12th International Software Product Line Conference 2008

Limerick, Ireland, 8 - 12 Sept 2008

Call for Workshop Participation

Service-Oriented Architectures and Software Product Lines - Putting Both Together (SOAPL 2008)
Monday, 8 September 2008

Description
Audience
Schedule
Submission Instructions
Workshop Organizers
SOAPL 2007

SPLC 2008
SPLC 2008 Workshops

Contact Information
Robert Krut
rk@sei.cmu.edu
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213
Phone: +1-412-268-8505
Fax: +1-412-268-5758

Presentations

Krut, Robert & Cohen, Sholom. Workhop on Service-Oriented Architectures and Software Product Lines - Putting Both Together (SOAPL 2008)

Dolog, Peter & Schafer, Michael. Feature Based Design of Web Service Transaction Compensations

Bartholdt, Jörg; Franke, Bernd; Schwanninger, Christa; & Stal, Michael. Combining Product Line Engineering and Service Oriented Architecture in Health Care Infrastructure Systems: Experience Report

Rusk, J. Jeffrey & Gasevic, Dragan. Semantic Web Services-based Reasoning in the Design of Software Product Lines

Gunther, Sebastian & Berger, Thorsten. Service-Oriented Product Lines: A Development Process and Feature Management Model for Web Services

Acher, Mathieu; Collet, Philippe; Lahire, Philippe; & Montagnat, Johan. Imaging Services on the Grid as a Product Line: Requirements and Architecture

Boffoli, Nicola; Caivano, Danilo; Castelluccia, Daniela; Maria Maggi, Fabrizio; & Visaggio, Giuseppe. Business Process Lines for SOA Development through the Software Product Lines Paradigm

Attendees

- Javier Baro, UPM, javierbaro@gmail.com
- Jörg Bartholdt, Siemens AG, joerg.bartholdt@siemens.com
- Thorsten Berger, University of Leipzig, berger@informatik.uni-leipzig.de
- Nicola Boffoli, University of Bari, boffoli@di.uniba.it
- Sholom Cohen, SEI, sgc@sei.cmu.edu
- Hyunsik Choi, Postech, nllbut@postech.ac.kr
- Peter Dolog, Aalborg University, dolog@cs.aau.dk
- Marius Dragouinoiu, University of Limerick, marius.dragouinoiu@il.ie
- Sebastian Guenther, Universität Magdeburg, sebastian.guenther@iti.cs.uni-magdeburg.de
- Paul Jensen, Overwatch Textron, paul.jensen@overwatch.com

- Mahvish Khorum, BTH, mkm@bth.se
- Bob Krut, SEI/CMU, rk@sei.cmu.edu
- Philippe Lahire, University of Nice, philippe.lahire@unice.fr
- Jaejoon Lee, Lancaster University, j.lee@comp.lancs.ac.uk
- Kwangchun Lee, Information and Communication University, statkclee@icu.ac.kr
- Tomi Männistö, Helsinki University of Technology, tomi.mannisto@tkk.fi
- James McGinley, Vitares LTD., james.mcginley@vitares.com
- Liam O'Brien, NICTA, liam.obrien@nicta.com.au
- Maryam Razavian, Politecnico di Torino, maryam.razavian@polito.it
- Jeff Rusk, Athabasca University, jrusk@nirb.ca
- Magnus Wilson, Ericsson AB, magnus.wilson@ericsson.com

Description

Service-Oriented Architecture (SOA) and software product line (SPL) approaches to software development share a common goal. They both encourage an organization to reuse existing assets and capabilities rather than repeatedly redeveloping them for new systems. The intent is that organizations can capitalize on reuse to achieve desired benefits such as productivity gains, decreased development costs, improved time to market, higher reliability, and competitive advantage. Their distinct goals may be stated as:

- SOA: "enable assembly, orchestration and maintenance of enterprise solutions to quickly react to changing business requirements" [Wienands]
- SPL: systematically capture and exploit commonality among a set of related systems while managing variations for specific customers or market segments

This workshop will build on results of the SOAPL 2007 workshop: Service-Oriented Architectures and Product Lines - What is the Connection? and the workshop report [Cohen & Krut]. This year's workshop, SOAPL 2008, will explore experiences in integrating SOA and SPL, specifically:

1. How web services have been used to support product lines using a service-oriented architecture?
2. How product line practices have been used to support web services and service-oriented architectures?

Topics of interest for the workshop include, but are not limited to:

- Practice areas that span both SOA and product lines (e.g., domain analysis, legacy mining, operations/governance, etc.)
- Handling variability through services
- Cost models to justify investment in SOA for product lines
- Use of support technology such as: domain specific languages, tools, other
- Differences between service-oriented and more conventional product line development approaches
- Architectural approaches: static vs. dynamic

Audience

Participants in the SOAPL 2008 will include product line and service-oriented practitioners who have experience in integrating service-oriented architectures and software product lines approaches. These include practitioners in product line engineering, product line management, and architects/ developers of SOA-based systems.

Schedule

The workshop will be highly interactive and focus on making tangible progress towards answering the two questions relating to results in integrating SOA and product line practices. The morning session will feature invited speakers and selected presentations based on position papers. Participants will be assigned to groups that reflect specific topics. After the workshop, the leader of each working group will be asked to write a summary of the working group's discussion and (especially) its conclusions.

Submission Instructions

Prospective participants are required to submit a 3-6 page position paper or experience report pertaining to the workshop topics listed above or describing the software architecture or other artifacts of a SOA-based product line.

All submissions will be reviewed by members of the program committee for quality and relevance. Accepted papers will become part of the workshop proceedings. Three or four papers will be chosen to be presented during the workshop to foment discussion. Submit your paper in PDF form to soa-workshop@sei.cmu.edu or by July 1, 2008. Notifications of paper or experience report acceptance will be sent by July 15, 2008. The camera-ready version of accepted papers is due July 31, 2008.

Workshop Organizers

- Sholom Cohen, Software Engineering Institute, USA
- Dragan Gasevic, Athabasca University, Canada
- Andreas Helferich, Universität Stuttgart, Germany
- Robert Krut, Software Engineering Institute, USA
- Jaejoon Lee, Lancaster University, UK
- Grace Lewis, Software Engineering Institute, USA
- Tomi Männistö, Helsinki University of Technology, Finland
- Curt Pederson, American Family Insurance, USA
- Dennis Smith, Software Engineering Institute, USA
- Christoph Wienands, Siemens Corporate Research, USA

[Wienands] Wienands, Christoph. "Studying The Common Problems With Service-oriented Architecture and Software Product Lines." *Service Oriented Architecture (SOA) & Web Services Conference*, Atlanta, GA, October 16-18, 2006.

[Cohen & Krut] Cohen, Sholom & Krut, Robert. *Proceedings of the First Workshop on Service-Oriented Architectures and Software Product Lines* (CMU/SEI-2008-SR-006). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2008.

Workshop on Service-Oriented Architectures and Software Product Lines - Putting Both Together (SOAPL 2008)

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Bob Krut & Sholom Cohen
8 September 2008
SPLC 2008, Limerick, Ireland

Software Engineering Institute | Carnegie Mellon

Agenda

09:00-09:15	Introductions and Goals
09:15-10:15	Invited Speaker: Dr Peter Dolog, Aalborg University
10:15-10:30	Break
10:30-12:30	Presentations and Discussions (20 minute time limit plus questions and discussion on each presentation)
12:30-14:00	Lunch
14:00-14:30	Final Presentation and Discussion (if necessary)
14:30-15:30	General Discussion
15:30-15:45	Break
15:45-17:00	General Discussion Continued
17:00-17:30	Conclusion: Goals Addressed, Topics for San Francisco, Future Work

Workshop Organizers

Sholom Cohen, Software Engineering Institute, USA

Dragan Gasevic, Athabasca University, Canada

Andreas Helferich, Universität Stuttgart, Germany

Robert Krut, Software Engineering Institute, USA

Jaejoon Lee, Lancaster University, UK

Grace Lewis, Software Engineering Institute, USA

Tomi Männistö, Helsinki University of Technology, Finland

Curt Pederson, American Family Insurance, USA

Dennis Smith, Software Engineering Institute, USA

Christoph Wienands, Siemens Corporate Research, USA

The First Workshop on Service-Oriented Architectures and Product Lines (SOAPL 2007)

Part of the 2007 Software Product Line Conference (SPLC 2007), 10 September 2007, Kyoto, Japan.

Service Oriented Architectures and Product Lines - What is the Connection?

SOAPL 2007 explored the connections from two perspectives:

1. Can services support product lines using a service-oriented architecture?

2. How can use of product line practices support services and service-oriented architectures?

Proceedings of the First Workshop on Service-Oriented Architectures and Product Lines (CMU/SEI-2008-SR-006).

http://www.sei.cmu.edu/publications/documents/08.reports/08sr006.html

The Second Workshop on Service-Oriented Architectures and Product Lines (SOAPL 2008)

Service Oriented Architectures and Product Lines - Putting Both Together

SOAPL 2008 explores experiences in integrating SOA and SPL:

1. How web services have been used to support product lines using a service-oriented architecture?

2. How product line practices have been used to support web services and service-oriented architectures?

Participants in the workshop hopefully includes product line and service-oriented practitioners who have experience in integrating service-oriented architectures and software product lines approaches.

Five position papers were accepted.

Accepted Papers

Combining Product Line Engineering and Service Oriented Architecture in Health Care Infrastructure Systems: Experience Report
Jörg Bartholdt, Bernd Franke, Christa Schwanninger, and Michael Stal, Siemens AG

Semantic Web Services-based Reasoning in the Design of Software Product Lines
J. Jeffrey Rusk and Dragan Gasevic, Athabasca University

Service-Oriented Product Lines: A Development Process and Feature Management Model for Web Services
Sebastian Gunther, Otto-von-Guericke-Universitat Magdeburg, and Thorsten Berger, Universitat Leipzig

Imaging Services on the Grid as a Product Line: Requirements and Architecture
Mathieu Acher, Philippe Collet, Philippe Lahire, and Johan Montagnat, Universite de Nice

Business Process Lines for SOA Development through SPL Paradigm
Nicola Boffoli, Danilo Caivano, Daniela Castelluccia, Fabrizio Maria Maggi, and Giuseppe Visaggio, University of Bari - Via E.

Workshop Theme

Two major themes for this year's workshop:

1. variability and variability mechanisms
2. product composition

both within the context of SOA and product lines.

All of the papers touched on one or both of those themes.

These topics will provide a starting point for the workshop.

Other suggested topics:

- in advance
- as papers are presented and discussed

Invited Speaker

Dr. Peter Dolog

Associate Professor

Computer Science Department

Aalborg University

Leads the Intelligent Web and Information Systems (IWIS) group

includes adaptive hypertext and hypermedia, user modelling, personalization, web based systems, web services, software product lines and technology enhanced learning.

Presentation Title:

Feature Based Design of Web Service Transaction Compensations

Workshop Topics

How have web services been used to support product lines using a service-oriented architecture?

How have product line practices been used to support web services and service-oriented architectures?

Additional topics:

- variability and variability mechanisms
- product composition

Conclusion

Were the goals of this workshop addressed? Comments?

What would you recommend as the topic for San Francisco?

What future work in this area will you be doing?

Software Engineering Institute | Carnegie Mellon

Feature Based Design of Web Service Transaction Compensations

Peter Dolog with Michael Schäfer
dolog@cs.aau.dk
CS Department
Intelligent Web Information Systems
http://www.cs.aau.dk, http://iwis.cs.aau.dk

SOAPL 2008 @ SPLC 2008, September 2008, Limerick, Ireland

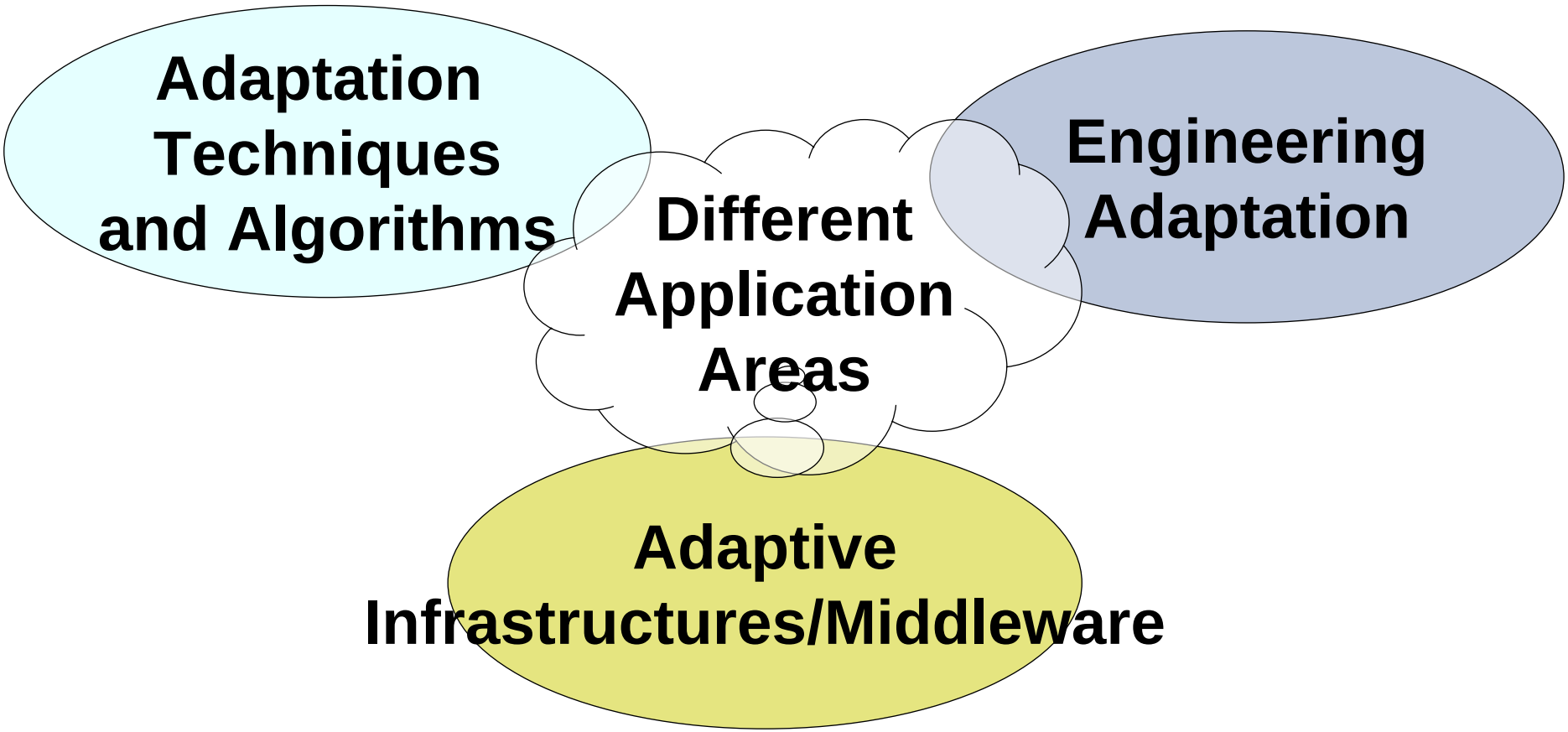
Outline

IWIS group and background
General problem
Business transactions
Middleware for advanced compensations
Service provider and client feature modelling
Matchmaking and restriction model
Further Challenges

Outline

IWIS group and background
General problem
Business transactions
Middleware for advanced compensations
Service provider and client feature modelling
Matchmaking and restriction model
Further Challenges

Intelligent Web and Information Systems
http://iwis.cs.aau.dk



Adaptation/Customization

- Customization by humans (designers)
- Dynamic adaptation by a system itself
- Adaptation is about decision on which information resource or function variant to provide or recommend access to,
- We need a knowledge to decide about appropriate information or service configuration in a certain processing step (user or other):
  - Resource and information access environment
  - Application domain
  - User/Context
  - And their configuration – variants and their meaningful combinations for certain purposes

Outline

IWIS group and background

General problem

Business transactions

Middleware for advanced compensations

Service provider and client feature modelling

Matchmaking and restriction model

Further Challenges

Open Web Service Environment

Service Providers

- A number of autonomous service providers exist
- They can provide similar functionality
- They can dis-/appear any time
- Each wants to maximize its profit for executing provided services by external consumers

Service Consumers

- Number of consumers with similar requirements exist
- They want to achieve high value for their expense
- To maximize their service
- By composing matched available services from different providers

Software Product Lines

Software Providers

- Number of reusable software assets exist
- They may vary in its functionality
- They want to maximize its profit by providing the assets in an application in a family mostly from one company

Software Consumers

- Number of consumers with similar requirements
- They want to achieve high value for their expense
- To maximize their service
- By composing a final application from the reusable assets

Difference

Client is composing in web service world

Client is composing from different providers in web service world

Services used in the composition may be exchanged

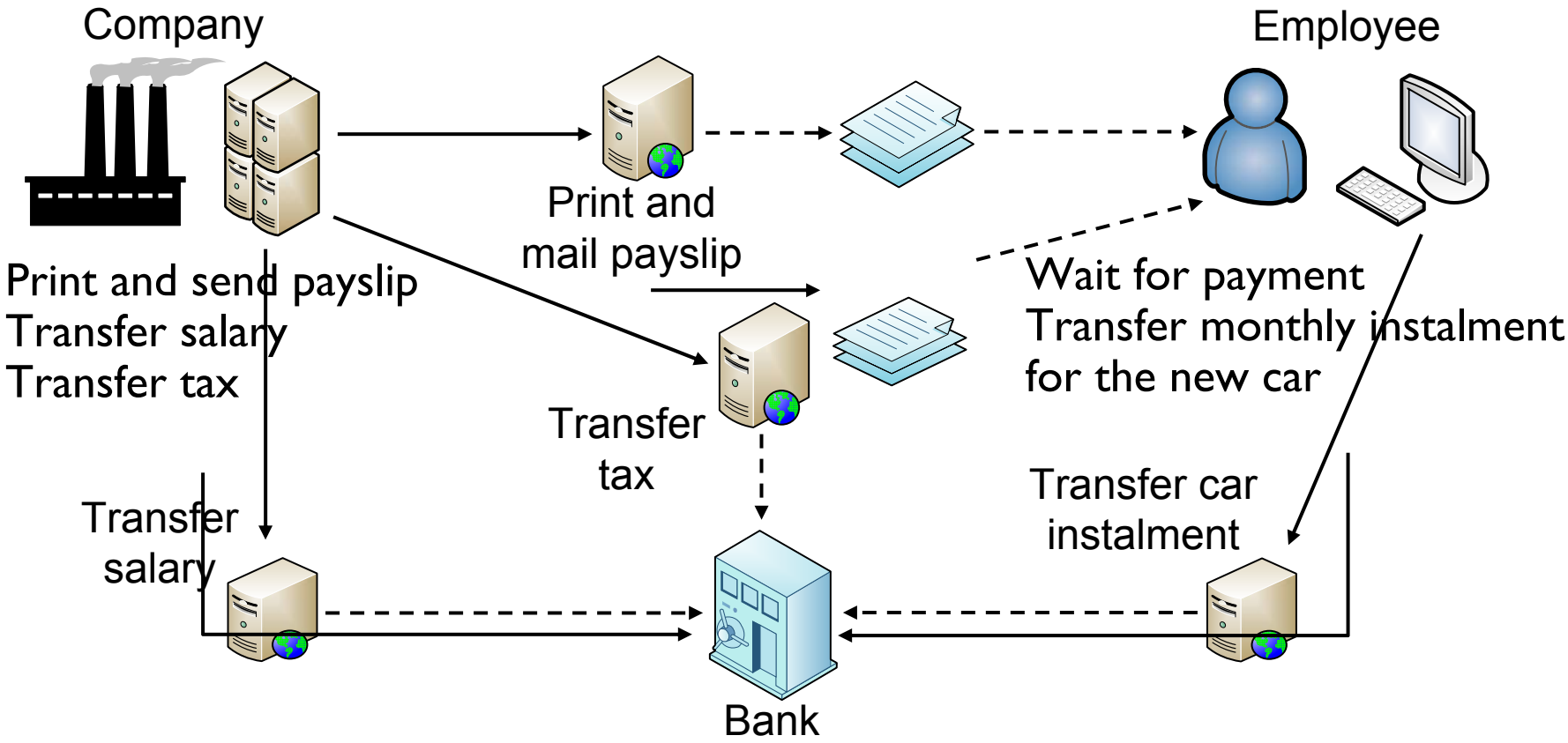
Question:

- What can be achieved by current state of the art software product lines techniques?

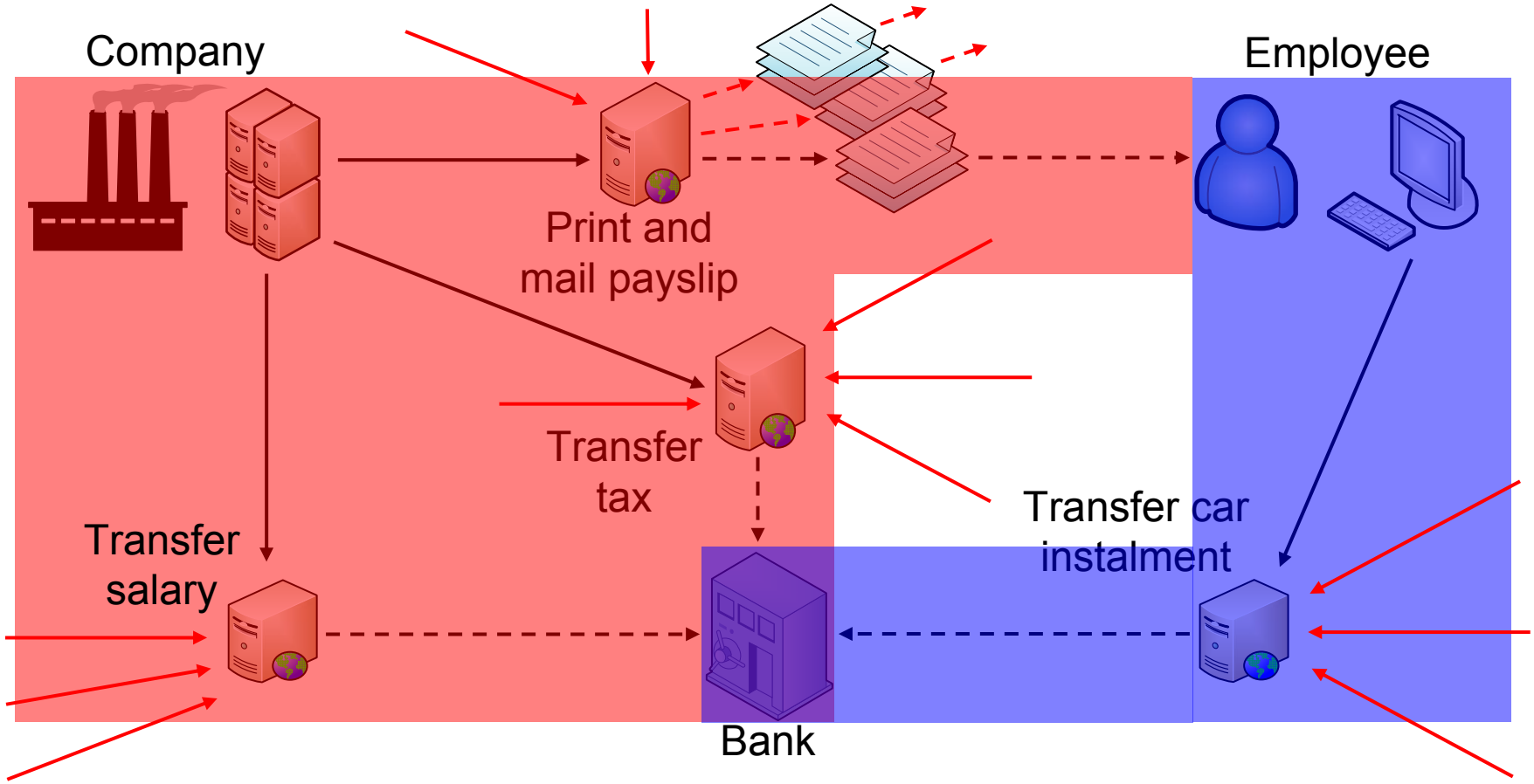
Outline

IWIS group and background
General problem
Business transactions
Middleware for advanced compensations
Service provider and client feature modelling
Matchmaking and restriction model
Further Challenges

Payroll Scenario



Service Oriented Payroll Scenario



To reach mutually-agreed outcome (commit/cancel)
In environment with concurrent access

Transactions

Control the execution of the required operations on the external services.
Consist of a set of operations (e.g. database operations) that are performed by multiple participants.
Control the collective outcome of the operations.

Distributed transactions control the execution of operations on multiple providers.

- Participant
- Coordinator

Error Compensation

Different transaction specifications exist for different purposes

Backward recovery

Normally, predefined *rollback operations* are executed in order to restore the state before the transaction.

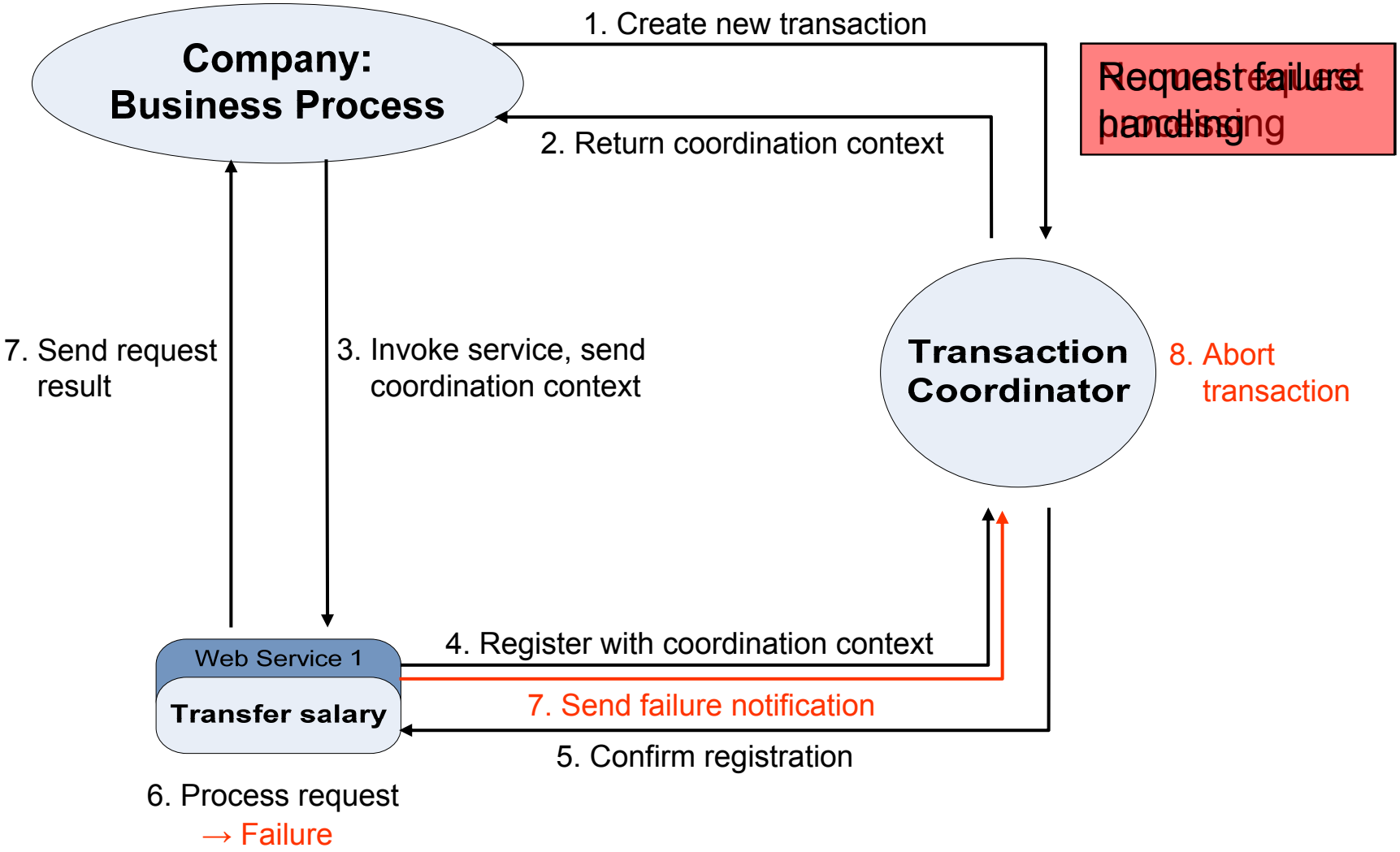
- Time and money is lost
- *Dependent transactions* also have to roll back (*domino effect*)

Forward recovery

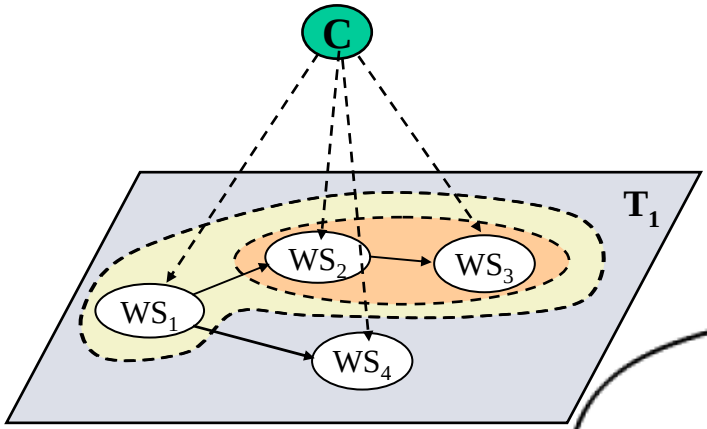
Aims at changing pro-actively the state of the participant or transaction to enable a successful execution after a failure.

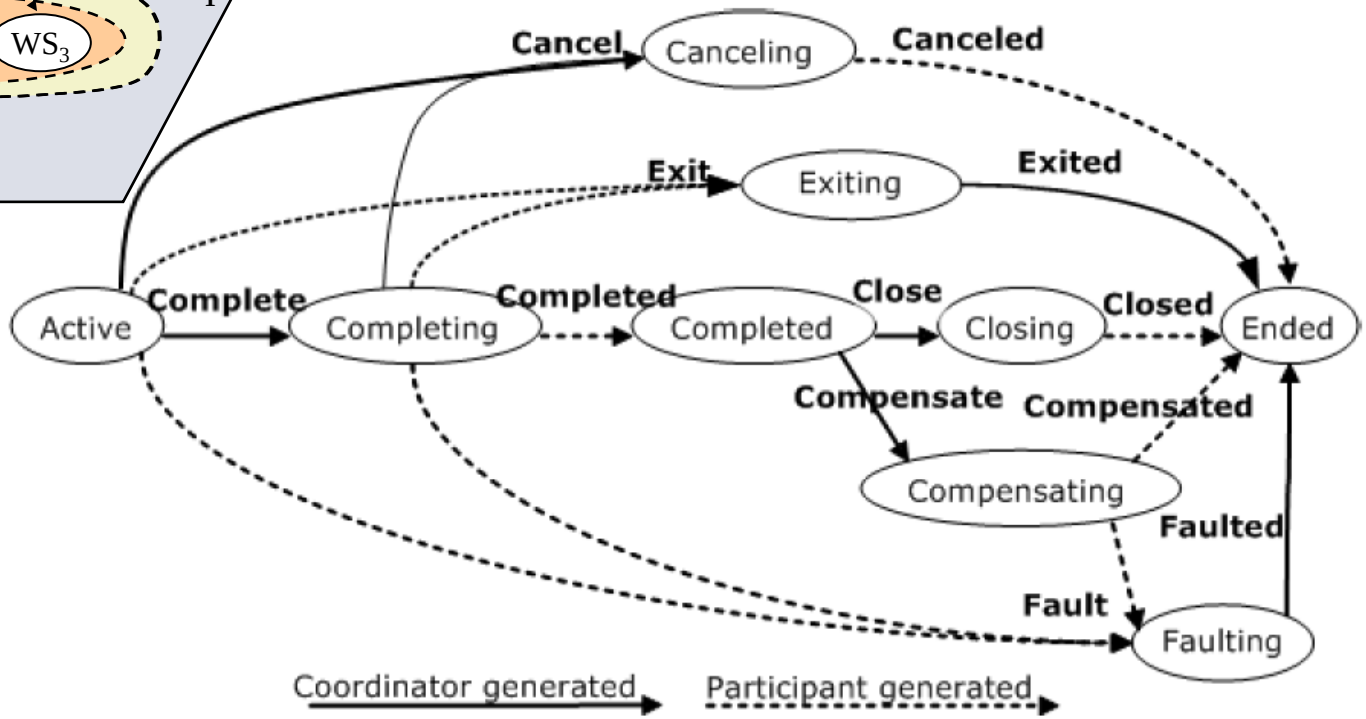
- Complex
- Can normally only be performed semi-automatically

Traditional WS-Transaction Coordin. Structure



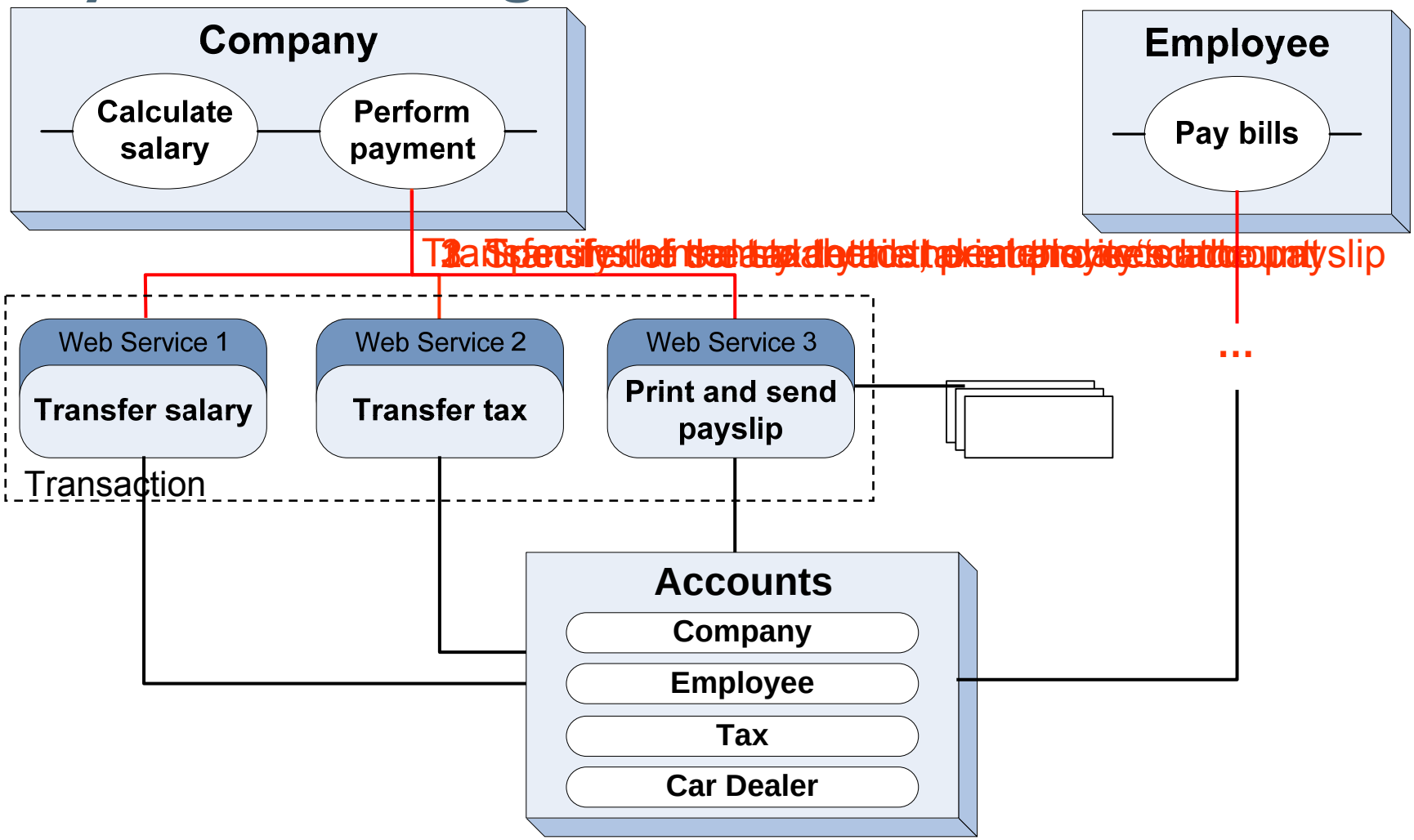
WS – Tx / Business Activity Coordination Type



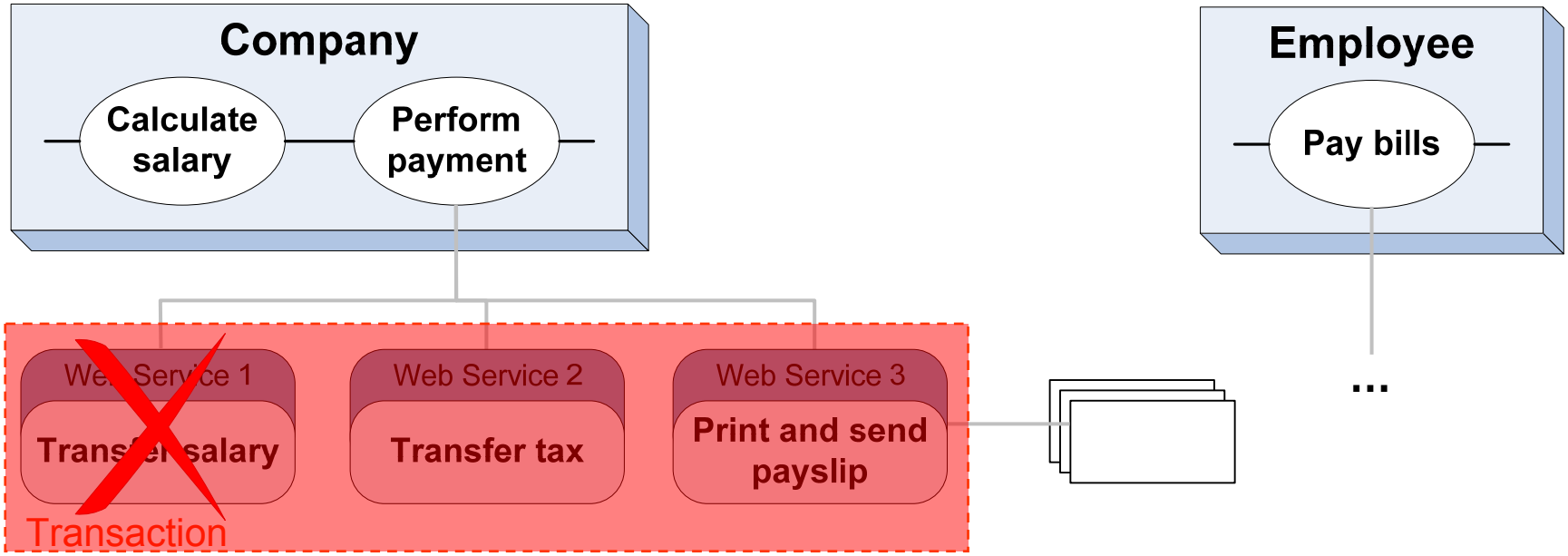


abstract state diagram

Payroll Processing



Motivating Scenario – Problem

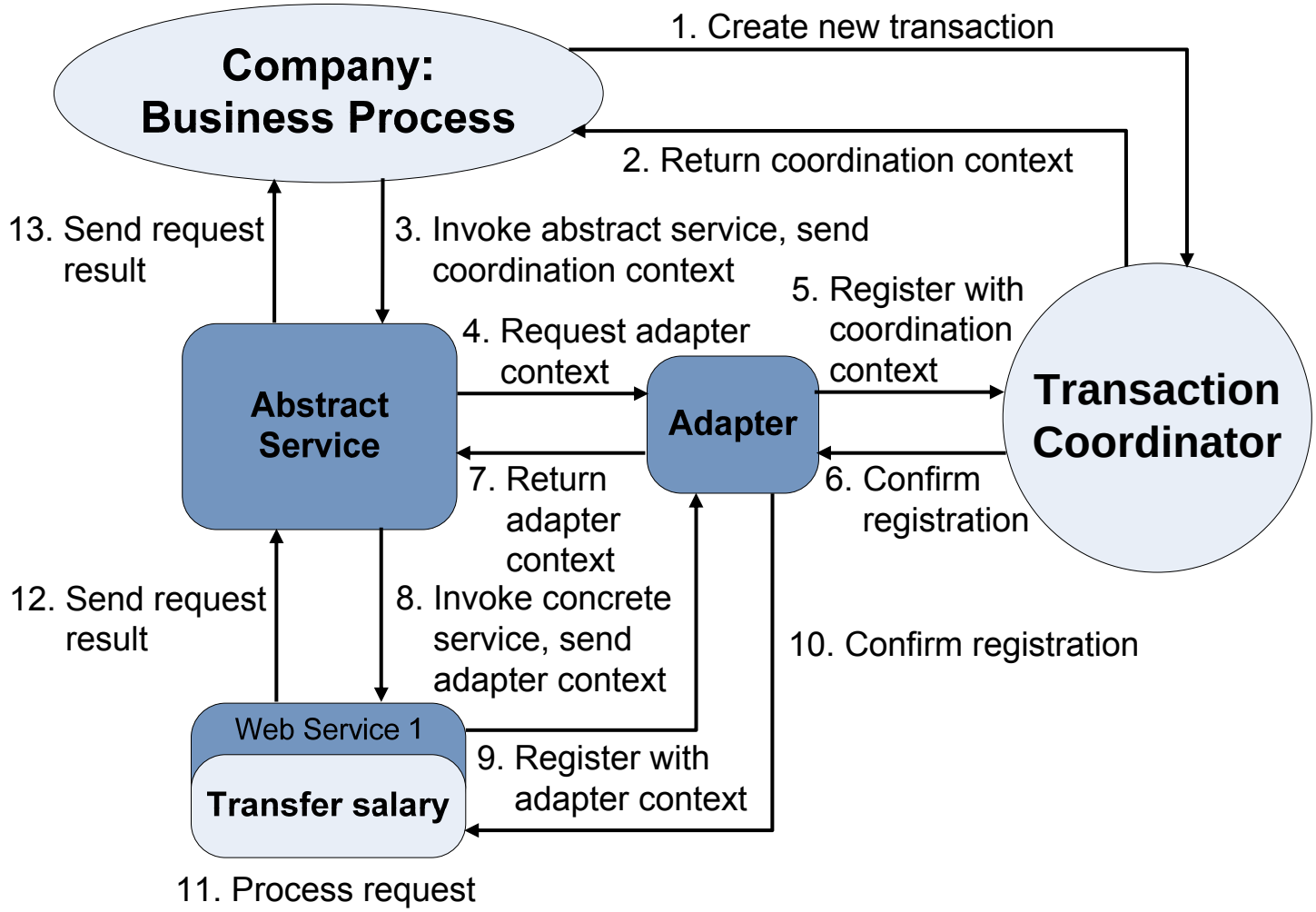


A service fails due to an internal error.
The error can only be compensated by aborting the complete transaction.
Why should the transaction be aborted, if a different service exists that can perform the same operations?

Outline

IWIS group and background
General problem
Business transactions
Middleware for advanced compensations
Service provider and client feature modelling
Matchmaking and restriction model
Further Challenges

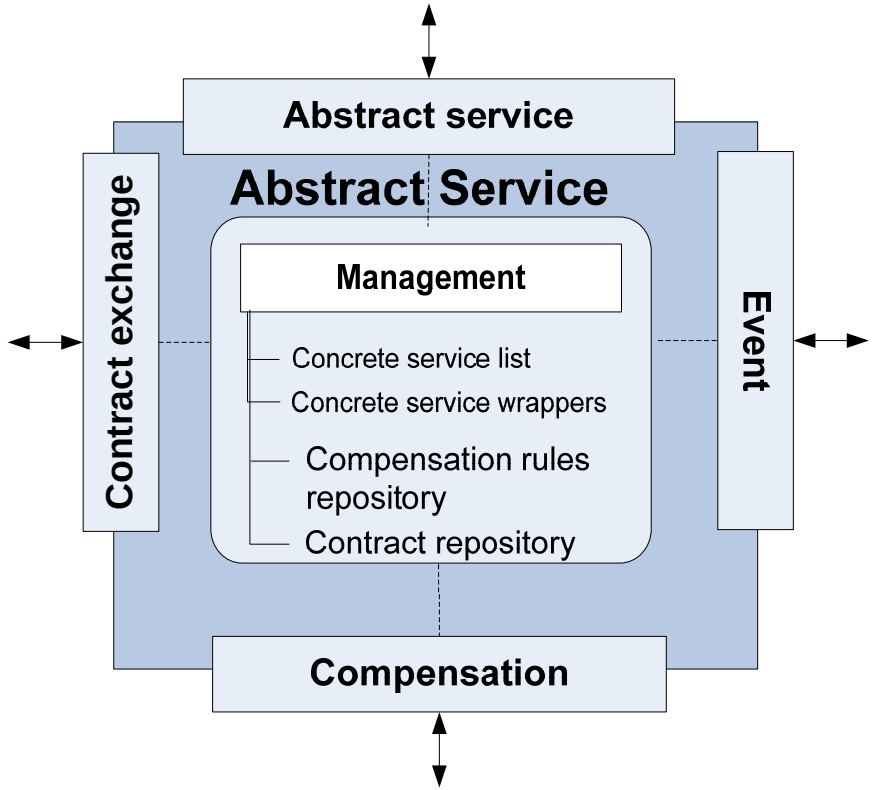
Extended Transaction Coordination Structure



New Components - Abstract Service

Does not directly implement functionalities.
Manages a list of *concrete services*.
Is a mediator between the client and the concrete service.
Manages and performs compensation actions.
Interfaces:

- Service
- Event (internal compensation handling)
- Compensation (external compensation handling)
- Contract exchange



Compensation Activities and Types

Nr.	Compensation Type		Compensation Activities: ServiceReplacement	LastRequestRepetition	PartialRequestRepetition	AllRequestRepetition	CompensationForwarding	AdditionalServiceInvocation	AdditionalRequestGeneration	ServiceAbortInitiation	RequestSequenceChange	ResultResending
01		NoCompensation										
02	Internal	Repetition		X								
03	Internal	Repetition			X							X
04	Internal	Replacement	X	X								
05	Internal	Replacement	X		X							X
06	Internal	Replacement	X			X						X
07	External	Forwarding	(X)	(X)	(X)	(X)	X	(X)	(X)	(X)	(X)	(X)
08	External	AdditionalService						X				
09	External	AdditionalRequest							X			
10	External	SessionRestart				X				X	X	X

X Included compensation activity (X) Possibly included compensation activity

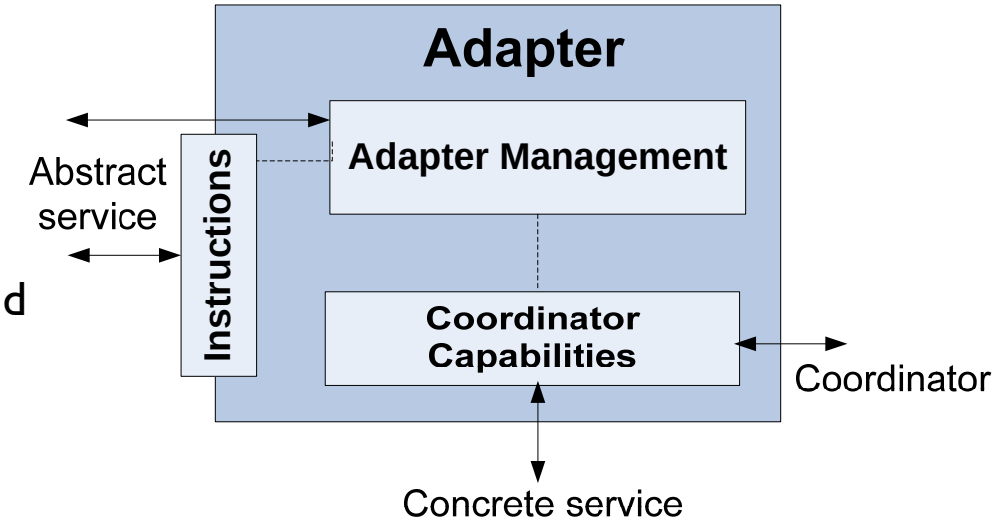
Example: Internal Compensation Rule

```
<cmp:InternalCompensationRule identifier="internalFailureLastRequestResending">
  <cmp:CompensationCondition>
    <cmp:ParticipantEvent eventCode=
   "http://sourceforge.net/projects/frogs/AdapterInteraction/ParticipantFault"/>
    <cmp:ParticipantState
      stateType='http://schemas.xmlsoap.org/ws/2004/10/wsba/Faulting' />
    <cmp:ReplacementService exists="true" isDirectReplacement="true" />
    <cmp:RequestSequence>
      <cmp:Request identifier="transferSalaryMethod" />
    </cmp:RequestSequence>
  </cmp:CompensationCondition>
  <cmp:CompensationPlan>
    <cmp:Compensation>
      <cmp:ServiceReplacement/>
    </cmp:Compensation>
    <cmp:Compensation>
      <cmp:RequestResending lastN="1" />
    </cmp:Compensation>
  </cmp:CompensationPlan>
</cmp:InternalCompensationRule>
```

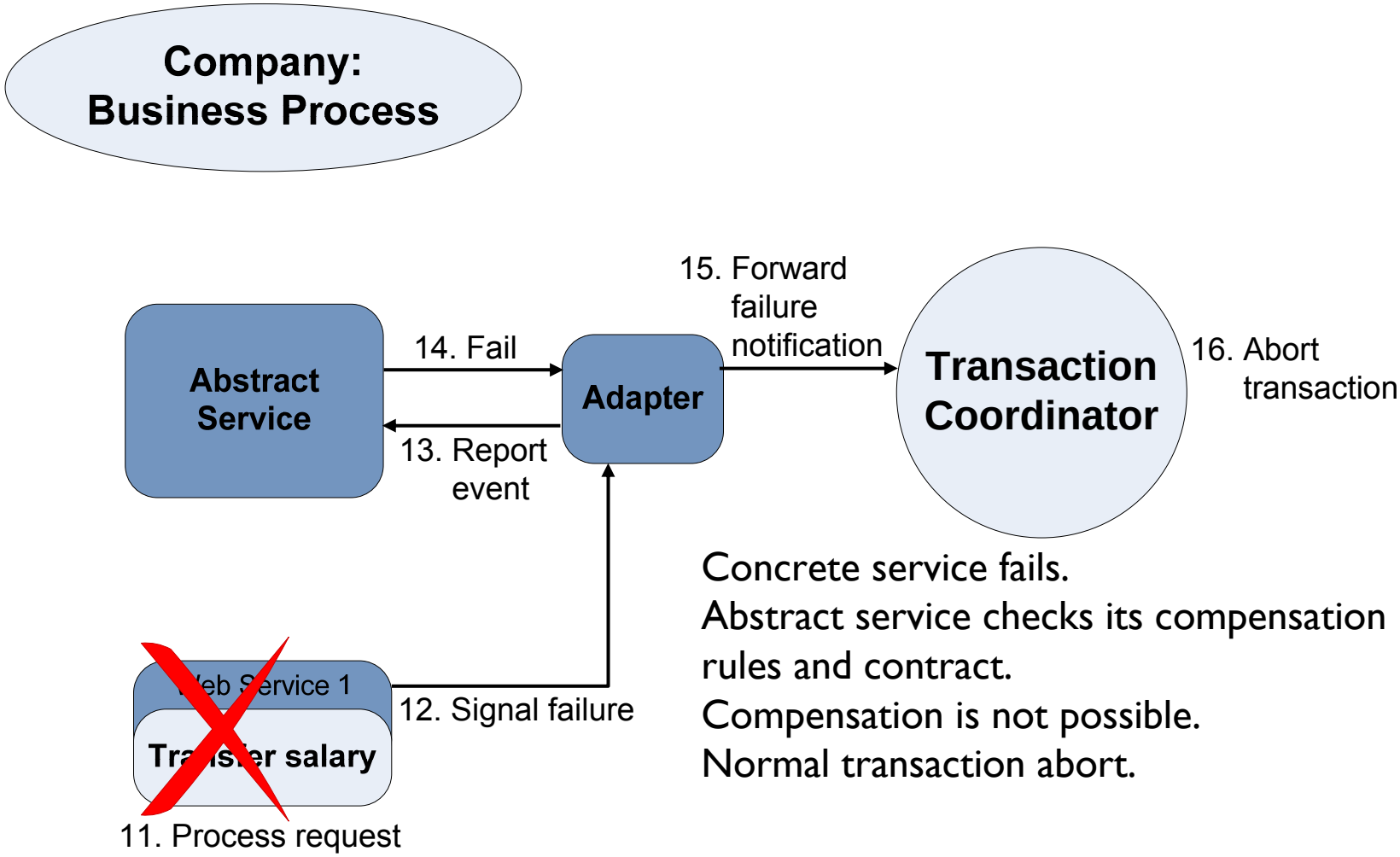
New Components - Adapter

Encapsulates coordinator-specific functionality.
Functions as a coordinator for the concrete service.
Manages messaging:

- Forwards normal messages between the real coordinator and the concrete service.
- Intercepts failure messages and informs the abstract service.
- Creates additional notifications as part of a compensation process.

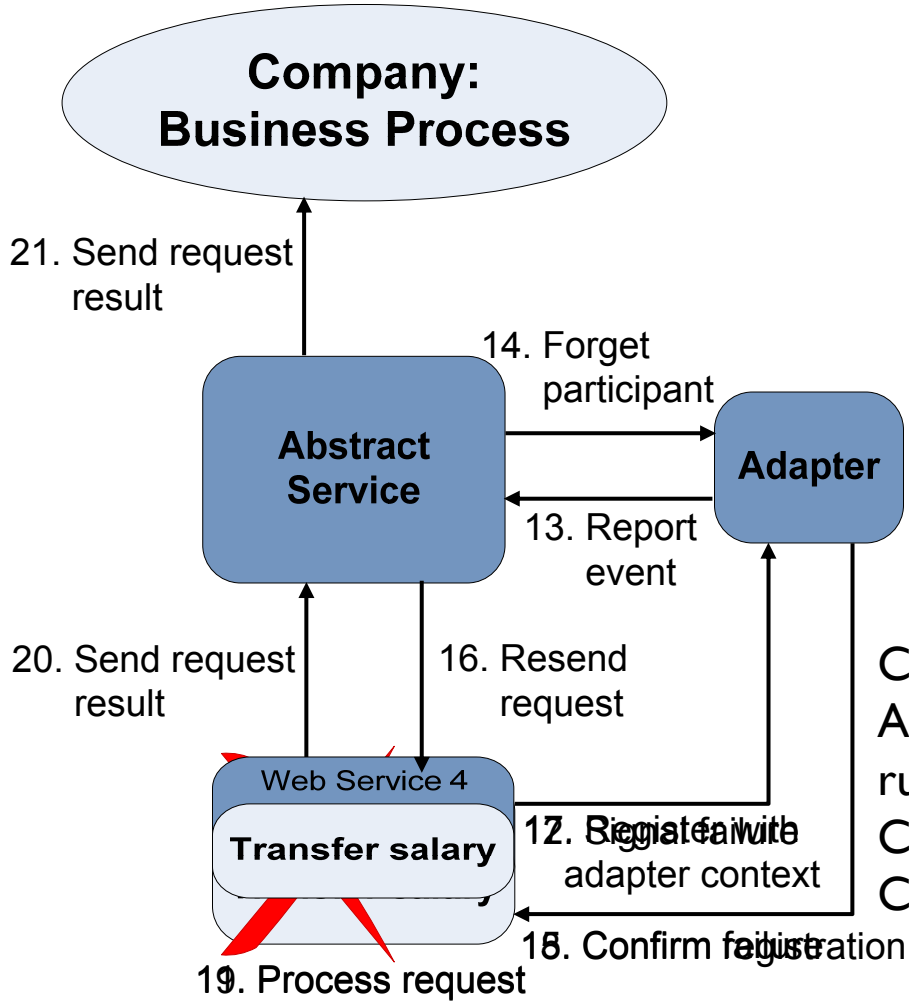


Internal Compensation Handling – No Action



Concrete service fails.
Abstract service checks its compensation rules and contract.
Compensation is not possible.
Normal transaction abort.

Internal Compensation Handling – Replacement



Concrete service fails.
Abstract service checks its compensation rules and contract.
Concrete service is replaced.
Coordinator was not notified!

Evaluation

Multiple scenarios for internal and external compensation handling have been implemented and tested.

An evaluation model has been created, which calculates *net values* for the standard environment and the abstract service environment.

- Allows an assessment whether the utilization of the new design is economical and beneficial.

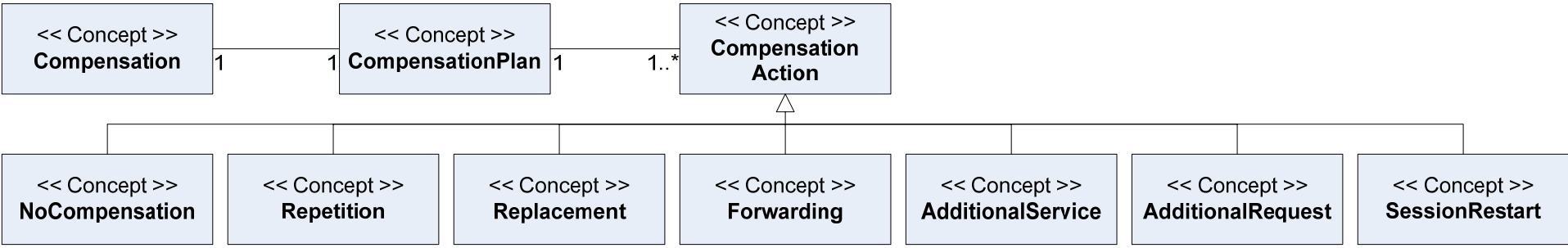
Experiment performed on a simalated environment

More in ACM TWEB paper

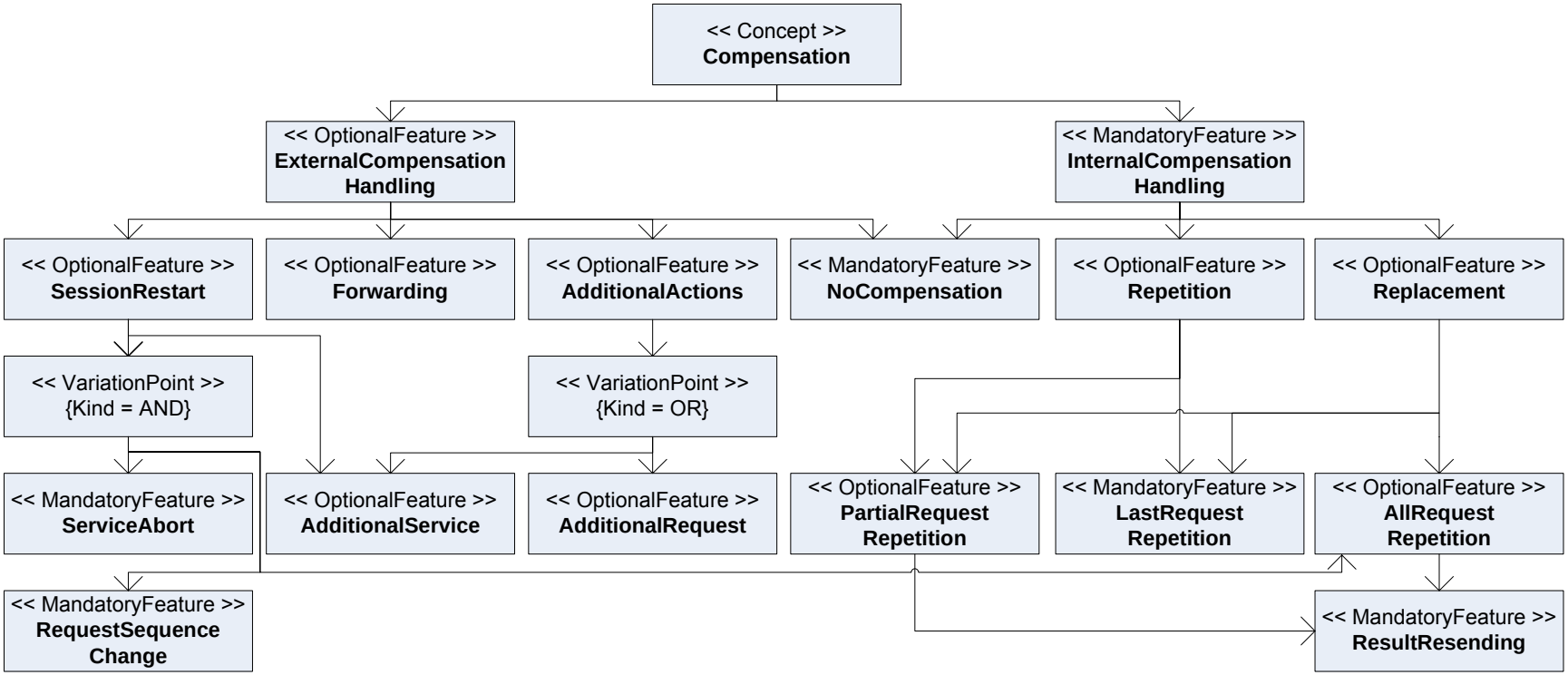
Outline

IWIS group and background
General problem
Business transactions
Middleware for advanced compensations
Service provider and client feature modelling
Matchmaking and restriction model
Further Challenges

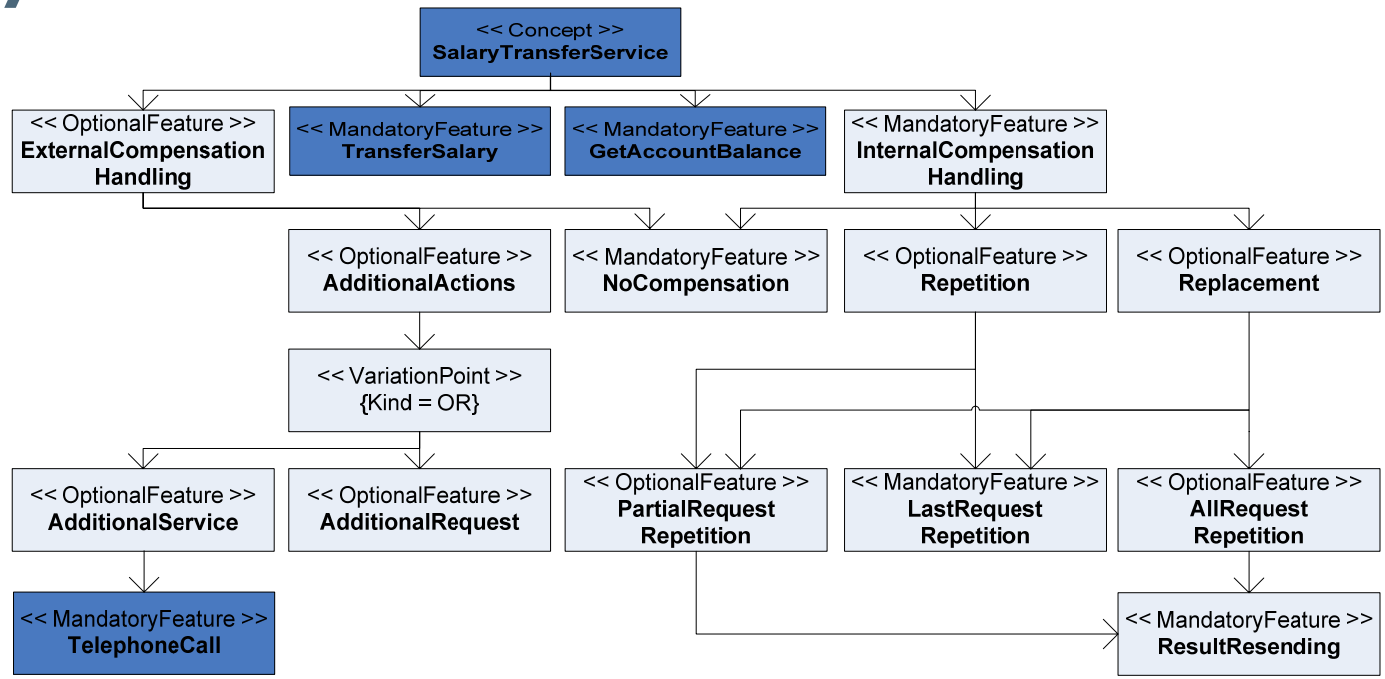
Compensation Types



Compensation Features



Capability Feature Model

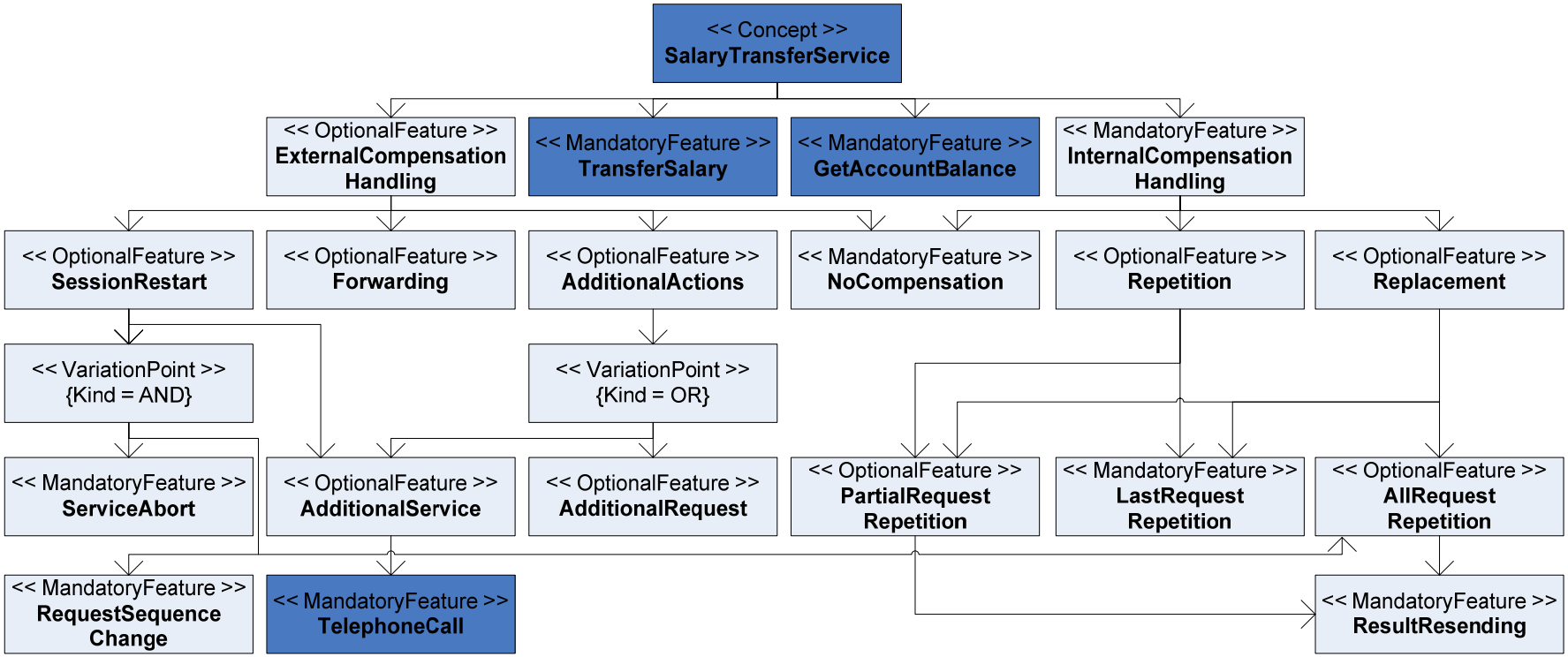


Consists of:

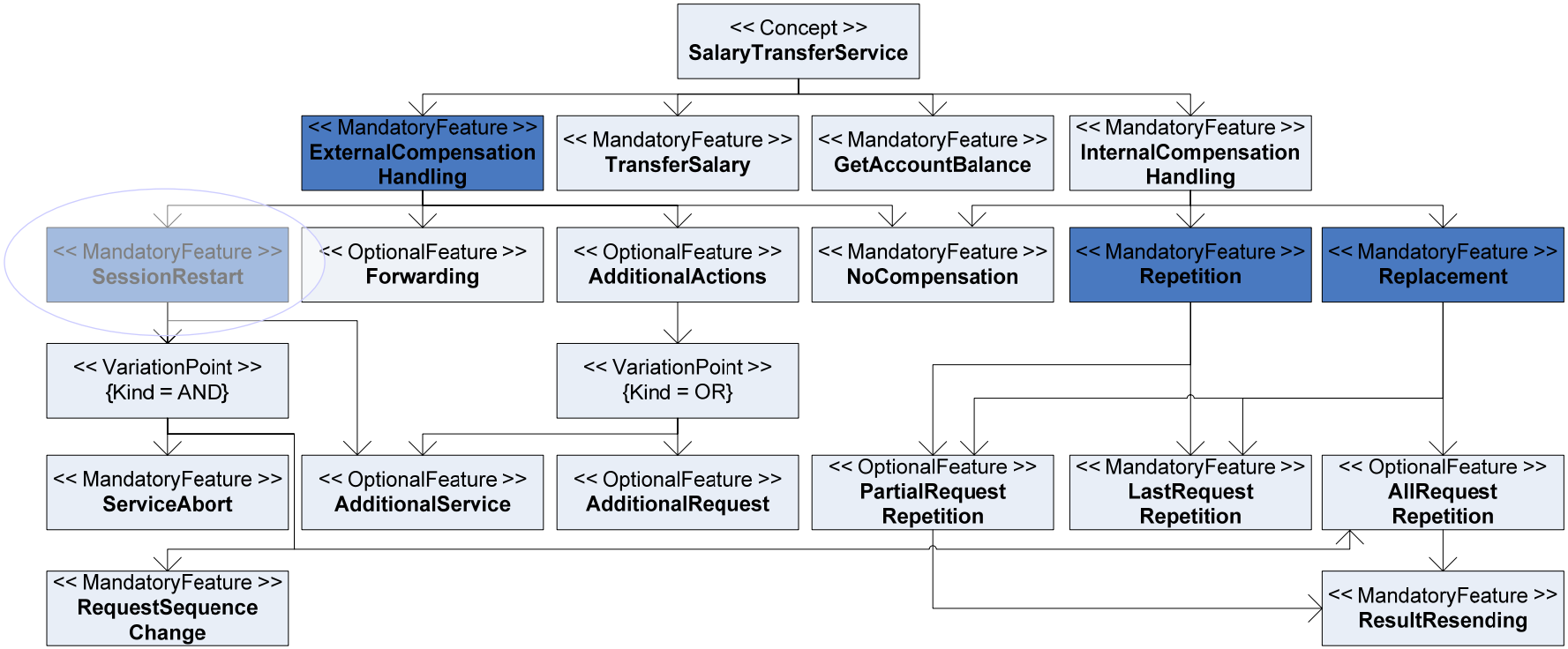
- *functionality feature model*
- *compensation feature model*

The compensation feature model can contain custom features.

Service Capabilities



Consumer Requirements



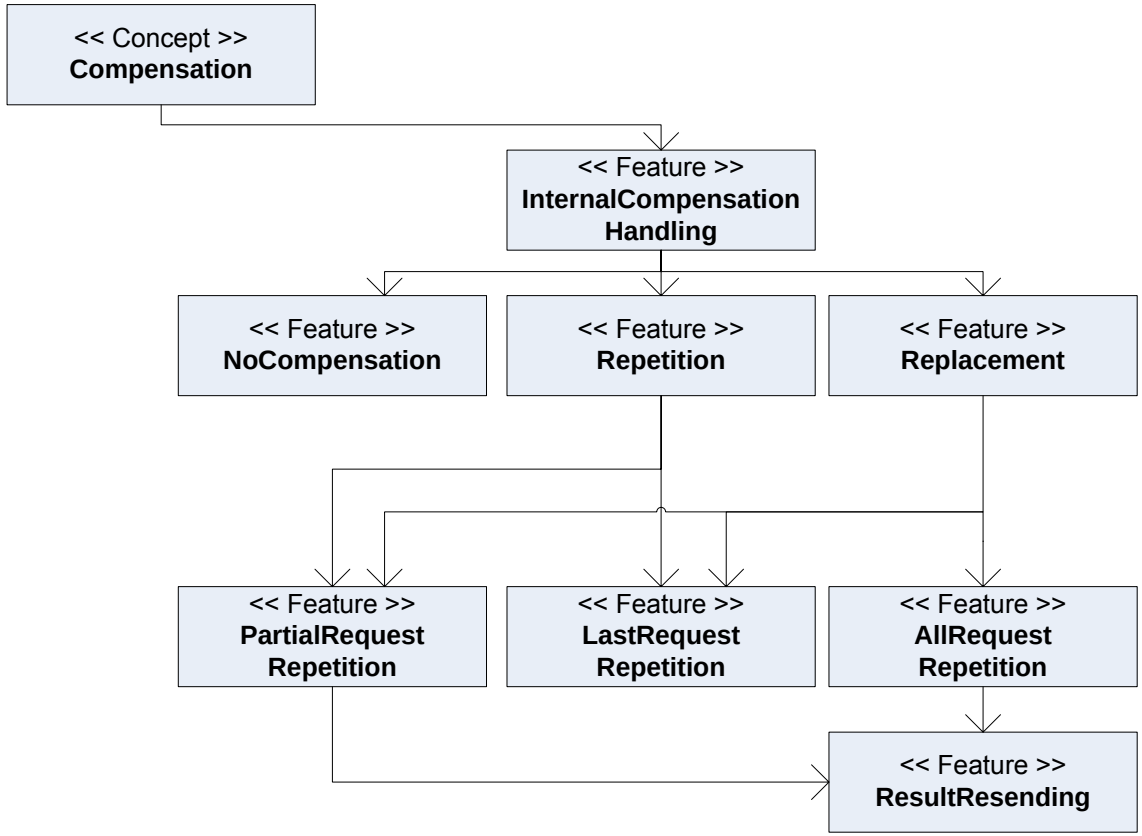
Outline

IWIS group and background
General problem
Business transactions
Middleware for advanced compensations
Service provider and client feature modelling
Matchmaking and restriction model
Further Challenges

Matchmaking between service and consumer feature models

Compatibility score calculation
Iteratively compares feature models
Features must appear at the same place in the graph
Mandatory features must all match but do not contribute to the compatibility score
If a mismatch is found in a mandatory feature, algorithm stops and a negative score is returned
Optional features add to the compatibility score when a match is found (in our case +1)
Additional features may contribute with different scores

Restriction Feature Model



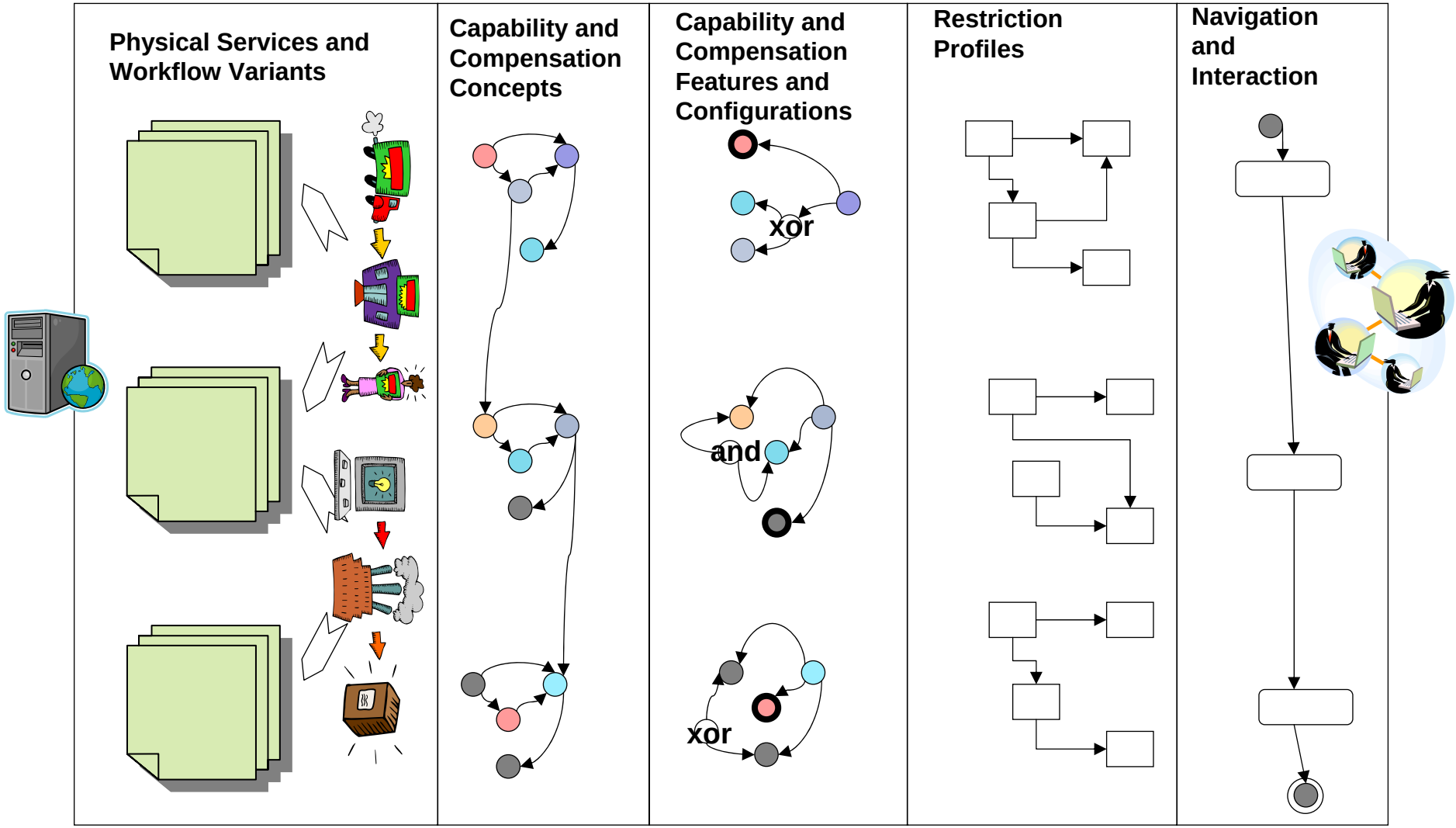
Example: Internal Compensation Rule

```
<cmp:InternalCompensationRule identifier="internalFailureLastRequestResending">
  <cmp:CompensationCondition>
    <cmp:ParticipantEvent eventCode=
   "http://sourceforge.net/projects/frogs/AdapterInteraction/ParticipantFault"/>
    <cmp:ParticipantState
      stateType='http://schemas.xmlsoap.org/ws/2004/10/wsba/Faulting' />
    <cmp:ReplacementService exists="true" isDirectReplacement="true" />
    <cmp:RequestSequence>
      <cmp:Request identifier="transferSalaryMethod" />
    </cmp:RequestSequence>
  </cmp:CompensationCondition>
  <cmp:CompensationPlan>
    <cmp:Compensation>
      <cmp:ServiceReplacement/>
    </cmp:Compensation>
    <cmp:Compensation>
      <cmp:RequestResending lastN="1" />
    </cmp:Compensation>
  </cmp:CompensationPlan>
</cmp:InternalCompensationRule>
```

Feature Model

```
<feature  name="Compensation" type="NONE" id="compensation">
  <feature  name="InternalCompensationHandling" type="NONE"
    id="internalCompensationHandling">
    …
   <feature  name="PartialRequestRepetition" type="NONE"
    id="reference3IXIpartialRequestRepetition">
        <feature  name="ResultResending" type="NONE"
    id="reference3IXIreferenceIXIresultResending">
        </feature>
      </feature>
    </feature>
    <feature  name="Replacement" type="NONE" id="replacement">
      <feature  name="LastRequestRepetition" type="NONE"
    id="reference4IXIlastRequestRepetition">
      </feature>
      <feature  name="PartialRequestRepetition" type="NONE"
    id="reference5IXIpartialRequestRepetition">
        <feature  name="ResultResending" type="NONE"
    id="reference5IXIreferenceIXIresultResending">
        </feature>
      </feature>
      <feature  name="AllRequestRepetition" type="NONE"
    id="reference6IXIallRequestRepetition">
        <feature  name="ResultResending" type="NONE"
    id="reference6IXIreferenceIXIresultResending">
        </feature>
      </feature>
    </feature>
  </feature>
    …
</feature>
```

Layers of Abstraction



Outline

IWIS group and background
General problem
Business transactions
Middleware for advanced compensations
Service provider and client feature modelling
Matchmaking and restriction model
Further Challenges

Workflows vs. Middleware

Compensations and adaptations can be specified at the design level in workflows

Copensations and adaptations can be encoded in an intelligent middleware

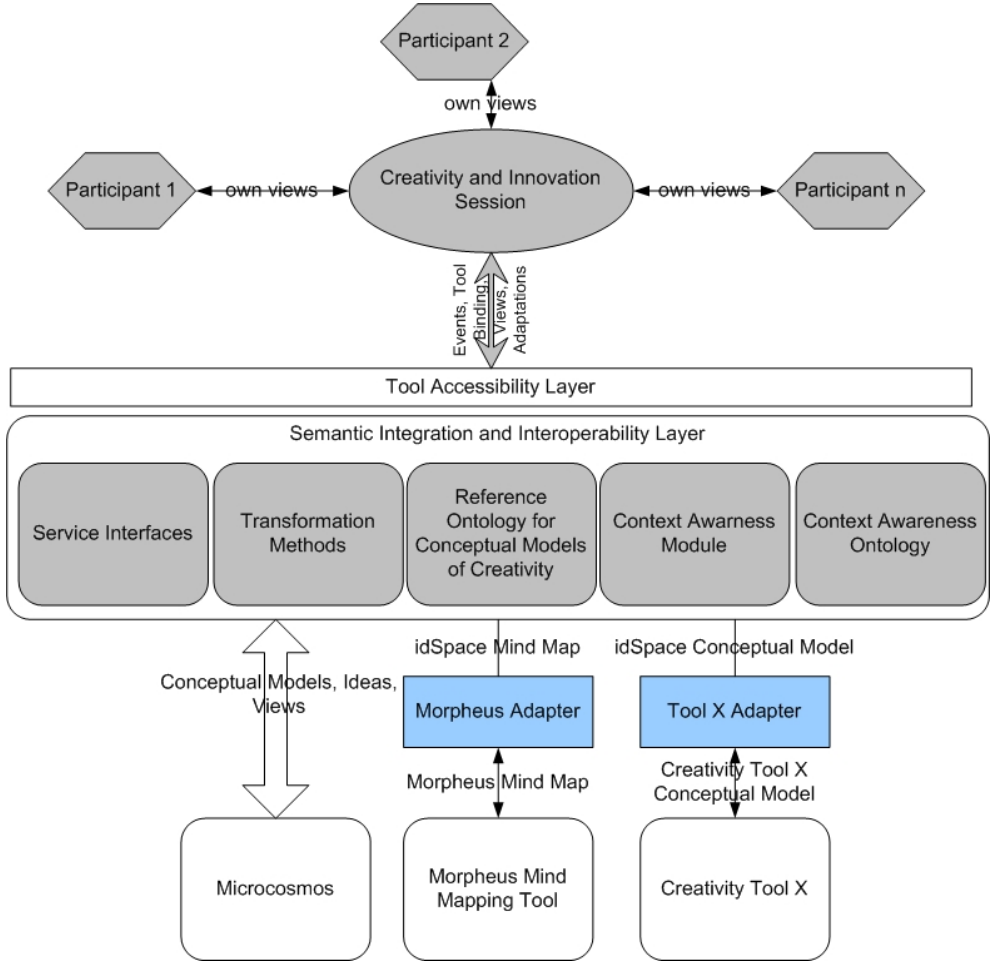
How to combine them

How to compose them

How to ensure consistency

…

FP7 ICT EU idSpace: Tooling of and training for collaborative, distributed product



References

- M. Schäfer, P. Dolog, W. Nejdl: An Environment for Flexible Advanced Compensations of Web Service Transactions. ACM TWEB, 2(2), 2008
- P. Dolog, W. Nejdl: Using UML-based feature models and UML collaboration diagrams to information modelling for web-based applications. UML 2004.

AALBORG UNIVERSITY

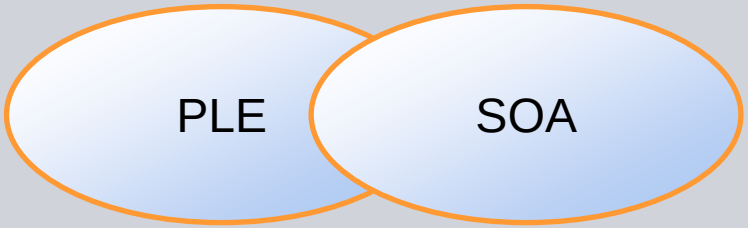
Thanks!!! Questions?

dolog@cs.aau.dk
http://www.cs.aau.dk/~dolog
http//iwis.cs.aau.dk

SIEMENS

Corporate Technology

Combining Product Line Engineering and Service Oriented Architecture in Health Care Infrastructure Systems: Experience Report



Jörg Bartholdt, Bernd Franke, Christa Schwanninger Michael Stal
Corporate Technology & Health Care
Siemens AG

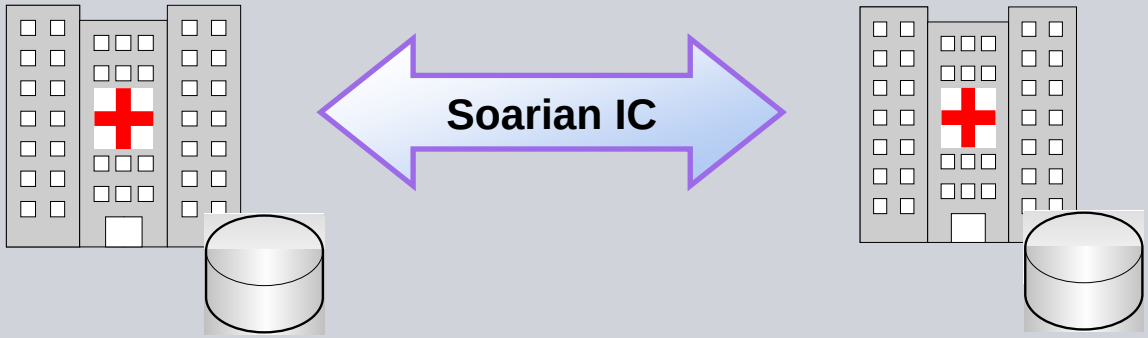
SIEMENS

Business Case

Hospitals have a HIS (Hospital Information System).
Data is shared between departments (intra-hospital)

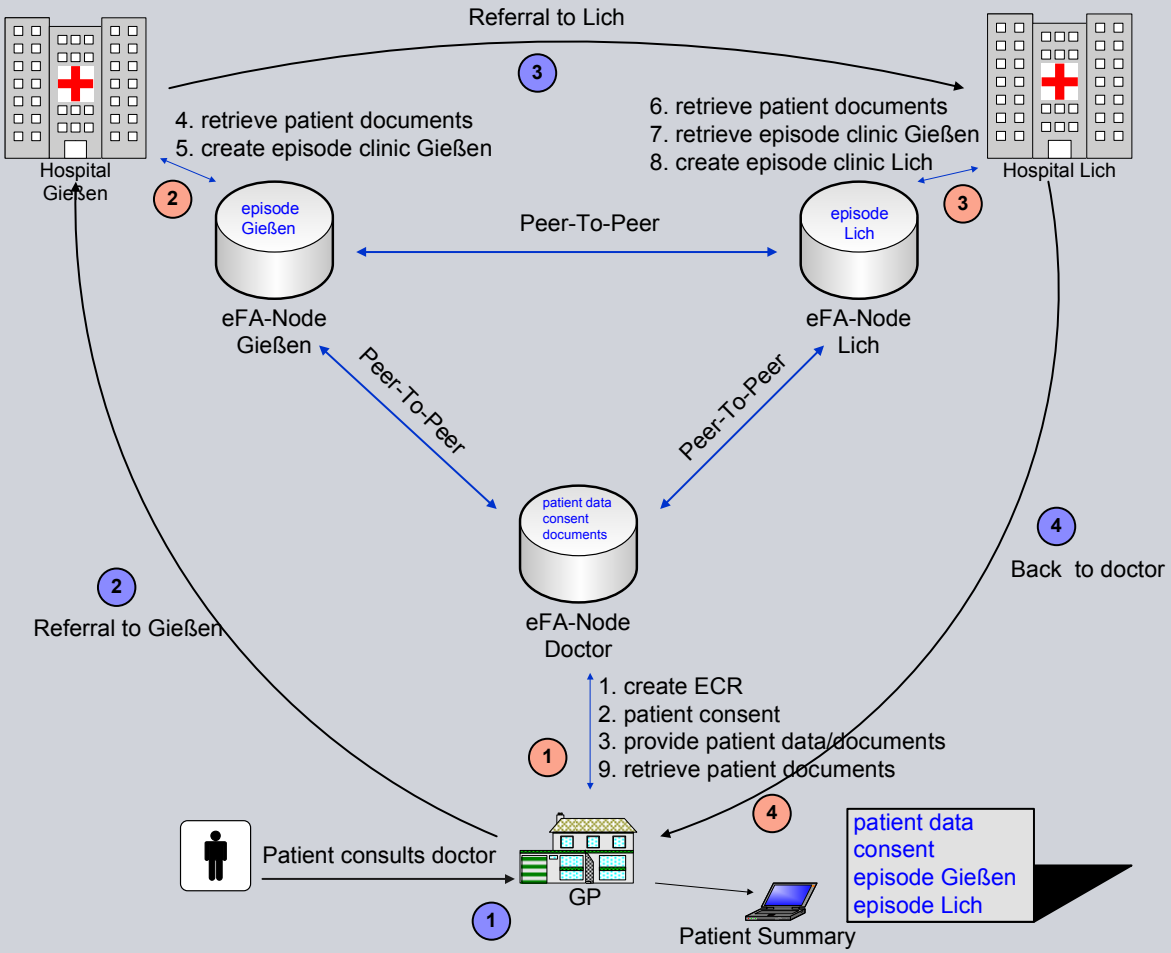
But what if it comes to transferring a patient to another hospital?
You carry your X-ray images with you

Soarian IC targets
- § inter-hospital communication
- § Special scenarios of external data integration



SIEMENS

In future: target residential doctors, too

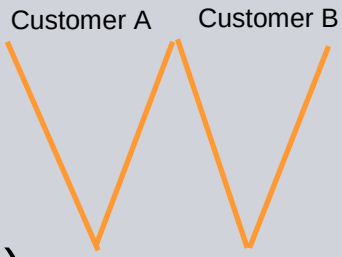


J. Bartholdt, B. Franke, C. Schwanninger M. Stal

SIEMENS

History

Product development was serialized
Previous version forms the bases for the next version (architecture erosion)
Results in monolithic application, interwoven dependencies



Assumptions:
Increased customer base (no serialization possible anymore)
Focus on main selling assets
Make system ready for integration

Goal:
Introduce SOA-approach: import/export via interfaces, composition of features via service chaining
Introduce PLE: focus on core assets, allow for customer specific variations, introduce new features in core if proven at one customer

SIEMENS

Challenges

1. *Increasing variability*
2. *Configurability/Subset-ability*
3. *Extensibility*
4. *Increased testability*
5. *Outsourcing*
6. *Risk effect mitigation*
7. *Exploitation of COTS (Common-Off-The-Shelf) products*
8. *Prioritization of features to be integrated in the platform*
9. *Positioning in the market (guide the customer)*
10. *Acceleration of tender preparation*
11. *Clinical workflows*
12. *Traceability*

SIEMENS

Approach

1. Scoping (2,8,9,10):
   - § Increasing customer base requires focus on most profitable features
   - § Starting point: Group current requirements to features
   - § Use feature model for reasoning with product mgmt, sales, development, etc („common language“)
2. Variability Management (1,3,4,12):
   - § Reduce variability points (expensive!) pre-configurations
3. Building re-use culture (1,2,4,10):
   - § Keep clear product portfolio strategy
   - § Focus to market commonalities
   - § Quick hacks forbidden in the core assets

SIEMENS

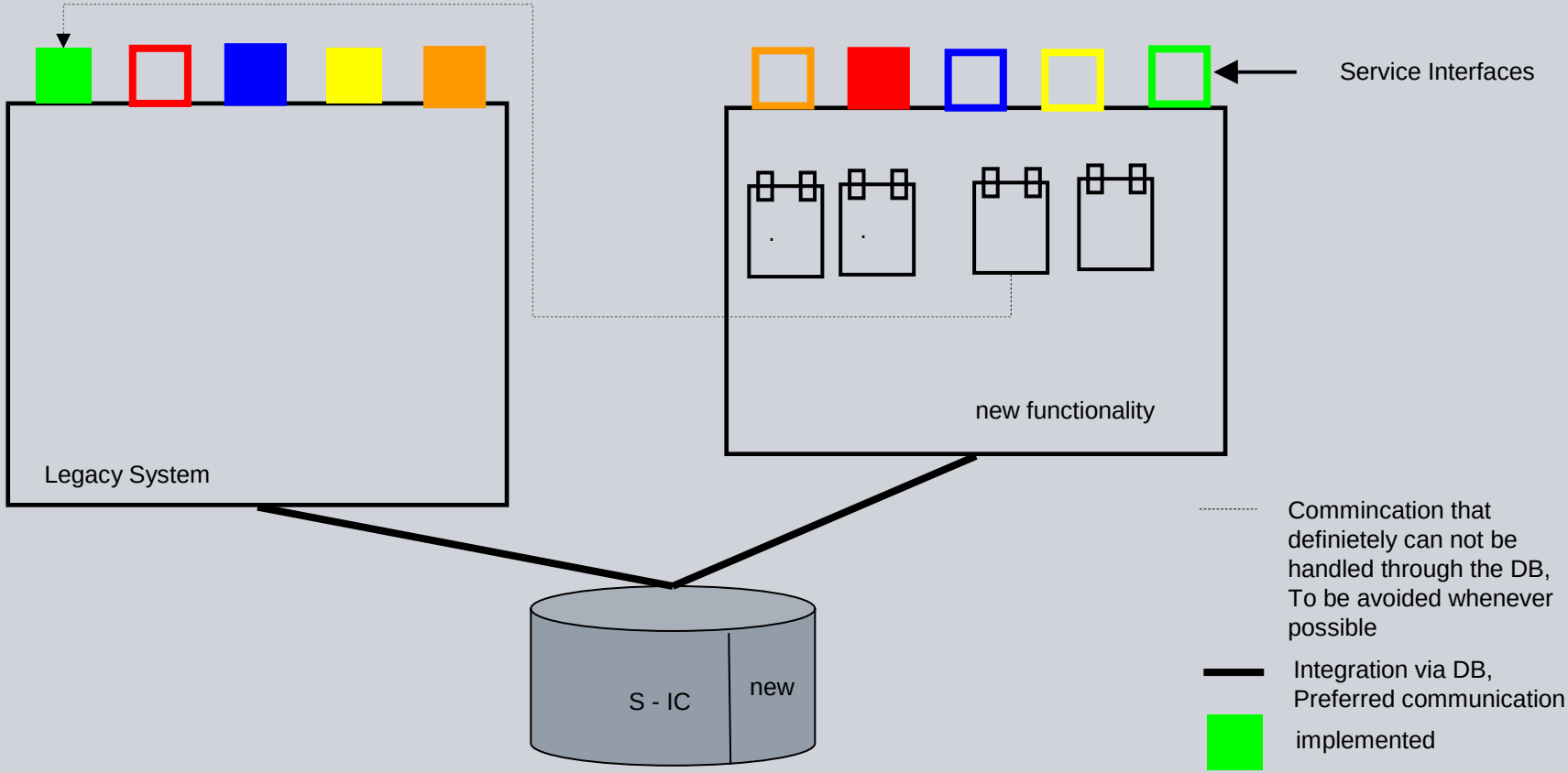
Approach

4. Self-containment (2,3,4,5,6,12):
   - § Fosters decoupling of components
   - § Allows for exchange to third-party components
   - § Allows to be used as a system, not only by humans via Web-Interface
   - § Improves testability
5. Integration (2,7):
   - § More freedom to tailor to customer needs
   - § Face the fact that Siemens is not the only supplier
6. Flexibility (5,11):
   - § Adding workflow or rule engines
   - § support specifics of each customer (ideally by the customer)
   - § Late (dynamic) binding

SIEMENS

Approach

Other projects showed the likelihood of failure in a big-bang approach
We favor a migration strategy



J. Bartholdt, B. Franke, C. Schwanninger M. Stal

SIEMENS

Conclusion

- SOA build a prominent, natural variation point with late (dynamic) binding capabilities
- Services as a variation point means flexible tooling available (Workflow engines, BPEL)
- Self-containment reduces coupling and fosters variation
- We will not follow the total unawareness of the usage context implied by SOA protagonists.

Future challenges

- Data model can not be changed as long as old application components exist
- Restructure the organization (nobody wants to loose influence, learning-curve)
- Wrap legacy system with new service interface without side-effects

SIEMENS

Questions & Answers

Now, or later …

Joerg.Bartholdt@Siemens.com

Semantic Web Services-based Reasoning in the Design of Software Product Lines

J. Jeffrey Rusk and Dragan Gasevic

Athabasca University

Canada

Research Goal

To evaluate the suitability of the Web Service Modeling Ontology (WSMO) in the encoding of product configurations and related constraints from a software product line (SPL) in such a manner as to better enable reasoning approaches which facilitate higher automation of service discovery, composition, invocation, and monitoring in service oriented architectures (SOA).

Outline

- Background and Motivation
- Feature Models (FM)
- Web Service Modeling Ontology (WSMO)
- Model Transformations
  - FM to WSMO
  - Product Configuration to WSMO
- Orchestration in WSMO
- Reasoning
- Implementation, Conclusion and Future Work

Background Issues

- Impediments to successful implementation of SPL when considering SOA
- Challenges representing SOA as SPL
- Limits to the expressiveness of FM
- Limited reasoning capabilities
- Ontology-related technology exists to support

Deliverables

- Mappings between FM and WSMO
- Transformation implementation
- Reasoning framework

What do the deliverables make possible?

The ability to explore and evaluate:

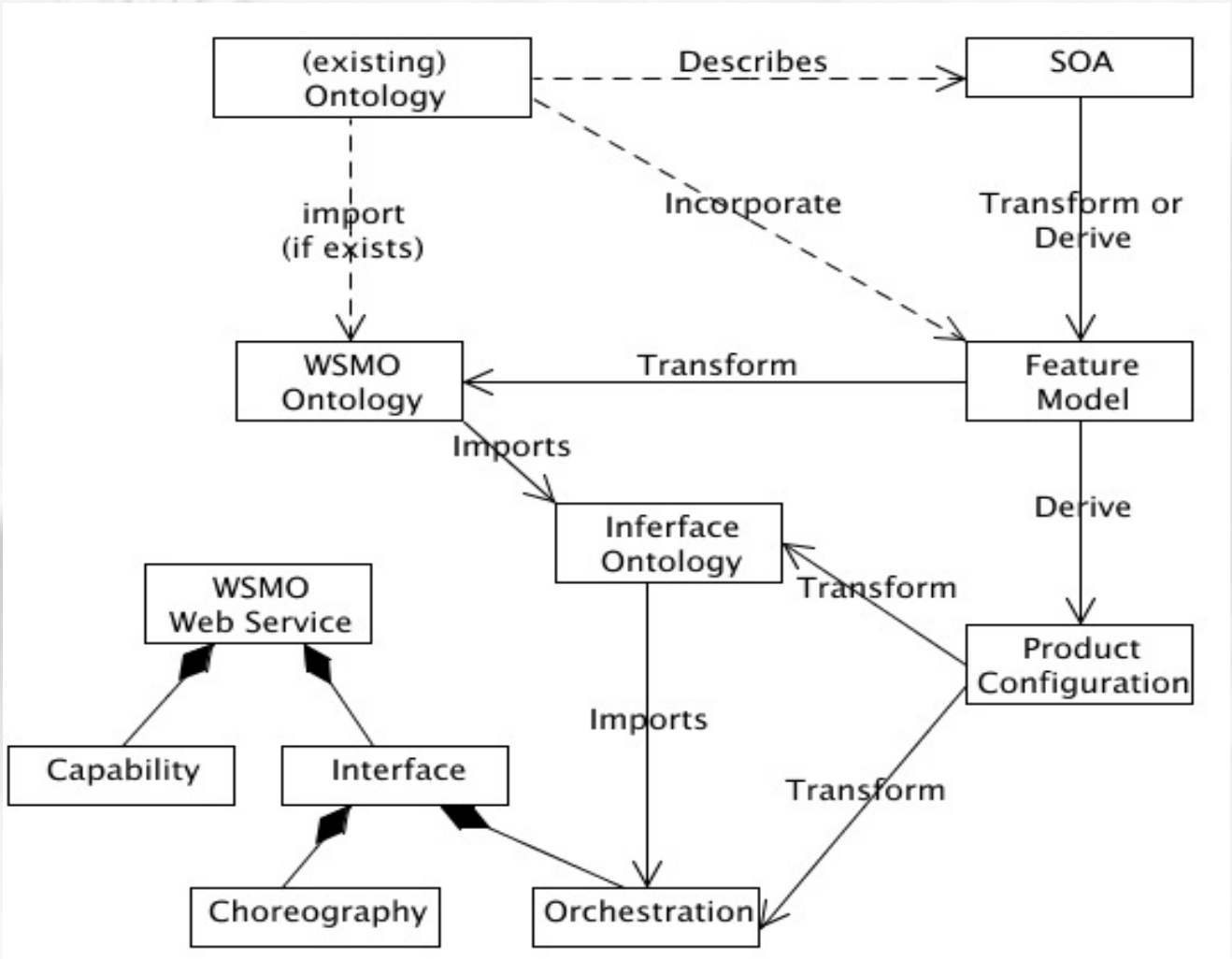
- accuracy of the mapping possible between the two formalisms.
- level of automation supported during transformation
- support or guidance that the ontology can provide to feature modeling.

Themes of this Workshop

- Variability and variability mechanisms
- Product composition

How does this work relate to these themes?

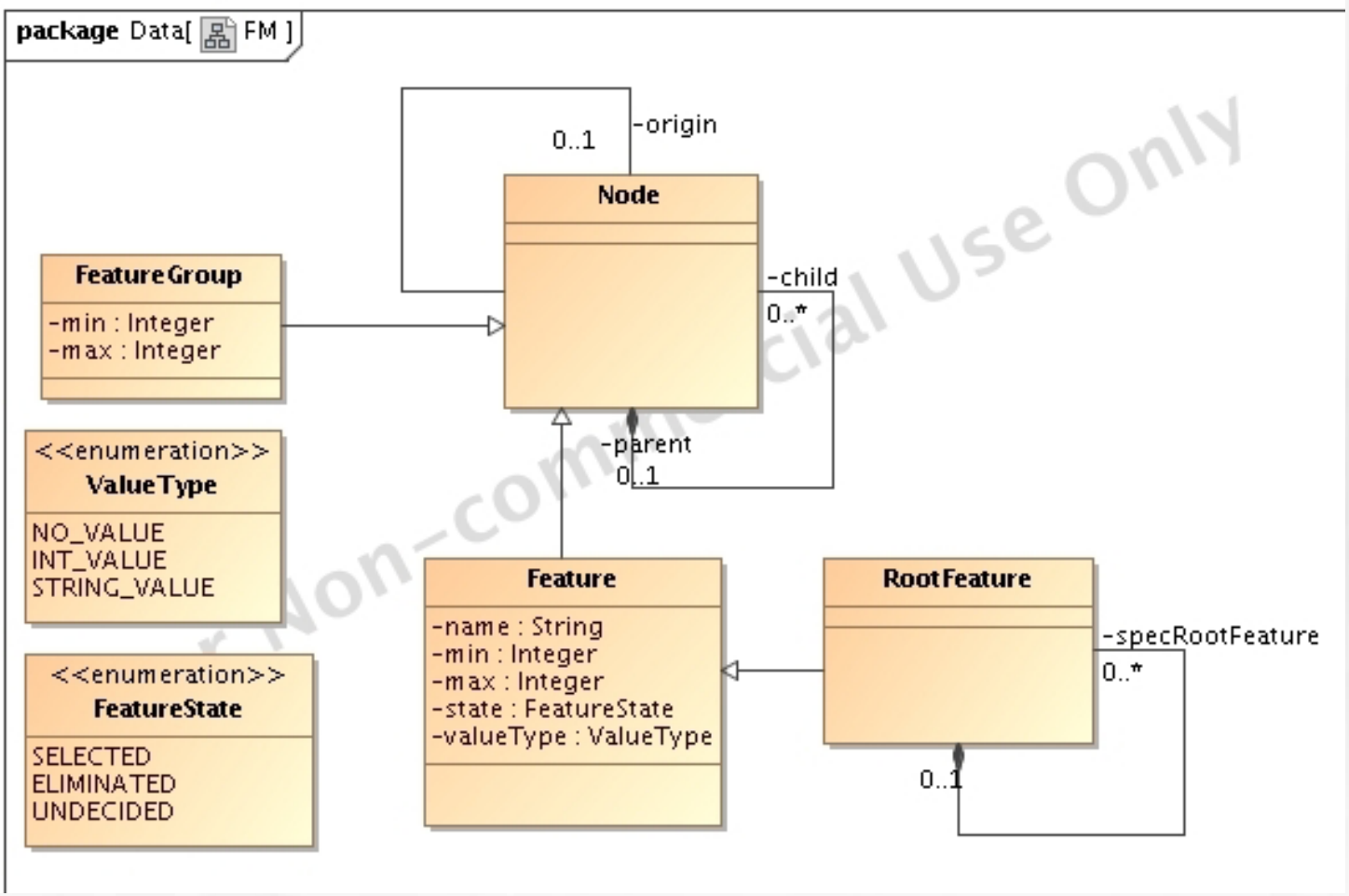
Overall Flow of Information



Feature Models

- SPL implementations typically feature-based
- FM ideal representation for SOA
- Using Czarnecki et al. notation and rendering
- Metamodel of FM and product configurations
- Tool support

Feature Model Metamodel

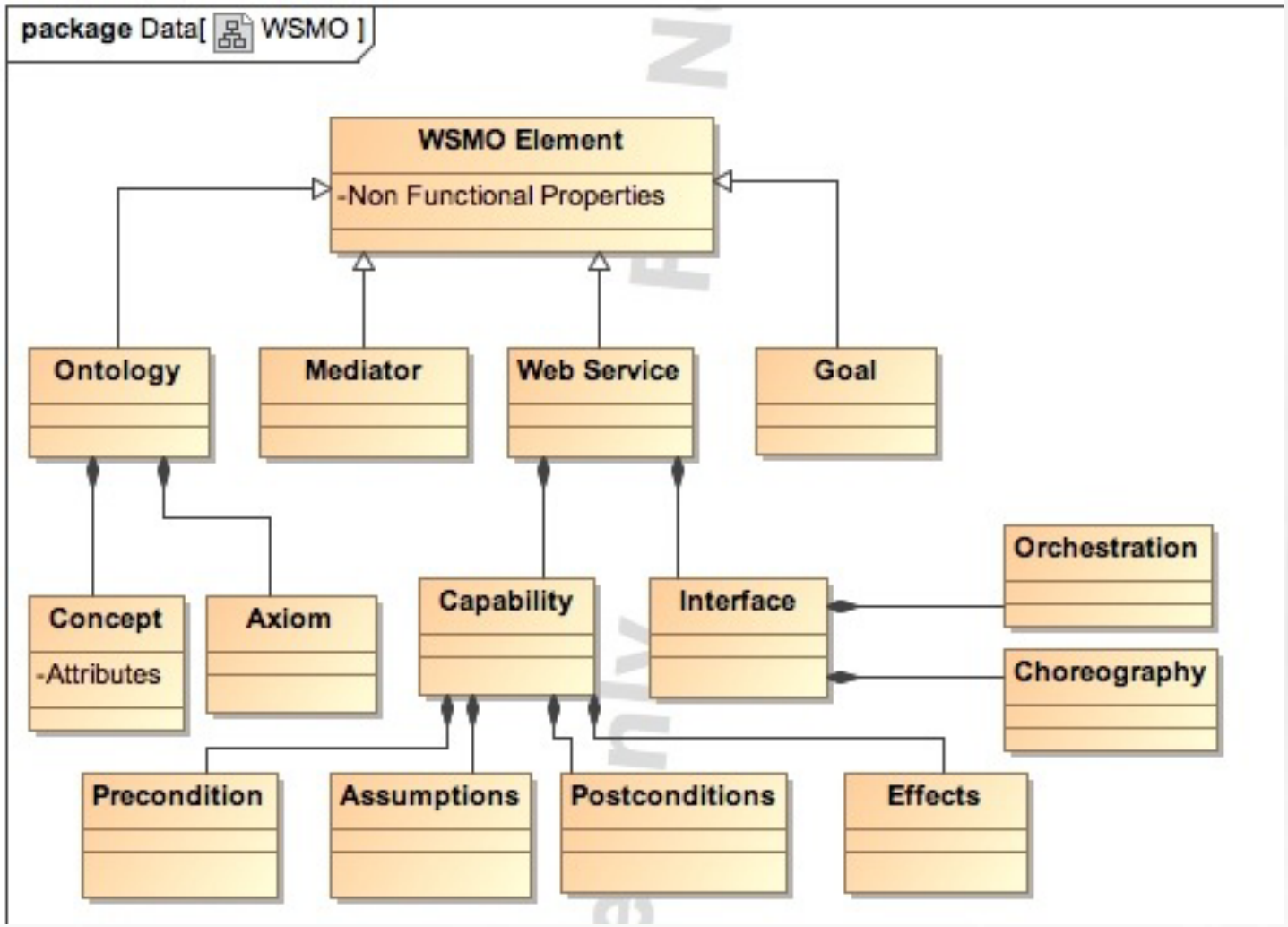


Adapted from: C.H.P. Kim, K. Czarnecki. Synchronizing cardinality-based feature models and their specializations. In *Model Driven Architecture – Foundations and Applications. 331-348. 2005.*

Web Service Modeling Ontology (WSMO)

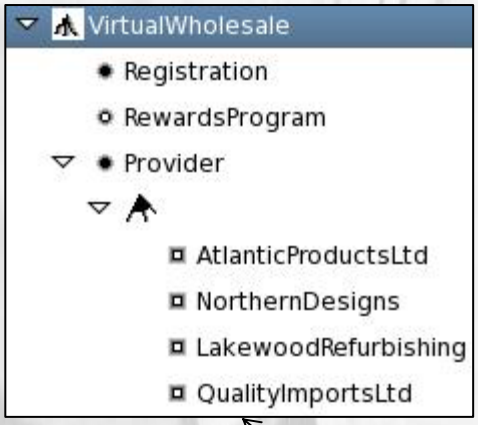
- Semantic describes all aspects of SWS
- Relatively new framework
- Tool support
- Four core elements
  - Ontologies
  - Web Services
  - Goals
  - Mediators

WSMO Metamodel



Model Transformation

Feature Model



WSMO

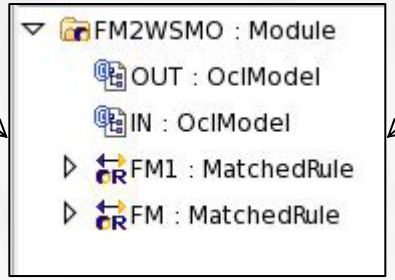
```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"
namespace { _"http://www.jeffrusk.org#"
,
     wsmostudio _"http://www.wsmostudio.org#" }

ontology VirtualWholesale
     nonFunctionalProperties
          wsmostudio#version hasValue "0.7.2"
     endNonFunctionalProperties

concept VirtualWholesale
     Registration impliesType  (1 1) Registration
     RewardsProgram impliesType  (0 1) RewardsProgram
     Payment impliesType  (1 1) Payment
     Provider impliesType  (1 4) Provider
     Shipping impliesType  (1 4) Shipping

concept Registration
```

ATL



XML Formats

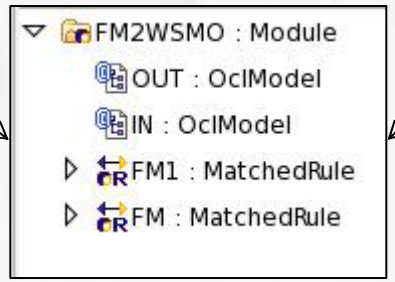
Feature Plugin XML Export

```
<feature min="1" max="1" name="VirtualWholesale" type=
  <feature min="1" max="1" name="Registration" type="I
  </feature>
  <feature min="0" max="1" name="RewardsProgram" type=
  </feature>
  <feature min="1" max="1" name="Provider" type="NONE"
    <featureGroup min="1" max="4" id="group">
      <feature min="0" max="1" name="AtlanticProductsL
      </feature>
      <feature min="0" max="1" name="NorthernDesigns"
      </feature>
      <feature min="0" max="1" name="LakewoodRefurbish
      </feature>
      <feature min="0" max="1" name="QualityImportsLtd
      </feature>
    </featureGroup>
  </feature>
```

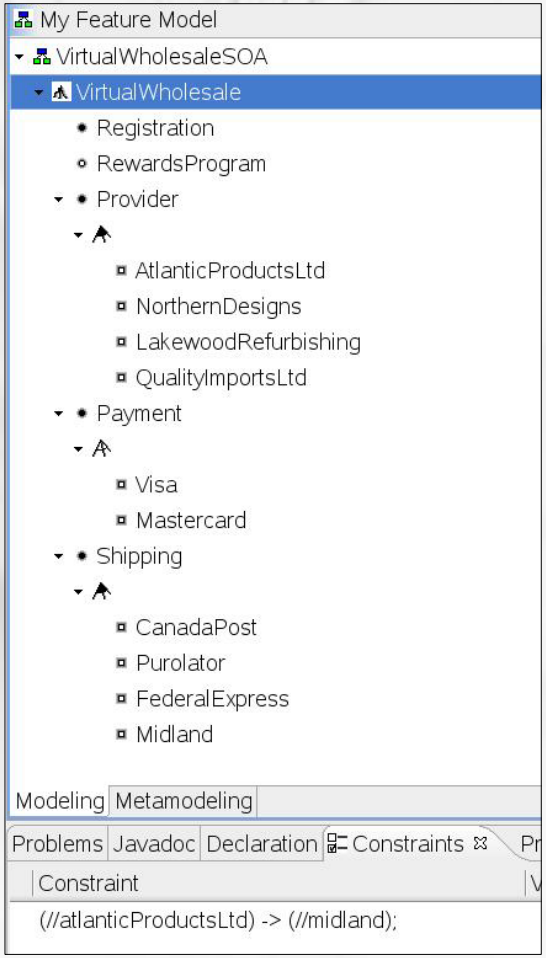
WSML-in-XML

```
<?xml version="1.0" encoding="UTF-8"?>
<wsml>
<ontology name="http://www.example.org#Virtual
<nonFunctionalProperties>
<attributeValue name="http://www.wsmostudio.or
<value type="http://www.wsmo.org/wsml/wsml-syn
</attributeValue>
</nonFunctionalProperties>
<concept name="http://www.example.org#Virtual
<attribute name="http://www.example.org#Regist
<range>http://www.example.org#Registration</ra
<minCardinality>1</minCardinality>
<maxCardinality>1</maxCardinality>
</attribute>
<attribute name="http://www.example.org#Reward
<range>http://www.example.org#RewardsProgram</
<maxCardinality>1</maxCardinality>
```

ATL



Feature Model to WSMO




```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"

ontology VirtualWholesale

concept VirtualWholesale
    Registration ofType  (1 1) Registration
    RewardsProgram ofType  (0 1) RewardsProgram
    Payment ofType  (1 1) Payment
    Provider ofType  (1 4) Provider
    Shipping ofType  (1 4) Shipping

concept Registration
concept Payment
concept Provider
concept Shipping
concept RewardsProgram
concept Visa subConceptOf Payment
concept Mastercard subConceptOf Payment
concept AtlanticProductsLtd subConceptOf Provider
concept NorthernDesigns subConceptOf Provider
concept LakewoodRefurbishing subConceptOf Provider
concept QualityImportsLtd subConceptOf Provider
concept CanadaPost subConceptOf Shipping
concept Purolator subConceptOf Shipping
concept FederalExpress subConceptOf Shipping
concept Midland subConceptOf Shipping

axiom DisjointPayment
    definedBy
        !- ?x memberOf Visa
  and ?x memberOf Mastercard.
```

Product Configuration to WSMO

- Most accurately represented as orchestration
- Overall executable business process that can be defined through interaction between Web services
- Choreography may be a factor as well

Orchestration in WSMO

- Unlike choreography, orchestration in WSMO is still under development
- Both based on abstract state machine
- Composed of state and set of guarded transitions
- State in form of ontology providing
  - Vocabulary for transition rules
  - Set of instances that change state.

Reasoning

- Effects of:
  - Product configuration choices
  - Adding, moving, deleting features
  - Assigning values to attributes
- Guidance for:
  - Constraints in ontology not present in FM
  - FM relationships not represented in ontology
  - Orchestration dependencies
  - Orchestrating services required

Implementation

- Feature Model Plugin
- WSMO Studio and KAON2 Reasoner
- ATL
- Eclipse

Conclusion

- Mappings between FM and WSMO
  - Accuracy of mappings
  - Level of automation attainable
  - Precision of feature discovery
  - Guidance provided by ontology
- Suitability of WSMO
  - Expressiveness
  - Related work

Future Work

- Refine mappings
- Improve transformation
- Explore further the available reasoning
- Integrate the various utilities into comprehensive plugin working in Eclipse environment

Thank you!
Questions?

Jeff Rusk
Director, Technical Services
Nunavut Impact Review Board
Cambridge Bay, Nunavut, Canada
(also MSc Candidate at Athabasca University)

jrusk@nirb.ca

Acknowledgements

The authors acknowledge the support of Athabasca University through the Graduate Student Research Fund.

Very Large Business Applications Lab

Service-Oriented Product Lines: Towards a Development Process and Feature Management Model for Web Services

SOAPL 2008

http://www.vlba-lab.de/

17. September 2008

Fakultät für Informatik
Institut für betriebliche und technische Informationssysteme
Arbeitsgruppe Wirtschaftsinformatik

Otto-von-Guericke Universität
Magdeburg

Introduction

- **What is software development?**
  - Usage of a software development process
  - Transform requirements into different artifacts (architectural descriptions, interface descriptions, source code…)
- **How to manage artifacts?**
  - Apply changes to existing artifacts
  - Reduce coupling of source code
- **What about reuse?**
  - Commonality and variability
- **Combination of Software Product Lines and Service-Oriented Architectures provides solutions to many common software problems**

Structure

- Introduction
- Definitorial Background
- Development Process for Software Product Lines
- Service-Oriented Product Lines
- Example
- Conclusions

Definitional Background: Software Product Lines

- **Withey: „Product Lines is a group of products sharing a common, managed set of features“ [1]**
- **Specifically, manage variability among features that represent requirements**
- **Goal: Structure and reuse software development artifacts**

Definitional Background: Service-Oriented Architectures

- **Loosely coupled and autonomous services**
- **Properties according to Josuttis: self-containment, coarse-grained interfaces, reusability and composability [2]**
- **Implementation: Web Services or Enterprise Service Bus**

Definitional Background: Web Services

- **„Software applications that can be discovered, described and accessed based on XML and standard Web protocols“ [3]**
- **Described by a WSDL**
  - Abstract definition describes interface, operations and messages
  - Concrete definition describes bindings to operations
- **Distinguish into service broker, provider and consumer**

Structure

- Introduction
- Definitorial Background
- Development Process for Software Product Lines
- Service-Oriented Product Lines
- Example
- Conclusions

Domain Engineering

- **Develop a software family**

- **Analysis**
  - Capture domain specific knowledge
  - Develop a domain model
  - Represent domain concepts and requirements in a central feature model
  - Identify variants with their distinguishing features

- **Design**
  - From architectural description to software entities
  - Decide used frameworks, libraries and programming languages
  - Form technological foundation for implementation of variants

- **Implementation**
  - Make or buy decision for software entities

Application Engineering

- **Develop individual member (of the software family)**

- **Five steps**
  - Problem Analysis (overall problem specification)
  - Product Specification (concrete set of selected features)
  - Collateral Development (Documentation)
  - Product Implementation (Executables and test cases)
  - Deployment

Structure

- Introduction
- Definitorial Background
- Development Process for Software Product Lines
- Service-Oriented Product Lines
- Example
- Conclusions

Service-Oriented Product Lines

- **Implement SPL with an SOA**

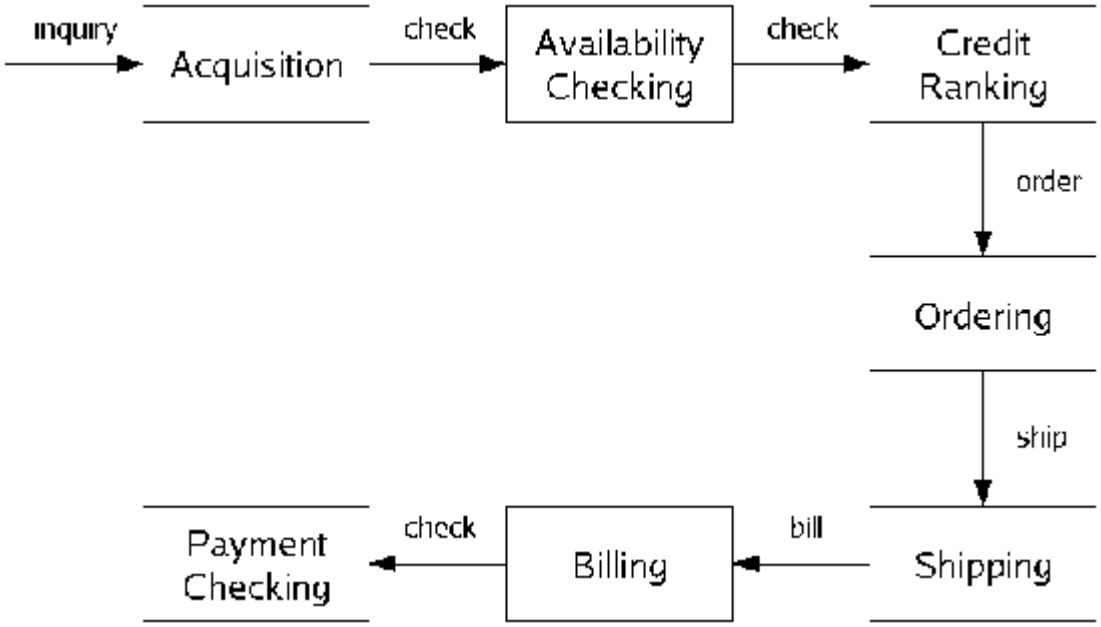
- **Different impacts on development phases**
  - Analysis:
    - Select SOA-specific modeling languages
    - Software requirements can be modeled as features or part of the ESB
  - Design
    - ESB as routing and messaging backbone, and also implements e.g. compliance requirements
    - ESB mostly forms common part of SPL
    - Web Service abstracts whole applications, databases or fine granular software entities
  - Implementation
    - Careful choice of purchased ESB
    - Wrap existing software with Web Services or use web service repositories
    - Full SOPL process (design interface and implementation) vs. light SOPL process (design only interfaces)

Structure

- Introduction
- Definitorial Background
- Development Process for Software Product Lines
- Service-Oriented Product Lines
- Example
- Conclusions

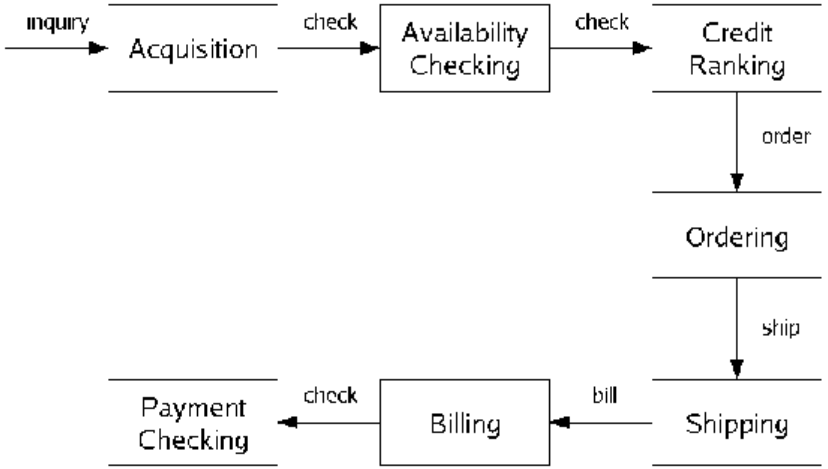
The Web Store

- **Domain Engineering for a Web Store (base taken from [4])**
- **Web store contains 7 modules:**



Web Store: A Feature Model

- **Abstract representation of the overall product line**
- **Uses set-like notations for features**
  - *Base = {Acq,Chk,Crd,Ord,Shp,Bil,Pay}*



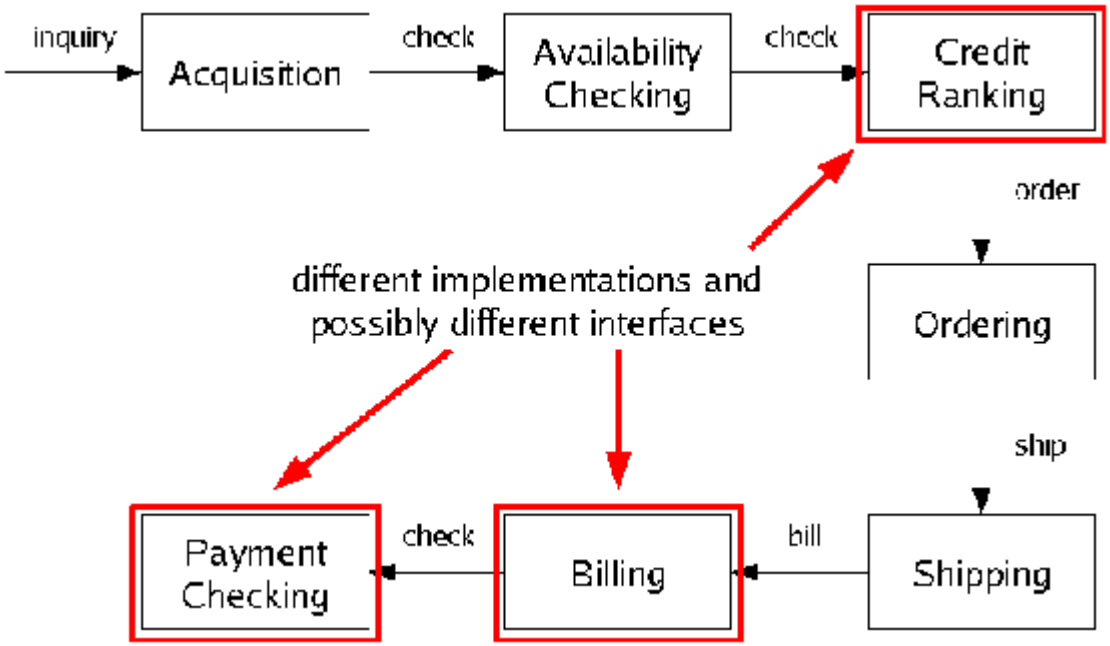
- **Detail out features**
  - Credit Ranking: Use an independent agency (Agc) or explanation of the bank (Bak)
    - *Crd = {Agc, Bak}*
  - Shipment via surface (Sur) or airmail (Air)
    - *Shp = {Sur, Air}*
  - Surface shipment with standard (Std) or Express (Exp) Mail
    - *Sur = {Std, Exp}*

- **Individual member is a composition of specific features**
  - *Store1 = Base*
  - *Acq • Chk • Agc • Bak • Ord • Std • Exp • Air • Bil • Pay*

Web Store: Variability Management

- **Customers demand new features**
  - Discounting for bigger quantities of ordered goods
  - Traceability of features

- **Impacts existing services of the Web Store**



Web Store: Variability Management 2

- **Discounting concepts refines four basic features**
  - Disc = {Crd = {ΔAgc, ΔBak}, ΔBil, ΔPay}

- **Build a new member**
  - Include discounting feature
  - Limit shipment to standard surface mail
  - Store2 = {Base – {Exp, Air}} • Disc
  - Store2 = Acq • Chk • Agc • Bak • Ord • Std • Bil • Pay • Disc
  - Store2 = Acq • Chk • Agc • Bak • Ord • Std • Bil • Pay • ΔAgc • ΔBak • ΔBil • ΔPay

- **Combination of a basic and refined feature leads to the final representation**
  - Store2 = Acq • Chk • Agc´ • Bak´ • Ord • Std • Bil´ • Pay´

Web Store: Variability Management with WSDL

- **Description of Web Services with WSDL gives a high level view**
- **Feature granularity must manage WSDL descriptions**
- **Example: WSDL for Billing**

```
<element name="CalcBillOutput">
 <!– Other definitions ommitted -->
   <xsd:sequence>
     <xsd:element name="customerName“ type="xsd:string"/>
     <xsd:element name="customerAddress“ type="xsd:string"/>
     <xsd:element name="items" type="ItemOrder“ minOcurs="1" maxOccurs="unbound"/>
     <xsd:element name="totalPrice“ type="xsd:integer"/>
   </xsd:sequence>
  <!– Other definitions ommitted -->
</element>
```

- **Variability Management with XAK [5]**

```
<element name="CalcBillOutput“ xak:artifact="STOREbillOutput">
  <!-- Other definitions omitted --!>
    <xsd:sequence xak:module="billOutput">
      <xsd:element name="customerName" type="xsd:string"/>
<!-- Other definitions omitted --!>
```

Web Store: Variability Management with WSDL 2

WSDL Refinement

```
<xak:refines xak:artifact="STOREbillOutput">
  <xak:extends xak:module="billOutput">
    <xak:super xak:module="billOutput"/>
    <xsd:element name="discount" type="xsd:integer"/>
    <xsd:element name="discountedPrice“ type="xsd:integer"/>
</xak:extends>
</xak:refines>
```

Combined WSDL

```
<element name="CalcBillOutput">
 <!– Other definitions ommitted -->
    <xsd:sequence>
      <xsd:element name="customerName“ type="xsd:string"/>
      <xsd:element name="customerAddress“ type="xsd:string"/>
      <xsd:element name="items" type="ItemOrder“ minOcurs="1" maxOccurs="unbound"/>
      <xsd:element name="totalPrice“ type="xsd:integer"/>
      <xsd:element name=“discount“ type="xsd:integer"/>
      <xsd:element name=“discountedPrice“ type="xsd:integer"/>
    </xsd:sequence>
  <!– Other definitions ommitted -->
</element>
```

Conclusions

- Introduction
- Definitorial Background
- Development Process for Software Product Lines
- Service-Oriented Product Lines
- Example
- Conclusions

Conclusions

- **Feature models and variability management models can be used for Service-Oriented Product Lines as well**
- **XML refinements allow practical solution to feature management**
- **Focus on models leads to a high-level view**
- **Promising**
  - If existing code base can be reused efficiently: focus on light SOPL process (only define interfaces)
  - Introduce Domain Specific Languages for domain modeling and SPL configuration, allowing participation of end-users

Fin

http://www.vlba-lab.de/

Thanks for your attention!

References

- [1] J. Withey, “Investment analysis of software assets for product lines,” Software Engineering Institute, Carnegie Mellon University, Technical Report CMU/SEI-96-TR-10, 1996.
- [2] N. M. Josuttis, SOA in Practice: The Art of Distributed System Design. Sebastopol, California, USA: O’Reilly Media, Inc., 2007.
- [3] M. C. Daconta, L. J. Obrst, and K. T. Smith, The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management. Indianapolis, Indiana, USA: Wiley Publishing, Inc., 2003.
- [4] S. Apel, C. Kästner, and C. Lengauer, “Research challenges in the tension between features and services,” Proceedings ICSE Workshop on Systems Development in SOA Environments (SDSOA). New York, NY, USA: ACM, 2008, pp. 53–58.
- [5] S. Trujillo, D. Batory, and O. Diaz, “Feature refactoring a multirepresentation program into a product line,” Proceedings of the 5th international conference on Generative programming and component engineering (GPCE). Portland, Oregon, USA: ACM, 2006, pp. 191– 200.

Imaging Services on the Grid as a Product Line : Requirements and Architecture

M. ACHER, Ph. COLLET, Ph. LAHIRE, J. Montagnat

Workshop SOAPL 2008

September 8th

Université Nice SOPHIA ANTIPOLIS

CNRS CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

i3S sophia antipolis

RAINBOW

Context : Services for the Grid

- ***Grid***

  sharing data, algorithms

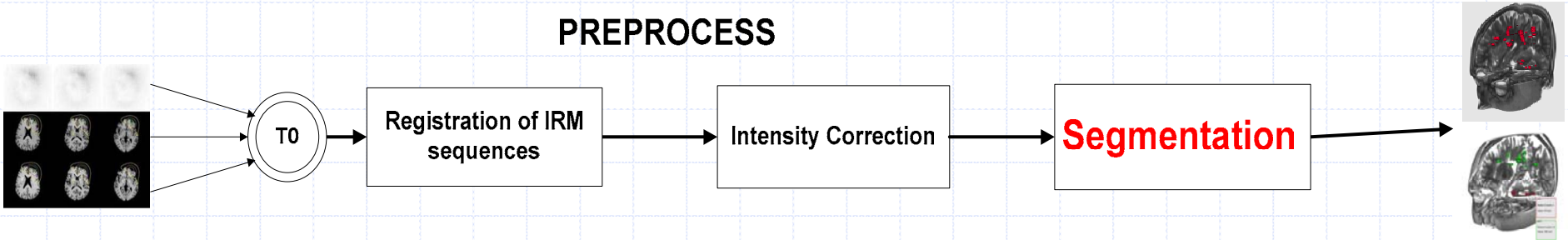
  computation power, data-intensive

- ***Workflows for the e-Science Grid***

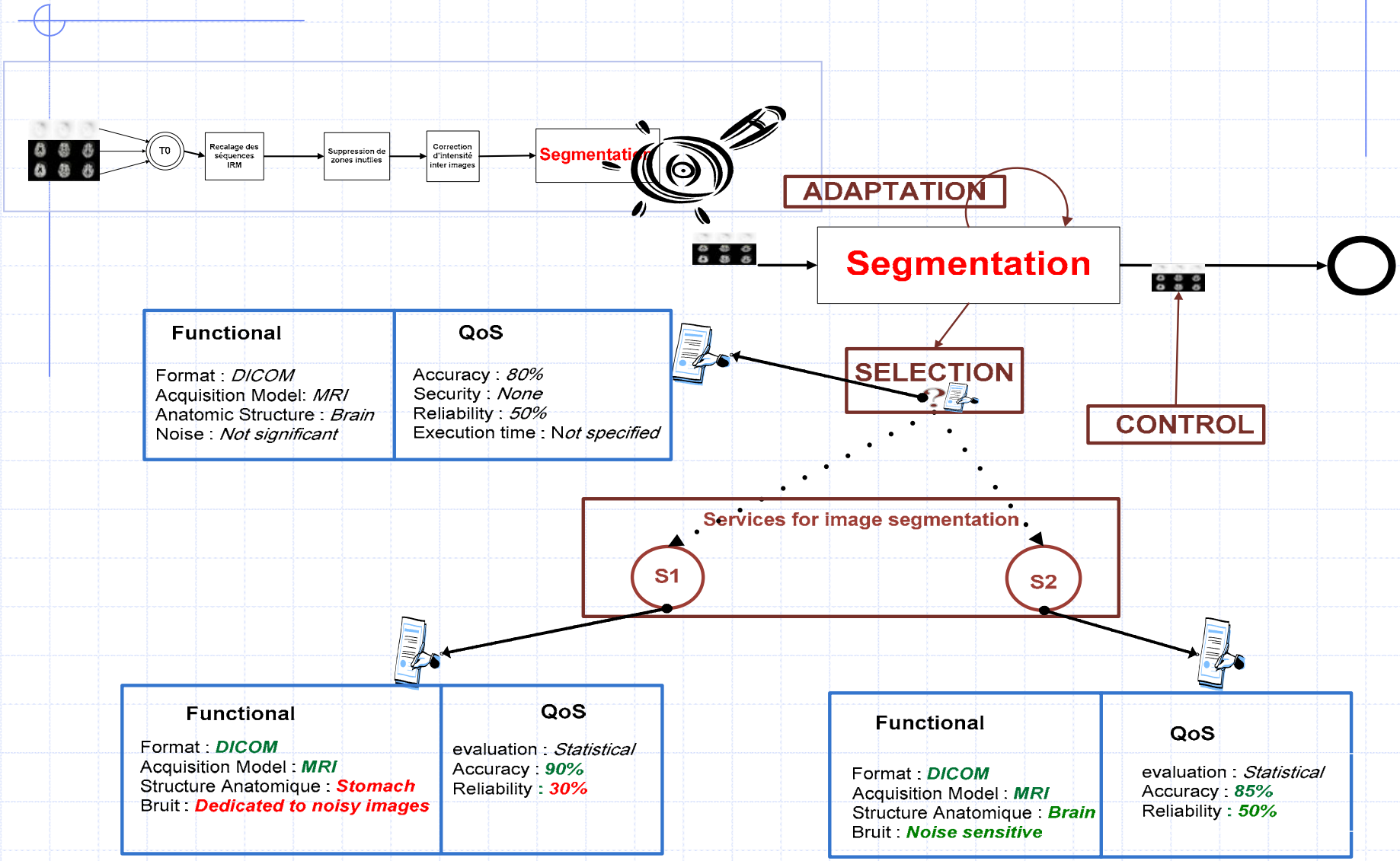
  process chain, pipeline, data flow

  reuse and compose (black) boxes

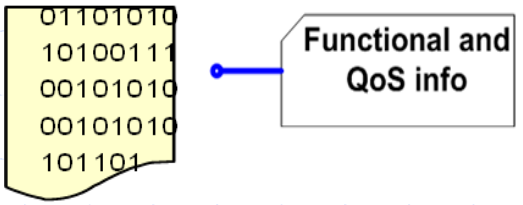
- ***Implemented as Services***



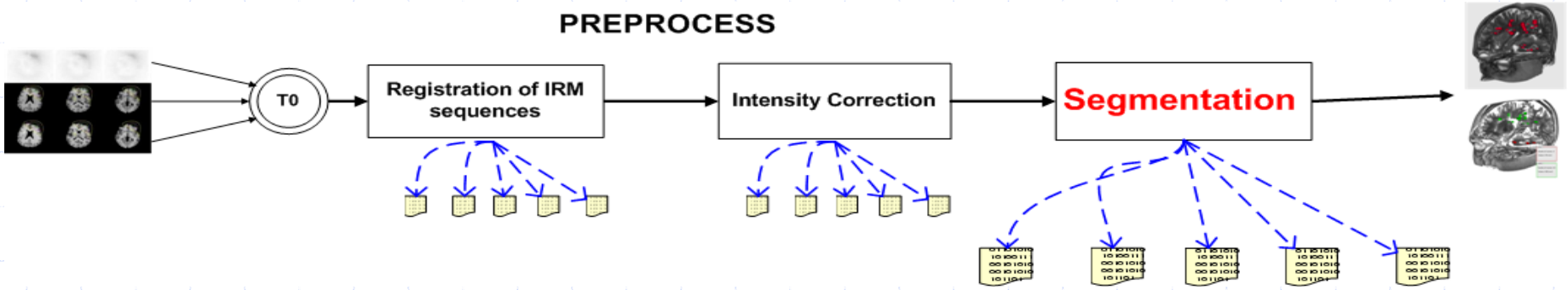
Requirements Overview



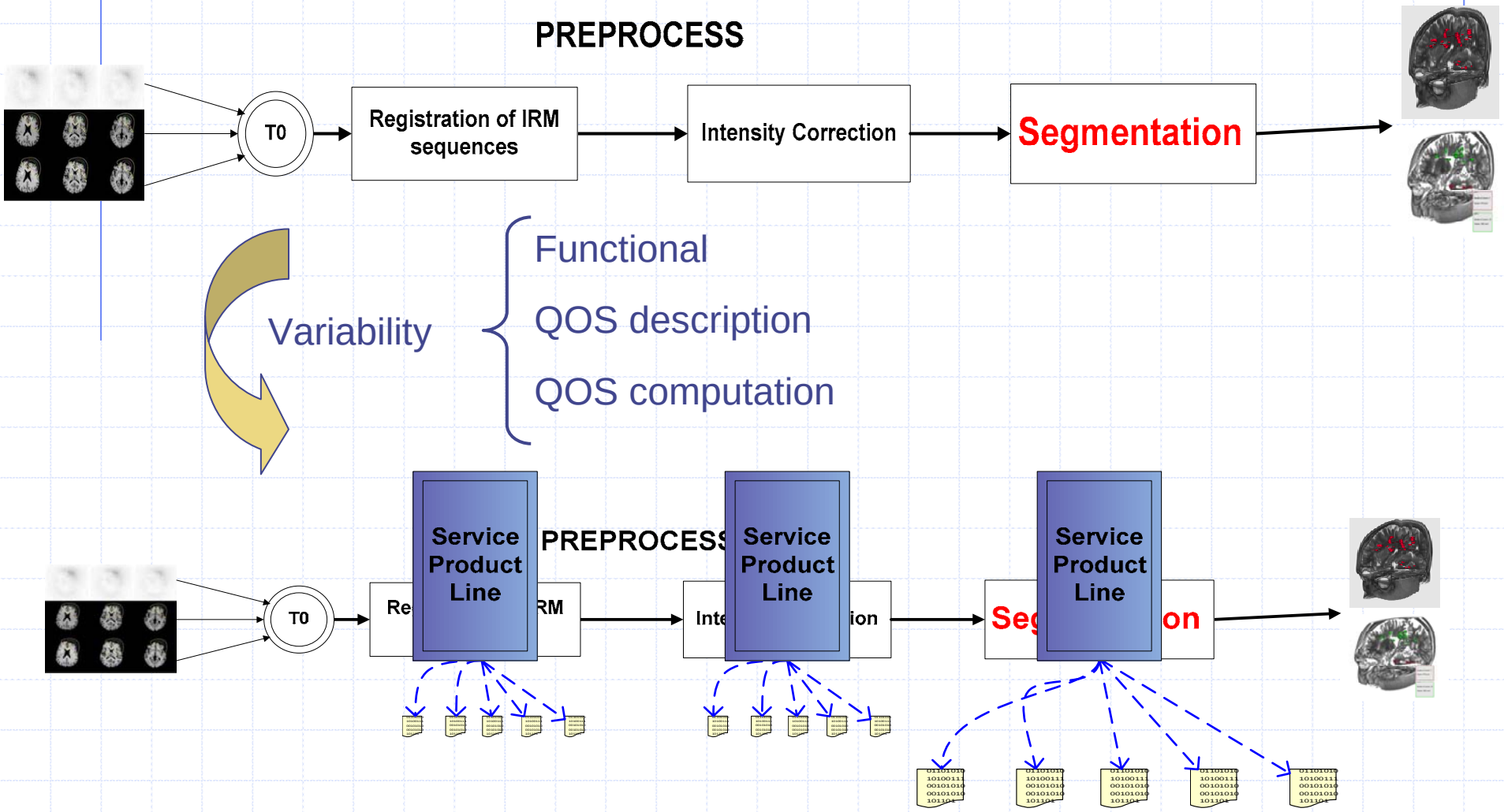
Composing Services on the Grid



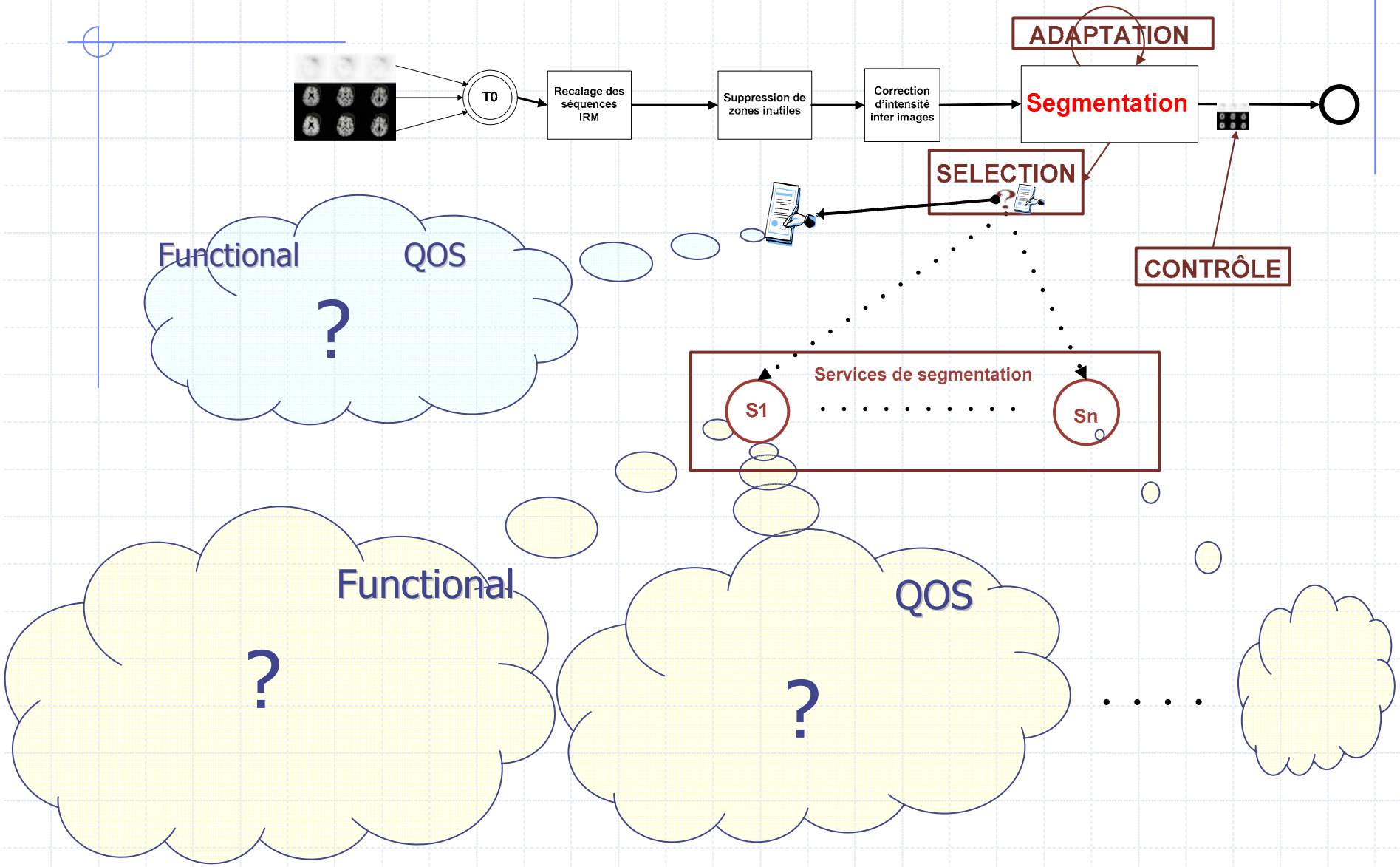
- How to deploy Grid Services ?
  - needs fine-grained information
- How to manage ***QoS*** (Quality of Service) ?
  - such as execution time, availability, reliability, etc. ?
- To give information to ...
  - worflow engine , software architect, scheduler
- Our position : a ***variability*** problem !



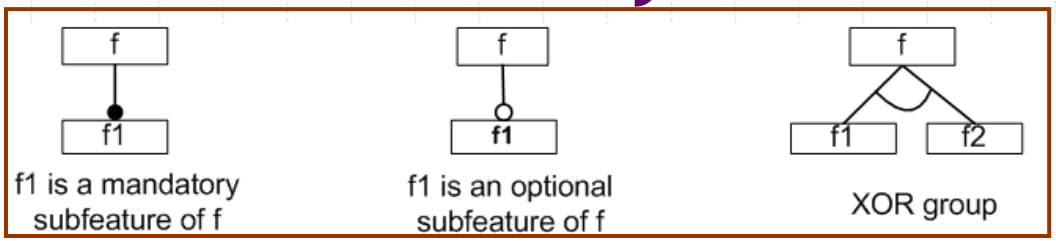
From Service to Product Line (1)



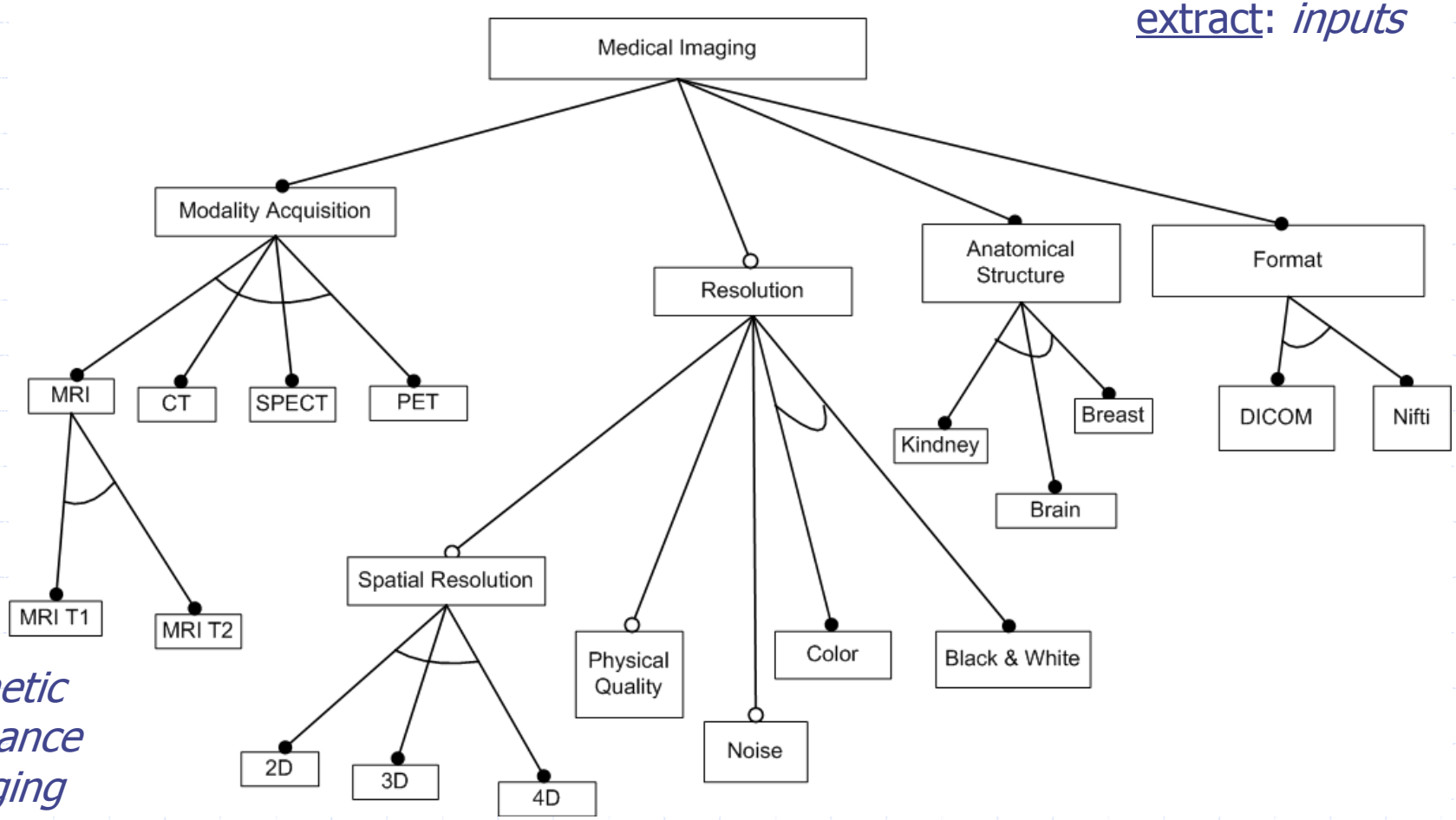
From Service to Product Line (2)



Functional Variability

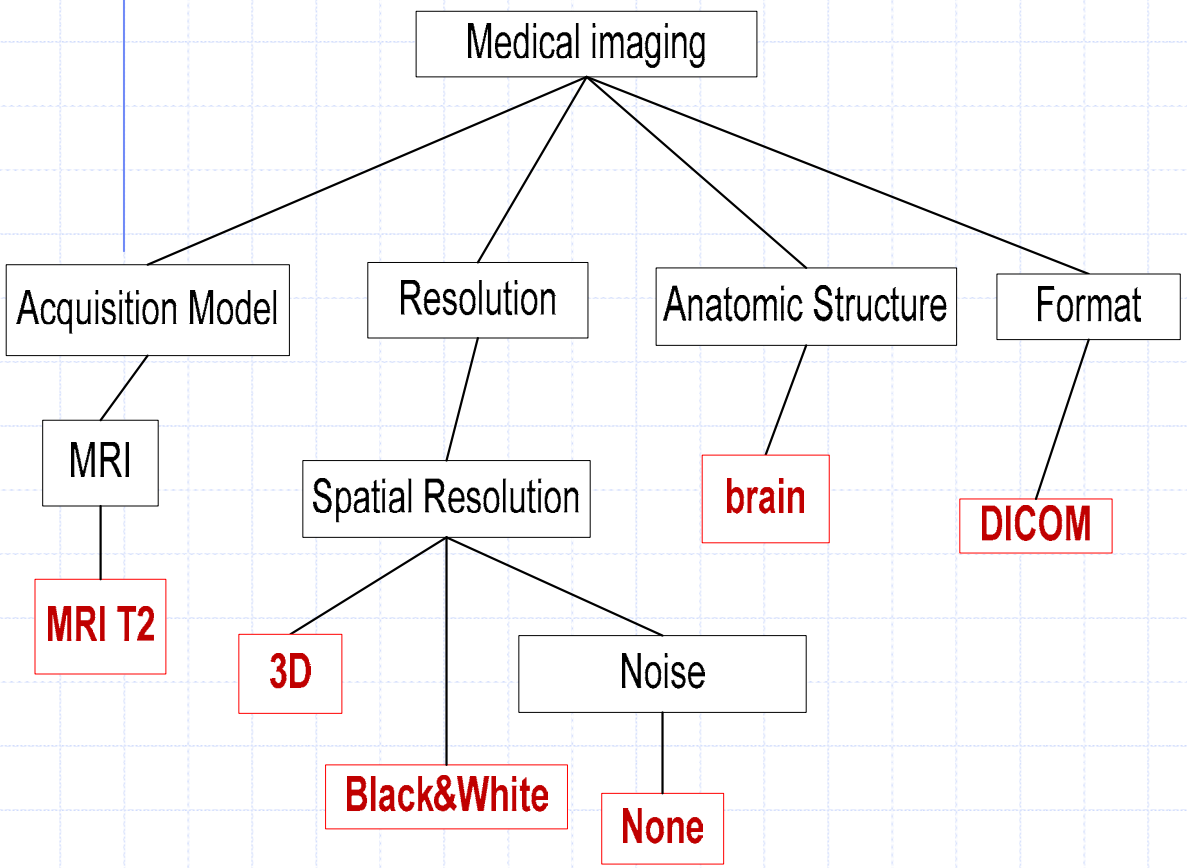


extract: *inputs*



Magnetic Resonance Imaging

Functional description : example



Acquisition Model

MRI = *MRI T2*

Resolution

Spatial Resolution

Dimension = *2D*

color = *B&W*

Noise = *none*

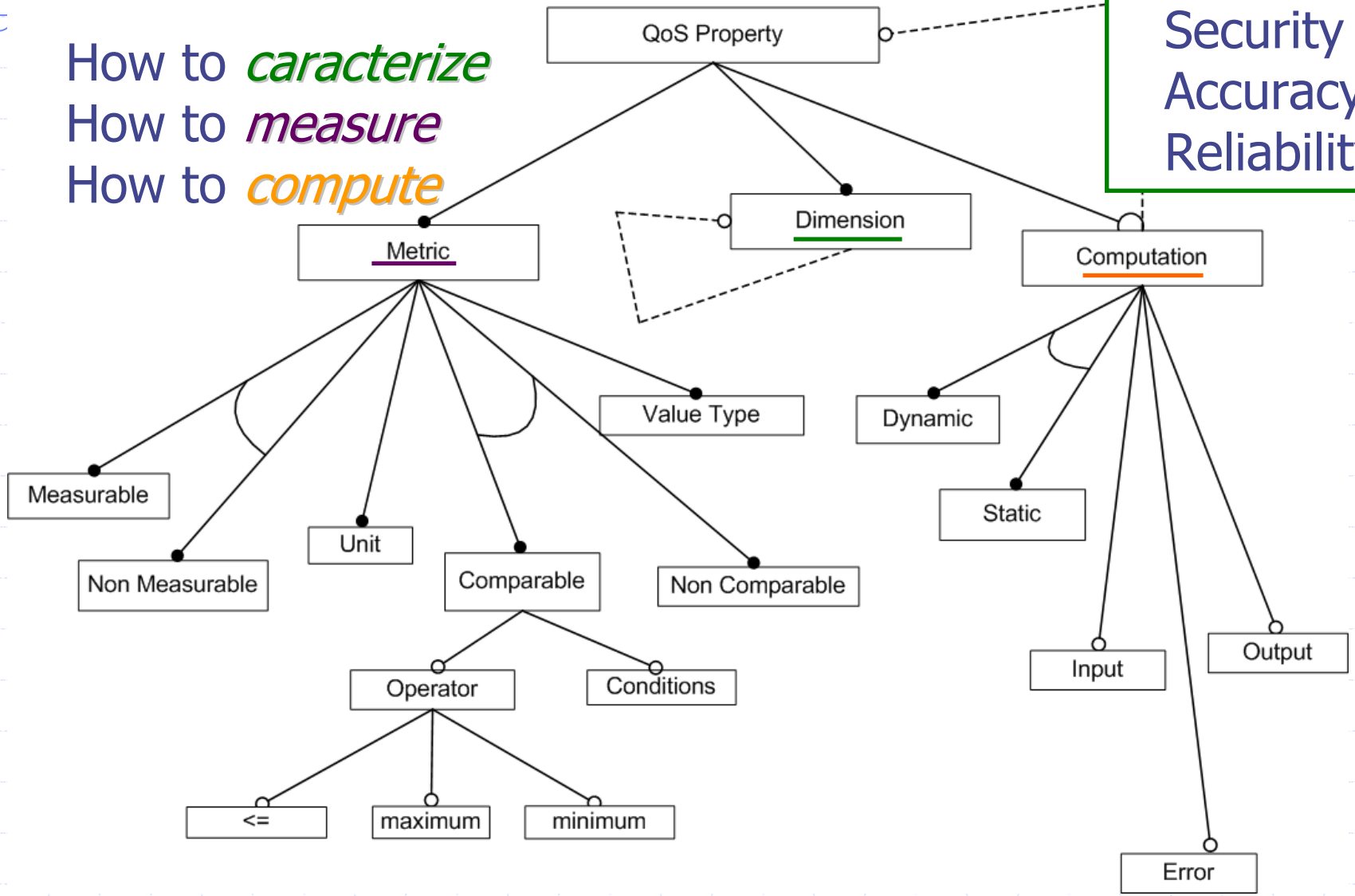
Anatomic Structure = brain

Format = DICOM

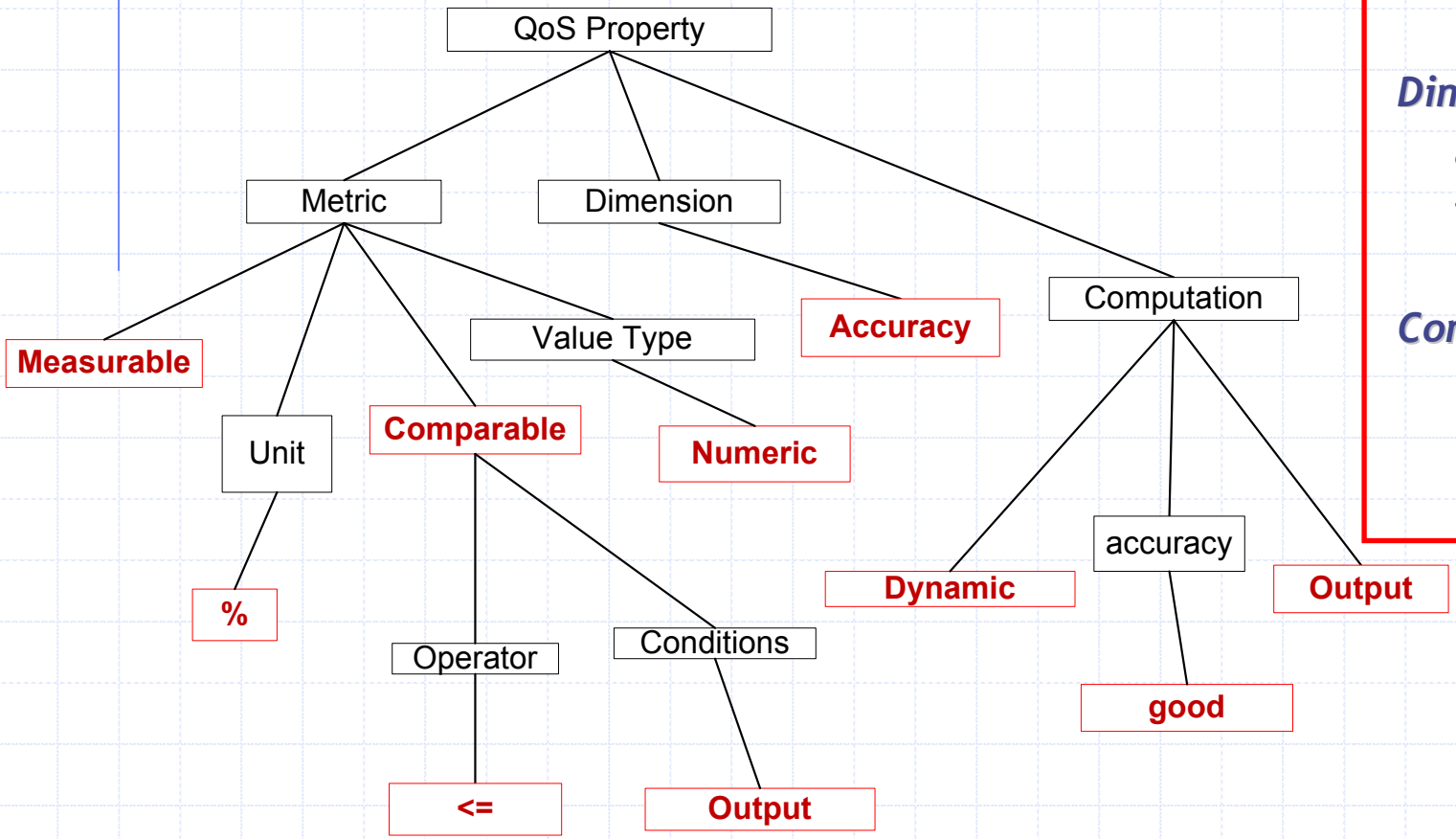
QOS Variability

How to *caracterize*
How to *measure*
How to *compute*

Time
Cost
Security
Accuracy
Reliability



QOS description : example



Metric

measurable = *true*

unit = *%*

comparable = *true*

type = *numeric*

Dimension

accuracy = *high*

time = *any*

…

Computation

dynamic = *true*

rely_on = *output*

accuracy = *good*

Segmentation: refining classification

QoS depends on application domain :

- goal of segmentation
- body region
- imaging protocol

"A particular segmentation may have *high performance* in determining the volume of a tumor in the brain on an MRI image,

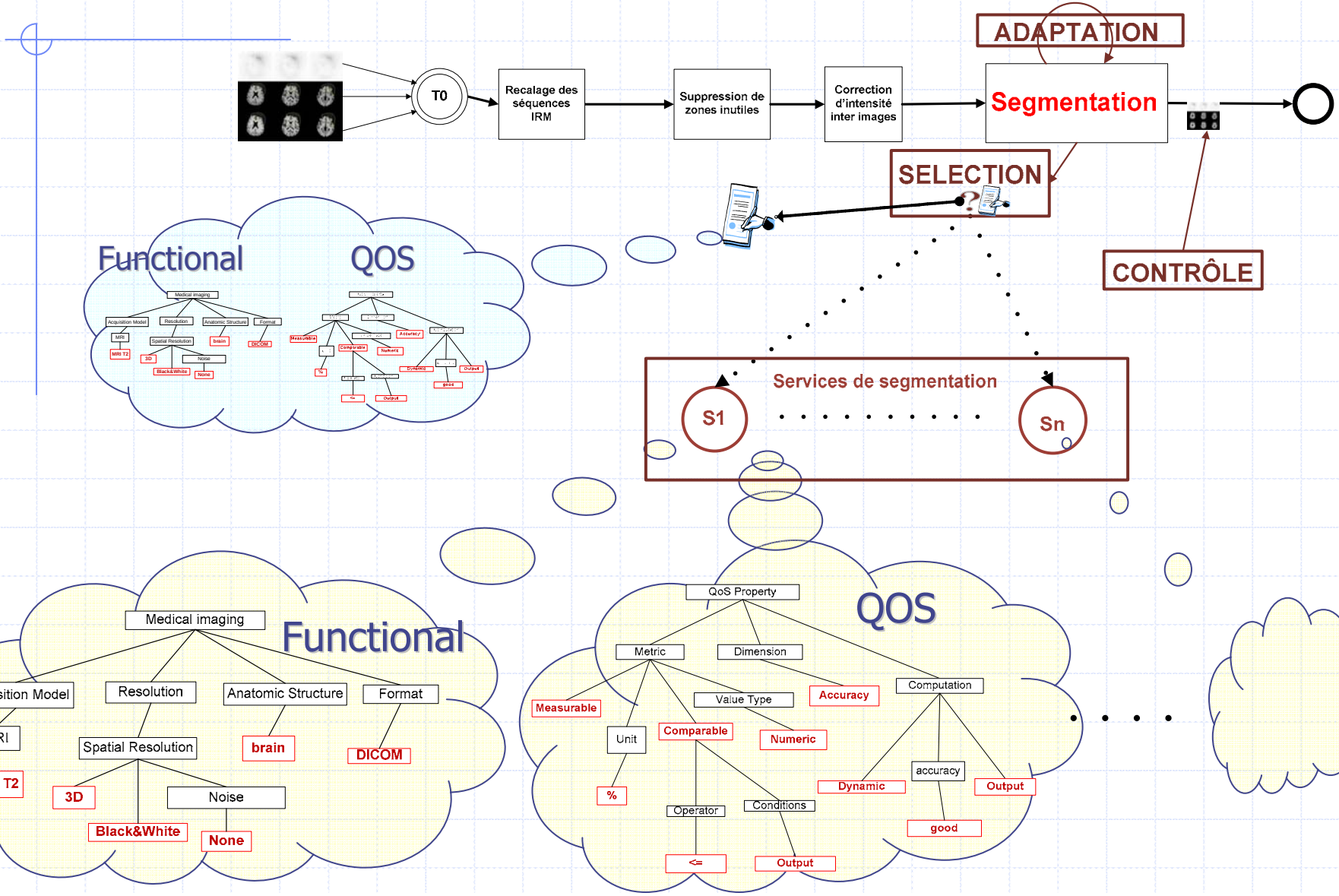
... but may have *low performance* in segmenting a cancerous mass from a mammography scan of a breast"

Dimensions : time and space complexity, accuracy, robustness, precision, specificity, sensibility

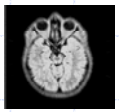
Interdependency between QOS and Computation of QoS :

- costly but precise
- quick but uncertain
- evaluation has a QoS too

Towards SPL: big picture



Towards Service product line



Service Product Line

segmentation service

ervice roduct Line

istration service

Behaviour + QOS

+ variability

Service Metamodel

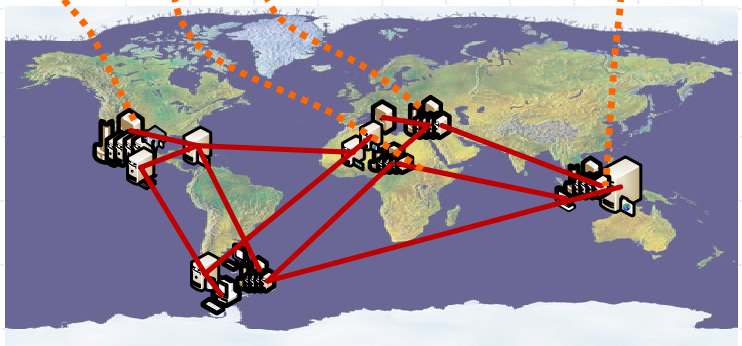
QoS Metamodel

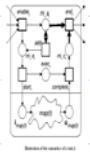
Grid Metamodel

Grid workflow expert

Service Repository

Medical imaging computation expert





Workflow

Service Product Line

One Workflow

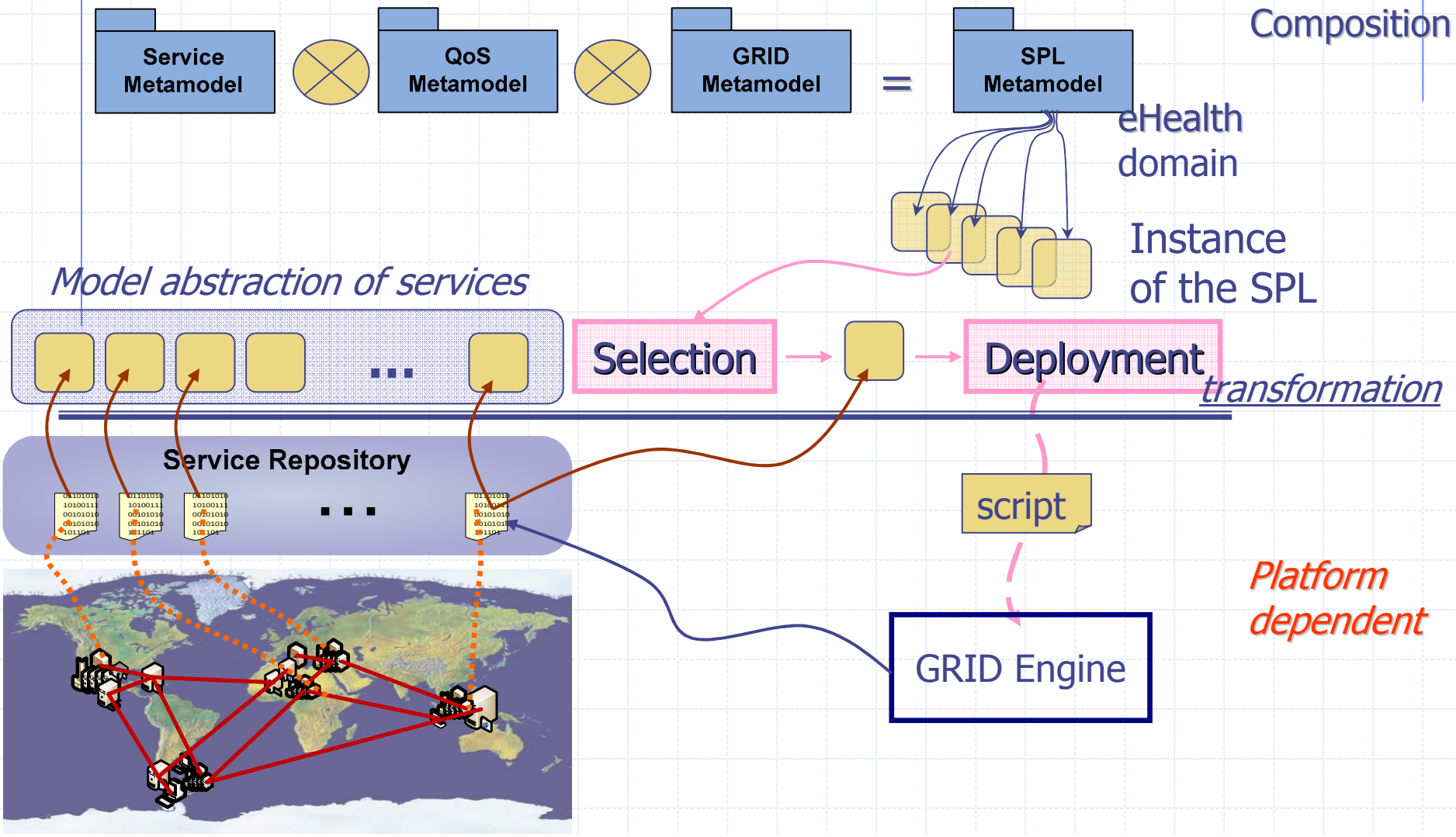
An MDE Approach

- Equipping Service/Workflow with meta information
  - A common core (QOS & service metamodels)
  - Specific branches
- Building the SPL
  - Describing a generic Domain-Specific service / workflow
  - Specifying composition protocol of one service
    - allow to address different workflow
    - includes also variability
- Approach
  - Model Driven Engineering (MDE)
  - Platform independent, abstraction
  - Model transformation and/or model composition

An MDE Approach

Model-Driven Engineering

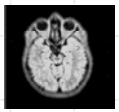
Workflow?
Service
Composition

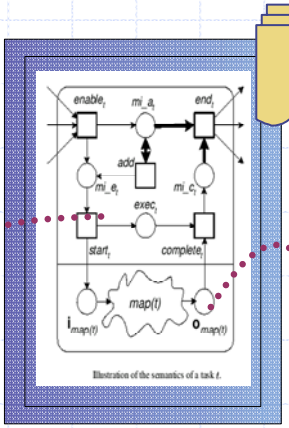


On-going Work

- QoS multi-views
  - experts collaboration
  - from end users to services
- How to infer a SPL ?
- Derivation process
  - who for the reasoning process ?
  - heuristics needed
- From Service to workflow

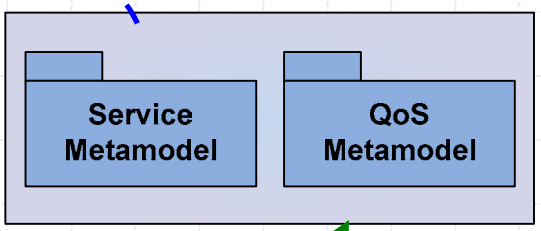
From Service to Workflow

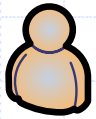




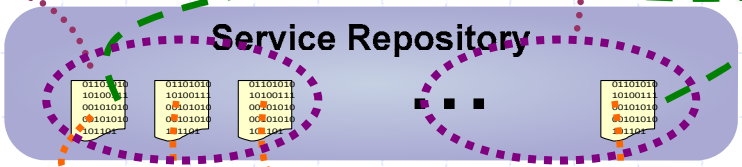
Behaviour + QOS

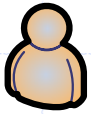
+ variability



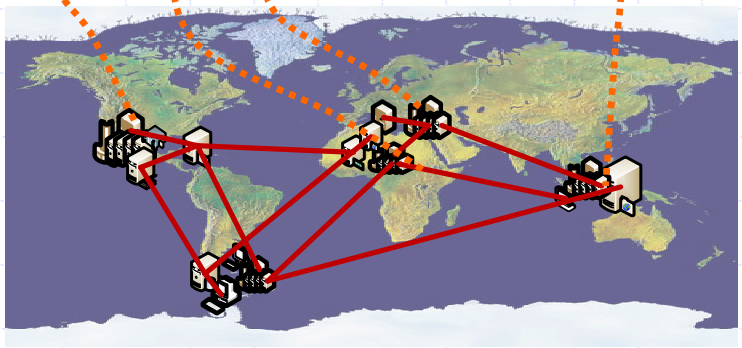


Grid workflow expert





Medical imaging computation expert



Questions ?

SOAPL 2008
Limerick, 8th September

Business Process Lines to develop Service-Oriented Architectures through the Software Product Lines paradigm

Nicola Boffoli, Danilo Caivano, Daniela Castelluccia,
Fabrizio Maria Maggi, Giuseppe Visaggio
SERLAB - Department of Informatics
University of Bari - Italy
{boffoli, caivano, castelluccia, maggi, visaggio}@di.uniba.it

Outline

SPL + SOA

- Why?
- What?
- How?

Our proposal

- Business Process Line
- Decision Models
- Case Study

SPL + SOA: *Why?*

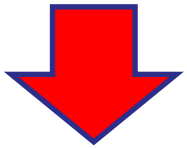
Two common perspectives

- Software reuse
  - implementing new software systems reusing existing software resources rather than developing the same software capabilities again
- Software flexibility
  - allowing to adapt the systems to the different customers of a whole market segment
    - ***SPL*** focuses on the commonality and variability to build a set of software products
    - ***SOA*** *allows to compose, orchestrate and maintain solutions based on services, implementing business processes*

SPL + SOA: *What?*

Our Proposal

- transferring peculiarities/advantages from SPL to SOA



- build a SOA systems line suitable to customers or market segments needs in a specific application domain

SPL + SOA: *How?*

We start from a deep analysis of the business processes identifying in them commonality and variability typical of the SPL paradigm

Business Process Line

+

Decision Models

Business Process Line (BPL)

A BPL realizes processes able to adapt themselves to different customer needs

- Each process of a BPL can be then transformed into the corresponding SOA system
  - ***If*** the business processes are adaptable to the customer needs
  - ***then*** the generated SOA system, it will result in its turn suitable to the specific customer requirements

From SPL to BPL: Analogies and Tailoring …

SPL

- Collection, organization and systematic refinement of the assets (invariant or variant)
- Automatic building of the products
  - Product Configuration: through asset integration procedures
  - Product Specialization: through the specification of the assets parametric part

… From SPL to BPL: Analogies and Tailoring

BPL

- Asset concept is referred to activities and work definitions
- Product Configuration Ł Process Configuration
  - *the assets (activities and work definitions) can be added to a basic business process in order to configure the target business process*
- Product Specialization Ł Process Specialization
  - *each asset of the target business process can be specialized through attributes indicating specific architectural characteristics to implement them*

BPL Decision Models

Hypothesis: two kind of relations

1. between the business capabilities (characterizing the customer needs) and the suitable processes elements (that have to be integrated in the target business process)
2. between the customer requirements and the specific peculiarities of the processes elements previously integrated in the target process.

Decision Table Formalism

A decision table (DT) is divided in four quadrants: conditions (Cond), conditional states (S), actions (Act) and rules (x)

The table is defined so that each combination of conditions and conditional states corresponds to a set of actions to carry out.

$Cond_1$	S_{11}				S_{12}			
$Cond_2$	S_{21}		S_{22}		S_{21}		S_{22}	
.....	...							
$Cond_N$	S_{N1}	S_{N2}	S_{N1}	S_{N2}	S_{N1}	S_{N2}	S_{N1}	S_{N2}
Act_1	-	-	-	-	X	X	X	X
Act_2	-	-	-	-	-	-	-	X
.....	...							
Act_M	-	X	-	X	-	X	-	

- *Compact overview*
- *Modular knowledge organization*
- *Evaluation of consistency, completeness and redundancy*

Configuring DT …

For each BPL a configuring DT is built in order to select the variant assets characteristic of the requested business capabilities

- They have to be composed with the invariant assets (integrated into a basic process) in order to generate the target business process

… Configuring DT

- the CONDITION quadrant contains a set of business capabilities, BC_i i=1,...n
- the CONDITIONAL STATE quadrant contains the possible values of each business capability $[BC_i]=\{bc_{i1}, bc_{i2}, \ldots, bc_{iq}\}$
- the ACTION quadrant contains all the possible variant assets $\{va_1, va_2, \ldots, va_r\}$ that can be added to the process commonality
- the RULE quadrant re
corresponding varian

BC_1	bc_{11}				bc_{12}			
BC_2	bc_{21}		bc_{22}		bc_{21}		bc_{22}	
…	.....							
BC_n	bc_{n1}	bc_{n2}	bc_{n1}	bc_{n2}	bc_{n1}	bc_{n2}	bc_{n1}	bc_{n2}
va_1	-	-	x	-	-	-	-	-
va_2	x	-	-	-	-	-	-	-
va_3	-	-	-	-	x	-	-	-
va_4	-	-	-	-	-	-	x	-
va_5	-	-	-	-	-	x	-	-
.....	.....							
va_r	-	-	-	-	-	-	-	x

DIB

Specializing DT

For each asset, variant or invariant, a specializing DT is built as follows

- the CONDITION quadrant contains a set of customer requirements, CR_j j=1,..,m, to specialize the parametric part of the asset
- the CONDITIONAL STATE quadrant contains the possible values of each requirement: $[CR_j]=\{cr_{j1}, cr_{j2}, \ldots, cr_{jt}\}$
- the ACTION quadrant contains the parameters $\{p_1, p_2, \ldots, p_s\}$ and their values allowing to specialize the parametric part of the asset
- the RULE quadrant set to the correspo

CR_1	fr_{11}				fr_{12}				fr_{13}			
CR_2	fr_{21}		fr_{22}		fr_{21}		fr_{22}		fr_{21}		fr_{22}	
...	.....											
CR_n	fr_{n1}	fr_{n2}	fr_{n1}	fr_{n2}	fr_{n1}	fr_{n2}	fr_{n1}	fr_{n2}	fr_{n1}	fr_{n2}	fr_{n1}	fr_{n2}
p_1 = "Y"	-	-	-	x	x	-	-	-	-	-	x	x
p_1 = "N"	x	x	x	-	-	x	x	-	x	x	-	-
.....	.....											
p_s = "x"	x	-	-	x	-	-	-	-	x	-	-	x
p_s = "y"	-	x	x	-	x	-	x	-	-	-	-	-
p_s = "z"	-	-	-	-	-	x	-	x	-	x	x	-

DIB

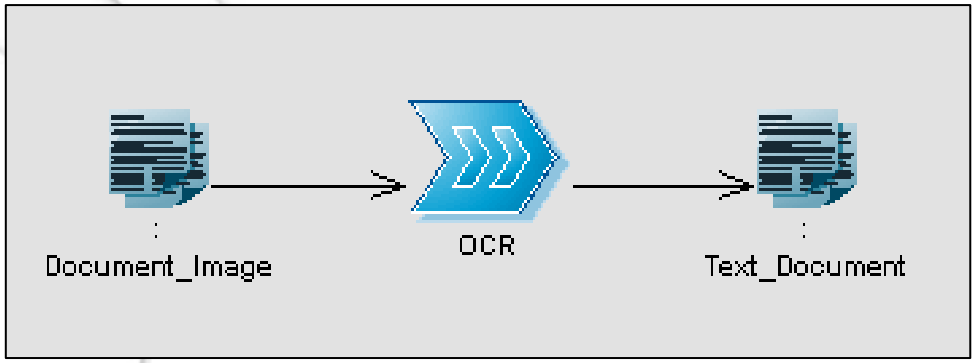
Case Study …

Our proposal has been investigated in an industrial case during the research project "DAMA" (Data Archiving Management and Acquisition)

- A specific part, Document Recognizing, is here summarized

Invariant Part

- the process contains an OCR (Optical Character Recognition) activity requiring a scanned Document Image as input and produces a recognized Text Document as output



… Case Study …

Configuring DT

- the table provides the following business capabilities: Signature Extraction, Layout Analysis and Image Extraction

Signed Documents	Digital				Autographic				Without Sign			
Document Type	Structured		Unstructured		Structured		Unstructured		Structured		Unstructured	
Documents with Images	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
Layout Analysis	X	X	-	-	X	X	-	-	X	X	-	-
Image Extraction	X	-	X	-	X	-	X	-	X	-	X	-
Autographic Sign Extraction	-	-	-	-	X	X	X	X	-	-	-	-
Digital Sign Extraction	X	X	X	X	-	-	-	-	-	-	-	-

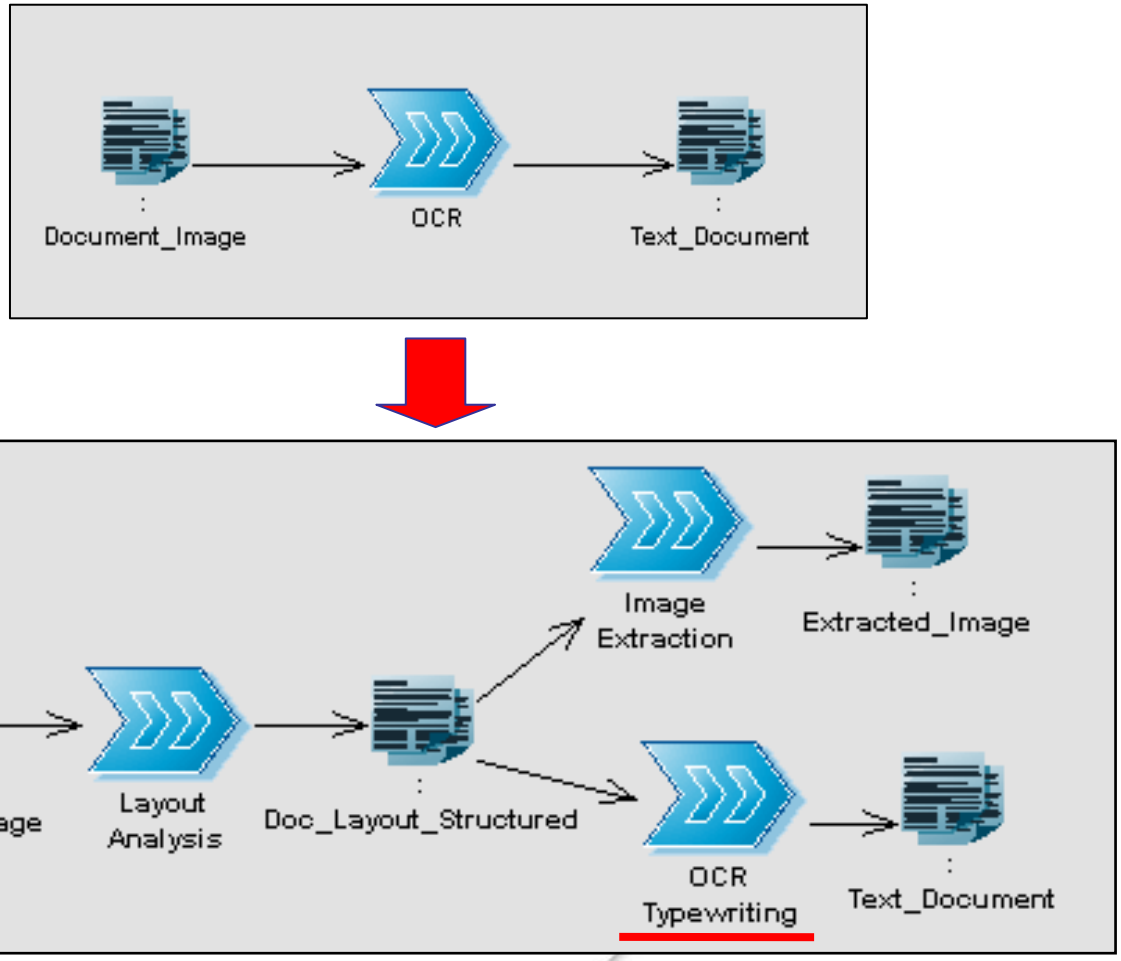
… Case Study …

Scenario

- "*The enterprise needs besides to elaborate and archive typewriting and structured documents, containing images and without signature*"

Signed Documents	Digital				Autographic				Without Sign			
Document Type	Structured		Unstructured		Structured		Unstructured		Structured		Unstructured	
Documents with Images	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
Layout Analysis	X	X	-	-	X	X	-	-	X	X	-	-
Image Extraction	X	-	X	-	X	-	X	-	X	-	X	-
Autographic Sign Extraction	-	-	-	-	X	X	X	X	-	-	-	-
Digital Sign Extraction	X	X	X	X	-	-	-	-	-	-	-	-

… Case Study


Document_Image
OCR
Text_Document
Image Extraction
Extracted_Image
Document_Image
Layout Analysis
Doc_Layout_Structured
OCR Typewriting
Text_Document

Conclusion …

This work proposes to apply the good practices of SPL to SOA, the authors introduce

- the concept of BPL in order to identify commonality and variability of SOA systems at the process level
- two kind decision models supporting BPL activities
  - Configuring Decision Model
  - Specializing Decision Model

The case study DAMA is ongoing and encourages further investigations in other applicative domains in order to confirm and generalize the preliminary results

… Conclusion

In order to support the application of the proposal here presented, the authors are developing two tools:

- the former aims to automate the decision tables management (design and consulting)
- the latter is able to transform business process models in executable workflows for SOA systems