

14.–17. 09. 2009
in Nürnberg



Herbstcampus

Wissenstransfer
par excellence

Schwimmen Lernen!

SOA mit Eclipse Swordfish

Dr. Halil-Cem Gürsoy

CDI Concepts Development Integration AG

Schwimmen lernen!

- Publikumsfragen...
- Historie
- Architektur von Swordfish
 - Komponenten und OSGi in Swordfish
 - Nachrichtenfluss und Erweiterungsmöglichkeiten
- Arbeiten mit Swordfish
 - Swordfish Tooling
 - Service-Erzeugung
 - Service Orchestrierung
- Aktueller Stand der Entwicklung und Meilensteine
- Fazit und Ausblick

Der Referent...

- Insgesamt ca. 10 Jahre Beratung, davor Entwicklung im Forschungsumfeld
- Senior Consultant - JEE Evangelist bei CDI AG



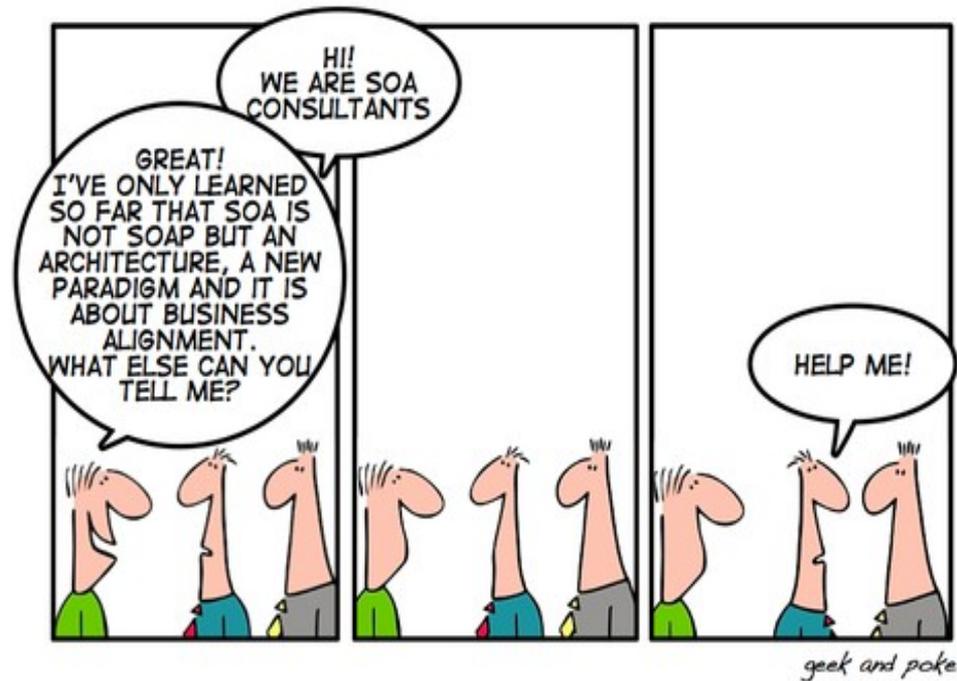
- Hauptsächlich EAI und SOA-Projekte im JEE Umfeld, aber auch klassische JEE-Projekte
- Autor und Referent auf Konferenzen

Schwimmen lernen!

- **Publikumsfragen...**
- Historie
- Architektur von Swordfish
 - Komponenten und OSGi in Swordfish
 - Nachrichtenfluss und Erweiterungsmöglichkeiten
- Arbeiten mit Swordfish
 - Swordfish Tooling
 - Service-Erzeugung
 - Service Orchestrierung
- Aktueller Stand der Entwicklung und Meilensteine
- Fazit und Ausblick

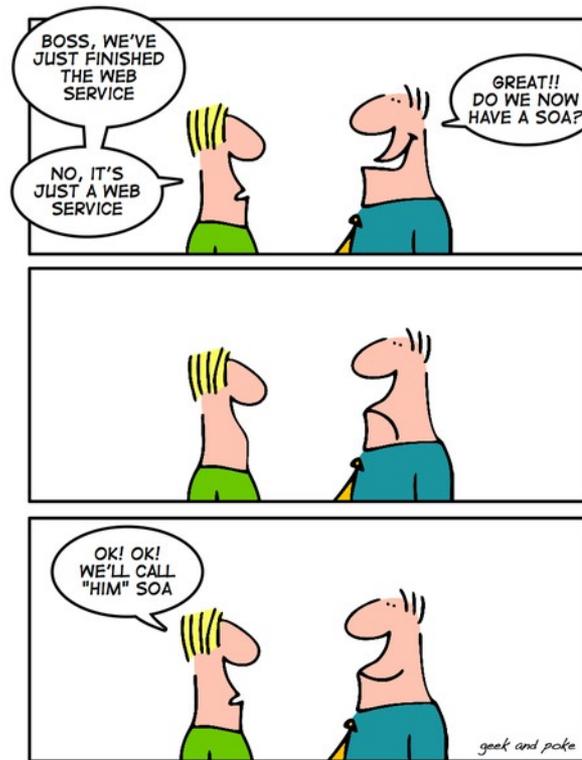
Publikumsfrage(n)...

- Was ist SOA?



Und eine weitere Herausforderung...

- Was zeichnet eine SOA aus?



HOW TO GET A SOA

Schwimmen lernen!

- Publikumsfragen...
- **Historie**
- Architektur von Swordfish
 - Komponenten und OSGi in Swordfish
 - Nachrichtenfluss und Erweiterungsmöglichkeiten
- Arbeiten mit Swordfish
 - Swordfish Tooling
 - Service-Erzeugung
 - Service Orchestrierung
- Aktueller Stand der Entwicklung und Meilensteine
- Fazit und Ausblick

Historie von Swordfish

- Nach strategischer Entscheidung (1999) Einführung der Service Oriented Platform (SOP) bei der Deutschen Post
- 2006 sind ca. 160 Services in unternehmenskritischen Anwendungen aktiv
- 2007 Gründung der SOPERa als unabhängigen Dienstleister

Historie von Swordfish

- 2007 SOPERÄ wird Eclipse Member
- 2007 Project Proposal für Swordfish
- 2009 Swordfish 0.9 Bestandteil von Eclipse Galileo
...und schnell wieder nicht mehr...
...aber dazu später mehr!



Swordfish in der Presse



heise **Developer** 

News Rubriken Blogs Podcast Szene Software FAQ-Liste Sie sind Gast • Einloggen | Registrieren

7-Tage-News News-Archiv

heise Developer > News > 2009 > KW 26 > SOA-Framework Swordfish als Bestandteil von Eclipse Galileo

25.06.2009 - 08:58 [« Vorige](#) | [Nächste »](#)

SOA-Framework Swordfish als Bestandteil von Eclipse Galileo

Das neue Galileo-Simultan-Release von unterschiedlichen Eclipse-Projekten enthält als Laufzeitkomponente unter dem Namen Swordfish ein erweiterbares Framework für serviceorientierte Architekturen (SOA). Die Technik basiert auf dem von der Deutschen Post 2007 an die Eclipse Foundation übergebenen Enterprise Service Bus (ESB). Im Rahmen von Swordfish wurde der ESB zu einem SOA-Framework ausgebaut.

"Ein wesentliches Ziel der Weiterentwicklung unseres SOA-Frameworks im Rahmen der Eclipse Foundation war die Schaffung der Grundlagen für Unternehmen, die innovative SOA-Markten

Schwimmen lernen!

- Publikumsfragen...
- Historie
- **Architektur von Swordfish**
 - Komponenten und OSGi in Swordfish
 - Nachrichtenfluss und Erweiterungsmöglichkeiten
- Arbeiten mit Swordfish
 - Swordfish Tooling
 - Service-Erzeugung
 - Service Orchestrierung
- Aktueller Stand der Entwicklung und Meilensteine
- Fazit und Ausblick

•Swordfish in wenigen Sätzen

- Swordfish basiert auf ServiceMix 4
- Viele wichtige Kernkomponenten kommen von ServiceMix wie NMR, SE's und BC's (→ JBI)
- Swordfish fügt zusätzliche Komponenten hinzu:
 - Management
 - Configuration
 - Monitoring
 - Interceptor Chain
- ...und ein Framework um alles zusammenzufügen

Architektur von Swordfish

- Swordfish basiert z.Z. auf zwei von drei (relevanten) Standards der SOA-Welt:



- Kommt später... vermutlich
- Messaging Abstraction
- Message Routing
- Protokoll intermediation
- Komponenten Modell

JBIGanz kurz...

- JBI = Java Business Integration
- JSR 208 (JBI 1.0), JSR 312 (JBI 2.0)
- Definiert einen Container für
 - Service Provider
 - Service Consumer
 - Weitere Container
- NMR (Normalized Message Router)
- „Service Units“ werden zu „Service Assemblys“
zusammengefasst und deployed
- Diverse Implementierungen, u.a. Open ESB,
Apache ServiceMix

OSGi wirklich sehr kurz... und vereinfacht

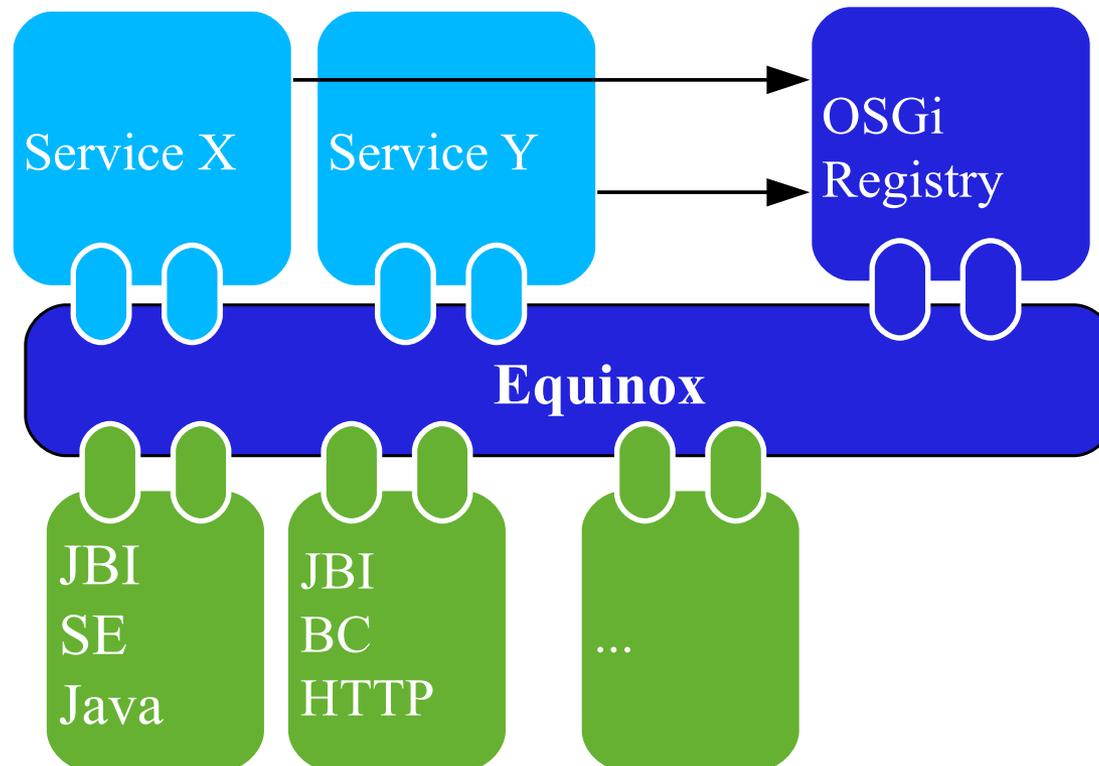
- Komponentenmodell, um Anwendungen zu modularisieren
- „Module über der Package-Ebene“
- Spezifiziert durch die OSGi Alliance
- im Rahmen der JCP als JSR291
- Wohl das bekannteste Produkt auf OSGi-Basis: Eclipse
- Diverse OS-Implementierungen:
 - Eclipse Equinox, Apache Felix, Knopflerfish
 - ...und viele andere

Und in Swordfish?

- (SCA over) JBI over OSGi
- Verteilen der Verantwortlichkeiten in Module
- JBI-Container (Apache ServiceMix) wird als OSGi-Modul deployed
- Als OSGi-Container kommt Eclipse Equinox zum Einsatz

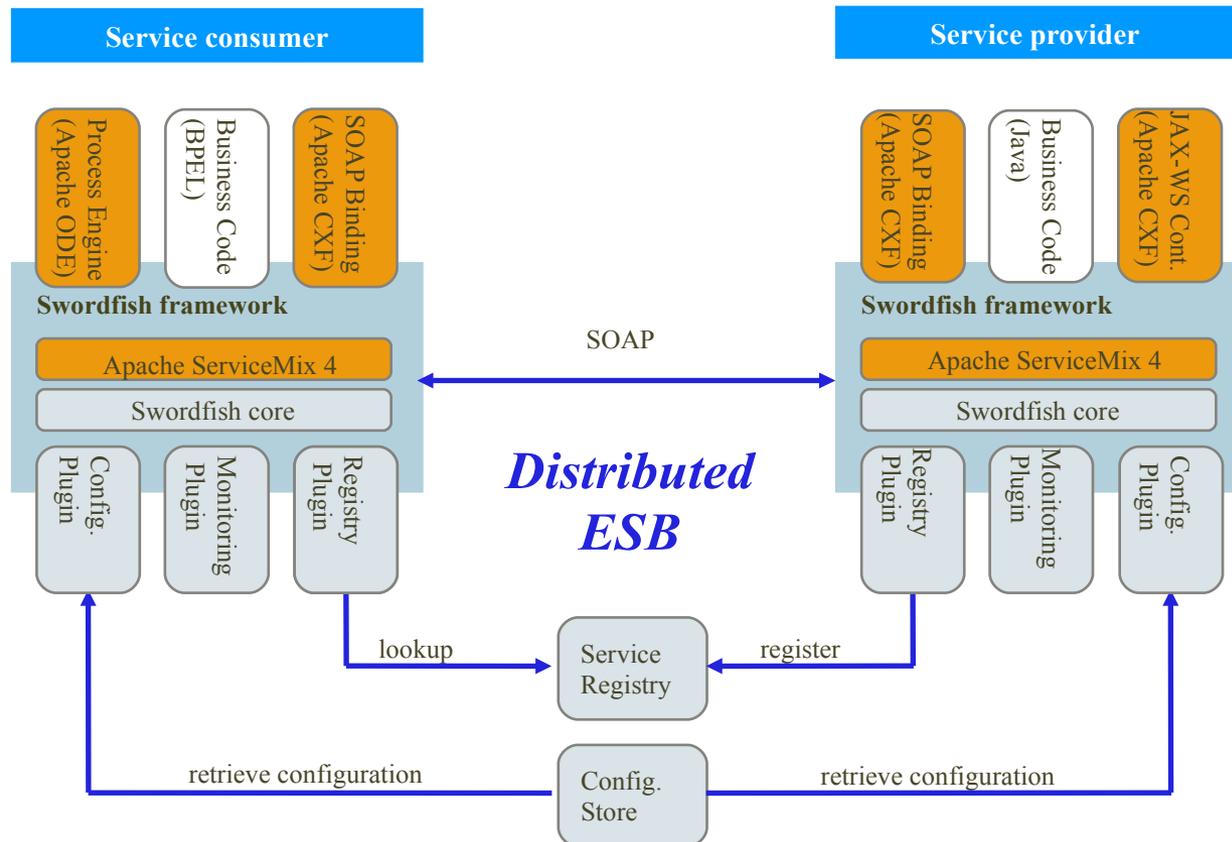
Services als OSGi-Module

- Eigene Services werden als OSGi-Module deployed



Distributed ESB

- Swordfish basiert auf dem *Distributed ESB Pattern*



Erweiterungsmöglichkeiten

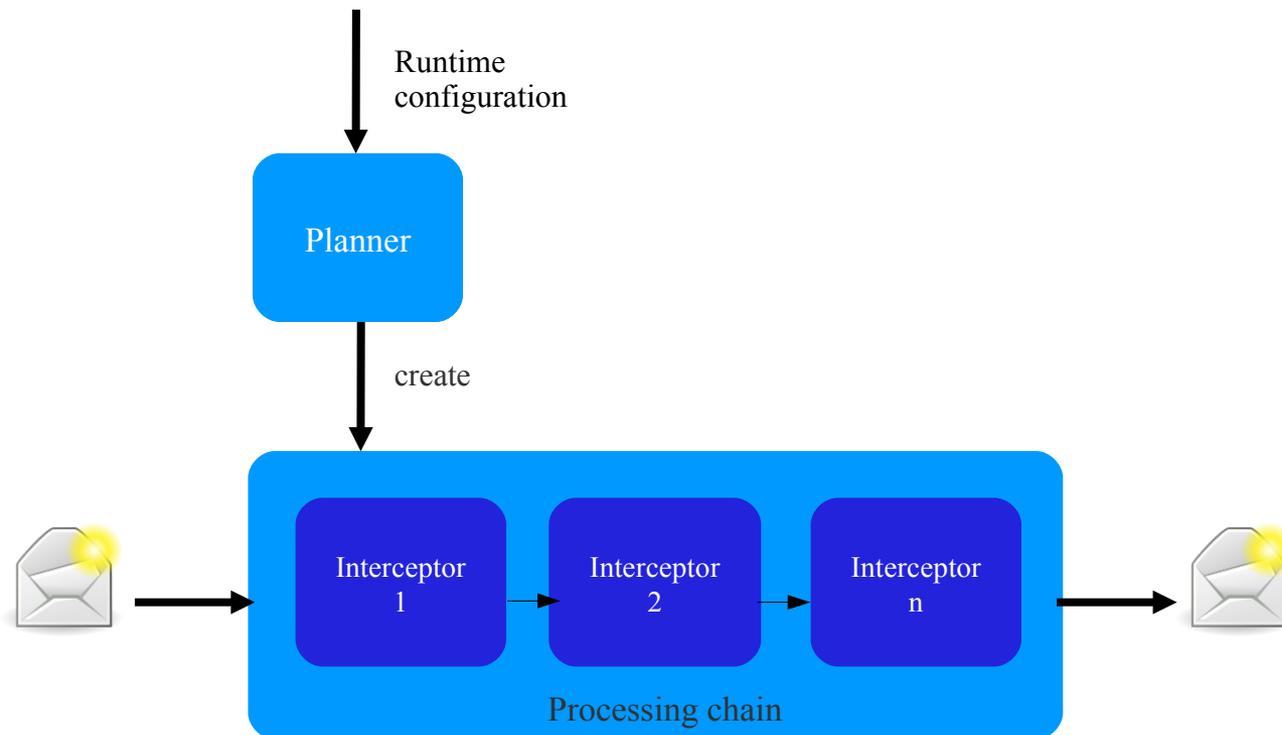
- Erweiterungen als OSGi Service registrieren
- Swordfish bietet z.Z. 4 API's an:
 - General Interceptor API
 - Service Resolver API
 - Event API
 - Configuration API

Beispiel General Interceptor API

- Swordfish integriert sich in NMR von ServiceMix, „intercepted“ alle Nachrichten und führt sie einer „Interceptor Chain“ zu
- Interceptoren haben vollen Zugriff auf eine Nachricht:
 - Message Properties ändern
 - Payload lesen / ändern
 - Routing
 - Neue Messages erzeugen
- Ein „Planner“ definiert die Interceptor Chain

Interceptor Chains in Swordfish

- Message-Processierung durch Interceptoren



Mediation von Nachrichten

- Es gibt z.Z. in Swordfish keine explizite Mediation von Nachrichten („Mapping“)
- Umweg: Über Interceptoren können die Payloads der Nachrichten verändert werden
- z.Z. Keine direkte grafische Unterstützung!

Schwimmen lernen!

- Publikumsfragen...
- Historie
- Architektur von Swordfish
 - Komponenten und OSGi in Swordfish
 - Nachrichtenfluss und Erweiterungsmöglichkeiten
- **Arbeiten mit Swordfish**
 - Swordfish Tooling
 - Service-Erzeugung
 - Service Orchestrierung
- Aktueller Stand der Entwicklung und Meilensteine
- Fazit und Ausblick

Swordfish Tooling

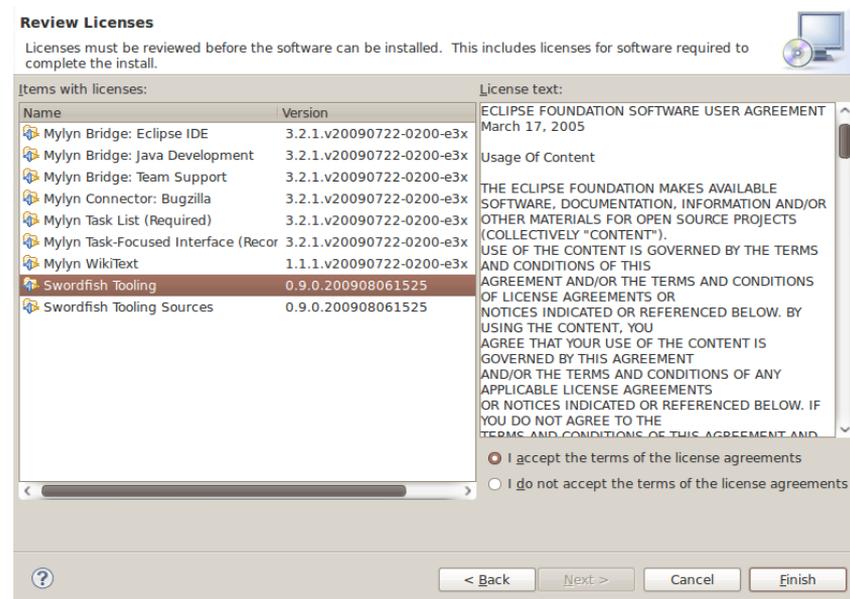
- Bietet Werkzeuge, um innerhalb von Eclipse Services zu erstellen, diese ggfs. zu Orchestrieren und in der SOA Runtime auszuführen
- Ist ein Bestandteil des Swordfish-Projektes (auch wenn es lange auf Google Code gehostet wurde)

Ein einfaches Swordfishprojekt

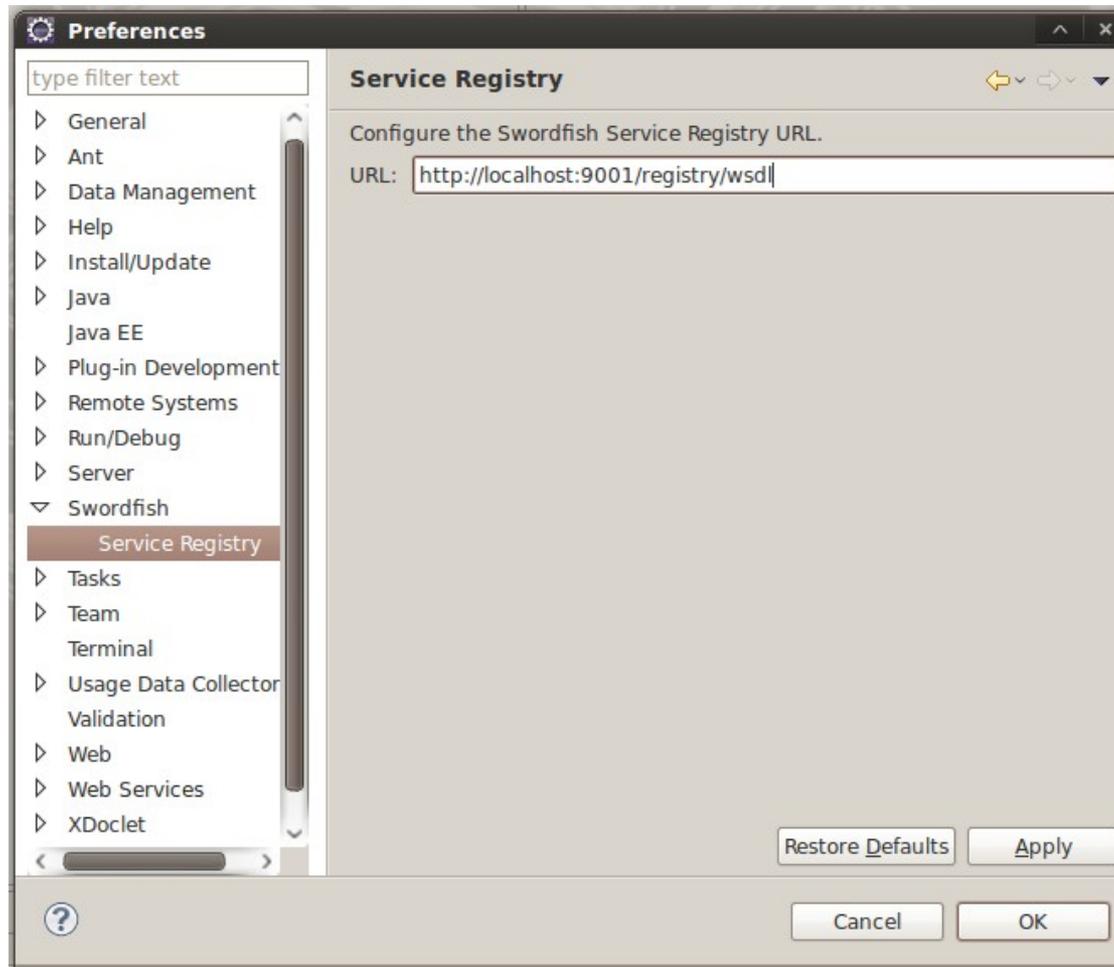
- Wir wollen ein einfaches Swordfish-Projekt anlegen
- In diesem Projekt soll ein Service definiert werden
- Service muss implementiert werden
- Service muss deployed werden
- Testen des Services

Installation

- z.Z. steht Swordfish und das Tooling nur über Google Code zur Verfügung:
- Site-Link: <http://swordfish-tooling.googlecode.com/files/swordfish-tooling-site.zip>



Registry konfigurieren



Target Platform Definition

- Es muss eine neue Target Platform definiert werden
- Swordfish-Tooling bietet ein Template an

Target Definition 

Create a new target definition.

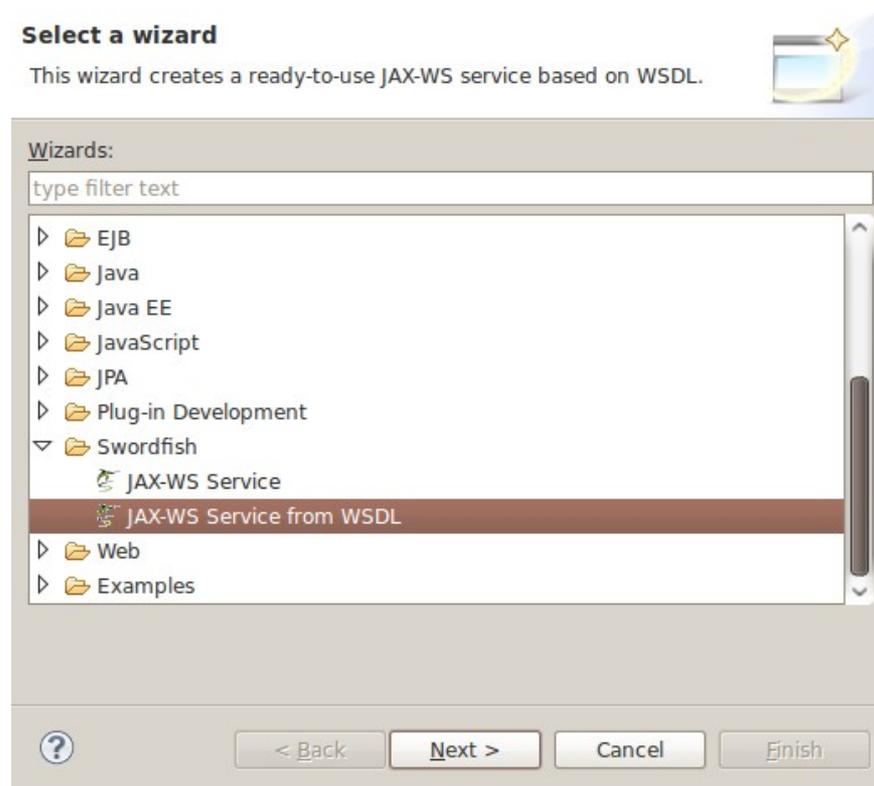
Initialize the target definition with:

- Nothing: Start with an empty target definition
- Default: Default target for the running platform
- Current Target: Copy settings from the current target platform
- Template:



Swordfish Projekte

- Es können Projekte „from the scratch“ oder auf eine WSDL basierend erzeugt werden.



Projekt-Wizard

Plug-in Project 

Create a new plug-in project

Project name:

Use default location

Location:

Choose file system:

Project Settings

Create a Java project

Source folder:

Output folder:

Target Platform

This plug-in is targeted to run with:

Eclipse version:

an OSGi framework:

Working sets

Add project to working sets

Working sets:

Service-Skelett

- Führt zu folgendem Code, der ausimplementiert werden muss:

```
23 ~/  
24  
25 @javax.jws.WebService(  
26     serviceName = "FlightReservationService",  
27     portName = "FlightReservationSOAP",  
28     targetNamespace = "http://www.eclipse.org/swordfish/samples/FlightReservation/",  
29     endpointInterface = "org.eclipse.swordfish.samples.flightreservation.FlightReservation")  
30  
31 public class FlightReservationImpl implements FlightReservation {  
32  
33     private static final Logger LOG = Logger.getLogger(FlightReservationImpl.class.getName());  
34  
35     /* (non-Javadoc)  
36     * @see org.eclipse.swordfish.samples.flightreservation.FlightReservation#reserveFlight(org.eclipse.swordfish.samp  
37     */  
38     public java.lang.String reserveFlight(org.eclipse.swordfish.samples.flightreservation.FlightData flightData) {  
39         LOG.info("** Executing operation reserveFlight **");  
40         LOG.info("Flightnumber: [" + flightData.getFlightNumber()+"]");  
41         LOG.info("Flightdate: [" + flightData.date + "]");  
42         return null;  
43     }  
44 }
```

Metadaten für CFX

- In der Datei `META-INF/spring/cxf-endpoint.xml` werden wichtige Konfigurationen des Services vorgenommen
- Die Datei ist eine „klassische“ Spring-Konfigurationsdatei (unterstützt durch CXF)
- Mit dem Eintrag *http:endpoint / locationURI* wird definiert, unter welcher URI der Service erreichbar sein wird
- Es gibt aber noch ein Problem (s. nächste Folie)

Ein Workaround ist nötig

- Ein Workaround ist nötig, da in der CXF-Konfiguration der Endpoint falsch definiert wird
- Hierzu Datei `META-INF/spring/cxf-endpoint.xml` öffnen und den Eintrag *`http:endpoint / locationURI`* anpassen
- `[SName] Service in [SName] _Service` ändern:

```
<http:endpoint
  endpoint="httpConsumerEndpointForFlightReservationService"
  service="httpConsumerEndpointForFlightReservationService"
  targetService="serviceNamespace:FlightReservationService"
  soap="true"
  role="consumer"
  locationURI="http://0.0.0.0:8197/FlightReservation_Service/"
  defaultMep="http://www.w3.org/2004/08/wsdl/in-out" />
```



Deployment als OSGi-Bundle

- Deployment erfolgt als OSGi-Bundle in der Swordfish-
Runtime
- Hierzu muss eine *Runtime Configuration* angelegt und
das Service-Projekt hinzugefügt werden:

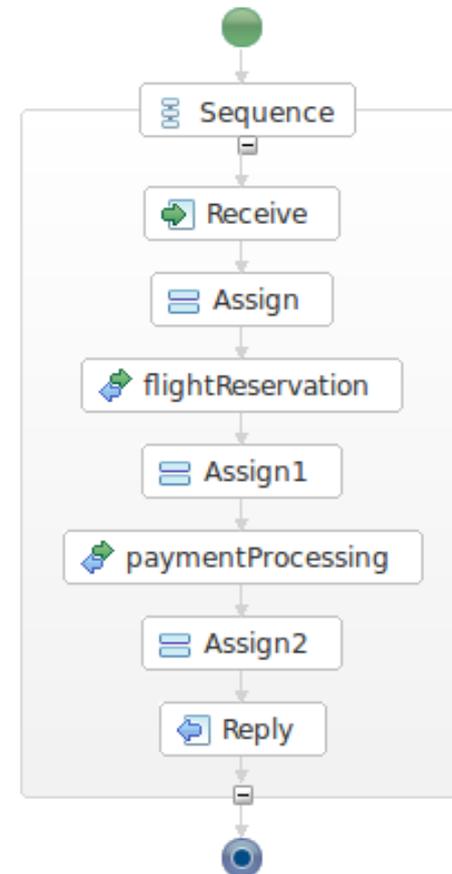
```
-Declipse.ignoreApp=true  
-Dosgi.noShutdown=true  
-Dservicemix.base=.  
-Dorg.osgi.service.http.port=9001  
-Dorg.eclipse.swordfish.registry.fileLocation=  
"${bundle_loc}/platform/registry"  
-Dorg.osgi.framework.system.packages=  
"com.sun.jimi.core,com.sun.net.ssl,com.sun.net.ssl.internal.ssl,..."
```


Service Testen

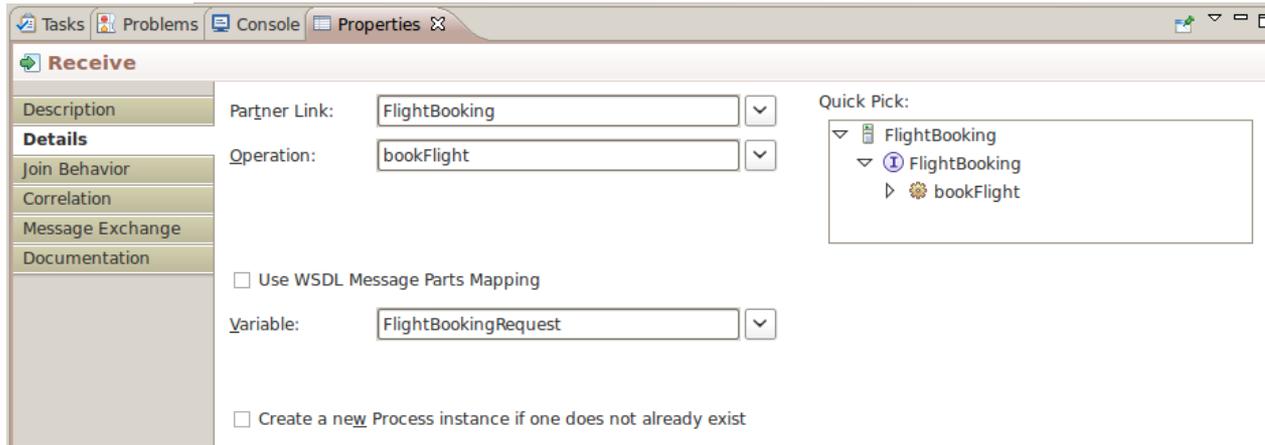
- Testen des Services mit Eclipse-Bordmitteln möglich:
 - In der View *Service Explorer* die WSDL des Services importieren und ausführen
 - Setzt voraus, dass der Endpoint in der WSDL richtig konfiguriert ist
 - Alternative: WSDL aus Registry → WSDL uploaden, dann über WSIL (default ist `http://localhost:9001/registry/wsil`)
 - Automatische WSDL-Erzeugung aus Service funktioniert fehlerhaft

Services orchestrieren

- Swordfish unterstützt BPEL über Apache ODE
- ODE-Unterstützung kann mit Swordfish installiert werden



BPEL-Prozess in Swordfish definieren



Receive

Description: Partner Link: FlightBooking

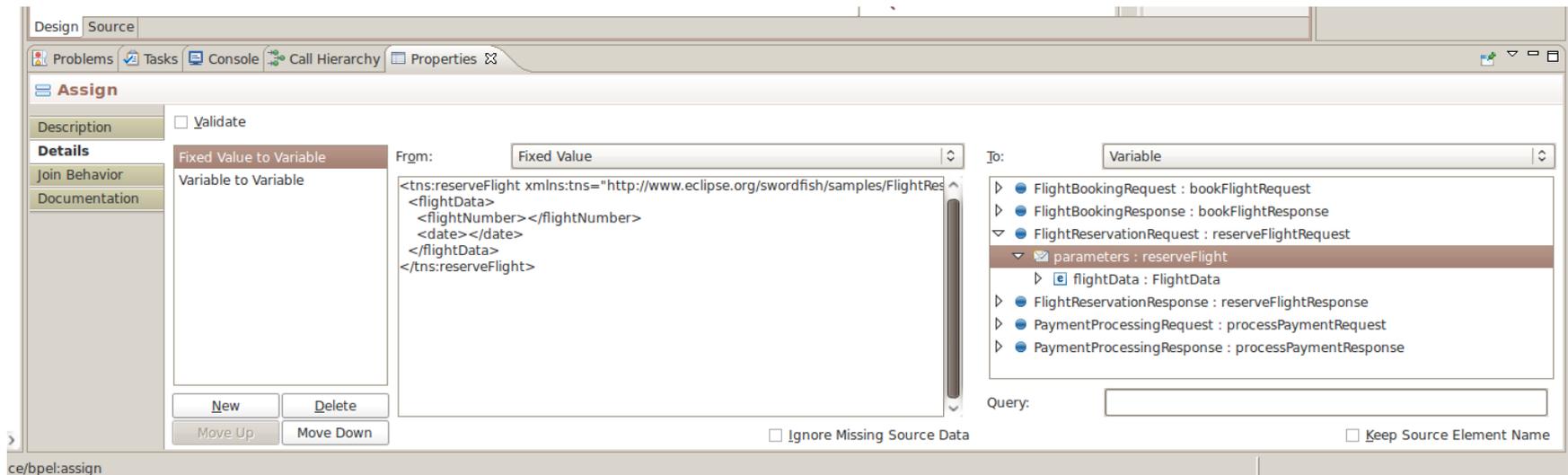
Details: Operation: bookFlight

Quick Pick: FlightBooking, FlightBooking, bookFlight

Use WSDL Message Parts Mapping

Variable: FlightBookingRequest

Create a new Process instance if one does not already exist



Assign

Description: Validate

Details: Fixed Value to Variable, Variable to Variable

From: Fixed Value

To: Variable

```
<tns:reserveFlight xmlns:tns="http://www.eclipse.org/swordfish/samples/FlightRes"
  <flightData>
    <flightNumber></flightNumber>
    <date></date>
  </flightData>
</tns:reserveFlight>
```

parameters: reserveFlight

- flightData: FlightData
- FlightBookingRequest: bookFlightRequest
- FlightBookingResponse: bookFlightResponse
- FlightReservationRequest: reserveFlightRequest
- FlightReservationResponse: reserveFlightResponse
- PaymentProcessingRequest: processPaymentRequest
- PaymentProcessingResponse: processPaymentResponse

Query:

Ignore Missing Source Data Keep Source Element Name

BPEL in Swordfish

- Auch BPEL-Prozesse können als OSGi Bundle deployed werden
- Alternativ: JBI
- ABER:
z.Z. Noch keine Unterstützung durch das Tooling!
- Im Detail beschrieben unter
http://www.eclipse.org/swordfish/assets/EC2009_BPEL.pdf

Schwimmen lernen!

- Publikumsfragen...
- Historie
- Architektur von Swordfish
 - Komponenten und OSGi in Swordfish
 - Nachrichtenfluss und Erweiterungsmöglichkeiten
- Arbeiten mit Swordfish
 - Swordfish Tooling
 - Service-Erzeugung
 - Service Orchestrierung
- **Aktueller Stand der Entwicklung und Meilensteine**
- Fazit und Ausblick

Aktueller Stand

- 5 Comitter
- 7 Contributor (Zahlen verändern sich regelmäßig)
- Swordfish 0.9 war Teil des Galileo Release:



The screenshot shows a web browser window displaying the Eclipse website's 'Downloads' page for the Swordfish project. The browser's address bar shows the URL <http://www.eclipse.org/swordfish/downloads/>. The website's navigation menu includes links for HOME, DOWNLOADS, USERS, MEMBERS, COMMITTERS, RESOURCES, PROJECTS, and ABOUT US. A search bar is visible with the text 'Google Custom Search'. The main content area features a heading 'Downloads' and a sub-heading 'Downloads are temporarily disabled'. The text below explains that the release is suspended due to outstanding process-related issues and that the project is currently in an 'Incubation Phase'. A sidebar on the left contains a list of links for the Swordfish project, including 'About This Project', 'Users', 'Getting Started', 'Downloads', 'Newsgroups', 'Bugs', and 'Integrators'. A small graphic on the right side of the page shows an egg with the text 'eclipse incubation' and 'This project is in the Incubation Phase'.

Swordfish nicht mehr im Release!

- Viele Abhängigkeiten durch ServiceMix 4, CXF, Spring usw.
- Einige dritt- bzw. „viert“-Bibliotheken waren nicht durch den Freigabeprozess von Eclipse gelaufen → Swordfish aus dem Release entfernt
- Zur Zeit nur über Google Code downloadbar (s. Links) oder aus dem Eclipse SVN Repository (build!)
 - Viele der Tags bauen nicht durch, da POM's nicht (mehr) konsistent



Alter Meilensteinplan

- Entnommen EC2009-Folien des SF-Teams:

	Galileo June 2009 Release 0.9	Eclipse Summit Europe November 2009 Release 1.0	EclipseCon March 2010 Release 1.1	Eclipse Release Summer 2010 Release 2.0
ESB	Swordfish framework basic plugins basic tool support	Integration with open source security framework	Test support	Full enterprise ESB
Process Orchestration	BPEL process engine	Integration with open source BPM suite (Spagic)		
Registry & Repository	Basic runtime registry	Advanced runtime registry/service locator basic repository	Service repository w/service lifecycle management	Full enterprise service repository
Management	JMX-based management	Plugin for integration into Hyperic HQ		
Data Integration		Plugins for open source ETL tools (e.g. Talend)	Plugins for open source EDI tools (e.g. Smooks)	
Service/Business Activity Monitoring		Service activity reporting	Complex event processing	Full business activity monitoring

Letzter Stand

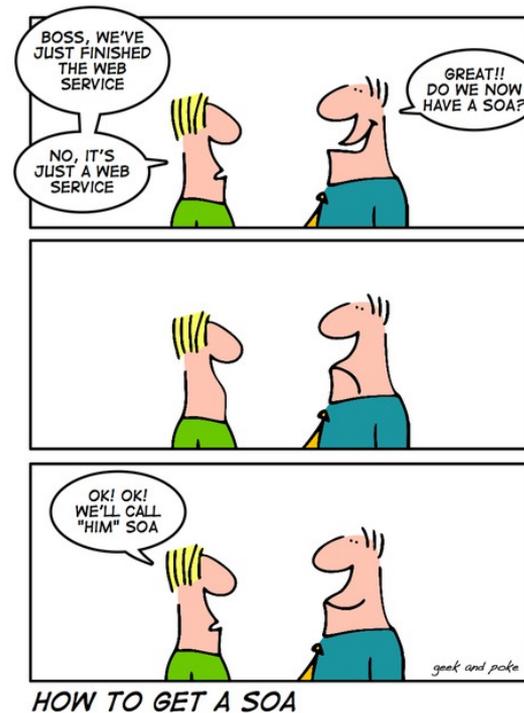
- Nach aktuellem Stand soll ab 25.09.2009 Swordfish **0.9.1** zum Download bei Eclipse mit dem *Galileo SR1* bereit stehen (bis dahin: Google Code)
- Release „1.0“ und damit das Verlassen der Inkubation Phase fraglich
 - Community noch zu klein (aber das Projekt liegt unter hoher Beobachtung), API-Stabilität
- Wichtigste Themen nach 0.9.1:
 - Verbesserung des Toolings
 - Unterstützung von Remote Configuration Repositories

Schwimmen lernen!

- Publikumsfragen...
- Historie
- Architektur von Swordfish
 - Komponenten und OSGi in Swordfish
 - Nachrichtenfluss und Erweiterungsmöglichkeiten
- Arbeiten mit Swordfish
 - Swordfish Tooling
 - Service-Erzeugung
 - Service Orchestrierung
- Aktueller Stand der Entwicklung und Meilensteine
- **Fazit und Ausblick**

Fazit

- Ein sehr interessantes Projekt
- Aber haben wir tatsächlich schon eine SOA-Plattform?



Fazit

- Reifegrad gemessen an SOPERa und anderen OSS ESB's / SOA-Plattformen niedrig
- Mediation auf Umweg von Interceptoren in Projekten nicht einfach handhabbar
- Das Tooling muss noch um wesentliche Elemente erweitert werden, z.B.
 - Deployment BPEL-Prozesse, Policy-Editoren
- Bug in Registry (?)
- Dokumentation (!)

Links

- Projekt Homepage:
<http://www.eclipse.org/swordfish/>
- Swordfish Dokumentation / Wiki:
http://wiki.eclipse.org/Swordfish_Documentation
- Swordfish Tooling bei Google Code:
<http://code.google.com/p/swordfish-tooling/>
- Swordfishing Blog:
<http://swordfishing.wordpress.com/>
- SOA Jobs bei CDI AG:
<http://www.cdi-ag.de/index.php?id=karriere>

Dank an...

- Dank an
 - Oliver Wolf (SOPERA), u.a. für sein Folien-Original von der EC2009
 - Renat Zubairov (SOPERA), u.a. für sponaten Hilfe über Skype, wenn es mal hakte

- Bilder von
 - „Geek & Poke“
(<http://geekandpoke.typepad.com/>)
 - „Two very dead Swordfish“ (
<http://flickr.com/photos/47565600@N00/63703559/>)

Das Ende naht...



14.–17. 09. 2009
in Nürnberg



Herbstcampus

Wissenstransfer
par excellence

Vielen Dank!

Dr. Halil-Cem Gürsoy

CDI Concepts Development Integration AG

halil-cem.guersoy [at] cdi-ag.de