Business Integration Technologies

Leveraging Reusable Architectural Decision Models as a Design Method for Service-Oriented Architecture Construction

ECOWS Keynote
Nov 28, 2007

Olaf Zimmermann
IBM Zurich Research Lab
olz@zurich.ibm.com

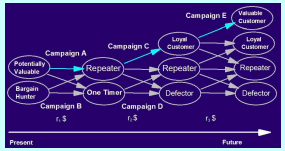
Abstract

Numerous research efforts regarding service-oriented computing and Web services are currently underway. Most of these efforts focus on advancing the state of the art in infrastructure and programming model design; service composition, dynamic matchmaking, and service virtualization are particularly popular topics. Few researchers investigate design-time aspects such as the identification, specification, and realization of business-aligned services of quality and style. However, the absence of a prescriptive service realization method is a key inhibitor for a broad and sustainable adoption of service-oriented architecture concepts and Web service technologies in practice.

In this talk, we review several existing service engineering methods from academia and industry and discuss how to leverage techniques from object-oriented analysis and design. Using two case studies from the finance and the telecommunications industries, we show that none of the existing approaches meets all practical requirements. We introduce reusable architectural decision models as a design method for service realization that is both comprehensive and comprehensible, and present excerpts of such a pre-populated SOA Decision Model. We close with a call to action for further research.

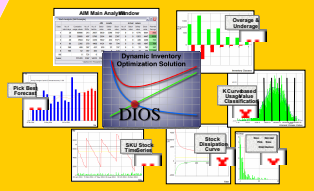
IBM Research Worldwide

Zurich Research Laboratory (ZRL) – Computer Science
Selected Highlights in Services and Software Research



Customer Equity Lifetime management solution for B2C marketing

Secure Trade Lane solution guaranteeing security and efficiency for worldwide transportation

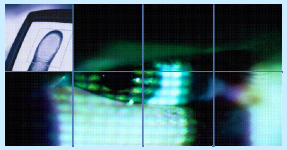


Inventory Management & Optimization solution

eclipse

No 1 platform for integrated software development tools

Federated Identity Management and Web services security in IBM Tivoli and WebSphere Products



***e-ID** solutions: Secure Identity Cards, Documents, and Processes for Governments and Enterprises*

Business Integration Technologies @ Zurich Research

- Focus on selected aspects of Business-Driven Development (BDD)
  - Behavioral modeling and algorithmic reasoning (Business-IT transition)
- Product-focused work
  - WebSphere Business Modeler (WBM) and Integration Developer (WID)
  - In 2007: ARIS import, SESE PST Infrastructure, transformation and error detection plugins
- Technology-focused work
  - Process version merging, role of object-life cycles in BDD, soundness of process models, model constraint patterns, BPMN 2.0 execution semantics
- Customer- and professional services-oriented work
  - Support of technical pre-sales, customer workshops, educational events
  - Architectural decision making in SOA design

Agenda

1. Motivating case studies and definitions
2. Requirements for service design methods
3. General-purpose and SOA-specific design methods
4. Architectural decision modeling concepts
5. SOA Decision Model content
6. Decision capturing wiki tool
7. Future work

Agenda

1. **Motivating case studies and definitions**
2. Requirements for service design methods
3. General-purpose and SOA-specific design methods
4. Architectural decision modeling concepts
5. SOA Decision Model content
6. Decision capturing wiki tool
7. Future work

What is a Service-Oriented Architecture (SOA)?

No single definition – “SOA is different things to different people”

Business Domain Analyst

- A *set of services* that a business wants to expose to their customers and partners, or other portions of the organization.

IT Architect

- An “architectural style” [SEI] which requires a *service provider*, a *service requestor* (consumer) and a *service contract* (a.k.a. client/server).
- A set of “architectural patterns” [SEI] such as *enterprise service bus*, *service composition*, and *service registry*, promoting principles such as *modularity*, *layering*, and *loose coupling* to achieve design goals such as separation of concerns, reuse, and flexibility.

Developer, Administrator

- A *programming and deployment model* realized by standards, tools and technologies such as Web services and Service Component Architecture (SCA).

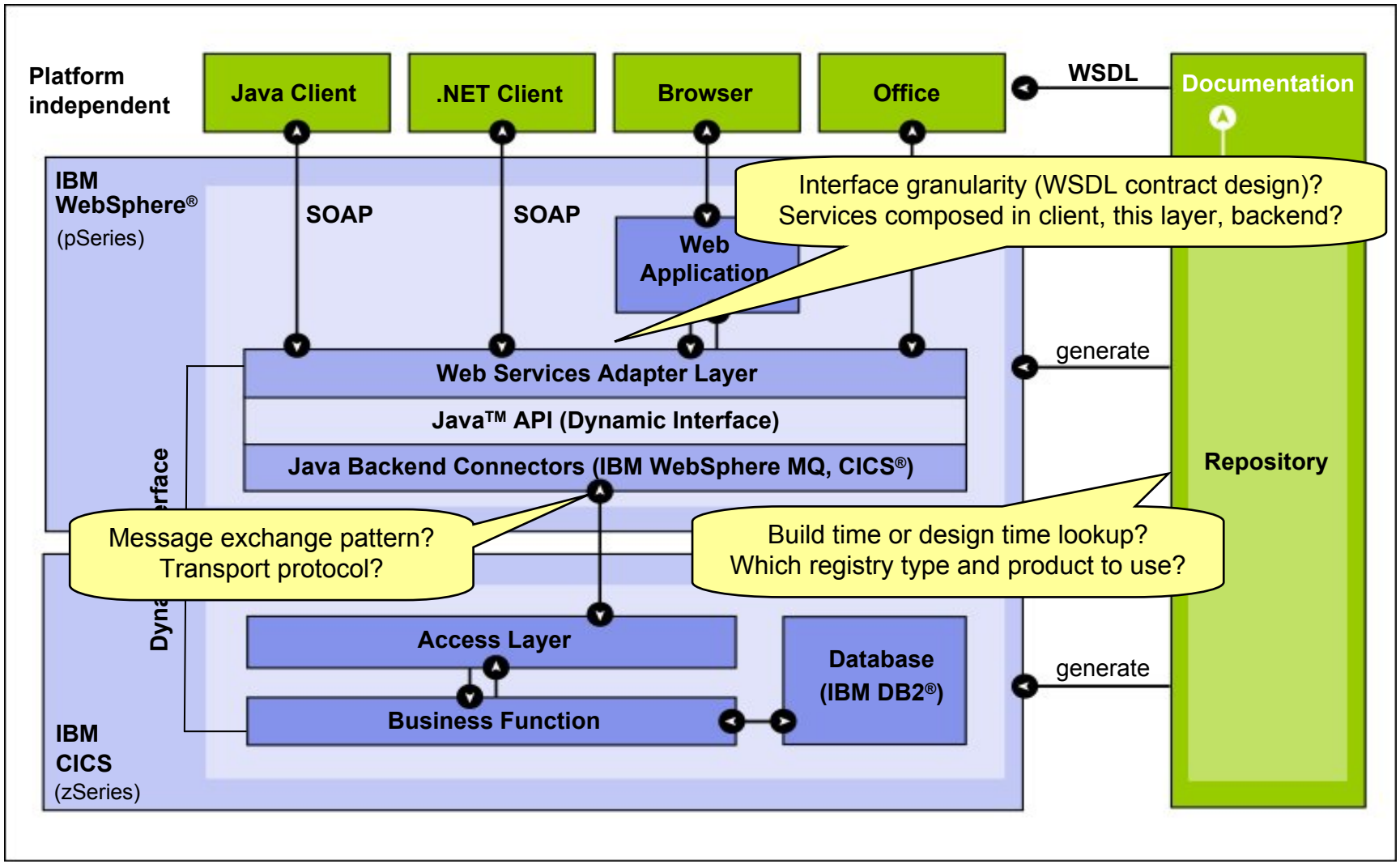
[SEI] *Bass, Clements, Kazman,* **„Software Architecture in Practice“**

Definitions adapted from **http://www.ibm.com/developerworks/webservices/library/ws-soa-enterprise1/**

Architectural Styles, Quality Attributes, and Patterns

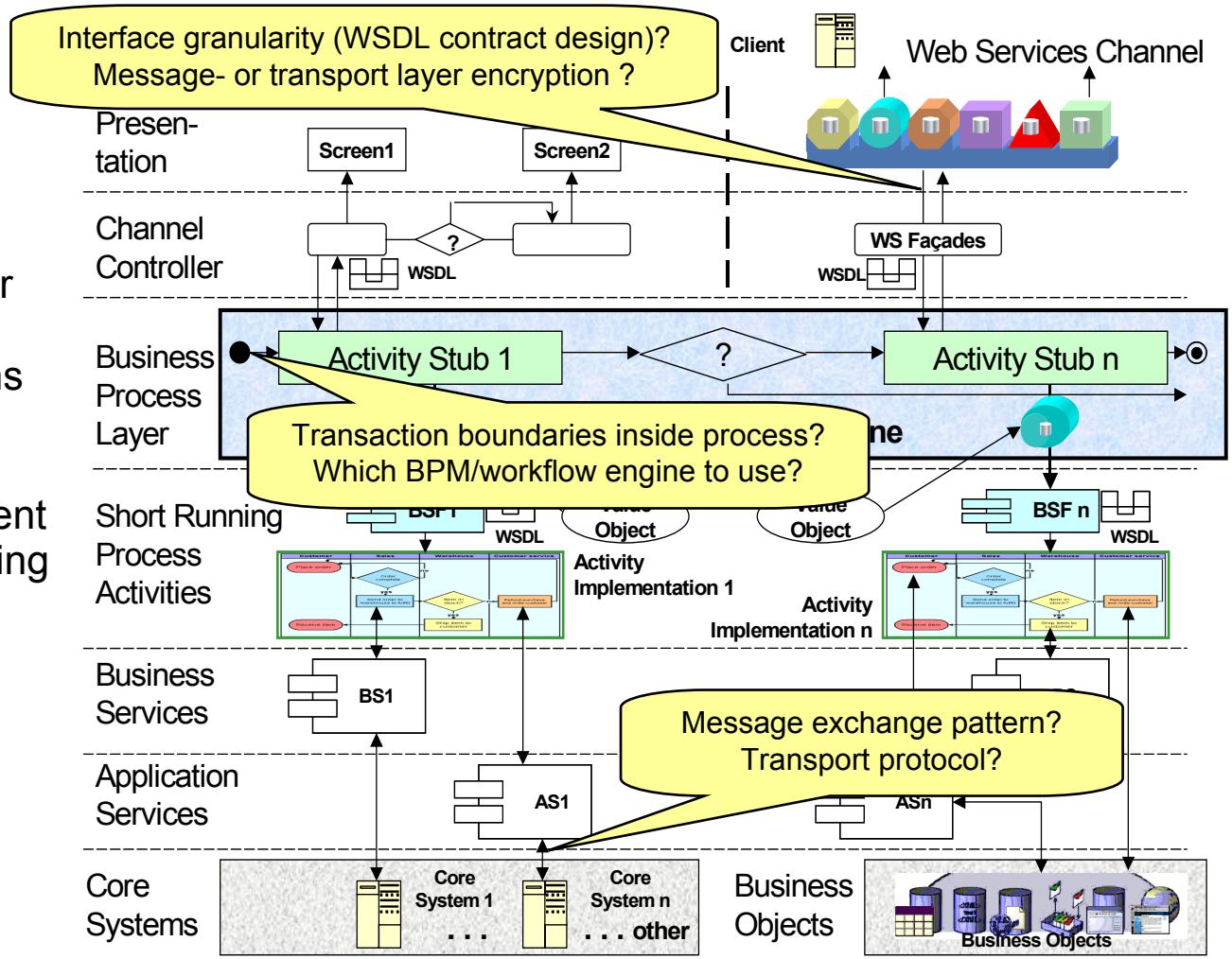
- L. Bass/P. Clements/R. Kazman “Software Architecture in Practice”
  - Architectural style expresses intent through top-level patterns and principles
  - Architectural drivers include software *quality attributes*
- G. Booch “Software Architecture Handbook” (www.booch.com/architecture)
  - Notion of application genres and design *forces*
  - Architectural digs, 2000+ patterns collected
- Architectural, design, integration patterns (mined in open community efforts)
  - General repeatable *solution* to a commonly occurring *problem* in software design
  - Gang of Four “Design patterns”, POSA series (Volumes 4 and 5 out now)
  - G. Hohpe, “Enterprise Integration Patterns”, www.eaipatterns.com (SOA patterns)

Core Banking SOA with Web Services [OOPSLA 2004]



Multi-Channel Order Management SOA in the Telecommunications Industry (in production since Q1/2005) [OOPSLA 2005]

- **Functional domain**
  - Order entry management
  - Two business processes: new customer, relocation
  - Main SOA drivers: deeper automation grade, share services between domains
- **Service design**
  - Top-down from requirement and bottom-up from existing wholesaler systems
  - Recurring architectural decisions:
    - Protocol choices
    - Transactionality
    - Security policies
    - Interface granularity



Recurring Service Design Issues and Decisions

1. Selection and adoption of application and integration patterns
2. Technology choices – protocols, containers, operating systems, etc.
3. Product selection and configuration

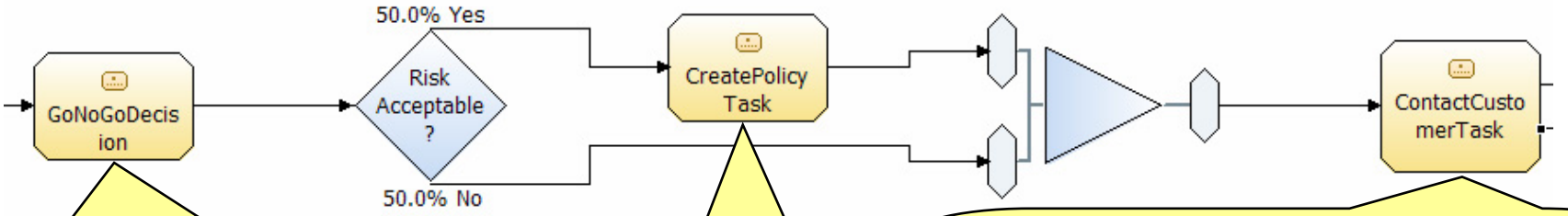
- Hundreds, if not thousands, of such decisions per *role*, *phase*, *scope*:
  - Many decisions already made before project starts
    - By client – previous projects, legacy systems
    - By other parties – software vendor, IT consultants
  - Many *forces* that the influence decision making, often contradictory [Booch]
  - Numerous complex *dependencies* between the decisions [Kruchten]
  - Never enough time to analyze all *options* in detail
  - Pattern literature is overwhelming

[Booch] **„Handbook of Software Architecture“**, http://www.booch.com/architecture

[Kruchten] **„Building Up and Reasoning About Architectural Knowledge“**, *QOSA 2006*

Decision Identification in Analysis and Design Models

Decision type: Transactionality (Tx), *Decision scope:* Task/Service Operation (from architectural style)
Forces: Resource protection needs, data currency, performance, legacy interfaces



Decision 1 (from analysis model):
Tx for GoNoGoDecision
Forces:
Read only access via idempotent service operations, frequent execution

Decision 3 (from analysis model):
Tx for ContactCustomerTask
Forces:
Modifying customer profile, so same threats as for 2
Only non-Tx messaging interface is available

Decision 2 (from analysis model): Tx for CreatePolicyTask
Forces: Multiple process instances might run concurrently, policies in database have to be protected from phantom reads, lost updates, deadlocks

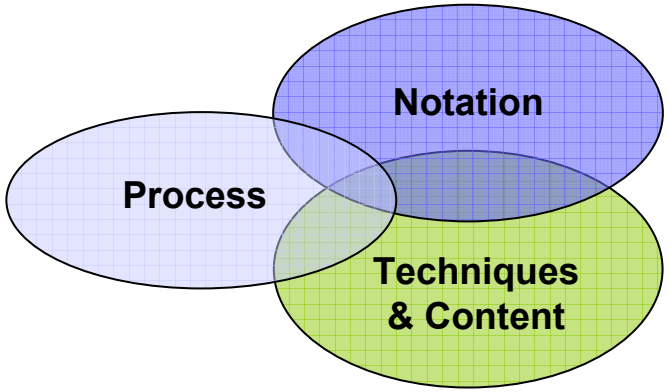
During the IT design (*phase*), the process modeler (*role*) defines the transaction boundaries for all invoke activities within a business process (*scope*), as well as the other architectural layers.

Agenda

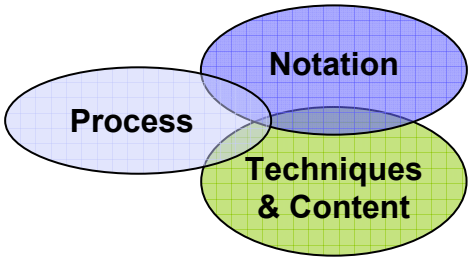
1. Motivating case studies and definitions
2. **Requirements for service design methods**
3. General-purpose and SOA-specific design methods
4. Architectural decision modeling concepts
5. SOA Decision Model content
6. Decision capturing wiki tool
7. Future work

General Requirements for Software Engineering Methods

- *Method = Process + Notation + Supporting Techniques and Content*
  - Must cover all phases of software development lifecycle, be repeatable
- Be consumable and customizable (all dimensions)
  - Throw away irrelevant parts (which ones?)
  - Add missing elements (authoring effort?)
- Tie in with methods for other disciplines
  - Business modeling, project management
- Have meta model or formal underpinning
  - Tool support, architecture validation
- Be detailed enough to be actionable
  - Support agile team organization practices (iterative, incremental design)



Specific Requirements for SOA Design Methods



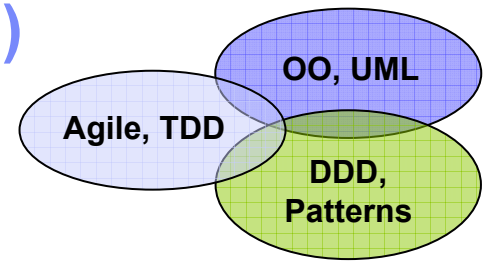
1. Support entire *service* development lifecycle (“P”)
   - Service analysis, design, implementation phases at a minimum
   - Often more than one project, multiple roles involved
2. Express functional contract (syntax), semantics, QoS policies (“N”)
   - Separate platform-independent from platform-specific design concerns
3. Provide techniques leveraging principles and patterns of SOA (“T”)
   - Service composition, Enterprise Service Bus (ESB), service registry scopes
4. Give advice r/ granularity and other SOA-specific “best practices” (“C”)
   - Must address service lifecycle management, e.g., ownership and versioning
   - Forces differs per service type, see N. Josuttis in “SOA in Practice”

http://www.soa-in-practice.com/

Agenda

1. Motivating case studies and definitions
2. Requirements for service design methods
3. **General-purpose and SOA-specific design methods**
4. Architectural decision modeling concepts
5. SOA Decision Model content
6. Decision capturing wiki tool
7. Future work

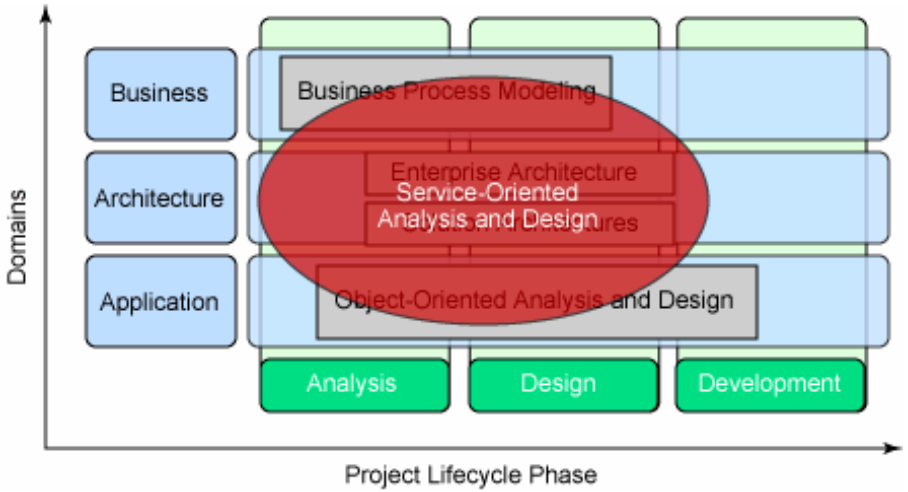
Object-Oriented Analysis and Design (OOAD)



- OOAD is a valid option for designing individual services:
  - Two prominent examples of *Process ("P") and Notation ("N"):*
    - G. Booch "Object-Oriented Analysis and Design With Applications" (704 pages)
    - Rational Unified Process (RUP) and Unified Modeling Language (UML)
  - *Techniques ("T"):* CRC cards, context maps, Test-Driven Development (TDD)
  - *Content ("C"):* E. Evans "Domain-Driven Design" (DDD) (529 pages), many more patterns

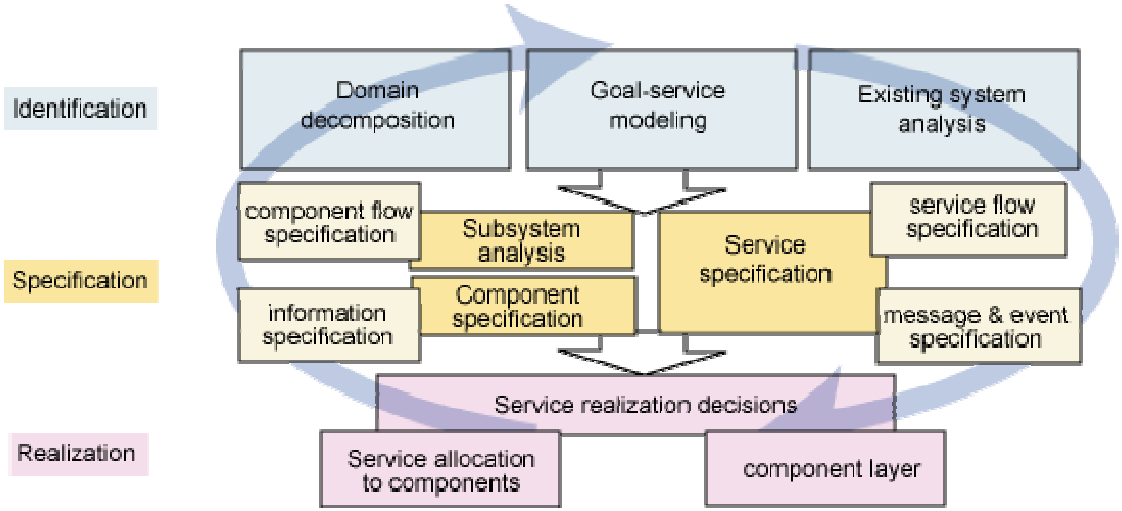
M. Fowler, "OOPSLA School of Thought" http://www.martinfowler.com/bliki/AltNetConf.html

  - SOA customization: no remote object references, no single owner of service model
- But: OOAD still program-centric
  - No good support for (enterprise) architectural concerns and SOA-specific principles and forces
  - Assumes green field development
- Enter *Service-Oriented Analysis and Design (SOAD)*



O. Zimmermann, P. Krogdahl, C. Gee, IBM developerWorks 2004, http://www.ibm.com/developerworks/library/ws-soad1/

A. Arsanjani "Service Modeling and Architecture (SOMA)"

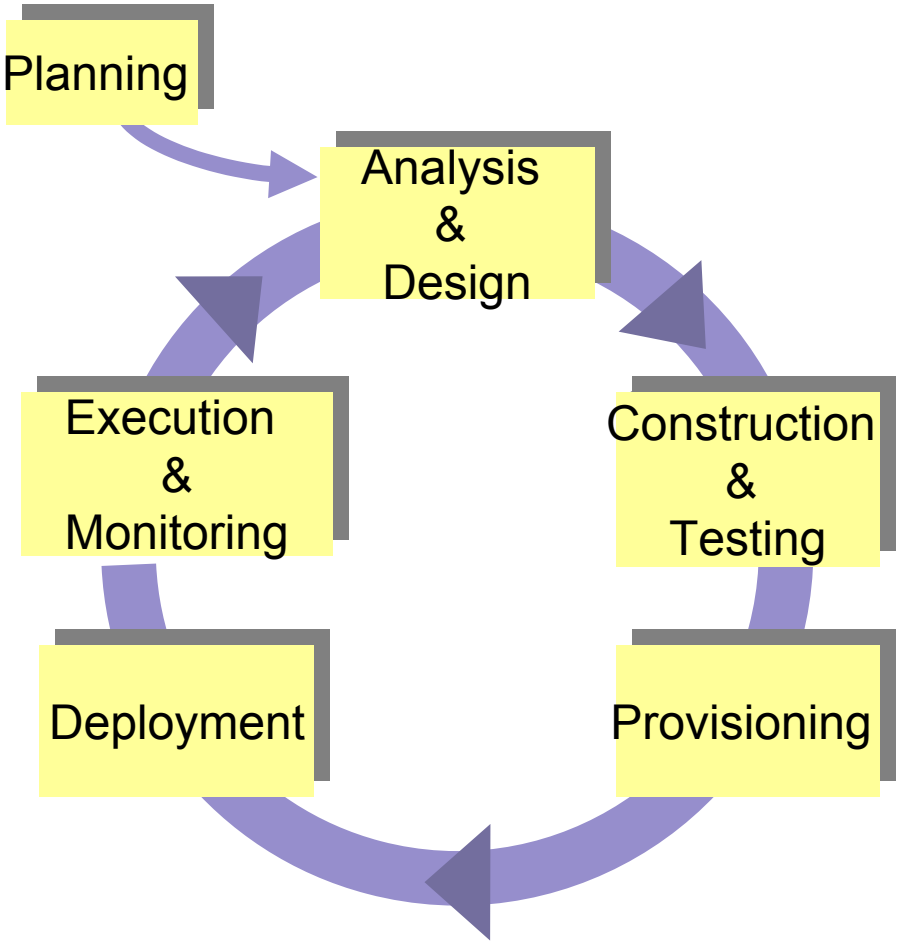


- SOAD offering from IBM, two ways to get access to full documentation:
  - Via consulting engagements
  - As a RUP extension

http://www.ibm.com/developerworks/webservices/library/ws-soa-design1/

- Introduces service identification, specification, realization steps ("P")
  - Integrates supporting techniques such as goal-service modeling ("T")
  - Service litmus test to assess whether candidate service meets principles ("T")
  - New work product "service model" to capture service specification ("N")

M. Papazouglou “Service Development Lifecycle (SDLC)”



- Full lifecycle process (“P”)
  - Book chapter (60 pages)
- Milestones and principles provide content (“C”)
  - Low coupling, high cohesion
  - Two granularity levels
  - Web services design
  - Dealing with legacy

Michael P. Papazoglou Web Services: Principles & Technology Prentice-Hall, October 2007 ©

Other Service Modeling Approaches

- CBDI-SAE series of articles (6-10 pages each)
  - Process and informal guidance (“P”, some “C”)
  - Only partially published, full content available to platinum subscribers only

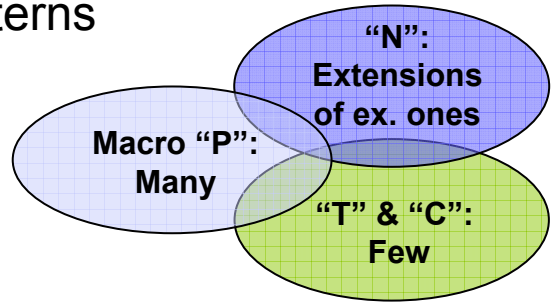
http://www.cbdiforum.com/

- A. Erradi et al “SOAF: An Architectural Framework for Service Definition and Realization” (8 pages in proceedings of SCC 2006, IEEE)
  - Main focus are process steps (“P”)
  - Some advice regarding granularity (high level “C”)
- SOA series by T. Erl (four books)
  - Concepts in method very close to technology specifications (“P”)
  - Attempts to be self-containing, uses own terminology
- Several SOAD articles by authors from industry and academia
  - Most of them not very deep and mature (yet)

http://www.google.ch/search?hl=de&q=SOAD+SOA

Assessment of Existing Service Modeling Approaches

- *Existing methods do not cover service design lifecycle in sufficient detail*
  - Typically good support for structuring parts or all of the design process (“P”)
    - Many service identification techniques, e.g., business process modeling
    - However, *weak service realization/construction support*
    - Weak links with general-purpose design methods and other disciplines
  - UML profiles or annotated WSDL/XSD as service models (“N”)
    - Domain-specific languages can also be found in literature
  - *Advice is given informally (in text), therefore not always actionable* (“C”)
    - E.g., granularity, versioning, service lifecycles in SDLC
    - No strong connection with SOA principles and patterns
    - No support for architectural thinking (forces)
  - Only SOMA provides several techniques (“T”)
    - For early design stages

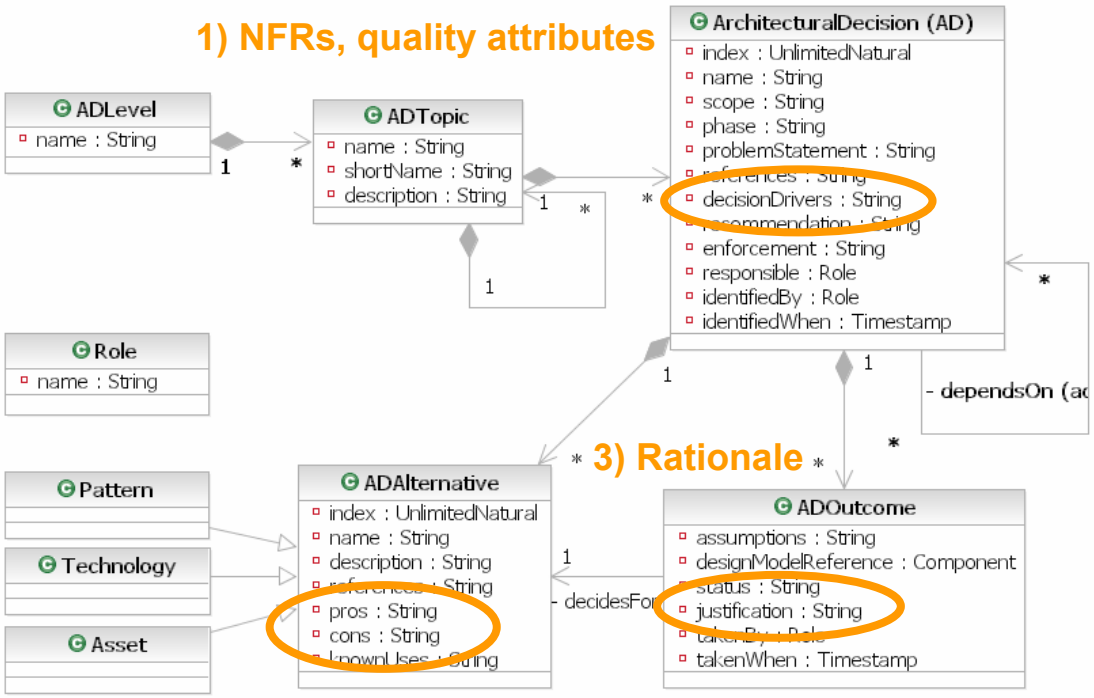


Agenda

1. Motivating case studies and definitions
2. Requirements for service design methods
3. General-purpose and SOA-specific design methods
4. **Architectural decision modeling concepts**
5. SOA Decision Model content
6. Decision capturing wiki tool
7. Future work

Architectural Decision Modeling (ADM) Overview

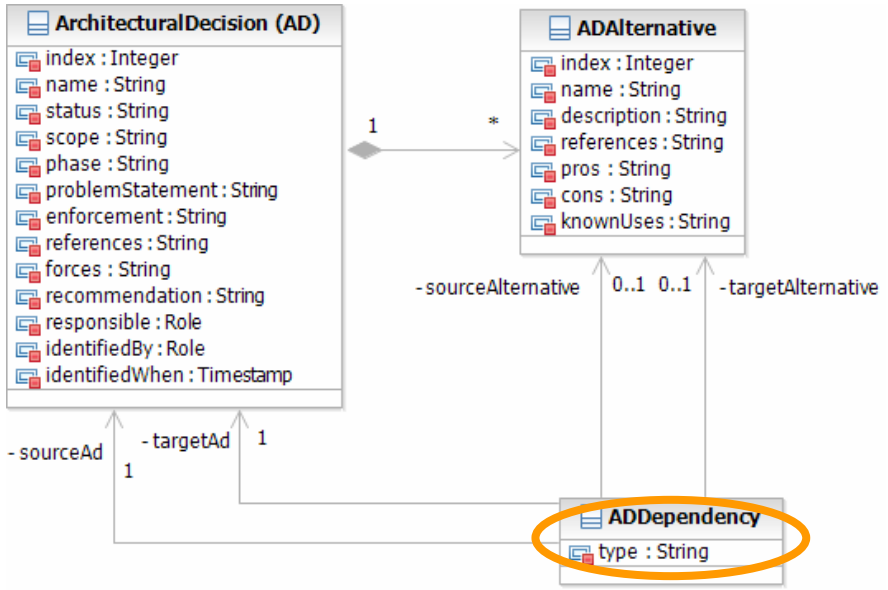
- Architectural decisions capture key design issues and the *rationale* behind a chosen solution:
  - Conscious design decisions concerning a software system as a whole, or one or more of its core components
  - With an impact on non-functional characteristics and quality factors



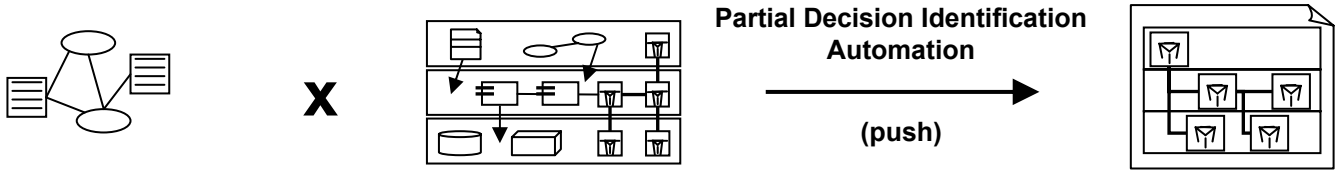
- From retrospective, template-based capturing to proactive decision *modeling*
  - Decision drivers include quality attributes (forces) and architectural principles (“C”)
  - Scope, phase, and role attributes provide method alignment (“P”)

Architectural Decision Modeling Concepts: Dependencies

- Dependencies can exist between decisions and alternatives
- Types of dependencies:
  - Influences, forces, constrains, triggers, forbids [Kruchten]



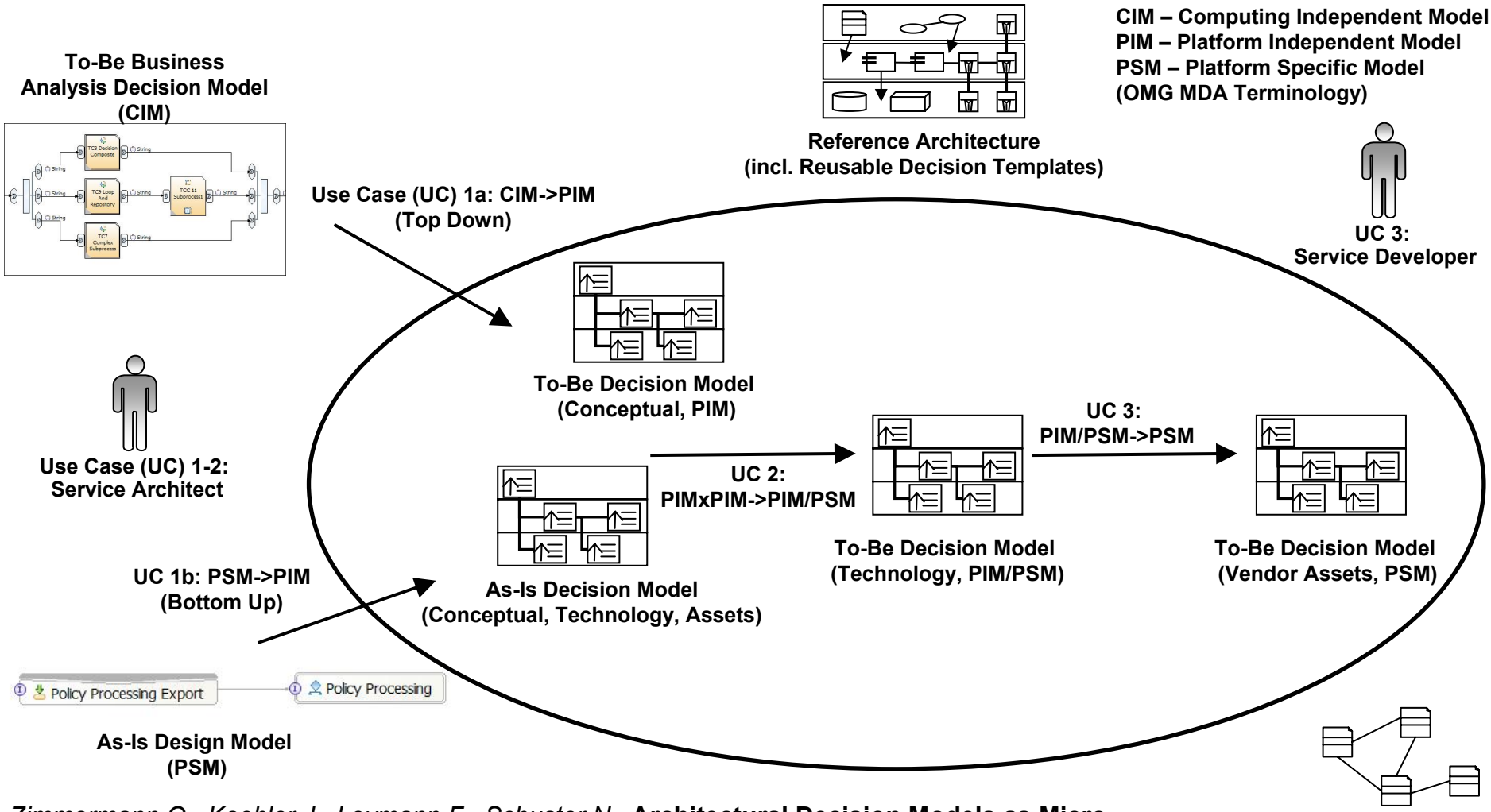
- Basis for usage of decision model as method plugin („P“):



Requirements Model (Machine and Human Readable) | **Reference Architecture (incl. Reusable Decision Templates)** | **Conceptual, Technology, and Vendor Asset Decision Model (To-Do List for Project Team)**

Zimmermann O., Gschwind T., Küster J., Leymann F., Schuster N., ***Reusable Architectural Decision Models for Enterprise Application Development****. In: Third International Conference on the Quality of Software-Architectures (QOSA 2007).*

Steps and Model Transformations in ADM and SOAD

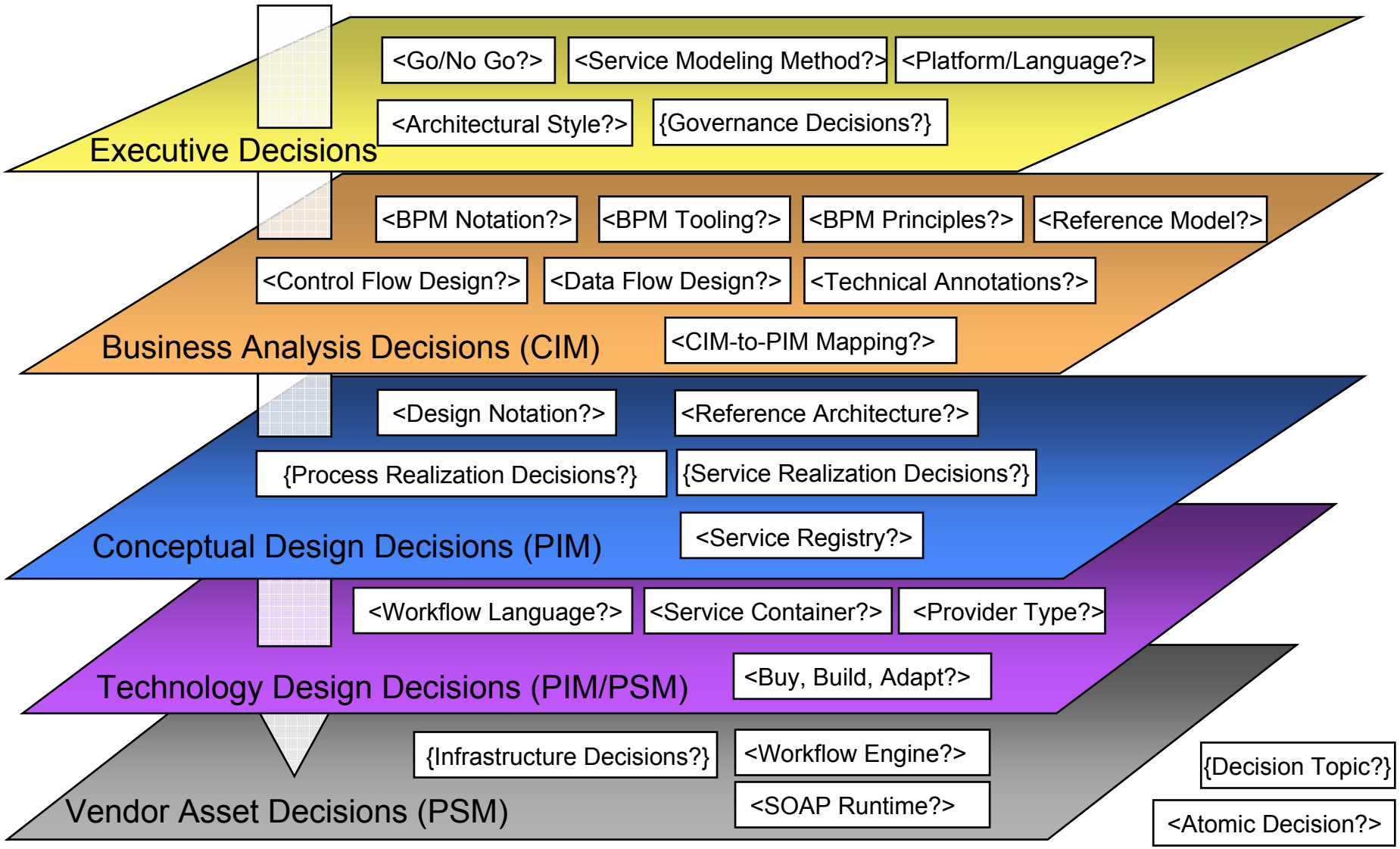


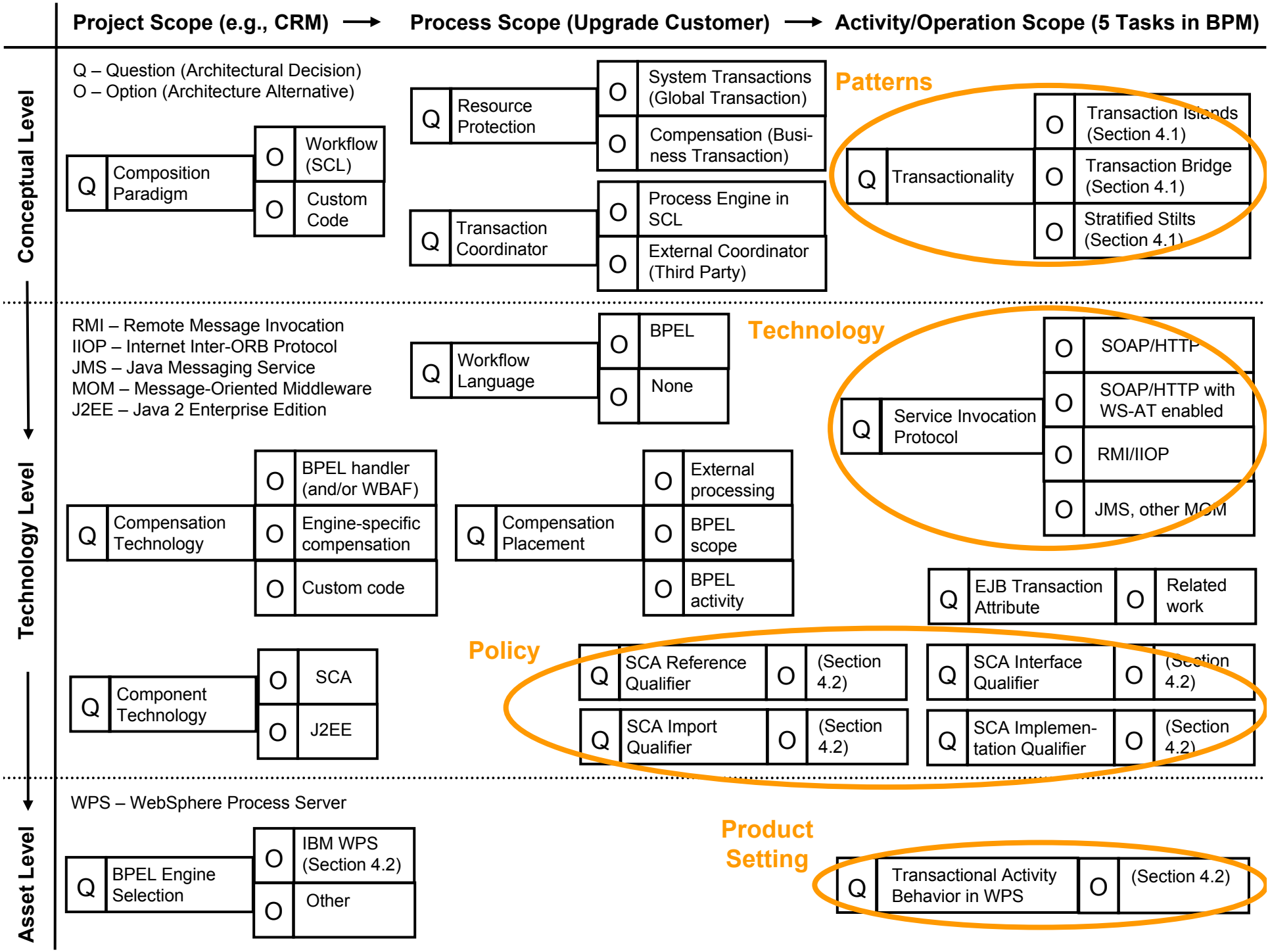
Zimmermann O., Koehler J., Leymann F., Schuster N., **Architectural Decision Models as Micro-Methodology for Service-Oriented Analysis and Design**. In: Lübke, D. (ed.), Proc. of the Workshop on Software Engineering Methods for Service-oriented Architecture 2007 (SEMSOA 2007), Hannover

Agenda

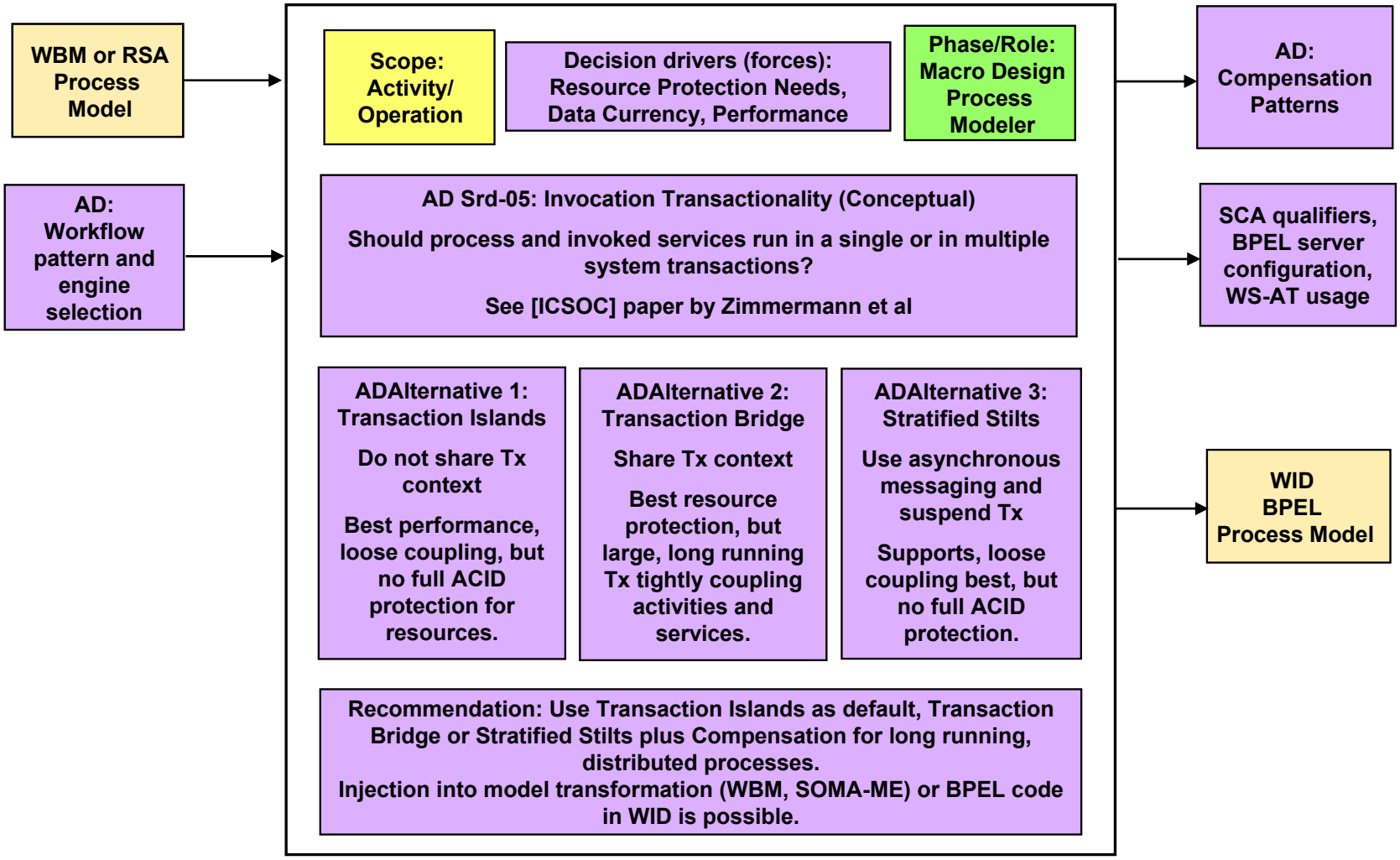
1. Motivating case studies and definitions
2. Requirements for service design methods
3. General-purpose and SOA-specific design methods
4. Architectural decision modeling concepts
5. **SOA Decision Model content**
6. Decision capturing wiki tool
7. Future work

ADMs as Service Realization Method for SOAD




Project Scope (e.g., CRM)
Process Scope (Upgrade Customer)
Activity/Operation Scope (5 Tasks in BPM)
Conceptual Level
Technology Level
Asset Level
Q – Question (Architectural Decision)
O – Option (Architecture Alternative)
Q Composition Paradigm
O Workflow (SCL)
O Custom Code
Q Resource Protection
O System Transactions (Global Transaction)
O Compensation (Business Transaction)
Q Transaction Coordinator
O Process Engine in SCL
O External Coordinator (Third Party)
Patterns
Q Transactionality
O Transaction Islands (Section 4.1)
O Transaction Bridge (Section 4.1)
O Stratified Stilts (Section 4.1)
RMI – Remote Message Invocation
IIOP – Internet Inter-ORB Protocol
JMS – Java Messaging Service
MOM – Message-Oriented Middleware
J2EE – Java 2 Enterprise Edition
Q Workflow Language
O BPEL
O None
Technology
Q Service Invocation Protocol
O SOAP/HTTP
O SOAP/HTTP with WS-AT enabled
O RMI/IIOP
O JMS, other MOM
Q Compensation Technology
O BPEL handler (and/or WBAF)
O Engine-specific compensation
O Custom code
Q Compensation Placement
O External processing
O BPEL scope
O BPEL activity
Q EJB Transaction Attribute
O Related work
Policy
Q SCA Reference Qualifier
O (Section 4.2)
Q SCA Interface Qualifier
O (Section 4.2)
Q SCA Import Qualifier
O (Section 4.2)
Q SCA Implementation Qualifier
O (Section 4.2)
Q Component Technology
O SCA
O J2EE
WPS – WebSphere Process Server
Q BPEL Engine Selection
O IBM WPS (Section 4.2)
O Other
Product Setting
Q Transactional Activity Behavior in WPS
O (Section 4.2)

QOCr Diagram for Conceptual Transactionality Decision



Decision Identification → **Decision Making** → **Decision Enforcement**

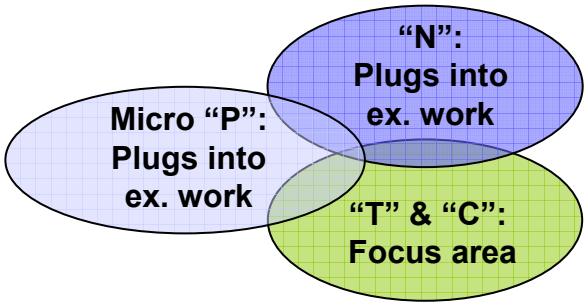
More SOA Decisions

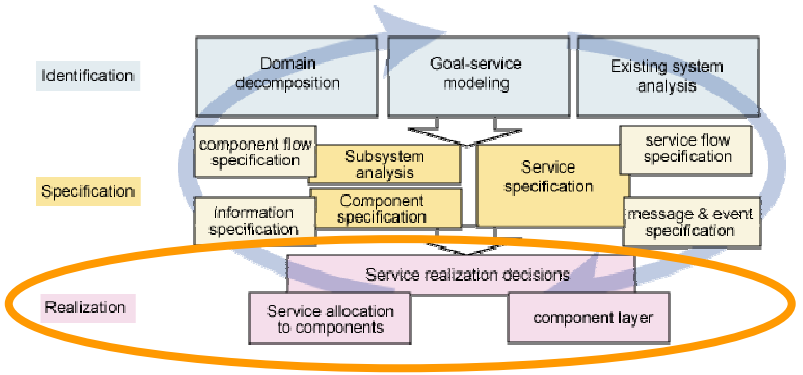
- Selection of a *reference architecture* and *layering schema*
  - IBM SOA Solution Stack, SOA/ODOE/WBI Reference Architecture, other
- Governance decisions
  - Service ownership and lifecycle management
  - *Registry type*, *publishing*, and *lookup* policies (who, when, how)
- Information management directions
  - Master data management yes/no?
  - Virtualization yes/no?
- Service parameter origin and granularity?
  - Enterprise data model reference/persistent or local copy/transient?
- Decisions and patterns and for transactional workflows in SOA

Zimmermann O., Grundler J., Tai S., Leymann F., **Architectural Decisions and Patterns for Transactional Workflows in SOA.** In: Krämer, B., Lin K.-J., Narasimhan, P. (eds.): ICSOC 2007, LNCS 4749, Springer, Heidelberg (2007)

SOA Decision Modeling…

- … complements existing design methods
  - Integration into process via phase and role attributes (“P”)
  - Decision model as an additional viewpoint on software architecture, scope attribute (“N”)
  - Decision driver attribute and QOCr diagrams as a decision making support technique (”T”)
- … and completes them with service realization/construction advice (“C”)
  - Several hundred SOA related decisions captured in actionable form (“C”)
    - Best practices exchange via recommendations attribute
  - Based on domain meta model





SOA Decision Modeling

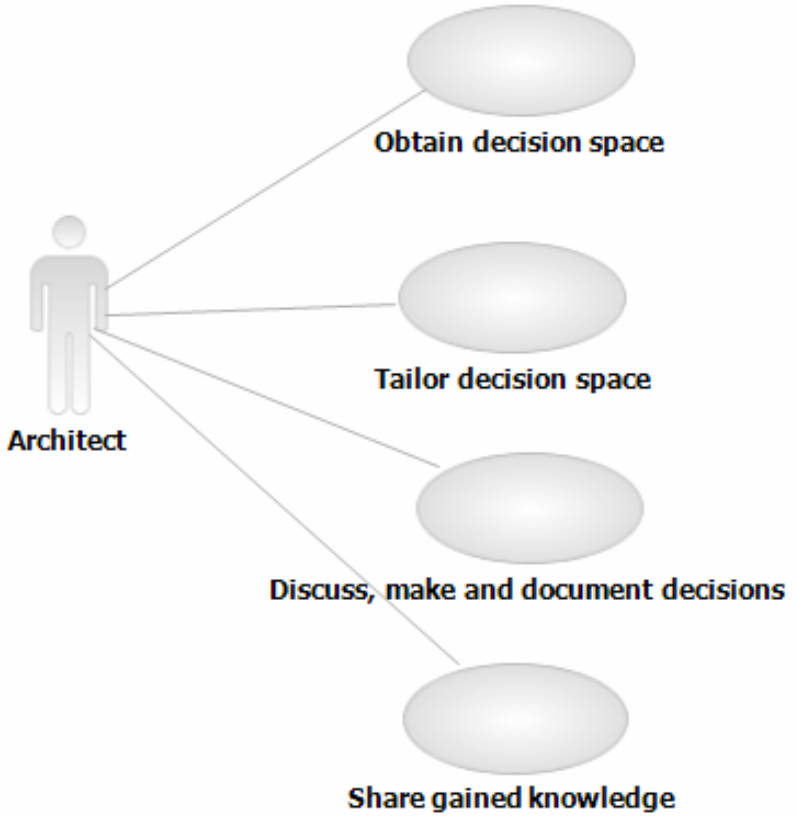
Agenda

1. Motivating case studies and definitions
2. Requirements for service design methods
3. General-purpose and SOA-specific design methods
4. Architectural decision modeling concepts
5. SOA Decision Model content
6. **Decision capturing wiki tool**
7. Future work

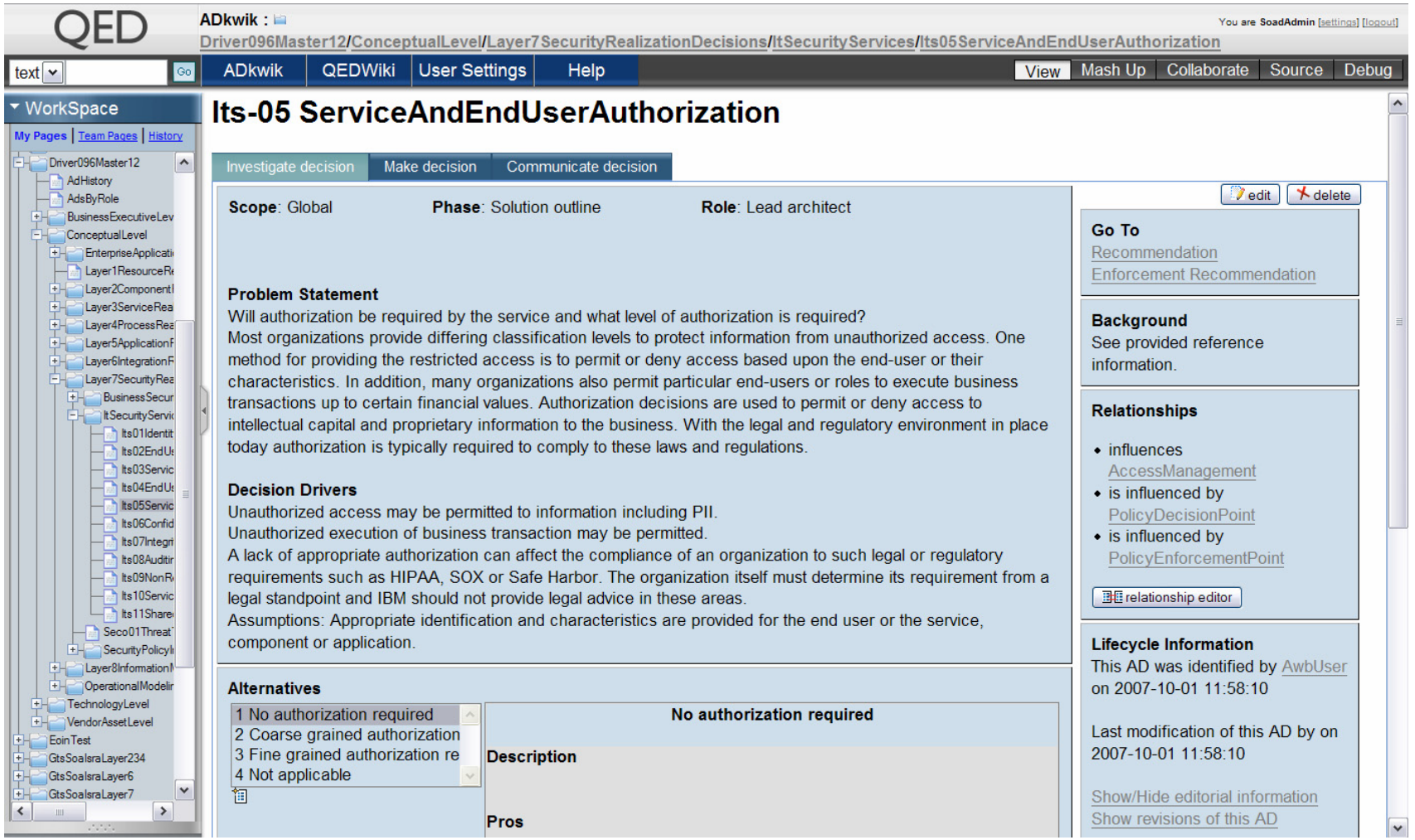
AD_{kwik} – Web 2.0 Collaboration Platform [SEKE]

- Regulatory compliance
  - E.g., maturity models
- Collaboration
  - In geographically distributed teams
- Reuse
  - Of already gained knowledge

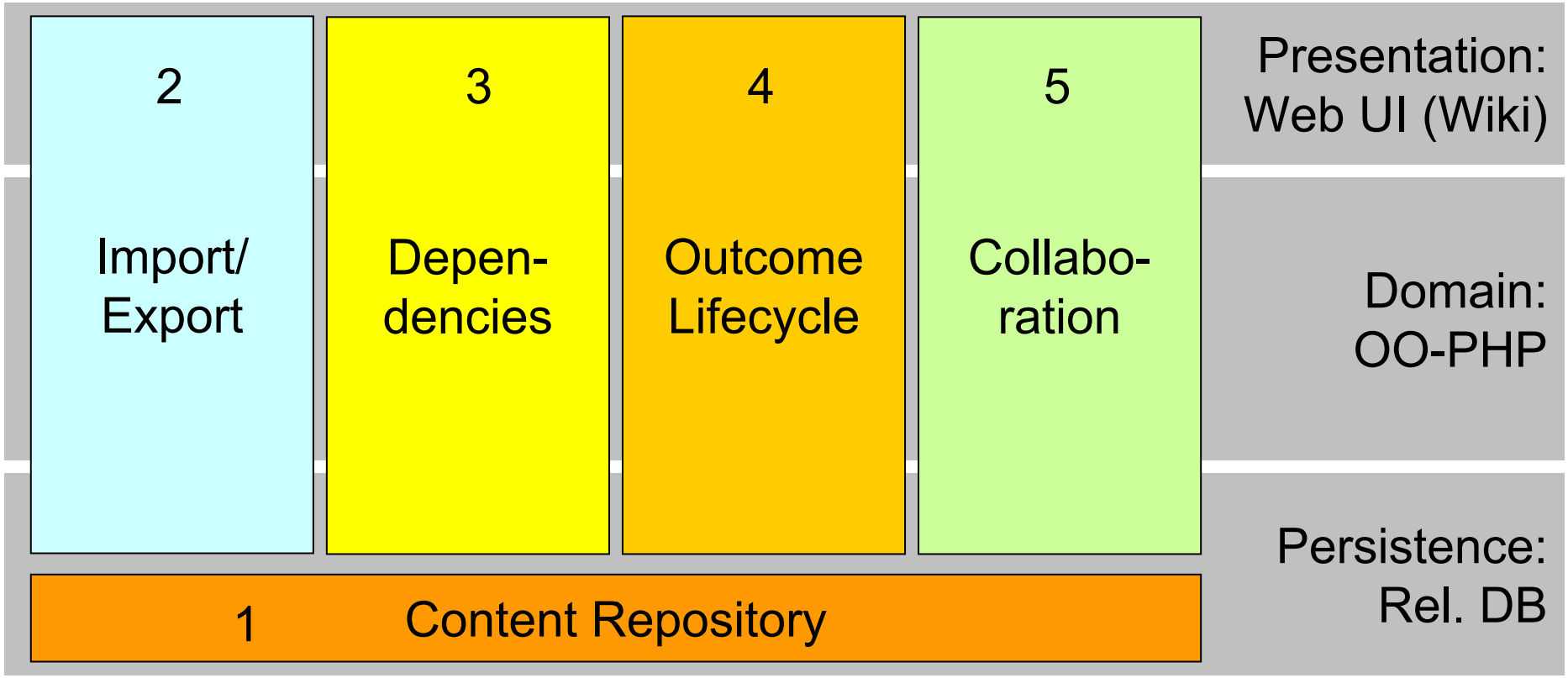
- Other required features:
  - Import and export
  - Searching and filtering
  - Dependency management
  - Report generation
  - Etc.



Decision Model Element in AD_{kwik} (and Database)



Architecture of Collaboration Platform AD_{kwik}



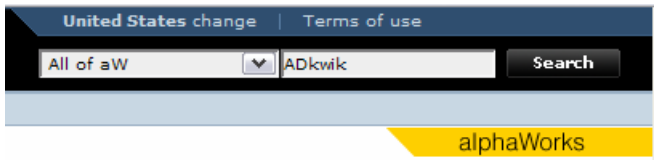
Schuster N., Zimmermann O., Pautasso C., **ADkwik: Web 2.0 Collaboration System for Architectural Decision Engineering**. In: Proc. of the Nineteenth International Conference on Software Engineering & Knowledge Engineering (SEKE 2007)

Agenda

1. Motivating case studies and definitions
2. Requirements for service design methods
3. General-purpose and SOA-specific design methods
4. Architectural decision modeling concepts
5. SOA Decision Model content
6. Decision capturing wiki tool
7. **Future work**

Future Work

- Integrate SOAD concepts into existing methods and tools more tightly
  - Non-functional requirements modeled as decision drivers
  - Design and decision model update protocol (mutual)
  - Enforcement of decisions via policies and code
- Improve usability of complex decision spaces
  - Decision propagation: pre-decide decisions, prune decision space
  - Filtering of decision space: by role, phase, scope
  - Consistency and completeness checking
- AD_{kwik} goes alphaWorks
  - IBM website providing access to emerging technology
  - http://www.alphaworks.ibm.com



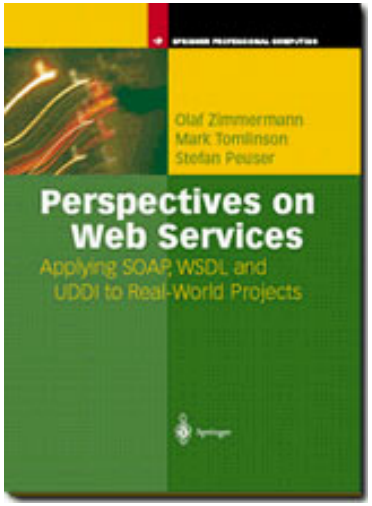
The Need for Service Modeling and Engineering Research

- Focus @ ECOWS 2007 and other conferences:
  - Dynamic matchmaking, automated service composition, semantics notations
  - Only 2/24 ECOWS 2007 papers have a pure design time focus!
- Service modeling (SOAD) is where OOAD was in ~1995
  - Several methods emerging, none of them is complete (page count!)
  - Many overlapping techniques – no “silver bullet”, no consensus, not actionable
  - None of them addresses detailed design concerns based on quality attributes
- Who advances state of the art in…
  - … IDEs for programming without call stack: resolving architectural forces, refactoring to services, build time matchmaking?
  - … quality metrics for interface granularity and other design issues?
  - … resolving tensions between different forms of pre- and postconditions in contract design (human service designer vs. logic-based)?

➔ Dagstuhl seminar proposed: “Software Service Engineering”

Thank You!

- Questions?
- Comments?
- Input to SOAD?



http://www.soadecisions.org

- "Perspectives on Web Services", Springer-Verlag
  - Captures project experience 2001-2003, incl. first 26 architectural decisions
  - Foreword by Grady Booch
  - Website also has case study reports and other papers
- Student projects @ Zurich Research Lab (ZRL) BIT

http://www.zurich.ibm.com/employment/2007/bit.html