

Business Track Session 2, Talk 2

**Architectural Decisions as Service Realization
Methodology in Model-Driven SOA
Construction**

**From Analysis-Level Process Models to
Service Abstractions of Quality and Style**

**Olaf Zimmermann
Executive IT Architect
Zurich Research Lab
olz@zurich.ibm.com**

Abstract

On Service-Oriented Architecture (SOA) delivery projects, practitioners concern themselves with the characteristics of good services and how such services can be designed. For instance, they look for advice regarding interface granularity and criteria to assess whether existing software assets are fit for reuse in SOA environments. As for all nontrivial modeling problems, there are no straightforward answers to these questions; a full-fledged service modeling methodology is required, consisting of service identification, specification and realization techniques. Service identification and specification are well covered by methodologies such as Service-Oriented Modeling and Architecture (SOMA); for service realization, architectural decision models can be leveraged. Unlike other approaches to documenting software architectures, architectural decision models capture the expert knowledge leading to certain designs rather than actual designs.

At present, the construction of architectural decision models is an education- and labor-intensive undertaking. Architectural decisions are therefore often taken without model support; the rationale behind the decision making is not captured. In this talk, we position SOA-specific architectural decision models as a prescriptive service realization technique and present an SOA decision catalog harvested from a number of full-scope SOA delivery projects. For illustration purposes, we will use two large-scale SOA and Web services deployments, the BPEL enablement of the order management application of a telecommunications wholesaler, and the Web services externalization approach followed by a shared service provider in the finance industry. We will look at drivers for the projects as well as the solution architectures, and share some of the lessons learned.

Agenda

- SOA definition, principles and patterns revisited
- Project examples
- SOA decision modeling overview
- Transaction management related decisions
- Summary

Agenda

- **SOA definition, principles and patterns revisited**
- Project examples
- SOA decision modeling overview
- Transaction management related decisions
- Summary

What is SOA?

“Service-Oriented Architecture (SOA) is different things to different people”

Business
Executive,
Consultant

- A *set of services* that a business wants to expose to their customers and partners, or other portions of the organization
- An *architectural style* which requires a service provider, requestor and a service description
- A *set of architectural principles, patterns* and criteria which address characteristics such as *modularity, encapsulation, loose coupling, separation of concerns, reuse, composability and single implementation*
- A *programming model* complete with standards, tools and technologies such as Web services

Architect

Programmer

SOA principles and patterns by role

**Business-Aligned Service Descriptions
(Interface Contracts)**

**Business
Executive,
Consultant**

**Separation of Concerns
and Modularity**

**Loose Coupling
and Messaging**

Architect

**Service Repository/
Registry**

Service Composition

**Enterprise Service Bus
(ESB)**

Programmer

**Development
Tools**

**Execution Runtimes
(e.g. J2EE, SCA)**

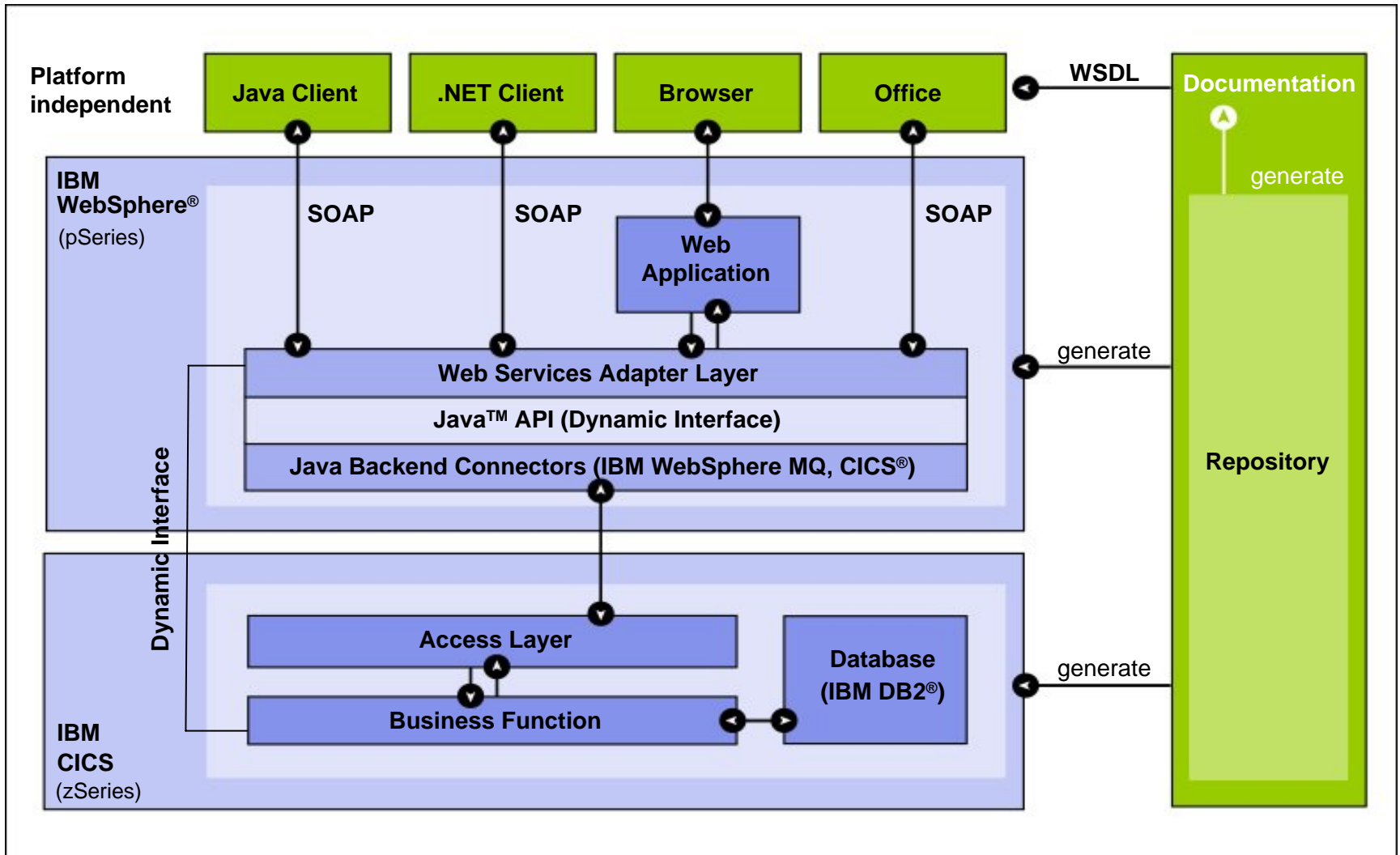
**XML & Web Services
Standards**

**Internet
Protocols**

Agenda

- SOA definition, principles and patterns revisited
- **Project examples**
- SOA decision modeling overview
- Transaction management related decisions
- Summary

Core banking SOA with Web services



Multi-channel order management SOA in the telecommunications industry

- **Functional domain**

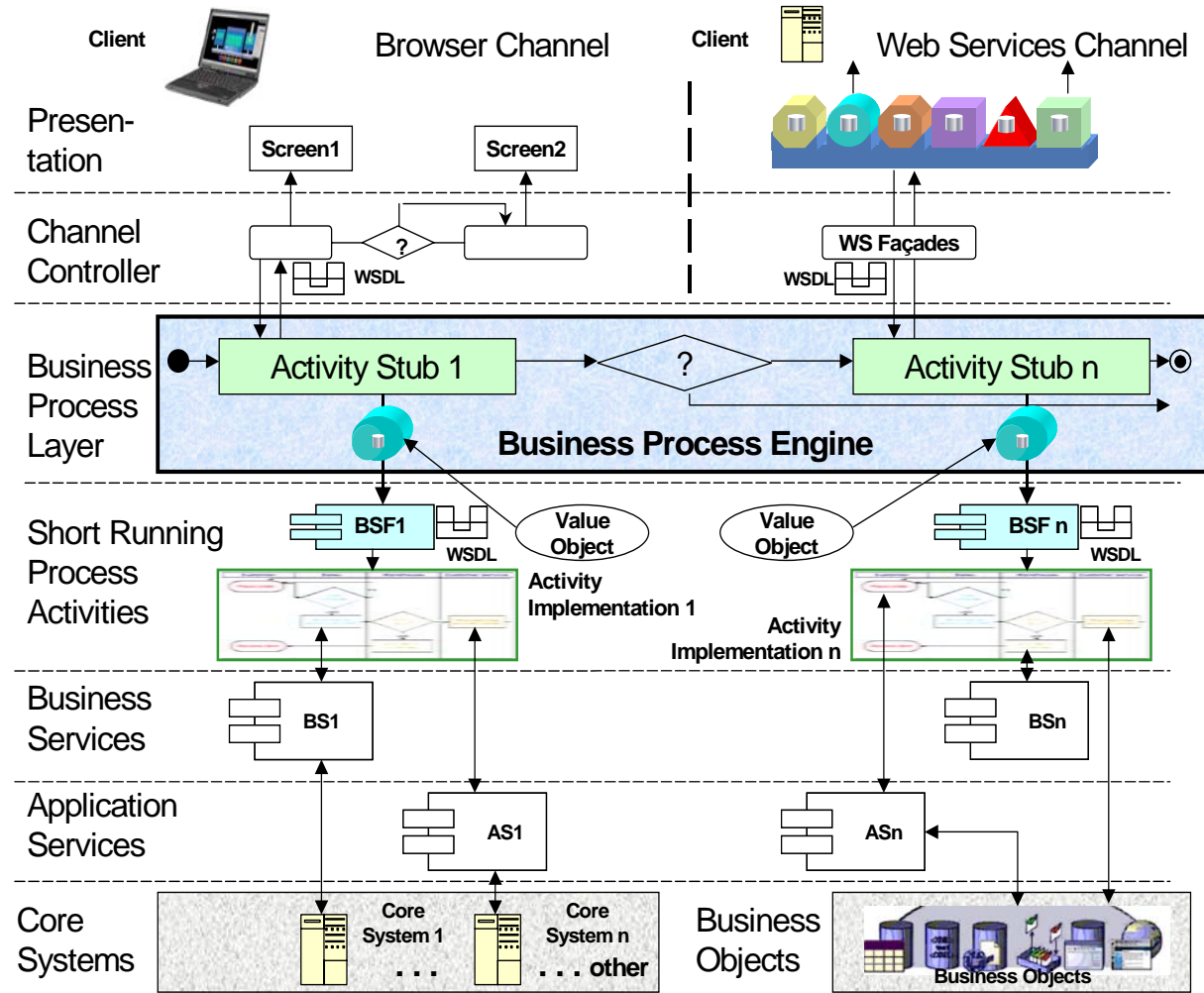
- Order entry management
- Two business processes: new customer, relocation

- **Main SOA drivers**

- Deeper automation grade (e.g. compensation)
- Services shared within and between domains

- **Service composition**

- Top-down from retailer interface and process
- Bottom-up from existing wholesaler systems



Lessons learned on these and other projects

- Requirements models always are incomplete
- Technology evolves rapidly, many variation points exist
- Reference architectures capture proven design elements
- Model-driven software development paradigm enables team communication/collaboration and improves quality

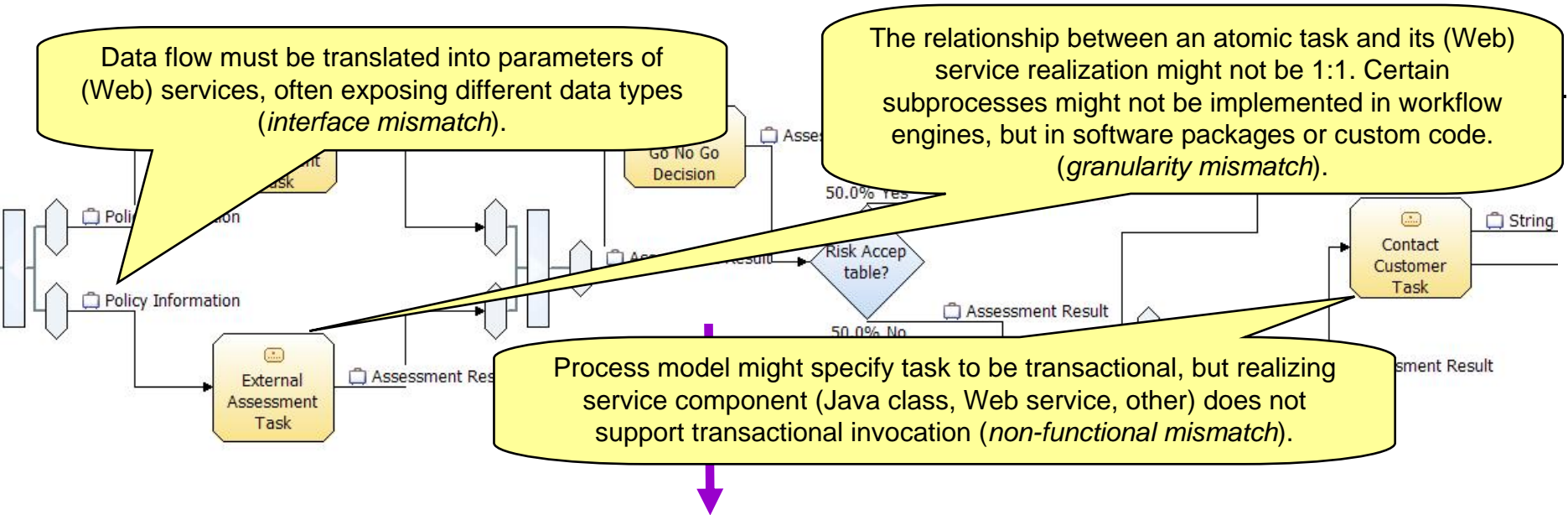
Engineering approach to service design is still required

Architectural decision models and patterns are a powerful combination for process realization decision making

Agenda

- SOA definition, principles and patterns revisited
- Project examples
- **SOA decision modeling overview**
- Transaction management related decisions
- Summary

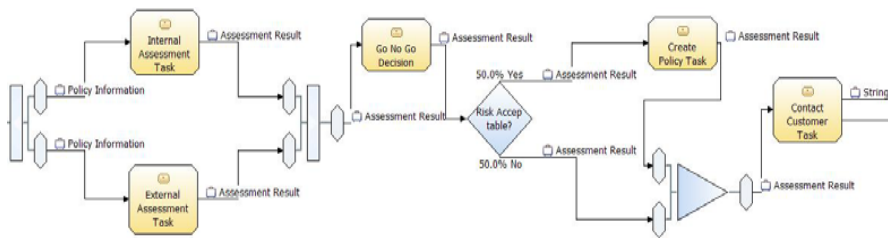
Insurance Policy Processing Example



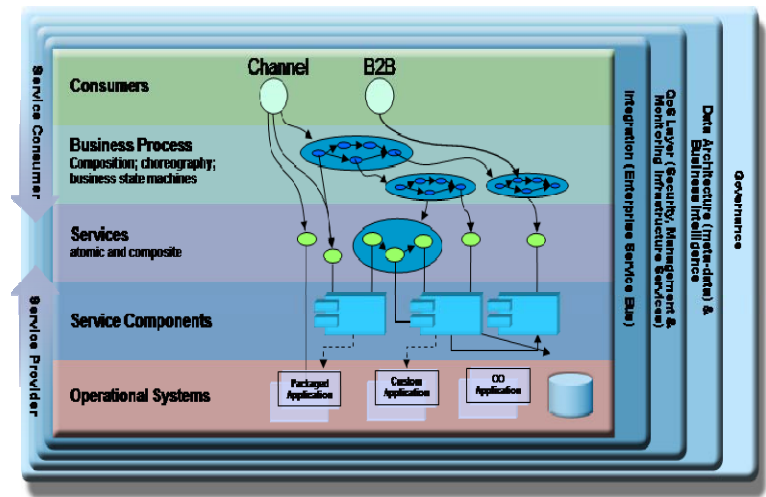
- **Services Architect** concern: how can the process be realized in IT?
 - Many process realization decisions, e.g. *composition decision*:
 - Can an existing service component be used to implement an atomic task?
 - Is it possible to introduce adapter or mediation technology to overcome *granularity, interface, and non-functional mismatches*?
 - Or does a new service component have to be implemented?

Architectural decisions can be identified in requirements models and reference architectures.

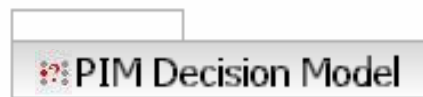
- Modeling *architectural decisions* is an emerging topic area in software architecture research, e.g. [Kruchten 2006]
 - Roots in design decisions research (1990s) and practitioner methodologies such as the IBM Global Services Method (free text or form)
 - Architectural decisions capture key architecture design issues along with the identified alternatives, their tradeoffs, and the rationale behind a certain design



Analysis-Level Business Process Model (BPM)



SO(M)A Reference Architecture (5+2+2 layers)



Platform-Independent Model (PIM) of Required Decisions

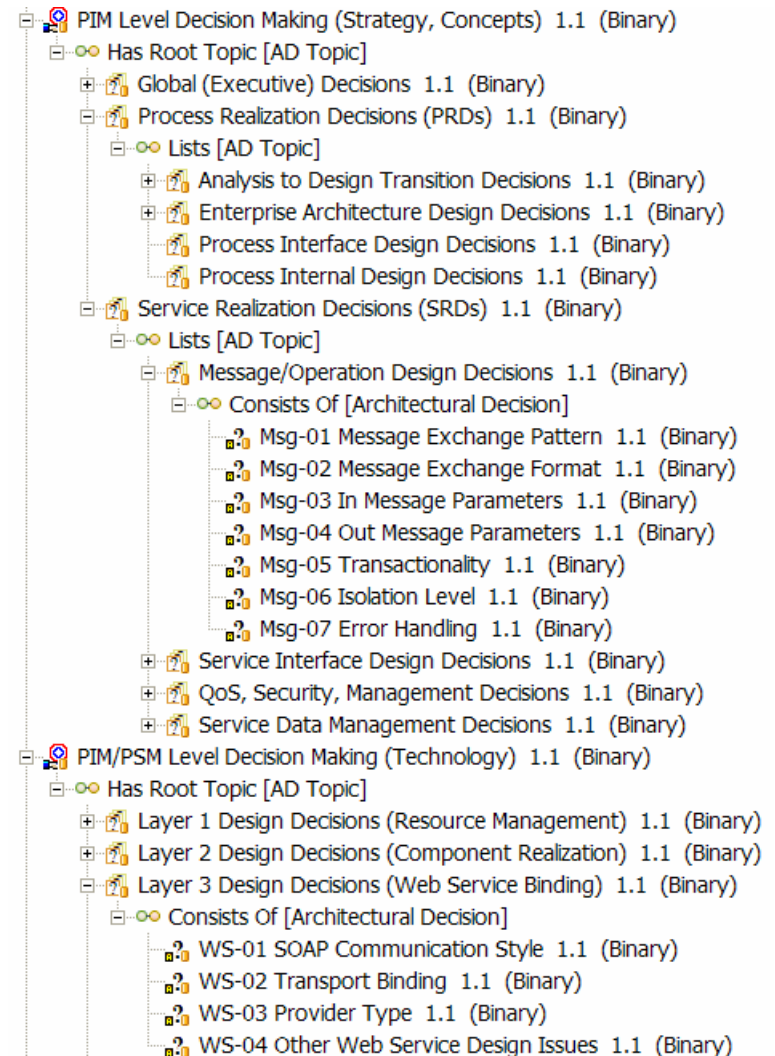
SOA Decision Tree

- Three levels (inspired by MDA)
 - Conceptual (pure PIM): abstract patterns
 - Technology (PIM/PSM): Web services and others
 - Assets/Products (pure PSM): IBM and others
- Conceptual (PIM) level
 - Structured according to [Kruchten] and SOMA ontologies
 - Many intra- and inter-level dependencies
- Technology (PIM/PSM) and Asset/Product (PSM) levels
 - Structured according to 5+2 SOMA / SOA Reference Architecture layers
 - Layers 8 and 9 factored in
 - Clear separation between technology related decisions (core) and vendor-specific refinements



Selected (Web) Services Decisions

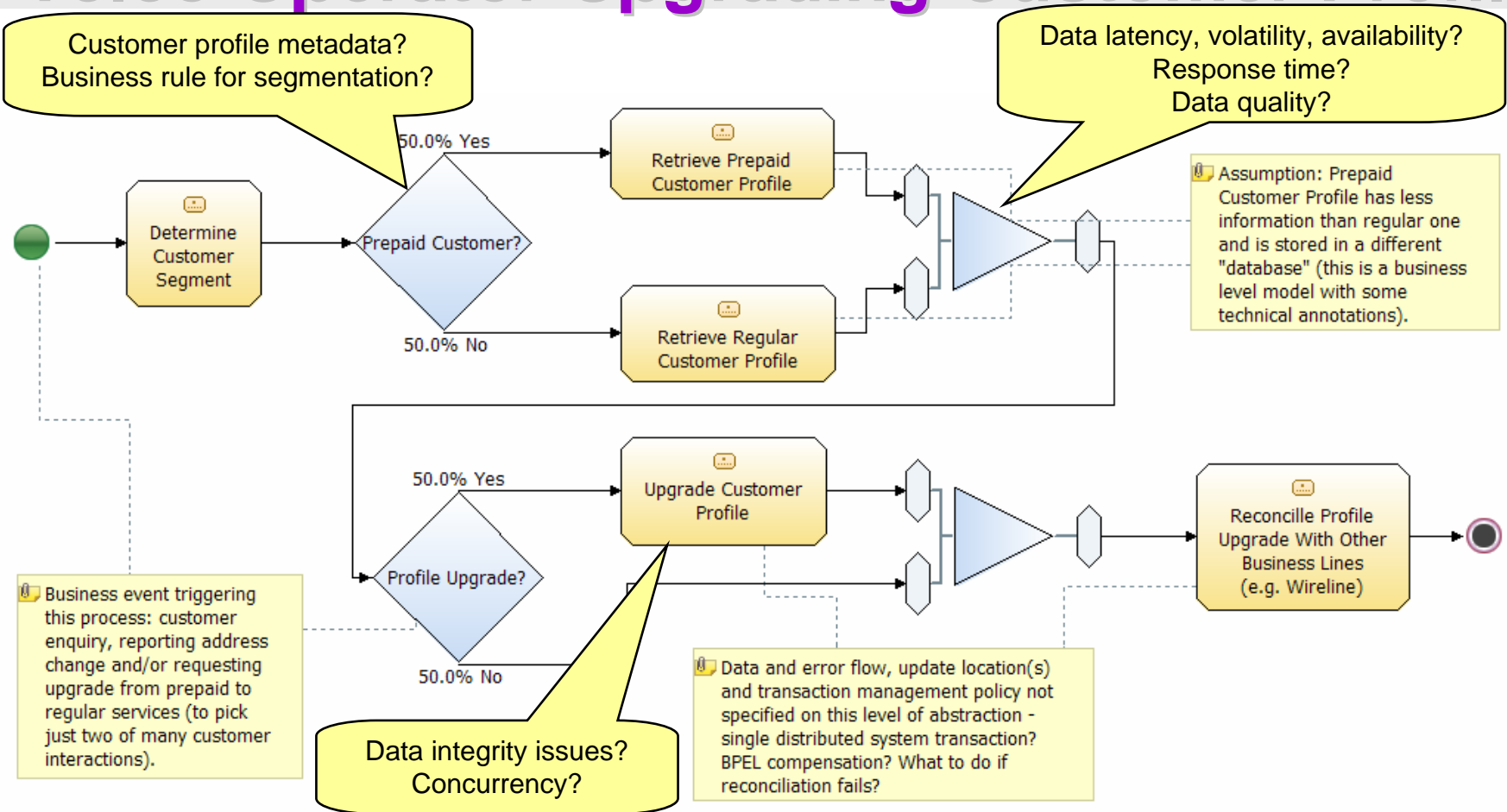
- Message (Msg)-01:
Message Exchange Pattern
 - One way or request-response
 - Request-response best covered by WS-I profile and tools (as of today)
- Msg-02:
Message Exchange Format
 - SOAP vs. REST vs. custom
 - SOAP recommended (WS-I), REST popular in open source community
- Web Service (WS)-01:
SOAP Communication Style
 - rpc/encoded, rpc/literal, (wrapped) document/literal
 - WS-I has banned rpc/encoded, still used by older SOAP engines
- WS-02: *Transport Binding*
 - HTTP or message queues (JMS)
 - Both have their place



Agenda

- SOA definition, principles and patterns revisited
- Project examples
- SOA decision modeling overview
- **Transaction management related decisions**
- Summary

Data Integrity Requirements Example: Telco Operator Upgrading Customer Profile



- Read vs. write access to “customer data” distributed over several locations
- Not all requirements specified on this level – many architecture alternatives

Key Decision: Transaction Management Pattern

- Each decision node:
 - Is uniquely identified and scoped
 - Describes a concrete problem to be solved
 - Refers to literature (short overview, tutorial)
 - Gives best practices recommendations
 - Lists dependencies from/to related decisions
- Alternatives are architectural patterns
 - Source: literature (when available) or to be developed (in this case)
 - [1] Simple Service Invocation (no transactionality)
 - [2] Atomic Service Invocation (*new* transaction per service call)
 - [3] Distributed Transaction (all service calls *join* a single transaction)

EAD-04 Transaction Management Pattern for PolicyProcessing

EAD-04 Transaction Management Pal [Architectural Decision]

Raw Notes

This decision is about selecting a the transactional workflow pattern on process level. Here: concurrently executing activities such as InternalAssessmentTask and ExternalAssessmentTask might operate on same data e.g. PolicyRecord, which leads to a potential data integrity issue.

Architectural Decision Properties

Name	Value
resolved	1/1 true

Short Description of Decision to be Made

Available patterns:

- [1] Simple Service Invocation (SRD Msg-05: [1] none)
- [2] (default) Atomic Service Invocation (SRD Msg-05: [2] new)
- [3] Distributed Transaction (SRD Msg-05: [3] join)
- [4] Idempotent Composition
- [5] Long-Running Invocation

Problem Statement

Details to be defined on lower levels, but can decide on project-wide architectural principle/pattern. In principle, decision is between system and business transactions, there can be one system transaction (transaction manager enforcing ACID characteristics) or multiple system transactions per process (pattern based).

Motivation

See TP literature such as chapter 7 in [ProdWF] for introduction to all related issues. See "SOA Decisions: Transaction Management Pattern" paper for short overview (work in progress). Devil is in the detail - protocol vs. wiring, impact of loose coupling, etc. Refer to famous "Piranha" quote from Fowler!

Recommendation

Make each atomic service transactional. Avoid keeping transaction open during client interaction. Do not let unit of work cross layer boundaries. Use BPEL compensation for business-level undos at least for long running flows. Design error cases explicitly, do not forget errors during error recovery.

Outcome of Decision Making

[3]
@scope=process, @phase=solution outline, @owner=lead architect, @enforcement=PSM transformation (pattern aware)

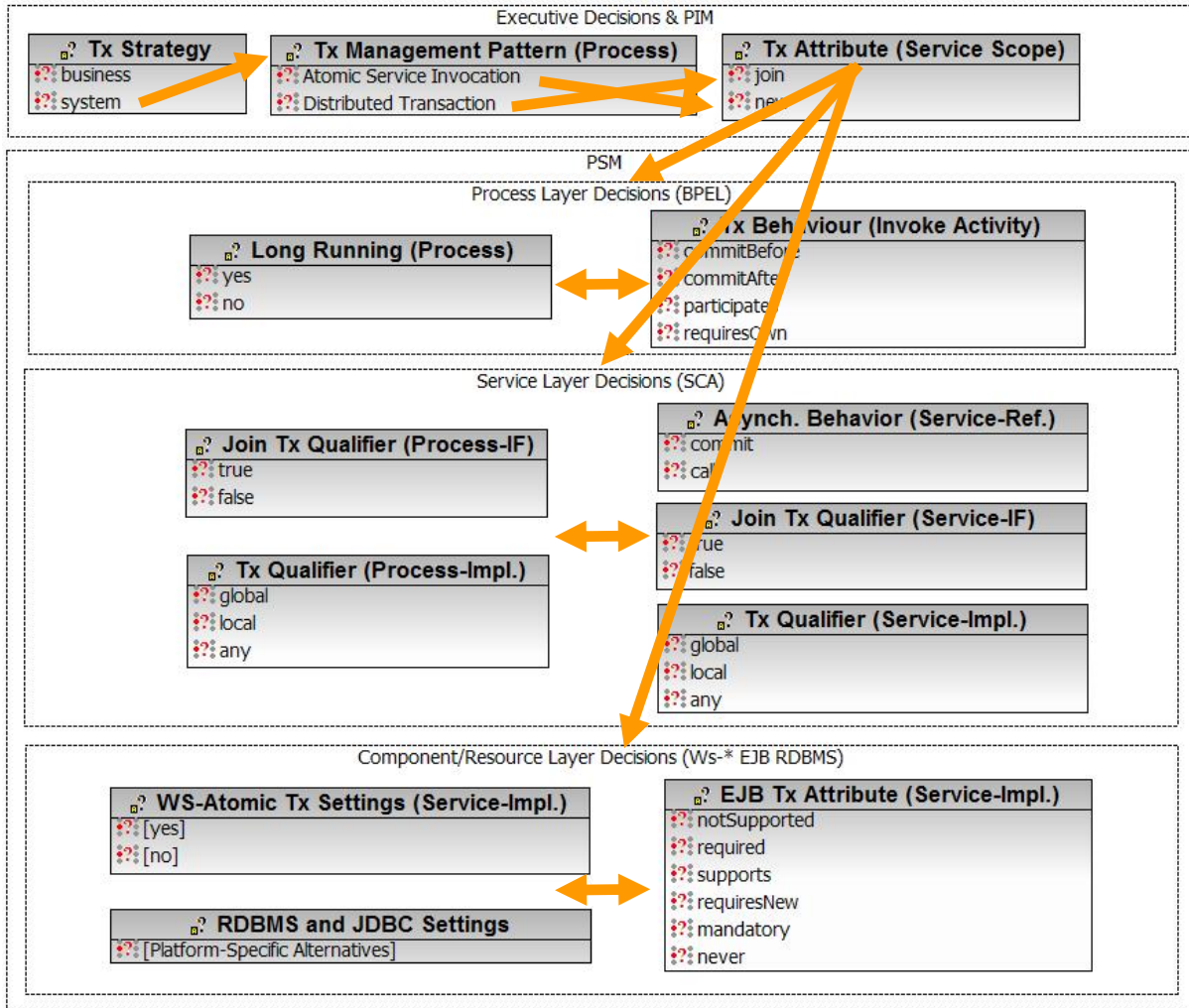
Justification

In line with best practices established by SWG lab services.

Decision Ma... Assumptions ... Traceability Foo Risks Extra Attrib... Appears In

Decision Layers and Dependency Modeling

Decision Dependency and Constraint Management Required



Platform-Independent Model (PIM) of Required Decisions
(Tx – Transaction)

Platform-Specific Model (PSM) of Required Decisions
(here: WebSphere Process Server)

Business Process Execution Language (BPEL)
(join ->participates, new->requiresOwn)

Service Component Architecture (SCA)

Web Services (WS)

Relational Database Management System (RDBMS)

Enterprise Java Beans (EJB)

Agenda

- SOA definition, principles and patterns revisited
- Project examples
- SOA decision modeling overview
- Transaction management related decisions
- **Summary**

Summary

- Experience from numerous successful SOA projects has been captured informally so far
 - Assets and knowledge gained on these projects have been harvested and captured in reference architectures, pattern toolkits, best practices documents, etc.
 - SOA decision modeling is a more formal and structured approach
- Designing “good” services continues to be hard
 - There are many design alternatives and tradeoffs
 - Business process transaction management is particularly hard

Architectural decision models and patterns are an excellent means of organizing the required (& available) knowledge

Text Book Authored by IBM Practitioners: Perspectives on Web Services (PoWS)

- The IBM architects for the Sparkassen Informatik and other projects have written a book, with an expansion on their ideas for Web services architecture design. Emphasis is on lessons learned and best practices.
- **Perspectives on Web Services**, by Olaf Zimmermann, Mark Tomlinson, Stefan Peuser, Springer 2003, ISBN 3-540-00914-0
- Foreword by Grady Booch, IBM Fellow and Chief Scientist IBM Rational
- Website support incl. 2005 updates: www.perspectivesonwebservices.de

