://publib.boulder.ibm.com/infocenter/cogic/v1r0m0/index.jsp. Updates to Release Notes are published directly to Information Centers.

You can also read PDF versions of the product release notes and installation guides directly from IBM Cognos product disks.

Forward-looking statements

This documentation describes the current functionality of the product. References to items that are not currently available may be included. No implication of any future availability should be inferred. Any such references are not a commitment, promise, or legal obligation to deliver any material, code, or functionality. The development, release, and timing of features or functionality remain at the sole discretion of IBM.

Accessibility features

This product does not currently support accessibility features that help users with a physical disability, such as restricted mobility or limited vision, to use this product.
Introduction
Chapter 1: What's new?

This section contains a list of new and changed features for this release. It will help you plan your upgrade and application deployment strategies and the training requirements for your users.

To locate the most current product documentation, go to the IBM® Cognos® Business Intelligence and Performance Management Information Center at http://publib.boulder.ibm.com/infocenter/cogic/v1r0m0/index.jsp and then click the link for either Business Intelligence or Financial Performance Management. In the Search box, type business viewpoint.

New features in version 10.1.0

Listed below are new features since the last release of IBM® Cognos® Business Viewpoint. Links to directly-related topics are included.

Run links

In IBM® Cognos® Business Viewpoint, you can now move fact data into IBM® Cognos® TM1® from a number of external data applications including other IBM Cognos TM1 servers. To move fact data between source and target data sources, you can create a link. After creating a link in Business Viewpoint Studio, you can use the "runLink" (p. 83) action to transfer and update fact data in the target data source.

For more information on moving fact data, see the Business Viewpoint Studio User Guide.

Changed features in version 10.1.0

In earlier releases, to allow IBM® Cognos® Business Viewpoint users to automate tasks, administrators were required to update the configuration of their IBM® Cognos® Business Intelligence environment by migrating files between Business Viewpoint and Business Intelligence installation folders. The dimension management service that handles Business Viewpoint requests is now included in the IBM Cognos BI v10.1.0 installation kit, and was removed from the Business Viewpoint v10.1.0 installation kit.

For information on using the dimension management service and the BI Bus API, refer to the IBM Cognos Software Development Kit Developer Guide.
Chapter 1: What's new?
Chapter 2: Understanding the IBM Cognos Business Viewpoint software development kit

As a developer, you can use the IBM® Cognos® Business Viewpoint software development kit to integrate Business Viewpoint into your existing software and to develop custom solutions. The software development kit provides the ability to automate Business Viewpoint tasks. It allows you to import data from various contributing sources, export data to various targets, create dimension versions, and so on.

Business Viewpoint software development kit components

The Business Viewpoint software development kit consists of a number of individual software components and supporting documentation, including the following:

- WSDL file describing the Business Viewpoint Web Service API
  
  `<installation_location>\sdk\wsdl\BusinessViewpoint.wsdl`

- XML schemas describing the Business Viewpoint XML API
  
  `<installation_location>\sdk\xmlapi\schemas`

- XML API samples
  
  `<installation_location>\sdk\xmlapi\samples`

  A number of samples are provided to demonstrate usage of supported actions and their corresponding responses.

  Note: The schemas and samples must be placed in the same directory.

- Java™ examples (included in subsequent sections of this document)

- Java samples

  The Java sample files are installed in subdirectories under the `<installation_location>\sdk\java` directory. The subdirectories are as follows:

  - **BVExecuteCLI**

    This sample runs Business Viewpoint software development kit actions from the command line interface.

  - **BVExecuteUI**

    This sample runs Business Viewpoint software development kit actions through a user interface.

  - **Common**

    This directory contains classes that are used by more than one Java sample.
The BVExecuteCLI and BVExecuteUI subdirectories contain the following files.

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>build.bat</td>
<td>to build the individual sample on a Microsoft® Windows® operating system</td>
</tr>
<tr>
<td>run.bat</td>
<td>to run the individual sample on a Microsoft Windows operating system</td>
</tr>
<tr>
<td>build.sh</td>
<td>to build the individual sample on a UNIX® or Linux® operating system</td>
</tr>
<tr>
<td>run.sh</td>
<td>to run the individual sample on a UNIX or Linux operating system</td>
</tr>
</tbody>
</table>

In addition, the `<installation_location>\sdk\java` directory contains these files.

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>build-samples.bat</td>
<td>to build all the Java samples on a Microsoft Windows operating system</td>
</tr>
<tr>
<td>build-samples.sh</td>
<td>to build all the Java samples on a UNIX or Linux operating system</td>
</tr>
</tbody>
</table>

The Java samples included in the installation use the Axis 1.4 Web service toolkit. The toolkit jar files shipped with the sample are standard Axis jars. They do not include any IBM Cognos modifications. The jar files are included in the following directory:

`<installation_location>\sdk\java\lib`

- C# .NET samples
  The C# .NET sample, `BVExecuteAction`, is located in the `<installation_location>\sdk\csharp\BVExecuteAction` directory. The executable program is located in `<installation_location>\sdk\csharp\BVExecuteAction\bin`. If you receive an automation error when you launch an `.exe` program, you must recompile the program for your environment.
  This sample sends a run request to the server using a user-specified file, which must contain a valid XML specification.

  Note: In your .NET project, add a Web reference to the Business Viewpoint WSDL file: `<installation_location>\BusinessViewpoint\sdk\wsdl\BusinessViewpoint.wsdl`. The sample includes a reference to `C:\Program Files\cognos\BusinessViewpoint\sdk\wsdl\BusinessViewpoint.wsdl`, which you must update if your installation location differs.
Business Viewpoint components

IBM® Cognos® Business Viewpoint helps to provide you with one version of the truth for dimensions used in an enterprise's performance management processes. Business Viewpoint consists of the following components.

IBM Cognos Business Viewpoint Server
IBM Cognos Business Viewpoint Server renders the IBM Cognos Business Viewpoint Studio interface. It runs requests to the database and connects to other IBM Cognos components.

IBM Cognos Business Viewpoint Studio
IBM Cognos Business Viewpoint Studio is a Web-based tool that allows business users to manage master dimensions by using workflows to approve and validate changes to dimensional data.

IBM Cognos Business Viewpoint Client
Using IBM Cognos Business Viewpoint Client, business users can nominate, or migrate, master dimensional data from external IBM Cognos applications into a Business Viewpoint Studio master repository, subscribe to Business Viewpoint Studio master dimensional data from external IBM Cognos tools or CSV files, and update the data to ensure it is synchronized between the two sources.

IBM Cognos Business Viewpoint for Microsoft Excel
Using IBM Cognos Business Viewpoint for Microsoft® Excel, business users can work with data offline, and import their results to Business Viewpoint Studio.

IBM Cognos Business Viewpoint Command Line Interface (CLI)
The CLI can be used to perform Business Viewpoint Client tasks without accessing the standard interface.

IBM Cognos Business Viewpoint Web Service
The Web service enables you to automate Business Viewpoint tasks. This includes the ability to import data from various contributing sources, export data to various targets, create dimension versions, and so on.

Supporting technologies

The Business Viewpoint software development kit is implemented as a Web service. Developers write software programs that connect remotely to the Web service. Communication is performed through a "Simple Object Access Protocol (SOAP)" (p. 24) message, an XML-based mechanism for exchanging typed information. The SOAP methods supported by the Web service are described in a "Web Service Definition Language (WSDL)" (p. 24) file.

The following explains the different technologies used by the Business Viewpoint software development kit.
Tomcat Application Server

Apache Tomcat is the application server that is used by IBM® Cognos® Business Viewpoint Server to serve dynamic and static content. Tomcat also acts as a host for the Business Viewpoint Web Service.

Web Service

Web Service is a software system that makes itself available over the Internet and uses a standardized XML messaging system. Since all communications with the Web service are accomplished in XML, Web services are not tied to any one operating system or programming language.

Extensible Markup Language (XML)

A general-purpose specification for creating custom markup languages. It is classified as an extensible language because it allows its users to define their own elements. Its primary purpose is to help information systems share structured data, particularly over the Internet, and it is used both to encode documents and to serialize data.

Web Service Definition Language (WSDL)

An XML-based language used to provide a model for describing Web services. It is a document that describes a Web service and how to access it by specifying the operations or methods that it exposes. It is often used in combination with SOAP and XML schemas to provide Web services over the Internet.

Simple Object Access Protocol (SOAP)

A protocol used for accessing a Web service. It allows applications to exchange XML-based messages and information over HTTP. It is platform and language independent and therefore provides a way to communicate between applications running on different operating systems, with different technologies and programming languages.

The SOAP message consists of the following components:

- An HTTP header, required as part of transmitting a request using that protocol.
- A SOAP action section that is used to identify the target service for the request.
- A SOAP header and body that contains the information required to complete the request.
The Business Viewpoint system is built on a three-tiered service-oriented architecture that consists of the following:

- **Web Client**
  The Business Viewpoint Web client tier, called Business Viewpoint Studio, is a framework and a set of cross-browser UI components (implemented in HTML and JavaScript or any other scripting language supported within the browser’s runtime environment). The presentation framework provides visualization of the information to the user and implements interaction with the user. All communication with the server-side data goes through the UI components except for the software development kit client, which interacts directly with the Business Viewpoint Web Service.

- **Application Server**
  The Business Viewpoint application server tier contains one Tomcat server. A Business Viewpoint Server runs on the Tomcat application server. It is a collection of JSP pages and server-side Java classes that provide data to the UI components. Business Viewpoint Server queries, creates, and modifies data in the data repository. Business Viewpoint Server includes a Web service that provides a public interface to the server.

- **Data Repository**
  The Business Viewpoint data tier consists of the data repository. The data repository is a relational database that Business Viewpoint uses to store data. Business Viewpoint supports IBM® DB2®, Oracle®, and Microsoft® SQL Server databases.

**Service-Oriented Architecture**

The Business Viewpoint architecture includes a generic service, the Business Viewpoint Web Service, that provides access to the Business Viewpoint application functionality through a Web Service API. The clients can use the Web Service API to automate recurring tasks, such as importing data, publishing data, dimension versioning, and so on.
The Business Viewpoint Web Service executes tasks using a synchronous protocol, which waits for a request to finish before accepting another request.

IBM Cognos Business Viewpoint components communicate with each other, and with any additional applications integrated with Business Viewpoint, in the following way:

- The Business Viewpoint Studio communicates with the Application Server through the internal JSON API over HTTP.
- Business Viewpoint Server is implemented using Java and runs on a Tomcat application server. Business Viewpoint Server includes a Web service that provides a public interface to the Server. The Web service is described in a WSDL file, which defines the operations and methods for accessing the Web service.
- Software development kit client applications can read the WSDL file to determine the methods that are available on the server. Any special datatypes used are embedded in the WSDL file in the form of XML Schema. The software development kit client can use a SOAP protocol to call one of the methods listed in the WSDL file.
- Business Viewpoint Server communicates with the data repository through JDBC.
- In Business Viewpoint Studio, a direct connection can be established to an IBM Cognos BI Server. Approved dimensions can be published directly to the IBM Cognos BI Server as a
package. To use the package, a data source connection to the Business Viewpoint data repository must be created in IBM Cognos BI Server.

- Lists and hierarchies can be imported into Business Viewpoint Studio from external data sources that include: CSV files, Microsoft Excel worksheets, and IBM Cognos packages.

- Business Viewpoint can publish dimensions to an external Framework Manager file. Framework Manager can then import the file and use it to create a new project or update an existing project. The model can then be published to the IBM Cognos BI Server.

- The Business Viewpoint Client can connect to other Cognos products like IBM Cognos Transformer and IBM Cognos TM1®. Business Viewpoint Client communicates with Business Viewpoint Server using an internal JSON API over HTTP.
Chapter 2: Understanding the IBM Cognos Business Viewpoint software development kit
Chapter 3: Using the IBM Cognos Business Viewpoint web service

IBM® Cognos® Business Viewpoint web service enables you to automate Business Viewpoint tasks. It includes a generic web service called BusinessViewpointService. All functional areas of the product are available through this one web service. The clients can use the web service API to automate recurring tasks, such as importing data, publishing data, dimension versioning, and so on.

BusinessViewpointService is integrated with Business Viewpoint Server. The web service is available whenever the server is running and does not require any special configurations. You cannot start and stop the service independently from Business Viewpoint Server.

To manually test if the service is available, you can open the service URL in a browser and the service will respond with its own name if it is available.

The web service is available at:

http://host:port/bv/services/BusinessViewpointService

where host is the Business Viewpoint server host name, and port is the Business Viewpoint server port (9410 by default).

The web service API is described in a WSDL file which is an XML-based document. The WSDL file describes how to access the web service by calling the methods that it exposes. This file can be obtained directly from the web service:


The WSDL file is also available in the Software Development Kit at:

<installation_location>/sdk/wsdl/BusinessViewpoint.wsdl

Using the Business Viewpoint web service API

BusinessViewpointService supports WSDL, SOAP, XML, and HTTP. Clients can implement an software development kit application in any programming language that supports these standards. A typical development life cycle for an software development kit application is as follows:

- Choose the implementation platform and language.

- Choose a SOAP toolkit for the client-side code. These toolkits are also known as web service toolkits. This step is optional if your development platform of choice includes a standard toolkit.
<table>
<thead>
<tr>
<th>Language / Platform</th>
<th>SOAP / web service toolkits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java™</td>
<td>Apache Axis</td>
</tr>
<tr>
<td></td>
<td>Apache Axis2</td>
</tr>
<tr>
<td></td>
<td>Sun JAX-RPC reference impl.</td>
</tr>
<tr>
<td></td>
<td>Sun JAX-WS reference impl.</td>
</tr>
<tr>
<td>C# .NET or VB .NET</td>
<td>The toolkit is included in the Microsoft® .NET connection software.</td>
</tr>
<tr>
<td>Perl</td>
<td>SOAP::Lite</td>
</tr>
</tbody>
</table>

- Optionally, use the tools included in the SOAP toolkit to generate client-side code (for example, client proxy code) based on the WSDL file. The generated code turns native language constructs, such as Java™ function calls, into SOAP messages. The toolkit transmits the SOAP messages to the web service using the HTTP protocol. This step is optional since many SOAP toolkits can generate the SOAP messages dynamically, based on the supplied WSDL file.

- Develop custom application logic that relies either on the generated client-side code, or the WSDL file.

**Access and authentication**

The Business Viewpoint application does not support anonymous access. The software development kit application must authenticate itself using the logon SOAP method by supplying valid user credentials as a parameter to the logon method.

An software development kit user account must be assigned to either the Modeler or Administrator role. If the user account is not assigned to either role, the logon method fails. The logon authentication process is as follows:

1. The software development kit application calls the logon SOAP method.
2. The web service authenticates the user.
3. If the user is successfully authenticated, the web service stores the user session details in the HTTP session maintained by the Tomcat application server.
4. The Tomcat application server sets the JSESSIONID cookie in the HTML response.
5. The application includes a JSESSIONID cookie in subsequent SOAP calls.

**Enabling HTTP sessions in an Axis toolkit**

Business Viewpoint web service uses HTTP sessions to maintain software development kit user sessions. The software development kit application must use a web service toolkit that supports
HTTP sessions. Most major Web toolkits support this technology. Some Web toolkits enable HTTP sessions by default, while others require the application developer to explicitly enable HTTP sessions.

The following code demonstrates how to enable HTTP sessions within the Axis toolkit:

```java
import com.cognos.bv.webservice.BusinessViewpointLocator;
BusinessViewpointLocator bvServiceLocator = new BusinessViewpointLocator();
bvServiceLocator.setMaintainSession( true );
```

### Sending a request to the Business Viewpoint web service

This section explains how to send a request from an software development kit client to the Business Viewpoint web service.

Using the web service includes the following tasks:

- "Connect to the Business Viewpoint web service" (p. 31).
- "Log on to the Business Viewpoint web service" (p. 32).
- "Submit a request to the Business Viewpoint web service" (p. 33).
- "Process the response" (p. 33).
- "Log off from the Business Viewpoint web service" (p. 33).

### Connect to the Business Viewpoint web service

Applications connect to the Business Viewpoint web service by creating a service proxy object and using it to issue a request.

When you send a connection request to the Business Viewpoint web service, you provide the server URL in the following form:

```
http://host:port/bv/services/BusinessViewpointService
```

After connecting, you can use the `ping()` method to ensure that the web service is running. This is a quick way to send a SOAP request to the service without logging on. If the action is successful, the web service returns the following echo response:

"Business Viewpoint Web Service is active."

If the web service is not available, the method throws an exception.

### Example: Connecting to the Business Viewpoint web service in Java

```java
import com.cognos.bv.webservice.BusinessViewpointLocator;
import com.cognos.bv.webservice.BusinessViewpointService;

BusinessViewpointLocator bvServiceLocator = new BusinessViewpointLocator();
bvServiceLocator.setMaintainSession( true );

URL theURL = new URL( "http://bvhost:9410/bv/services/BusinessViewpointService" );
BusinessViewpointService bvService = bvServiceLocator.getBusinessViewpointService( theURL );
```
Log on to the Business Viewpoint web service

Applications using the Business Viewpoint web service do not require a special account. Applications log on to the Business Viewpoint web service using a valid user account defined by a Business Viewpoint administrator. The user account must have sufficient privileges to perform the desired XML API action. XML API actions are subject to the security rules created in the Business Viewpoint Studio for that user.

Anonymous access is not supported.

You must include your credentials during the logon process. The credentials contain information about the identity of the user, typically a user name and password.

The credential parameter of the logon(credential) method contains XML-encoded authentication information. The XML must conform to the <installation_location>\sdk\xmlapi\BV_UserCredential.xsd file. If the toolkit you are using does not escape reserved characters in XML strings that are sent to the Business Viewpoint Server, you must explicitly escape these characters in the XML credentials string in your application.

Example: Logging on to the Business Viewpoint service in Java

```java
String username = ...;
String password = ...;
StringBuilder credential = new StringBuilder();

credential.append( "<credential>" );
credential.append( "   <username>" + username + "</username>" );
credential.append( "   <password>" + password + "</password>" );
credential.append( "</credential>" );

try {
    bvService.logon( credential.toString() );
} catch ( RemoteException e ) {
    // handle the exception, for example
    throw new RuntimeException( "Logon operation failed", e );
}

// success
```
Submit a request to the Business Viewpoint web service

The Business Viewpoint web service uses HTTP sessions. Software development kit developers are required to use a web service toolkit that maintains HTTP sessions. The toolkit must be able to extract a JSESSIONID cookie from the HTTP response (from Tomcat server) and set a JESESSIONID cookie in the subsequent HTTP request.

When submitting a request, XML should be passed to the `execute()` method as a regular string. The XML request must conform to the Business Viewpoint XML API Request schema. For more information, see "XML API request reference" (p. 65).

Example: Sending a request to the Business Viewpoint service in Java

```java
// Load the spec from an XML file or generate it "on the fly" in the Java code
String xmlRequest = ...;

String xmlResponse = null;

try {
    xmlResponse = bvService.execute( xmlRequest );
} catch ( RemoteException e ) {
    // handle the exception, for example
    throw new RuntimeException( "Execute operation failed", e );
}

// success: parse the XML response
```

Process the response

Once a request is submitted and the synchronous conversation status is complete, the requested information is returned in a string containing URL-encoded XML. The XML specification conforms to the Business Viewpoint XML API Response schema. For more information, see "XML API response reference" (p. 115).

If the synchronous conversation takes a long time to run, the user session might timeout but the method should still be able to return the result. The user will need to log in again to send another request.

You might need to decode the result in order to access the returned information.

Note: Some SOAP toolkits (Java™ Axis) automatically encode and decode the XML. If the toolkit cannot perform the encoding and decoding, the software development kit developer must handle URL-encoding and decoding.

Log off from the Business Viewpoint web service

Use the `logoff()` method to terminate the user session. An application logs off only once to end the session for all namespaces.
Example: Logging off from the web service in Java

```java
try {
    // do not call logoff() if logon() failed
    bvService.logoff();
} catch ( RemoteException e ) {
    // handle the exception, for example
    throw new RuntimeException( "Logoff failed", e );
}
```

If the `logoff()` method is not used and Tomcat does not see any new session requests coming in, the user session will timeout after 30 minutes, which is the default Tomcat timeout. Users can change the default timeout in the Business Viewpoint Web application descriptor file (`<installation_location>/webapps/bv/WEB-INF/web.xml`). To change the default timeout, add a `<session-config>` section as a child of the `<web-app> root element:

```xml
<web-app>
  ...
  <session-config>
    <session-timeout>60</session-timeout>
  </session-config>
  ...
</web-app>
```

The `session-timeout` element defines the default session timeout interval for all sessions created in this Web application. The specified timeout must be expressed in a whole number of minutes. Session timeout applies to all Web application sessions, including interactive Business Viewpoint Studio sessions. Do not change the default interval unless absolutely necessary.

Example: Sending a request to the Business Viewpoint service in Java

The following Java™ code demonstrates how to use the `execute(xmlApiRequest)` method with the Business Viewpoint service. It is a concatenation of the examples presented earlier in this chapter.

```java
import com.cognos.bv.webservice.BusinessViewpointLocator;
import com.cognos.bv.webservice.BusinessViewpointService;

BusinessViewpointLocator bvServiceLocator = new BusinessViewpointLocator();
bvServiceLocator.setMaintainSession( true );

URL theURL = new URL( "http://bvhost:9410/bv/services/BusinessViewpointService" );
BusinessViewpointService bvService = bvServiceLocator.getBusinessViewpointService( theURL );

String EXPECTED_PING_RESPONSE = "Business Viewpoint Web Service is active.";
String pingResponse = null;
try {
    pingResponse = bvService.ping();
```
catch ( RemoteException e )
{
    throw new RuntimeException( "Remote exception", e );
}

if ( ! pingResponse.equals( EXPECTED_PING_RESPONSE ) )
{
    throw new RuntimeException( "Unexpected ping response: " + pingResponse );
}

String username = ...;
String password = ...;
StringBuilder credential = new StringBuilder();

    credential.append( "<credential>" );
    credential.append( " <username>" + username + "</username>" );
    credential.append( " <password>" + password + "</password>" );
    credential.append( "</credential>" );

try
{
    bvService.logon( credential.toString() );
}
catch ( RemoteException e )
{
// handle the exception, for example
    throw new RuntimeException( "Logon operation failed", e );
}

// success
...

// Load the spec from an XML file or generate it "on the fly" in the Java code
String xmlRequest = ...;

String xmlResponse = null;
try
{
    xmlResponse = bvService.execute( xmlRequest );
}
catch ( RemoteException e )
{
// handle the exception, for example
    throw new RuntimeException( "Execute operation failed", e );
}

// success: parse the XML response
...

try
{
    // do not call logoff() if logon() failed
    bvService.logoff();
}
catch ( RemoteException e )
{
// handle the exception, for example
    throw new RuntimeException( "Logoff failed", e );
}
Chapter 4: Using the IBM Cognos Software Development Kit with Business Viewpoint

This section shows how to use the IBM® Cognos® Software Development Kit to automate Business Viewpoint tasks. To use this product, you must configure IBM Cognos Business Intelligence to run the dimension management service. For more information, see the Business Viewpoint Server Installation and Configuration Guide.

To effectively use the IBM Cognos Software Development Kit, you must first understand its architecture and functionality. For more information, see the IBM Cognos Software Development Kit Developer Guide.

Sending a request to the IBM Cognos Dimension Management Service

This section explains how to send a request from the software development kit client to the IBM Cognos server.

Using the dimension management service involves the following tasks:

- "Connect to IBM Cognos" (p. 37).
- "Log on to IBM Cognos" (p. 38).
- "Submit a request to the Dimension Management Service" (p. 39).
- "Process the response" (p. 41).
- "Log off from IBM Cognos" (p. 42).

Connect to IBM Cognos

Applications using the BI Bus API connect to IBM Cognos by creating an IBM Cognos service class object and using it to issue a request. When you send a connection request to the IBM Cognos server, you provide the server URL. If you are connecting through IBM Cognos Connection, you launch your Web browser and supply the gateway URL provided by your administrator in the following format:

http://host:port/cognos8/cgi-bin/cognos.cgi

You cannot use this method of access with your software development kit applications because the gateway is configured by default to use the following URI:

http://localhost:9300/p2pd/servlet/dispatch/ext

This causes the gateway to block requests from software development kit applications.

Depending on your network configuration, set up your software development kit application to do one of the following:
• connect directly to the dispatcher using the internal dispatcher URI (http://localhost:9300/p2pd/servlet/dispatch)

• connect to an additional dedicated gateway that is configured to connect to the dispatcher using the internal dispatcher URI. You must configure appropriate security for this gateway. This method is useful when the software development kit application is outside of a network firewall.

Note: Do not change your main gateway to use the internal dispatcher URI. Doing so reduces the security of the IBM Cognos portal and studios.

For more information about the internal dispatcher URI, see the IBM Cognos BI Installation and Configuration Guide.

Example: Connecting to IBM Cognos Dimension Management Service in Java

```java
private DimensionManagementService_ServiceLocator dimensionManagementServiceLocator = null;
private DimensionManagementService_Port dimensionManagementService = null;
public static String DMS_URL = "http://localhost:9300/p2pd/servlet/dispatch";
dimensionManagementServiceLocator = new DimensionManagementService_ServiceLocator();
...
try {
    java.net.URL serverURL = new java.net.URL(DMS_URL);
dimensionManagementService = dimensionManagementServiceLocator.
    getDimensionManagementService(serverURL);
} ...
// catch exceptions
```

Example: Connecting to IBM Cognos Dimension Management Service in C# .NET

```csharp
try {
    dimensionManagementService1 dmsRS = new dimensionManagementService1();
dmsRS.Url = "http://localhost:9300/p2pd/servlet/dispatch";
...
}
// catch exceptions
```

Log on to IBM Cognos

Applications using the BI Bus API might log on to IBM Cognos using an account defined by an authentication provider. For information about securing capabilities, see Managing Capabilities in the IBM Cognos Software Development Kit Developer Guide.

You must include your credentials with the logon. The credentials contain information about the identity of an IBM Cognos user, typically a user name and password. The credentials parameter of the logon(credentials, roles) method contains authentication information encoded as XML.

If you configured IBM Cognos with multiple namespaces, you can log on to all available namespaces using a BI Bus API program, provided that you have valid credentials for these namespaces. You cannot log on to the same namespace multiple times.
Note: If your environment has Secure Sockets Layer (SSL) applied, all logon messages sent to and from IBM Cognos are secured based on the SSL configuration in IBM Cognos.

Step

- Use the `logon(credentials, roles)` method to log on to a configured namespace.

Example: Logging on to IBM Cognos in Java

If you are writing your application in Java™, log on to IBM Cognos using the following code:

```java
StringBuffer credentialXML = new StringBuffer();
credentialXML.append("<credential>");
credentialXML.append("<namespace>");
credentialXML.append(namespace);
credentialXML.append("</namespace>");
credentialXML.append("<username>");
credentialXML.append(uid);
credentialXML.append("</username>");
credentialXML.append("<password>");
credentialXML.append(pwd);
credentialXML.append("</password>");
credentialXML.append("</credential>");
...
XmlEncodedXML credentialXEX = new XmlEncodedXML();
credentialXEX.setValue(credentialXML.toString());
connection.getCMService().logon(credentialXEX, null);
```

Example: Logging on to IBM Cognos in C# .NET

If you are writing your application in C#, log on to IBM Cognos using the following code:

```csharp
credentialXML.AppendFormat("<namespace>{0}</namespace>", userNamespace);
credentialXML.AppendFormat("<username>{0}</username>", userName);
credentialXML.AppendFormat("<password>{0}</password>", userPassword);
credentialXML.Append("</credential>");
//The csharp toolkit encodes the credentials
string encodedCredentials = credentialXML.ToString();
xmlEncodedXML xmlEncodedCredentials = new xmlEncodedXML();
xmlEncodedCredentials.Value = encodedCredentials;
searchPathSingleObject[] emptyRoleSearchPathList = new searchPathSingleObject[0];
c8CMS.logon(xmlEncodedCredentials, null);
```

Submit a request to the Dimension Management Service

Several steps are required to successfully submit a request to the dimension management service.

Steps

1. Prepare the BI Bus Header.

   The biBusHeader defines the class for the SOAP 1.1 header entry used by IBM Cognos BI. This header entry must be included in all BI Bus API SOAP messages.

   When a connection is made to the content manager, the BI bus header is created. You must ensure that the dimension management service also contains that header.
Example: Preparing the BI Bus Header in Java

```java
if (dimensionManagementService != null) {
    BiBusHeader header = (BiBusHeader) ((Stub) dimensionManagementService).getHeaderObject("","biBusHeader");
    if (header == null) {
        BiBusHeader cmHeader = (BiBusHeader) ((Stub) cmService).getHeaderObject("","biBusHeader");
        ((Stub) dimensionManagementService).setHeader("","biBusHeader", cmHeader);
    }
}
```

2. Prepare the XML specification.

The `runSpecification(specification, parameterValues, options)` method runs a task based on a specification supplied by the user. Options and parameter values are not derived from an object in the content store in this case.

Example: Preparing the XML specification in Java

```java
DimensionManagementServiceSpecification bvSpec = new DimensionManagementServiceSpecification();
StringBuffer requestXML = new StringBuffer();
requestXML.append("<?xml version="1.0" encoding="UTF-8"?>");
requestXML.append("<create>");
requestXML.append("<class>Dimension</class>");
requestXML.append("<constraints>");
requestXML.append("<constraint>");
requestXML.append("<property>Name</property>");
requestXML.append("<value>D1</value>");
requestXML.append("</constraint>");
requestXML.append("</constraints>");
requestXML.append("</create>");
requestXML.append("</action>");
String sXML = requestXML.toString();
bvSpec.setValue( new Specification( sXML ) );
```

3. Send the request.

When you use the `runSpecification(specification, parameterValues, options)` method to execute an object, you call the service that implements that method directly. In this situation, you use the `runSpecification(specification, parameterValues, options)` method implemented by `dimensionManagementService` to create a request to run the task.

Example: Sending a request to the IBM Cognos Dimension Management Service in Java

```java
ParameterValue[] parameters = new ParameterValue[] {};
AsynchOptionInt primaryThreshold = new AsynchOptionInt();
```
primaryThreshold.setName( AsynchOptionEnum.primaryWaitThreshold );
primaryThreshold.setValue( 0 );

AsynchOptionInt secondaryThreshold = new AsynchOptionInt();
secondaryThreshold.setName( AsynchOptionEnum.secondaryWaitThreshold );
secondaryThreshold.setValue( 0 );

Option[] options = new Option[2];
options[0] = primaryThreshold;
options[1] = secondaryThreshold;

AsynchReply reply = null;

try {
    reply = dimensionManagementService.runSpecification( bvSpec, parameters, options );
    while ( !reply.getStatus().equals( AsynchReplyStatusEnum.complete ) &&
        !reply.getStatus().equals( AsynchReplyStatusEnum.conversationComplete ) )
    {
        reply = dimensionManagementService.wait( reply.getPrimaryRequest(), new
ParameterValue[]{}{}, new Option[]{}{} );
    }
} catch ( RemoteException e )
{
    // Exception-handling code
}

### Process the response

After a request is submitted and the asynchronous conversation status is complete, an asynchDetailMIMEAttachment object is returned in the details property of the AsynchReply class. The requested information is always returned inline, in base64-encoded format.

You need to retrieve the result and decode it in order to view the returned information.

#### Example: Retrieving a response from the IBM Cognos Dimension Management Service in Java

AsynchDetail[] details = reply.getDetails();
for ( AsynchDetail detail : details )
{
    if ( detail instanceof AsynchDetailMIMEAttachment )
    {
        try
        {
            AsynchDetailMIMEAttachment replyDetail = (AsynchDetailMIMEAttachment) detail;
            String replyXML = new String( replyDetail.getData(), "UTF-8" );
        }
        catch ( UnsupportedEncodingException e )
        {
            return "UnsupportedEncodingException occurred when trying to decode the response";
        }
    }
}
Log off from IBM Cognos

Use the `logoff()` method to terminate the user session and remove the passport from the CAM object.

An application logs off only once to end the session for all namespaces.

If the `logoff()` method is not called, the user session will timeout after a specified period of inactivity or after a preconfigured time span, regardless of the level of activity.

IBM Cognos Software Development Kit reference information

Dimensions can be created, modified, deleted, retrieved, imported, and exported by using the `runSpecification(specification, parameterValues, options)` method to send the request.

This section describes the IBM Cognos Software Development Kit services, methods, and classes that are used to access Business Viewpoint.

Services

The IBM Cognos architecture includes a number of services for interacting with and managing runnable objects. Services exist for each of the major components of the product. This chapter describes the Web services defined by IBM Cognos to access Business Viewpoint. The services listed in this chapter communicate using the BI Bus API.

Each service shares a set of generic methods for running objects associated with the service. Users send requests directly to the appropriate service, based on the class of the object.

To support this architecture, services execute tasks using a generalized asynchronous protocol. The protocol uses a consistent mechanism for passing parameters to standardize how tasks are executed. Classes and methods exist to support this consistent process of executing tasks. Asynchronous conversations are managed to optimize overall performance.

Services in Java

In Java™, the behavior protocols for each of the services are defined by a set of interfaces. For each BI Bus API service, there are two corresponding Java interface definitions: the `<service>_Service` interface and the `<service>_Port` interface.

The implementation of the `<service>_Service` interface is a class named `<service>_ServiceLocator`. This locator class contains implementations for the methods named in the `<service>_Service` interface. Use the methods in the `<service>_ServiceLocator` class to acquire the port information for the associated service.

The implementation of the `<service>_Port` interface is a class named `<service>Stub`. This stub class contains the implementations for the methods named in the `<service>_Port` interface. Use the methods in the `<service>Stub` class to access the functionality provided by the service.

For each service, there is an additional class called `<service>_Type` that is used to manipulate configuration settings for the associated service.
The following table shows the classes that are associated with the dimension management BI Bus API service.

<table>
<thead>
<tr>
<th>Java Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DimensionManagementService_ServiceLocator</td>
<td>Implements the DimensionManagementService_Service interface. Use this class to retrieve the port information for the dimension management service.</td>
</tr>
<tr>
<td>DimensionManagementServiceStub</td>
<td>Implements the DimensionManagementService_Port interface. Use this class to call methods provided by the dimension management service.</td>
</tr>
<tr>
<td>DimensionManagementService_Type</td>
<td>Use this class to manipulate configuration settings for the dimension management service.</td>
</tr>
</tbody>
</table>

**Services in C# .NET**

In C# .NET, there are two C# .NET classes for each of the BI Bus API Services. These classes are named `<service>` and `<service>1`. One class contains the implementation for the service methods and the other is used to manipulate configuration settings for the associated service. The following table shows the classes that are associated with the dimension management BI Bus API service.

<table>
<thead>
<tr>
<th>C# .NET Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dimensionManagementService1</td>
<td>Use to call methods provided by the dimension management service.</td>
</tr>
<tr>
<td>dimensionManagementService</td>
<td>Use to manipulate configuration settings for the dimension management service.</td>
</tr>
</tbody>
</table>

**dimensionManagementService**

Defines the service responsible for dimension management. The dimension management service acts as a gateway to the Business Viewpoint application.

The dimension management service has a set of configuration parameters defined by the properties of the `dimensionManagementService`.

The dimension management service performs the following tasks when a `runSpecification` call is made:

- Extracts the Business Viewpoint XML API request from `dimensionManagementServiceSpecification`.

- Sends the XML API request to the Business Viewpoint application which executes the request.

- Receives the XML API response from the Business Viewpoint application.

- Prepares and returns the response.
Available Methods

asynch Method Set

cancel( conversation )
release( conversation )
runSpecification(specification, parameterValues, options)
wait( conversation, parameterValues, options )

Runs Instances Of Classes
dimensionManagementServiceSpecification

Methods

Use the BI Bus API methods to send requests to the IBM Cognos dimension management service. These methods are defined as operations in the IBM Cognos Web Service Definition Language (WSDL) file.

For information about permissions and capability requirements for methods, see Security Prerequisites for Requests in the IBM Cognos Software Development Kit Developer Guide and the individual method descriptions provided in this section.

Methods in Java

In Java™, the methods provided by each BI Bus API service are defined by an interface named <service>_Port, which in turn is implemented by a class named <service>Stub. You access BI Bus API methods using an instance of this class. For example, methods that belong to dimensionManagementService are available through instances of the dimensionManagementServiceStub class. For more information about the Java classes that correspond to the dimension management BI Bus API service, see “Services” (p. 42).

runSpecification(specification, parameterValues, options)

This method initiates an asynchronous conversation with a service. The return value of the method contains information about the conversation. This information can be used by an application to determine the status of the conversation, or which secondary tasks are allowed.

If the request takes longer than the timeout value, a status of working might be returned. You should then call wait(conversation, parameterValues, options) until the status is complete or conversationComplete, at which time the requested data is returned.

You might make secondary requests after using this method, depending on the server response. For more information about the asynchronous conversation status and secondary requests in asynchronous conversations, see the section on Secondary Requests in the IBM Cognos Software Development Kit Developer Guide.

Signatures

- Java and Apache Axis

C#.NET

```csharp
public asyncReply runSpecification(asynchSpecification specification, parameterValue[] parameterValues, option[] options)
```

dimensionManagementService

To initiate an asynchronous conversation with the `dimensionManagementService`, you must call the `runSpecification(specification, parameterValues, options)` method. This method sends a request to execute a task in Business Viewpoint.

Provide a `dimensionManagementServiceSpecification` object for the specification parameter. An `asynchDetailMIMEAttachment` object is returned in the details property of the `asyncReply` class when the asynchronous conversation status is complete. The requested information is always returned inline, in base64-encoded format. The response schema for this specification is described in "XML API response reference" (p. 115).

The following capability rule is enforced for this method:

<table>
<thead>
<tr>
<th>Capabilities</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>canUseSpecifications</td>
<td><code>dimensionManagementServiceSpecification</code></td>
</tr>
</tbody>
</table>

To use this method, all conditions specified by any row must be satisfied:

- The object must be a member of one of the specified classes. If a class is not specified, then the class of the object is not used to determine whether the user can execute the method.
- The user must have all specified capabilities.

Method Set

This method belongs to the following method set:

- async method set

Input Parameters

Use the following parameters when calling this method.

**specification**

Defines the specification associated with the request.

Class: `asyncSpecification`

**parameterValues**

Specifies the parameter values for the request.
Parameter values can also be specified in other locations. For more information, see *Specifying Options and Parameters* in the IBM Cognos Software Development Kit Developer Guide.

**Class:** `parameterValue` [[]]

**options**
Specifies the options for the request.

Options can also be specified in other locations. For more information, see *Specifying Options and Parameters* in the IBM Cognos Software Development Kit Developer Guide.

**Class:** `option` [[]]

**Return Values**
This method returns the following values.

**result**
Returns the state of the asynchronous conversation. Possible asynchronous conversation states are defined in the `asynchReplyStatusEnum` enumeration set.

Any data requested can be obtained by examining the `details` property of the `asynchReply` class.

**Class:** `asynchReply`

**Example: Using the `runSpecification(specification, parameterValues, options)` Method with the Dimension Management Service in Java**

The following Java™ code snippet demonstrates how to use the `runSpecification(specification, parameterValues, options)` method with the dimension management service.

```java
ParameterValue[] parameters = new ParameterValue[ ] { }

AsynchOptionInt primaryThreshold = new AsynchOptionInt();
primaryThreshold.setName( AsynchOptionEnum.primaryWaitThreshold );
primaryThreshold.setValue( 0 );

AsynchOptionInt secondaryThreshold = new AsynchOptionInt();
secondaryThreshold.setName( AsynchOptionEnum.secondaryWaitThreshold );
secondaryThreshold.setValue( 0 );

Option[ ] options = new Option[2];
options[0] = primaryThreshold;
options[1] = secondaryThreshold;

AsynchReply reply = null;
try {
    reply = dimensionManagementService.runSpecification( bvSpec, parameters, options );
    while ( !reply.getStatus().equals( AsynchReplyStatusEnum.complete ) && !reply.getStatus().equals( AsynchReplyStatusEnum.conversationComplete ) )
    {
        reply = dimensionManagementService.wait( reply.getPrimaryRequest(), new ParameterValue[ ]{}, new Option[ ]{} );
    }
} catch ( RemoteException e )
{

```
AsynchDetail[] details = reply.getDetails();
for ( AsynchDetail detail : details )
{
    if ( detail instanceof AsynchDetailMIMEAttachment )
    {
        try
        {
            AsynchDetailMIMEAttachment replyDetail = (AsynchDetailMIMEAttachment) detail;
            String replyXML = new String( replyDetail.getData(), "UTF-8" );
        }
        catch ( UnsupportedEncodingException e )
        {
            return "UnsupportedEncodingException occurred when trying to decode the response";
        }
    }
}

Classes
The IBM Cognos content store is a hierarchical collection of objects. A single instance of the root class, represented in a search path by a slash (/), is at the top of the hierarchy.

The classes of these objects are described in this section of the document.

Classes in Java
In the Java™ toolkit, you cannot manipulate the properties of the classes listed in this section by dereferencing instances of these classes. Instead you use accessor methods. The names of these accessor methods are derived from the corresponding property names.

For example, if a class has properties propX, in Java™, these properties are manipulated using the methods getpropX() and setpropX().

Content Manager containment relationships
The following table shows the supported classes and the classes of objects that they can contain.

<table>
<thead>
<tr>
<th>Container Class</th>
<th>Contained Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>dimensionManagementService</td>
<td>runTimeState</td>
</tr>
<tr>
<td></td>
<td>systemMetricThresholds</td>
</tr>
</tbody>
</table>

asynchReply
Defines the properties of the asynchronous responses returned by a dimension management service.

Properties
This class has the following properties.
**details**
Contains the detailed reply information from the service.
This property
- is read-only
- has items that must be of class `asynchDetail`

Class: `asynchDetail[]`

**primaryRequest**
Returns the request that initiated the asynchronous conversation. The request might be updated by the server to facilitate processing on a different server.

Normally this property is returned only when the `status` property is complete to improve performance and reduce network load. The option `alwaysIncludePrimaryRequest` can be used to force this value to be returned.
This property
- is read-only

Class: `asynchRequest`

**secondaryRequests**
Contains the secondary requests that are available for processing as part of the asynchronous conversation.
This property
- is read-only
- has items that must be of class `asynchSecondaryRequest`

Class: `asynchSecondaryRequest[]`

**status**
Specifies the status of the last request sent to the server for the asynchronous conversation.
This property
- is read-only
- has values that are defined in the `asynchReplyStatusEnum` enumeration set

**dimensionManagementService**
Defines runtime configuration parameters for the `dimensionManagementService`.
Values for many of the properties of this class can be acquired from the parent object. You can use property acquisition to simplify the configuration of IBM Cognos installations.
A dispatcher creates an instance of this class within its associated dispatcher object if the object does not already exist.
Instances of this type should be represented as leaf nodes in a tree structure. These objects can be manipulated independent of their containing object.

This class

- inherits properties from the `uiClass` class

**Contained by Instances of Class**

`dispatcher`

**Contains Instances of Classes**

`runTimeSate, systemMetricThresholds`

**Properties**

This class has the following properties.

**advancedSettings**

Specifies advanced configuration information in XML format.

The settings can be changed at run time, without having to restart the server. However, changing these settings using the software development kit requires advanced XML programming skills.

For more information, see the appendix *Advanced Configuration Settings* in the IBM Cognos Software Development Kit *Developer Guide*.

This property

- can be acquired from a containing object

**Class:** `antTypeProp`

**dimsAffineConnections**

Specifies the number of connections that a dimensionManagement service process can use to execute high affinity requests during non-peak hours. This property limits the number of these requests that can be executed concurrently by a dimensionManagement service process.

High affinity requests are requests that are closely associated with a particular process. These requests are usually executed faster than low affinity requests.

If the request is sent to a different process, that process usually requires more time to execute the request because it must perform all the activities performed by the process that received the previous request.

For more information about request affinity, see the IBM Cognos Business Intelligence *Architecture and Deployment Guide*.

This property

- has a default value of 1
- must contain a value greater than or equal to 1
- can be acquired from a containing object
Class: initProp

**dimsAuditLevel**
Specifies the auditing level for the dimensionManagement service.
This property
- has a default value of `minimal`
- has values that are defined in the `auditLevelEnum` enumeration set
- can be acquired from a containing object

Class: auditLevelEnumProp

**dimsExecutionTimeLimit**
Specifies the maximum number of seconds that a task can run before being canceled by the dispatcher.
The dispatcher logs an error (DPR-ERR-2087) indicating that the task execution was canceled due to the execution time limit set being exceeded. A secondary request made against a conversation that has exceeded the time limit returns an error message.
Use a value of 0 when you want to allow the task to complete execution, regardless of the amount of time necessary.
This property
- has a default value of 0
- can be acquired from a containing object

Class: intProp

**dimsMaximumProcesses**
This property is not used and might be deprecated in a future release.

**dimsNonAffineConnections**
Specifies the number of connections that a dimensionManagement service process can use to execute low affinity requests during non-peak hours. This property limits the number of low affinity requests that can be executed concurrently by a dimensionManagement service process.

Low affinity requests establish the context for requests that might follow by caching information. Low affinity requests usually take longer to execute than subsequent high affinity requests. There are no benefits to sending low affinity requests to a particular process because these requests do not use cached information.

For more information about request affinity, see the IBM Cognos Business Intelligence Architecture and Deployment Guide.
This property
- has a default value of 4
- must contain a value greater than or equal to 1


- can be acquired from a containing object

Class: intProp

**dimsPeakAffineConnections**

Specifies the number of connections that a dimensionManagement service process can use to execute high affinity requests during peak hours. This property limits the number of these requests that can be executed concurrently by a dimensionManagement service process.

High affinity requests are requests that are closely associated with a particular process. These requests are usually executed faster than low affinity requests.

If the request is sent to a different process, that process usually requires more time to execute the request because it must perform all the activities performed by the process that received the previous request.

For more information about request affinity, see the IBM Cognos Business Intelligence Architecture and Deployment Guide.

This property

- has a default value of 1
- must contain a value greater than or equal to 1
- can be acquired from a containing object

Class: intProp

**dimsPeakNonAffineConnections**

Specifies the number of connections that a dimensionManagement service process can use to execute low affinity requests during peak hours. This property limits the number of low affinity requests that can be executed concurrently by a dimensionManagement service process.

Low affinity requests establish the context for requests that might follow by caching information. Low affinity requests usually take longer to execute than subsequent high affinity requests. There are no benefits to sending low affinity requests to a particular process because these requests do not use cached information.

For more information about request affinity, see the IBM Cognos Business Intelligence Architecture and Deployment Guide.

This property

- has a default value of 4
- must contain a value greater than or equal to 1
- can be acquired from a containing object

Class: intProp
**dimsQueueLimit**

Specifies the number of seconds that a request for the dimensionManagement service can be queued before it exceeds the timeout period.

This property

- has a default value of 240
- must contain a value greater than or equal to 5
- can be acquired from a containing object

Class: intProp

**runningState**

Specifies the running state of the service.

This property

- has values that are defined in the runningStateEnum enumeration set

Class: runningStateEnumProp

**dimensionManagementServiceSpecification**

Defines the type for dimension management service specifications.

This specification is used with the dimensionManagementService to describe a request to create, modify, delete, import, and export a dimension. It can also describe requests to find dimensions or to retrieve a specific dimension.

For more information about the request schema for this specification, see "XML API request reference" (p. 65).

This class can be used with the following method:

<table>
<thead>
<tr>
<th>Action</th>
<th>Mode</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run</td>
<td>All</td>
<td>runSpecification(specification, parameterValues, options)</td>
</tr>
</tbody>
</table>

This class

- inherits properties from the asynchSpecification class

**Example: Using the runSpecification method with the Dimension Management Service in Java**

The following Java™ code snippet demonstrates how to use the runSpecification(specification, parameterValues, options) method with the dimension management service.

```java
private DimensionManagementService_ServiceLocator dimensionManagementServiceLocator = null;
private DimensionManagementService_Port dimensionManagementService
```
public static String DMS_URL = "http://localhost:9300/p2pd/servlet/dispatch";
dimensionManagementServiceLocator = new DimensionManagementService_ServiceLocator () ;
...
try
{
    java.net.URL serverURL = new java.net.URL (DMS_URL);
dimensionManagementService = dimensionManagementServiceLocator.
getDimensionManagementService (serverURL);
...
} // catch exceptions
if ( dimensionManagementService != null )
{
    BiBusHeader header = (BiBusHeader) ((Stub) dimensionManagementService).
getHeaderObject("", "biBusHeader" );
    if ( header == null )
    {
        BiBusHeader cmHeader = (BiBusHeader) ((Stub) cmService).getHeaderObject
("", "biBusHeader" );
        ((Stub) dimensionManagementService).setHeader("", "biBusHeader",
cmHeader);
    }
}

DimensionManagementServiceSpecification
dvSpec = new DimensionManagementServiceSpecification();
StringBuffer requestXML = new StringBuffer();
requestXML.append("<?xml version="1.0" encoding="UTF-8"?>");
requestXML.append( "<action xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://developer.cognos.com/schemas/bv/xmlapi/request/1/"
xsi:schemaLocation="http://developer.cognos.com/schemas/bv/xmlapi/request/1/
BV_XmlApiRequest.xsd">" );
requestXML.append("<create>");
requestXML.append("<class>Dimension</class>");
requestXML.append("<constraints>");
requestXML.append("<constraint>");
requestXML.append("<property>DisplayName</property>");
requestXML.append("<value>D1</value>");
requestXML.append("</constraint>");
requestXML.append("</constraints>");
requestXML.append("</create>");
requestXML.append( "</action>" );
String sXML = requestXML.toString();
bvSpec.setValue( new Specification( sXML ) );
ParameterValue[]
parameters = new ParameterValue[] {};

AsynchOptionInt primaryThreshold = new AsynchOptionInt();
primaryThreshold.setName( AsynchOptionEnum.primaryWaitThreshold );
primaryThreshold.setValue( 0 );

AsynchOptionInt secondaryThreshold = new AsynchOptionInt();
secondaryThreshold.setName( AsynchOptionEnum.secondaryWaitThreshold );
secondaryThreshold.setValue( 0 );

Option[] options = new Option[2];
options[0] = primaryThreshold;
options[1] = secondaryThreshold;
AsynchReply reply = null;

try {
    reply = dimensionManagementService.runSpecification( bvSpec, parameters, options );
    while ( !reply.getStatus().equals( AsynchReplyStatusEnum.complete ) && 
           !reply.getStatus().equals( AsynchReplyStatusEnum.conversationComplete ) )
    {
        reply = dimensionManagementService.wait( reply.getPrimaryRequest(), new 
                                              ParameterValue[][], new Option[][] );
    }
}
catch ( RemoteException e )
{
    // Exception-handling code
}

AsynchDetail[] details = reply.getDetails();
for ( AsynchDetail detail : details )
{
    if ( detail instanceof AsynchDetailMIMEAttachment )
    {
        try
        {
            AsynchDetailMIMEAttachment replyDetail = (AsynchDetailMIMEAttachment) detail;

            String replyXML = new String( replyDetail.getData(), "UTF-8" );
        }
        catch ( UnsupportedEncodingException e )
        {
            return "UnsupportedEncodingException occurred when trying to decode the 
response";
        }
    }
}
IBM® Cognos® Business Viewpoint contains many classes of objects that are required to properly model your master dimension. This section provides a brief definition for the key Business Viewpoint classes that are essential to the understanding of the application object model. For more information, refer to the Business Viewpoint Studio User Guide.

**Dimension**

A dimension is a container that holds data organized into hierarchies. It is a broad grouping of descriptive data about a major aspect of a business, such as products, customers, or locations. Each dimension includes members in one or more hierarchies. Distinct lists of members can be used as levels in one or more hierarchies.

Dimensions can be conformed. A conformed dimension has a single definition that is reused or shared across multiple data subject areas. Physically, these subject areas can include multiple cubes or multiple relational star schemas. Conformed dimensions provide a useful design approach for dimensions that are widely used throughout the enterprise, for example product or time. This is because common definitions for common dimensions allow data from different subject areas to be meaningfully compared.

**Hierarchy**

A hierarchy represents a collection of dimensional members organized into a tree structure. A hierarchy can be dynamic or static.

Dynamic hierarchies contain members that are pointers to the physical members in one or more lists and thus the members of the hierarchy are updated automatically when you change the information in the list. Use a dynamic hierarchy when the structure of the hierarchy is constant but members are added, updated, or deleted. For example, a Customer hierarchy contains customers' names and addresses. This structure does not change but more customers are added and addresses change.

Static hierarchies are self-contained hierarchies that contain the physical members that you import or add manually. You import members that are contained in lists or IBM Cognos packages. Use a static hierarchy when you want to experiment with the structure of a hierarchy. For example, you want to see the impact of moving a product to several different product lines.

For more information on hierarchies, refer to the Business Viewpoint Studio User Guide.

**Level**

A level represents related data within a hierarchy. Levels define the structure of hierarchies in a dimension. For example, a geographical dimension might contain levels for country, region, and city.
MemberList

A memberList, also referred to as a list, is most often used as the basis for dynamic hierarchies. You can also use lists for lookup attributes as well as using the lists on their own. Changes that you make to the list are reflected automatically in the dynamic hierarchies and in the lookup attributes. Changes that you make to the dynamic hierarchies are reflected automatically in the lists with the exception of attributes. If you create an attribute in a dynamic hierarchy, the attribute appears only in the dynamic hierarchy.

MemberSet

A memberSet, also referred to as a set, is a filter that you can reuse.

For example, the Products dimension contains all products in all product lines. You create a set for each product line.

Member

A member is an item within a hierarchy, level, list, or set.

Attribute

Attributes are the characteristics of a member that the business wants to quantify. For example, an Employees dimension has attributes for name, title, department, employee status, start date, and salary. A Products dimension has attributes for name, introduction date, size, and color.

Snapshot

A snapshot, also referred to as a version, captures the state of a dimension at a particular point in time. There are different approaches when creating a new version of a dimension. You could create a new version after a major event, such as completing a planning process that contributes data. You could also create a new version at a specified time, such as the first day of every month.

A version can be public or private. A private version cannot be published. When you are ready to publish a version, you create a public version.

Person

To use Business Viewpoint Studio, users must log in using an account from an IBM Cognos namespace that is associated with a Business Viewpoint Studio user identity. The minimum required credentials are a user name and password.

Users are created for authentication and authorization purposes.
Publication

To allow dimensions in IBM® Cognos® Business Viewpoint Studio to be used in other applications, you must create a publication and publish it.

You can set up a publication that will always publish the latest version. You can specify how many of the latest versions are to be published. For example, each version represents one month of data. You specify that the latest three versions are to be published. You do not need to update the definition of the publication to include the last three months. You just have to publish the publication.

To download publications that you published, you must disable pop-ups in your Web browser.

You can create these types of publications:

- an IBM Cognos package for reporting applications to use
  A data source must be set up in IBM Cognos before you can publish the dimension as a package. For more information, see the Business Viewpoint Server Installation and Configuration Guide.

- an IBM Cognos Framework Manager model
  You can then extend the dimension by using features available in Framework Manager.

- a csv file
  You can edit the csv file and re-import it or use the csv file as a back-up of your data.

You then publish these publications to a network location or to an IBM Cognos system.

You can also notify users that a publication has been created for a dimension.

Workflow

You use workflows to manage and control who works with data.

You assign a list, hierarchy, or set to nominators. The nominators are business users who are experts in a business domain. They will add, define, and modify objects in their assigned areas. They ensure that the objects are appropriate for their business usage. When they are done, they submit the nomination task back to you.

After reconciling the nominated data, you assign it to reviewers, who approve it before you publish it. Reviewers are business users with the authority to approve data structures.
Chapter 6: Web service methods reference

You can use the methods described in this section to start and stop web service user sessions, verify web service availability, and execute XML API requests.

API methods

Web service API methods are formally defined in the following WSDL file:
<installation_location>/sdk/wsd1/BusinessViewpoint.wsdl

Many web service toolkits can process a WSDL file and automatically generate a client-side proxy code to call the Business Viewpoint web service API.

The web service is available at the following URL:
http://host:port/bv/services/BusinessViewpointService

where host is the Business Viewpoint server host name and port is the Business Viewpoint server port. By default, the port number is "9410".

execute(xmlApiRequest)

You must establish a user session before you can use this method. This method initiates a synchronous conversation with the service.

This method sends an XML API request to the web service. The web service executes the request. If the execution completes successfully, the web service provides an XML response. If the execution fails, the web service throws an exception. See "Handling exceptions" (p. 63) for details.

Provide a URL-encoded XML specification for the request parameter. The request schema for this specification is described in "XML API request reference" (p. 65).

A URL-encoded XML specification is returned when the synchronous conversation status has completed. The response schema for this specification is described in "XML API response reference" (p. 115).

Method signatures

- Java and Apache Axis
  ```java
  public java.lang.String execute( java.lang.String xmlAPIRequest )
  ```

- C# .NET
  ```csharp
  public string execute( string xmlAPIRequest )
  ```

Input parameter

xmlApiRequest

The parameter is a string containing a URL-encoded XML specification. The XML specification must conform to the Business Viewpoint XML API Request schema.
Output parameter

xmlApiResponse

The result is a string containing a URL-encoded XML specification. The XML specification conforms to the Business Viewpoint XML API Response schema. Note that some SOAP toolkits automatically encode and decode the XML. One such example is the Java™ Axis toolkit. If the toolkit cannot perform the encoding and decoding, the application developer must handle the URL-encoding and decoding.

Example: Using the execute(xmlApiRequest) method with the Business Viewpoint service in Java

The following Java code snippet demonstrates how to use the execute(xmlApiRequest) method with the Business Viewpoint service.

```java
// Load the spec from an XML file or generate it "on the fly" in the Java code
String xmlRequest = "...";

String xmlResponse = null;
try {
    xmlResponse = bvService.execute( xmlRequest );
} catch ( RemoteException e ) {
    // handle the exception, for example
    throw new RuntimeException( "Execute operation failed", e );
}

// success: parse the XML response
```

logon(credential)

Use this method to log on through the Business Viewpoint software development kit. If you are authenticated by the Business Viewpoint application, the action is considered successful. This method does not return any results. If an authentication failure occurs, the logon method throws an exception. See "Handling exceptions" (p. 63) for details.

Method signatures

- **Java and Apache Axis**
  
  `public java.lang.String logon(com.cognos.developer.schemas.bibus._3.XmlEncodedXML credentials)`

- **C# .NET**
  
  `public string logon( xmlEncodedXML credentials )`

Input parameter

Use the following parameter when calling this method.

credential

The parameter is a string containing a URL-encoded XML specification. The XML specification must conform to the `<installation_location>/sdk/xmlapi/BV_UserCredential.xsd` schema included in the software development kit.
Example:

```xml
<credential>
  <username>text</username>
  <password>text</password>
</credential>
```

**Output parameter**

None.

**Example: Using the logon(credentials) method with the Business Viewpoint service in Java**

The following Java code snippet demonstrates how to use the logon(credentials) method with the Business Viewpoint service.

```java
String username = ...;
String password = ...;
StringBuilder credential = new StringBuilder();

credential.append( "<credential>" );
credential.append( " <username>" + username + "</username>" );
credential.append( " <password>" + password + "</password>" );
credential.append( "</credential>" );

try
{   
    bvService.logon( credential.toString() );
}
catch ( RemoteException e )
{
   // handle the exception, for example
   throw new RuntimeException( "Logon operation failed", e );
}

// success
...
```

**logoff()**

Use this method to log off through the software development kit. If the action is successful, the method does not return any results. If a failure occurs, the method throws an exception. See "Handling exceptions" (p. 63) for details.

**Method signatures**

- **Java and Apache Axis**
  ```java
  public void logoff()
  ```

- **C# .NET**
  ```csharp
  public void logoff()
  ```

**Input parameter**

None.
Output parameter
None.

Example: Using the logoff() method with the Business Viewpoint service in Java

The following Java code snippet demonstrates how to use the logoff() method with the Business Viewpoint service.

```java
try {
    // do not call logoff() if logon() failed
    bvService.logoff();
} catch (RemoteException e) {
    // handle the exception, for example
    throw new RuntimeException("Logoff failed", e);
}
```

ping()

Use this method to verify whether the web service is running. This is a quick way to send a SOAP request to the service without logging on. If a failure occurs, the method throws an exception. See "Handling exceptions" (p. 63) for details.

Method signatures

- **Java and Apache Axis**
  ```java
  public String ping()
  ```
- **C# .NET**
  ```csharp
  public string ping()
  ```

Input parameter
None.

Output parameter

**pingResponse**

If the action is successful, the web service returns the following string:

"Business Viewpoint Web Service is active."

Example: Using the ping() method with the Business Viewpoint service in Java

The following Java code snippet demonstrates how to use the ping() method with the Business Viewpoint service.

```java
String EXPECTED_PING_RESPONSE = "Business Viewpoint Web Service is active.");
String pingResponse = null;
try {
    pingResponse = bvService.ping();
}
```
trustedLogon()

This method is reserved for internal use only.

Handling exceptions

This section provides information about handling Business Viewpoint web service exceptions. Business Viewpoint web service is implemented in the Java language using the Axis web service toolkit. The web service reports errors as Java org.apache.axis.AxisFault exceptions. The Axis toolkit maps the exceptions to SOAP faults, which contain the following:

1. A fault string
2. A fault code
3. A fault actor
4. Fault details

For more details on SOAP faults, refer to the Simple Object Access Protocol (SOAP) 1.1 specification. The client-side SOAP toolkit is responsible for mapping SOAP faults to the native error reporting mechanism in your programming language of choice, such as Java exceptions, C# exceptions, or Visual Basic 6.0. Consult your SOAP toolkit documentation for details on how to handle SOAP faults.

Java exceptions (Axis toolkit)

If you choose Axis as the client-side SOAP toolkit for your software development kit application, Business Viewpoint errors are reported as Java java.rmi.RemoteException exceptions. The actual type of exception thrown by Axis is org.apache.axis.AxisFault, a subtype of java.rmi.RemoteException.

Use the getMessage() method from the RemoteException class to get the exception message as a string.

Use the getFaultDetails() method from the AxisFault class to extract additional details about the Business Viewpoint error. The getFaultDetails() method returns an array of XML org.w3c.dom.Element objects containing the exception details.

Catching AxisFault (Java)

The following code demonstrates the use of a try block to catch the org.apache.axis.AxisFault exception:
try
{
    ... 
    // Code using one or more Business Viewpoint Web Service methods.
    ... 
}
catch ( AxisFault fault )
{
    String message = fault.getMessage();
    Element[] details = fault.getFaultDetails();

    // Exception-handling code
    ...
}
catch ( RemoteException e )
{
    // not expected to happen
    // the exceptions thrown by Axis toolkit are expected to be of type AxisFault
    ...
}
Chapter 7: XML API request reference

The XML API request schema defines the actions supported by IBM® Cognos® Business Viewpoint. Executing an action includes the following tasks:

- Create or generate an XML specification that contains one action. The action definition must conform to the XML API Request schema.
- Submit the action to the Business Viewpoint Web Service using the `execute(xmlApiRequest)` method or to the IBM Cognos BI dimension management service using the `runSpecification(specification, parameterValues, options)` method. The supported actions are shown in the following diagram.

![Diagram showing supported actions](image_url)

The request specification schema is defined in `BV_XmlApiRequest.xsd`, which is located in the `<installation_location>\sdk\xmlapi\schemas\` directory.

This schema references two other schemas:

- **BV_ApplicationActions.xsd**
  defines the IBM Cognos Business Viewpoint application actions. You can use the application actions to automate Business Viewpoint Studio tasks. For example: import data, export data, and create a new dimension version.

- **BV_RepositoryActions.xsd**
  defines the IBM Cognos Business Viewpoint repository actions. You can use the repository actions to manage the objects stored in the Business Viewpoint data repository. For example:
find the objects that match the specified criteria, create a new object, and delete an existing object.

For each element within the schema, this section provides

- the name and description of the element
- information about attributes that apply to the element, including each attribute’s name, description, optionality, legal values, and default value, if applicable
- content model information, consisting of a list of valid child elements presented as an element model group
- a list of valid parent elements

**Element model group**

The list of children for each element is presented using the following standard notation.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plus sign (+)</td>
<td>The preceding element can be repeated more than once but must occur at least once.</td>
</tr>
<tr>
<td>Question mark (?)</td>
<td>The preceding element is optional. It can be absent or it can occur exactly once.</td>
</tr>
<tr>
<td>Asterisk (*)</td>
<td>An asterisk (*) after an element specifies that the element is optional. It can occur zero or more times.</td>
</tr>
<tr>
<td>None</td>
<td>If an element has no plus sign (+), question mark (?), or asterisk (*) following it, the element must occur only once.</td>
</tr>
<tr>
<td>#PCDATA</td>
<td>The symbol #PCDATA specifies character data that is parsed.</td>
</tr>
<tr>
<td>Parentheses</td>
<td>Parentheses group elements. Element groups are controlled using the same symbols as elements.</td>
</tr>
<tr>
<td>Bar (</td>
<td>)</td>
</tr>
<tr>
<td>Comma (,)</td>
<td>The elements that it separates must be present in the specified order.</td>
</tr>
</tbody>
</table>

**action**

The root element of the XML API request schema.
Content Model

(createVersion | exportData | importData | runPublication | createUser | deleteUser | setObjectPermissions | runLink | create | delete | find | get | modify)

Parent Elements

None.
This schema defines the IBM® Cognos® Business Viewpoint application actions available to a software development kit user. You can use the application actions to automate Business Viewpoint Studio tasks. The following actions are available:

- **createVersion** - create a new version of a dimension.
- **exportData** - export Business Viewpoint data to an external target.
- **importData** - import data from an external source into Business Viewpoint.
- **runPublication** - run an existing publication in Business Viewpoint Studio.
- **createUser** - create a new Business Viewpoint user account.
- **deleteUser** - delete an existing Business Viewpoint user account.
- **setObjectPermissions** - specify user permissions for a given Business Viewpoint object.
- **runLink** - run an existing link in Business Viewpoint Studio.

The application actions schema is defined in BV_ApplicationActions.xsd, which is located in the `<installation_location>/sdk/xmlapi/schemas/` directory.

BV_ApplicationActions.xsd is a supplementary schema to the main XML API Request schema, BV_XmlApiRequest.xsd. Use the main schema to validate the application actions XML.

For each element, this section provides the following:

- the name and description of the element
- information about attributes that apply to the element, including each attribute's name, description, optionality, legal values, and default value, if applicable
- content model information, consisting of a list of valid child elements presented as an element model group
- a list of valid parent elements

### Element model group
The list of children for each element is presented using the following standard notation.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plus sign (+)</td>
<td>The preceding element can be repeated more than once but must occur at least once.</td>
</tr>
<tr>
<td>Question mark (?)</td>
<td>The preceding element is optional. It can be absent or it can occur exactly once.</td>
</tr>
</tbody>
</table>
Meaning

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asterisk (*)</td>
<td>An asterisk (*) after an element specifies that the element is optional. It can occur zero or more times.</td>
</tr>
<tr>
<td>None</td>
<td>If an element has no plus sign (+), question mark (?), or asterisk (*) following it, the element must occur only once.</td>
</tr>
<tr>
<td>#PCDATA</td>
<td>The symbol #PCDATA specifies character data that is parsed.</td>
</tr>
<tr>
<td>Parentheses</td>
<td>Parentheses group elements. Element groups are controlled using the same symbols as elements.</td>
</tr>
<tr>
<td>Bar (</td>
<td>)</td>
</tr>
<tr>
<td>Comma (,)</td>
<td>The elements that it separates must be present in the specified order.</td>
</tr>
</tbody>
</table>

**applicationRoles**

Assigns one or more application roles to a user. Valid roles include Administrator, Modeler, Reviewer, Nominator, and Consumer.

**Content Model**

```
(role)+
```

**Parent Elements**

createUser

**What's New**

**Version 8.4.1**

The applicationRoles element has been added.

**class**

Specifies the name of a class. See "Business Viewpoint class reference" (p. 55) to find legal class names.

**Content Model**

Empty element.
Parent Elements

setObjectPermissions

What's New

Version 8.4.1
The class element has been added.

cognos8Package
Specifies the name of the IBM Cognos package to be created by the exportData action.

Content Model

(packageName, systemName)

Parent Elements

exportTarget

cognos8Package
Specifies the name of the IBM Cognos package to be used by the importData action.

Content Model

(systemName, searchPath)

Parent Elements

importSource

create
Specifies the permission to create a new instance of a class, a property, or an attribute. A value of "true" grants permission.

Default: true

Content Model

Content type is boolean.

Parent Elements

permissions
What's New

Version 8.4.1
The create element has been added.

createUser

Use this action to create a new Business Viewpoint user account.

Content Model

(name, email, securityIdentity, applicationRoles)

What's New

Version 8.4.1
The createUser element has been added.

createVersion

Use this action to create a new version of a dimension.

Content Model

(dimension, versionName, versionDescription?, versionType)

csvFile

Specifies the CSV file used as an external target by the exportData action.

Attributes

delimiter
Specifies the character used to separate the values in the csvFile. The default value is a comma (,).
Usage: optional.

qualifier
Specifies the character used to indicate that everything within a qualifier set should be interpreted exactly as it appears in the csvFile. The default value is double-quotes (").
Usage: optional.

Content Model

(filePath)
**Parent Elements**

`exportTarget`

**csvFile**

Specifies the CSV file used as an external data source by the `importData` action.

**Attributes**

- **delimiter**
  Specifies the character used to separate the values in the `csvFile`. The default value is a comma (,).
  Usage: optional.

- **encoding**
  Specifies the character set of the `csvFile`. If the encoding attribute is not specified, the IBM Cognos Business Viewpoint Server uses a default that depends on the locale and charset of the underlying operating system.
  The value of the encoding attribute must be a legal character set name, as described in [http://java.sun.com/j2se/1.5.0/docs/api/java/nio/charset/Charset.html](http://java.sun.com/j2se/1.5.0/docs/api/java/nio/charset/Charset.html)
  Usage: optional.

- **qualifier**
  Specifies the character used to indicate that everything within a qualifier set should be interpreted exactly as it appears in the `csvFile`. The default value is double-quotes (").
  Usage: optional.

**Content Model**

`(filePath)`

**Parent Elements**

`importSource`

**delete**

Specifies the permission to delete the object. A value of "true" grants permission.
Default: true

**Content Model**

Content type is boolean.
What's New

**Version 8.4.1**
The delete element has been added.

**deleteUser**
Use this action to delete an existing Business Viewpoint user account.

**Content Model**

(securityIdentity)

What's New

**Version 8.4.1**
The deleteUser element has been added.

**description**
Specifies the description of the new surrogateKeyGroup.

**Content Model**

Empty element.

**Parent Elements**

surrogateKeyGroup

dimension
Specifies the name of the new dimension to be created.

**Content Model**

Empty element.

**Parent Elements**

createVersion
What's New

Version **8.4.1**
The dimension element has been added.

**email**

Specifies the email address of the new Business Viewpoint user account.

**Content Model**
Empty element.

**Parent Elements**
createUser

What's New

Version **8.4.1**
The email element has been added.

**excel**

Specifies the Microsoft® Excel file used as an external data source by the importData action.

**Content Model**
(filePath)

**Parent Elements**
importSource

**exportData**

Use this action to export data from Business Viewpoint to an external target. The exportSpecification element is used to specify the dimension to be exported and security rules.

**Content Model**
(exportTarget, exportSpecification)

**exportTarget**

Specifies the type of output data to export. The available options include an IBM Cognos package, a Framework Manager model, or a CSV file.
Content Model

(cognos8Package | fmModel | csvFile)

Parent Elements

exportData

external

Specifies whether to set internal or external permissions. Internal permissions apply to Business Viewpoint users. External permissions apply to IBM Cognos users working with the published Business Viewpoint data.

Default: false

Content Model

Content type is boolean.

Parent Elements

setObjectPermissions

What's New

Version 8.4.1

The external element has been added.

filePath

Specifies the complete file path of an external data source (import action) or an external data target (export action). An example of a file path for a CSV file would be the following:

<csvFile>
  <filePath>c:\sample.csv</filePath>
</csvFile>

Content Model

Empty element.

Parent Elements

csvFile, csvFile, excel, fmModel

fmModel

Specifies the IBM Cognos Framework Manager (FM) model to be created by the exportData action.
Content Model
(packageName, filePath)

Parent Elements
exportTarget

groupId
Specifies an existing surrogateKeyGroup ID to perform incremental data import from the same import source.

Content Model
Empty element.

Parent Elements
surrogateKeyGroup

importData
Use this action to import data from an external data source into IBM Cognos Business Viewpoint. The importSpecification element is used to specify the root import.

Content Model
(importSource, importSpecification, surrogateKeyGroup?, previewOnly?)

importSource
Specifies the type of external data source from which the data is imported. The available options include an IBM Cognos package, a Microsoft Excel file, or a CSV file.

Content Model
(cognos8Package | excel | csvFile)

Parent Elements
importData

linkName
Specifies the name of the link created in Business Viewpoint Studio.
**Content Model**

Content type is string.

**Parent Elements**

`runLink`

**What's New**

**Version 10.1.0**

The `linkName` element has been added.

**name**

Specifies the new `surrogateKeyGroup` name of the data to import.

**Content Model**

Empty element.

**Parent Elements**

`createUser`, `surrogateKeyGroup`

**What's New**

**Version 8.4.1**

The `name` element has been added.

**namespace**

Specifies the namespace in the external authentication system when invoking a `runLink` action against a source data store.

**Content Model**

Content type is string.

**Parent Elements**

`sourceDataStoreAuthentication`

**namespace**

Specifies the namespace in the external authentication system when invoking a `runLink` action against a target data store.
Content Model

Content type is string.

Parent Elements

targetDataStoreAuthentication

namespace

Specifies the namespace in the external authentication system when managing Business Viewpoint user accounts.

Content Model

Empty element.

Parent Elements

securityIdentity

objectId

Specifies the unique ID number of the object.

Content Model

Empty element.

Parent Elements

setObjectPermissions

What's New

Version 8.4.1
The objectId element has been added.

packageName

Specifies the name of the new IBM Cognos package to be created by the exportData action.

The combination of systemName and packageName identifies the package to be exported.

Content Model

Empty element.
Parent Elements

cognos8Package, fmModel

password

Specifies the user password in the external authentication system when invoking a runLink action against a source data store.

Content Model

Content type is string.

Parent Elements

sourceDataStoreAuthentication

password

Specifies the user password in the external authentication system when invoking a runLink action against a target data store.

Content Model

Content type is string.

Parent Elements

targetDataStoreAuthentication

permissions

Specifies the set of permissions that apply to the user requesting access to the given object. If this element is not specified, this action clears any existing permissions.

Content Model

(create, read, update, delete, propagate)

Parent Elements

setObjectPermissions

What's New

Version 8.4.1

The permissions element has been added.
**previewOnly**
Set this boolean element to true if you want to preview the results of the import action. Set the value to false if you want to execute the action.
Default: false

**Content Model**
Content type is boolean.

**Parent Elements**
importData

**propagate**
If this element is set to "true", the specified permissions are propagated to all child objects.
Default: true

**Content Model**
Content type is boolean.

**Parent Elements**
permissions

**What's New**

**Version 8.4.1**
The propagate element has been added.

**publication**
Specifies the name of a publication to run in Business Viewpoint Studio.

**Content Model**
Empty element.

**Parent Elements**
runPublication
Chapter 8: Application actions reference

**What's New**

**Version 8.4.1**
The publication element has been added.

**read**

Specifies the permission to view the object. A value of "true" grants permission.
Default: true

**Content Model**

Content type is boolean.

**Parent Elements**

permissions

---

**What's New**

**Version 8.4.1**
The read element has been added.

**role**

Specifies the roles to assign to a user. You can assign more than one role to each user. When users are assigned more than one role, the least restrictive permissions are applied. For example, if a user is assigned the roles of Modeler and Nominator, the permissions for the Modeler role are applied when the user logs on to Business Viewpoint Studio.

**Content Model**

Content type is string.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator</td>
<td>Can create, read, write to, and set permissions for any master dimension. Can also create users and assign roles.</td>
</tr>
<tr>
<td>Modeler</td>
<td>Can create master dimensions and read any master dimension. Can create users and assign roles. Can also write to and set permissions for objects for which they are the owner.</td>
</tr>
<tr>
<td>Reviewer</td>
<td>Can create, read, write to, approve, and set permissions for the master dimension that they have been assigned.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Nominator</td>
<td>Can create, read, write to, and set permissions for the master dimension that they have been assigned.</td>
</tr>
<tr>
<td>Consumer</td>
<td>Can read the master dimension.</td>
</tr>
</tbody>
</table>

**Parent Elements**

applicationRoles

**What's New**

**Version 8.4.1**

The role element has been added.

**runLink**

Use this action to run an existing link, created in Business Viewpoint Studio, to synchronize fact data in source and target data sources.

**Content Model**

(linkName, sourceDataStoreAuthentication, targetDataStoreAuthentication)

**What's New**

**Version 10.1.0**

The runLink element has been added.

**runPublication**

Use this action to run an existing publication created in Business Viewpoint Studio.

**Content Model**

(publication)

**What's New**

**Version 8.4.1**

The runPublication element has been added.
searchPath

Specifies the IBM Cognos package search path, in IBM Cognos Content Manager, used by the importData action.

The combination of systemName and searchPath identifies the package to be imported.

Content Model

Empty element.

Parent Elements

cognos8Package

securityIdentity

Specifies the security identity of the user in the external authentication system.

Content Model

(systemName, namespace, userId)

Parent Elements

createUser, deleteUser, setObjectPermissions

What's New

Version 8.4.1

The securityIdentity element has been added.

setObjectPermissions

Use this action to specify the set of access permissions and conditions for a given object.

Content Model

(class, objectID, securityIdentity, permissions?, external?)

What's New

Version 8.4.1

The setObjectPermissions element has been added.
**sourceDataStoreAuthentication**

Specifies a user’s authentication parameters for a Business Viewpoint link to an external source data store.

**Content Model**

(userId, password, namespace)

**Parent Elements**

runLink

**What's New**

*Version 10.1.0*

The sourceDataStoreAuthentication element has been added.

**surrogateKeyGroup**

Specifies the surrogate key group definition. This element is optional.

If the element is not specified, IBM Cognos Business Viewpoint will create a new group with a default name and description. Specify the group ID of an existing group to perform incremental data import from the same import source.

**Content Model**

(groupId | (name, description?))

**Parent Elements**

importData

**systemName**

Specifies the IBM Cognos system name, as defined in Business Viewpoint Studio. IBM Cognos Business Viewpoint can be configured to work with multiple IBM Cognos BI systems.

**Content Model**

Empty element.

**Parent Elements**

cognos8Package, cognos8Package, securityIdentity
What's New

Version 8.4.1
The systemName element has been added.

targetDataStoreAuthentication
Specifies a user’s authentication parameters for a Business Viewpoint link to an external target data store.

Content Model
(userId, password, namespace)

Parent Elements
runLink

What's New

Version 10.1.0
The targetDataStoreAuthentication element has been added.

update
Specifies the permission to update the object. A value of "true" grants permission.
Default: true

Content Model
Content type is boolean.

Parent Elements
permissions

What's New

Version 8.4.1
The update element has been added.

userId
Specifies the user ID in the external authentication system when invoking a runLink action against a source data store.
**Content Model**
Content type is string.

**Parent Elements**
sourceDataStoreAuthentication

**userId**
Specifies the user ID in the external authentication system when invoking a `runLink` action against a target data store.

**Content Model**
Content type is string.

**Parent Elements**

targetDataStoreAuthentication

**userId**
Specifies the user ID in the external authentication system when managing Business Viewpoint user accounts.

**Content Model**
Empty element.

**Parent Elements**
securityIdentity

**versionDescription**
Optionally provide a description for the new version being created.

**Content Model**
Empty element.

**Parent Elements**
createVersion
Chapter 8: Application actions reference

What's New

**Version 8.4.1**
The versionDescription element has been added.

**versionName**
Specifies the name of the new version to be created.

Content Model
Empty element.

Parent Elements
createVersion

**versionType**
Specifies the type of version to be created (public or private).

Content Model
Content type is string.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public</td>
<td>A public dimension version can be published.</td>
</tr>
<tr>
<td>private</td>
<td>A private dimension version cannot be published.</td>
</tr>
</tbody>
</table>

Parent Elements
createVersion
Chapter 9: Repository actions reference

This schema defines the IBM Cognos® Business Viewpoint repository actions. You can use the repository actions to manage the objects stored in the Business Viewpoint data repository. The following actions are available:

- **create** - create a new repository object
- **delete** - delete an existing repository object
- **find** - find repository objects that match a specified criteria
- **get** - get a specified repository object
- **modify** - modify the properties of an existing repository object

The repository actions schema is defined in BV_RepositoryActions.xsd, which is located in the \<installation_location>\sdk\xmlapi\schemas\ directory.

BV_RepositoryActions.xsd is a supplementary schema to the main XML API request schema, BV_XmlApiRequest.xsd. Use the main schema to validate the repository actions XML.

For each element, this section provides the following:

- the name and description of the element
- information about attributes that apply to the element, including each attribute’s name, description, optionality, legal values, and default value, if applicable
- content model information, consisting of a list of valid child elements presented as an element model group
- a list of valid parent elements

**Element model group**

The list of children for each element is presented using the following standard notation.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plus sign (+)</td>
<td>The preceding element can be repeated more than once but must occur at least once.</td>
</tr>
<tr>
<td>Question mark (?)</td>
<td>The preceding element is optional. It can be absent or it can occur exactly once.</td>
</tr>
<tr>
<td>Asterisk (*)</td>
<td>An asterisk (*) after an element specifies that the element is optional. It can occur zero or more times.</td>
</tr>
</tbody>
</table>
### Symbol | Meaning
--- | ---
None | If an element has no plus sign (+), question mark (?), or asterisk (*) following it, the element must occur only once.

#PCDATA | The symbol #PCDATA specifies character data that is parsed.

Parentheses | Parentheses group elements. Element groups are controlled using the same symbols as elements.

Bar (|) | A bar (|) between elements specifies that one of the listed elements must be present.

Comma (,) | The elements that it separates must be present in the specified order.

---

**class**

Specifies the name of the IBM Cognos Business Viewpoint class to create, delete, modify, find, or get. See "Business Viewpoint class reference" (p. 55) to find legal class names.

**Content Model**

Content type is string.

**Parent Elements**

create, delete, find, get, modify

**constraint**

Defines the IBM Cognos Business Viewpoint object properties and their values.

In the context of the create or modify actions, this element specifies the property name and the value to be set by the action.

In the context of the find action, this element is used to filter the results. Multiple constraints are combined using an AND boolean operation.

**Content Model**

(property, value)

**Parent Elements**

customs
**constraints**

Contains one or more constraint elements.

**Content Model**

(constraint+)

**Parent Elements**

create, find, modify

**create**

Creates a new Business Viewpoint repository object. The object is determined by the class element.

**Content Model**

(class, constraints?)

**delete**

Deletes an existing Business Viewpoint repository object.

**Content Model**

(class, objectID)

**find**

Finds one or more Business Viewpoint repository objects based on a given search criteria.

**Content Model**

(class, requestedProperties, constraints?, hints?)

**get**

Retrieve a specified Business Viewpoint repository object.

**Content Model**

(objectID, class, requestedProperties)

**hint**

Specifies the name and value of a hint. When present, this element modifies the standard behavior of the actions.
**Content Model**

(name, value?)

**Parent Elements**

hints

**hints**

Contains one or more hint elements.

**Content Model**

(hint+)

**Parent Elements**

find, modify

**modify**

Changes the properties of an existing Business Viewpoint repository object.

**Content Model**

(objectID, class, constraints, hints?)

**name**

Specifies the hint type, which alters the default behavior of the find action.

**Content Model**

Content type is string.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>START_QUERY_RANGE_POSITION</td>
<td>This hint applies to the find action. Use this hint to specify the starting position for the set of objects returned by the find action. For example, the find action might match 100 member objects. Specify START_QUERY_RANGE_POSITION=50 to return the results starting from object number 50. Use this hint in combination with the QUERY_RANGE_REQUESTED_COUNT hint to page through a large result set.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>QUERY_RANGE_REQUESTED_COUNT</td>
<td>This hint applies to the find action. Use this hint to specify the number of objects to be returned by the find action. If not specified, the find action returns 50 objects or less. Use this hint in combination with the START_QUERY_RANGE_POSITION hint to page through a large result set.</td>
</tr>
<tr>
<td>GET_SUMMARY</td>
<td>This hint applies to the find action. Use this hint to request the total count of objects that match the search criteria. By default, the find action does not return the total count.</td>
</tr>
</tbody>
</table>
| REFERENCE_DELETED_OBJECTS                 | This hint applies to the find action. Use this hint to specify whether deleted objects are included in the list of objects returned during object history retrieval. The following values are supported:  
  ● 0: Excludes deleted objects  
  ● 1: Includes deleted objects (default setting)  
  ● 2: Exclusively includes deleted objects |
| SNAPSHOT_REFERENCE                        | This hint applies to the find action. Use this hint to return the objects that belong to the specified dimension version. The value of the hint is the object ID of a Snapshot class instance. |
| SORT_PROPERTY                             | This hint applies to the find action. Use this hint to sort the returned objects by a property. The value of the hint is the Business Viewpoint property ID. |
| SORT_ORDER                                | This hint applies to the find action. Use this hint in combination with the SORT_PROPERTY hint to specify the sort order. The following values are supported:  
  ● 1: Sort ascending  
  ● 2: Sort descending |
| PROPERTY_VALIDATION_STATE                 | This hint applies to the find action. Use this hint to constrain the results of the find action. The following values are supported:  
  ● Ok: Returns the objects that pass validation  
  ● Invalid: Returns the objects that fail validation |

**Parent Elements**

hint
Chapter 9: Repository actions reference

**objectID**

Specifies the unique IBM Cognos Business Viewpoint object ID of the object to **modify**, **delete**, or **get**.

**Content Model**

Content type is string.

**Parent Elements**

delete, get, modify

**property**

Specifies the name of the object property to be returned by an XML API action.

**Content Model**

Content type is string.

**Parent Elements**

constraint, requestedProperties

**requestedProperties**

Defines the object properties to be returned by the action.

This element contains one or more **property** elements.

**Content Model**

(property+)

**Parent Elements**

find, get

**value**

Specifies the value of a constraint or hint.

**Content Model**

Content type is string.

**Parent Elements**

constraint, hint
Chapter 10: Import specification

The import specification schema defines the structure of the "importSpecification" (p. 100) element, the child element of the "importData" (p. 77) application action. Use the import specification to specify the details of the import action. Some parameters that you might want to specify include the following:

- the data in the external data source you want to import
- the areas of the master dimension where you want to import the data
- how to map the external data to the master dimension structures

The import specification schema is defined in BV_ImportSpecification.xsd, which is located in the <installation_location>\sdk\xmlapi\schemas\ directory.

BV_ImportSpecification.xsd is a supplementary schema. BV_ApplicationActions.xsd imports BV_ImportSpecification.xsd to define the structure of the "importSpecification" (p. 100) element. BV_XmlApiRequest.xsd is the main schema of the XML API request. Use the main schema to validate the "importData" (p. 77) action.

For each element, this section provides the following:

- the name and description of the element
- information about attributes that apply to the element, including each attribute's name, description, optionality, legal values, and default value, if applicable
- content model information, consisting of a list of valid child elements presented as an element model group
- a list of valid parent elements

Element model group
The list of children for each element is presented using the following standard notation.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plus sign (+)</td>
<td>The preceding element can be repeated more than once but must occur at least once.</td>
</tr>
<tr>
<td>Question mark (?)</td>
<td>The preceding element is optional. It can be absent or it can occur exactly once.</td>
</tr>
<tr>
<td>Asterisk (*)</td>
<td>An asterisk (*) after an element specifies that the element is optional. It can occur zero or more times.</td>
</tr>
<tr>
<td>Symbol</td>
<td>Meaning</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>None</td>
<td>If an element has no plus sign (+), question mark (?), or asterisk (*) following it, the element must occur only once.</td>
</tr>
<tr>
<td>#PCDATA</td>
<td>The symbol #PCDATA specifies character data that is parsed.</td>
</tr>
<tr>
<td>Parentheses</td>
<td>Parentheses group elements. Element groups are controlled using the same symbols as elements.</td>
</tr>
<tr>
<td>Bar (l)</td>
<td>A bar (l) between elements specifies that one of the listed elements must be present.</td>
</tr>
<tr>
<td>Comma (,)</td>
<td>The elements that it separates must be present in the specified order.</td>
</tr>
</tbody>
</table>

**annotation**

Defines a role-specific extension that can be used to annotate the role.

**Attributes**

**bvPropertyExtensionId**

Specifies a reference to an existing property extension in Business Viewpoint.

Usage: optional. Type: long.

**What's New  Version 8.4.1**

The bvPropertyExtensionId attribute has been added.

**name**

Specifies the name of the role.

Usage: optional. Type: string.

**Content Model**

Empty element.

**Parent Elements**

role

**What's New**

**Version 8.4.1**

The annotation element has been added.
crossReference

Defines the cross-references for a list to other lists.

Attributes

cardinality

Specifies whether this property can be assigned a single or multiple values.

The cardinality only applies from the source list to the target list. The relationship between target list and the source list is always "multiple".

Example 1: `<list id='1'> <crossReference listId='2' cardinality='single'/> </list>` Resulting relationships: list1 <--1------N--> list2

Example 2: `<list id='1'> <crossReference listId='2' cardinality='multiple'/> </list>` Resulting relationships: list1 <--N------N--> list2

Example 3: `<list id='2'> <crossReference listId='1' cardinality='single'/> </list>` Resulting relationships: list2 <--1------N--> list1

Example 4: `<list id='2'> <crossReference listId='1' cardinality='multiple'/> </list>` Resulting relationships: list2 <--N------N--> list1

Usage: optional.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>single</td>
<td>The property can be assigned a single value.</td>
</tr>
<tr>
<td>multiple</td>
<td>The property can be assigned multiple values.</td>
</tr>
</tbody>
</table>

listId

Specifies a reference to the ID of the list within the Import specification.

Usage: required. Type: short.

Content Model

Empty element.

Parent Elements

list
dimension

Defines a dimension instance in Business Viewpoint.
When importing into an existing dimension, the `bvObjectId` attribute must be specified; otherwise, a new dimension will be created. When creating a new dimension, at least one `role` that defines the dimension name is required.

**Attributes**

- **bvObjectId**
  - Specifies a reference to an existing instance in Business Viewpoint.
  - Usage: optional. Type: long.

- **crossReferenceAllLists**
  - Specifies whether all lists in the dimension should be cross-referenced.
  - Default: true
  - Usage: optional. Type: boolean.

- **shareMembers**
  - Specifies that members should be shared across a dimension. This allows IBM Cognos Transformer to import converging levels.
  - Default: false
  - Usage: optional. Type: boolean.

**What’s New Version 8.4.1**
The `shareMembers` attribute has been added.

**Content Model**

```
(role*, ((list? | hierarchy? | dynamicHierarchy?)*) )
```

**Parent Elements**

- `importSpecification`

**dynamicHierarchy**

Defines a dynamic hierarchy instance in Business Viewpoint. For more information on dynamic hierarchies, see "Hierarchy" (p. 55).

**Attributes**

- **bvObjectId**
  - Specifies a reference to an existing instance in Business Viewpoint.
  - Usage: optional. Type: long.
Content Model

(role*, (level+ | members+), memberApex*)?

Parent Elements
dimension

hierarchy

Defines a static hierarchy instance in Business Viewpoint. For more information on static hierarchies, see "Hierarchy" (p. 55).

Attributes

bvObjectId
Specifies a reference to an existing instance in Business Viewpoint.
Usage: optional. Type: long.

bvParentMemberObjectId
Specifies a reference to an existing instance in Business Viewpoint. It is used when attaching members to an existing parent.
Usage: optional. Type: long.

nullMemberContainerName
When suppressNullMembers is true, null members will be placed into a container to maintain the levelization of the hierarchy. The name of this container is controlled by this attribute.
Usage: optional. Type: string.

suppressNullMembers
Specifies whether null members should be suppressed in a level-based hierarchy, maintaining the levelization of the hierarchy.
Default: true
Usage: optional. Type: boolean.
importSpecification
The root import specification element.

Content Model
(dimension+)

level
 Defines a level instance in Business Viewpoint.

Attributes

bv0bjectId
Specifies a reference to an existing instance in Business Viewpoint.
Usage: optional. Type: long.

Content Model
(role*, members)?

Parent Elements
dynamicHierarchy, hierarchy

list
 Defines a list instance in Business Viewpoint.

Attributes

bv0bjectId
Specifies a reference to an existing instance in Business Viewpoint.
Usage: optional. Type: long.

id
Specifies the ID of the list within the import specification.
Usage: optional. Type: short.

Content Model
(role*, members, crossReference*)?
**Parent Elements**

* dimension

**memberApex**

Defines an explicit root member in a dynamic hierarchy.

**Attributes**

* **bvObjectId**

  Specifies a reference to an existing instance in Business Viewpoint.

  Usage: required. Type: long.

**Content Model**

Empty element.

**Parent Elements**

* dynamicHierarchy

**members**

Defines the roles assigned to member instances that will be created from the tabular source.

**Attributes**

* **defaultUpdateRule**

  Specifies the default update rule for all contained roles. An update rule is used to reconcile the source with data that already exists in Business Viewpoint. Update rules can be overridden by roles.

  Default: applySource

  Usage: optional.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maintainTarget</td>
<td>Maintain the data that exists in Business Viewpoint.</td>
</tr>
<tr>
<td>applySource</td>
<td>Update Business Viewpoint with the changes from the source.</td>
</tr>
</tbody>
</table>

**Content Model**

( role+ )
Chapter 10: Import specification

**Parent Elements**

dynamicHierarchy, hierarchy, level, list

**role**

Defines the mapping of source items to repository objects and their attributes.

**Attributes**

**bvPropertyId**

Specifies a reference to an existing property in Business Viewpoint.

Usage: optional. Type: long.

**isHidden**

Specifies whether the new property must be hidden.

Default: false.

Usage: optional. Type: boolean.

**What's New  Version 8.4.1**

The isHidden attribute has been added.

**isIdentifier**

An identifier role (a role that has this attribute defined with a value of 'true') is used to uniquely identify members in the source data and to merge members into an existing dimension.

A set of roles can have at most one identifier role of type 'surrogateKeyRole' and one identifier role that is not of type 'surrogateKeyRole'.

Identifier roles of type 'surrogateKeyRole' are used to uniquely identify members in the source data and to reconcile members that already exist in Business Viewpoint when performing an update from the same source. Roles of type 'surrogateKeyRole' are not created as business attributes, they are created as key definitions and placed into the Business Viewpoint Key Management System.

Identifier roles that are not of type 'surrogateKeyRole' are used to merge the source data with existing members in the target dimension.

Usage: optional. Type: boolean.

**metadataId**

Specifies a reference to a metadata ID from the metadata for the data source being processed.

Usage: required. Type: string.

**name**

Specifies the name of the role.

Usage: optional. Type: string.
source
Specifies whether the source of the role value is text, a dataItem, or an expression.
Usage: required.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>The text of the role will be interpreted as a literal value.</td>
</tr>
<tr>
<td>dataItem</td>
<td>The text of the role will be interpreted as a reference to a metadata ID. Its value will be substituted while processing the source data.</td>
</tr>
<tr>
<td>expression</td>
<td>The text of the role will be interpreted as an expression. Quoted values will be interpreted as string literals, everything else will be interpreted as a reference to a metadata ID. Only the '+' operator is currently supported.</td>
</tr>
</tbody>
</table>

updateRule
Specifies the update rule for this role, which overrides the default update rule. An update rule is used to reconcile the source with data that already exists in Business Viewpoint.
Default: applySource
Usage: optional.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maintainTarget</td>
<td>Maintain the data that exists in Business Viewpoint.</td>
</tr>
<tr>
<td>applySource</td>
<td>Update Business Viewpoint with the changes from the source.</td>
</tr>
</tbody>
</table>

Content Model
(annotation*)

Parent Elements
dimension, dynamicHierarchy, hierarchy, level, list, members
Chapter 11: Using import specification

The import specification provides developers the ability to specify the details of an import action, which defines the external data source and mechanism for importing data into Business Viewpoint. This section provides developers with recommendations on using the import specification.

Logging and reusing import actions

Software development kit developers can simplify their task of authoring an import action by logging and reusing an import specification generated by the Business Viewpoint Server. By enabling the SDKImportActionLogger parameter in the log4.properties file, the Business Viewpoint Server will capture and store an import action each time a user runs the Import wizard in the Business Viewpoint Studio.

To automate the logging of import actions, perform the following steps:

1. Start the IBM® Cognos® Configuration tool.
2. Stop the Business Viewpoint Server.
   For more information on starting and stopping the Business Viewpoint Server, see the Business Viewpoint Server Installation and Configuration Guide.
3. Go to the following directory: \installation_location\webapps\bv\WEB-INF\classes
4. Open the log4.properties file in a text editor.
5. Find this line:
   log4j.logger.com.cognos.mdm.server.dataimport.sdkhelper.SDKImportActionLogger=OFF, SDKImportActionAppender
6. Change the OFF setting to INFO:
   log4j.logger.com.cognos.mdm.server.dataimport.sdkhelper.SDKImportActionLogger=INFO, SDKImportActionAppender
7. Save the log4.properties file.

By default, the log4.properties file is defined with the following settings:

- Business Viewpoint Server will log the import actions in the \installation_location\logs directory.
- Each import action is logged in a separate file using the following naming convention:
  sdkImportAction.xml.<number>
  Note: sdkImportAction.xml.1 represents the latest version.
Business Viewpoint Server will retain the 10 most recent files.

Depending on the type of data source being imported, a developer might be required to make changes before using the generated file:

- CSV or Microsoft Excel files: Update the `<filePath>` element with the location of the data source file.
- IBM Cognos BI package: No change is required.

**Typical usage scenario**

The following sequence of actions represents a typical use case:

1. Run the Import wizard in Business Viewpoint Studio to set up a dimension.
2. Run the Import wizard again to incrementally update the existing dimension.
3. Copy the `sdkImportAction.xml.1` file (logged in the previous step).
4. Edit the `<filePath>` element in the generated file.
5. Author a script that runs the incremental update on a scheduled basis.

The script can be based on the `BVExecuteCLI` Java™ sample provided in the following location:

```
<installation_location>/sdk/java/BVExecuteCLI
```

For example:

1. Compile the `BVExecuteCLI` Java sample.
2. Log the `sdkImportAction.xml.1` file, as described above.
3. Copy `sdkImportAction.xml.1` to the `BVExecuteCLI` directory.
4. Modify the `<filePath>` element in the XML file by specifying the location of the data file used for updating.
5. Run the import action as follows:

```
run.bat sdkImportAction.xml.1 sdkImportAction_response.xml
```
Chapter 12: Export specification

The export specification schema defines the structure of the "exportSpecification" (p. 108) element, the child element of the "exportData" (p. 75) application action. Use the export specification to specify the details of the exportData action. You can specify the following parameters:

- the master dimension version(s) you want to export
- the areas of the master dimension you want to export
- how to map Business Viewpoint data types to external data types
- how to map Business Viewpoint security rules to external security rules

The export specification schema is defined in BV_ExportSpecification.xsd, which is located in the <installation_location> \sdk\xmlapi\schemas\ directory.

BV_ExportSpecification.xsd is a supplementary schema. BV_ApplicationActions.xsd imports BV_ExportSpecification.xsd to define the structure of the "exportSpecification" (p. 108) element. BV_XmlApiRequest.xsd is the main schema of the XML API request. Use the main schema to validate the "exportData" (p. 75) action.

For each element, this section provides the following:

- the name and description of the element
- information about attributes that apply to the element, including each attribute's name, description, optionality, legal values, and default value, if applicable
- content model information, consisting of a list of valid child elements presented as an element model group
- a list of valid parent elements

**Element model group**

The list of children for each element is presented using the following standard notation.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plus sign (+)</td>
<td>The preceding element can be repeated more than once but must occur at least once.</td>
</tr>
<tr>
<td>Question mark (?)</td>
<td>The preceding element is optional. It can be absent or it can occur exactly once.</td>
</tr>
<tr>
<td>Asterisk (*)</td>
<td>An asterisk (*) after an element specifies that the element is optional. It can occur zero or more times.</td>
</tr>
<tr>
<td>Symbol</td>
<td>Meaning</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>None</td>
<td>If an element has no plus sign (+), question mark (?), or asterisk (*) following it, the element must occur only once.</td>
</tr>
<tr>
<td>#PCDATA</td>
<td>The symbol #PCDATA specifies character data that is parsed.</td>
</tr>
<tr>
<td>Parentheses</td>
<td>Parentheses group elements. Element groups are controlled using the same symbols as elements.</td>
</tr>
<tr>
<td>Bar (</td>
<td>)</td>
</tr>
<tr>
<td>Comma (,)</td>
<td>The elements that it separates must be present in the specified order.</td>
</tr>
</tbody>
</table>

### dimensionVersion

Specifies which dimension version(s) should be exported. If no specific lists, hierarchies, or member sets are specified, then the entire dimension will be exported.

### Attributes

- **bvObjectId**
  - Business Viewpoint object ID.
  - **Usage:** optional. **Type:** long.

### Content Model

```
((versionId+ | numLatestVersions), (list* | hierarchy* | memberSet* )* )
```

### Parent Elements

- exportSpecification

### exportSpecification

Specifies the dimension that is to be exported and the export security rules.

### Attributes

- **orderExportResults**
  - Ordering of the export results. Setting this to true has serious performance implications and should not be used in production. Default setting is "false".
  - **Usage:** optional. **Type:** boolean.
Content Model

(dimensionVersion+ | securityRulesExport)

hierarchy

Specifies the object ID of the hierarchy to be exported.

Attributes

 bvObjectId
Business Viewpoint object ID.
Usage: required. Type: long.

exportAsManufacturedLevels
Exports a parent-child format in level-based format by manufacturing levels.
Usage: optional. Type: boolean.

exportAsParentChild
Exports a level-based hierarchy in parent-child format.
Usage: optional. Type: boolean.

Content Model

(level*)?

Parent Elements

dimensionVersion

level

Specifies the object ID of the level to be exported.

Attributes

 bvObjectId
Business Viewpoint object ID.
Usage: required. Type: long.

Content Model

Empty element.
Chapter 12: Export specification

**Parent Elements**

- `hierarchy`

**list**

- Specifies the object ID of the list to be exported.

**Attributes**

- `bvObjectId`
  - Business Viewpoint object ID.
  - Usage: required. Type: long.

**Content Model**

- Empty element.

**Parent Elements**

- `dimensionVersion`

**memberSet**

- Specifies the object ID of the member set to be exported.

**Attributes**

- `bvObjectId`
  - Business Viewpoint object ID.
  - Usage: required. Type: long.

**Content Model**

- Empty element.

**Parent Elements**

- `dimensionVersion`

**numLatestVersions**

- Specifies the number of latest versions that should be exported.

**Content Model**

- Content type is int.
Greater than or equal to: 1

Parent Elements

dimensionVersion

securityAttribute

Specifies the mapping between predefined security attributes and the output result.

Attributes

role

Predefined role of the security attribute.

Usage: required.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crud</td>
<td>Security rule, the value is based on the mapping defined by securityMapElement.</td>
</tr>
<tr>
<td>Name</td>
<td>The name of the Business Viewpoint object (for internal use only).</td>
</tr>
<tr>
<td>bvObjectId</td>
<td>Business Viewpoint object ID</td>
</tr>
<tr>
<td>CamId</td>
<td>IBM Cognos CAM (Cognos Access Manager) ID of the Content Manager security identity (user, group, or role).</td>
</tr>
<tr>
<td>systemUrl</td>
<td>IBM Cognos URL referencing the instance of Content Manager where the security identity is defined (for internal use only).</td>
</tr>
<tr>
<td>UserId</td>
<td>Name of the Content Manager security identity (user, group, or role).</td>
</tr>
<tr>
<td>Namespace</td>
<td>The name of the IBM Cognos namespace where the security identity is defined.</td>
</tr>
<tr>
<td>NamespaceCamId</td>
<td>IBM Cognos namespace CAM (Cognos Access Manager) ID, where the security identity is defined.</td>
</tr>
<tr>
<td>UserType</td>
<td>The security identity type (user, role, or group).</td>
</tr>
</tbody>
</table>
**textCase**
Controls the case of output text for attribute values. For internal use only.
*Usage:* optional.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>upperCase</td>
<td>For internal use only</td>
</tr>
<tr>
<td>lowerCase</td>
<td>For internal use only</td>
</tr>
</tbody>
</table>

**Content Model**
Mixed content.

**Parent Elements**
- `securityRule`

**securityMapElement**
Specifies the mapping between the CRUD security rule and the output rule name.

**Attributes**

- **destinationRule**
  Security rule specification for the external application.
  *Usage:* required. *Type:* string.

- **sourceRule**
  CRUD security rule specified in Business Viewpoint.
  *Usage:* required. *Type:* string.

**Content Model**
Empty element.

**Parent Elements**
- `securityRule`

**securityRule**
Specifies the output format for the security rules.
Content Model
   (securityAttribute+, securityMapElement*)

Parent Elements
   securityRulesExport

securityRulesExport
   Specifies an export definition for the security rules.

Content Model
   (securityRule)

Parent Elements
   exportSpecification

versionId
   Specifies which versions should be exported by ID.

Content Model
   Content type is long.

Parent Elements
   dimensionVersion
Chapter 12: Export specification
Chapter 13: XML API response reference

The XML API response schema defines the structure of the XML document returned by IBM® Cognos® Business Viewpoint in response to an XML API request. Use the schema details documented in this chapter to design the code that parses the XML response.

The response schema is defined in BV_XmlApiResponse.xsd, which is located in the `<installation_location>\sdk\xmlapi\schemas\` directory.

For each element, this section provides the following:

- the name and description of the element
- information about attributes that apply to the element, including each attribute’s name, description, optionality, legal values, and default value, if applicable
- content model information, consisting of a list of valid child elements presented as an element model group
- a list of valid parent elements

### Element model group

The list of children for each element is presented using the following standard notation.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plus sign (+)</td>
<td>The preceding element can be repeated more than once but must occur at least once.</td>
</tr>
<tr>
<td>Question mark (?)</td>
<td>The preceding element is optional. It can be absent or it can occur exactly once.</td>
</tr>
<tr>
<td>Asterisk (*)</td>
<td>An asterisk (*) after an element specifies that the element is optional. It can occur zero or more times.</td>
</tr>
<tr>
<td>None</td>
<td>If an element has no plus sign (+), question mark (?), or asterisk (*) following it, the element must occur only once.</td>
</tr>
<tr>
<td>#PCDATA</td>
<td>The symbol #PCDATA specifies character data that is parsed.</td>
</tr>
<tr>
<td>Parentheses</td>
<td>Parentheses group elements. Element groups are controlled using the same symbols as elements.</td>
</tr>
<tr>
<td>Bar (</td>
<td>)</td>
</tr>
</tbody>
</table>
### Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commas (,)</td>
<td>The elements that it separates must be present in the specified order.</td>
</tr>
</tbody>
</table>

### actionResponse

The root element of the XML API response specification.

### Content Model

```xml
(runLinkResult | createResult | deleteResult | modifyResult | findResult | getResult | importDataResult |
  | exportDataResult | createVersionResult | runPublicationResult | createUserResult | deleteUserResult |
  | setObjectPermissionsResult)
```

### association

Specifies the association of an object.

#### Attributes

- **id**
  
  Specifies a Business Viewpoint object ID. The import action returns empty IDs in preview mode.
  
  **Usage:** required. **Type:** long.

- **name**
  
  Specifies the name of a Business Viewpoint property.
  
  **Usage:** required. **Type:** string.

### Content Model

```xml
(associationCount | associationObjectID? | classType? | dataType? | description? | isCore | isHighlighted | key | maxOccurs | name? | order | value | whatAmI? | modifiedBy? | modifiedDate? | systemVersion? | depthFromRoot? | isInternal?)+
```

### Parent Elements

- associations

### associationCount

Specifies the number of the object associations returned by the find action.

#### Content Model

Content type is int.
Parent Elements
association, objectValue, property, valueContexts, viewContexts

associationObjectID
Reserved for internal use.

Content Model
Content type is long.

Parent Elements
association, objectValue, property, valueContexts, viewContexts

associations
 Specifies the list of object associations.

Content Model
(association+)

Parent Elements
object

class
Returns the name of the Business Viewpoint class that was created, deleted, or modified.

Content Model
Empty element.

Parent Elements
createResult, createUserResult, createVersionResult, deleteResult, deleteUserResult, modifyResult, setObjectPermissionsResult

classType
Specifies the name of the object class. For more information on class types, see "Business Viewpoint class reference" (p. 55)

Content Model
Empty element.
Parent Elements

association, objectValue, property, valueContexts, viewContexts

create

Returns the results of the setObjectPermissions action. If the action is successful, this element will include a copy of the original request. If the permission was not specified in the original request, the result is true, by default.

Content Model

Content type is boolean.

Parent Elements

permissions

What's New

Version 8.4.1

The create element has been added.

createResult

Returns the results of the create action. The class and objectID of the new repository object are provided in the response.

Content Model

(class? objectID)

Parent Elements

actionResponse

createUserResult

Returns the results of the createUser action. The class and objectID of the new user object are provided in the response.

Content Model

(class? objectID)

Parent Elements

actionResponse
**createVersionResult**

Returns the results of the `createVersion` action. The `class` and `objectID` of the new version object is returned in the response.

**Content Model**

(class? objectID)

**Parent Elements**

actionResponse

**dataType**

Specifies the data type of a property (for example, `String`).

**Content Model**

Empty element.

**Parent Elements**

association, objectValue, property, valueContexts, viewContexts

**delete**

Returns the results of the `setObjectPermissions` action. If the action is successful, this element will include a copy of the original request. If the permission was not specified in the original request, the result is true, by default.

**Content Model**

Content type is boolean.

**Parent Elements**

permissions

**What's New**

**Version 8.4.1**

The delete element has been added.

**deleteResult**

Returns the results of the `delete` action. The `class` and `objectID` of the deleted repository object are provided in the response.
**Content Model**

```
(class? objectID)
```

**Parent Elements**

```
actionResponse
```

**deleteUserResult**

Returns the results of the `deleteUser` action. The `class` and `objectId` of the deleted user object are provided in the response.

**Content Model**

```
(class? objectID)
```

**Parent Elements**

```
actionResponse
```

**What's New**

**Version 8.4.1**

The `deleteUserResult` element has been added.

**depthFromRoot**

Reserved for internal use.

**Content Model**

Empty element.

**Parent Elements**

```
association, objectValue, property, valueContexts, viewContexts
```

**description**

Specifies the internal description of an object or property.

**Content Model**

Empty element.

**Parent Elements**

```
association, objectValue, property, valueContexts, viewContexts
```
The result type returned by the `exportData` and `runPublication` actions. The `exportData` action returns one instance of this type. The `runPublication` action can return one or more instances of this type.

The result of the `exportData` action.

The result of the export action performed by the publication.

**Content Model**

(exportTargets, exportSummary)

**Parent Elements**

actionResponse, runPublicationResult

**exportSummary**

Provides a summary of the exported objects.

**Content Model**

(objectSummary*)

**Parent Elements**

exportDataResult

**exportTarget**

Specifies the filename and path of the export target.

**Content Model**

Empty element.

**Parent Elements**

exportTargets

**exportTargets**

Contains one or more export targets created by the `exportData` action.

**Content Model**

(exportTarget+)
Parent Elements

exportDataResult

external

Returns the results of the setObjectPermissions action. If the action is successful, this element will include a copy of the original request. If the permission was not specified in the original request, the result is false, by default; indicating that internal permissions will be applied to the specific user.

Content Model

Content type is boolean.

Parent Elements

setObjectPermissionsResult

What's New

Version 8.4.1

The external element has been added.

findResult

Returns the results of the find action. The list of the repository objects that match the search criteria specified in the find request is included in the response.

Content Model

(objects)

Parent Elements

actionResponse

getResult

Returns the results of the get action. The repository object requested in the get action is included in the response.

Content Model

(objects)

Parent Elements

actionResponse
**importDataResult**
Returns the result of the `importData` action. A comprehensive list of imported objects and surrogate key group ID are included in the response.

**Content Model**
(surrogateKeyGroupId, importSummary)

**Parent Elements**
actionResponse

**importSummary**
Provides a summary of the imported objects.

**Content Model**
(objectSummary*)

**Parent Elements**
importDataResult

**isCore**
Reserved for internal use.

**Content Model**
Content type is boolean.

**Parent Elements**
association, objectValue, property, valueContexts, viewContexts

**isHighlighted**
Reserved for internal use.

**Content Model**
Content type is boolean.

**Parent Elements**
association, objectValue, property, valueContexts, viewContexts
isInternal
Reserved for internal use.

Content Model
Empty element.

Parent Elements
association, objectValue, property, valueContexts, viewContexts

key
Specifies the unique ID of a property, or PRIMARY_KEY_PARAM when the item represents an object or a class descriptor.

Content Model
Empty element.

Parent Elements
association, objectValue, property, valueContexts, viewContexts

maxOccurs
Specifies the cardinality of the reference-type property.

Content Model
Empty element.

Parent Elements
association, objectValue, property, valueContexts, viewContexts

modifiedBy
Specifies the name of the user who created or modified the item.

Content Model
Empty element.

Parent Elements
association, objectValue, property, valueContexts, viewContexts
modifiedDate
Specifies the date when the item was created or modified.

Content Model
Empty element.

Parent Elements
association, objectValue, property, valueContexts, viewContexts

modifyResult
Returns the results of the modify action. The class and objectID of the modified object are provided in the response.

Content Model
(class? objectID)

Parent Elements
actionResponse

name
Specifies the name of a property or class. Not used if the item represents an object.

Content Model
Empty element.

Parent Elements
association, objectValue, property, valueContexts, viewContexts

namespace
Returns the results of the setObjectPermissions action. If the action is successful, this element will include a copy of the original request.

Specifies the namespace in the external authentication system.

Content Model
Empty element.
Parent Elements

securityIdentity

What's New

Version 8.4.1
The namespace element has been added.

object

Specifies the repository object returned by the find action.

Attributes

id
Specifies a Business Viewpoint object ID. The import action returns empty IDs in preview mode.
Usage: required.

version — deprecated
Specifies the version of an object. The find action returns the last version of the object.
Usage: optional.

What's New  Version 8.4.1
The version attribute is deprecated and will be removed in a future version of the product.

Content Model

(objectValue properties? associations? viewContexts? valueContexts?)

Parent Elements

objects

objectId

Specifies the Business Viewpoint object ID.

Content Model

Empty element.

Parent Elements

createResult, createUserResult, createVersionResult, deleteResult, deleteUserResult, modifyResult, setObjectPermissionsResult
objects

Specifies the objects returned by the find action. This element includes one or more objects.

Attributes

count

Specifies the number of objects returned. By default, the find action returns 50 objects (or less, if the totalCount value is less than 50). If the totalCount is more than 50, use the START_QUERY_RANGE_POSITION or/and QUERY_RANGE_REQUESTED_COUNT hints to retrieve all objects.

Usage: required.

totalCount

Specifies how many objects of this class were transferred.

Usage: required.

Content Model

(object*)

Parent Elements

findResult, getResult

objectSummary

Specifies the summary of the transferred objects. The importSummary includes a separate objectSummary for each combination of class and modificationType. For example, created dimensions, modified dimensions, created hierarchies, modified hierarchies, and so on. The exportSummary includes a separate objectSummary for each exported class. For example, exported dimensions, exported hierarchies, exported levels, and so on.

Attributes

class

Specifies the class of objects in an objectSummary, such as dimension, hierarchy, level, and member.

Usage: required.

modificationType

Modification type can be "created" or "updated". This attribute is present in the importSummary, but not in the exportSummary.

Usage: optional.
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>created</td>
<td>The objectSummary contains the objects created by the importData action.</td>
</tr>
<tr>
<td>updated</td>
<td>The objectSummary contains the objects updated by the importData action.</td>
</tr>
</tbody>
</table>

**totalCount**

Specifies how many objects of this class were transferred.

Usage: required.

**Content Model**

(sampleObjects)

**Parent Elements**

exportSummary, importSummary

**objectValue**

Specifies the properties of a repository object.

Note: Some of the represented properties are reserved for internal use only.

**Content Model**

(associationCount | associationObjectID? | classType? | dataType? | description? | isCore | isHighlighted | key | maxOccurs | name? | order | value | whatAmI? | modifiedBy? | modifiedDate? | systemVersion? | depthFromRoot? | isInternal?)+

**Parent Elements**

object

**order**

Reserved for internal use.

**Content Model**

Empty element.

**Parent Elements**

association, objectValue, property, valueContexts, viewContexts
permissions

Returns the results of the setObjectPermissions action. If the action is successful, this element will include a copy of the original request. If permissions are not specified in the original request, the setting for each permission type is true, by default.

Content Model

(create, read, update, delete, propagate)

Parent Elements

setObjectPermissionsResult

What's New

Version 8.4.1
The permissions element has been added.

propagate

Returns the results of the setObjectPermissions action. If the action is successful, this element will include a copy of the original request. If the permission was not specified in the original request, the result is true, by default.

Content Model

Content type is boolean.

Parent Elements

permissions

What's New

Version 8.4.1
The propagate element has been added.

properties

Specifies the list of object properties. Contains one or more property elements.

Content Model

(property*)
Parent Elements

property

Specifies an object property.

Attributes

id

Specifies a Business Viewpoint object ID. The import action returns empty IDs in preview mode.
Usage: required. Type: long.

name

Specifies the name of a Business Viewpoint property.
Usage: required. Type: string.

Content Model

(associationCount | associationObjectID? | classType? | dataType? | description? | isCore | isHighlighted | key | maxOccurs | name? | order | value | whatAmI? | modifiedBy? | modifiedDate? | systemVersion? | depthFromRoot? | isInternal?)+

Parent Elements

properties

read

Returns the results of the setObjectPermissions action. If the action is successful, this element will include a copy of the original request. If the permission was not specified in the original request, the result is true, by default.

Content Model

Content type is boolean.

Parent Elements

permissions

What's New

Version 8.4.1
The read element has been added.
**runLinkResult**

Returns the results of the runLink action, consisting of a status message and additional detailed information, if available.

**Content Model**

(statusMessage, statusDetails)

**Parent Elements**

actionResponse

**What's New**

**Version 10.1.0**
The runLinkResult element has been added.

**runPublicationResult**

Returns the result of the runPublication action. The exportTarget(s) and summary of exported objects are provided in the response.

**Content Model**

(exportDataResult+)

**Parent Elements**

actionResponse

**sampleObject**

Specifies a sample transferred object.

**Attributes**

**id**
Specifies a Business Viewpoint object ID. The import action returns empty IDs in preview mode.

*Usage: required.*

**path**
Specifies the path of an object. The path contains the IDs of the parent objects, starting from the top-most object (Dimension) and progressing down to the object itself.

*Usage: required.*
Chapter 13: XML API response reference

Content Model
Empty element.

Parent Elements
sampleObjects

sampleObjects
Contains one or more sample transferred objects. The number of sample objects included in the summary can be less than the number specified in totalCount.

Content Model
(sampleObject+)

Parent Elements
objectSummary

securityIdentity
Returns the results of the setObjectPermissions action. If the action is successful, the result is a copy of the executed request.

Content Model
(systemName, namespace, userId)

Parent Elements
setObjectPermissionsResult

What's New

Version 8.4.1
The securityIdentity element has been added.

setObjectPermissionsResult
Returns the results of the setObjectPermissions action. If the action is successful, the result is a copy of the executed request.

Content Model
(class, objectID, securityIdentity, permissions?, external)
Parent Elements

statusDetails
Contains detailed status information from a runLink request. Depending on the status, the results can be empty.

Content Model
Empty element.

Parent Elements

runLinkResult

What's New

Version 10.1.0
The statusDetails element has been added.

statusMessage
Contains the status message from a runLink request. For example, Running link complete.

Content Model
Empty element.

Parent Elements

runLinkResult

What's New

Version 10.1.0
The statusMessage element has been added.

surrogateKeyId
Specifies the unique ID of the surrogate key group created by an importData action.

Content Model
Empty element.
Parent Elements

importDataResult

systemName

Returns the results of the setObjectPermissions action. If the action is successful, this element will include a copy of the original request.

Specifies the IBM Cognos system name, as defined in Business Viewpoint Studio.

Content Model

Empty element.

Parent Elements

securityIdentity

What's New

Version 8.4.1

The systemName element has been added.

systemVersion

Reserved for internal use.

Content Model

Empty element.

Parent Elements

association, objectValue, property, valueContexts, viewContexts

update

Returns the results of the setObjectPermissions action. If the action is successful, this element will include a copy of the original request. If the permission was not specified in the original request, the result is true, by default.

Content Model

Content type is boolean.

Parent Elements

permissions
What's New

Version 8.4.1
The update element has been added.

userId
Returns the results of the setObjectPermissions action. If the action is successful, this element will include a copy of the original request.
Specifies the user ID in the external authentication system.

Content Model
Empty element.

Parent Elements
securityIdentity

What's New

Version 8.4.1
The userId element has been added.

value
Specifies the value of a property, ID of an object, or ID of an association.

Content Model
Empty element.

Parent Elements
association, objectValue, property, valueContexts, viewContexts

valueContexts
Reserved for internal use.

Attributes

id
Specifies a Business Viewpoint object ID. The import action returns empty IDs in preview mode.
Usage: required. Type: long.
name
Specifies the name of a Business Viewpoint property.
Usage: required. Type: string.

Content Model
(associationCount | associationObjectID? | classType? | dataType? | description? | isCore | isHighlighted | key | maxOccurs | name? | order | value | whatAmI? | modifiedBy? | modifiedDate? | systemVersion? | depthFromRoot? | isInternal?)+

Parent Elements
object

viewContexts
Reserved for internal use.

Attributes
id
Specifies a Business Viewpoint object ID. The import action returns empty IDs in preview mode.
Usage: required. Type: long.

name
Specifies the name of a Business Viewpoint property.
Usage: required. Type: string.

Content Model
(associationCount | associationObjectID? | classType? | dataType? | description? | isCore | isHighlighted | key | maxOccurs | name? | order | value | whatAmI? | modifiedBy? | modifiedDate? | systemVersion? | depthFromRoot? | isInternal?)+

Parent Elements
object

whatAmI
Reserved for internal use.

Content Model
Empty element.
Parent Elements

association, objectValue, property, valueContexts, viewContexts
Appendix A: Application actions reference release notes

The following changes have been made to the application actions specification.

The release note sets have the following sections:

- **Added** means the listed items have been added to the reference specification.
- **Deprecated** means the listed items are deprecated and will be removed in a future version of the product. These items are still functional but should be removed from all of your software development kit applications.
- **Obsolete** means that the listed items are obsolete and will be removed in a future version of the product.
- **Internal** means the listed items are for internal use only and should not be added to your software development kit applications.

Schema Version 8.4.1

**Added**

- `applicationRoles` element.
- `class` element.
- `create` element.
- `createUser` element.
- `delete` element.
- `deleteUser` element.
- `dimension` element.
- `email` element.
- `external` element.
- `name` element.
- `objectId` element.
- `permissions` element.
- `propagate` element.
- `publication` element.
Appendix A: Application actions reference release notes

- read element.
- role element.
- runPublication element.
- securityIdentity element.
- setObjectPermissions element.
- systemName element.
- update element.
- versionDescription element.
- Administrator enumeration value of role element.
- Modeler enumeration value of role element.
- Reviewer enumeration value of role element.
- Nominator enumeration value of role element.
- Consumer enumeration value of role element.

**Deprecated**
- No deprecated items.

**Obsolete**
- No obsolete items.

**Internal**
- No internal-only items.

**Schema Version 10.1.0**

**Added**
- linkName element.
- runLink element.
- sourceDataStoreAuthentication element.
- targetDataStoreAuthentication element.

**Deprecated**
- No deprecated items.
Obsolete
  • No obsolete items.

Internal
  • No internal-only items.
The following changes have been made to the repository actions specification.

The release note sets have the following sections:

- **Added** means the listed items have been added to the reference specification.
- **Deprecated** means the listed items are deprecated and will be removed in a future version of the product. These items are still functional but should be removed from all of your software development kit applications.
- **Obsolete** means that the listed items are obsolete and will be removed in a future version of the product.
- **Internal** means the listed items are for internal use only and should not be added to your software development kit applications.

### Schema Version 8.4.1

**Added**

- PROPERTY_VALIDATION_STATE enumeration value of name element.

**Deprecated**

- No deprecated items.

**Obsolete**

- No obsolete items.

**Internal**

- No internal-only items.
Appendix C: Import specification reference release notes

The following changes have been made to the import specification.

The release note sets have the following sections:

- **Added** means the listed items have been added to the reference specification.
- **Deprecated** means the listed items are deprecated and will be removed in a future version of the product. These items are still functional but should be removed from all of your software development kit applications.
- **Obsolete** means that the listed items are obsolete and will be removed in a future version of the product.
- **Internal** means the listed items are for internal use only and should not be added to your software development kit applications.

**Schema Version 8.4.1**

**Added**
- bvPropertyExtensionId attribute of annotation element.
- shareMembers attribute of dimension element.
- isHidden attribute of role element.
- annotation element.

**Deprecated**
- No deprecated items.

**Obsolete**
- No obsolete items.

**Internal**
- No internal-only items.
The following changes have been made to the API response specification.

The release note sets have the following sections:

- **Added** means the listed items have been added to the reference specification.
- **Deprecated** means the listed items are deprecated and will be removed in a future version of the product. These items are still functional but should be removed from all of your software development kit applications.
- **Obsolete** means that the listed items are obsolete and will be removed in a future version of the product.
- **Internal** means the listed items are for internal use only and should not be added to your software development kit applications.

### Schema Version 8.4.1

**Added**

- `create` element.
- `delete` element.
- `deleteUserResult` element.
- `external` element.
- `namespace` element.
- `permissions` element.
- `propagate` element.
- `read` element.
- `securityIdentity` element.
- `systemName` element.
- `update` element.
- `userId` element.

**Deprecated**

- version attribute of `object` element.
Appendix D: XML API response reference release notes

**Obsolete**
- No obsolete items.

**Internal**
- No internal-only items.

**Schema Version 10.1.0**

**Added**
- `runLinkResult` element.
- `statusDetails` element.
- `statusMessage` element.

** Deprecated**
- No deprecated items.

**Obsolete**
- No obsolete items.

**Internal**
- No internal-only items.
Index

A
action element, 66
actionResponse element, 116
administrator enumeration value, 82
annotation element, 96
applicationRoles element, 70
applySource enumeration value, 101, 103
associationCount element, 116
association element, 116
associationObjectId element, 117
associations element, 117
asynchReply class, 47
attribute
  bvObjectId, 98, 99, 100, 101, 108, 109, 110
  bvParentMemberObjectId, 99
  bvPropertyExtensionId, 99
  bvPropertyId, 102
cardinality, 97
class, 127
count, 127
crossReferenceAllLists, 98
defaultUpdateRule, 101
delimiter, 72, 73
destinationRule, 112
encoding, 73
exportAsManufacturedLevels, 109
exportAsParentChild, 109
id, 100, 116, 126, 130, 131, 135, 136
isHidden, 102
isIdenfitier, 102
listId, 97
metadataId, 102
modificationType, 127
name, 96, 102, 116, 130, 136
nullMemberContainerName, 99
orderExportResults, 108
path, 131
qualifier, 72, 73
role, 111
shareMembers, 98

source, 103
sourceRule, 112
suppressNullMembers, 99
textCase, 112
totalCount, 127, 128
updateRule, 103
version, 126

B
bvObjectId attribute, 98, 99, 100, 101, 108, 109, 110
bvObjectId enumeration value, 111
bvParentMemberObjectId attribute, 99
bvPropertyExtensionId attribute, 99
bvPropertyId attribute, 102

c
C
C# .NET
  services, 43
CamId enumeration value, 111
cardinality attribute, 97
class attribute, 127
class element, 70, 90, 117
classes
  Java, 47
  relationships, 47
classType element, 117
cognos8Package element, 71
connections
  connecting to IBM Cognos, 37
  connecting to the Business Viewpoint Web Service, 31
constraint element, 90
constraints element, 91
consumer enumeration value, 83
count attribute, 127
created enumeration value, 128
create element, 71, 91, 118
createResult element, 118
createUser element, 72
createUserResult element, 118
createVersion element, 72

Licensed Materials – Property of IBM
Index

createVersionResult element, 119
crossReferenceAllLists attribute, 98
crossReference element, 97
Crud enumeration value, 111
csvFile element, 72, 73

crossReference, 97
csvFile, 72, 73
dataType, 119
delete, 73, 91, 119
deleteResult, 119
deleteUser, 74
deleteUserResult, 120
deleteUser, 74
delimiter attribute, 72, 73
depthFromRoot element, 120
description element, 74, 120
destinationRule attribute, 112
dimension element, 74, 97
dimensionManagementService class, 48
dimensionManagementService service, 43
dimensionManagementServiceSpecification class, 52
dimensionVersion element, 108
dynamicHierarchy element, 98

deleteResult element, 119
deleteUser element, 74
deleteUserResult element, 120
delimiter attribute, 72, 73
depthFromRoot element, 120
description element, 74, 120
destinationRule attribute, 112
dimension element, 74, 97
dimensionManagementService class, 48
dimensionManagementService service, 43
dimensionManagementServiceSpecification class, 52
dimensionVersion element, 108
dynamicHierarchy element, 98

element
action, 66
actionResponse, 116
annotation, 96
applicationRoles, 70
association, 116
associationCount, 116
associationObjectID, 117
associations, 117
class, 70, 90, 117
classType, 117
cognos8Package, 71
constraint, 90
constraints, 91
create, 71, 91, 118
createResult, 118
createUser, 72
createUserResult, 118
createVersion, 72
createVersionResult, 119
csvFile element, 72, 73
dataType, 119
delete, 73, 91, 119
deleteResult, 119
deleteUser, 74
deleteUserResult, 120
depthFromRoot, 120
description, 74, 120
dimension, 74, 97
dimensionVersion, 108
dynamicHierarchy, 98
email, 75
eexc, 75
exportData, 75
exportDataResult, 121
exportSpecification, 108
exportSummary, 121
exportTarget, 75, 121
exportTargets, 121
external, 76, 122
filePath, 76
find, 91
findResult, 122
fmModel, 76
get, 91
getResult, 122
groupId, 77
hierarchy, 99, 109
hint, 91
hints, 92
importData, 77
importDataResult, 123
importSource, 77
importSpecification, 100
importSummary, 123
isCore, 123
isHighlighted, 123
isInternal, 124
key, 124
level, 100, 109
linkName, 77
list, 100, 110
maxOccurs, 124
memberApex, 101
members, 101

150 IBM Cognos Business Viewpoint Server
memberSet, 110
modifiedBy, 124
modifiedDate, 125
modify, 92
modifyResult, 125
name, 78, 92, 125
namespace, 78, 79, 125
numLatestVersions, 110
object, 126, 131
objectId, 79, 94, 126
objects, 127
objectSummary, 127
objectValue, 128
order, 128
packageName, 79
password, 80
permissions, 80, 129
previewOnly, 81
propagate, 81, 129
properties, 129
property, 94, 130
publication, 81
read, 82, 130
requestedProperties, 94
role, 82, 102
runLink, 83
runLinkResult, 131
runPublication, 83
runPublicationResult, 131
sampleObject, 131
sampleObjects, 132
searchPath, 84
securityAttribute, 111
securityIdentity, 84, 132
securityMapElement, 112
securityRule, 112
securityRulesExport, 113
setObjectPermissions, 84
setObjectPermissionsResult, 132
sourceDataStoreAuthentication, 85
statusDetails, 133
statusMessage, 133
surrogateKeyGroup, 85
surrogateKeyGroupId, 133
systemName, 85, 134
systemVersion, 134
targetDataStoreAuthentication, 86
update, 86, 134
userId, 86, 87, 135
value, 94, 135
valueContexts, 135
versionDescription, 87
versionId, 113
versionName, 88
versionType, 88
viewContexts, 136
whatAml, 136
element model group notation, 65, 69, 89, 95, 105, 107, 115
e-mail element, 75
e-mail encoding attribute, 73
enumeration value
administrator, 82
applySource, 101, 103
bvObjectId, 111
CamId, 111
consumer, 83
created, 128
Crud, 111
dataItem, 103
e-xpression, 103
GET_SUMMARY, 93
lowerCase, 112
maintainTarget, 101, 103
modeler, 82
multiple, 97
Name, 111
Namespace, 111
NamespaceCamId, 111
nominator, 83
private, 88
PROPERTY_VALIDATION_STATE, 93
public, 88
QUERY_RANGE_REQUESTED_COUNT, 93
REFERENCE_DELETED_OBJECTS, 93
reviewer, 82
single, 97
SNAPSHOT_REFERENCE, 93
SORT_ORDER, 93
SORT_PROPERTY, 93
START_QUERY_RANGE_POSITION, 92
systemUrl, 111
Index

text, 103
updated, 128
upperCase, 112
UserId, 111
 UserType, 111
excel element, 75
exceptions, 63
execute method, 59
exportAsManufacturedLevels attribute, 109
exportAsParentChild attribute, 109
exportData element, 75
exportDataResult element, 121
exportSpecification element, 108
exportSummary element, 121
exportTarget element, 75, 121
exportTargets element, 121
expression enumeration value, 103
external element, 76, 122

F
filePath element, 76
find element, 91
findResult element, 122
fmModel element, 76

G
GET_SUMMARY enumeration value, 93
get element, 91
getResult element, 122
groupId element, 77

H
hierarchy element, 99, 109
hint element, 91
hints element, 92

I
IBM Business Viewpoint Service, 29
IBM Cognos Software Development Kit Reference Information, 42
IBM Cognos Software Development Kit with Business Viewpoint using, 37
id attribute, 100, 116, 126, 130, 131, 135, 136
importData element, 77
importDataResult element, 123
importSource element, 77
importSpecification element, 100
importSummary element, 123
isCore element, 123
isHidden attribute, 102
isHighlighted element, 123
isIdentifier attribute, 102
isInternal element, 124

J
Java
classes, 47
services, 42, 44

K
key element, 124

L
level element, 100, 109
linkName element, 77
list element, 100, 110
listId attribute, 97
logging on, 32, 38
logOff method, 61
logOn method, 60
lowerCase enumeration value, 112

M
maintainTarget enumeration value, 101, 103
Master Dimension Class Objects Reference, 55
maxOccurs element, 124
memberApex element, 101
members element, 101
memberSet element, 110
metadataId attribute, 102
modeler enumeration value, 82
modificationType attribute, 127
modifiedBy element, 124
modifiedDate element, 125
modify element, 92
modifyResult element, 125
multiple enumeration value, 97

N
name attribute, 96, 102, 116, 130, 136
name element, 78, 92, 125
Name enumeration value, 111
NamespaceCamId enumeration value, 111
namespace element, 78, 79, 125
Namespace enumeration value, 111
nominator enumeration value, 83
nullMemberContainerName attribute, 99
numLatestVersions element, 110

object element, 126, 131
objectID element, 79, 94, 126
objects element, 127
objectSummary element, 127
objectValue element, 128
order element, 128
orderExportResults attribute, 108

packageName element, 79
password element, 80
path attribute, 131
permissions element, 80, 129
ping method, 62
previewOnly element, 81
private enumeration value, 88
propagate element, 81, 129
properties element, 129
PROPERTY_VALIDATION_STATE enumeration value, 93
property element, 94, 130
publication element, 81
public enumeration value, 88

qualifier attribute, 72, 73
QUERY_RANGE_REQUESTED_COUNT enumeration value, 93

read element, 82, 130
REFERENCE_DELETED_OBJECTS enumeration value, 93
relationships
    between classes, 47
report specifications
    using, 21
requestedProperties element, 94
reviewer enumeration value, 82
role attribute, 111
role element, 82, 102
runLink element, 83
runLinkResult element, 131
runPublication element, 83
runPublicationResult element, 131
runSpecification(specification, parameterValues, options), 44

sampleObjects element, 132
searchPath element, 84
securityAttribute element, 111
securityIdentity element, 84, 132
securityMapElement element, 112
securityRule element, 112
securityRulesExport element, 113
services
    C# .NET, 43
    Java, 42, 44
setObjectPermissions element, 84
setObjectPermissionsResult element, 132
shareMembers attribute, 98
single enumeration value, 97
SNAPSHOT_REFERENCE enumeration value, 93
SORT_ORDER enumeration value, 93
SORT_PROPERTY enumeration value, 93
source attribute, 103
sourceDataStoreAuthentication element, 85
sourceRule attribute, 112
START_QUERY_RANGE_POSITION enumeration value, 92
statusDetails element, 133
statusMessage element, 133
Submitting a request, 33, 39
suppressNullMembers attribute, 99
surrogateKeyGroup element, 85
surrogateGroupId element, 133
systemName element, 85, 134
systemUrl enumeration value, 111
systemVersion element, 134

targetDataStoreAuthentication element, 86

Developer Guide 153
Index

textCase attribute, 112
text enumeration value, 103
totalCount attribute, 127, 128
trustedLogon method, 63

U
updated enumeration value, 128
update element, 86, 134
updateRule attribute, 103
upperCase enumeration value, 112
userId element, 86, 87, 135
UserId enumeration value, 111
UserType enumeration value, 111

V
valueContexts element, 135
value element, 94, 135
version attribute, 126
versionDescription element, 87
versionId element, 113
versionName element, 88
versionType element, 88
viewContexts element, 136

W
whatAmI element, 136