THILO FROTSCHER
Software Architect & Trainer

Enterprise Java, Web Services and XML

Are your services loosely coupled?

Web Services Training & Consulting
thilo@frotscher.com
http://www.frotscher.com

THILO FROTSCHER
Software Architect & Trainer

Introduction

- Independent Software Architect and Trainer
  - Enterprise Java
  - Web Services, XML & Interoperability
  - Enterprise Architectures
- Customizable inhouse training courses
  - Apache Axis2, Web Service Standards, Best Practices etc
- (Co-)author of three books on Web Services & SOA
- Articles for software development magazines
- Expert advisor for other books on XML & Web Services
- Currently based in New Zealand
- Always interested in exciting project opportunities ☺

Motivation

- SOA and Web Services have been hot topics for some time
- Web Service technology is widely used these days
- Many organizations have begun to adopt the approach to take advantage of the characteristics of services
  - One of the most widely known benefits is “loose coupling”

> *"One of the goals of a service oriented architecture is to ensure that components are as loosely coupled as possible. Loosely coupled architectures are much easier to maintain and reuse.”*
> (http://www.serviceoriented.org)

- So does that mean, just building services will always result in a loosely coupled system? Let’s see…

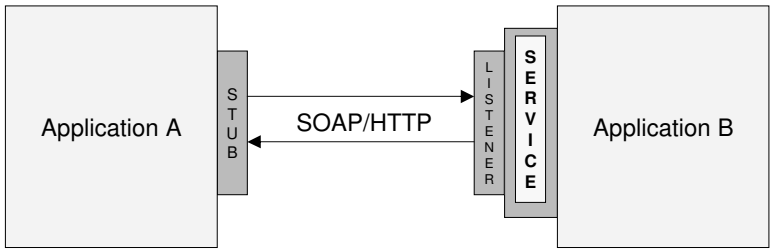
What does “coupling” actually mean?

> *Coupling refers to the interrelatedness and interdependencies between two or more components or web services.*
>
> *How easy is it to change something about service A without having to change service B?*
>
> *If many changes in service A require something in B to be changed, they are tightly coupled.*
>
> *If A can be changed easily without B being touched, they are loosely coupled.*

(http://www.serviceoriented.org)

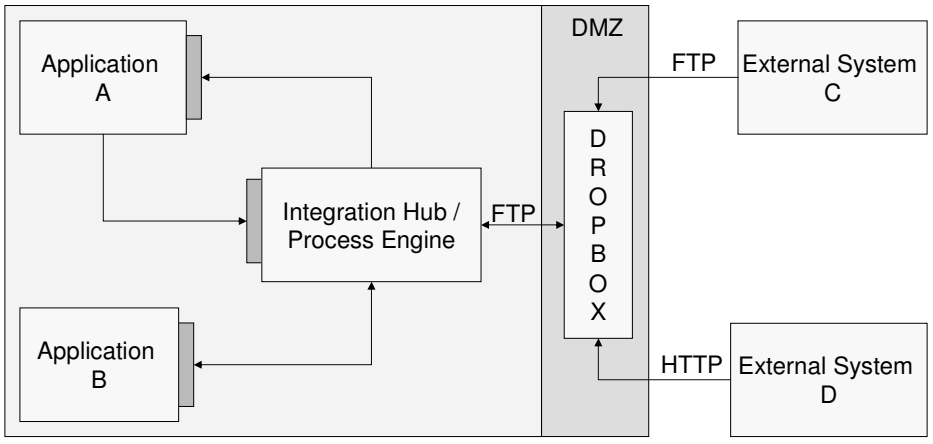
Typical usage pattern for Web Services

- Contract-First development…
  - Define the service interface (WSDL / XSD)
  - Generate a service skeleton and client stubs
  - Implement and deploy the service
  - Integrate the generated stub into a consuming application



- Now, are these applications loosely coupled?

A more complex scenario…

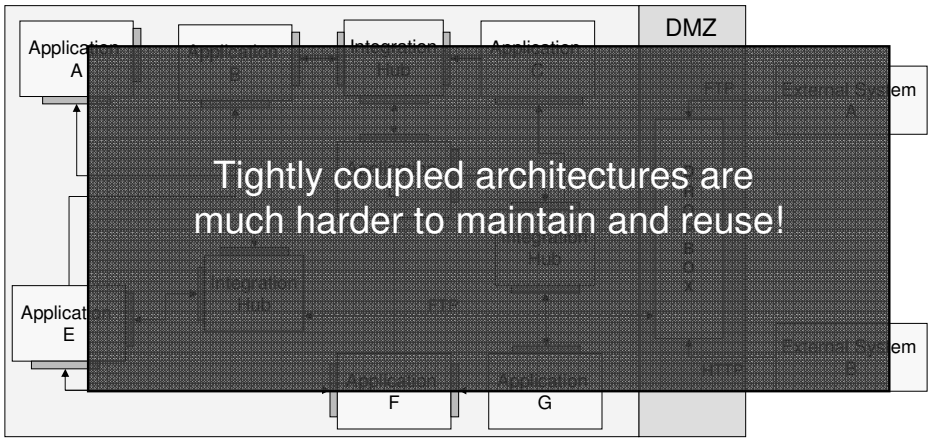


- Are these services loosely coupled?

Are your services loosely coupled?

A few years later…

- As more systems are integrated over time, the increasing number of point-to-point connections results in a big knot



Are your services loosely coupled?

…

RPC-type integration leads to architectures where applications are fairly tightly coupled together.

This also applies to Web Service technology.

Using SOAP for your communication does not mean you're automatically creating a loosely coupled system.

Are your services loosely coupled?

Making assumptions…

- Coupling can occur in many ways - the more assumptions parties make about each other, the tighter the coupling

- **Location** – what's the physical address of the other party?
- **Identity** – what other parties are interested my message?
- **Availability** – are the other systems actually running?
- **Data Format** – what data format(s) do they support?
- **Transport Protocol** – how to communicate with them?
- **Duration** – how long does it take for them to respond?

- To achieve loose coupling, remove these dependencies

Are your services loosely coupled?

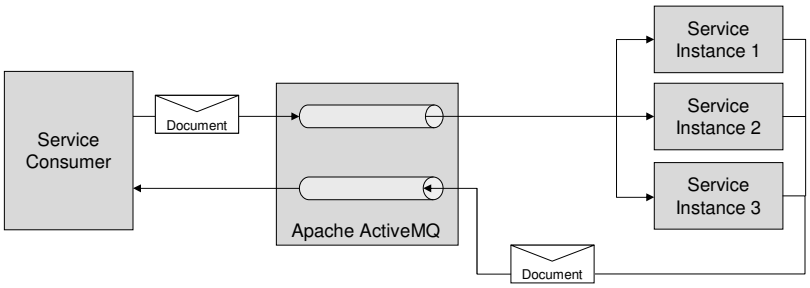
Messaging

- Integration based on sending messages to channels, rather than the invocation of specific procedures
- Communication is asynchronous – nobody needs to wait
  - Applications don't have to be running at the same time
- Messaging system is responsible for transferring the data in a reliable fashion
- Point-to-point channel vs. Publish-subscribe channel
- Command message vs. Document message

➢ Independence of location, identity, duration & availability

Are your services loosely coupled?

Building a more loosely coupled solution



- Rebuild RPC by sending SOAP messages in RPC format
- Or even better – use the Document format
- Need a Web Service framework that supports JMS for sending / receiving messages

Are your services loosely coupled?

SOAP message formats: RPC vs. Document

- Document format leaves decision to the receiver, which service method a message is dispatched to

```
<Reservation>
  <hotelCode>HDHSYD</hotelCode>
  <roomType>DBL</roomType>
  <arrival>05-07-2008</arrival>
  <departure>07-07-2008</departure>
  <guestName>Bart Simpson</guestName>
</Reservation>
```

? → `public Confirmation makeReservation(…)`

? → `public void updateReservation(…)`

? → `public void sendInvoiceForReservation(…)`

- RPC message format results in tight coupling!

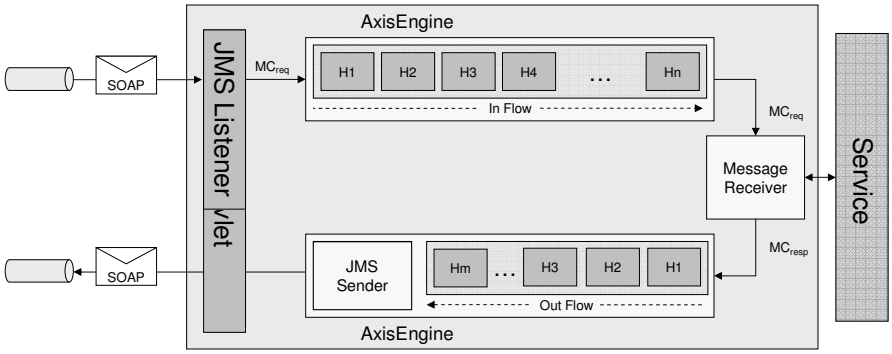
```
<makeNewReservation>
  <hotelCode>HDHSYD</hotelCode>
  <roomType>DBL</roomType>
  <arrival>05-07-2008</arrival>
  <departure>07-07-2008</departure>
  <guestName>Bart Simpson</guestName>
</makeNewReservation>
```

```
public Confirmation makeNewReservation (
                  String hotelCode,
                  String roomTye,
                  Date arrival
                  Date departure,
                  String guestName);
```

- SOAP was developed with messaging in mind - most applications use it for synchronous RPC-style calls though

Example: Receiving JMS messages with Axis2

- Using a different transport is just a matter of configuration
- The service implementation does not need to be changed
- Changing service consumers for using JMS is equally easy



Message correlation with WS-Addressing

```
<soap:Envelope xmlns:wsa="http://www.w3.org/2005/08/addressing"
       xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <wsa:MessageID>urn:uuid:E9D931DE6E606E115D116722946291042</wsa:MessageID>
      ...
  </soap:Header>
  <soap:Body>
    ...
  </soap:Body>
</soap:Envelope
```

```
<soap:Envelope xmlns:wsa="http://www.w3.org/2005/08/addressing"
       xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <wsa:MessageID>urn:uuid:E9D931DE6E606E115D116727390803123</wsa:MessageID>
    <wsa:RelatesTo wsa:RelationshipType="http://www.w3.org/2005/08/addressing/reply">
        urn:uuid:E9D931DE6E606E115D116722946291042
    </wsa:RelatesTo>
      ...
  </soap:Header>
  <soap:Body>
    ...
  </soap:Body>
</soap:Envelope
```

Are your services loosely coupled?

Where to send the reply?

```
<soap:Envelope xmlns:wsa="http://www.w3.org/2005/08/addressing"
       xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <wsa:To>jms:/queue?destination=jms/MyApp2/NewReservationsQueue</wsa:To>
    <wsa:ReplyTo>
       <wsa:Address>jms:/queue?destination=jms/MyApp1/ConfirmationQueue</wsa:Address>
    </wsa:ReplyTo>
    <wsa:FaultTo>
      <wsa:Address>jms:/queue?destination=jms/MyApp1/FaultQueue</wsa:Address>
    </wsa:FaultTo>
    ...
  </soap:Header>
  <soap:Body>
    ...
  </soap:Body>
</soap:Envelope
```


Are your services loosely coupled?

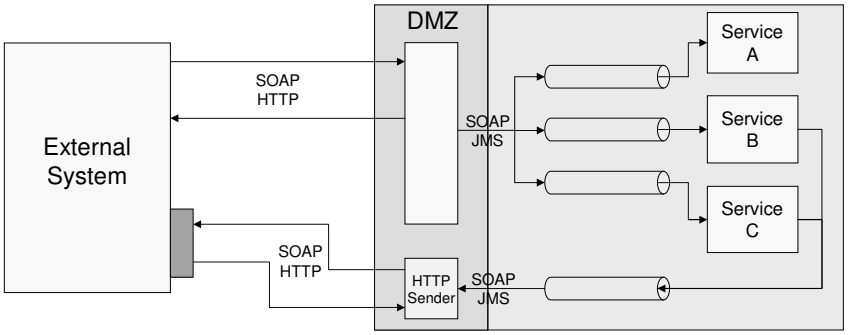
So what about HTTP?

- SOAP was designed to be transport-independent – the majority of applications use HTTP
- Seems to be a natural choice: most commonly used protocol on the web and it can sneak through firewalls
- But HTTP was never intended to be used by applications to communicate with each other!
  - inherently unreliable
  - designed for synchronous retrieval of documents
- Not well suited for asynchronous communication
- The use of alternative transport protocols for SOAP is becoming more widespread
- Problem: too many applications take HTTP for granted!

Are your services loosely coupled?

Don't rely on any transport protocol

- Messages might be transported using various protocols on their way to the ultimate receiver
- All metadata should be stored in the SOAP Header
- Use WS-Addressing to specify recipient(s)



Are your services loosely coupled?

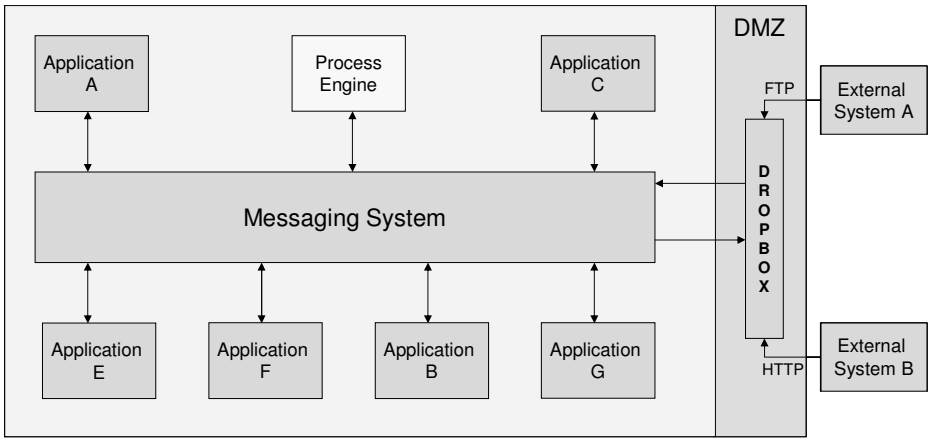
More concepts & features of messaging systems

- Pipes and Filters
  - add components that process the message on its way
- Transformers
  - Translate between different data formats
  - Move messages from one transport protocol to another
- Routers
  - Message might travel through several channels on the way to its ultimate receiver
  - Completely decouples sender from receiver: the sender is unaware of the channel the message is finally sent to
- Aggregators and Splitters
- Process Engines
- Channel Adapters

Are your services loosely coupled?

A loosely coupled solution

- Remember the big knot of point-to-point connections?



Are your services loosely coupled?

Message Bus

- Good solution for cases where multiple applications are supposed to work together in a loosely coupled way
- Add / remove applications without affecting the others

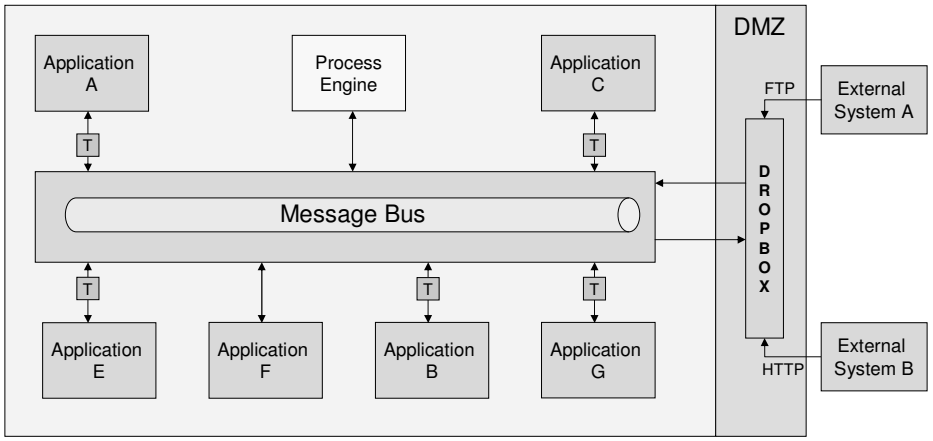
- Canonical data model to reduce the number of transformers
- Shared set of shared interfaces / command command set
- Messaging infrastructure

- Adapters for standard technologies or packaged applications

Are your services loosely coupled?

Enterprise Architecture with a Message Bus

- The bus as a universal connector routes messages to the underlying systems



Are your services loosely coupled?

Enterprise Service Bus

- Adding an abstraction layer around the idea of messaging
- No clear definition for the term “ESB”
- ESBs typically include
  - multiple adapters for different transport protocols and standards
  - standards-based adapters for integrating legacy apps (e.g. JCA)
  - support for WS-* standards
  - support for orchestration and/or choreography
  - routing, security, validation, transformation
  - monitoring facility
- Various commercial ESBs available
- Open Source ESBs include
  - Mule
  - Apache Synapse
  - Apache ServiceMix

Are your services loosely coupled?

Contracts and Interfaces

- Versioning is a common problem in software development
- What happens if an interface needs to be changed?
- What if we need to add operations or send more data?

- WSDLs and XML Schemas should ideally be backward- and forward-compatible
- Introduce a versioning scheme for your organization
  - Minor versions should be compatible
  - Major versions can break compatibility
- Create extensible schemas

Are your services loosely coupled?

Don't make service interfaces too generic!

- This is a bad practice…

```
<wsdl:definitions ...>
  <wsdl:types>
    <xsd:schema ...>
      <xsd:element name="message" type="xsd:string"/>
    </xsd:schema>
  </wsdl:types>
  <wsdl:message name="processMessageRequest">
    <wsdl:part name="input" element="tns:message"/>
  </wsdl:message>
  <wsdl:message name="processMessageResponse">
    <wsdl:part name="output" element="tns:message"/>
  </wsdl:message>
  <wsdl:portType name="MyGenericPortType">
    <wsdl:operation name="processMessage">
      <wsdl:input message="tns:processMessageRequest"/>
      <wsdl:output message="tns:processMessageResponse"/>
    </wsdl:operation>
  </wsdl:portType>
</wsdl:definitions>
```

Summary

- So are your services loosely coupled? ☺
- Using Web service technology does not automatically lead to loosely coupled systems!
- Web services can be used for both integration styles: RPC and Messaging
- Always remember that loose coupling and asynchronous communication makes your system more complex
  - often a tighter coupling is fine (or good enough)
- Tightly coupled solutions are typically more efficient, but also brittle and more vulnerable to change
- For enterprise architecture and enterprise integration avoid a growing knot of point-to-point connections

Thank you very much for your attention!

Web Services Training & Consulting
thilo@frotscher.com
http://www.frotscher.com