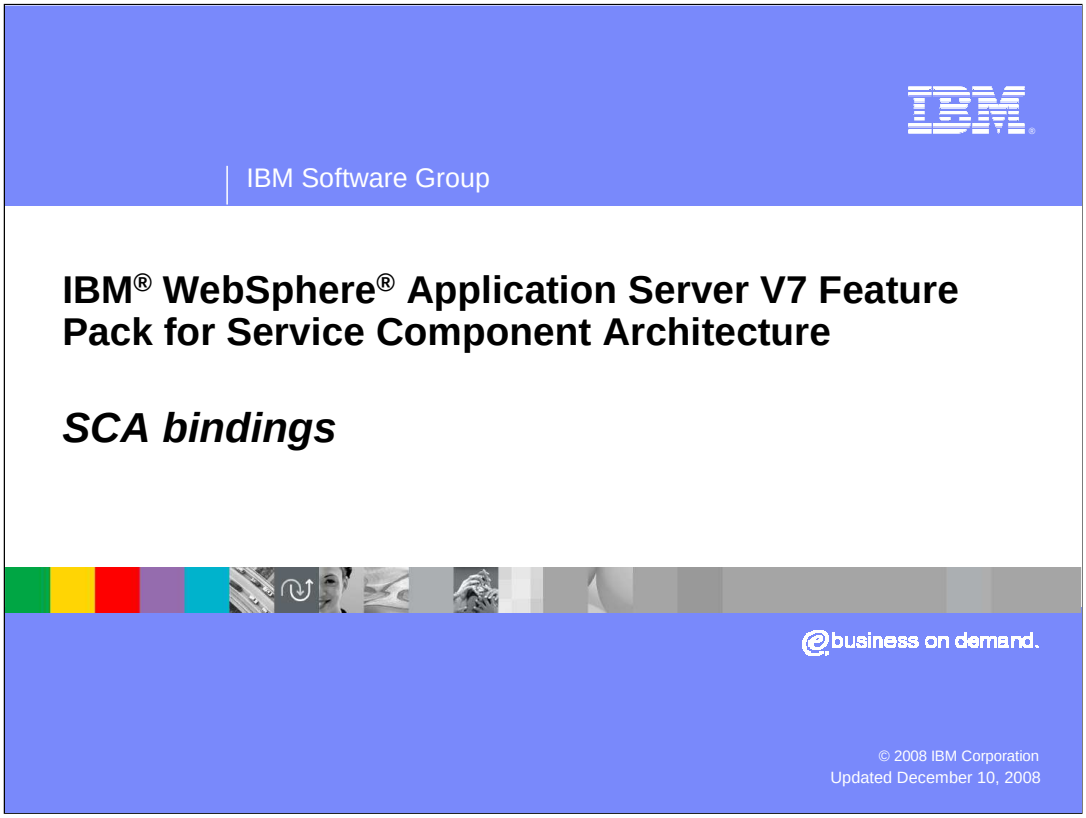
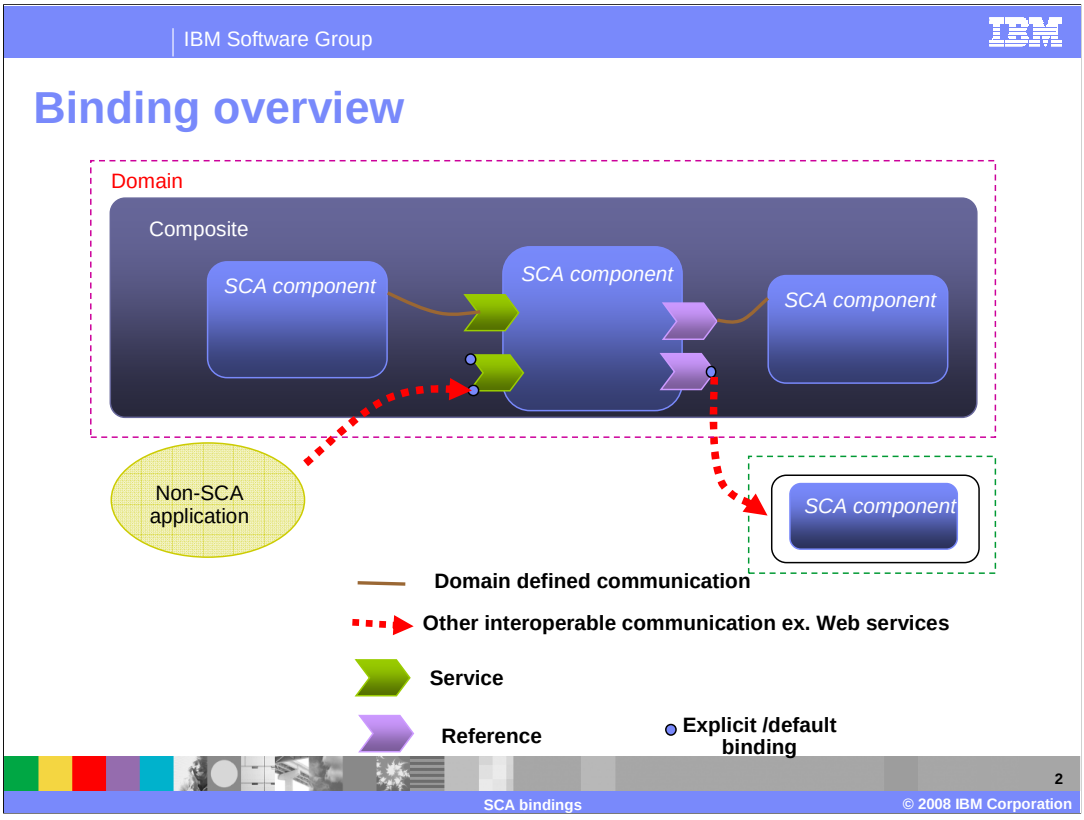


This presentation covers the SCA bindings.



Services and references allow a component to communicate with other applications. By design, however, they say nothing about how that communication happens. *Bindings* do the job of specifying this communication. The figure shows where bindings fit into the SCA picture.

A binding specifies exactly how communication should be done between an SCA component and something else. Depending on what it is communicating with, a component might or might not have explicitly specified bindings. As the figure shows, a component that communicates with another component in the same domain, even one in another process or on another machine, need not have any explicit bindings specified. In this scenario, the default binding would be used.

To communicate outside its domain, a component's creator (or perhaps the person who deploys the component) must specify one or more bindings for this communication. Outside domain would include a non-SCA application or an SCA application running in some other domain. Each binding defines a particular protocol that can be used to communicate with this service or reference. A single service or reference can have multiple bindings, allowing different remote software to communicate with it in different ways.

Note that an SCA application communicating with another SCA application in a different domain sees that application just like a non-SCA application; therefore would require a binding to be specified for that communication.

Binding: Definition

- A binding is…
  - A description of an access mechanism offered by a service
  - A description of an access mechanism needed by a reference
  - An aspect of a service or a reference

As mentioned in the previous slide, binding specifies exactly how communication should be done between an SCA component and something else. Therefore binding can be defined in three ways. One way is as a description of an access mechanism offered by a Service. Another way is as a description of an access mechanism needed by a Reference or as an aspect of a Service or a Reference.

Specifying bindings

- Bindings are specified indirectly in Java™
- Bindings are:
  - Set explicitly in the component's SCDL configuration file
  - If not specified the chosen binding is always the SCA default binding
- A binding is defined by a ***binding element***
  - Example:

According to the SCA Java component implementation specification version 1.0, bindings are specified **indirectly** in Java.

They are independent of the programming language used. They may be specified in SCDL or defaulted.

The Web service or EJB bindings are specified in a component's SCDL configuration file. If bindings are not specified the SCA default binding is used.

Here is an example of how a binding for a component's service might be specified:

<binding.ws uri="http://www.qwickbank.com/services/serviceA"/>

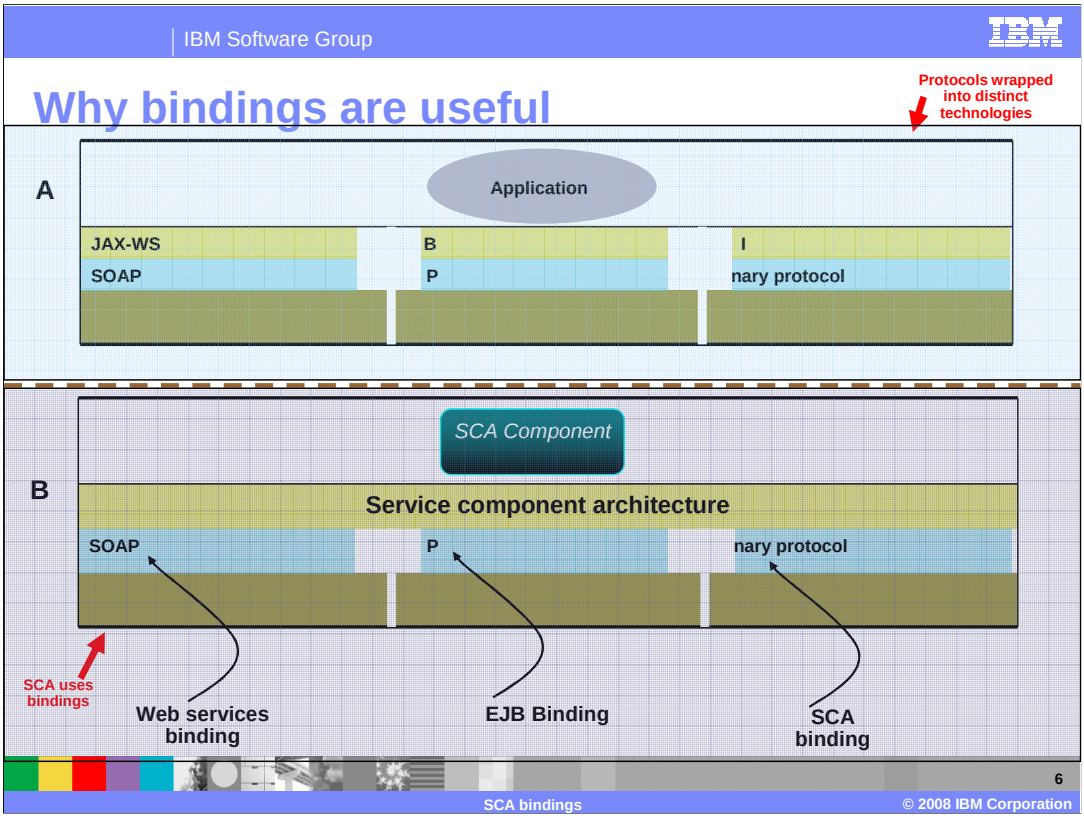
In this example, the **binding** element specifies two things: what protocol the binding uses and where the service can be accessed using this protocol.

The **binding.ws** in the element's name indicates the first of these, specifying the Web services binding. The element's **URI** attribute indicates the second, specifying the URL at which the service can be found. Note that it is also possible and much more likely for a Web service binding to use a relative URL rather than the absolute form shown here.

Purpose of bindings

- Bindings determine how a component communicates with the world outside its domain
- References use bindings to describe the access mechanism used to call a service
- Services use bindings to describe the access mechanism that clients have to use to call the service

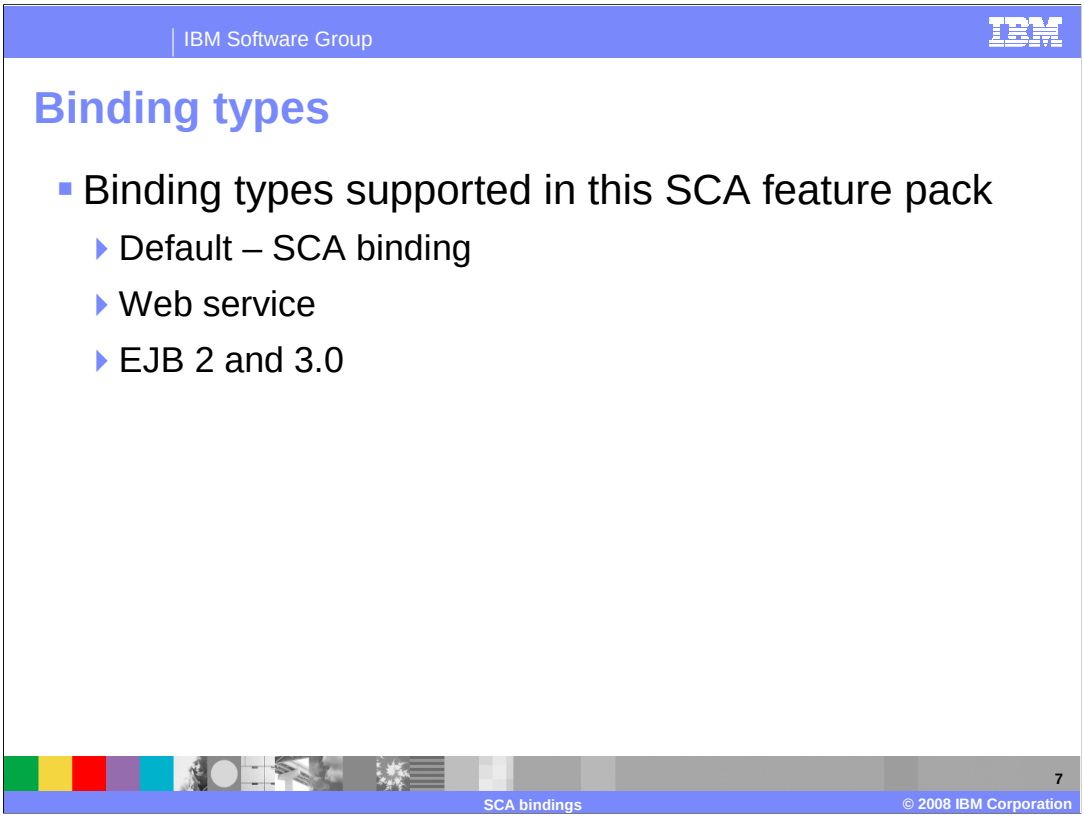
Bindings determine how a component communicates with the world outside its domain. They are used by both services and references. References use bindings to describe the access mechanism used to call a service, which can be a service provided by another SCA component. Services use bindings to describe the access mechanism that clients, which can be a client from another SCA component, have to use to call the service. In general, SCA allows each remotable service and each reference to specify the protocols it supports using bindings.



This figure shows why bindings are useful. Think of how applications use different protocols in Java EE5 and its JEE predecessors. As shown in A, each protocol is provided by a distinct technology, so each one has its own application programming interface. Using SOAP over HTTP, for example, typically means building on JAX-WS (or JAX-RPC in JEE 1.4), while using IIOP means using EJBs. This approach mixes business logic.

SCA takes a simpler approach. Rather than wrapping different protocols into distinct technologies with different APIs, SCA allows each remotable service and each reference to specify the protocols it supports using bindings. The programming model seen by an application remains the same regardless of which protocol is used, as the figure B shows.

To be accessible through SOAP over HTTP, for example, an SCA service uses the Web services binding. Similarly, the EJB session bean binding allows access to session beans using IIOP.



Binding types supported in the current release of the SCA feature pack include default (SCA) binding, Web service binding and EJB 2 and 3 bindings. These bindings will be discussed in more detail in their individual presentations.

Note that: SCA in itself supports the use of multiple different types of bindings like ***data base stored procedure binding,*** and ***EIS service binding.*** The current release of the SCA feature pack does not support these bindings.

Summary

- Bindings are useful in that they specify exactly how communication should be done between an SCA component and something else
- Bindings separate communication details from business logic simplifying life for you
- The three supported bindings in the SCA feature pack are:
  - Default binding
  - Web service binding
  - EJB binding

Services and references allow a component to communicate with other applications. By design, however, they say nothing about how that communication happens. *Bindings* do the job of specifying this communication. Bindings are useful in that they specify exactly how communication should be done between an SCA component and something else. They separate communication details from business logic simplifying life for you. There are three supported bindings in the SCA feature pack namely default, Web service and EJB bindings. Default binding which is the default binding and is used when no other binding is specified for a configuration of a component reference or service. Web service binding defines the manner in which a service can be made available as a Web service, and in which a reference can invoke a Web service. Last but not least, EJB session beans are used to implement business services.

References

- SCA Specifications:

http://www.osoa.org/display/Main/Service+Component+Architecture+Specifications

- EJB 3.0 bindings:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/cejb_bindingsejbfp.html

The Service Component Architecture specifications and the information center article on EJB 3.0 bindings are available at the addresses shown here.

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WASv7SCA_SCABindings_overview.ppt

This module is also available in PDF format at: ../WASv7SCA_SCABindings_overview.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, and the following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

WebSphere

If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of other IBM trademarks is available on the Web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

EJB, Java, and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. 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 in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.