IBM SPSS Collaboration and Deployment Services
8.3

Introduction to Web Services

Note

Before using this information and the product it supports, read the information in "Notices" on page 11.

Product Information

This edition applies to version 8, release 3, modification 0 of IBM® SPSS® Collaboration and Deployment Services and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Chapter 1. Introduction to web services

What are web services?

At a high level, a web service is a set of functionality distributed across a network (LAN or the Internet) using a common communication protocol. The web service serves as an intermediary between an application and its clients, providing both a standardized information structure and a standardized communication protocol for interaction between the two.

Where other methods of distributed application architecture rely on a single programming language being used on both the application and its clients, a web service allows the use of loosely coupled services between non-homogenous platforms and languages. This provides a non-architecture-specific approach allowing, for example, Java services to communicate with C# clients, or vice versa.

Advantages to implementing application functionality as web services include the following:

- Software written in different languages (Java or C#) running on different platforms (UNIX or Windows) can exchange services and data
- Application functionality can be accessed by a variety of clients. For example, both a thin-client interface and a rich-client interface can take advantage of the web service operations.
- Updates to the service are immediately available to all service clients

Web service system architecture

Web services are deployed and made publicly available using an application server, such as WebSphere® or JBoss Application Server. The published web services are hosted by this application server to handle application requests, access permissions, and process load. A high-level architecture of how web services are implemented is displayed in the following diagram.

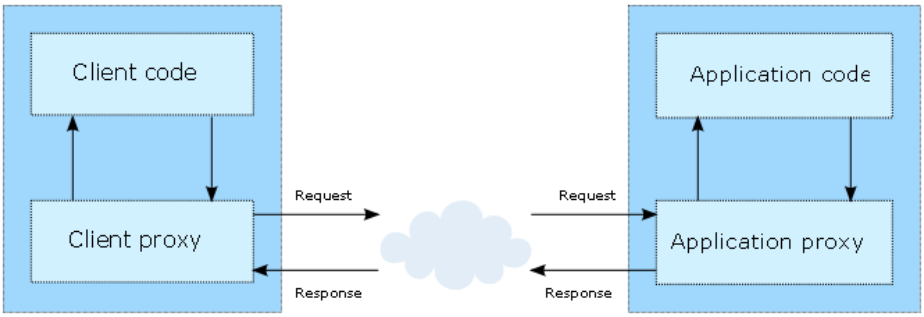

Figure 1. Web service architecture

The client code supplies input to an operation offered by a proxy class. The proxy class generates a request containing a standardized representation of the input and sends it across the network to the application. A proxy class on the server receives the request and unmarshals the contents into objects for processing by the application. Upon completing the operation, the application supplies a proxy with the output. The proxy creates a standardized representation of that output and sends the response back to the client. The client proxy unmarshals the response into native objects for subsequent processing by the client code.

Standardizing the format of the information passing between the client and the application allows a client written in one programming language to communicate with an application written in another. The proxy classes, which are automatically generated from a web service description by a variety of toolkits, handle the translation between native programming objects and the standardized representation. See the topic "Proxies" on page 5 for more information.

Web service protocol stack

A web service implementation depends on technologies often organized in a layered stack. The implementation itself defines a standard protocol for each technology layer, with each layer depending on the layers appearing below it in the stack.

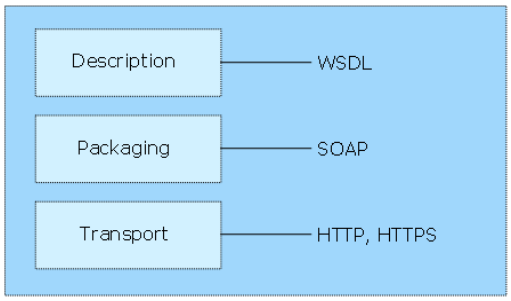

Figure 2. Web service protocol stack

Beginning at the bottom of the stack, the Transport layer defines the technology standards for communication, allowing information to move across the network. HTTP or HTTPS are often used as the standard for the transport layer.

The Packaging layer rests on top of Transport and defines the standard for structuring information for transport across the network. The SOAP format is commonly used, which offers an XML structure for packaging the data. See the topic "Simple Object Access Protocol" on page 2 for more information.

The topmost layer is Description and identifies the standards used by the layers below it in the stack, as well as providing the definition of the interface available for client use. The most common means of conveying this information is through the use of a WSDL file. See the topic "Web Service Description Language" on page 3 for more information.

Simple Object Access Protocol

The Simple Object Access Protocol (SOAP) is a way to pass information between applications in an XML format.

SOAP messages are transmitted from the sending application to the receiving application, typically over an HTTP session. The actual SOAP message is made up of the Envelope element, which contains a Body element and an optional Header element.

- **Envelope.** This mandatory element is the root of the SOAP message, identifying the transmitted XML as being a SOAP packet. An envelope contains a body section and an optional header section.
- **Header.** This optional element provides an extension mechanism indicating processing information for the message. For example, if the operation using the message requires security credentials, those credentials should be part of the envelope header.
- **Body.** This element contains the message payload, the raw data being transmitted between the sending and receiving applications. The body itself may consist of multiple child elements, with an XML schema typically defining the structure of this data.

A SOAP packet and the corresponding XML is structured in the following way:

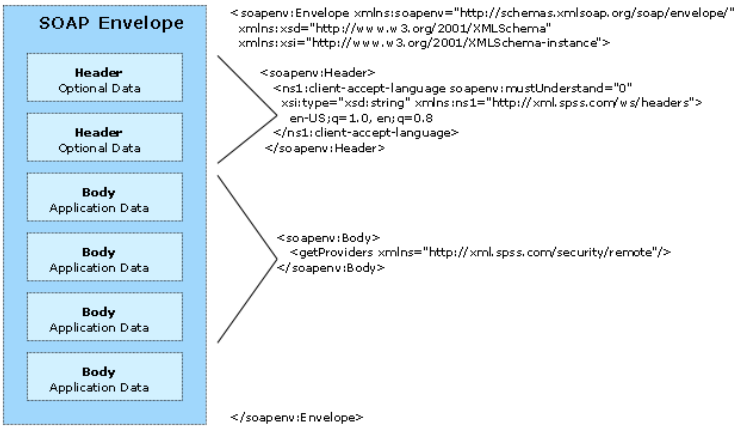

Figure 3. An example SOAP packet

Web Service Description Language

A Web Service Description Language (WSDL) file provides an XML-based map of what functionality the published web service allows, separating the implementation in the service from the interface. The WSDL defines the following:

- The access location of the web service
- Operations the web service exposes
- Parameters the exposed operations accept
- Any request or response messages associated with the operations

The WSDL provides the information necessary to generate a client-side proxy in the target programming language.

In accordance with the WSDL specification adopted by the World Wide Web Consortium, information in the WSDL is organized into the following sections:

- **Types.** Content definitions for web service operation input and output. See the topic "Types" on page 3 for more information.
- **Messages.** Input and output definitions for the web service operations. See the topic "Messages" on page 4 for more information.
- **PortTypes.** Groups of operations offered by the web service. See the topic "Port types" on page 4 for more information.
- **Bindings.** Protocols and formats for the web service operations. See the topic "Bindings" on page 4 for more information.
- **Services.** Endpoints at which the web service functionality can be accessed. See the topic "Services" on page 5 for more information.

Types

The types element of a WSDL file contains the data type definitions employed by messages processed by the web service. These definitions use XML to organize the information relevant to the type element being defined. Consider the following example type definitions:

```
<wsdl:types>
  <schema targetNamespace="http://xml.spss.com/security/remote"
    xmlns="http://www.w3.org/2001/XMLSchema">
    <element name="getProviders">
      <complexType />
    </element>
    <element name="getProvidersResponse">
      <complexType>
        <sequence>
          <element name="providerInfo[unbounded]" type="tns1:providerInfo" />
        </sequence>
      </complexType>
    </element>
```

```
  </schema>
</wsdl:types>
```

This section defines two elements, *getProviders* and *getProvidersResponse*. The former is an empty element. The latter contains a sequence of *providerInfo* child elements. These children are all of the *providerInfo* type, which is defined elsewhere.

In practice, the WSDL file typically references type element definitions found in an external XML schema. For instance, the following definition uses `security-remote.xsd` to define type elements.

```
<wsdl:types>
  <xs:schema>
    <xs:import namespace="http://xml.spss.com/security/remote"
      schemaLocation="security-remote.xsd"/>
  </xs:schema>
</wsdl:types>
```

Messages

The message elements of a WSDL file defines the input or output parameters for operations available in the web service. Each message can consist of one or more parts, with the parts similar to the parameters of a function call in a traditional programming language. Consider the following two example message definitions:

```
<wsdl:message name="getProvidersResponse">
  <wsdl:part element="tns2:getProvidersResponse" name="parameters" />
</wsdl:message>
<wsdl:message name="getProvidersRequest">
  <wsdl:part element="tns2:getProviders" name="parameters" />
</wsdl:message>
```

The *getProvidersResponse* message contains a single part, corresponding to the *getProvidersResponse* element defined in the types section of the WSDL file. Similarly, the *getProvidersRequest* message also contains a single part, as defined by the *getProviders* element in the types section. See the topic "Types" on page 3 for more information.

Port types

The portType element of a WSDL file defines the actual interface to the web service. A port type is simply a group of related operations and is comparable to a function library, module, or class in a traditional programming language. The definition specifies the parameters for the operations, as well as any values returned. The parameters and return values correspond to messages defined elsewhere in the WSDL file. Consider the following example port type definition:

```
<wsdl:portType name="ProviderInformation">
  <wsdl:operation name="getProviders">
    <wsdl:input message="impl:getProvidersRequest" name="getProvidersRequest" />
    <wsdl:output message="impl:getProvidersResponse" name="getProvidersResponse" />
  </wsdl:operation>
</wsdl:portType>
```

The *ProviderInformation* port type consists of a single operation, *getProviders*. Input to this operation corresponds to the *getProvidersRequest* message. The operation returns information in the structure defined by the *getProvidersResponse* message. See the topic "Messages" on page 4 for more information.

Bindings

The binding element of a WSDL file binds the interface defined by the port type to transport and messaging protocols. Consider the following example binding definition:

```
<wsdl:binding name="ProviderInformationSoapBinding" type="impl:ProviderInformation">
  <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
  <wsdl:operation name="getProviders">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="getProvidersRequest">
      <wsdlsoap:body namespace="http://xml.spss.com/security/remote" use="literal" />
    </wsdl:input>
    <wsdl:output name="getProvidersResponse">
      <wsdlsoap:body namespace="http://xml.spss.com/security" use="literal" />
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
```

In this case, the transport attribute of the `wsdlsoap:binding` element defines HTTP as the transport protocol. The `getProviders` operation in the interface is bound to the SOAP messaging protocol.

Services

The service element of a WSDL file identifies the network location at which the service interface can be accessed. Consider the following example service definition:

```
<wsdl:service name="ProviderInformationService">
  <wsdl:port binding="impl:ProviderInformationSoapBinding" name="ProviderInformation">
    <wsdlsoap:address location="http://pes_server:8080/security-ws/services/ProviderInformation" />
  </wsdl:port>
</wsdl:service>
```

In this example, the operations comprising the *ProviderInformation* port type can be accessed at:

`http://pes_server:8080/security-ws/services/ProviderInformation`

Proxies

Proxies serve as bridges between the client and the web service. A client-side proxy marshals the input objects into a standardized representation which is sent to the web service. A server-side proxy unmarshals the information into input objects for the service operations. The results of the operation are marshalled into standard representations and returned to the client. The client proxy unmarshals the response information into objects for any additional processing by the client.

Creating a proxy is the first step when developing a web service client; the proxy is the translation-unit between your application and the web service the application is using. Fortunately, many development environments include tools for automatically generating the client proxy from the web service WSDL file, allowing the client developer to focus on the client application code instead of transport and packaging protocols.

The proxy classes generated from a WSDL file depend on the tool used. For Java, the `wsdl2java` tool, which is part of the Apache Axis project, can be used. This tool produces a Java class for each type in the WSDL. Each port type results in a Java interface. A binding creates a stub class, and a WSDL service yields a service interface with a locator implementation. These generated classes and interfaces can be called directly from a client application written in Java to access the web service functionality.

An alternative Java proxy tool is `wsimport`, which is part of JAX-WS. The general structure of the generated classes is similar to that created by the Axis tool, but there are some differences. For example, instead of using arrays for input fields and returned items, the code generated from the `wsimport` tool uses `List` collections. In addition, if an input type matches an output type for a method, the `wsimport` tool uses a `Holder` class for the parameter.

In contrast, on the .NET platform, the `wsdl.exe` tool is often used to generate a web service proxy. This tool creates a single source file in a specified language containing the proxy class. This class includes both synchronous and asynchronous methods for each operation defined in the WSDL. For example, the web service operation `getProviders` results in the methods `getProviders`, `getProvidersBegin`, and `getProvidersEnd`. The latter two can be used for asynchronous processing.

A variety of other tools exist for other programming languages. For details, consult the documentation for those tools. In each case, the tool creates native programming constructs that permit leveraging a web service regardless of the service implementation language.

Chapter 2. IBM SPSS Collaboration and Deployment Services web services

In IBM SPSS Collaboration and Deployment Services, the IBM SPSS Collaboration and Deployment Services Repository serves as a centralized location for storing analytical assets, such as models and data. Client applications access those assets using web services available on the repository server. The client sends a functional request as a SOAP message to a specific service, and receives a SOAP message response from the service in return.

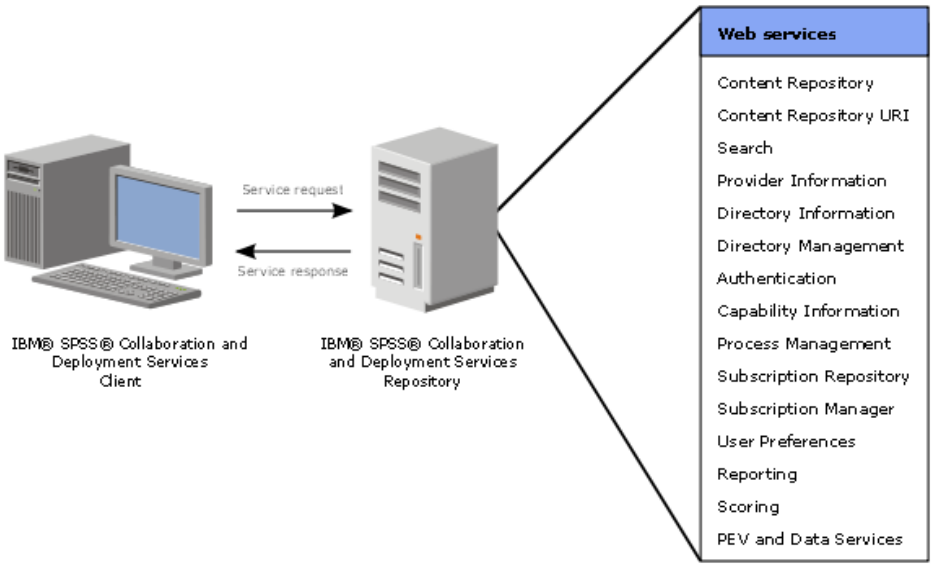

Figure 4. IBM SPSS Collaboration and Deployment Services web services

IBM SPSS Collaboration and Deployment Services includes web services for item storage and retrieval, search, security, scheduled asset execution, and notification.

Content Repository Service

The IBM SPSS Collaboration and Deployment Services Repository provides the storage facilities for IBM SPSS Collaboration and Deployment Services. The repository stores objects in a hierarchical system similar to the folder/file structure used in operating systems. The objects themselves may exist in several different versions to accommodate and track changes to the object over time.

The Content Repository Service provides remote access to the repository for general storage and retrieval of content and metadata. For example, a new object can be stored in the repository with particular name and keyword values. To retrieve the object, a query can be defined to search for objects having the specified name and keywords. Objects returned by the query can be retrieved for subsequent use. Retrieval of the object also returns the metadata associated with it.

Content Repository URI Service

The Content Repository URI Service provides remote access to the repository for general storage and retrieval of content and meta-data. For example, a new object can be stored in the repository with particular name and keyword values. Retrieval of the object also returns the meta-data associated with it. The Content Repository URI Service offers functionality similar to that found in the Content Repository Service. The primary difference is the use of **uniform resource identifiers** to reference objects stored in the repository instead of identification specifiers.

Search Service

The Search Service provides a query mechanism for locating content in the repository that meets specified criteria. The query may be a global search for a specified string or a more structured search for information in specific fields. The information contained in the search result set can be customized to be as broad or focused as desired. In addition, large result sets can be returned as individual pages containing a specified number of hits to optimize client performance.

Provider Information Service

A user's login/password combination must be verified before access to IBM SPSS Collaboration and Deployment Services can be granted. The Provider Information Service provides information about the enabled security providers against which the user credentials can be authenticated. The data returned by the service is typically used to allow users to select the provider to use for authentication of their login information.

Directory Information Service

The Directory Information Service returns information about principals available to IBM SPSS Collaboration and Deployment Services.

Principals fall into one of three categories:

- A **user** is an individual who needs access to the system.
- A **group** is a set of users who need access to the system.
- A **role** is a set of one or more privileges, or actions. Roles are assigned to users or groups to manage system security.

Directory Management Service

The Directory Management Service allows control over user and group access to IBM SPSS Collaboration and Deployment Services. Specifically, the service offers the ability to perform the following tasks:

- Configure security providers for user authentication
- Create users and groups for the system, and edit their properties
- Assign users and groups to security roles that control access to system functionality

Authentication Service

The Authentication Service provides methods for users of client applications to connect to and disconnect from the IBM SPSS Collaboration and Deployment Services server. When a user attempts to log in to the system, the service provides access if the user credentials are valid. For a valid user, the service reports information about the user, such as actions the user can perform, as controlled by the authorization mechanism. In addition, the Authentication Service offers validated users the ability to modify their passwords.

Capability Information Service

The Capability Information Service allows a client to obtain detailed information about the IBM SPSS Collaboration and Deployment Services server. The information returned in the capabilities message includes the following:

- User details

- The service version and instance
- Actions the user can perform, as controlled by the authorization mechanism
- Web services available on the server
- Settings for the web services that can be configured by the user

The Capability Information Service is primarily designed to allow clients to collect a variety of information when first connecting to a IBM SPSS Collaboration and Deployment Services server. The capabilities information has ramifications for how a particular user interacts with the server. Consequently, a request for capabilities data should be included as part of the action of logging in to the server.

Process Management Service

The Process Management Service allows a client to manage the artifacts associated with running jobs. The client can submit an existing job for processing, and retrieve the execution meta-data and results. In addition, schedules can be created and assigned to jobs, allowing execution at a future date or on a recurring basis.

Subscription Repository Service

The Subscription Repository Service provides a general facility for sending event notifications to subscribers. Components supplying events that can trigger notifications use the service to register the events and their properties in IBM SPSS Collaboration and Deployment Services. Clients can query the IBM SPSS Collaboration and Deployment Services Repository to discover the properties that they can expect to be contained in events of a certain type, and can write well-formed subscription expressions to match events in which they are interested. Subscribers subscribe to the events of interest, having the notification messages sent to any defined destinations.

Subscription Manager Service

The Subscription Manager Service allows a client to manage notification plug-ins, which augment the standard services with additional functionality. For instance, plug-ins can generate e-mail distribution lists from database queries. The service also includes operations for message template management.

User Preferences Service

The User Preferences Service allows users of a client application to store and retrieve individual values for preference items defined in the system, permitting a customized experience for each user. For example, a user can specify his or her e-mail address and have it persist across sessions. In addition, the service includes administrative functionality for managing preference items, such as identifying which users have specified preference values.

Coordinator of Processes Service

The Coordinator of Processes Service provides remote interaction with the COP server. Servers use the web service to store configuration files and status information in the IBM SPSS Collaboration and Deployment Services Repository. Clients can retrieve a list of available servers and server clusters for subsequent connections.

Reporting Service

The Reporting Service allows a client to submit a predefined report for processing. Using information contained within the report, the client can validate input sources and create prompts for user input to

direct report processing. Report output is available in a variety of formats for optimal display in any type of client.

Scoring Service

The Scoring Service allows client applications to employ real-time scores derived form predictive models developed in IBM SPSS Modeler, IBM SPSS Statistics, or third party tools. The service fetches a specified model, loads it, invokes the correct scoring implementation, and returns the result to the client. Supported models include regression (linear and logistic), decision trees, decision lists, neural networks, and naïve Bayes defined in IBM SPSS Modeler streams or in PMML from IBM SPSS Statistics.

Scoring can be either synchronous or asynchronous, depending on whether the client needs to wait for a score before proceeding or not. The service can load multiple models simultaneously for scoring and can be virtualized across multiple servers in a cluster configuration to handle large processing loads. The service logs all scoring activity for regulatory audit purposes. Configuring models for scoring and monitoring the service performance can be done using the IBM SPSS Deployment Manager.

Data Services Service

The Data Services Service provides functionality used when working with the data sources defined in the IBM SPSS Collaboration and Deployment Services Repository for analytic and scoring tasks. In general, the service provides the ability to perform the following tasks:

- Retrieve metadata about the tables available in data sources
- Retrieve information about table columns and links

The Data Services Service is often used in conjunction with the Scoring Service. Use the Data Services Service to access information about the data used for a particular scoring configuration.

Single sign-on services

Single sign-on functionality for IBM SPSS Collaboration and Deployment Services clients is enabled by the following web services:

- **SSO Authentication Service** . Enables single sign-on access.
- **SSO Directory Management Service** . Enables management and configuration of IBM SPSS Collaboration and Deployment Services single sign-on.

The SSO Authentication Service provides methods for users of client applications to connect to a single sign-on-enabled IBM SPSS Collaboration and Deployment Services server by supplying the client single sign-on provider information and distributing Kerberos tokens.

The SSO Directory Management Service allows control over single sign-on configuration and user and group access to IBM SPSS Collaboration and Deployment Services. Specifically, the service offers the ability to perform the following tasks:

- Configure the single sign-on provider
- Create users and groups for the system, and edit their properties
- Assign users and groups to security roles that control access to system functionality

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBMproducts. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Privacy policy considerations

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, See IBM’s Privacy Policy at http://www.ibm.com/privacy and IBM’s Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled “Cookies, Web Beacons and Other Technologies” and the “IBM Software Products and Software-as-a-Service Privacy Statement” at http://www.ibm.com/software/info/product-privacy.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be

trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other product and service names might be trademarks of IBM or other companies.

Glossary

Index