EclipseWorld 2006

Sep 6-8, 2006

Developing and Deploying Services using STP

Oisín Hurley
Karl Reti

Demos
Johnson Ma

Contents

- Historical elements of the project
- Sub-projects and contents thereof
- Summary of what's at HEAD
- Where we are intending to go next
- Community experiences
- Q's

Introduction

- Project was started to
  - serve as condensation point for existing SOA-relevant technologies
  - deliver integrated suite of tool elements that are relevant to SOA construction
- Charter is quite open in terms of project elements
  - testing, management, configuration, etc

Introduction

- The elements of the vision stretch over the full gamut of SOA construction, provisioning and testing
- Each of the sub-projects is related to a particular piece of the process of SOA
- We'll go through the projects one at a time and where possible show some stuff working

Sub-Projects

- Service Creation
- Core Models
- SOA System
- B2J
- BPMN

eclipse WORLD The Enterprise Development Conference

eclipse WORLD The Enterprise Development Conference

EclipseWorld 2006

Sep 6-8, 2006

Service Creation

Service Creation

- Being able to create services is a key aspect of SOA :-)
- The service creation project is intended to focus on the many aspects of service creation
  - working out integration with existing service creation technologies in Eclipse – c.f. WST specifically
  - developing new service creation technologies
    - using native language
    - adapting existing ways, e.g. CORBA interfaces

Service Creation

- This sub-project should also include some pieces that are peripheral but relevant
  - e.g. policy language editing
  - e.g. bindings editor framework
  - e.g. starter project/code generation
- Where possible, SC will integrate, adapt or extend what is already available.

Service Creation

- Service creation is responsible for working through any issues that arise with the integration of multifarious service creation approaches
- And it's responsible for the adaptation of these technologies to the Core SOA Model that is at the heart of STP

Contribution Highlight

- Initial contribution to this project is a set of tooling that supports JAX-WS developers in their creation of web services
- This code introduces
  - JAX-WS project *nature*
  - *incremental builder* that hosts *extension points*
    - generating WSDL from a JAX-WS interface
    - generating client code, server code and JAXB wrapper classes
- *demo ensues....*

EclipseWorld 2006

Sep 6-8, 2006

Core Models

Core

- SOA is about services, plural, and how they interact with each other
- So to make an effective set of tools, there needs to be a way to talk about multiple services and model this within the toolset

- When we set up the project, we chose to take on the SCA assembly model for this purpose
- Remember that diversity bit in the service creation? That's the main reason for choosing.

Core

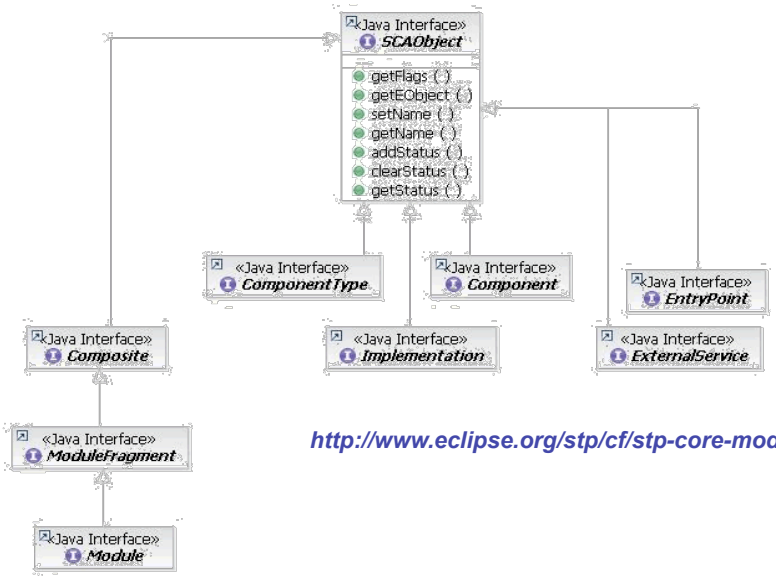
- Having a significant code contribution purposed to this area was a factor too
- When we started off with this, the SCA stuff was being developed by a small group
- This has now expanded to include lots of other companies
- Looking forward to the various parts of it being part of a ‘mainstream’ standards group

Core

- Core Model provides an API for the SCA specification that is friendly to Eclipse-based environments.
- SOA Assembly Framework (SAF)
- Core EMF Infrastructure
- Core Introspection Framework provides a framework and an example of how to construct a new ComponentType from an implementation

Core Model

- EMF model: SCA 0.95 + enhancements



http://www.eclipse.org/stp/cf/stp-core-model.html

Core Introspection Framework

- Mechanism to derive ComponentType* from an Implementation
- Translation: drills down through a set of implementation artifacts, like java files, WSDL files, etc to construct elements in the Core Model
- *Introspectors* are defined through the *org.eclipse.stp.core* extension point

** SCA 0.95 – will be updated to the latest 0.96*

Core Introspection Framework

- Sample introspector is part of the tests in this project – it consumes Java properties files
- We need to develop more introspectors to deal with the popular service creation approaches:
  - WSDL
  - Java/SCA
  - JAX-WS
  - C++/SCA

```
<extension point="org.eclipse.stp.core.componentTypeIntrospector">
  <componentTypeIntrospector
     class=
        "org...introspection.PropertiesComponentTypeIntrospector"
     extension="properties"
     implementationElementType=
        "implementation.properties"
     shareableURIFactoryClass=
        "org...ShareablePropertyComponentTypeFactory"
  />
</extension>
```

http://www.eclipse.org/stp/cf/stp-core-introspection.html

Core EMF Infrastructure

- A standard approach for management of EMF resources that contain model elements
- Implements a memory-efficient way to share resources and include reference counting
- Ensures that various workbench components are rendering the same model and that changes are reflected accurately

http://www.eclipse.org/stp/cf/stp-core-infrastructure-emf.html

SOA Assembly Framework

- Core Contribution
  - Allows definition of SCA extension handlers
  - Allows definition of 'authoring contexts'
- Properties UI Contribution
  - Allows SCA extension providers to define how their extension is viewed/edited
  - The integration point between model extension and tooling

SOA Assembly Framework

- SCA Extension – what is it?
- The SCA Assembly model has specific extension points:
  - Implementation
  - Binding
  - Interface
- For example, if you want to implement your service in Intercal[0], you will need to develop an implementation extension

[0] http://www.catb.org/~esr/intercal/

SOA Assembly Framework

- Integrating your implementation extension with the tooling will involve registering a *handler* with SAF
- Other extensions, such as *binding* and *interface* will need handlers too

implemention	*IComponentHandler*
binding	*IEntryPointHandler* *IExternalServiceHandler*
interface	*IInterfaceHandler*

http://www.eclipse.org/stp/cf/saf/SAFcore.html

SOA Assembly Framework

- So what's the 'authoring context'?
- The *context* is a mechanism by which you can affect availability of certain extension handlers
- Two extension points for this:

`o.e.stp.core.saf.context`	Define a new keyed context with appropriate constraints
`o.e.stp.core.saf.contextExtension`	Augment an existing context definition, if permitted, to add a new handler contribution

SOA Assembly Framework

- For example, if you are constructing an SCA 0.95 *module* for deployment to a Tomcat server, then you could add a context that would disallow availability of C++ implementation handlers
- It's also possible to *add* a handler to an existing *context* definition, provided the *context* permits it.

EclipseWorld 2006

Sep 6-8, 2006

SOA System

SOA System

- The SOA System sub-project is all about delivering the assemblies to the hosting environments
- One of the main challenges in deploying service assemblies is in supporting deploying elements of the assembly to different hosts and doing this transactionally

SOA System

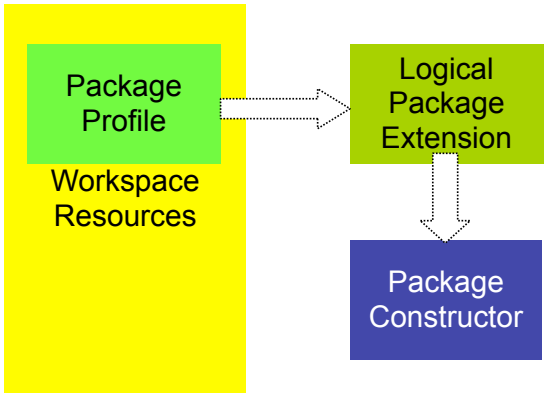
- The *Deployment Framework*
  - *packages* are created from the workspace - contents identified by extensions
    - logical packages
    - physical packages
  - *deployment extensions* - which control the details of deployment - are registered against *connection profiles* (DTP)

SOA System

- The *Deployment Profile Editor*
  - Allows users to target packages for deployment to specific servers
  - Allows users to tailor package configuration for a server
  - Provides a mechanism for repeatable deployments
- You will see this in the demo later...

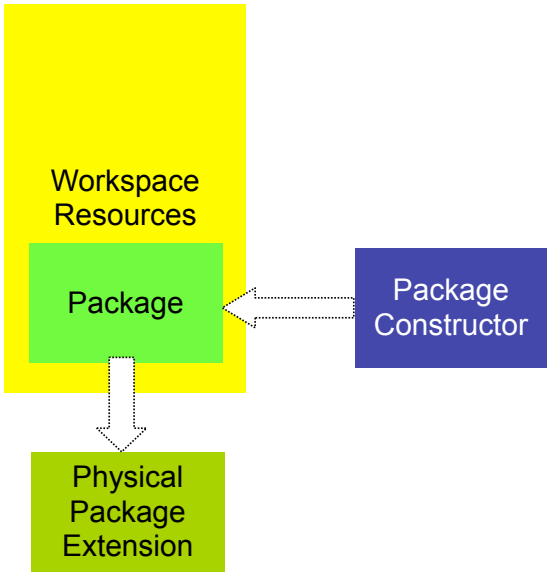
Deployment Framework

- User creates a *package profile* in workspace
- A *logical package* extension identifies the package profile to the framework.
- This helps locate a package constructor.



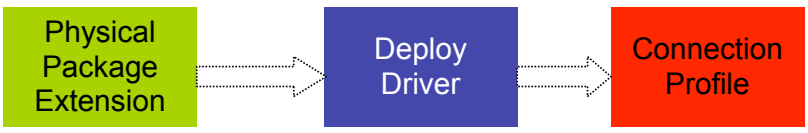
Deployment Framework

- A *package constructor* creates a deployable package
- A *physical package* extension identifies the deployable package to the framework.

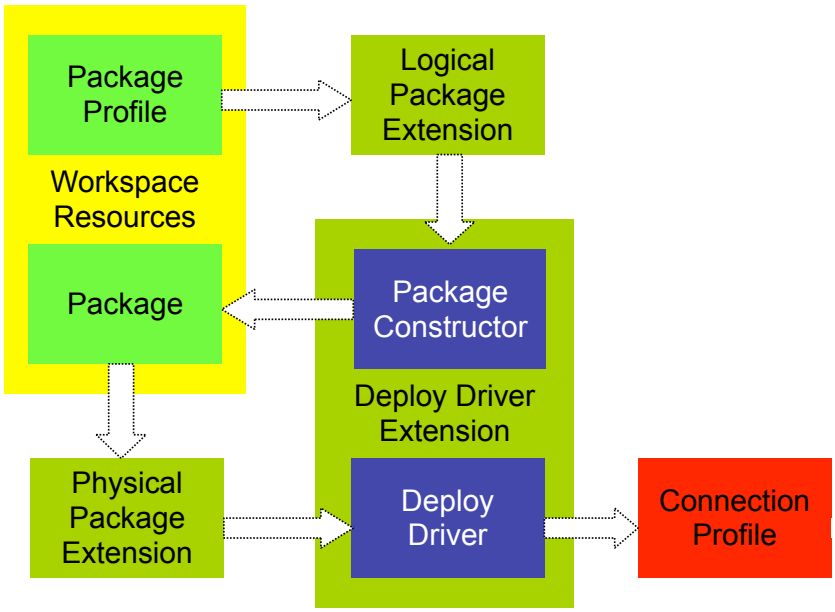


Deployment Framework

- The physical package extension is used to locate a deploy driver.
- A *deploy driver* used to identify *connection profiles* to which the package may be deployed and to deploy the package to a server represented by a connection profile.



Deployment Framework



Extension points

- Deployment framework provides a number of extension points:
  - `logicalPackage`
  - `configurablePackage`
  - `physicalPackage`
  - `deployDriver`
  - `technologyDefinition`
  - `serverDefinition`
  - `technologyMap`

Logical/Physical Packages

- Logical Package Extension
  - Might also be considered “abstract” or non-deployable packages
  - Identify items to be included in a package for deployment
  - May also include configuration details related to packaged items, including global settings
  - Present framework with a technology type identifier (for use in locating package constructors)
- Physical Package Extension
  - Might also be considered “concrete” or “deployable” packages
  - Present framework with a server type identifier (for use in locating deployment drivers)

Deployment Driver

- Deployment Driver Extension
  - Adds deployment capabilities to a DTP connection profile
  - Presents framework with a server type identifier (e.g. Tuscany, JBI)
  - May include a list of package constructors
  - Package Constructors
    - Used for constructing deployable packages from logical package definitions (identified through technology type)
    - Typed to specific technology type and version
    - Also provide validation of logical packages based on a targeted server (e.g. does the specified target support BPEL services)

Technology/Server Type

- Technology Types
  - Allows packages to be typed as belonging to a specific technology type (e.g. JAX-WS 2.0)
- Server Types
  - Types physical packages to be typed to a specific version and class of server (e.g. WAR, JBI-SCA)
  - Types deployment drivers as supporting a specific version and class of server

Technology Map

- Map
  - Allows specific version and class of server to be mapped as supporting a specific version and class of technology
  - Used by framework to tie package constructors (identified through a server type) to logical packages (identified through technology types); e.g. constructing a Tuscany WAR from an SCA 0.9 assembly

Contribution Highlight

- *demo time...!*

EclipseWorld 2006

Sep 6-8, 2006

BPEL2Java

BPEL 2 Java

- B2j originated from the TPTP project – originally known as the *TPTP Choreography Component*
- BPEL, as you know, is an XML vocabulary use to specify 'orchestration' between web services
- The original Choreography component was designed to convert BPEL into Java
- Execution was done with a replaceable engine framework
- The B2J project contains this design

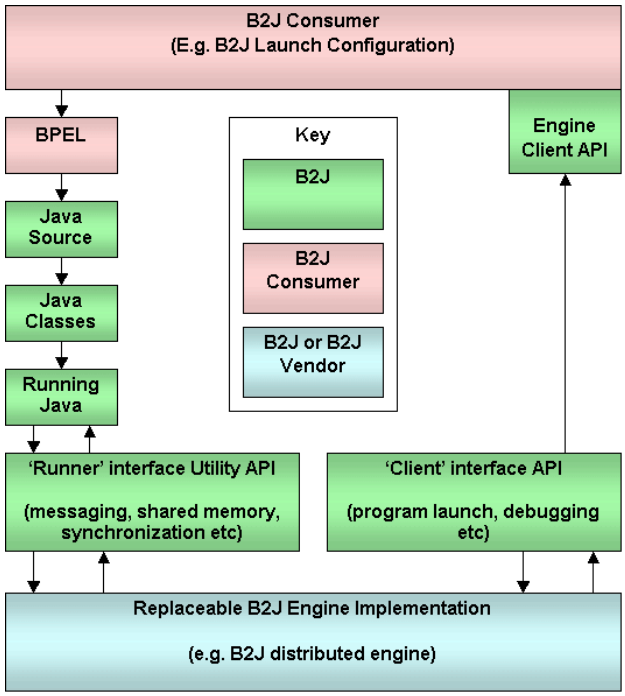
BPEL 2 Java

- Part 1: Generates Java from BPEL source
  - Generated code can use any engine which implements the B2J engine APIs.
- Part 2: B2J engine framework
  - Implementation runs the compiled class files.
- Plus: two example engine implementations
  - For local-only execution
  - For distributed execution

Developing and Deploying Services With STP | © 2006 by Oisín Hurley; made available under the EPL v1.0

B2J

- Scope for use:
  - testing choreo as before
  - hook with BPMN code gen
  - others to be explored!



Contribution Highlight

- *demo time...!*

Developing and Deploying Services With STP | © 2006 by Oisín Hurley; made available under the EPL v1.0

EclipseWorld 2006

Sep 6-8, 2006

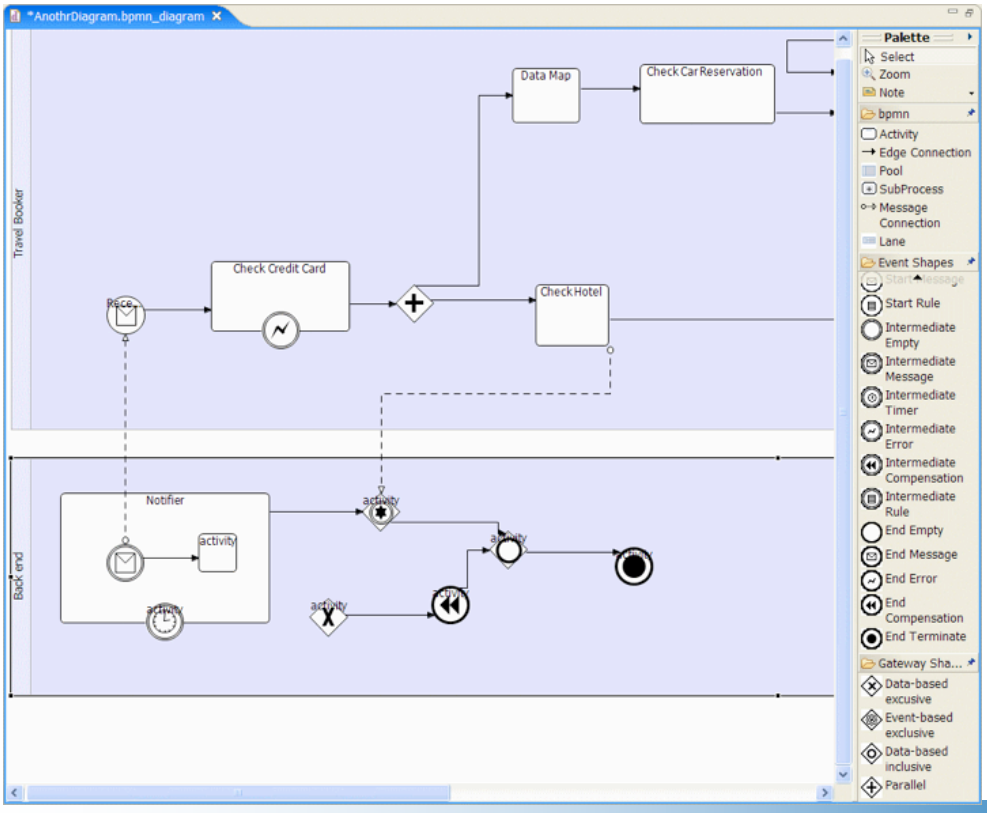
BPMN

What is BPM?

- Business process modeling is an activity performed by business analysts within an organization
- These analysts use modeling tools to depict both the current state of an enterprise and the desired future state
- The relationships of a business process in the context of the rest of the enterprise create greater capabilities in analyzing and planning a change

BPMN

- *Business Process Modeling Notation* is a standardized graphical notation for drawing business processes in a workflow
- BPM is used to communicate a wide variety of information to a wide variety of audiences
- BPMN is designed to cover this wide range of uses



Developing and Deploying Services With STP | © 2006 by Oisín Hurley; made available under the EPL v1.0

BPMN

- One of the BPM ‘visions’ is to make the models fully executable and capable of round-trip engineering
- Often BPM tools will generate an implementation of the process flow - for example a collection of BPEL scripts
- A contribution of a BPMN editor and code generation hooks has just been put into bugzilla (156018)

Contribution Highlight

- *demo time...!*

EclipseWorld 2006

Sep 6-8, 2006

Status

HEAD

- JAX-WS web service application development support framework
  - Celtix exemplar, re-purposable to Axis/Reference Implementation
- Core Models and Frameworks with introspection example
- Deployment framework based on DTP connection profile
- BPEL 2 Java generation and run framework
- Incoming: GMF-based BPMN editor

TAIL - what we need

- **Integration** between JAX-WS/Core based upon the whitepaper at osoa.org
- SCA Java service creation and core **integration**
- **Integration** with Deployment Framework and WTP deployment technologies
- A vision for testing of service assemblies
- SCDL editor
- Applicability to other development models
- Visualization of core model

...and yet more...

Activities

- **Integration work**
- JAX-WS code improvements
- Filling out the STP website
- Advancing Core Models to use *composites*
- Knowledge sharing
- Did I mention **integration work**?

Activities

- Evangelism - blog and articles etc
- Examination of interaction with rest of Eclipse community & others
  - WTP : deployment and service creation
  - DTP : connection framework
  - Modeling Project : EMF, GMF and BPMN
  - Celtix - JAX-WS runtime
  - Tuscany - SCA runtime

Experiences

- Code contributions are substantial, so understanding them enough to make good integration decisions takes time and focus
- As a result progress has been slow, which can hurt developer motivation...
- Understanding how STP will consume from and provide to other projects is not as easy as it looks :-)

Questions...

- We’re avidly looking for developers that can help us in our integration work... :-)
- We’re avidly looking for all kinds of SOA scenarios to help us plan feature roadmaps and prioritize
  - *http://wiki.eclipse.org/index.php/STP_Call_for_Scenarios*
- Questions now...

