INF5120  
Modellbasert systemutvikling

- **F15 – Oppsummering av kurset**
- **Pensum**
- **Eksamens-eksempler 2000-2006**
- **Eksamen 2007**

**Forelesning 14.05.2007**
Arne-Jørgen Berre
Arne.J.Berre@sintef.no

---

**INF5120 - Forelesninger - 2007**

AJB: Arne-Jørgen Berre, BRE: Brian Elvesæter, GKO: Gøran K. Olsen

- **MDSU** (Bok 1 – part 1 og 2) – fra 22/1
  1. 22-1: Introduction to MDSU, MDA, OO and modeling (M1-M5)
  2. 22-1: Metamodeling and UML profiles, MDA technologies (Eclipse technologies) (M2, M4, M6)
  3. 22-1: Language Engineering and DSL (M8)
  4. 22-1: Model transformations with ATL and other technologies (M10, M11)
  5. 22-1: Code generation with MOFScript and other technologies (M9)

- **COMET – New development** (Bok 1 part 3, Bok 2) – fra 26/2
  6. 26-2: Method Engineering and Process – and COMET overview (M13)
  8. 26-2: Requirements Engineering and Use cases – (and Agile methods and modeling), Patterns and aspects
  9. 26-2: SOA Architectures and platforms (SOA) and COMET Architecture Modeling and PIM4SOA and UPMS
  10. 26-2: PIM and Web Services teknologi (PSM) med WSDL/XML/BPEL

- **MDI** (Område 3) - fra 16/4
  11. 16/4: Interoperability and MDI – AIF, EIF and iEMS
  12. 23/4: Ontologies and Semantic web and Modeling
  13. 30/4: MDI – Model Driven Interoperability, PIM4SOA -> Agent transformations, AIF
  14. 7/5: Aspekt-orientert Programmering og Modellering
  15. 14/5  Course summary
F2: Outline

- Model-driven interoperability (MDI) framework
- MDA (revisited)
- Applied metamodelling
- Standards and technologies
  - OMG MOF and metamodelling
  - Eclipse Modeling Framework (EMF)
- Examples
  - Metamodel for PIM4SOA
  - Metamodel for Web services
  - Metamodel for BDI agents
- Eclipse Modeling Framework (EMF) exercise
- References

F3: Outline

- Metamodelling (revisited)
- UML profiles
- Domain-specific languages (DSLs)
- Eclipse technologies
  - Eclipse Modeling Framework (EMF)
  - Eclipse Graphical Modeling Framework (GMF)
- Example
  - Metamodels and UML profiles for SOA and Web services
- References
Model-to-Model Transformation idea

A PIM can be transformed into different PSM
Or transformed to a new PIM

PIM.. 1..* → MT

COM / DCOM  Java EJB  C#.net  XML SOAP  PIM2
Atlas Transformation Language (ATL)

- Will be used in INF 5120
- ATL is a hybrid language (a mix of declarative and imperative constructions) designed to express model transformations as described by the MDA approach.
- It is not QVT, but similar and with the corresponding functionality
- A transformation model in ATL is expressed as a set of transformation rules.
- The recommended style of programming is declarative.
- OCL is used to express constraints on rules
  - Guards (constraints) on the entry point for a rule

INF 5120 - Lecture #5
Code generation with MOFScript and other technologies

Gøran K. Olsen
SINTEF ICT
Cooperative and Trusted Systems
MOFScript placed in the 4-layer architecture

M3

conforms to

MOF

conforms to

M2

Source Metamodel

based on

MOFScript language

conforms to

M1

Source model

MOFScript transformation

executed by

MOFScript tool engine

output

Target text

INF5120
Lecture #6 (F6)
26 February 2007

- Method engineering and process
- COMET overview

Based on material developed in the ATHENA (IST-507849), COMBINE (IST-1999-20839), INTEROP (IST-508011), and MODELWARE (IST-511731) research projects.

Brian Elvesæter, SINTEF ICT
brian.elvesater@sintef.no
ATHENA Model-Driven Interoperability (MDI) Framework

MDA & Interoperability

Metamodelling

UML Profiles & DSLs

Model Transformations

Method Engineering

Reusable MDI Assets

- Method chunks
- Tools and services
- Models and metamodels
- Model transformations
- DSLs and UML profiles
- Reference examples

EPF Concepts to Create Process Frameworks

Process Framework

- Responsible for creating and modifying work products
- Input or output of performing roles
- Assigned to a role in a creation of modification of a work product
- Express process knowledge for a key area of interest
- Complete process template for a specific type of project
- Used to define processes, can relate to other activities to create work flows
EPF Composer

EPF Composer is a tool platform for process engineers, project leads, project and program managers who are responsible for maintaining and implementing processes for development organizations or individual projects.

Aims to:
- provide for development practitioners a knowledge base of intellectual capital that allows them to browse, manage and deploy content.
- provide process engineering capabilities by supporting process engineers and project managers in selecting, tailoring, and rapidly assembling processes for their concrete development process.

Tutorial:
Overview of the COMET Methodology

Developed by:
Brian Elvesæter, Arne-Jørgen Berre, Arnor Solberg
SINTEF ICT
COMET model architecture

Phases - organisation along time

Models
Business Model
Requirements Model
Architecture Model
Platform specific Model
Supporting activities
Project management
Work product management
Test

Iterations:
preliminary iteration(s) iter. #1 iter. #2 iter. #n iter. #n+1 iter. #n+2 iter. #m iter. #m+1

Review milestones:
Inception Review Prototype Launch Iteration Launch Technical Audit Product Launch Demonstrator Launch Iteration Launch Demo / Delivery Beta Test Launch Accept Meeting

COMET process architecture

ICT Resource Service Tier
Business Service Tier
User Service Tier
User Interface Tier

Concepts & Artifacts
Processes
Actors

Web Services model
- Web Services profile model
- Web Services implementation model

Service-Oriented Architecture model
- Component structure model
- Service interaction model
- Service interface model

Requirements model
- System boundary model
- Use case Scenario model
- Other requirements model

Business model
- Domain model
- Risk analysis
- Product vision & product desc.

Business domain

Technical domain

Real world

Model world

ICT

SINTEF
Outline

- Motivation – Why systems development methodology?
- Method engineering
- MDD Process Framework (MDD PF) and Software Process Engineering Metamodel (SPEM)
- Eclipse Process Framework (EPF)
- COMET methodology
- References
F7: Enterprise Architecture and Enterprise Modeling

- Enterprise Architecture
- Systems Architecture
- Methodologies (e5, p111-140, SEI/CMM, Zachman)
- Architectural Frameworks (C4ISR-AF, RM-ODP, MAF, TOGAF/DODAF, ...)
- Enterprise Unified Process
- Enterprise Modeling
- ATHENA POP*
- ArchiMate example
- Next: COMET Methodology

<table>
<thead>
<tr>
<th>Information</th>
<th>Process</th>
<th>PIM-K Services</th>
<th>Rules</th>
<th>NFA</th>
<th>UI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Wflow/Comp</td>
<td>PSM-K Interfaces</td>
<td>Rules</td>
<td>NFA</td>
<td>UI</td>
</tr>
<tr>
<td>XML, OWL,</td>
<td>BPEL/XPDL, OWL-S/WSML</td>
<td>WSDL, WSDL-S, SWRL, Induction, ...</td>
<td>Security, QoS</td>
<td>AJAX</td>
<td></td>
</tr>
</tbody>
</table>

Technologies/Realisation-K

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>BPMN, POP*, ARIS, ArchiMate, GERAM, GRAI, Zachman, UEMI, B.Rules, ..</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BPDM, SBVR, EDOC, UPMS, PIM4SOA, ODM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Legacy and New systems/services, ERPs/ESAs
F8: COMET Business Modelling

with WesternGeco Survey Booking

Reference Example
F9: INF5120
Modellbasert systemutvikling

- SOA architectures and platforms
- COMET architecture modelling
- PIM4SOA and UPMS

Forelesning 19.03.2007
Brian Elvesæter
brian.elvesater@sintef.no
F9: Outline

- SOA architecture and platforms
  - Service-oriented architecture (SOA)
  - SOA reference models
  - SOA platforms
- COMET architecture modelling
- PIM4SOA and UPMS
Reference architecture analysis

- Recall the Reference Architecture with associated tiers and component types.
- Analyse the System Boundary part of the Use Case Model with regards to the Reference Architecture.
  - The System Boundary correspond to the boundaries of the component of concern for the actual project, typically an Application Component.
- After the identification of this "main" component the next step is to analyze the System Boundary Model with respect to its set of use cases and actors.
- Divide the System Boundary Model into a set of subsystems.
  - Each subsystem will cover a collection of the System Boundary use cases.
  - The subsystems are non-exclusive, implying that a use case might be part of more than one subsystem.
  - The result of this task is a functional decomposition of the main component, with reiterated versions of the use cases.
Application component (AC) structure

Component model in RSM
PIM4SOA → platform specific models

PIM4SOA addresses four system aspects

**Metamodel for (software) services**
- Services are an abstraction and an encapsulation of the functionality provided by an autonomous entity. Service architectures are composed of functions provided by a system or a set of systems to achieve a shared goal.
  - Web Services Architecture as proposed by W3C (W3C 2004)
  - UML Profile for Enterprise Distributed Object Computing (OMG 2002)

**Metamodel for (automated software) processes**
- Processes describe sequencing of work in terms of actions, control flows, information flows, interactions, protocols, etc.
  - Business Process Definition Metamodel (BPDM) (IBM et al. 2004)
  - UML Profile for Enterprise Distributed Object Computing (OMG 2002)

**Metamodel for information**
- Information is related to the messages or structures exchanged, processed and stored by software systems or software components.
  - Structural constructs for class modelling in UML 2.0 (OMG 2003)
  - UML Profile for Enterprise Distributed Object Computing (OMG 2002)

**Metamodel for quality of service (QoS)**
- Extra-functional qualities that can be applied to services, information and processes.
  - UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms (OMG 2004)
Outline

- Web Services – Infrastructure and Architecture
- XML Introduction
- XML Schema
- Web Service Description Language (WSDL)
- Business Process Execution Language (BPEL)

XSD: UML profile for XSD

- UML representation of XML schema.
- Useful in a UML-centric development method if the modelling environment supports generation/import of XSD documents.
UML profile for WSDL

UML representation

Text representation

UML profile for BPEL

Customer | InvoiceProvider | ShippingProvider | SchedulingProvider
---|---|---|---

OwnedBehaviorActivity1

<receive> receivePO

<invoke> InitiatePriceCalculation

<assign> InitialiseShippingRequest

<invoke> requestScheduling

<invoke> sendShippingPrice

<invoke> requestShipping

<receive> receiveInvoicePrice

<reply> returnInvoice

<invoke> sendSchedule
### PIM4SOA main mappings to BPEL

<table>
<thead>
<tr>
<th>PIM4SOA element</th>
<th>BPEL equivalent</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServiceProvider Process</td>
<td>Process</td>
<td></td>
</tr>
<tr>
<td>Task (i) (participating in collaboration)</td>
<td>Receive, Invoke, Reply</td>
<td>The type of communication with other service providers must be deduced from parameters passed to or from the task in question.</td>
</tr>
<tr>
<td>Task (ii) (no collaboration use)</td>
<td>Empty</td>
<td>This might be a task requiring further implementation or human interaction beyond the scope of the PIM4SOA.</td>
</tr>
<tr>
<td>Flow</td>
<td>Sequence, Flow, {...}</td>
<td>The structure of flows between Steps must be analysed to deduce the applicable BPEL structure.</td>
</tr>
<tr>
<td>Interaction, Pin</td>
<td>Assign</td>
<td>Interactions have a role to play both in determining collaboration type (see Task (ii) above), and passing particular parts of messages between tasks (data flow, a BPEL assign).</td>
</tr>
<tr>
<td>Message</td>
<td>Variable</td>
<td>All messages sent and received must have appropriate variables defined within the BPEL.</td>
</tr>
<tr>
<td>CollaborationUse RoleBinding</td>
<td>PartnerLink</td>
<td>The CollaborationUses defined for the ServiceProvider tell us what PartnerLinks we will need. See CollaborationUsePath.</td>
</tr>
<tr>
<td>CollaborationUsePath</td>
<td>PartnerLink, Role(…)</td>
<td>This defines a specific use of a PartnerLink, alongside what role we are playing, and even the PortType being used. See links to the WSDL transformation described below.</td>
</tr>
</tbody>
</table>

---

### F11: INF5120

**Modellbasert systemutvikling**

- **Interoperability**
- **ATHENA Interoperability Framework (AIF)**
- **European Interoperability Framework (EIF)**

**Forelesning 16.04.2007**
Brian Elvesæter
brian.elvesater@sintef.no
Outline

- Interoperability
  - Interoperability scenarios
- ATHENA Interoperability Framework (AIF)
  - ATHENA Interoperability Profiles (AIPs)
  - Application of the AIF
  - Evaluation, conclusion and future work
- European Interoperability Framework (EIF)

ATHENA Interoperability Reference Architecture

<table>
<thead>
<tr>
<th>Provided</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise/Business</td>
<td>Enterprise/Business</td>
</tr>
<tr>
<td>Processes</td>
<td>Processes</td>
</tr>
<tr>
<td>Services</td>
<td>Services</td>
</tr>
<tr>
<td>Information/Data</td>
<td>Information/Data</td>
</tr>
<tr>
<td>Model-Driven Interoperability</td>
<td>Semantics and Ontologies</td>
</tr>
<tr>
<td>Collaborative Enterprise Modelling</td>
<td></td>
</tr>
<tr>
<td>Cross-Organisational Business Processes</td>
<td></td>
</tr>
<tr>
<td>Flexible Execution and Composition of Services</td>
<td></td>
</tr>
<tr>
<td>Information Interoperability</td>
<td></td>
</tr>
</tbody>
</table>
Interoperability Framework

Baseline Methodology

Methodology Overview (V-model view)
F12: OWL Classes

The Tree of Knowledge Technologies
(Utvidet med basis fra Top Quadrant)
UML Ontology profile

Current MDA Architecture with Semantic annotations

Enterprise modeling expert

System modeling expert

System realisation installation expert

CIM/EM models

PIM System models

PSM System models

System
Current MDA Interoperability Architecture

Semantiske Teknologier

- Ontologier
- RDF og OWL
- Topic Maps, ISO 15926, UML og andre

2 papers:
- Semaphore model mapping (Semaphore paper)
- Sammenlikning av model mapping og bruk av ontologier (Semantic mapping: ontology-based vs. model-based approach Alternative or complementary approaches?)
Semantisk Interoperabilitet gjennom referanse til felles ontologier

Service-orientert IT system

Virksomhet

Modellorientering

Forretningsorientering

Informasjon

Tjeneste

Prosess

Semantisk orientering

Brukerorientering

Service orientert IT

Ontologier

Forretningsorientering

ICT

SINTEF

Annotation

Reference Ontology

SwApp#1

SwApp#2

Sem Annot Set #1

Sem Annot Set #2

Sem Rec Rules #1

Sem Rec Rules #2

Local Software & Data

Design-time

Run-time

Internet
Step 3 – Create mediator/transformation

SAWSDL - Semantic Annotations for WSDL and XML Schema

- W3C Working Draft 10 April 2007

- This specification defines a set of extension attributes for the Web Services Description Language and XML Schema definition language that allows description of additional semantics of WSDL components. The specification defines how such semantic annotation is accomplished using references to semantic models, e.g. ontologies

- 3 constructs: modelReference, liftingSchemaMapping, loweringSchemaMapping
A Web Service Composition Scenario with Ontology Reasoning

A Possible Web service composition that fulfills the above request

Semantic Annotations for WSDL and XML Schema

- W3C Working Draft 10 April 2007

This specification defines a set of extension attributes for the Web Services Description Language and XML Schema definition language that allows description of additional semantics of WSDL components. The specification defines how such semantic annotation is accomplished using references to semantic models, e.g. ontologies. SAWSDL does not specify a language for representing the semantic models. Instead it provides mechanisms by which concepts from the semantic models, typically defined outside the WSDL document, can be referenced from within WSDL and XML Schema components using annotations.
F13: Model-driven Development of Agent Systems:

A Platform-Independent Model for Agents to improve Interoperability between Platforms

Christian Hahn
Multiagent System Group
German Research Center for Artificial Intelligence (DFKI GmbH), Saarbrücken

Guest researcher, SINTEF, Oslo, as part of the INTEROP project
Platform-Independent Model for Agents (PIM4Agents)

**Agent** is the autonomous entity capable of acting in the environment

**Cooperation** between agents

**Role**: domain-specific & application-specific roles

**Message** exchange is described in protocols

**Behaviour** describes the internal processes

**Protocol** describes interaction between agents/organisations

---

**F14**

**AOSD - Enhancing SoC**

AOSD :: Aspect Oriented Software Development
SoC :: Separation of Concerns
Agenda

1. SoC mechanisms in software development
   a. SoC – a definition
   b. OO technology / Object “thinking”
   c. Enhancements to OO-technology
   d. Agents and Services
   e. Model-centric “thinking”

2. AOSD – Aspect-Oriented Software Development
   a. Motivation
   b. Introduction
   c. Aspects
   d. Benefits

3. AOP - Example
   a. AspectU
   
   ■ AOM – Aspect Oriented Modeling
     a. AOM in general
     b. AOMDF – Overview
     c. AOMDF - Interactions
     d. The Theme Approach
   
   ■ Summarizing thoughts
     ■ Summarizing thoughts
     ■ Future
   
   ■ References

4a. AOM in general
   
   ■ Fusing AO-concepts and Model Driven Engineering
     ■ Leveraging the strengths of two great advancements in software development
   
   ■ The Theme Approach
     ■ Theme/Doc (analysis)
     ■ Theme/UML (design)
     ■ Partial tool support
   
   ■ AOMDF (Aspect Oriented Model Driven Framework)
     ■ Structural and behavioural AOM, extends UML
     ■ Automated model compositions of Class diagrams and Sequence diagrams
     ■ Tools support in near future
     ■ Design only
   
   ■ Many other approaches, but the one’s mentioned above are the most mature / complete. The Theme approach is the most known of them all.
4a. AOM in general (2)

- Reference Architecture for AOM:

Eksamen

- Case-basert oppgave (ref. tidligere eksamen), men der mer av de initielle modellene/metamodeller nå blir gitt i oppgaven (pga begrenset tid)

- Alle skriftlige hjelpemidler er tillatt

- Fredag 1. juni kl. 14:30 (3 timer)
Pensum litteratur

- F – Foiler fra alle forelesningene, F1 – F15

- Artikler: ”MDI - Model Driven Interoperability”,
  - ATHENA Interoperability Framework
  - Semaphore
  - Model Driven Ontology
  - Aspects, AOM

- Bok: **Model-Driven Software Development: Technology, Engineering, Management (Paperback)**
  by Thomas Stahl, Markus Voelter, Krzysztof Czarnecki


Model-Driven Software Development

- Part I. – Introduction (1-5)
- Part II – Domain architectures (6-12)
- Part III – Processes and Engineering (13-17)
- Part IV – Management (18-20)

1. Introduction
2. MDSD – Basic ideas and terminology
3. Case-study: A typical Web application
4. Concept Formation
5. Classification
6. Metamodeling
7. MDSD-capable target architectures.
8. Building Domain Architectures
9. Code Generation Techniques
10. Model transformations with QVT
Model-Driven Software Development

11. MDSD Tools: Roles, architecture, selection criteria, and pointers
12. The MDA standard
13. MDSD Process building blocks and best practices
14. Testing
15. Versioning
16. Case study: Embedded component infrastructures
17. Case study: An Enterprise system
18. Decision support
19. Organisational aspects
20. Adoption strategies for MDSD
   A. Model transformation code: QVT relations/mappings