

# Building Applications with Apache Tuscany

**Luciano Resende**  
lresende@apache.org  
<http://lresende.blogspot.com>



**Jean-Sebastien Delfino**  
jsdelfino@apache.org  
<http://jsdelfino.blogspot.com>



**Simon Laws**  
slaws@apache.org



# Abstract

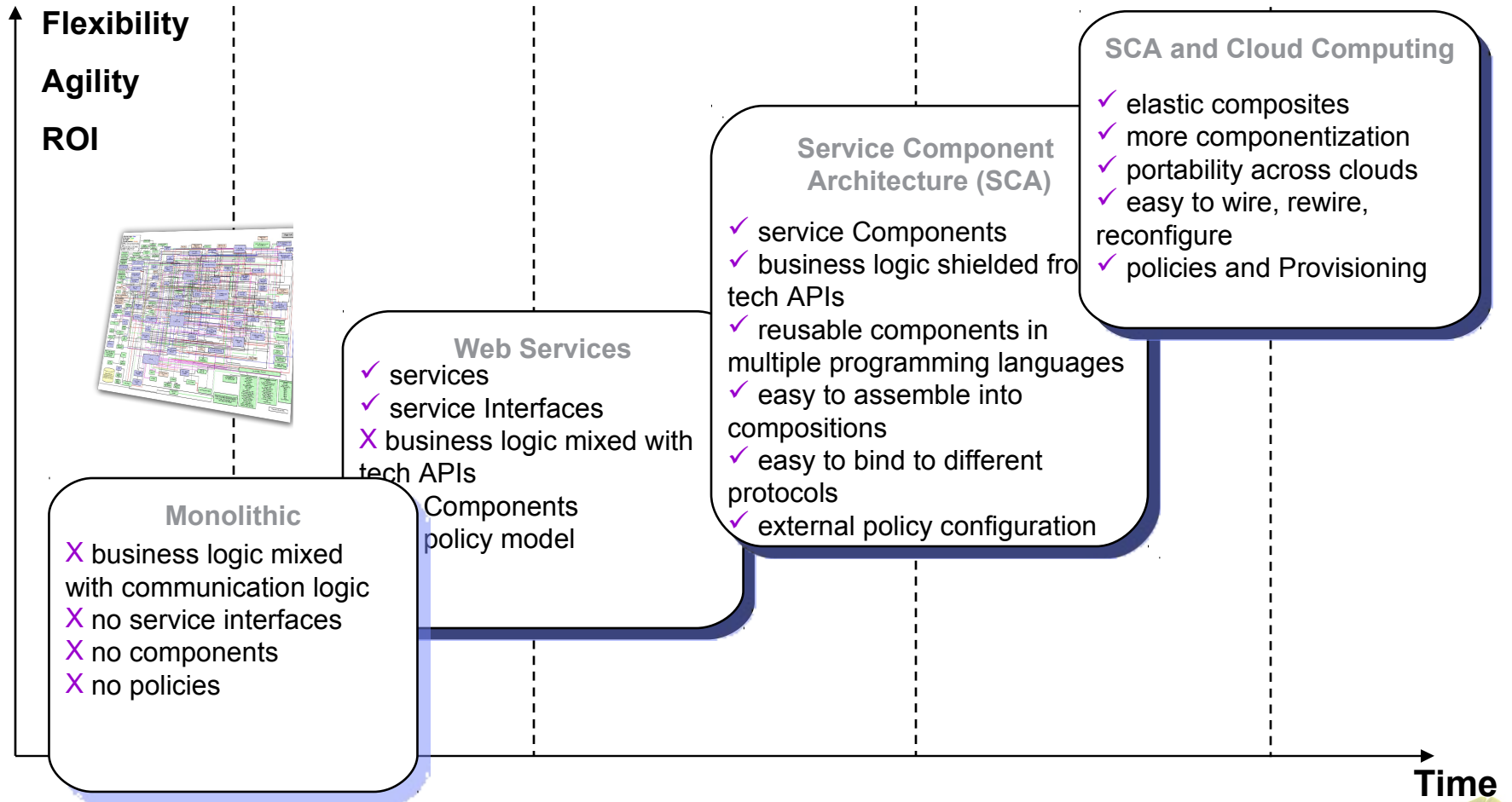
**Building and composing the components of a distributed application can be a challenge and complex bespoke solutions are commonplace. The Apache Tuscany runtime, and the Service Component Architecture (SCA) on which the runtime is based, simplify the process by presenting a component based application assembly model. In this talk we look at the Tuscany travel booking application and explain how the individual components of the application are constructed using a variety of technologies including Java, Spring, BPEL and Python. We also look at how these services are wired together using a variety of communication protocols such as SOAP/HTTP and JSON-RPC. The complete model can then be deployed to both stand-alone and distributed runtimes without changes to the application itself.**

# Agenda

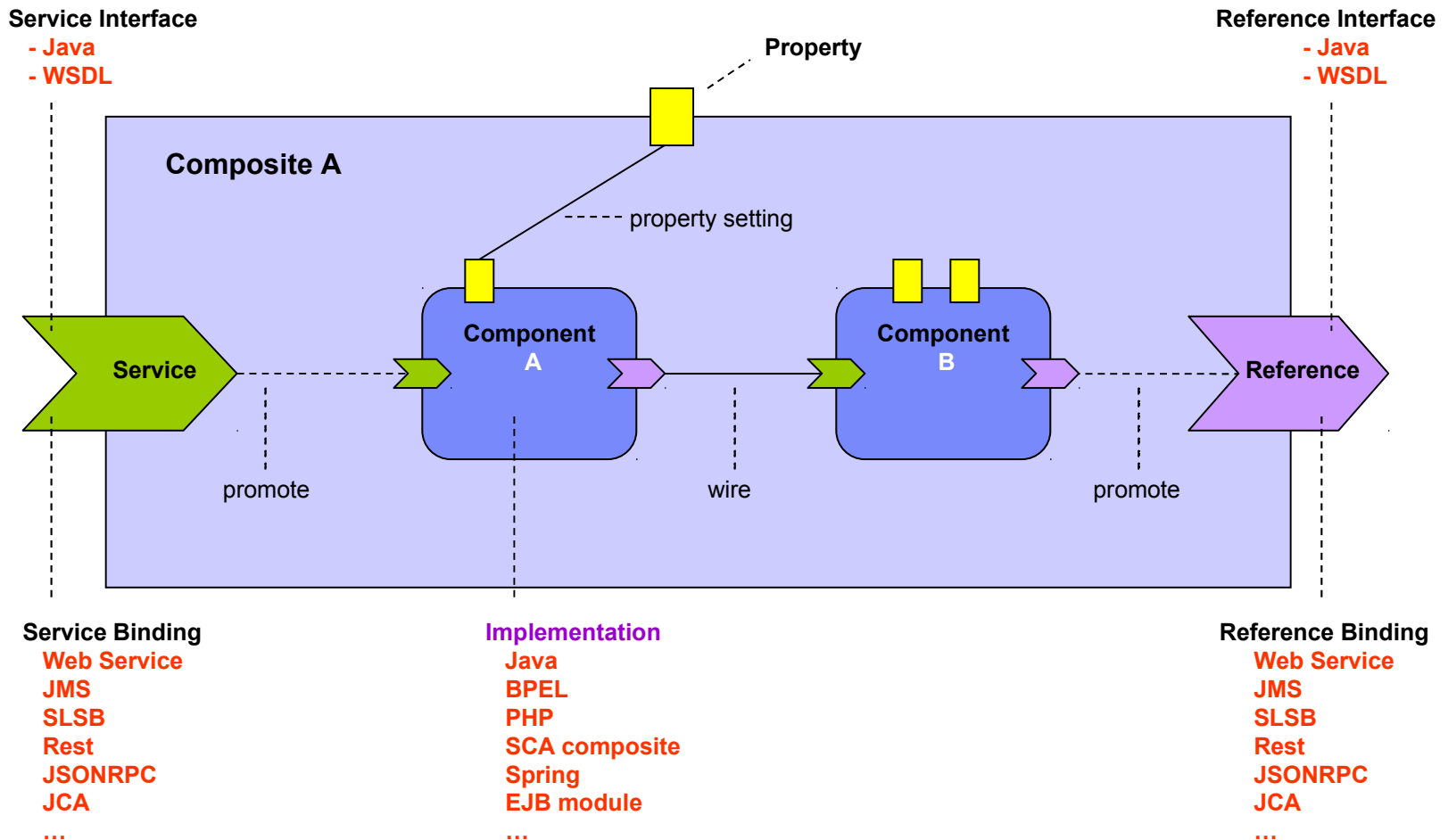
- **Service Component Architecture**
- **Apache Tuscany**
- **TuscanySCATours, a Sample Travel Booking App**
  - Development Process
  - Packaging, Build, Test
  - Extending the App
    - Bindings
    - Spring
    - BPEL
    - Policies
  - Deploying to an SCA Domain
- **Getting Involved**

# SCA and Apache Tuscany

# Enterprise App Development - History

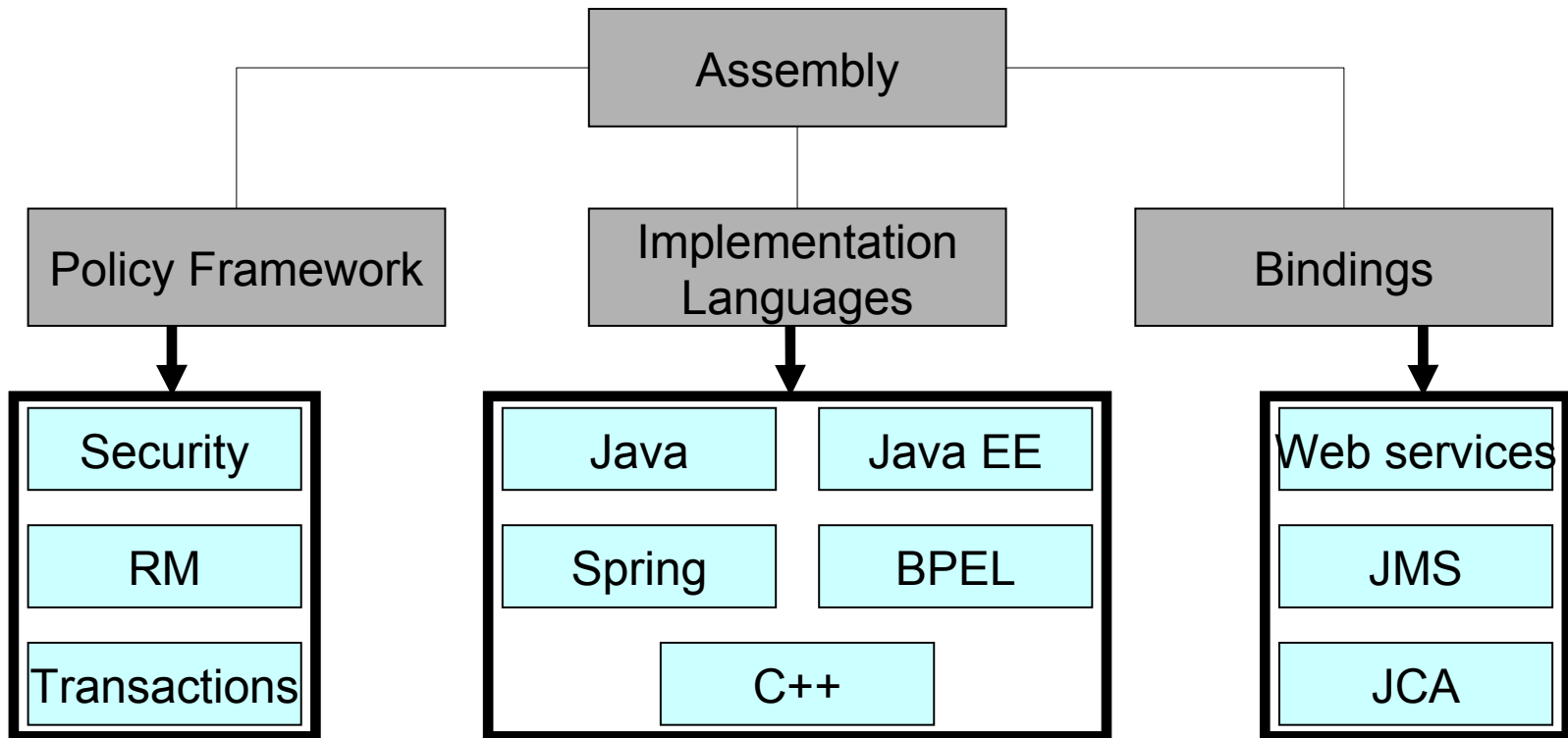


# SCA Assembly Model



# SCA Specifications

- **SCA is going through a formal standardization process at OASIS OpenCSA (<http://www.oasis-opencsa.org>)**



# OASIS Open CSA - <http://www.oasis-opencsa.org/>



## Navigation

Members

About

The **OASIS Open Composite Services Architecture (CSA) Member Section** advances open standards that simplify SOA application development. Open CSA brings together vendors and users from around the world to collaborate on the further development and adoption of the Service Component Architecture (SCA) and Service Data Objects (SDO) families of specifications.

## Steering Committee

23 March 2007 - 1:07pm — [jeff.mischkinsky](#)

Open CSA activities are managed by a Steering Committee. Open CSA membership in an open process. The

- Graham Barber - IBM
- David Burke - TIBCO
- Patrick Leonard - Rogue Wave Software
- Mark Little - Red Hat
- Jeff Mischkinsky - Oracle
- Sanjay Patil - SAP
- Michael Rowley - BEA Systems

## Committees

Several technical committees are affiliated with Open CSA:

### **OASIS Service Component Architecture / Assembly (SCA-Assembly) TC**

*Defining core SCA composition model to simplify SOA application development*

### **OASIS Service Component Architecture / Policy (SCA-Policy) TC**

*Defining an SCA policy framework to simplify SOA application development*

### **OASIS Service Component Architecture / Bindings (SCA-Bindings) TC**

*Standardizing bindings for SCA services and references to communication protocols, technologies and frameworks*

### **OASIS Service Component Architecture / BPEL (SCA-BPEL) TC**

*Specifying how SCA component implementations for SOA can be written using BPEL*

### **OASIS Service Component Architecture / C and C++ (SCA-C-C++) TC**

*Standardizing C and C++ use within an SCA domain for SOA*

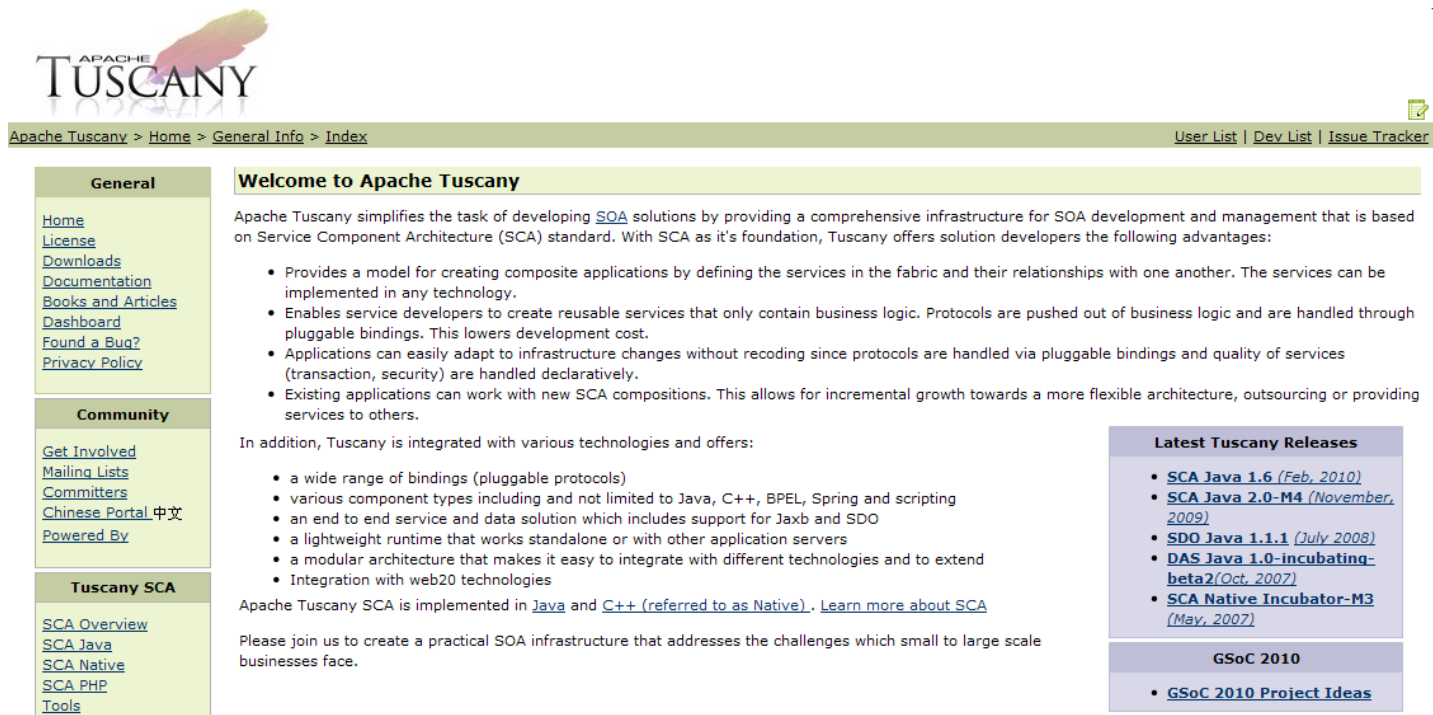
### **OASIS Service Component Architecture / J (SCA-J) TC**

*Standardizing Java (tm) use within an SCA domain for SOA*



# Apache Tuscany

- Apache Tuscany provides a component based programming model which simplifies development, assembly and deployment and management of composite applications.
- Apache Tuscany implements SCA standards defined by OASIS OpenCSA + extensions based on community feedback.



The screenshot shows the Apache Tuscany website homepage. At the top is the Apache Tuscany logo. Below it is a navigation bar with links for Home, General Info, and Index. On the right side of the navigation bar are links for User List, Dev List, and Issue Tracker. The main content area is divided into three columns. The left column contains a 'General' section with links for Home, License, Downloads, Documentation, Books and Articles, Dashboard, Found a Bug?, and Privacy Policy. Below this is a 'Community' section with links for Get Involved, Mailing Lists, Committers, Chinese Portal, and Powered By. The bottom section is 'Tuscany SCA' with links for SCA Overview, SCA Java, SCA Native, SCA PHP, and Tools. The middle column features a 'Welcome to Apache Tuscany' heading followed by a paragraph describing the project's purpose and a bulleted list of advantages. Below this is a section titled 'In addition, Tuscany is integrated with various technologies and offers:' followed by another bulleted list. The right column has a 'Latest Tuscany Releases' section with a list of recent releases and a 'GSoC 2010' section with a link to project ideas.

**General**

- [Home](#)
- [License](#)
- [Downloads](#)
- [Documentation](#)
- [Books and Articles](#)
- [Dashboard](#)
- [Found a Bug?](#)
- [Privacy Policy](#)

**Community**

- [Get Involved](#)
- [Mailing Lists](#)
- [Committers](#)
- [Chinese Portal](#) 中文
- [Powered By](#)

**Tuscany SCA**

- [SCA Overview](#)
- [SCA Java](#)
- [SCA Native](#)
- [SCA PHP](#)
- [Tools](#)

**Welcome to Apache Tuscany**

Apache Tuscany simplifies the task of developing [SOA](#) solutions by providing a comprehensive infrastructure for SOA development and management that is based on Service Component Architecture (SCA) standard. With SCA as it's foundation, Tuscany offers solution developers the following advantages:

- Provides a model for creating composite applications by defining the services in the fabric and their relationships with one another. The services can be implemented in any technology.
- Enables service developers to create reusable services that only contain business logic. Protocols are pushed out of business logic and are handled through pluggable bindings. This lowers development cost.
- Applications can easily adapt to infrastructure changes without recoding since protocols are handled via pluggable bindings and quality of services (transaction, security) are handled declaratively.
- Existing applications can work with new SCA compositions. This allows for incremental growth towards a more flexible architecture, outsourcing or providing services to others.

In addition, Tuscany is integrated with various technologies and offers:

- a wide range of bindings (pluggable protocols)
- various component types including and not limited to Java, C++, BPEL, Spring and scripting
- an end to end service and data solution which includes support for Jaxb and SDO
- a lightweight runtime that works standalone or with other application servers
- a modular architecture that makes it easy to integrate with different technologies and to extend
- Integration with web20 technologies

Apache Tuscany SCA is implemented in [Java](#) and [C++](#) (referred to as [Native](#)). [Learn more about SCA](#)

Please join us to create a practical SOA infrastructure that addresses the challenges which small to large scale businesses face.

**Latest Tuscany Releases**

- [SCA Java 1.6](#) (Feb, 2010)
- [SCA Java 2.0-M4](#) (November, 2009)
- [SDO Java 1.1.1](#) (July 2008)
- [DAS Java 1.0-incubating-beta2](#) (Oct, 2007)
- [SCA Native Incubator-M3](#) (May, 2007)

**GSoC 2010**

- [GSoC 2010 Project Ideas](#)

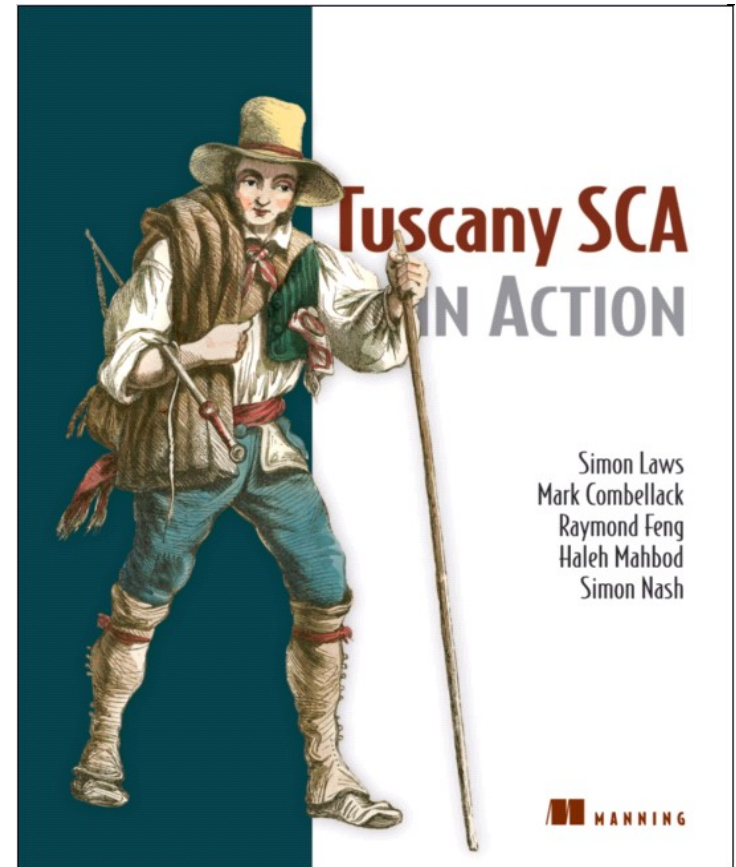


# TuscanySCATours

## Sample Travel Booking Application

# Tuscany SCA In Action

- **Example from Tuscany Book**
  - <http://www.manning.com/laws/>



## Example can be downloaded from Apache Tuscany

- <http://tuscany.apache.org/sca-java-travel-sample-1x-releases.html>

# The TuscanySCATours Shopping Site

SCA Tours - Mozilla Firefox

http://localhost:8080/scatours/

SCA Tours

**TuscanySCATours**  
The reference for great service

**Search for trips, hotels, flights and cars**

From Location:  To Location:   
Start Date:  End Date:   
Number of people:

**Packaged Trip Items**

Select	Name	Description	Location	From - To	Price
<input type="checkbox"/>	FS1DEC06	Florence and Siena pre-packaged tour	LGW - FLR	06/12/09 - 13/12/09	337.5 GBP

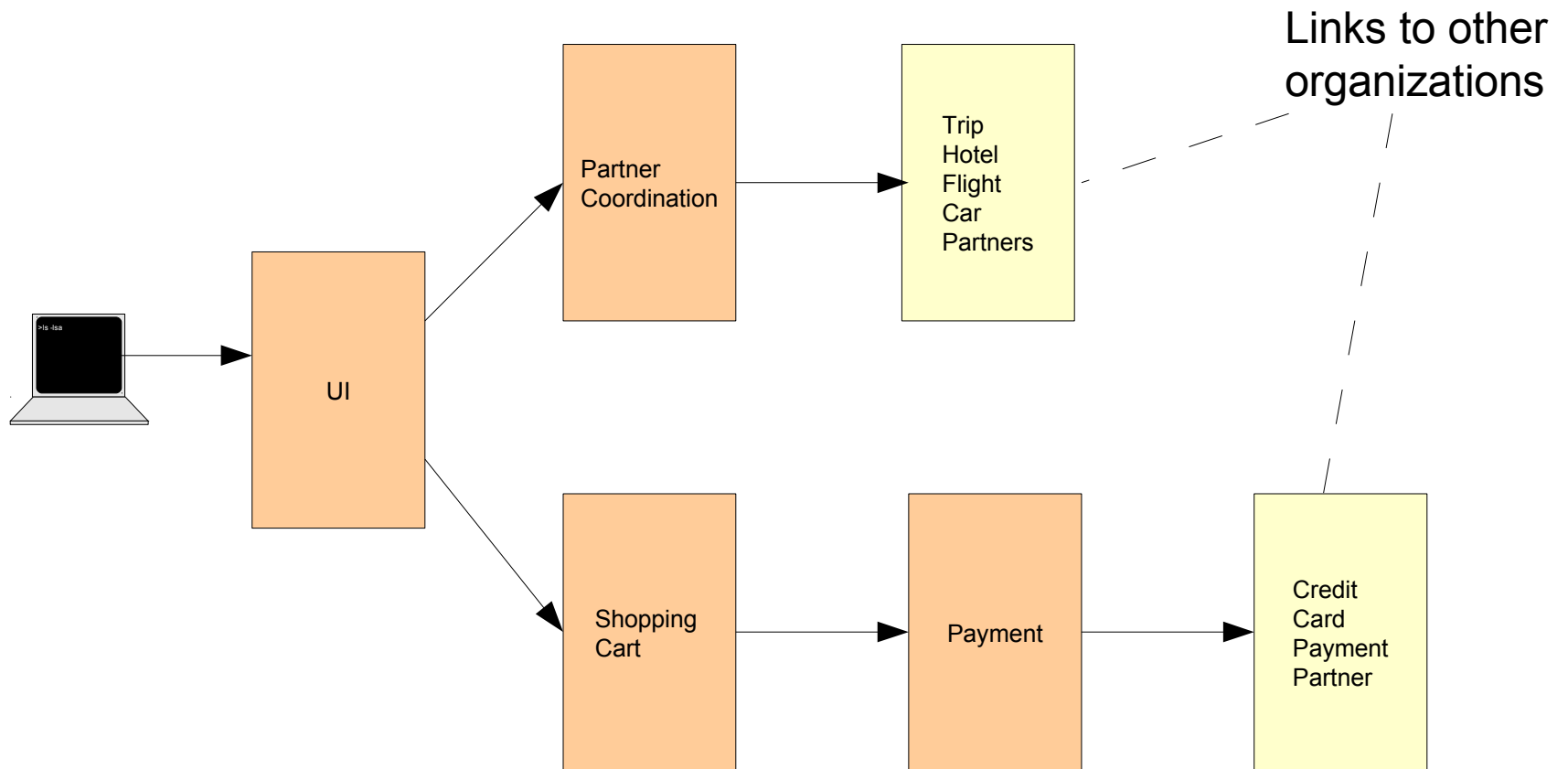
**Bespoke Trip Items**

Select	Name	Description	Location	From - To	Price
<input type="checkbox"/>	EA26	Europe Airlines Airbus A300	LGW - FLR	06/12/09 - 06/12/09	187.5 GBP
<input type="checkbox"/>	EA27	Europe Airlines Airbus A300	FLR - LGW	13/12/09 - 13/12/09	187.5 GBP
<input type="checkbox"/>	Deep Bay Hotel	Wonderful sea views and a relaxed atmosphere	FLR	06/12/09 - 13/12/09	75 GBP

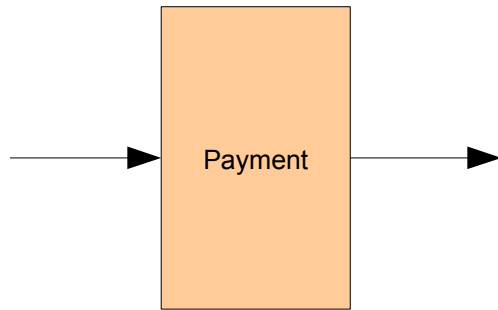
Find:      Match case

Done

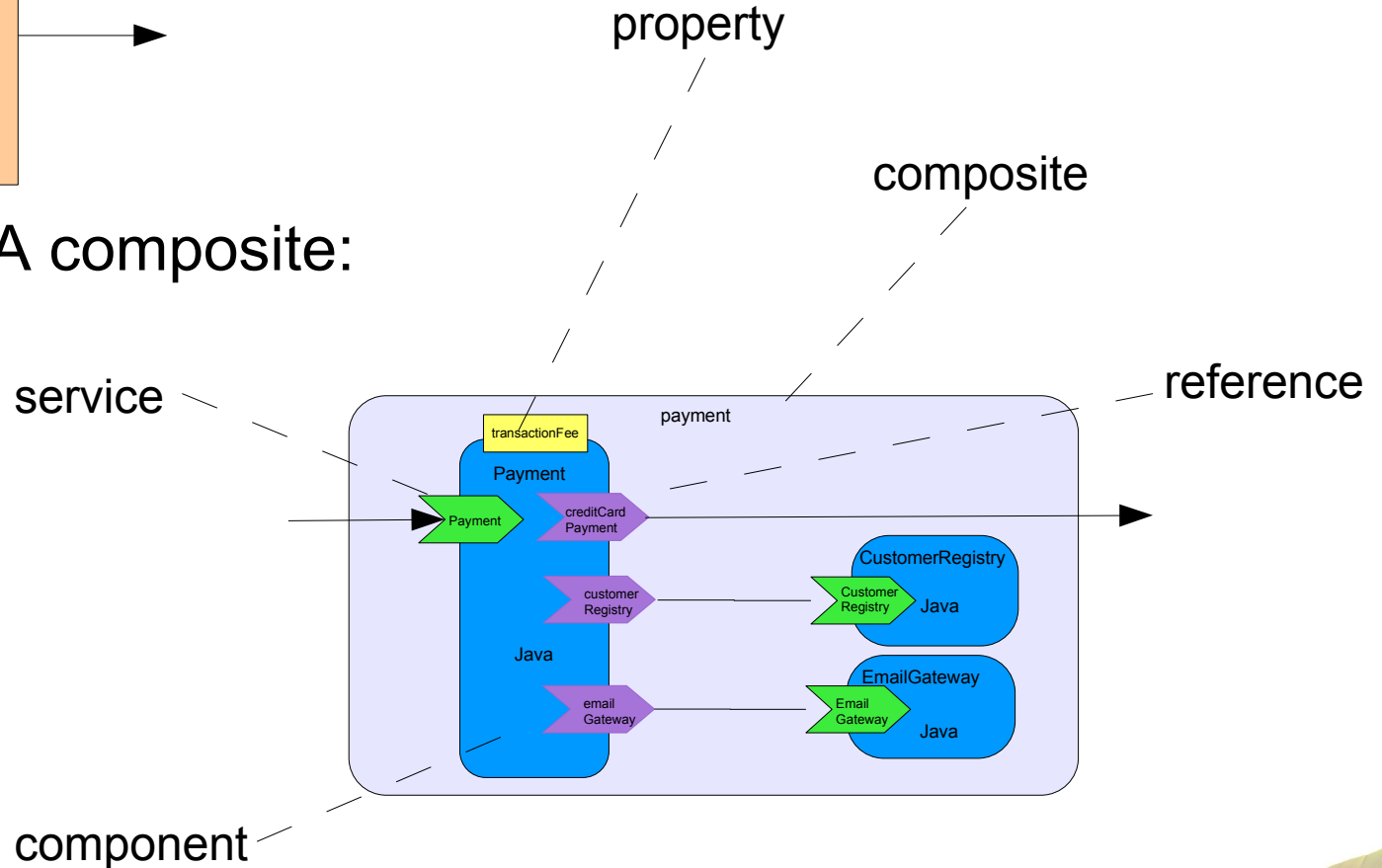
# TuscanySCATours App on a Napkin



# Map to SCA composites and components



As an SCA composite:



# Prototype and Fill in the Blanks

- **Very easy to create simple components, wire them together and get them running**
- **Doing this with simple implementations allows you to get a feel for the app without implementing all of the moving parts in detail**
- **A quick way to prototype an app and define the important components and their service interfaces**
- **The prototype components can then be distributed to a team for detailed implementation and testing**

# Payment Java Component Implementation

```
@Service(Payment.class)
public class PaymentImpl implements Payment {

    @Reference
    protected CustomerRegistry customerRegistry;

    @Reference
    protected CreditCardPayment creditCardPayment;

    @Reference
    protected EmailGateway emailGateway;

    @Property
    protected float transactionFee = 0.01f;

    public String makePaymentMember(String customerId, float amount) {
        try {
            Customer customer = customerRegistry.getCustomer(customerId);
            String status = creditCardPayment.authorize(customer.getCreditCard(), amount + transactionFee);
            emailGateway.sendEmail("order@tuscanyscatours.com",
                customer.getEmail(),
                "Status for your payment",
                customer + " >>> Status = " + status);

            return status;
        } catch (CustomerNotFoundException ex) {
            return "Payment failed due to " + ex.getMessage();
        } catch (AuthorizeFault_Exception e) {
            return e.getFaultInfo().getErrorCode();
        } catch (Throwable t) {
            return "Payment failed due to system error " + t.getMessage();
        }
    }
}
```



# Payment Composite

```
<composite xmlns="http://www.oxa.org/xmlns/sca/1.0"
  targetNamespace="http://tuscanyscatours.com/"
  name="payment">

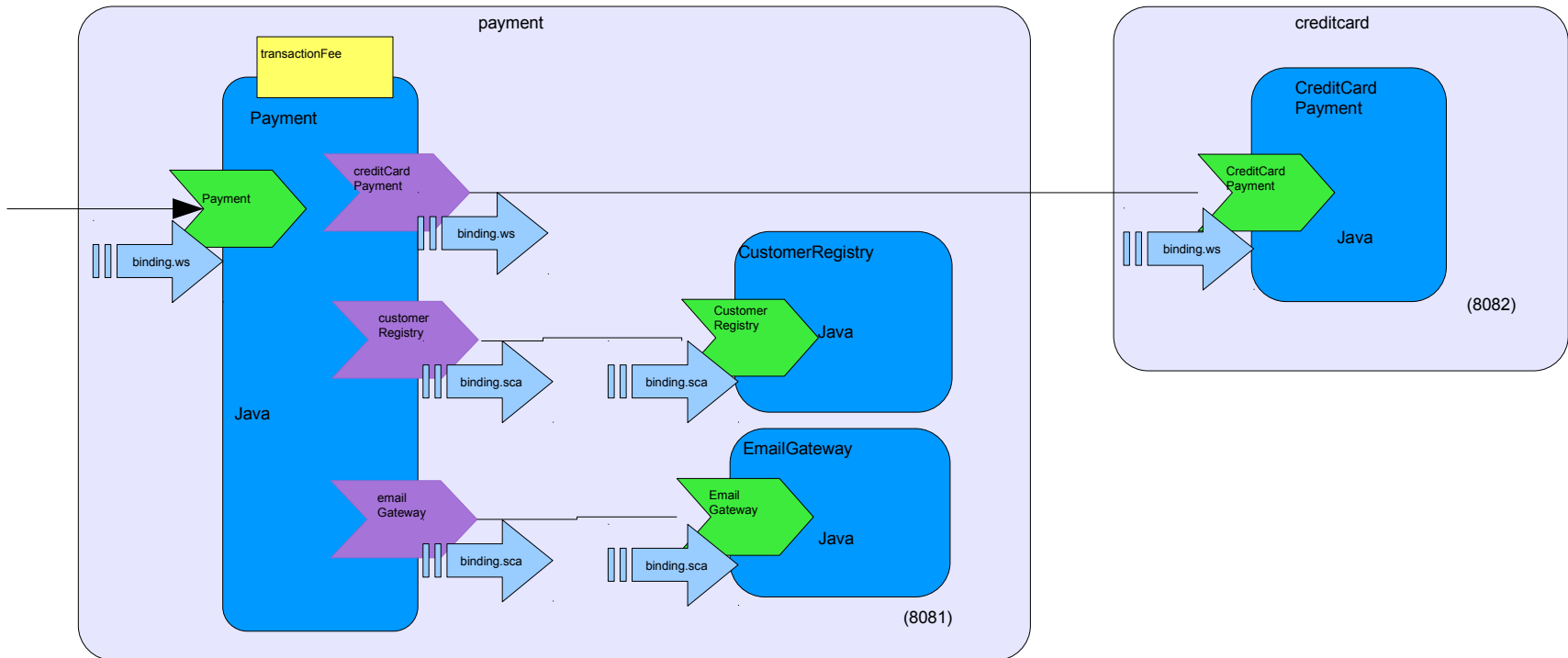
  <component name="Payment">
    <implementation.java class="com.tuscanyscatours.payment.impl.PaymentImpl" />
    <service name="Payment"/>
    <reference name="customerRegistry" target="CustomerRegistry" />
    <reference name="creditCardPayment" />
    <reference name="emailGateway" target="EmailGateway" />
    <property name="transactionFee">0.02</property>
  </component>

  <component name="CustomerRegistry">
    <implementation.java class="com.tuscanyscatours.customer.impl.CustomerRegistryImpl" />
  </component>

  <component name="EmailGateway">
    <implementation.java class="com.tuscanyscatours.emailgateway.impl.EmailGatewayImpl" />
  </component>

</composite>
```

# SCA and WS Bindings



# SCA Payment Composite – with WS Bindings

```
<composite xmlns="http://www.oxa.org/xmlns/sca/1.0"
  targetNamespace="http://tuscanyscatours.com/"
  name="payment">

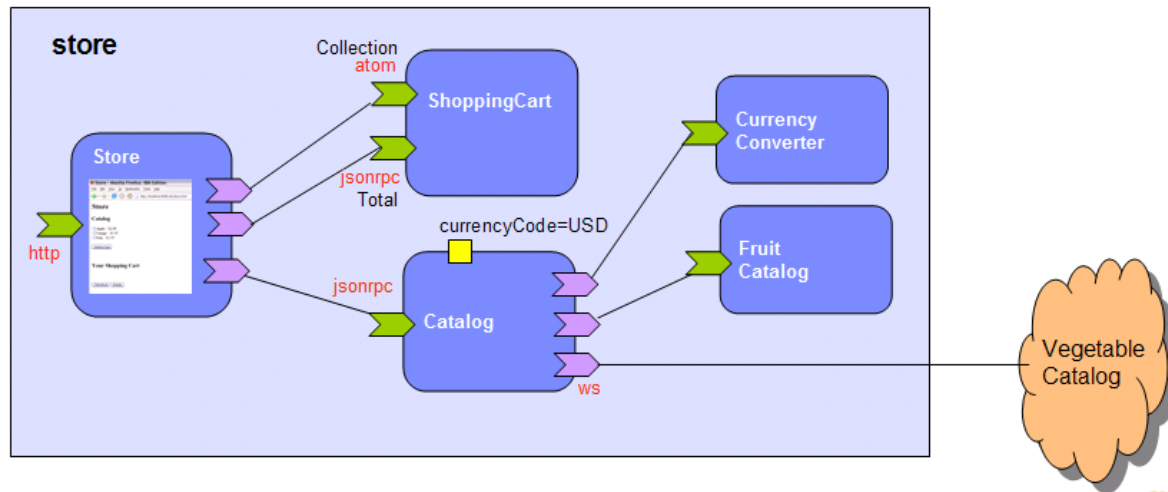
  <component name="Payment">
    <implementation.java class="com.tuscanyscatours.payment.impl.PaymentImpl" />
    <service name="Payment">
      <binding.ws uri="http://localhost:8081/Payment" />
    </service>
    <reference name="customerRegistry" target="CustomerRegistry" />
    <reference name="creditCardPayment">
      <binding.ws uri="http://localhost:8082/CreditCardPayment" />
    </reference>
    <reference name="emailGateway" target="EmailGateway" />
    <property name="transactionFee">0.02</property>
  </component>

  <component name="CustomerRegistry">
    <implementation.java class="com.tuscanyscatours.customer.impl.CustomerRegistryImpl" />
  </component>

  <component name="EmailGateway">
    <implementation.java class="com.tuscanyscatours.emailgateway.impl.EmailGatewayImpl" />
  </component>
</composite>
```

# Web 2.0 Bindings and Widget Components

- Web 2.0 bindings: REST, JSON, JSONRPC, DWR, Feeds (ATOM, RSS)
- Tuscany Widget implementation representing Web components, with Javascript dependency injection
- Other scripting implementation types



# SCA Payment Composite – JSON-RPC Bindings

```
<composite xmlns="http://www.oxa.org/xmlns/sca/1.0"
  targetNamespace="http://tuscanyscatours.com/"
  name="payment">

  <component name="Payment">
    <implementation.java class="com.tuscanyscatours.payment.impl.PaymentImpl" />
    <service name="Payment">
      <binding.jsonrpc uri="http://localhost:8081/Payment" />
    </service>
    <reference name="customerRegistry" target="CustomerRegistry" />
    <reference name="creditCardPayment">
      <binding.jsonrpc uri="http://localhost:8082/CreditCardPayment" />
    </reference>
    <reference name="emailGateway" target="EmailGateway" />
    <property name="transactionFee">0.02</property>
  </component>

  <component name="CustomerRegistry">
    <implementation.java class="com.tuscanyscatours.customer.impl.CustomerRegistryImpl" />
  </component>

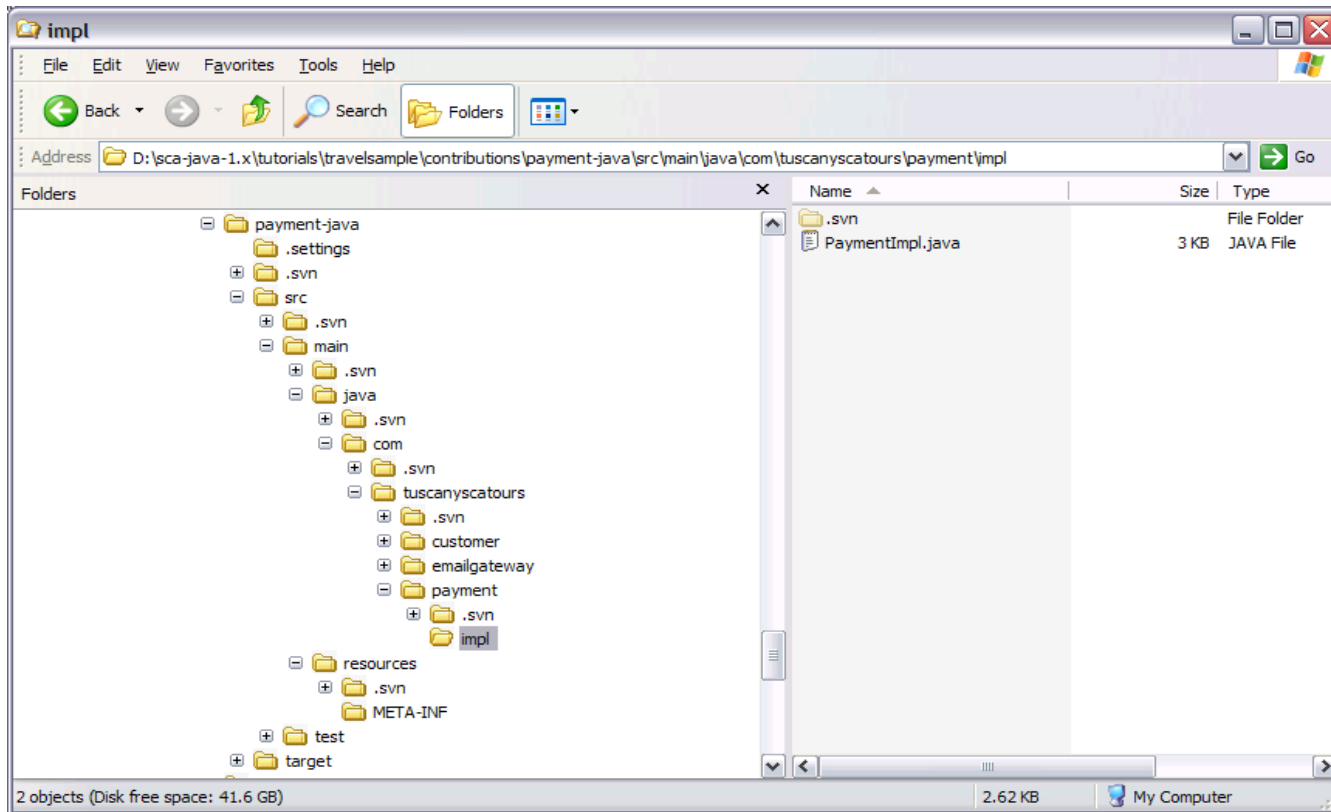
  <component name="EmailGateway">
    <implementation.java class="com.tuscanyscatours.emailgateway.impl.EmailGatewayImpl" />
  </component>
</composite>
```

# Packaging and Deployment

- **Service implementations are deployed into an **SCA Domain****
  - Represents the SCA runtime configuration
  - In general heterogeneous with distributed **SCA runtime Nodes**.
  - Defines the scope of what can be connected by SCA Wires
- **SCA Domain configuration is a **Domain Composite****
  - Final configuration for service dependencies, properties, bindings, policies
- **Implementation artifacts and their configuration added to a Domain as **Contributions****
  - Many packaging formats (JAR, ZIP, Folder etc.)
  - Artifacts (Classes, XSD, WSDL, BPEL etc.) may be shared between Contributions

# Building Contributions with Maven

- **Using Maven project layout, place composites, component implementations, interfaces and other required artifacts together**



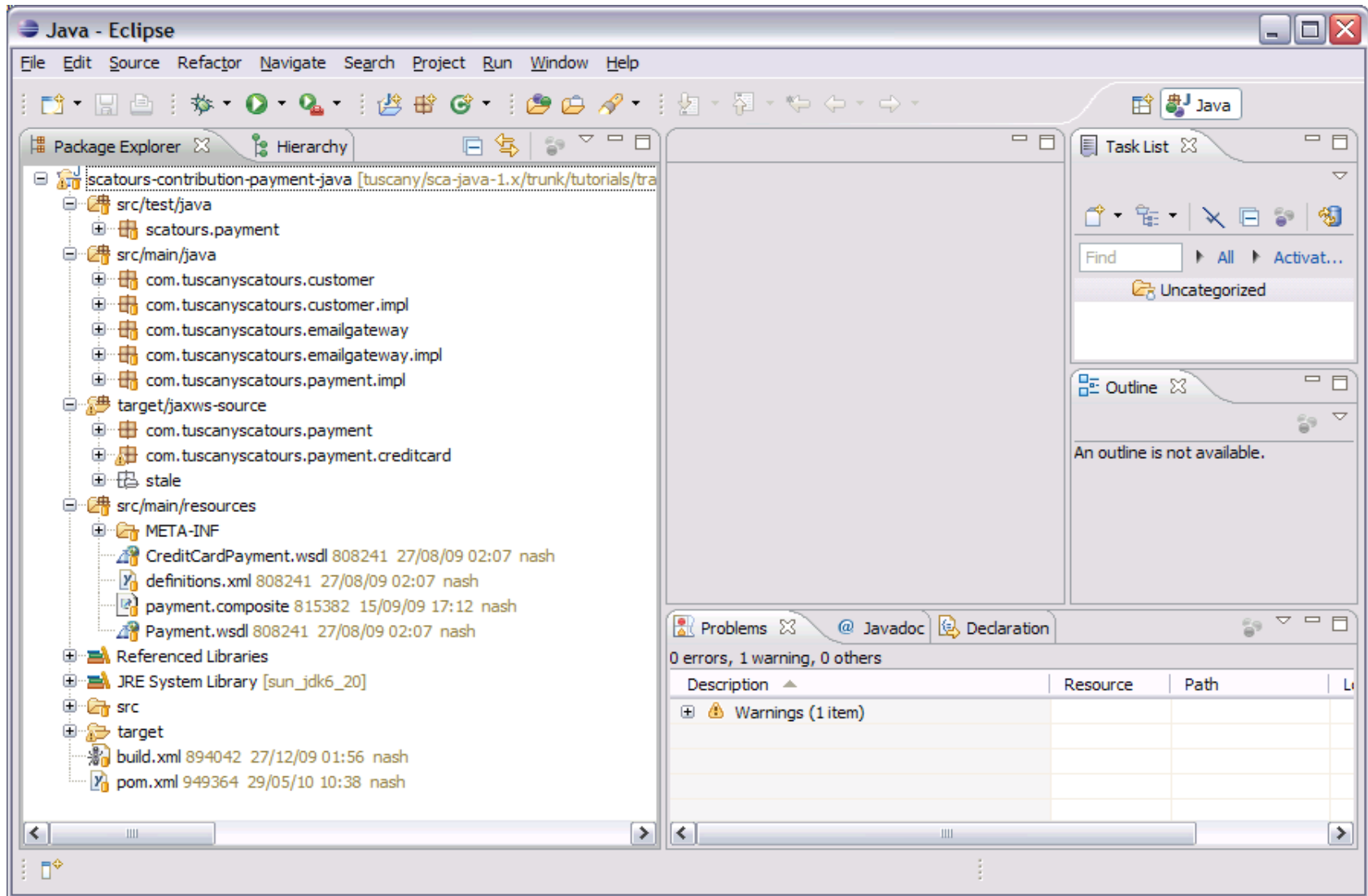
- **Use `mvn eclipse:eclipse` to build an Eclipse project**

# Developing in Eclipse

- **To import an SCA contribution built using Maven into Eclipse**
  - Set M2\_REPO
  - Import your contribution project previously built using mvn eclipse:eclipse
  
- **If you're building an SCA contribution from scratch in Eclipse, add a dependency on the Tuscany runtime Jars**
  - An easy way to do this is to create an Eclipse user defined library that includes all the Tuscany Jars and dependencies, and add this library to your new contribution



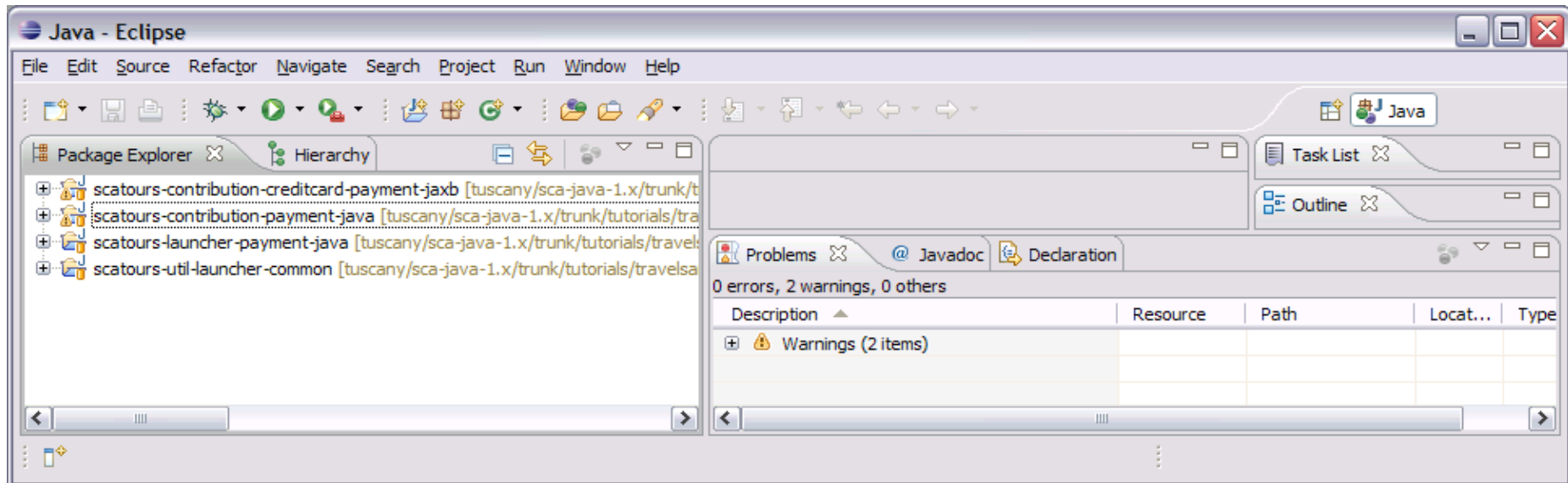
# Payment-java Contribution in Eclipse



# Running in a Single JVM

## ➤ To test payment-java we need three more things

- The creditcard-payment-jaxb contribution. This contains the composite that defines the CreditCardPayment service that the Payment component references
- The launcher-payment-java project that loads these two contributions into Tuscany and then calls the Payment component
- The util-launcher-common project which contains a few utilities that we wrote for the TuscanySCATours application



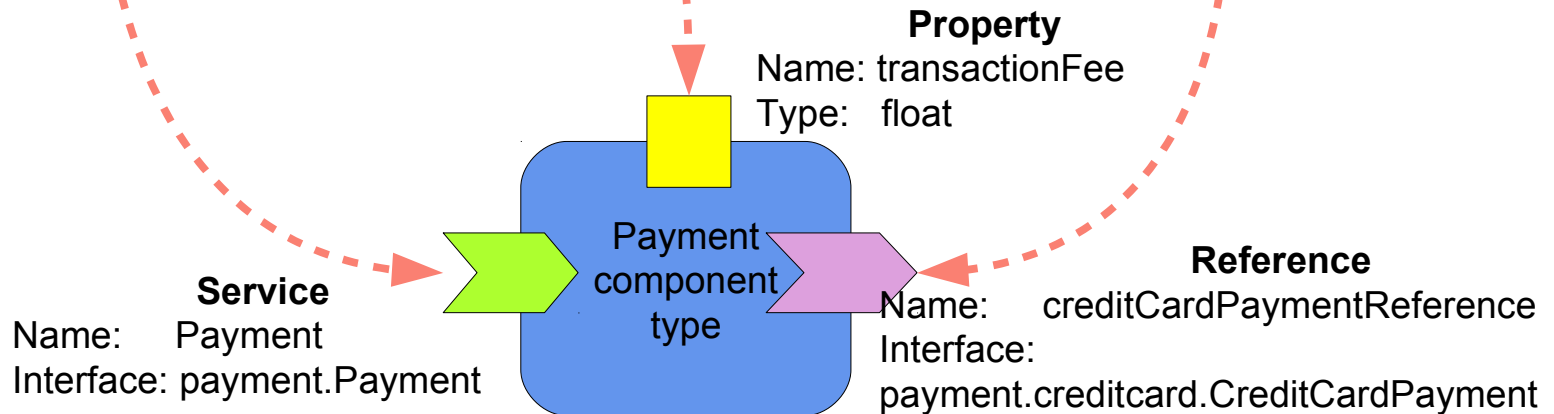
# Embedding Tuscany

```
public class PaymentLauncher {  
  
    public static void main(String[] args) throws Exception {  
        SCANode node = SCANodeFactory.newInstance().createSCANode(null,  
                                                                    locate("payment-java"),  
                                                                    locate("creditcard-payment-jaxb"));  
  
        node.start();  
  
        SCAClient client = (SCAClient)node;  
        Payment payment = client.getService(Payment.class, "Payment");  
  
        System.out.println("Payment Java test");  
        System.out.println("\nSuccessful Payment - Status = \n\n" + payment.makePaymentMember("c-0", 100.00f));  
        System.out.println("\n\nFailed Payment - Status = \n\n" + payment.makePaymentMember("c-1", 100.00f));  
  
        node.stop();  
    }  
}
```

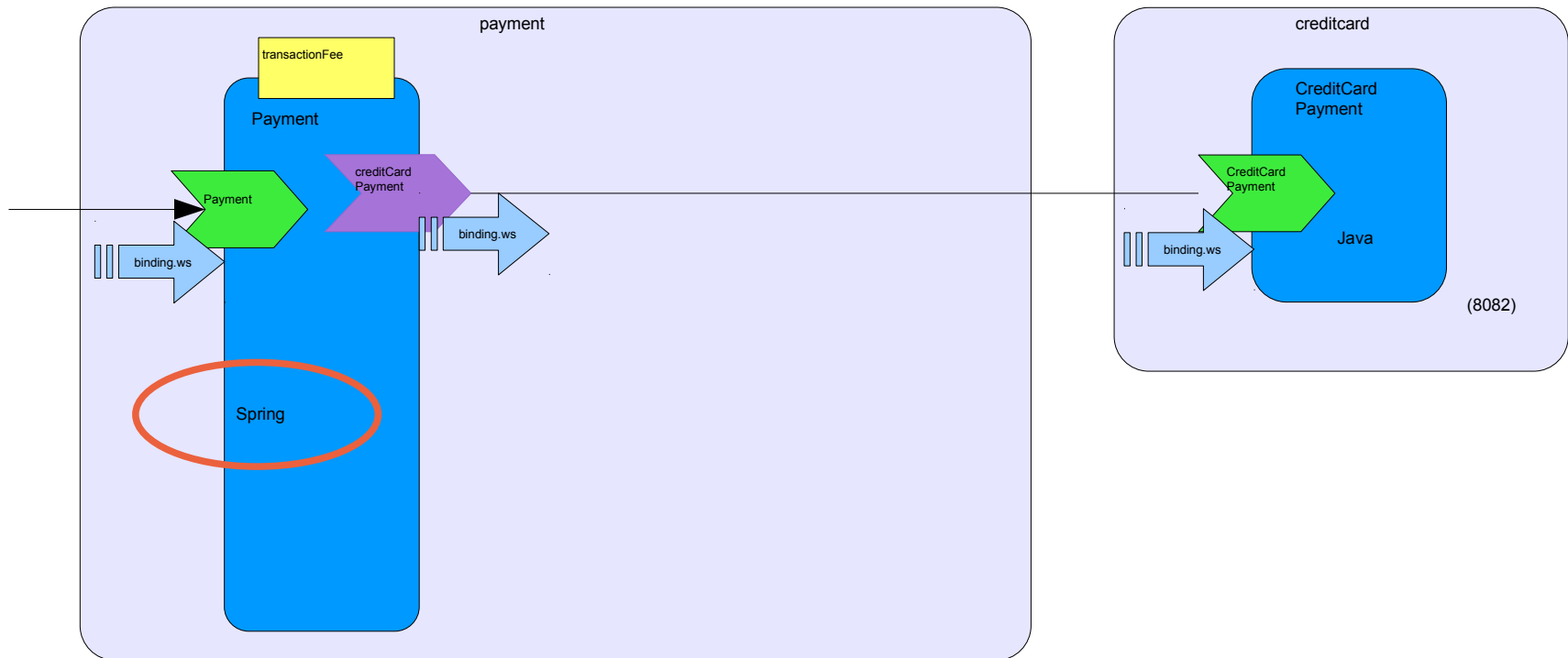
The locate() operation is a utility we created for this sample which simply locates a contribution in the sample directory structure and returns its location.

# Spring Bean Component Implementation

```
<bean id="Payment"  
      class="com.tuscanyscatours.payment.impl.PaymentImpl">  
  
  <property name="creditCardPayment"  
           ref="creditCardPaymentReference"/>  
  
  <property name="transactionFee"  
           value="0.5f"/>  
  
</bean>
```



# Payment Spring Component



# Payment Spring Context

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:sca="http://www.springframework.org/schema/sca"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
                           http://www.springframework.org/schema/beans/spring-beans.xsd">

    <bean id="Payment" class="com.tuscanyscatours.payment.impl.PaymentImpl">
        <property name="creditCardPayment" ref="creditCardPaymentReference"/>
        <property name="emailGateway" ref="EmailGateway"/>
        <property name="customerRegistry" ref="CustomerRegistry"/>
        <property name="transactionFee" value="0.5f"/>
    </bean>

    <bean id="CustomerRegistry"
class="com.tuscanyscatours.customer.impl.CustomerRegistryImpl">
    </bean>

    <bean id="EmailGateway"
class="com.tuscanyscatours.emailgateway.impl.EmailGatewayImpl">
    </bean>

</beans>
```

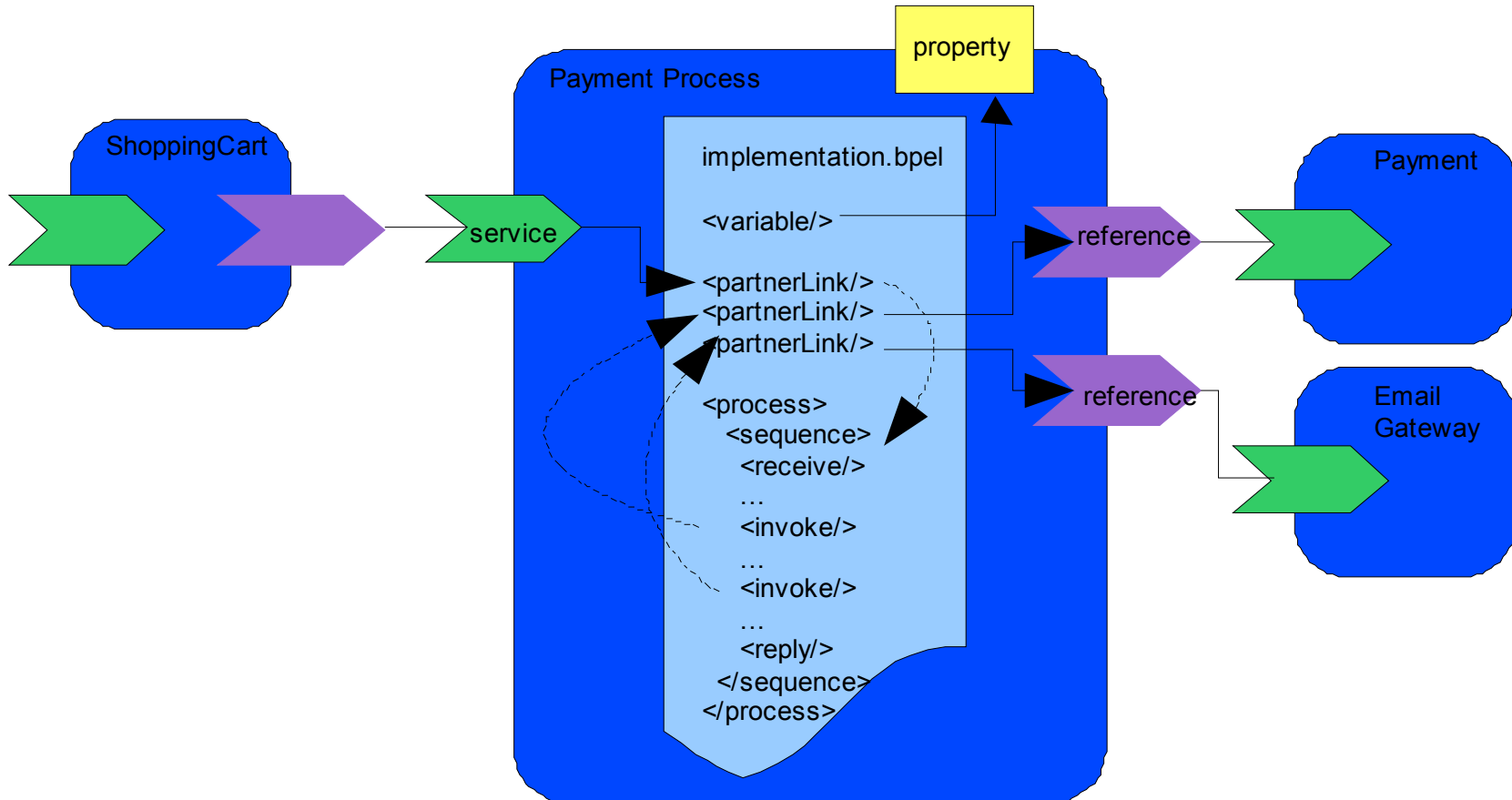
# Payment Composite

```
<composite xmlns="http://www.oxa.org/xmlns/sca/1.0"
  targetNamespace="http://tuscanyscatours.com/"
  name="payment">

  <component name="Payment">
    <implementation.spring location="Payment-context.xml"/>
    <service name="Payment">
      <binding.ws uri="http://localhost:8081/Payment"/>
    </service>
    <reference name="creditCardPaymentReference">
      <binding.ws uri="http://localhost:8082/CreditCardPayment"/>
    </reference>
    <property name="transactionFee">1.23</property>
  </component>

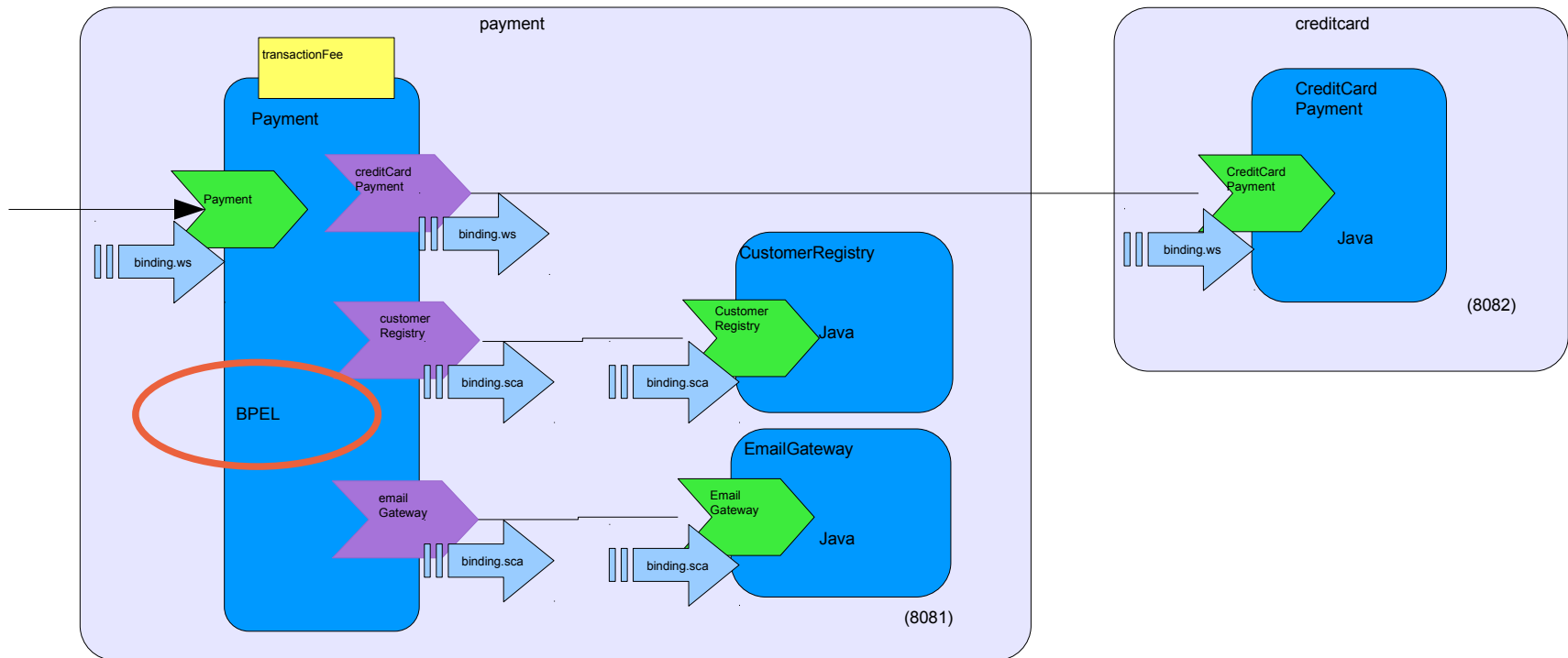
</composite>
```

# BPEL Process Component Implementation





# Payment BPEL Process Component



# Payment BPEL process

```
<process name="Payment" targetNamespace="http://www.tuscanyscatours.com/Payment" ...>

  <import location="Payment.wsdl"
    importType="http://schemas.xmlsoap.org/wsdl/"
    namespace="http://www.tuscanyscatours.com/Payment/" />
  <import location="CreditCardPayment.wsdl"
    importType="http://schemas.xmlsoap.org/wsdl/"
    namespace="http://www.tuscanyscatours.com/CreditCardPayment/" />
  <import location="EmailGateway.wsdl"
    importType="http://schemas.xmlsoap.org/wsdl/"
    namespace="http://www.tuscanyscatours.com/EmailGateway/" />

  <partnerLinks>
    <partnerLink name="paymentPartnerLink" partnerLinkType="pp:PaymentLinkType" myRole="forward" />
    <partnerLink name="creditCardPaymentPartnerLink" partnerLinkType="ccp:CreditCardPaymentLinkType"
      partnerRole="forward" initializePartnerRole="yes" />
    <partnerLink name="emailGatewayPartnerLink" partnerLinkType="eg:EmailGatewayLinkType"
      partnerRole="forward" initializePartnerRole="yes" />
  </partnerLinks>

  <variables>
    <variable name="makePaymentMemberRequestMessage" messageType="pp:MakePaymentMemberRequest" />
    <variable name="makePaymentMemberResponseMessage" messageType="pp:MakePaymentMemberResponse" />
    ....
  </variables>

  <sequence>
    <receive name="start"
      partnerLink="paymentPartnerLink"
      portType="pp:Payment"
      operation="makePaymentMember"
      variable="makePaymentMemberRequestMessage"
      createInstance="yes" />
  </sequence>
</process>
```

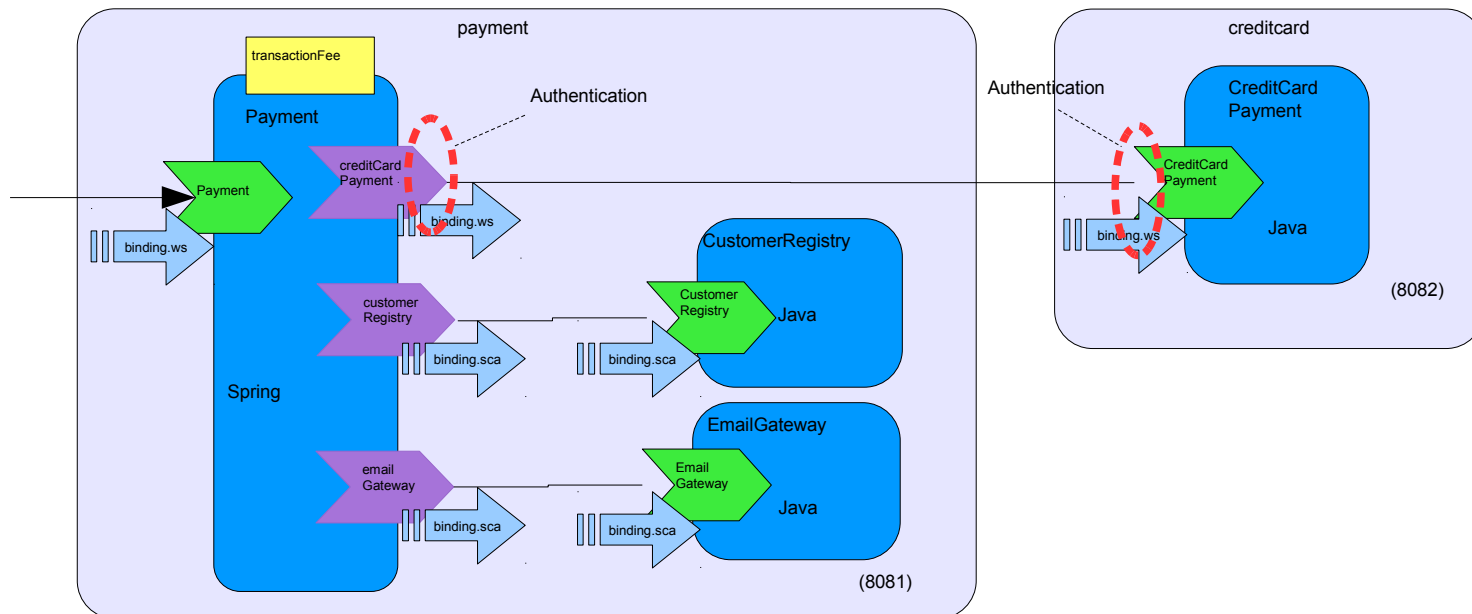
# Payment Composite

```
<composite xmlns="http://www.oxa.org/xmlns/sca/1.0"
  xmlns:t="http://tuscanyp.apache.org/xmlns/sca/1.0"
  xmlns:pp="http://www.tuscanyscatours.com/Payment"
  targetNamespace="http://www.tuscanyscatours.com/Payment"
  name="payment">

  <component name="Payment">
    <implementation.bpel process="pp:Payment"/>
    <service name="paymentPartnerLink">
      <interface.wsdl interface="http://www.tuscanyscatours.com/Payment/#wsdl.interface(Payment)" />
      <binding.ws uri="http://localhost:8080/Payment"
        wsdlElement="http://www.tuscanyscatours.com/Payment/#wsdl.service(PaymentService)"/>
    </service>
    <reference name="creditCardPaymentPartnerLink">
      <binding.ws uri="http://localhost:8082/CreditCardPayment"/>
    </reference>
    <reference name="emailGatewayPartnerLink">
      <binding.ws uri="http://localhost:8088/EmailGateway"/>
    </reference>
  </component>

</composite>
```

# SCA Policies



# Adding Policy Intents

```
<component name="Payment">
  <implementation.spring location="Payment-context.xml"/>
  <service name="Payment">
    <binding.ws uri="http://localhost:8081/Payment"/>
  </service>
  <reference name="creditCardPaymentReference" >
    <binding.ws uri="http://localhost:8082/CreditCardPayment" requires="authentication"/>
  </reference>
  <reference name="emailGateway" target="EmailGateway"/>
  <reference name="customerRegistry" target="CustomerRegistry"/>
  <property name="transactionFee">1.23</property>
</component>
```

```
<component name="CreditCardPayment">
  <implementation.java class="com.tuscanyscatours.payment.creditcard.impl.CreditCardPaymentImpl" />
  <service name="CreditCardPayment">
    <interface.wSDL
      interface="http://www.tuscanyscatours.com/CreditCardPayment#wsdl:interface(CreditCardPayment)" />
    <binding.ws uri="http://localhost:8082/CreditCardPayment" requires="authentication"/>
    <binding.sca/>
  </service>
</component>
```

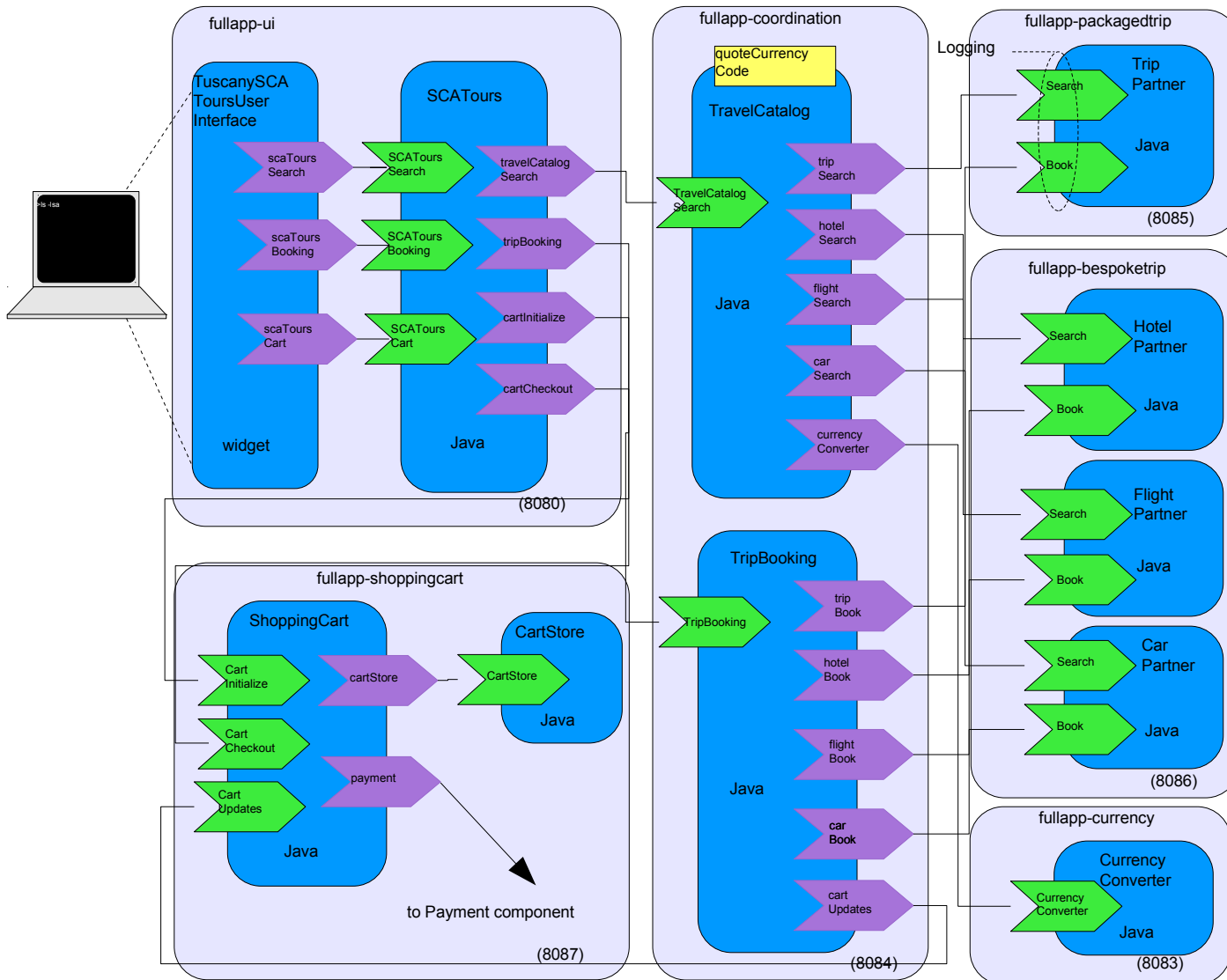
# Defining policy sets

```
<definitions xmlns="http://www.oxa.org/xmlns/sca/1.0"
  targetNamespace="http://www.oxa.org/xmlns/sca/1.0"
  xmlns:sca="http://www.oxa.org/xmlns/sca/1.0"
  xmlns:tuscany="http://tuscany.apache.org/xmlns/sca/1.0">

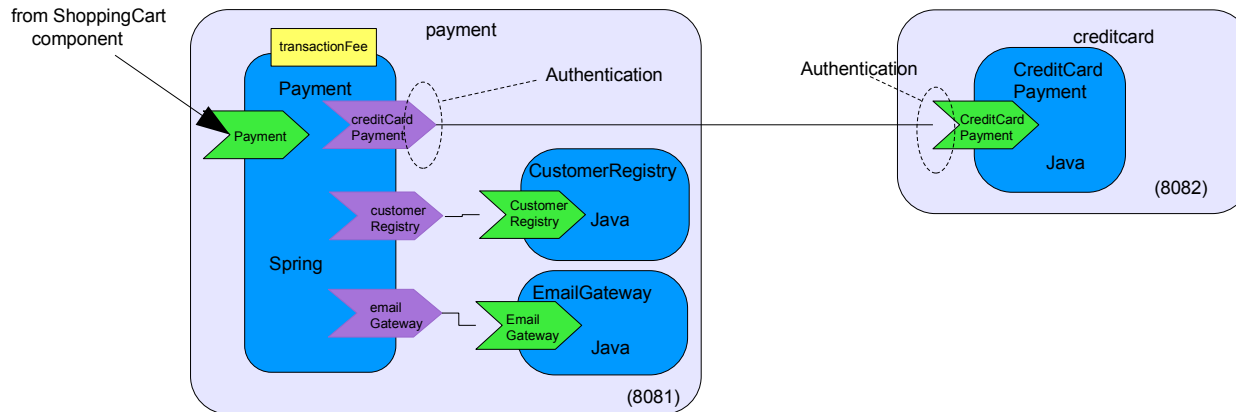
  <policySet name="BasicAuthenticationPolicySet"
    provides="authentication"
    appliesTo="sca:binding.ws">
    <tuscany:basicAuthentication>
      <tuscany:userName>myname</tuscany:userName>
      <tuscany:password>mypassword</tuscany:password>
    </tuscany:basicAuthentication>
  </policySet>

</definitions>
```

# Adding more components - 1



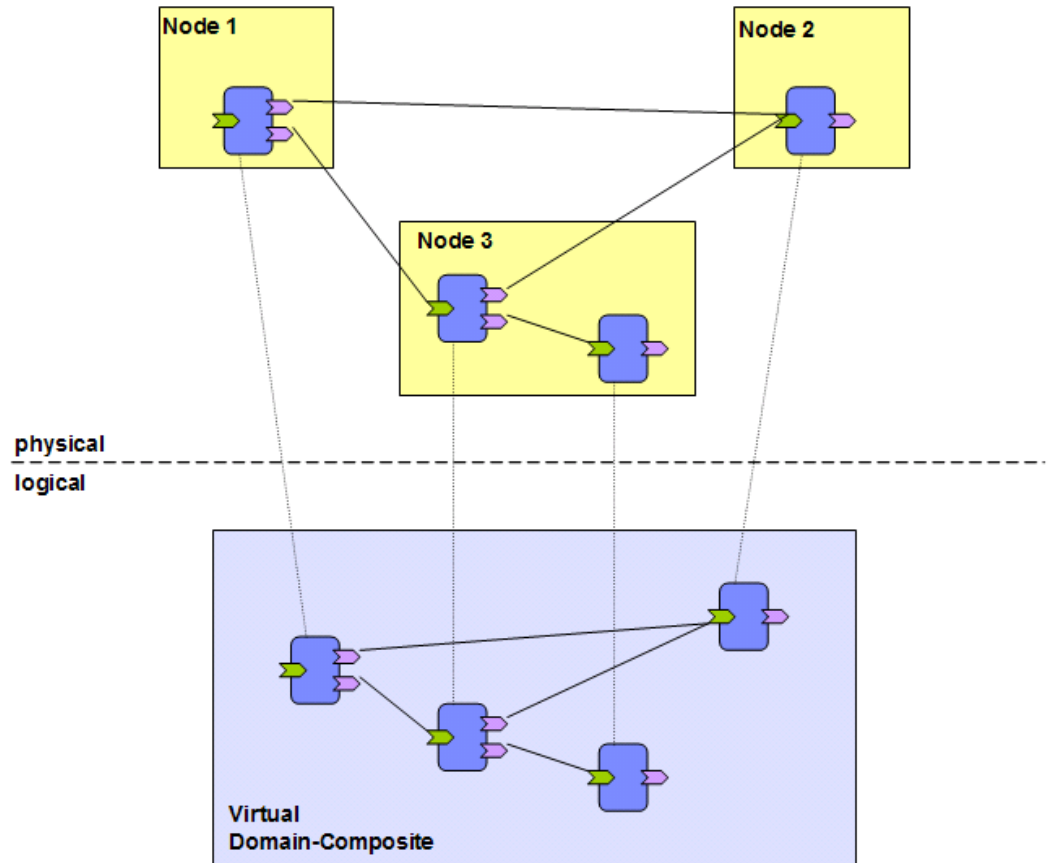
# Adding more components - 2





# SCA Domain

**A distributed deployment of the assembly.**



# Running Multiple Tuscany Nodes

```
public class FullAppNodesLauncher {

    public static void main(String[] args) throws Exception {
        SCANode nodeCreditcard =
            SCANodeFactory.newInstance().createSCANodeFromURL("http://localhost:9990/node-config/creditcard");
        nodeCreditcard.start();

        SCANode nodePayment =
            SCANodeFactory.newInstance().createSCANodeFromURL("http://localhost:9990/node-config/payment");
        nodePayment.start();

        SCANode nodeShoppingcart =
            SCANodeFactory.newInstance().createSCANodeFromURL("http://localhost:9990/node-config/shoppingcart");
        nodeShoppingcart.start();

        SCANode nodeCurrency =
            SCANodeFactory.newInstance().createSCANodeFromURL("http://localhost:9990/node-config/currency");
        nodeCurrency.start();

        SCANode nodePackagedtrip =
            SCANodeFactory.newInstance().createSCANodeFromURL("http://localhost:9990/node-config/packagedtrip");
        nodePackagedtrip.start();

        SCANode nodeBespoketrip =
            SCANodeFactory.newInstance().createSCANodeFromURL("http://localhost:9990/node-config/bespoketrip");
        nodeBespoketrip.start();

        SCANode nodeFrontend =
            SCANodeFactory.newInstance().createSCANodeFromURL("http://localhost:9990/node-config/coordination");
        nodeFrontend.start();

        SCANode nodeUI = SCANodeFactory.newInstance().createSCANodeFromURL("http://localhost:9990/node-config/ui");
        nodeUI.start();
        ...
    }
}
```

# Getting Involved with Apache Tuscany

# SCA - Resources

## ➤ **Good introduction to SCA**

- [http://www.davidchappell.com/articles/Introducing\\_SCA.pdf](http://www.davidchappell.com/articles/Introducing_SCA.pdf)

## ➤ **OASIS Open CSA – <http://www.oasis-opencsa.org>**

### ➤ **V1.1 level specs**

- <http://www.oasis-opencsa.org/sca>

### ➤ **Open CSA Technical Committees**

- <http://www.oasis-opencsa.org/committees>

## ➤ **OSOA**

- <http://osoa.org/display/Main/Home>

### ➤ **More information on that site**

- <http://osoa.org/display/Main/SCA+Resources>

# Apache Tuscany Resources

## ➤ **Apache Tuscany**

- <http://tuscany.apache.org>

## ➤ **Getting Involved**

- <http://tuscany.apache.org/getting-involved.html>

## ➤ **Tuscany SCA Java Releases**

- <http://tuscany.apache.org/sca-java-2x-releases.html>
- <http://tuscany.apache.org/sca-java-releases.html>

## ➤ **Tuscany SCA Java Documentation**

- <http://tuscany.apache.org/java-sca-documentation-menu.html>

## ➤ **Tuscany Dashboard**

- <http://tuscany.apache.org/tuscany-dashboard.html>

# Getting Involved with Apache Nuvem

# Apache Nuvem Resources

## ➤ Apache Nuvem

- <http://incubator.apache.org/nuvem/>

## ➤ Getting Involved

- <http://incubator.apache.org/nuvem/nuvem-getting-involved.html>

Thank You !!!

