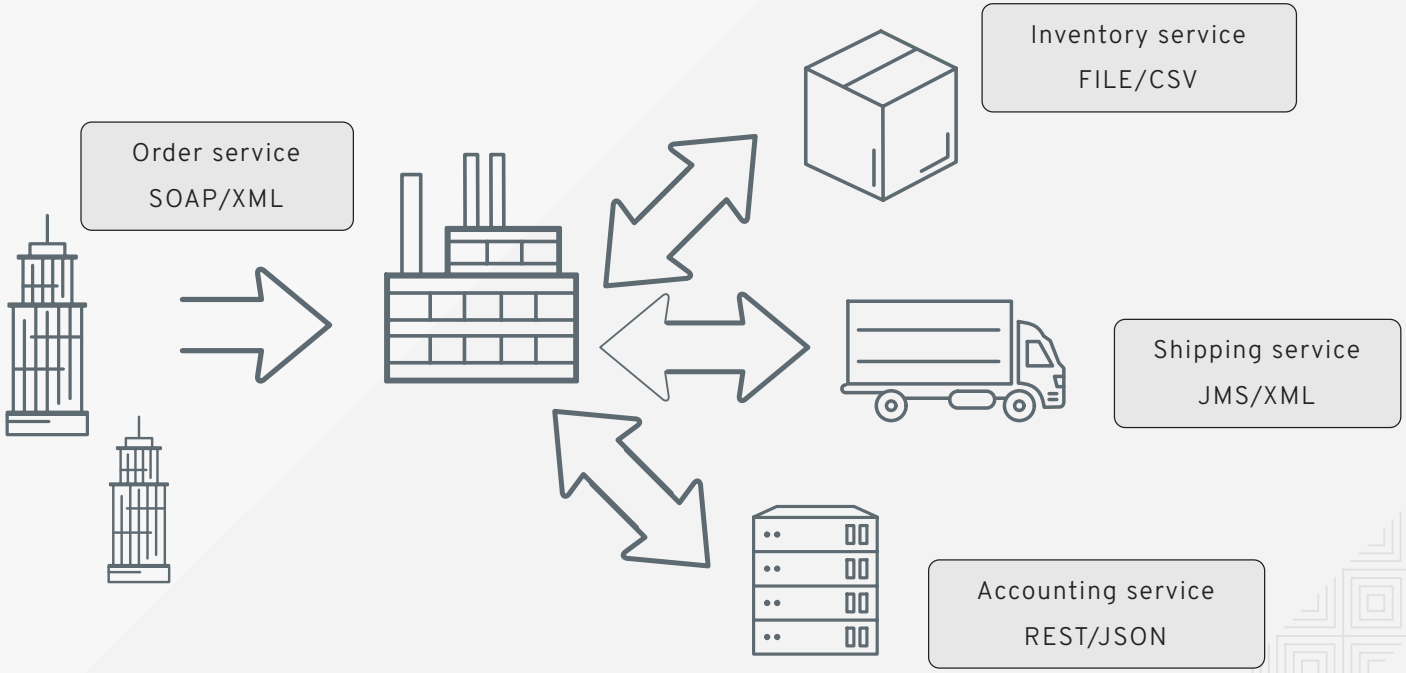
SWITCHYARD

EXERCISE

Tomáš Turek - tturek@redhat.com

WHAT WE WANT TO ACHIEVE?

USE SWITCHYARD TO COMPOSE 3 APPLICATIONS



ENVIRONMENT

RED HAT JBOSS FUSE + JBOSS DEVELOPER STUDIO WITH INTEGRATION STACK

INSTALATION GUIDE:

HTTP://WWW.JBOSS.ORG/PRODUCTS/FUSE/GET-STARTED/

PREPARATION

Clone git projects:

> git clone https://github.com/qa/course-sys-int-systems.git

> git clone https://github.com/qa/course-sys-int-switchyard-seminar.git

- copy file $WORKSPACE/course-sys-int-switchyard-seminar/src/resources/keystore.jks to $FUSE_HOME/bin folder
- copy file $WORKSPACE/course-sys-int-switchyard-seminar/activemq.xml to $FUSE_HOME/etc folder
- add a user to $FUSE_HOME/etc/users.properties:
  shipuser=shippwd,admin,manager,viewer,Monitor, Operator, Maintainer, Deployer, Auditor, Administrator, S
- start/restart JBoss Fuse $FUSE_HOME/bin/fuse

run course-sys-int-system application

> mvn clean camel:run

LAB APPLICATION

project: course-sys-int-switchyard-seminar

initial branch: switchyard-00

run all tests

> mvn clean verify

run specific test

> mvn clean verify -Dtest=Lab01Test

STEP 1

COMPONENTS, BEAN

Goals: create OrderStatusService and wire it with internal components

Steps:

- create new component named OrderStorageComponent
  - add component service use OrderStatusService contract
  - assign OrderStatusServiceBean as component implementation
- modify OrderComponent component
  - add component reference use OrderStatusService contract
  - assign OrderServiceBean as component implementation
  - inject OrderStatusService in implementation of component
- wire InventoryReplyComponent with OrderStorageComponent

Test: Lab01Test

STEP 2

JMS BINDING, JAXB

Goals: Integrate shipping application with OrderComponent

Problem: send orderId via JMS message header

Steps:

- create component ShipmentReplyComponent:
  - add component service use ShipmentReplyService contract
  - promote ShipmentReplyService via contract ShipmentService.wsdl#wsdl.porttype(ShipmentReplyServicePortType) and use JMS binding:
    - queue: SHPMNT.RESP
    - includes all message headers (Bindings -> Message Composer)
  - add component reference use OrderStatusService contract
  - assign ShipmentReplyServiceBean as component implementation
- declare JAXB and Java transformators

Test: Lab02Test

STEP 3 - 1/2

FILE BINDING, PROPERTIES, COMPOSER

Goals: Integrate inventory application with OrderComponent

Problem: how wire reply message from file service with order status

Steps:

- set path to course-sys-int-systems project in service.properties file
- add service.properties in switchyard domain configuration

- assign File binding to Inventory composite reference:
  - directory: ${sys.base}/target/inbox/inventory
  - add custom message composer MessageComposer
  - includes all context properties (Bindings -> Message Composer)
- edit method decompose in class MessageCompose

STEP 3 - 2/2

Steps:

- assign File binding to Inventory reply composite service:
  - directory: ${sys.base}/target/outbox/inventory
  - add custom message composer MessageComposer
  - includes all context properties (Bindings -> Message Composer)
- edit method compose in class MessageCompose
- declare transformations

Test: Lab03Test

STEP 4

REST BINDING, JAVA INTERFACE

Goals: Integrate accounting application with OrderComponent

Steps:

- create AccountingService interface with method:

```
InvoiceIssueReply account(Order order);
```

- add component reference to OrderComponent use AccountingService contract
- promote AccountingService reference via rest binding:
  - RESTful Interface: AccountingResource
  - address: https://localhost:7171
  - authentication type: Basic, user: admin, password: foo, host: localhost, port: 7171
- edit OrderServiceBean

Test: Lab04Test

STEP 5

SOAP BINDING, WSDL

Goals: Promote OrderService as SOAP web service

- Receive : *src/test/resources/xml/soap-order.xml*
- Response: *src/test/resources/xml/soap-order-response.xml*

Steps:

- generate WSDL from JAVA class OrderService
  - disable: Use "wrapped" messages
- promote OrderService with generated WSDL as contract
  - assing SOAP binding
- JAXB transformation for order
- Create java transformation for response

Test: Lab05Test

STEP 6

DEPLOY APPLICATION

Checkout branch switchyard-06 and compile project:

> git checkout switchyard-06

> mvn clean install -DskipTests

JBoss Fuse console:

> features:addurl mvn:com.redhat.brq.integration/switchyard-seminar/0.0.1-SNAPSHOT/xml/features

> features:install switchyard-integration-course

Run test 6 on project:

> mvn clean verify -Dtest=Lab06Test