An Oracle White Paper
August 2009

Oracle SCA – The Power of the Composite

Author: Pat Shepherd; Oracle Certified Enterprise Architect

Introduction

This whitepaper introduces the Service Component Architecture standard (SCA) and the SCA capabilities featured in Oracle SOA Suite 11g which are aimed squarely at reducing the complexity of developing, deploying and managing today's sophisticated SOA applications.

Overview – IT Complexity Motivates SCA Need

The goal of Service Component Architecture (SCA), simply stated, is to reduce IT complexity through a standardized framework for assembling disparate enterprise Service Oriented Architecture (SOA) components into a higher-level composite, thus simplifying development, deployment and management of enterprise applications.

Standards such as SOA, Java Connector Architecture (JCA), Business Process Execution Language (BPEL), and Web services have gone a long way towards standardizing some fundamental aspects of application development. Mature SOA implementations usually are built with combinations of these technologies and others, such as rules engines and enterprise service buses. Any functionality built with them becomes a component of larger IT sub-systems, systems, and architectures.

However, as the number of technologies increases, so does the number of separate tools and skills typically required to build, deploy and manage them. Because each tool produces its own metadata, an architect might well throw up his/her hands and say "How do I make this all work together. The complexity of these separate tools, separate metadata, and separate deployment paradigms is a real challenge."

This paper is intended for technical-minded people who want to better understand SCA, how it works and the value it provides. There is plenty of documentation out there about the details of the SCA standard. This paper will not rehash that information, nor will it offer a tutorial on building SCA applications -- though it will illustrate the basic steps as foundational knowledge. Instead, this paper will show how and why SCA simplifies development, deployment, and management of applications and, ultimately, why Oracle SOA Suite 11g is so well suited for developing your systems.

For more information about SCA, please see the "Helpful Links and Information" section at the end of this paper.

What is SCA

The problem of complexity is very much an impediment to rapid application delivery. Enter SCA, a specification that defines how the various enterprise pieces are created and assembled

together as modular components to, ultimately, increase IT sustainability and flexibility. SCA started out as collaboration between several vendors, including Oracle, under the auspices of the Open SOA group. SCA has now moved to the OASIS body (http://www.oasis-opencsa.org/sca) who has now taken over future development and stewardship.

SCA provides the Power of the Composite, a framework for standardizing and simplifying the building, deployment and management of atomic service components into a more granular composite. SCA is analogous to an integrated circuit, replete with a circuit board, individual capacitors, resistors, etc.., and wires coming out of it for integration into a larger system such as a stereo or computer.

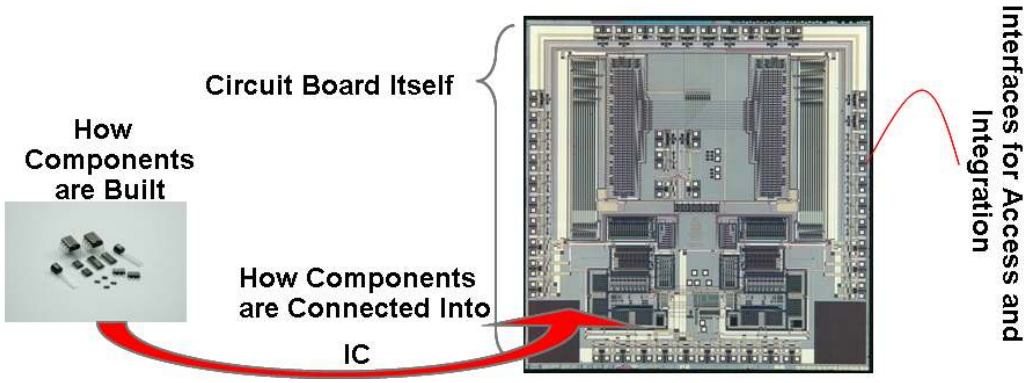

Figure 1 - SCA and IC Analogy

In this analogy, SCA is the plan for how to build the individual electrical components so they can snap on a circuit board, what the circuit board should look like, and tooling for combining, deploying and monitoring components, circuit boards and the assembled appliance.

Whether you believe in SOA in the large (Enterprise SOA), or SOA in the small (using Web services for simpler connection between systems), the missing piece is a standard that wraps the individual services into a highe- level composite service.

Think about an application such as a Purchase Order Processing system through which your company accepts orders from many sources (file, web, etc.). The credit card status for the customer is validated and if the credit card is good, the order continues. An order for a large purchase price requires a manual approval step. Finally, the order is written to a text file to be processed by the fulfillment house. For approved orders, the order is sent to the fulfillment carrier. In the "old" way of doing things you would likely open up several tools to build this system consisting of Web services, system orchestration, human involvement, business rules, and adapters.

With Oracle SCA, you can build all of these components in JDeveloper and wire them together into a composite application with SCA as the unifying framework. The metadata describing these components are wrapped in the unifying composite.xml file and deployed to the Metadata Store (MDS) that backs your application (and hence your composites) at design-time as well as

runtime. Think of MDS as an artifact repository that is used all across the platform to share common artifacts at design-time and runtime. See Figure 9 for an example of composite.xml.

SCA Terminology and Concepts

The diagram below shows the SCA composite and the various constituent elements. The composite is the main unit of deployment which wraps together any number of components each providing an atomic portion of the business functionality required to make the composite useful.

The entry point to the composite itself is the Service which exposes a chain of one or more components (each with its own Service interface) that are bound/linked together as dependencies (known as References) through Wires. The components can be local (such as a BPEL process or JCA adapter) or they can be remote using a protocol such as Web services. Finally, the specification allows for each of the components to have properties for customization within a particular use or deployment (e.g. the directory to which a completed Purchase Order document should be written).

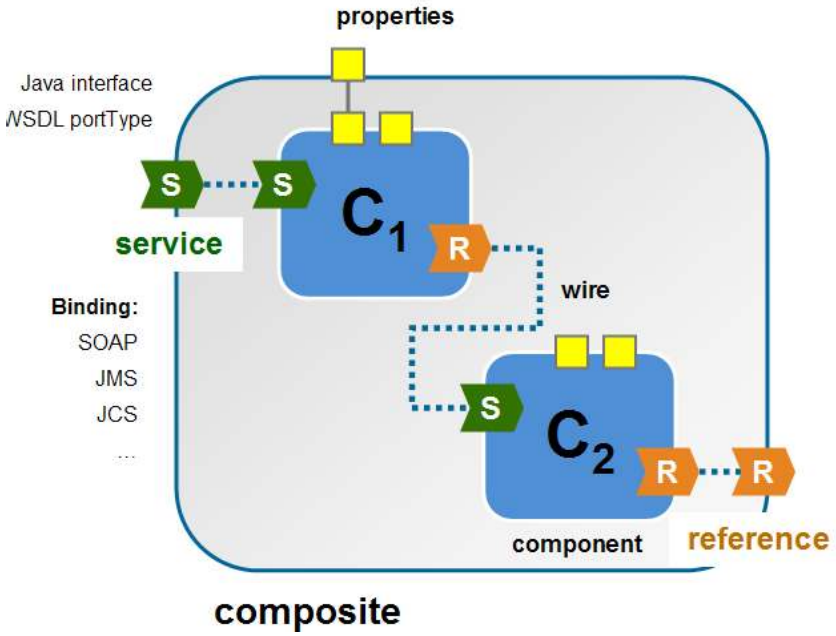

Figure 2 - Anatomy of a Composite

The following summarizes the 5 key elements of SCA shown above:

- **Composite**: deployment unit
- **Service**: entry-point into the composite. Services exposed by the component that are callable from outside the composite are called promoted services

- **Component**: provides the logic to be used within the composite
- **Reference**: refers to internal and external services. As per the SCA spec, references to external services are called "promoted references"
- **Wire**: connects services, components and references – no special semantic.
- **Properties**: allow for customization of a component's behavior in a particular deployment

There are a number of pre-defined components that you can drag-and-drop in the JDeveloper IDE such as BPEL, Business Rules, Web services, Human Task, and Adapters. In addition, Oracle's SOA Suite provides a **Mediator** component for convenience, a tool to interconnect components of varying interfaces to match impedances (using an electric circuit analogy I will extend later) between components as shown below. Uses for this are abundant where, for example, you need to decide which component to send a request to based on information in the incoming XML payload which may, in turn, require transformation to meet the interface needs of the target component. The mediator provides content based routing and transformation at the component level. Note: it is recommended that you place enterprise services on a Service Bus and use BPEL for orchestration at a higher level leveraging the Mediator for lower-level coordination across individual components and references *inside* the composite.

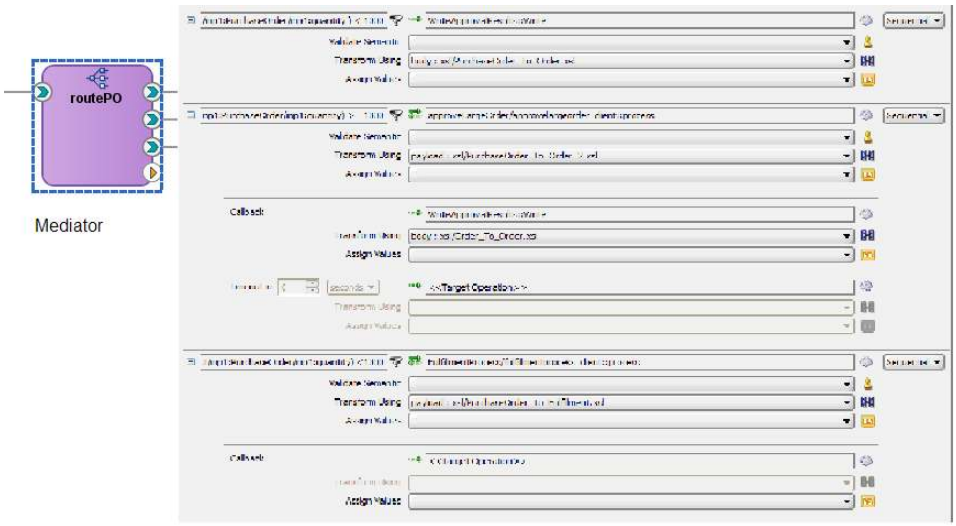

Figure 3 - Mediator Micro-orchestration

How SOA Suite 11g Leverages SCA

SOA Suite 11g benefits greatly from SCA because it fundamentally simplifies the entire application life-cycle from development through deployment and management. The IDE tooling in JDeveloper makes application development easier than ever, as I will illustrate later. Because there is now a unifying metadata model that wraps the application and its components into a

single unit (known as a composite), an application that once consisted of numerous different pieces is wrapped into the composite having a single deployment unit; a SCA Service Archive (SAR). This SAR can now be versioned as one, deployed as one, and managed as one instead of worrying about the individual pieces. Finally, management of applications benefits greatly because the flow of messages into, and through, the entire composite, and constituent components, can easily be traced as we will show later.

All of this translates into numerous tangible design-time, runtime and monitoring/management benefits.

3 Major Parts of Oracle SCA

The three major parts of SOA Suite 11g that support the development of SCA composite applications are the

1. Development tooling in JDeveloper
2. Unified server-side SOA Suite runtime infrastructure
3. Management capabilities in Enterprise Manager

The unified runtime infrastructure is provided as a set of service engines which provide the functionality for the SCA composite and its constituent elements, such as BPEL processes, Rules invocation, etc.

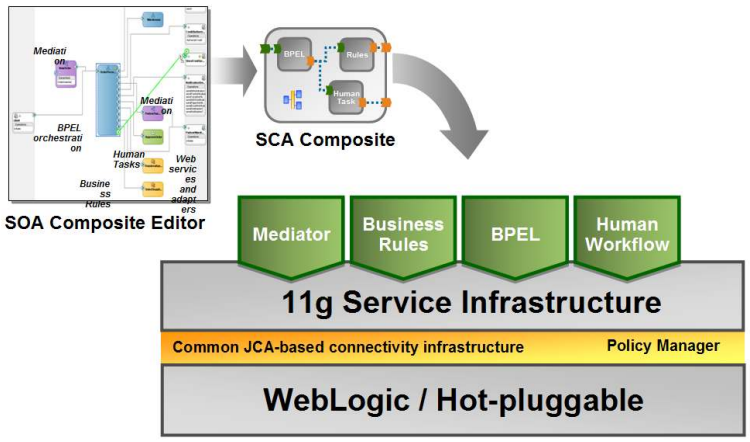

Figure 4 - Unified SCA Runtime Infrastructure

Building SCA Composites

SCA is now the fundamental framework and building blocks for your SOA applications. Gone are the days of having to open up different tools to create different types of projects (each with its own metadata to manage and lifecycle to govern). Now, the composite is the cornerstone of SOA application development. The beauty of this is that SOA Suite 11g and JDeveloper provide a much simpler approach to creating even the most complex applications.

As mentioned before, we will not belabor "how to build an application" beyond the basic steps, you can delve into the steps yourself. The assumption is that you have installed SOA Suite, have JDeveloper up and running, and know how to create WLS domains, server and database connections, etc. The following describes the high-level process to create a new SCA application.

1. Select **File > New…** from the menu and select **Generic Application** from the **Items** field

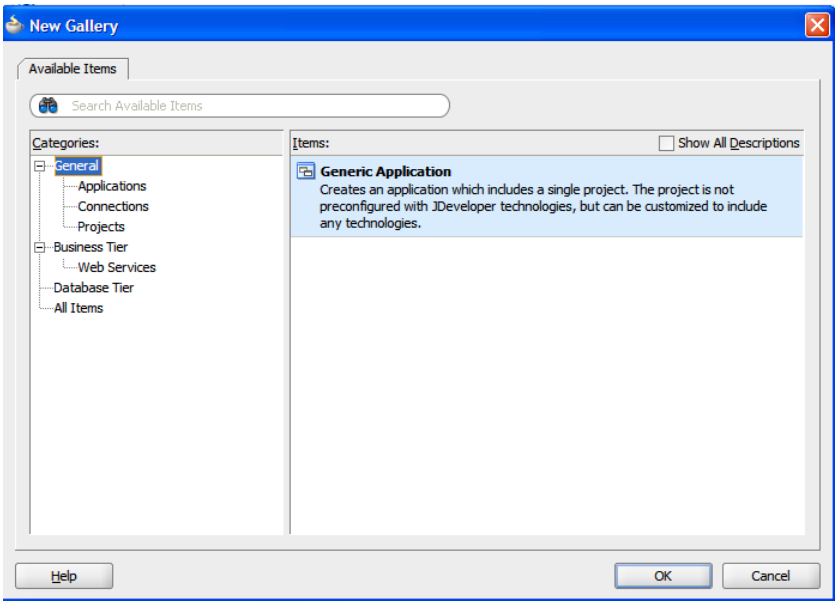

Figure 5 - New Application

2. Name your application

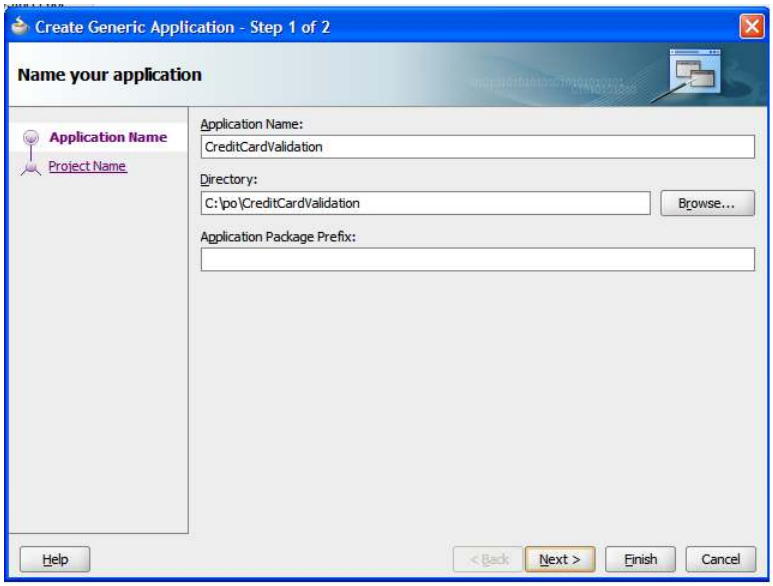

Figure 6 - Name Application

3. Select SOA as the Project Technology (very important)

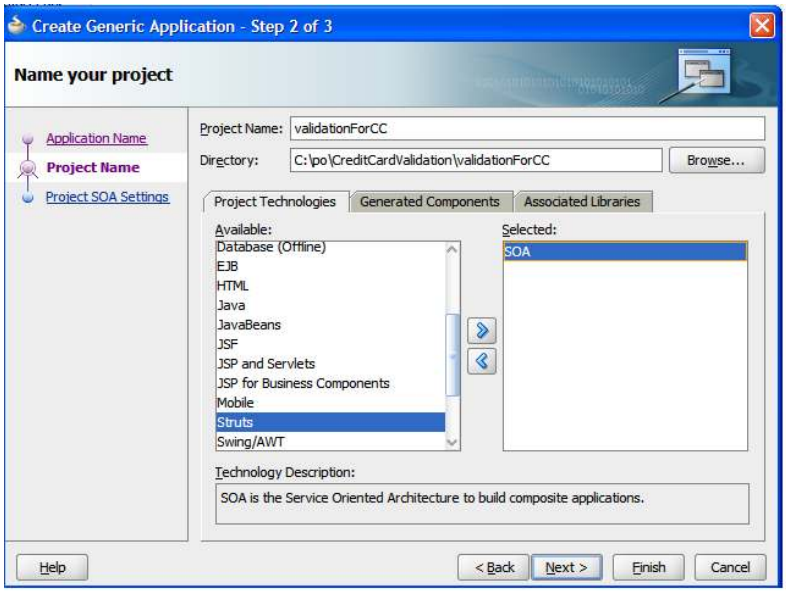

Figure 7 - Select Project Technologies

4. Select **Empty Composite** or one of the other options

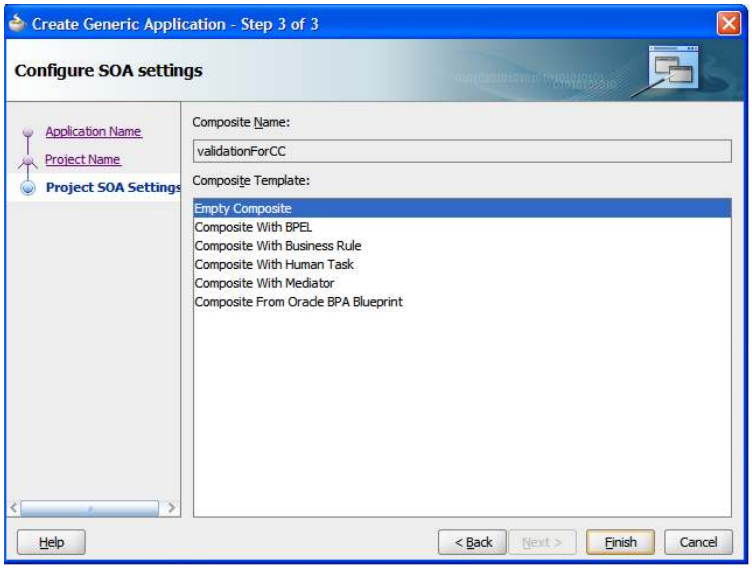

Figure 8 - Select Composite Type

You will now have an empty canvas displaying three swim lanes: services, components, and references.

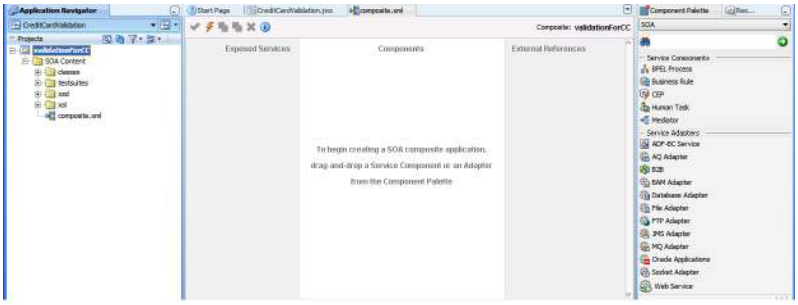

Figure 9 - Blank Composite

Figure 10 illustrates the composite.xml file generated by creating the empty composite as described above. This XML defines all the components used in the composite, the wiring between them and any added properties. Note that the application starts with a revision number of 1.0.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by Oracle SOA Modeler version 1.0 at [7/17/09 10:43 AM]. -->
<composite name="Project1"
        revision="1.0"
        label="2009-07-17_10-43-38_390"
        mode="active"
        state="on"
        xmlns="http://xmlns.oracle.com/sca/1.0"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
        xmlns:orawsp="http://schemas.oracle.com/ws/2006/01/policy"
        xmlns:ui="http://xmlns.oracle.com/soa/designer/">
</composite>
```

Figure 10 - composite.xml

When built out, an SCA composite application might resemble Figure 11 with Services (interfaces), components (BPEL, Rules, Human Tasks, and Mediator in this case), references (to adapters in this case), properties, and wires connecting everything together.

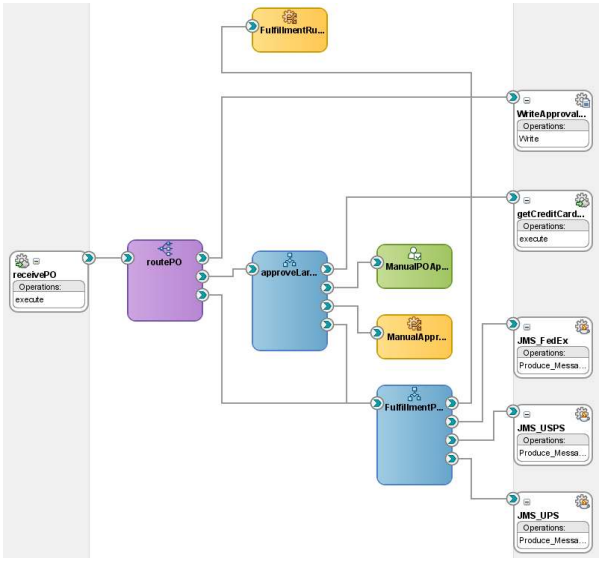

Figure 11 - Built Out Composite

The resulting composite.xml has more entries for components, references and wires, but is still a relatively simple expression of an otherwise complex SOA system.

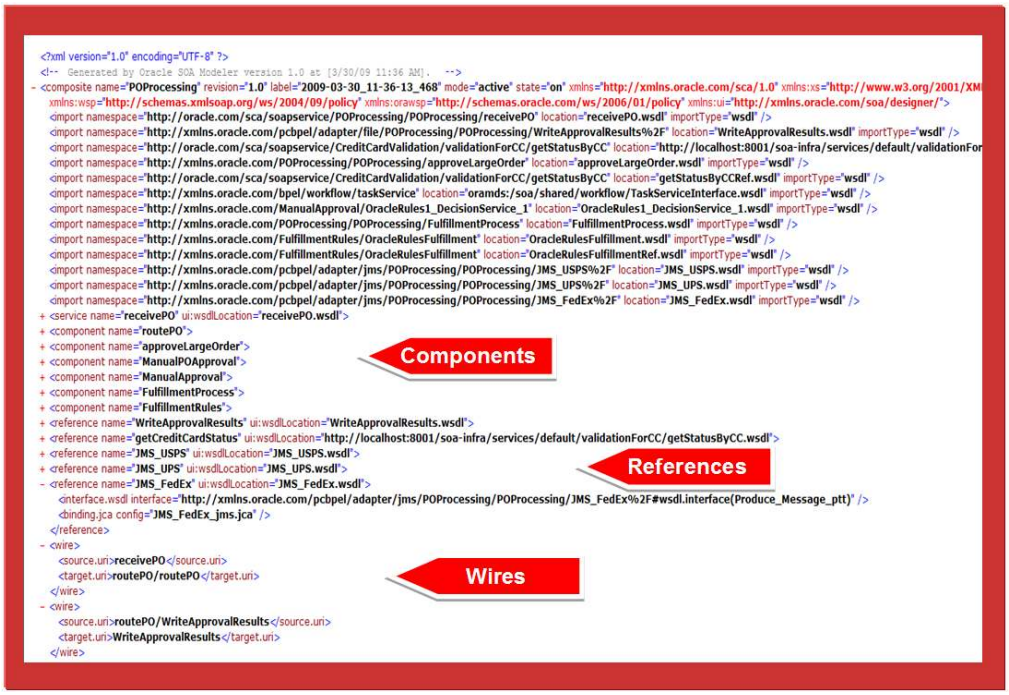

Figure 12 - composite.xml Built Out

Deploying SCA Composites

There are several ways to deploy an SCA Service Archive (SAR) into a server. The method you choose should be determined by the lifecycle (development, test, production) that you are in. The following helps depict the typical flow of an SCA composite through it's lifecycle. Generally JDeveloper is used for deploying to development servers, ANT for deploying to test, and Oracle Enterprise Manager for deploying to production as shown below.

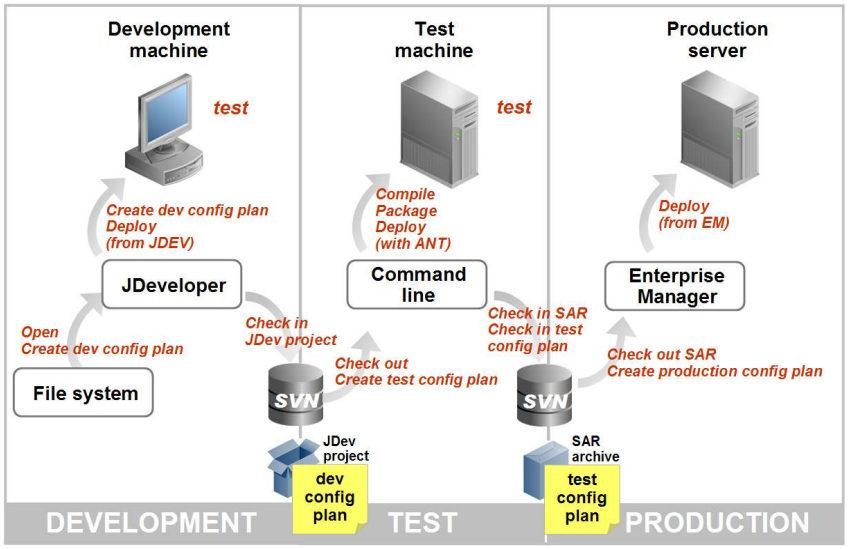

Figure 13 - Deployment Lifecycle

Deploying a composite is simple to do within JDeveloper. Just right click on the project name and select Deploy as illustrated below.

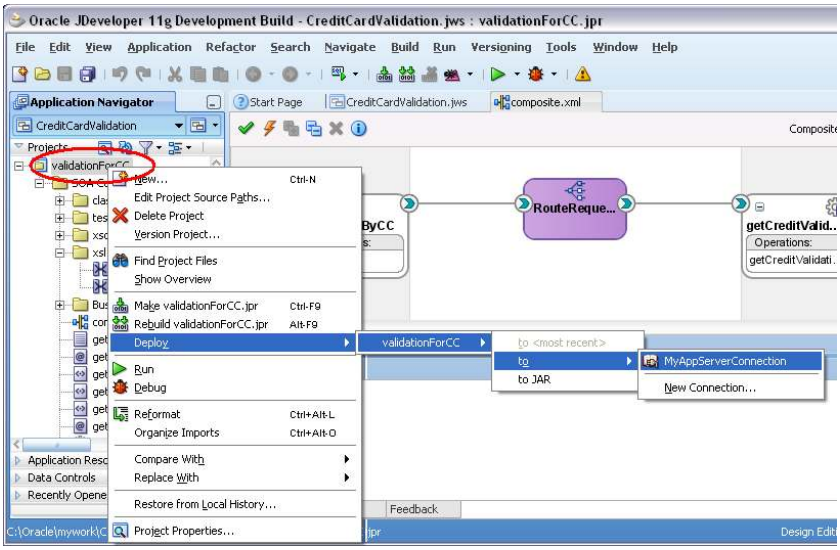

Figure 14 - Deploy from JDeveloper

JDeveloper provides some powerful options for deployment such as the ability to designate a newly created version of the composite as the default deployment which can leave the former one in place for graceful quiescence. You may also choose to overwrite the old version with the new deployment.

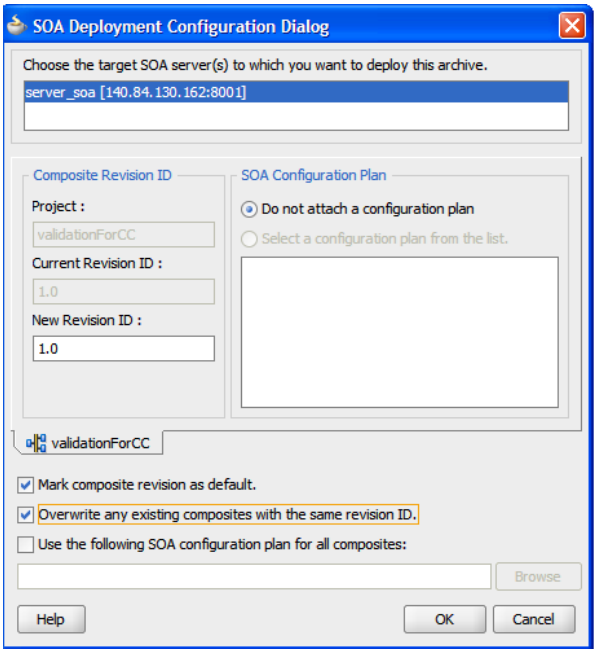

Figure 15 - Deployment Versioning and Options

Oracle SOA Siute also provides the option to create and use JSR-88 deployment plans for promoting projects between Development, Test, and Production environments. The power here is the ability to set volatile deployment properties, such as the file-adapter read/write directory and naming convention which can be search-and-replaced for deployments across lifecycle boundaries. Anyone who has had to open up JAR/EAR files to alter deployment descriptors will appreciate this feature. Below is a screen shot demonstrating how, from JDeveloper, you can identify the properties that are candidates for replacement.

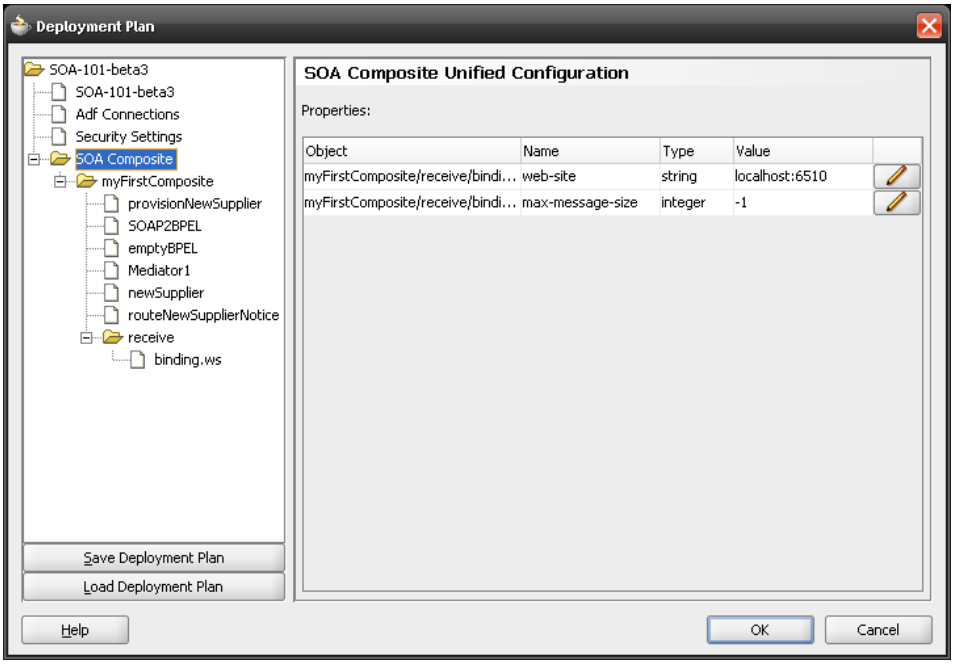

Figure 16 - JSR88 Deployment Plan from JDeveloper

Managing SCA Composites

With all this discussion of building SCA applications, it is easy to overlook the potential for dramatic improvement in the management of production applications. The power of the composite for management in Oracle SOA Suite 11g is that SCA simplifies the deployment of one unit (the SAR, as mentioned above) through which the path of any message can be monitored end-to-end all via one Oracle Enterprise Manager Console.

When a SOA composite application is invoked, a new composite instance is created. This instance is identified by a unique instance ID. You can click these IDs to access more specific details about the state of SOA composite application instances. From the Instances page, you can also monitor the state of SOA composite application instances. Figure 17 illustrates the Flow Trace screen showing the flow of a message through various composite and component interfaces, giving you an unprecedented end-to-end view of your system. This is unprecedented level of visibility into the runtime behavior across all the components that make up the composite helps to answer vexing questions, including "How is all this stuff connected?" and "Where is the customer's order anyway?"

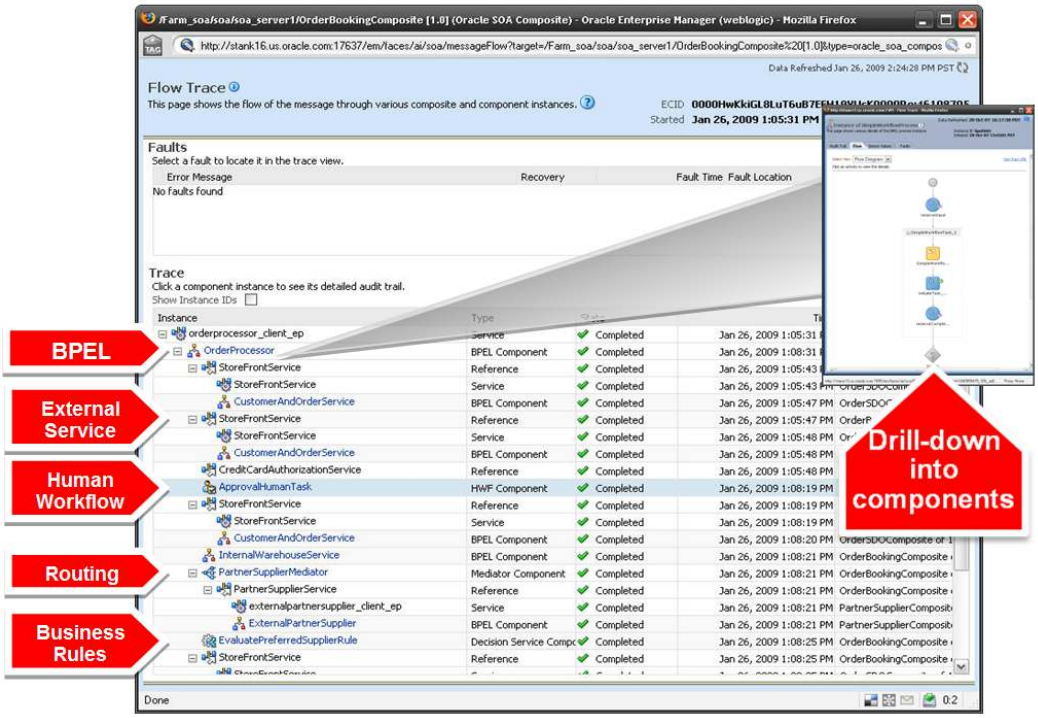

Figure 17 - Enterprise Manager SCA Trace Flow

Composite Sensors provide another useful management capability to SCA. Composite Sensors let you decide which key data fields will be tracked (such as "PO_ID") and attach a sensor to these fields in JDeveloper. Once the sensor is deployed, you can use them to search specific instances at runtime.

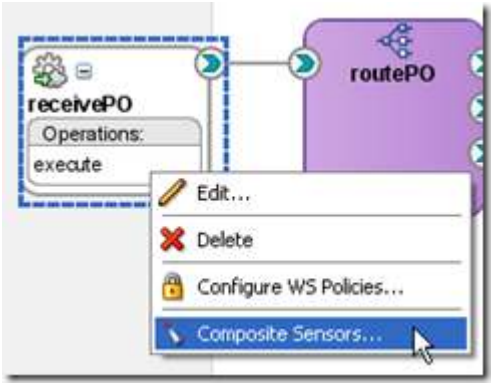

Figure 18 - Composite Sensor

Figure 19 - Search SCA Instances

Best Practices

SCA is best thought of as a complimentary to the SOA technology you are likely already using and comfortable with. Yet SCA provides a better framework for combining things like Web services and BPEL processes into higher level components.

A best practice approach is to use SCA to help facilitate both top-down and bottom-up application design and development. An architect can, for example, build out the high-level flow of the SCA application and relegate parts of the implementation to developers who have particular strengths in specific areas. The bottom-up approach allows SCA to be the glue that binds together lower-level components into a higher level of abstraction within a number of applications.

In general, all the best practices that are relevant to SOA still apply. However, questions come up about how SCA fits into SOA components/layers such as Service Bus, Business Process (Orchestration), and the like. It is important to understand the difference between Composition and Orchestration. SCA is about the **Composition** of components together into an "integrated circuit" that might well be part of a bigger application. **Orchestration** concerns the control of a process across multiple systems and people. That control involves branching, state management, and human task flows. It bears repeating that you should use a Service Bus and BPEL for service abstraction and orchestration at a higher level, leveraging the Mediator for lower-level coordination across individual components and references *inside* the composite.

The following table summarizes high-level SOA needs with the appropriate technology to use - and the Oracle product(s) that map to the technology:

SOA Need	Solution Category	Specific Oracle Product(s)
Service Virtualization – ability to establish indirection of service producer and consumer, providing the ability to change either, or to change the steps taken between request and response, minimizing the affect one has on the other while increasing service reuse	Enterprise Service Bus	Oracle Service Bus
Service Composition – ability to create, assemble, deploy and manage services based on a standardized framework encompassing a broad range of enterprise technologies	SCA	JDeveloper, SOA Suite 11g
Service Orchestration (and Human Orchestration) – control, where required, of processing and flow across multiple systems including human involvement where necessary	Business Process Management	BPEL PM
SOA Governance – management of the development and production lifecycle of SOA and SCA applications (and associated artifacts) to foster more reuse, track ROI, and gain tighter control of assets	Governance	Oracle Enterprise Registry/Repository
Enterprise Management – management of your SCA applications	Production Assurance	Oracle Enterprise Manager

Summary

SCA is a fundamentally important technology for creating modern SOA applications. The addition of composites and composite-based application through SCA is a cornerstone of the Oracle SOA Suite 11g release. The power of the composite is its ability to simplify the development, deployment and management of today's complex SOA applications.

This paper was written with two goals in mind. The first was to illustrate just how easy it is to create these SOA applications, deploy them to various environments leveraging versioning and deployment plans for maximum flexibility, and utilize the end-to-end visibility that this new approach provides from a management standpoint. The second goal was to present a high-level discussion of where SCA fits in relation to other SOA technologies and principles. It is the author's sincere hope that these goals have been met.

Helpful Links and Information

1. Understanding SCA; Jim Marino and Michael Rowley, publisher: Addison Wesley

- http://www.amazon.com/Understanding-Component-Architecture-Independent-Technology/dp/0321515080/ref=sr_1_1?ie=UTF8&s=books&qid=1248280480&sr=8-1

2. OASIS Open CSA
   - http://www.oasis-opencsa.org/sca
3. Getting Started With Oracle SOA Suite 11g R1 A Hands-On Tutorial; Heidi Buelow, Manas Deb, Jayaram Kasi, Demed LHer, Prasen Palvankar; publisher: Packt Publishing
   - http://www.amazon.com/Getting-Started-Oracle-Hands-Tutorial/dp/184719978X/ref=sr_1_2?ie=UTF8&qid=1248872691&sr=8-2
4. Oracle SOA Suite page page contains everything one would need to get started with SCA: download, docs, etc..
   - http://www.oracle.com/technology/products/soa/soasuite/index.html
5. Oracle Blogs:
   - http://blogs.oracle.com/
6. My Enterprise Architecture Blog
   - http://blogs.oracle.com/enterprisearchitecture/

White Paper Title
August 2009
Author: Pat Shepherd

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:

Oracle is committed to developing practices and products that help protect the environment