COSMOS

User's Guide

Version 1 Release 0

COSMOS

IBM

User's Guide

Version 1 Release 0

Note

Before using this information and the product it supports, read the information in .

First Edition (May 2008)

This edition applies to version 1, release 1, modification 0 of COSMOS and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Chapter 1. Overview

Community Systems Management Open Source (COSMOS) is a generic, extensible, standards-based set of components for a tools platform that software developers can use to create specialized, differentiated, and interoperable offerings of tools for system management

This topic provides the following information.

- Project Objectives
- A description of the overall product design
- A description of the design of COSMOS components

What are the problems that COSMOS solves?

The main problems COSMOS is being built to help resolve are the following.

- Integration - see “Addressing the integration problem.”
- Resource modeling for the purpose of exchanging and reconciling resource models. See “Addressing the resource modeling problem.”
- Deployment - see “Addressing the deployment problem” on page 2.

Need information about the standards focus

Addressing the integration problem

- Proliferation of custom databases
- Proliferation of management databases

Addressing the resource modeling problem

In the IT industry, there are a number of systems management applications with proprietary data formats. There has long been a need for an industry-standard format for expressing and modeling configuration management data, and for exchanging this data between applications and vendors.

SML and SML-IF

The Service Modeling Language (SML) is a specification, submitted to W3C, for modeling complex services and systems, including structure, policies, and constraints. SML is based on a subset of XML Schema 1.0. It includes Schematron and XPath 1.0 support for validating constraints. It also includes some new extensions of XML Schema that allow for inter-document references for expressing data relationships.

The SML-Interchange Format (SML-IF) is a peer specification of the SML specification that defines the means for packaging related SML documents as an aggregate. This schema allows for large amounts of data to be easily exchanged between applications, and also provides context for the aggregate. It also gives the ability to bind SML documents to specific Schematron rules

CMDBf's relationship to SML

The CMDBf specification defines an XML syntax for providing query results. Embedded in a query result is a configuration record containing data specific to the repository, possibly using a unique syntax. SML is a recommended syntax for expressing a configuration record, given the unique capabilities it provides over standard XML Schema. The example SML MDR provided with COSMOS makes use of SML references to highlight this capability.

SML and SML-IF COSMOS Tooling

COSMOS provides a rich set of tooling to support working with SML and SML-IF data.

SML-IF editor
This editor allows you to graphically browse and navigate an SML-IF document, and serves as a hub for other SML-IF operations.

SML-IF validator
This tool validates SML documents and SML-IF files against the specification schemas and the Schematron rules, ensuring that the syntax and data conforms to these schemas and ruls.

SML repository
An SML repository provides a flat-file based repository for storing SML documents, including a binding to CMDBf to handle query service requests.

SML-IF import
This function allows you to import SML-IF files into the SML repository as SML documents. These SML-IF files could have been previously exported from another COSMOS SML repository, or even another management application.

SML-IF export
This function allows you to export SML documents from the SML repository into an SML-IF file. This is essentially the reverse operation from the SML-IF import, and allows data to be exchanged with other applications.

See the section Service Modeling Language Tooling and Usage for more information on the included tooling.

Addressing the deployment problem

- Frame the deployment problem
- Position COSMOS to P2, Mayaninstall. How does COSMOS work with and is different from P2 and Mayainstall
- Why is SDD important?

The vision of COSMOS

Pulling it all together

- The end-to-end demo.
- How standards work together

The importance of working as a community

- Understanding the specifications

- Shared plumbing
- Common platform and building blocks

Future Direction

COSMOS prerequisites

This section provides links to prerequisites that are needed for COSMOS

Prerequisites

Table 1.

Prerequisite software	URL
Business Intelligence and Reporting Tools - BIRT 2.3	http://www.eclipse.org/birt/phoenix/
Dojo 1.0.2 Redistribute	http://download.dojotoolkit.org/release-1.0.2/
Apache Tomcat 5.5.23	http://tomcat.apache.org/download-55.cgi#5.5.23
Axis2 1.3	http://ws.apache.org/axis2/download/1_3/download.cgi
Apache Muse 2.2	http://ws.apache.org/muse/update/btm-muse-2.2-developer-download.zip
Drools 4.0.4	http://www.jboss.org/drools
MVEL 1.2.21	http://mvel.codehaus.org/Home
ASM 3.0	http://asm.objectweb.org/
ANTLR 3.0	http://www.antlr.org/
eclipse-jdt-core (ships with Eclipse 3.4)	http://www.eclipse.org/jdt/
ICU4J (ships with Eclipse 3.4)	http://icu-project.org/
SML/SML-IF schema files 1.1	http://www.w3.org/2008/01/sml-schema.xsdhttp://www.w3.org/2008/01/sml-if-schema.xsd
SDD specification files 1.0	http://docs.oasis-open.org/sdd/v1.0/pr01/sdd-spec-v1.0-pr01.html
CMDBf Registration WSDL	http://cmdbf.org/schema/1-0-0/cmdbfRegistration.wsdl
Query WSDL	http://cmdbf.org/schema/1-0-0/cmdbfQuery.wsdl
data model schema	http://cmdbf.org/schema/1-0-0/cmdbfDatamodel.xsd
service metadata schema	http://cmdbf.org/schema/1-0-0/cmdbfServiceMetadata.xsd
WTP R3.0	http://www.eclipse.org/webtools/
EMF R2.4	http://www.eclipse.org/modeling/emf/
GEF R2.4	http://www.eclipse.org/gef/
Eclipse Platform R3.4 (SDK)	http://www.eclipse.org/platform/
XSD Runtime R2.4 Prereq	http://www.eclipse.org/modeling/emf/

Chapter 2. Managing COSMOS components

COSMOS consists of the following components.

Management Domain
This component is used to boot the COSMOS system.

Data broker
This component is where Web services that share data self-register.

Data Adapter
This is a product-specific component that each product needs in order to share its data.

Data Manager
This component hosts one or more data adapters and handles communication with the COSMOS environment.

Client APIs and GUIs
These components are used to visualize or access data from multiple applications.

Starting Data broker and data manager

Because this involves a Java program you will need JRE_HOME or JAVA_HOME set in order to use the startup bat file.

Run the bat file from the <tomcat_install_dir>

1. Open command line window and change to the Tomcat install directory
2. Change to the bin directory, then type `startup.bat` and press Enter. By starting the web server, the Management Domain, Data Broker and the example MDR are started.

Starting the Management Domain and Data Broker

Because this involves a Java program you will need JRE_HOME or JAVA_HOME set in order to use the startup bat file.

Run the bat file from the <tomcat_install_dir>

1. Open command line window and change to the Tomcat install directory
2. Change to the bin directory, then type `startup.bat` and press Enter. By starting the web server, the Management Domain, Data Broker and the example MDR are started.

Registering the Data Broker and Management Domain

Registering the sample MDR with the Data Broker

Stopping the Management Domain and the Data Broker

Stopping the sample MDR and the Data Manager

Chapter 3. What is a federated configuration management database?

A federated configuration management databases is an industry standard for federating management data from multiple Configuration Management Databases (CMDBs). The federation provides an aggregate view of heterogeneous resources from multiple CMDBs. Each CMDB is a Management Data Repository (MDR) containing configuration item resources and their relationships.

Federated configuration management database components

The following list comprise federated configuration management database components.

- Client application
- Federating a CMDB
- Management Data Repository

The main federated configuration management database components are the MDR and the federating CMDB. The MDR is a Web service that exposes its CMDB configuration items and relationships according to the query interface defined in the federated configuration management database specification. The federated CMDB aggregates configuration items and relationships from multiple MDRs using the same query interface.

Federated configuration management database services

The client application typically uses the Query Service of the federated configuration management database or the MDR to access the configuration items and relationships. The federated configuration management database may also use the Query Service to access the MDRs.

The Registration Service is used by the MDRs to register their configuration items and relationships with the federated configuration management database.

Query service
This service is produced by the MDR and the federated configuration management database and is consumed by the client (which may be a federated configuration management database)

Registration service
This service is produced by the federated configuration management database and is consumed by the MDR

Federated configuration management database interactions

The MDRs self-register with the federated configuration management database using the Registration Service.

The Client application invokes the Query Service of a federated configuration management database, which aggregates the results of the Query Service of one or more MDRs.

```
Client Application <- Query Service -> CMDBf  <- Query Service  ->          MDR
                                         CMDBf  <- Registration Service  ->  MDR
```

COSMOS helps you build and deliver federated configuration management database components to provide customer solutions

COSMOS provides federated configuration management database specific features that enable you to quickly develop, test and deploy federated configuration management database components.

In addition, since the federated configuration management database and MDRs are implemented in COSMOS as Data Managers, you can integrate and leverage all the COSMOS components.

The COSMOS federated configuration management database features include the following.

Eclipse wizard for creating an MDR
The COSMOS tooling includes a wizard that generates an MDR. You can customize the generated MDR to utilize your specific management interfaces. Refer to the ″Constructing a Data Manager″ chapter of the COSMOS Development Guide for more information.

Federated configuration management database transformation services
The transformation services provides utilities for processing federated configuration management database requests and responses. These services can be used to assist with XML parsing and conversions. Refer to the ″Query and Registration Services″ chapter of the COSMOS Development Guide for more information.

Federated configuration management database registration services
The registration services assist in building MDRs which support federated configuration management database registration Refer to the ″Query and Registration Services″ chapter of the COSMOS Development Guide for more information.

Support for federated configuration management database service meta data
The federated configuration management database service meta data transformations assist in parsing and processing service meta data. COSMOS provides an extension point for the integration of service meta data into the MDR WSDL. Refer to the ″Query and Registration Services″ chapter of the COSMOS Development Guide for more information.

A Broker service that supports federated configuration management database components
The COSMOS Domain and Broker provide a registry that simplifies the task of locating federated configuration management database components (MDRs and federating CMDBs). Refer to the ″Core COSMOS Components″ chapter of the COSMOS User Guide for more information.

Web User Interface that supports federated configuration management databases
The COSMOS Web UI supports the federated configuration management database Query and Registration services. Federated configuration management database queries can be constructed visually using the query builder. The query can be executed and the responses are visually rendered in the UI. The Web UI also supports the federated configuration management database registration services. In addition to demos, the Web UI is useful for testing your federated configuration management database

components during development. Refer to the ″Web Console″ chapter of the COSMOS User Guide for more information.

Sample code for MDRs
COSMOS includes an example MDR and a registration example. Refer to Chapter 7, “Running the example,” on page 29.

Chapter 4. Understanding query capabilities

Supported query types

COSMOS does not restrict the types of queries that an adopter may send to a Data Manager or an MDR, as long as the query is well-formed and can be handled by the data source. COSMOS does offer *special handling* for queries that pertain to Federated CMDB data; this is described in Using federated CMDB queries. Later on, there may be additional special handling for other types of data as well, for example, WEF, Statistical, and so on.

- an interesting point
- another interesting point

How to send queries to COSMOS

Queries may be sent to a data manager through two primary means: programmatically with APIs and through the COSMOS Web user interface. Additionally, once a client obtains the addressing information for a Data Manager or an MDR from the COSMOS broker, the client can directly submit queries to the data manager or MDR without going through the COSMOS broker again.

How to access the result set from COSMOS

The result set is returned as is, by the data manager or MDR to the client. COSMOS does not alter the result set in any way. The client must have the ability to utilize the query result set. However, the COSMOS Web user interface does offer some visualization capabilities for federated CMDB data using basic web widgets. This is detailed in COSMOS Web component library.

Example queries for the sample MDR and data manager

See COSMOS Web component library.

Chapter 5. COSMOS Web component library

COSMOS defines a set of web-based components that provides access to the underlying COSMOS services. These web-based components can be configured and integrated with each other to create a very dynamic web based application.

Each Web component provides one of the following basic actions.

- **Navigate** - components that help the end user locate a data manager within the COSMOS system
- **Query** - components that are used to build and submit queries to a data manager
- **View** - components that visualizes information from a data manager
- **Manage** - components that manages a data manager
- **Utility** - components that provide utility functions

Table 2. COSMOS Web action components

Action Component	**COSMOS provided action component**	**Purpose**
Navigate	Data Manager Navigator	The Data Manager Navigator displays a list of data managers that are registered with a broker. The end user uses this user interface component to submit queries to a particular data manager.
Query	Basic federated configuration management database Query Dialog	This dialog provides a simple text input box to submit a federated configuration management database query document to an MDR.
	Federated configuration management database Query Builder Dialog	This web based component provides a dialog box to construct a federated configuration management database query document and submit the constructed document to an MDR. End users familiar with the federated configuration management database query document structure use this dialog box to construct a federated configuration management database query document instead of hand typing the document. The query builder processes the service meta data from an MDR to determine what kind of query capabilities the user is allowed to specify.

Table 2. COSMOS Web action components (continued)

Action Component	COSMOS provided action component	Purpose
View	XML Viewer	This view visualizes the structure of a XML document as a tree structure. The user has the ability to expand and collapse nodes to explore the XML structure. The names of the tree nodes corresponds to the element names.
	Frame Viewer	This view provides a frame to embed html content. This web component is used to embed html content from external systems. For example, reports generated from a reporting system can be embedded in this component. This allows an application to compose pages that are made up of content from various external sources.
	Properties Viewer	This view displays name value pairs within a table. The properties view displays the property names and values of a selected item within a table. The user can select an item using other web components. For example, the user can select a node within the Data Manager Navigator.
	Federated configuration management database Graph Response Viewer	This view visualizes the structure of a federated configuration management database response document. The CMDBf Graph Response Viewer contains two sections. The top sections is composed of a table that shows the list of nodes and edges. The first column of the table lists the set of sources nodes, the second column lists the edges while the third column lists the target. As a result, the user can determine how the nodes relate to each other by viewing this table. The user can select a template id from the drop down box to filter the nodes and edges shown in the table. The detail section shown at the bottom of the view displays the contents of a particular item or edge that is selected in the above table. The detail section shows the templateId, mdrId and localId of the selected item. The table shown below displays the list of records associated with the item or relationship. There is a green circle icon, with a white cross, that is shown to the left of each Record ID. When this icon is clicked, the following dialog box will pop up, that shows the xml representation of the record detail:
Manage	Configuration Item Registration Dialog	This dialog box provides the ability for users to register configuration items to a federating CMDB
	Configuration Item DeRegistration Dialog	This dialog box provides the ability for users to deregister configuration items to a federating CMDB.
Utility	Error Dialog	This dialog box displays error message to the user.
	Progress Dialog	This dialog box indicates that the user should wait for an action to complete.

Data Manager Navigator

This web component is composed of a tree widget that contains the list of registered data managers that are the top level nodes. Icons can be be associated with each node to visually distinguish between data managers.

These actions can be initiated on a node.

- Expanding a node
- Opening a pop up menu on a node
- Selecting a node

Expanding a node

Expanding a node will initiate a query to the data manager that the node represents. This query is configurable by the application. For example, consider a ″CBE″ data manager. This data manager contains a repository of log files that have been collected from various applications. Expanding the node of the ″CBE″ data manager will submit a query to get the list of log files that the this data manager is aware of. As a result the user will see a list children nodes under the ″CBE″ data manager node.

Note: Expanding a node may result in different queries for different data managers. These queries are configurable at the application level.

Opening a pop up up menu on a node

Right clicking a node will bring up a pop-up menu with a set of options. These options may vary from one data manager to the next. The types of options that are displayed in the pop-up menu are configured at the application level.

There are however some basic menu options that are available to data managers that are MDRs.

- Submit a Query

The user can select the Submit Query option to bring up a web component that will allow the user to construct a federated CMDB query and submit this query to the MDR. These web components are further explained under the [Query] section.

Selecting a node

Selecting a node may cause a section within the application console to refresh. The section that is refreshed within the application as a result of selecting a particular node is configured at the application level. For example, an application console may be made up of this navigator component and the [properties] component. When the user selects a node in the navigator component the properties component is refreshed and shows the list of properties of the selected node in a table. This is illustrated in the following screen shot.

Basic CMDBf query dialog

An application can configure the Data Manager Navigator component to show this dialog box by selecting the **Submit a Query** menu option. Typically, the user would right-click the MDR to which they want to submit a query. This opens a pop-up menu with an option to `Submit a Query`.

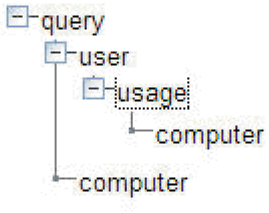
The user would then type in the CMDBf query document they would like to send. Once the user is done creating the query document they would click on the ″Submit″ button to send the document to the MDR.

Once the user submits the a node is added under the selected MDR that represents the query.

CMDBf query builder dialog

Application can configure the Data Manager Navigator component to show this dialog box by selecting the **Submit a Query** menu option. Typically, the user would right-click the MDR to which they want to submit a query. This opens a pop up menu with an option to `Submit a Query`.

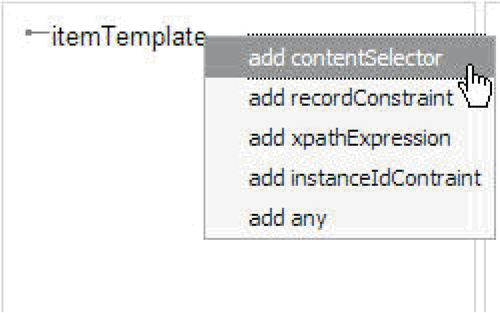
The query builder is composed of three panes. The top level pane manages the <itemTemplate> and <relationshipTemplate> nodes of a CMDBf query document. Users can add and remove itemTemplate and relationshipTemplate nodes within this pane. The top level nodes in the tree shows the list of itemTemplates defined in the query. <relationshipTemplate> nodes are defined as children nodes of an <itemTemplate>. Adding a <relationshipTemplate> underneath an <itemTemplate> node indicates that the parent node of the <relationshipTemplate> is the source template and the child node of the <relationshipTemplate> is the target template. Consider the following illustration.



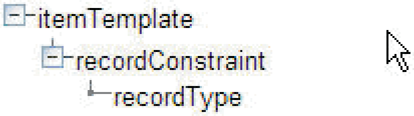
The above picture illustrates a query that has two <itemTemplate> nodes. The first <itemTemplate> has an id set to *user* while the second <itemTemplate> has an id set to *computer*. The pane further shows that a <relationshipTemplate> is defined where the source template node is the *user* and the target template node is the *computer*. Furthermore, the name of the relationshipTemplate is set to *usage*.

To add an <itemTemplate> the user can right click the **query** node in the top pane and select **Add Item**. A dialog box prompts the user to enter the id of the <itemTemplate>. To add a <relationshipTemplate> the user can right click an <itemTemplate> node and select **Add Relationship**. This prompts the user with a dialog box that allows the user to enter the id of the <relationshipTemplate> and the id of the target template. Note that a drop-down box is provided that lists existing itemTemplates. The user should note that if the MDR does not support queries that include <relationshipTemplate> nodes the user will not have the ability to add <relationshipTemplate> nodes.

The bottom left pane manages the structure of an <itemTemplate> or <relationshipTemplate> element that is selected in the top pane. The user can add query capabilities to these elements. The types of capabilities that the user can add is dictated by what is supported in the service meta data provided by the MDR. The user can right click the top level node to add a particular query capability as shown below:



The user can add or remove capabilities within this pane. As the user adds capabilities under either an <itemTemplate> or <relationshipTemplate> they are building the XML structure of the <itemTemplate> or <relationshipTemplate> element. Consider the following illustration.



The above tree structure produces the following XML structure under the <itemTemplate> element.

```
<s:itemTemplate id="user" suppressFromResult="false" >
      <s:recordConstraint >
         <s:recordType namespace="" localName="" >
         </s:recordType>
      </s:recordConstraint>
```

At any point the user can preview the CMDB query document by clicking the **Preview XML** button. A pop up box will display the XML document that represents the query.

Once the user is done creating the query document they would click on the ″Submit″ button to send the document to the MDR.

Once the user submits the a node is added under the selected MDR that represents the query.

Configuration item registration dialog

An application can configure the Data Manager Navigator component to show this dialog box by selecting the **Register Configuration Items...** menu option. Typically, the user would right-click a federating CMDB that they want to have register configuration items. This opens a pop up menu with an option to `Register Configuration Items...`.

The dialog box presents the user with a list of MDRs that they can select. When the user clicks the **Register** button all the configuration items associated with the selected MDRs will be registered with the federating CMDB. A response is returned from the federating CMDB that indicates if the configuration items were registered successfully. The response is displayed to the user as a list of configuration items. The display also indicates if the registration request for each configuration item was approved or declined.

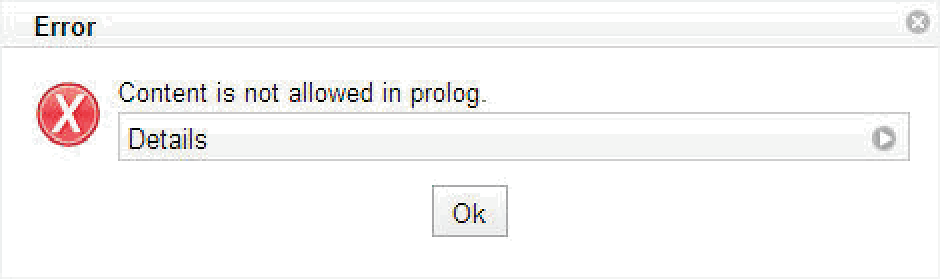
Configuration item deregistration dialog

It is possible to configure the Data Manager Navigator component so that this dialog box is shown when the user selects the **Deregister Configuration Items...** menu option. Typically, the user would right-click a federating CMDB that want to have deregister configuration items. This opens a pop up menu with an option to `Deregister Configuration Items...`.

The dialog box presents the user with a list of MDRs that they can select. When the user clicks the **Deregister** button all the configuration items associated with the selected MDRs will be deregistered with the federating CMDB. A response is returned from the federating CMDB that indicates if the configuration items were deregistered successfully. The response is displayed to the user as a list of configuration items. The display also indicates if the deregistration request for each configuration item was approved or declined.

Error dialog

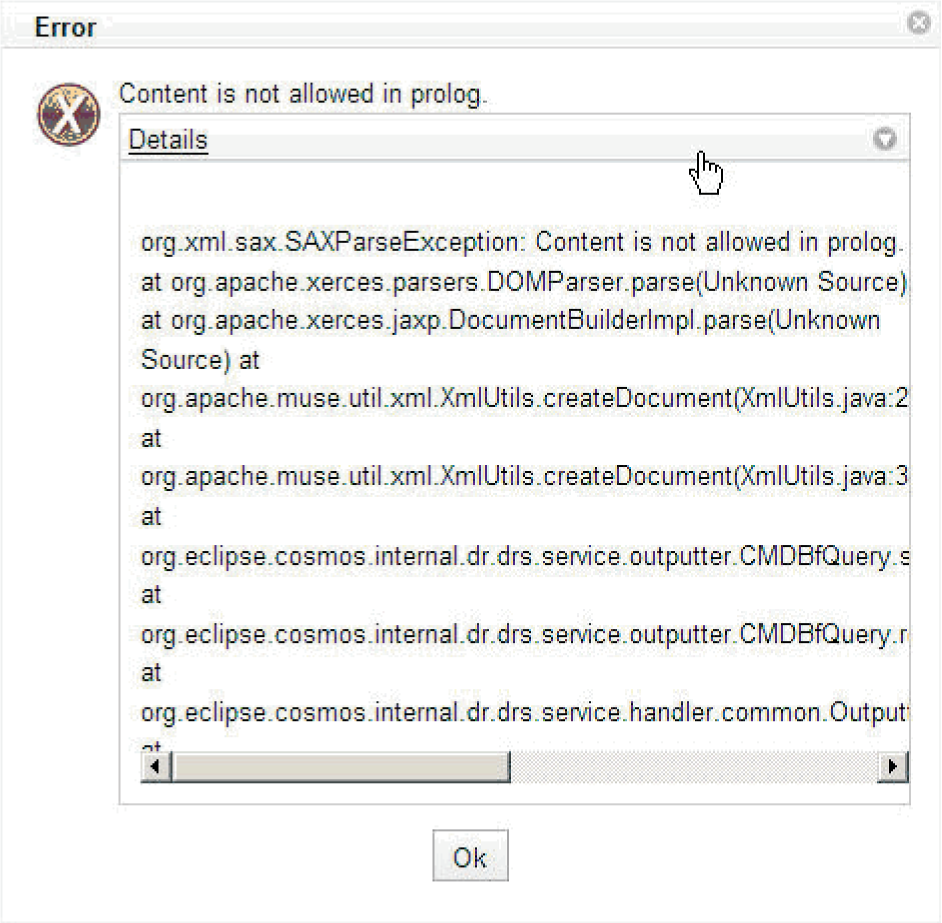
When an error occurs this dialog box is presented to the user.



Error

Content is not allowed in prolog.

The error dialog displays a message and a details message to the user. The user can expand or collapse the details message by clicking the 'details' bar as shown in this illustration.

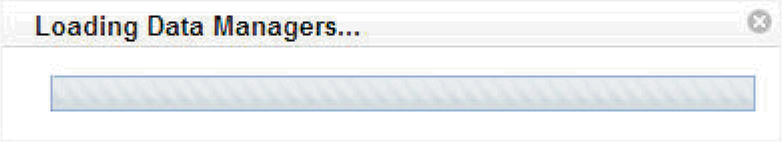


Content is not allowed in prolog.

```
org.xml.sax.SAXParseException: Content is not allowed in prolog.
at org.apache.xerces.parsers.DOMParser.parse(Unknown Source)
at org.apache.xerces.jaxp.DocumentBuilderImpl.parse(Unknown
Source) at
org.apache.muse.util.xml.XmlUtils.createDocument(XmlUtils.java:2
at
org.apache.muse.util.xml.XmlUtils.createDocument(XmlUtils.java:3
at
org.eclipse.cosmos.internal.dr.drs.service.outputter.CMDBfQuery.s
at
org.eclipse.cosmos.internal.dr.drs.service.outputter.CMDBfQuery.r
at
org.eclipse.cosmos.internal.dr.drs.service.handler.common.Output
```

Progress dialog

This dialog indicates that user must wait for an action to complete.



Loading Data Managers...

Chapter 6. Using the Service Modeling Language

The purpose of the SML tooling is to view and edit SML-IF documents, and to validate the content of SML and SML-IF documents. The SML-IF tooling plug-in also provides export and import wizards that are integrated into the Eclipse SDK Platform.

- Export operation creates an SML-IF document from the repository
- Import operation parses an SML-IF document into individual separate SML model documents and stores them in the repository (which can be a workspace folder).

The export operation packages and validates a set of documents representing an SML model to be interchanged into a single XML=based document in a standard way. The set of documents to be interchanged is called the SML-IF document's interchange set. The documents in the interchange set constitute the SML model being interchanged.

Note: During the import operation, there would normally be some loss of data when an SML-IF document is broken into its respective SML model units. For example, the aliases associated with a document or schematron rule binding are not preserved in SML model units. To preserve this information, a meta-data file is generated with every SML-IF document import.

The SML-IF validator helps to ensure an SML-IF document meets the following criteria.

- Conforms to XML 1.0 (Second Edition)
- Conforms to Schematron rules bound to documents
- Is valid under the SML-IF and SML schemas
- Conforms to the SML-IF and SML specifications.

The SML-IF specification defines the format for a single XML-based document that can be used to exchange SML models between other consumers of SML models.

The SML-IF tooling plug-in includes a multi-page editor that contains three tabs.

- **Overview –** The overview window displays general information about the SML-IF document.
- **Documents –** The documents window displays definitions, instances, logical representation, alias names and logical tree, and a source representation of the selected document section.
- **Source view –** Source view window shows the raw source of the SML-IF file, color coded with syntax highlighting.

Summary

SML is an important step to drive significant, cross-platform business modeling improvements for IT professionals and their business platforms. SML provides an industry-agreed foundation for model-based management. SML allows developers to build modeling information for applications, devices and services that can be used during all stages of applications or service lifecycle, including configuration, problem, change, and release management.

As a common modeling language, SML simplifies IT management in multi-vendor environments. This provides a way for information to be shared across diverse tools and applications. It reduces both the cost and complexity associated with delivering the levels of service and responsiveness businesses need from IT today.

The SML tooling provides an editor for an SML-IF document. The tool also provides the ability to import SML-IF document into a project and export SML model units to an SML-IF document. A SML-IF document packages the collection of SML documents to be interchanged as a single XML document. All exported SML-IF documents conform to the schema defined by the SML-IF specification.

Terms you should know

The following list contains some basic terms you should know to make using this documentation easier

SML
Service Modeling Language (SML) is an XML-based specification that is used to model complex IT resources, services and systems, including their structure, constraints, policies and best practices. An SML model defines a consistent way for computer networks, applications, servers and other IT resources are described or modeled to assist businesses to manage their services that are built on these. The SML specification leverages Schematron and unique XML Schema extensions to implement these models.

SML-IF
Service Modeling Language – Interchange Format (SML-IF) defines an implementation-neutral interchange format that preserves the content and interrelationships among documents.

Definition
A definition is one of two types of documents supported by SML-IF. A definition document is used to describe the structure and constraints of the model.

Instance
An instance is one of two types of documents supported by SML-IF. An instance document describes a model entity.

SML repository
In the SML tooling, this is a flat-file based directory structure. Each file in the structure contains an SML document. The Import operation creates SML document files based on the top level resources found in an input SML-IF document.

SML validation
Validating an SML model in accordance with the SML specification.

SML-IF validation
Validating an SML-IF document in accordance with the SML-Interchange Format specification.

Domain model
The root of an SML-IF document. Contains a set of definition and instance documents.

SML resources

Additional resources for SML is found at these locations:

- W3C Architecture domain SML Working Group

- W3C Architecture domain SML Charter
- W3C SML Version 1.0 submission
- W3C SML Interchange Format Version 1.0 submission
- Schematron
- XML Schema

Opening an SML-IF document in an editor

The SML tooling provides an editor that allows you to view and edit an SML-IF file. Use one of two ways to open an SML-IF document in this editor:
- Select **File > Open...** from the menu bar, and choose the file from the file system
- Double-click on an SML-IF file that exists in the Eclipse workspace

The SML-IF editor has these three main sections.
- Overview section
- Document section
- Source View section.

To view a specific section, select one of the tabs located at the bottom of the editor.

Overview section

Table 3. Fields on the Overview section of the SML-IF document editor

Field	**Required**	**Description**
Name	Yes	Name field is the namespace for the SML-IF document.
Version	No	Specifies the version level of the model.
Display name	No	Descriptive name of the model intended for display
Base URI	No	The base Uniform Resource Identifier (URI) provides a simple and extensible means for identifying a resource. For example, the URL http://www.eclipse.org/ is a URI that identifies a resource.
Description	No	This is a detailed description of the model

Validate the entire document function on the Overview section

To validate the SML-IF document, click **Validate the entire document**. This function validates the entire SML-IF document and reports any errors in the Problems view of your workbench. See Validating an SML-IF document for more information on validation.

Import model into repository function

To import the SML-IF document, click **Import model into repository**. This function imports the SML resources represented in the SML-IF document into the SML repository. See Importing an SML-IF document for more information on importing.

Documents section

The SML-IF editor Documents window displays an SML-IF document's definition and instance elements and their aliases, with elements organized in a logical tree.

You can view a tree display of definition and instance documents, and change the aliases of an individual document by selecting a resource and clicking the Add or Remove button. You can also view the subset of the SML-IF file that defines a document or instance element by selecting that element in the tree view, and that element's source will be displayed in the XML Representation field

Table 4. Fields on the Documents section of the SML-IF document editor

Field	Description
Documents	Two types of documents are supported by SML-IF. • Definition documents that describe the structure (schema) and constraints (Schematron rule) of the model • Instance documents that describe a model entity.
Logical Representation	Two types of logical representations are contained in an SML-IF document. • Alias is an alternative and usually easier-to-understand or more significant name for a defined data object. • Logical Tree is a logical representation of the relevant structure specified by the tags and attributes in the document.
XML Representation	XML source of the selected element

Source section

The Source section of the editor is displayed by clicking the Source tab. The Source section displays the text content of the SML-IF document with color syntax highlighting.

Validating an SML-IF document

To perform this task you must have a Navigator window open.

An SML model is a set of interrelated XML documents. Each SML model contains a special subset of documents that describe the schema and rules governing the structure and content of the model's documents.

SML model validation is the process of verifying that all documents in a model are valid with respect to this special subset. The key contribution of SML to validation lies in its handling of inter-document constraints, which are layered on top of the single-document validation of XML Schema and Schematron.

To validate an SML-IF document use the following procedure.

1. From the Navigator window, select a desired document to be validated.
2. Do one of the following to Validate the document
   a. Right-click the document and select **Validate SML-IF**
   b. Alternatively, open the file with the SML-IF editor, select the **Overview** tab and click **Validate entire document**. All errors or warnings information are reported in the problems view. If there is no warning, error or information to be displayed, then a Validation Passed message dialog appears to notify you that the document is valid and well formed.
3.

If you receive an error or warning, go to "Navigating and fixing a problem" on page 25.

Navigating and fixing a problem

If you have a validation problem, the errors are displayed in the Problems view. To view a validation error, perform the following steps.

1. Double-click an error or warning entry in the problems view to navigate to the Source section of the SML-IF editor and the appropriate line number that is in violation, if line number information is available for that entry. The line-number of the SML-IF document is marked with an error or warning icon
2. After resolving the error in the SML-IF document, save and re-validate the document.

Importing an SML-IF document into the repository

Import operation divides an SML-IF document into individual separate SML model documents and stores them in a desired target repository. Remember that the SML repository is directory structure of flat files, each containing an SML resource.

How to import a document

The following import procedure is based on the assumption that you already have an SML-IF document

1. Start the Import wizard using one of the following three methods
   - Select **File →Import**
   - Right-click within the Eclipse Navigator and click **Import**
   - click the **Import model into repository** option in the Overview section of the SML-IF editor
2. From the Import Select window perform these steps.
   a. Expand the **Service Modeling Language** category
   b. Select **Interchange Format**
   c. Click **Next**
3. From the Import-Interchange Format window preform these steps.
   a. Select the SML-IF document to import
   b. Select the Destination folder to store the SML documents
   c. Click **Finish**

After the SML documents are created, refresh the Eclipse workspace to display the files in the Navigator

Exporting SML resources from the repository

The export operation creates an SML-IF document from an SML repository. The export operation is integrated with Eclipse's export wizard under the Service Modeling Language category.

An SML-IF document can be formed by selecting Export... from the context menu of one or more SML resources that appears under a project.

How to export resources

The following export procedure assumes you already have an SML document. If you do not have an SML document, use the export wizard to create an SML-IF document

1. To start the Export wizard, perform these steps.
   a. Select the folders or model units that are to be included in the interchange document.
   b. Right-click on the items and select **Export**. The Export Select window is displayed.
2. From the Export Select window, perform these steps.
   a. Expand the **Service Modeling Language** category
   b. Click **Interchange Format**
   c. Click **Next**. The Export Interchange Format window is displayed.
3. From the Export Interchange Format window perform these steps.
   a. Select the resources to export
   b. Click **Next**. The Export Interchange Format window is displayed.
4. From the Export Interchange Format window perform these steps. Specify aliases, if desired, for document and SML-IF identity information.
   a. (Optional) Add an alias to a Detected Document. Click the Detected Document, click **Add**, enter the new aliases name in the pop-up window, click **OK**.
   b. Add identity information. Enter the name, display name, description, and base URI
   c. Click **Next**.
5. (Optional) Add rule bindings to bind a named Schematron rule to an SML instance.
   a. Click **Add**.
   b. Click the **Alias** dropdown arrow to select the alias to be bound for the selected document.
   c. Click the **Rule** dropdown arrow to bind the document to an existing Schematron rule.
   d. Click **Finish**.

After the SML-IF document is created, open the document in the SML-IF editor to display the document source.

This is an example that shows the contents of an exported SML-IF document.

```
<?xml version="1.0" encoding="UTF-8"?>
<smlif:model xmlns:smlif="http://www.w3.org/2008/03/sml-if"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.w3.org/2008/03/sml-if smlif.xsd">
 <smlif:identity xml:base="http://interop.serviceml.org">
  <smlif:name>University</smlif:name>
  <smlif:description> This represents the classes, instructors, and students
   at Cosmos State University.
  </smlif:description>
 </smlif:identity>
 <smlif:definitions>
  <smlif:document>
   <smlif:docinfo>
    <smlif:aliases>
     <smlif:alias>university50445.xsd</smlif:alias>
 </smlif:aliases>
</smlif:docinfo>
<smlif:data>
 <xs:schema targetNamespace="urn:university"
  elementFormDefault="qualified" xmlns:tns="urn:university"
```

Chapter 7. Running the example

Demo objective

This demonstration is provided to help you become familiar with the COSMOS interface, tools and usages.

Installation and Configuration

Before you can run the example, you must download, install, and configure the software.

- For instructions, see "Install software."
- For instructions, see "Configure software."

Install software

This section provides instructions for installing and running the demo program that is provided in the COSMOS i9 build.

1. Install Tomcat 5.5.25. The directory where the Tomcat server is installed will be referred to as the `<tomcat_install_dir>` in this document.
2. From the COSMOS download page, click on the link of the release candidate build for version 1.0. Download the zip file under the **COSMOS Demo** section.
3. Unzip the demo zip file to a directory. You will see a directory called `cosmos-demo`. This directory will be referred to as `<cosmos-demo>` in this document.
4. Copy all directories under `<cosmos-demo>\webapps` to `<tomcat_install_dir>\webapps`.

Next you must configure the software for your system environment. Go to "Configure software."

Configure software

These instructions assume you have installed the demo software as described in the "Install software" section.

1. At a Windows command prompt, change to the `<enter cosmos-demo>\bin>` directory.
2. Run **configDemo.bat** to update some configuration files according to your system environment. The configDemo program assumes you have done the following steps.
   - You have copied the contents of the `<cosmos-demo>\webapps` directory in Tomcat as described in step 4 of "Install software."
   - Your `<cosmos-demo>` directory and the Tomcat installation are on the same host.
3. The program prompts you for input. A successful configuration is indicated by file results similar to the sample below.

```
This program will configure some environment variables for the COSMOS demo to run.
We assume that the Tomcat server and the directory where you unzipped the demo program
are on the same computer.
Apache Tomcat install directory: D:\Program Files\apache-tomcat-5.5.25
Hostname of the Tomcat server (e.g. www.eclipse.org): mydomain.com
Tomcat port number: 8080
COSMOS Demo install directory: D:\cosmos\i9\cosmos-demo
Set management domain EPR for broker successfully.
Set management domain EPR for Example MDR successfully.
Set management domain EPR for SML MDR successfully.
Set management domain EPR for logging data manager successfully.
Set CBEData.dir for logging data manager successfully.
Set management domain EPR for stat data manager successfully.
Set derby.system.home for statistical data manager successfully.
Updated SML MDR config file successfully.
```

Now, you can use the example.

Running the program

This topic describes how to start COSMOS and takes you through common tasks to enable you to quickly become familiar with the capabilities of COSMOS.

Starting COSMOS components

You need to set JRE_HOME or JAVA_HOME before you can run the `startup.bat` file.

After you have configured the program by following instructions in the previous section, you can start the COSMOS components.

Start Tomcat by running `<tomcat_install_dir>/bin/startup.bat`. By starting the web server, the management domain, broker, data managers, reporting and user interface are started. The broker and data managers are registered with the management domain and the broker respectively, automatically on start up. This activity may take a few seconds and you can monitor it by viewing the Tomcat console.

After the COSMOS components have successfully started, you can use the COSMOS command line demo application and the COSMOS user interface to interact with the COSMOS components.

Brokers registered at the management domain query

1. Execute `<cosmos-demo>\bin\domainEPR.bat`. This program prompts you for your hostname and the port number of the Tomcat server, and prints out the Endpoint Reference (EPR) of the management domain; this reference is expressed in XML format. Copy the EPR of the management domain to a scratch editor window, such as Notepad, for later use.
2. If the COSMOS command line client has not started, execute `<cosmos-demo>\cosmos-client\cosmosClient.bat`.
3. Get a list of brokers at the management domain with the `domain getBrokers` command. The sample shown below is displayed.

```
COSMOS> domain getBrokers
Management Domain EPR (terminate by an empty line):
<?xml version="1.0" encoding="UTF-8"?>
<wsa:EndpointReference xmlns:wsa="http://www.w3.org/2005/08/addressing">
    <wsa:Address>http://localhost:8080/cosmos/services/domain/wsa:Address>
</wsa:EndpointReference>
```

```
EPR of brokers:
Broker 1:
<?xml version="1.0" encoding="UTF-8"?>
<wsa:EndpointReference xmlns:wsa="http://www.w3.org/2005/08/addressing">
    <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing"/>
    <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">
       http://localhost:8080/cosmos/services/broker/wsa:Address>
</wsa:EndpointReference>
```

Data Managers registered at the broker query

Before you use this query, process the “Brokers registered at the management domain query” on page 30 to obtain the required data for this query.

1. At the COSMOS prompt, type `broker getDataManagers`.
2. When prompted for the broker EPR, paste in the broker EPR. You see a list of EPRs of data managers registered at the broker.

```
COSMOS> broker getDataManagers
Broker EPR (terminate by an empty line):

<wsa:EndpointReference xmlns:wsa="http://www.w3.org/2005/08/addressing">
    <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing"/>
    <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">
      http://___:8080/cosmos/services/broker</wsa:Address>
</wsa:EndpointReference>

EPR of data mangers:
Data Manager 1:

<wsa:EndpointReference xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">
   http://____:8080/stat/services/org.eclipse.cosmos.dc.stat.datamgr.StatisticalDataManager
    </wsa:Address>
    <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing">
        <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing">
          StatDataManager</muse-wsa:ResourceId>
    </wsa:ReferenceParameters>
</wsa:EndpointReference>
Data Manager 2:

<wsa:EndpointReference xmlns:wsa="http://www.w3.org/2005/08/addressing">
    <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">
      http://____:8080/smlmdr/services/org.eclipse.cosmos.rm.sml.mdr.impl.SmlRepositoryMdr
    </wsa:Address>
    <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing">
     <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing">SML_REPO_ID
     </muse-wsa:ResourceId>
    </wsa:ReferenceParameters>
</wsa:EndpointReference>
Data Manager 3:

<wsa:EndpointReference xmlns:wsa="http://www.w3.org/2005/08/addressing">
    <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">
       http://____:8080/examplemdr/services/org.eclipse.cosmos.example.mdr.ExampleMdr
    </wsa:Address>
    <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing">Example
      </muse-wsa:ResourceId>
    </wsa:ReferenceParameters>
</wsa:EndpointReference>
Data Manager 4:

<wsa:EndpointReference xmlns:wsa="http://www.w3.org/2005/08/addressing">
 <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">
  http://____:8080/logging/services/org.eclipse.cosmos.dc.logging.datamgr.LoggingDataManager
 </wsa:Address>
 <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing">LoggingDataManager
  </muse-wsa:ResourceId>
 </wsa:ReferenceParameters>
</wsa:EndpointReference>
```

Using federated configuration management database queries

This topic explains federated configuration management database queries.

Using a command line client for federated configuration management database queries

You can execute federated configuration management database queries using the command line client.

At the COSMOS prompt, type **smlmdr query** followed by the path to the query file, to run a query against the SML MDR. If you are querying the Example MDR, the command is instead **examplemdr**.

The query files for the SML MDR are located in the <cosmos-demo>\cosmos-client\cmdbf-queries\SMLMDR directory.

The query files for the Example MDR are located in the <cosmos-demo>\cosmos-client\cmdbf-queries\ExampleMDR directory.

When prompted for the EPR for the SML MDR or the Example MDR, you can copy the EPR from the output of the `broker getDataManagers` command described in the "Data Managers registered at the broker query" on page 31 section.

Output samples

```
COSMOS>smlmdr query __\cosmos-demo\cosmos-client\cmdbf-queries\SMLMDR\testInstanceIdQuery.txt
EPR of SML Repository MDR (terminate by an empty line):
<wsa:EndpointReference xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <wsa:Address>
     http://____:8080/cosmos/services/org.eclipse.cosmos.rm.sml.mdr.impl.SmlRepositoryMdr
  </wsa:Address>
    <wsa:ReferenceParameters>
      <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing">SML_REPO_ID
      </muse-wsa:ResourceId>
    </wsa:ReferenceParameters>
</wsa:EndpointReference>

Query response:
<?xml version="1.0" encoding="UTF-8"?>
   <dyn:queryResponseType xmlns:dyn="http://www.eclipse.org/cosmos/mdrquery">
        <dyn:queryResponseType xmlns="http://cmdbf.org/schema/1-0-0/datamodel">
                <nodes templateId="cableTemplate">
........ Content omitted .........
                </nodes>
            </dyn:queryResponseType>
        </dyn:queryResponseType>
```

You can invoke queries on the Example MDR using the command line client.

```
COSMOS>examplemdr query __\cosmos-demo\cosmos-client\cmdbf-queries\ExampleMDR\all-students.txt
EPR of SML Repository MDR (terminate by an empty line):
<wsa:EndpointReference xmlns:wsa="http://www.w3.org/2005/08/addressing">
    <wsa:Address>
      http://_____:8080/examplemdr/services/org.eclipse.cosmos.example.mdr.ExampleMdr
    </wsa:Address>
    <wsa:ReferenceParameters>
     <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing">Example
     </muse-wsa:ResourceId>
    </wsa:ReferenceParameters>
</wsa:EndpointReference>

Query response:
<?xml version="1.0" encoding="UTF-8"?>
    <dyn:queryResponseType xmlns:dyn="http://www.eclipse.org/cosmos/mdrquery">
            <dyn:queryResponseType xmlns="http://cmdbf.org/schema/1-0-0/datamodel">
                <nodes templateId="items">
........ Content omitted .........
                </nodes>
            </dyn:queryResponseType>
        </dyn:queryResponseType>
```

Using the COSMOS Web User Interface for federated configuration management database queries

The section explains how to use the COSMOS Web User interface to use CMDBf queries.

The section explains how to use the COSMOS Web User interface to use federated configuration management database queries.

Register Configuration Items with a Federating CMDB

Querying the Federating CMDB

Collect Log Events from the Logging Data Manager

This topic describes collecting log events from the logging data manager.

Using a command line client for log collection

Put your short description here; used for first paragraph and abstract.

Using the COSMOS Web User Interface for log collection

Collect Web server performance data from the statistical data manager

Put your short description here; used for first paragraph and abstract.

Using a command line client Web server performance

This task describes how to use the command line to collect Web server performance infomation.

Chapter 8. COSMOS Web console

The COSMOS user interface is a web application, comprised of a set of COSMOS web components that allow users to manage, query and locate data managers. A user can launch the COSMOS UI by opening http://localhost:8080/COSMOSUI with their browser. This assumes that the COSMOS UI web application is deployed on the localhost at port 8080. The initial page of the COSMOS Web User Interface displays three main panes. The left pane shows a list of data managers that are registered with the broker. The top right pane shows a detail view that is initially blank, while the bottom right pane shows a properties view that is also initially blank.

Additional capabilities may be added to the COSMOS User Interface by providing new web components that can be integrated and configured for the COSMOS User Interface. The COSMOS Uset Interface is built on a framework that provides a list of predefined re-usable components, like query dialog boxes, CMDBf repsponse viewers, etc. The list of web components is outline under the COSMOS Web Component Library.

Web console tutorials

These tutorials are provided to help you become familiar with common tasks you can perform by using the Web Console.

- Navigate to a data manager
- Submit queries to a data manager
- Submitting a federated CMDB query to a data manager
- Generate a log report
- Generate a statistical report
- Register configuration items
- Deregister configuration items .

Error messages

If an error occurs in the COSMOS Web Console a dialog box is presented to the user. The dialog box contains the error message and a detailed message. The user can expand or collapse the detail message section to hide or show more information.

Appendix. Legal

Notices

Glossary

Glossary.

Index

Printed in USA