



JavaOne

Building, Assembling, and Deploying Composite Service Applications

Michael Rowley, Ph.D.

Director, Technology, BEA Systems, Inc.

Michael Beisiegel

Distinguished Engineer, IBM Corporation

TS-8554



Click

What You Will Gain

Learn how to build and deploy composite service applications using Service Component Architecture (SCA).



Agenda

In a Nutshell

Overview

Construction

Assembly

Deployment

Policy

Agenda

In a Nutshell

Overview

Construction

Assembly

Deployment

Policy

In a Nutshell

- A development and deployment model for SOA
- Service-based models for:
 - Construction
 - Assembly
 - Deployment
- In a distributed and heterogeneous environment of:
 - Multiple languages
 - Multiple container technologies
 - Multiple service access methods

History and Status

- Outgrowth of BEA and IBM Java™ technology collaboration
 - See “CommonJ” Specifications
- November 2005: 0.9 specs published
 - BEA, IBM, Oracle, SAP, IONA, and Sybase
- July 2006: 0.95 specs and OSOA.org (Open SOA)
 - Added: Cape Clear, Interface21, Primeton Technologies, Progress Software, Red Hat, Rogue Wave, Siemens AG, Software AG, Sun, TIBCO
- **March 2007: 1.0 specs published**
 - **Submitted to OASIS**
- Open source
 - SCA and SDO Runtime: Apache Tuscan(y) (Incubator)
 - Tooling: Eclipse SOA Tools Project (STP)

Open SOA—<http://www.osoa.org>

The OSOA Collaboration



Supporters of the OSOA Collaboration



Agenda

In a Nutshell

Overview

Construction

Assembly

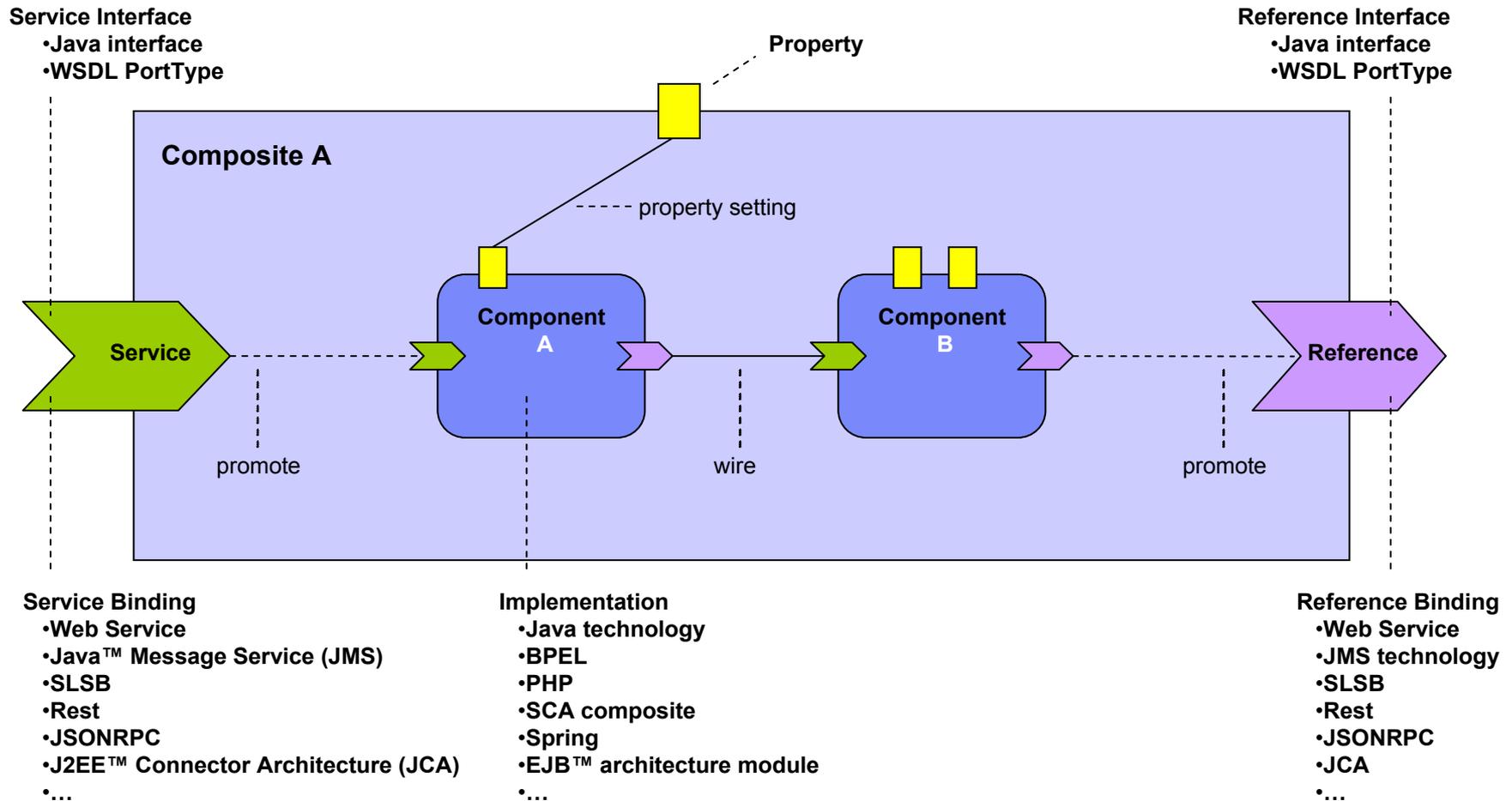
Deployment

Policy

Construction

- Constructing **service implementations**
 - Implementer focuses on business logic
 - No code is dependent on the means of accessing the service
- Defining **service dependencies**
 - Use business services without knowing how they will be accessed
 - Only the interface is known
- Defining **other configuration features**
 - Properties
 - Policies

Assembly

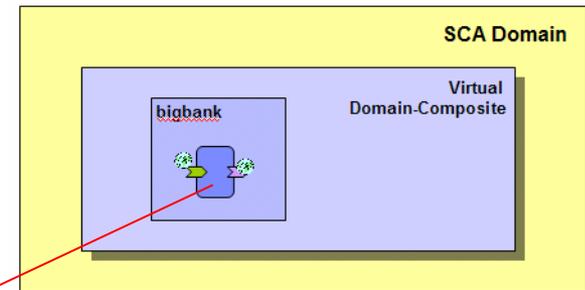


Deployment

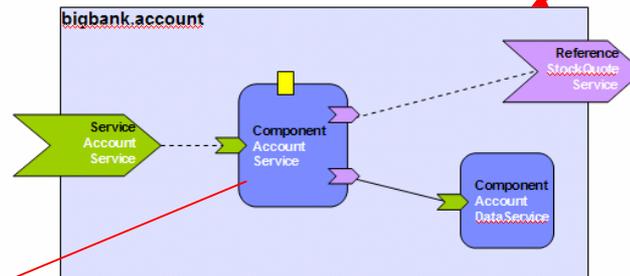
- Service implementations are deployed into an **SCA Domain**
 - Represents the SCA run-time configuration
 - In general an SCA run-time is distributed and heterogeneous
 - Defines the scope of what can be connected by SCA wires
- Domain configuration is a composite
 - Final configuration for service dependencies, properties, bindings, policies...
- Implementation artifacts and their configuration added as **Contributions**
 - Allows many different packaging formats, although ZIP is portable format
 - Artifacts may be shared between contributions
 - Java class files, XSD files, WSDL files...

Construction, Assembly, Deployment

Deployment



Assembly



Construction

```

@Remotable
public interface AccountService {
    AccountReport getAccountReport(String customerID);
}

public class AccountServiceImpl implements AccountService {
    ...
    @Reference
    public void setAccountDataService(AccountDataService value) {
        accountDataService = value;
    }
    @Reference
    public void setStockQuoteService(StockQuoteService value) {
        stockQuoteService = value;
    }
    @Property
    public void setCurrency(String value) {
        currency = value;
    }
    ...
}

```

Agenda

In a Nutshell

Overview

Construction

Assembly

Deployment

Policy

Construction

Component Implementation

- Provides the code necessary to execute a component
- Support for different **implementation technologies**, samples are:
 - Java class file
 - BPEL Process Definition
 - PHP script
 - SCA **Composite**
- Declares SCA visible and **configurable features**
 - **Services** provided
 - **References** to required services
 - **Properties** required
 - **Policy Intents** required
- Maybe used and configured by multiple components
- Services and References are typed by **Interfaces**

Construction

Implementation: Java technology

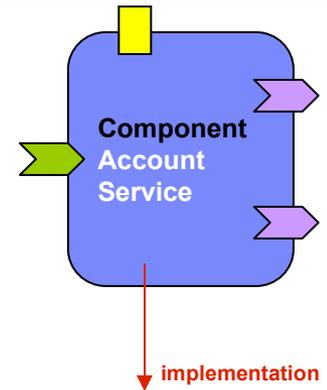
```

@Remotable
public interface AccountService {

    AccountReport getAccountReport(String customerID);
}

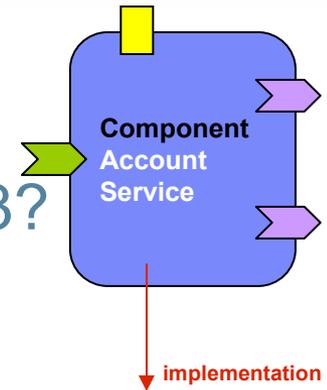
public class AccountServiceImpl implements AccountService {
    ...
    @Reference
    public void setAccountDataService(AccountDataService value) {
        accountDataService = value;
    }
    @Reference
    public void setStockQuoteService(StockQuoteService value) {
        stockQuoteService = value;
    }
    @Property
    public void setCurrency(String value) {
        currency = value;
    }
    ...
}

```



Construction

Implementation: What about EJB™ Specification 3?



```
@Remote
public interface AccountService {
    @WebMethod
    AccountReport getAccountReport(String customerID);
}
```

```
@Stateless @WebService
public class AccountServiceImpl implements AccountService {
    ...
    @EJB(beanName="...")
    public void setAccountDataService(AccountDataService value) {
        accountDataService = value;
    }
    @WebServiceRef(wsdlLocation="...")
    public void setStockQuoteService(StockQuoteService value) {
        stockQuoteService = value;
    }
    @Resource
    public void setCurrency(String value) {
        currency = value;
    }
    ...
}
```

Construction

Implementation: BPEL

```

<process name="AccountServiceImpl"
  targetNamespace="http://bigbank.com"
  xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
  xmlns:sca="http://www.osoa.org/xmlns/sca/1.0"
  xmlns:bb="http://bigbank.com">

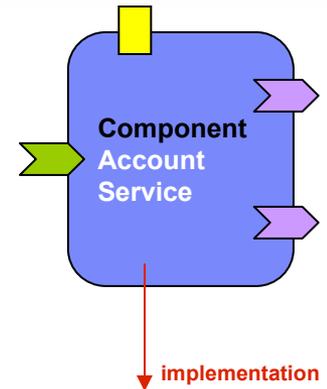
  <partnerLinks>
    <partnerLink name="AccountService" >
      partnerLinkType="bb:acctntServiceLT" myRole="accountService" />
    <partnerLink name="accountDataService" >
      partnerLinkType="bb:accountDataLT" partnerRole="accountDataService" />
    <partnerLink name="stockQuoteService" >
      partnerLinkType="bb:stockQuoteLT" partnerRole="stockQuoteService" />
    </partnerLinks>

    <variables>
      <variable name="currency" type="xsd:string" sca:property="yes" />
    </variables>

    ...

  </process>

```



Construction

Implementation: PHP

```

<?php
include "SCA/SCA.php";

/**
 * @service ➡
 */
class AccountServiceImpl {

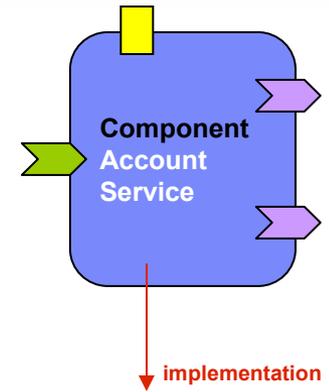
    /**
     * @reference ➡
     */
    public $accountDataService;

    /**
     * @reference ➡
     */
    public $stockQuoteService;

    /**
     * @property 🟡
     */
    public $currency;

    function getAccountReport($customerID) {
        ...
    }
}
?>

```



Construction

Interfaces

- Service Interfaces in **multiple languages**, samples are:
 - WSDL 1.1 portType
 - WSDL 2.0 interface
 - Java programming language
 - C++
- SCA specific **interface extensions**
 - **Remotable** or **Local-only**
 - **Conversational**
 - Session identifying information will be sent in headers
 - **Bi-directional**
 - Interface with companion callback interface
 - **One-way** operations

Agenda

In a Nutshell

Overview

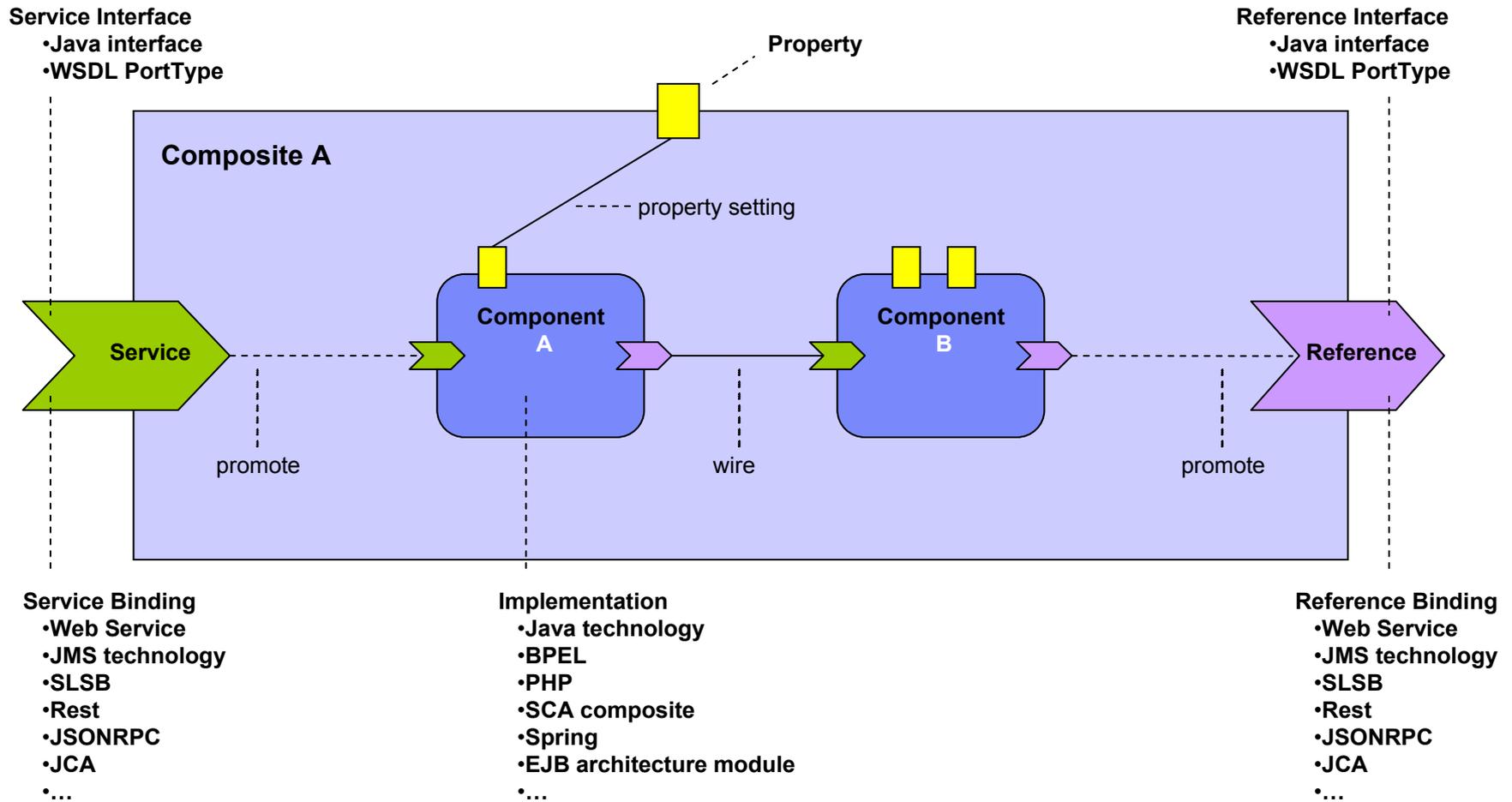
Construction

Assembly

Deployment

Policy

Assembly



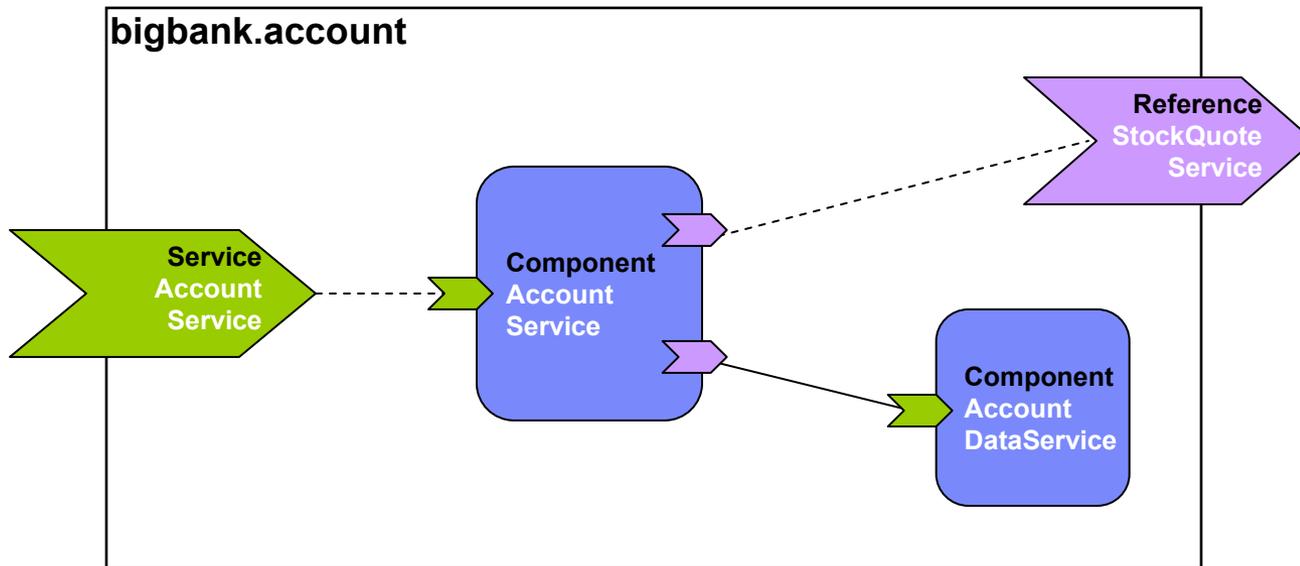
Assembly

Component

- Configured instance of an implementation within a composite
 - More than one component can use the same implementation
- **Configure** Implementation
 - Set **Property Values and Reference Targets**
 - Set **Bindings** for Services and References
- **Promote** Services, References, and Properties to composite level
- Declare additional required Policy Intents for components, services, and references, for example:
 - Authentication and Authorization
 - Reliability
 - Transaction Requirements

Assembly

Sample: bigbank.account Composite Diagram



Assembly

Sample: bigbank.account Composite XML

Standard representation
is XML. Other
representations possible

```

<composite name="bigbank.account" xmlns="http://www.oxa.org/xmlns/sca/1.0"
           targetNamespace="http://bigbank.com" ... >

  <service name="AccountService" promote="AccountService">
    <binding.ws requires="soap.1_1"/>
  </service>

  <reference name="stockquoteService"
             promote="AccountService/stockQuoteService">
    <binding.ws uri="http://www.quickstockquote.com/StockQuoteService"/>
  </reference>

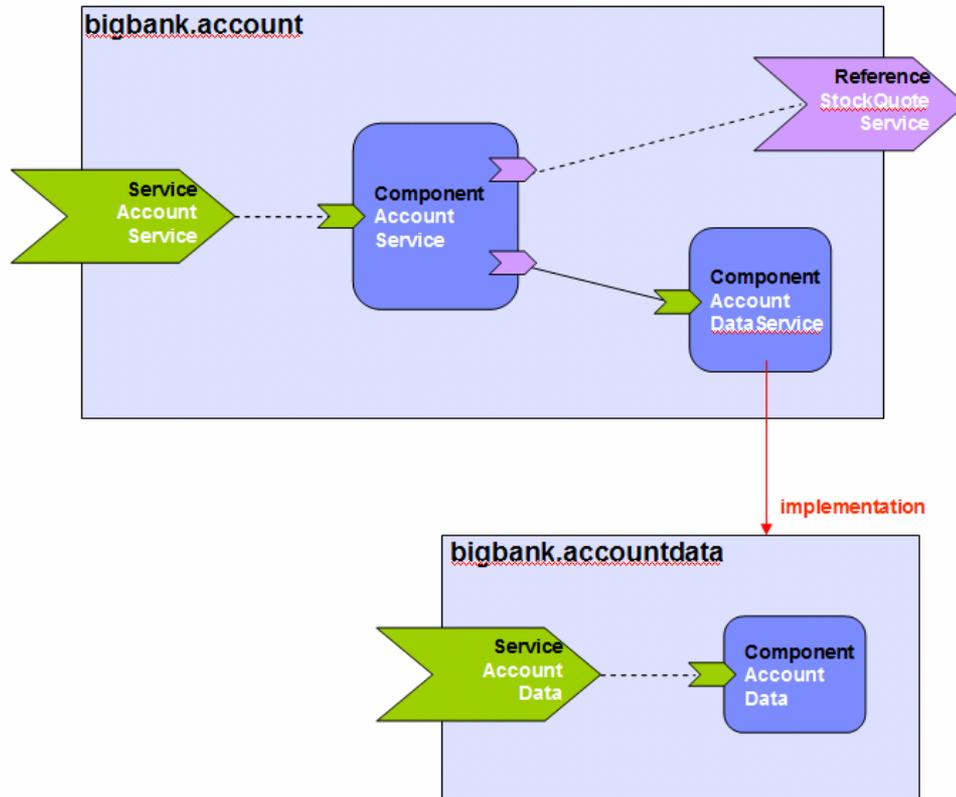
  <component name="AccountService">
    <implementation.java class="bigbank.account.AccountServiceImpl"/>
    <reference name="accountDataService" target="AccountDataService"/>
    <property name="currency">EURO</property>
  </component>

  <component name="AccountDataService">
    <implementation.java class="bigbank.accountdata.AccountDataServiceImpl"/>
  </component>

</composite>
  
```

Assembly

Sample: Recursive Composition



Assembly

Sample: Recursive Composition

```

<composite name="bigbank.account" xmlns="http://www.oesa.org/xmlns/sca/1.0"
           xmlns:bb="http://bigbank.com"
           targetNamespace="http://bigbank.com" ... >

  <service name="AccountService" promote="AccountService">
    <binding.ws requires="soap.1_1"/>
  </service>

  <reference name="stockquoteService"
             promote="AccountService/stockQuoteService">
    <binding.ws uri="http://www.quickstockquote.com/StockQuoteService"/>
  </reference>

  <component name="AccountService">
    <implementation.java class="bigbank.account.AccountServiceImpl"/>
    <reference name="accountDataService" target="AccountDataService"/>
    <property name="currency">EURO</property>
  </component>

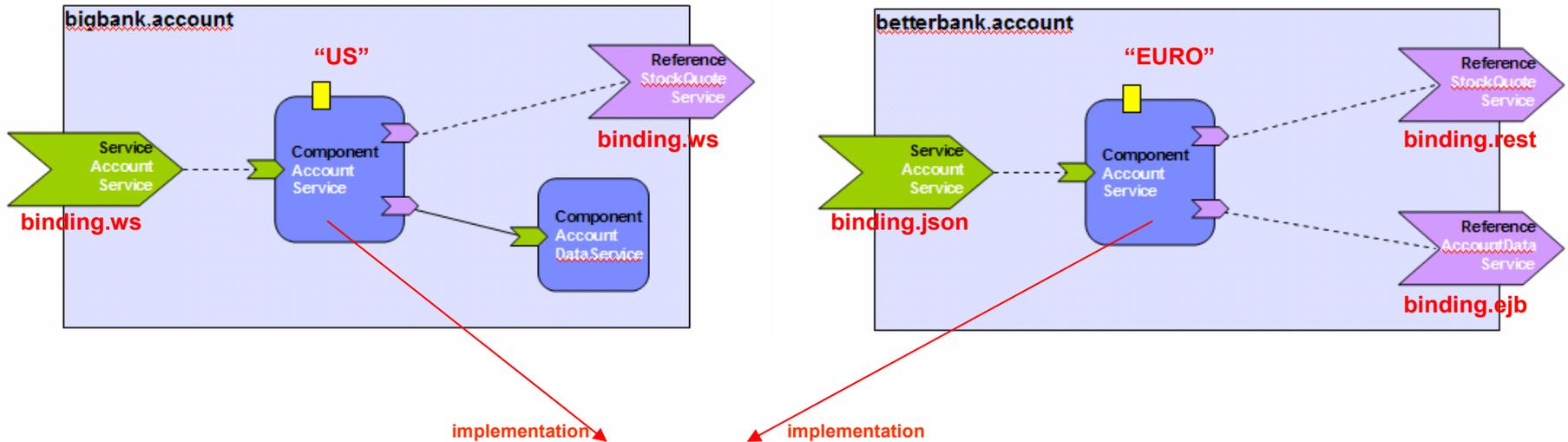
  <component name="AccountDataService">
    <implementation.composite name="bb:bigbank.accountdata"/>
  </component>

</composite>

```

Assembly

Sample: Building Custom Solutions from Assets



```

@remotable
public interface AccountService {
    AccountReport getAccountReport(String customerID);
}

public class AccountServiceImpl implements AccountService {
    ...
    @Reference
    public void setAccountDataService(AccountDataService value) {
        accountDataService = value;
    }
    @Reference
    public void setStockQuoteService(StockQuoteService value) {
        stockQuoteService = value;
    }
    ...
    @Property
    public void setCurrency(String value) {
        currency = value;
    }
    ...
}
    
```

Agenda

In a Nutshell

Overview

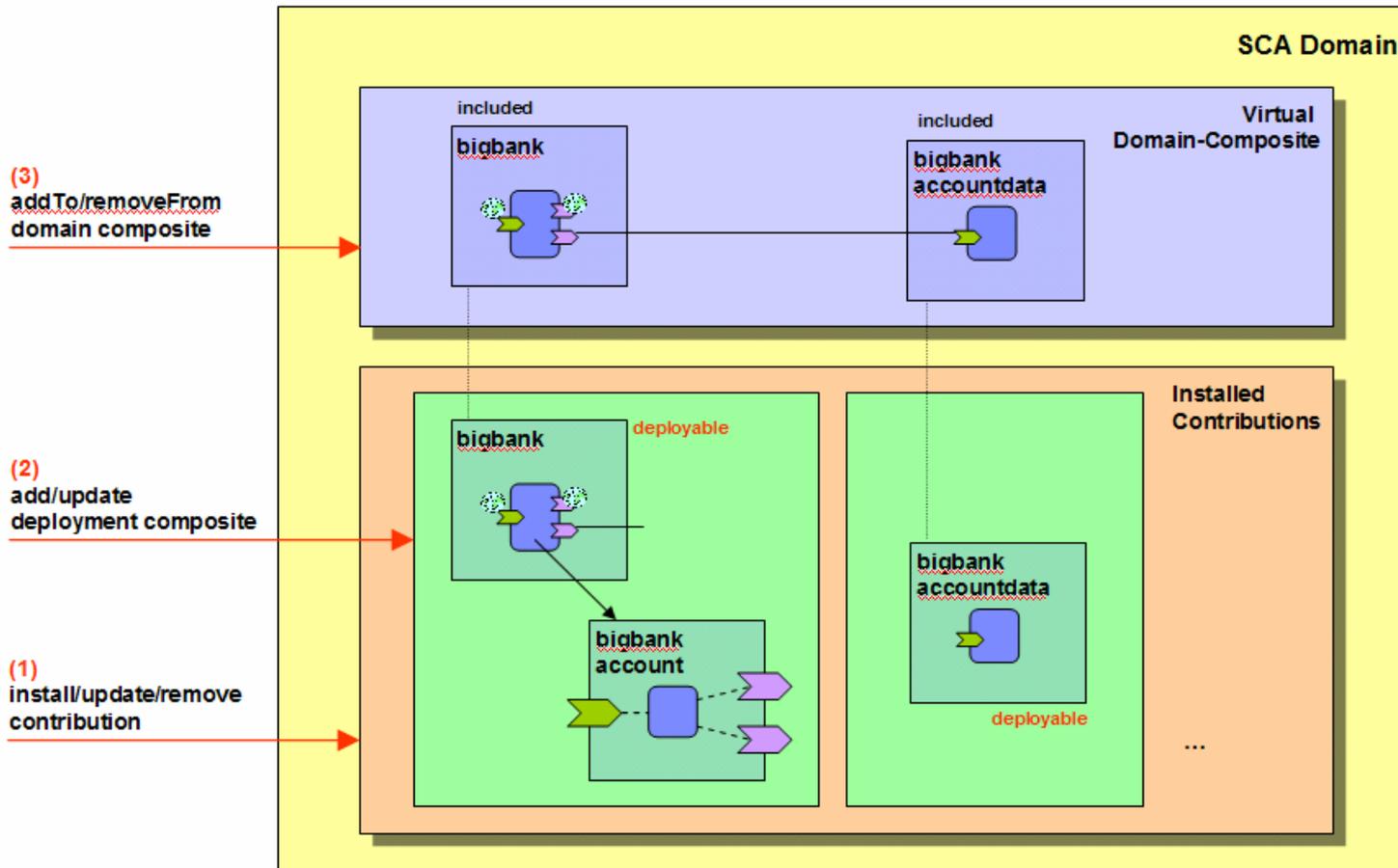
Construction

Assembly

Deployment

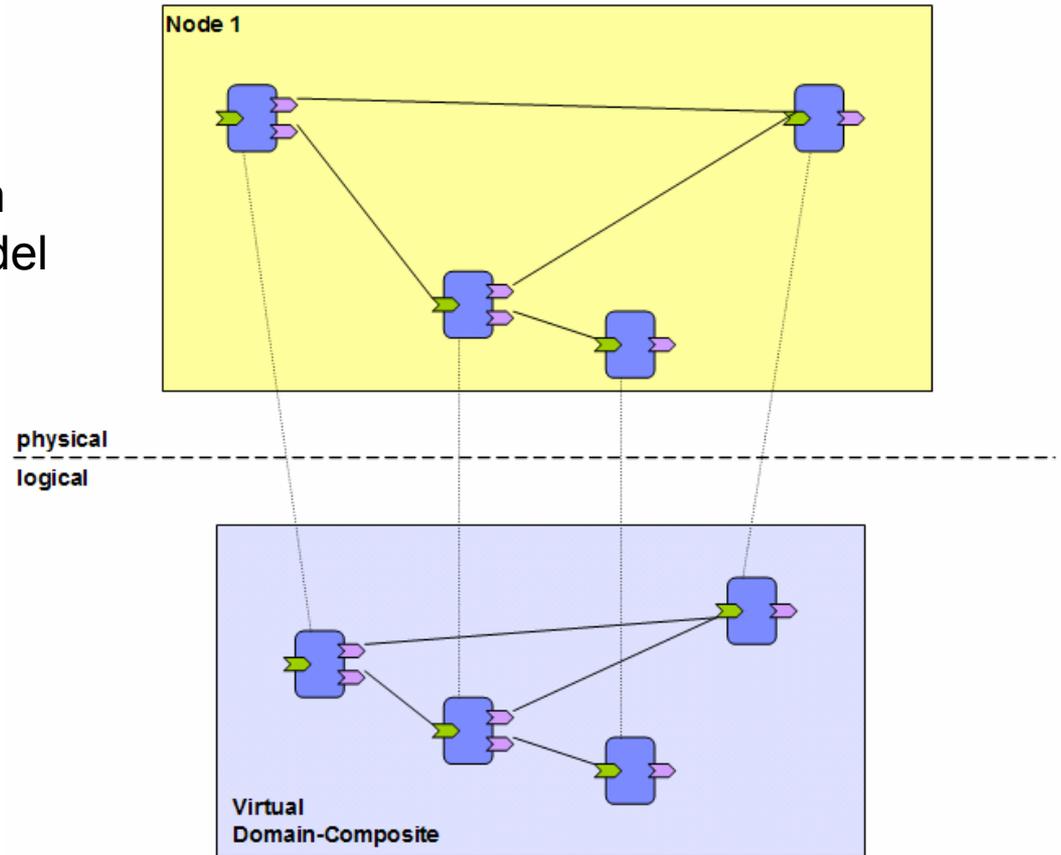
Policy

Deployment



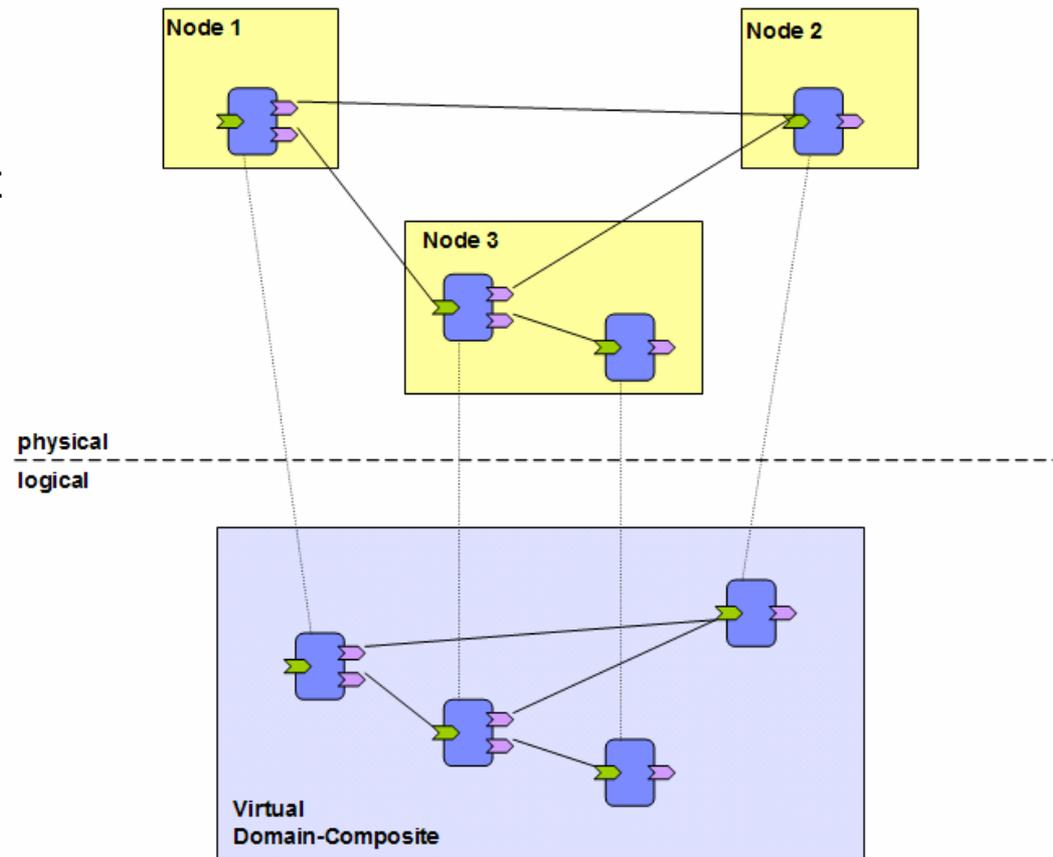
Deployment

Physical topology is specified separately from the logical assembly model



Deployment

A distributed deployment of the assembly



Deployment

Contribution

- **Contains XML artifacts**
 - SCA: Composite, Definitions (Intents, PolicySets)...
 - Non-SCA: BPEL, WSDL, XSD...
 - XML constructs use qualified names for their identity (ns + local name)
- ...and **non-XML artifacts**
 - Java class files, C++ classes, PHP scripts...
- **Packaging**
 - Allows many different packaging formats
 - Filesystem directory, OSGI bundle, ZIP, Java™ Archive (JAR), WAR, EAR...
 - Requires ZIP as interoperable packaging format
 - Packaging Requirements
 - Contents presentable as hierarchy of resources based of a single root
 - META-INF/sca-contribution.xml

Agenda

In a Nutshell

Overview

Construction

Assembly

Deployment

Policy

Policy

- SCA “Policy” is infrastructure configuration
 - **Interaction policies**
 - Affects contract between service provider and consumer
 - Authentication, Encryption, Non-Repudiation, Reliable Messaging...
 - **Implementation policies**
 - Affects contract between component and container
 - Authorization, Transactions, Monitoring and Logging...
- SCA simplifies the use of policy
 - **Policy Administrator** creates complex reusable policies with simple names (intents)
 - **Component Developer** uses small fixed set of intents

Policy

Framework Elements

- SCA policy **intent**
 - Each represent a single abstract QoS intent
 - May be qualified
- SCA **policy sets**
 - Represent a collection of concrete policies to realize an abstract QoS intent
- WS-Policy
 - A syntax for concrete policies in policy sets
 - Others possible...

Policy

Administrator

- Policy Administrator defines the intents

```
<intent name="authentication.basic" appliesTo="sca:binding">  
  <description>requires basic authentication</description>  
</intent>
```

```
<intent name="reliability" appliesTo="sca:binding">  
  <description>requires reliable interaction</description>  
</intent>
```

Policy

Administrator

- ...and defines policy sets that achieve them

```
<policySet name="SecureReliablePolicy"
  provides="authentication.basic reliability"
  appliesTo="binding.ws" ... >
  <wsp:PolicyAttachment>
    <!-- policy expression and policy subject for "basic authentication" -->
    ...
  </wsp:PolicyAttachment>
  <wsp:PolicyAttachment>
    <!-- policy expression and policy subject for "reliability" -->
    ...
  </wsp:PolicyAttachment>
</policySet>
```

Policy

Developer, Assembler, Deployer

```
<composite name="usingComposite" ... >
...
  <component name="aComponent">
    <implementation.composite name="usedComposite"/>
    <reference name="aService">
      <binding.ws policySets="SecureReliablePolicy" ... />
    </reference>
  </component>
...
</composite>
<composite name="usedComposite" ... >
...
  <reference name="aService" promote="..."
    requires="authentication.basic reliability" />
</composite>
```

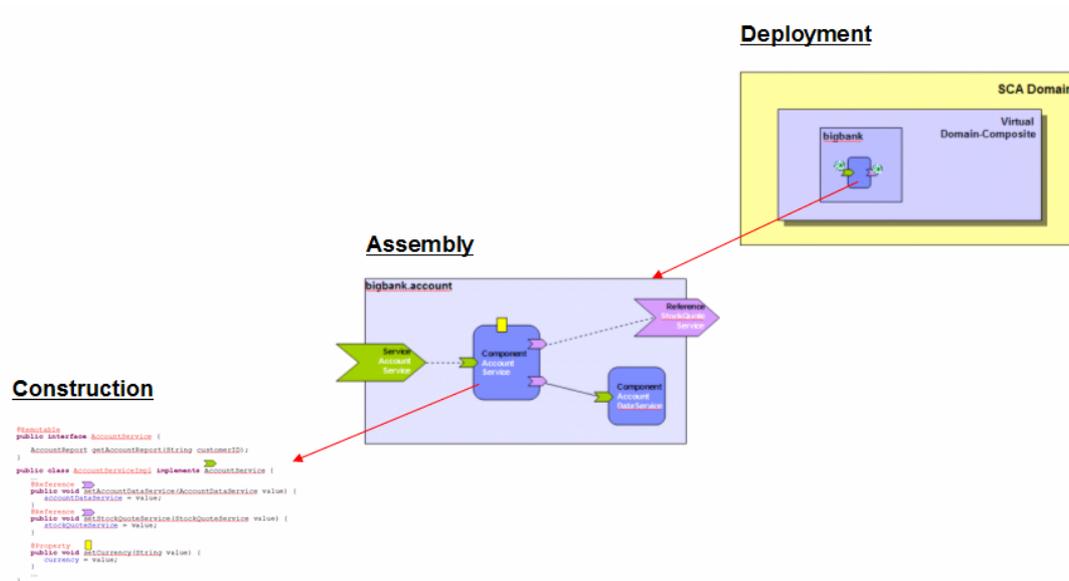


2) Configure

1) Declare Requirement

Summary

- SCA is a multi-language service-oriented approach to construction, assembly, and deployment



For More Information

- Other SCA events at this JavaOneSM conference
 - **TS-8554** Building, Assembling, and Deploying Composite Service Applications
 - **TS-80955** Open for Business: Using Open Standards to Make SOAs Safe for Developer
 - **TS-8194** Spring and SCA as the Basis for Distributed Service Applications
 - **TS-41500** Service Component Architecture Meets the Java Platform, Enterprise Edition
 - **TS-8835** SCA/SDO and Java Technology: Complementary Technologies That Drive Open SOA Environments
 - **BOF-8238** Building Composite Service Applications
- Links
 - <http://www.osoa.org>
 - <http://www.oasis-opencsa.org/>
 - <http://cwiki.apache.org/TUSCANY/home.html>
 - <http://uk.php.net/manual/en/ref.SCA.php>



Q&A

<code />



JavaOne

Building, Assembling, and Deploying Composite Service Applications

Michael Rowley, Ph.D.

Director, Technology, BEA Systems, Inc

Michael Beisiegel

Distinguished Engineer, IBM Corporation

TS-8554