Using Web Services and Northbound API

Overview

CiscoWorks Common Services (CWCS) is a collection of subsystems, execution environments, engines, and shared code libraries, forming a software platform that provides services to web-based network management applications. For more information, see SDK Developer’s Guide for CiscoWorks Common Services 3.2. The following section describe Common Services Northbound API and Web Services:

- What’s New
- Understanding Common Services Northbound API
- Northbound Interface Framework
- Northbound Interface Services
- Using Web Services

What’s New

This following are the new enhancements in this release:

- Using Apache Axis SOAP Server version 1.4: The SOAP Server per web application used in this release is "Apache Axis" SOAP Server version 1.4
- CiscoWorks Northbound Interface (CW-NBI) Framework and Services are based on CiscoWorks Common Services 3.2 version. Since the CW-NBI framework and services have been integrated in CS3.2, CW-NBI is not installed, with basis as CS 3.2, as a separate product.
- DCR credentials includes SNMPv3 AuthPriv support. For more information, see Understanding DCR.

Understanding Common Services Northbound API

CiscoWorks suite of applications provides a set of Web Services. Third Party applications that want to integrate with CiscoWorks applications can use these services to provide end-to-end solutions. As the first phase, only CiscoWorks Common Services and CiscoWorks Resource Manager Essentials (RME) will provide web services.

SDK Developer’s Guide for CiscoWorks Common Services 3.2 Northbound API 1.1 describes the Web Services provided by Common Services.
The CW-NBI Framework and Services are based on CiscoWorks Common Services 3.2 version. The Web Services uses some of the existing components like Tomcat, Apache and require changes in CSTM (Common Services Transport Mechanism). The list of features and changes expected in the existing components are as follows:

- Cisco Works NBAPI Framework which provides a base platform for developing web services over Common Services 3.2.
- Security Service for login, logout, authorization and user management activities.
- Device Service to add, delete, modify and query the devices into / from DCR.
- Job Service to query / cancel / delete the Jobs. Application need to provide NB APIs for creating / editing the jobs.
- Grouping Services to create / modify / delete / query groups.
- Common Service Grouping services for getting the OGS class definitions.
- Common Service Information Service.
- New Version of CSTM with Axis 1.4.

The NBI Framework uses Tomcat and Apache Web Server internally for receiving requests. The framework will do AAA checks before passing the SOAP request to the actual services.

**Northbound Interface Framework**

The Web Services uses SOAP/XML with HTTP (only through SSL) as the transport protocol. The North Bound Interface Framework provides the basic infrastructure for communicating using SOAP over HTTP. This requires changes to be made in applications (CMF SRC component) and changes that are common for one system (CMF R component). CMF SRC parts will be packaged by individual applications and the CMF R part will be shipped by Common Services Web Services.

The major functionalities for the SRC section are as follows:

- Provide a SOAP Server which can communicate using SOAP. There will be one SOAP Server per web application. A SOAP Server can host many web services. The SOAP Server Version used is "Apache Axis" SOAP Server version 1.4
- Access all services through HTTPS only.
- SOAP Server uses the existing Apache Web Server and Tomcat Servlet Engine version 4.1.29 which is available in CS 3.2.
- Authentication check for all incoming SOAP requests. NBI Framework performs authentication checks for all NBI's except for some API's like login() and initiateSession() which creates the valid sessions.
- Authorization check for all incoming NBAPI calls. Only the task based authorization is performed here. Device based authorization is done by the individual service (method) itself.
- Optional accounting for all incoming North Bound calls. Accounting will not be enabled by default. Services can enable it based on their requirement.
- All the AAA checks are designed to be customized by individual applications.
- All the CS Web Services do not perform Licensing check. Application services can do Licencing Check if required.
- Framework provides utility methods for accessing the Logged in User, Session-Id, Remote Host Name and the Remote Application Name.
Northbound Interface Services

Northbound Interface Services consists of a set of Web Services for the remote application to integrate with CiscoWorks Common Services. The services are described in detail in the following sections:

- Chapter 3, “Using Security Service”
- Chapter 4, “Using Object Grouping Services”
- Chapter 5, “Using Job Service”
- Chapter 6, “Using DCR Service”
- Chapter 7, “Using Event Service”
- Chapter 8, “Using Information Service”

See relevant section for more information about a particular service.
Using Web Services

A web service is a collection of protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks such as the Internet in a manner similar to inter-process communication on a single computer.

One of the most popular protocol is SOAP (Simple Object Access Protocol) over HTTP. Clients access Web service through its interfaces and bindings, which are defined using XML artifacts, such as a Web Services Definition Language (WSDL), exposing management capabilities to external applications. This is also known as Northbound Interfaces (NBI).

Web services-based applications can perform a range of functions from simple requests for information to complex business processes and interactions. Browser-based applications are concerned with the representation of data to end users. Whereas, Web services allow clients programmatically use the Web to obtain information and access these service components and their functionality. Applications can also incorporate Web service functionality for their own use.

The main advantages and the reasons to use of Web Services are:

- Web services promote interoperability across different platforms, systems, and languages.
- Web services also reduces operational costs by enabling organizations to extend and reuse their existing system functionality.

Service-oriented Architecture

A service-oriented architecture is a collection of services that communicate with each other. The services are self-contained and do not depend on the context or state of the other service. They work within a distributed systems architecture.

A Web service enables a service-oriented architecture. A service-oriented architecture is an architectural style that promotes software reusability by creating reusable services.

A service-oriented architecture consists of:

- A service that implements the business logic and exposes this business logic through well-defined interfaces.
- A registry where the service publishes its interfaces to enable clients to discover the service.
- Clients (including clients that may be services themselves) who discover the service using the registries and access the service directly through the exposed interfaces.

An important advantage of a service-oriented architecture is that it enables development of loosely-coupled applications that can be distributed and are accessible across a network.

To enable this architecture, you need the following:

- A mechanism that enables clients to access a service and registry.
- A mechanism to enable different services to register their existence with a registry and a way for clients to look up the registry of available services. Web services are based on an architecture in which the service can be located over the network and its location is transparent. This means that clients may dynamically discover a particular service they wish to use.
- A mechanism for services to expose well-defined interfaces and a way for clients to access those interfaces.
Benefits of Web Services

Web services use open standards and protocols. Protocols and data formats are text-based and is easy to understand.

By utilizing HTTP, web services can work through many common firewall security measures without requiring changes to the firewall filtering rules.

Web services allow the reuse of services and components within an infrastructure.

The major benefits of Web Services are:

- Interoperability in a Heterogeneous Environment
- Business Services Through the Web
- Integration With Existing Systems

Interoperability in a Heterogeneous Environment

The key benefit of the Web service model is that it permits different distributed services to run on a variety of software platforms and architectures, and allows them to be written in different programming languages.

The greatest strength of Web services is their ability to enable interoperability in a heterogeneous environment. As long as the various systems are enabled for Web services, they can use the services to easily interoperate with each other.

Business Services Through the Web

An enterprise can use Web services to leverage the advantages of the World Wide Web for its operations. For example, an enterprise might make its product catalog and inventory available to its vendors through a Web service to achieve better supply chain management.

Integration With Existing Systems

Web Services are independent of platforms and programming languages. So they can be easily integrated with existing systems.

Most enterprises have an enormous amount of data stored in existing enterprise information systems, and the cost to replace these systems is such that discarding these legacy systems may not be an option.

Web services let enterprise application developers reuse and even commoditize these existing information assets. Web services provide developers with standard ways to access middle-tier and back-end services, such as database management systems and transaction monitors, and integrate them with other applications.

In addition, because these services are provided consistently, developers do not need to learn new programming models or styles as integration needs expand.

Reference links

Chapter 1      Using Web Services and Northbound API

Using Web Services