INF5120
”Modellbasert Systemutvikling”
”Modelbased System development”

Lecture 6: 18.02.2013
Arne-Jørgen Berre

arneb@ifi.uio.no or Arne.J.Berre@sintef.no
Content

- Oblig 1 – details
- Business-SoaML, Goal Modeling, BMM, and Non Functional requirements
- Requirements Modeling
- BMM – Business Motivation Model
- Goal Modeling - KAOS
- Gilb - PLanguage
- Non functional requirements, ISO etc.
- …
1 (14/1): Introduction – overview Enterprise Architecture with UML and BPMN and DSLs
2 (21/1): Service Innovation and Design, AT ONE method/workshop – myServiceFellow (Marika Lüders)
3: (28/1): Value Networks/VDML BPMN, vs. UML Activity diagrams - Oryx
5 (11/2): UML and Req.Modeling – Agile User stories versus Use cases
6 (18/2): Business-SoaML, Requirements Modeling, Goal Modeling, BMM, and Non Functional requirements

7 (25/2): Model driven engineering – Metamodels, DSL, UML Profiles etc., more on NFR
8 (4/3): Model driven engineering EMF, Eclipse, GMF
9 (11/3): Model driven engineering, transformation technologies (Franck Fleurey)

10(18/3): UML Service Modeling – SoaML, UML 2.0 Service composition, USDL, ISO 19119
12(15/4): UML and Entity and Information modeling, UML, ISO 19103
13(22/4): UML and Semantic models, Facts, SBVR, Ontologies, Rules
14(29/4): UML and Platform models, realisation, migration, Java, Apps, CloudML
16(13/5): Conclusion and Summary for INF5120 - Preparation for Exam

Exam: Monday June 3rd, 2013, (4 hours)
INF5120 – Oblig plan - 2013

- 1 (14/1): Introduction
- 2 (21/1): myServiceFellow
- 3: (28/1): Oryx
- 4 (4/2): Balsamiq
- 5 (11/2): Use cases 2.0
- 6 (18/2): Oblig 1 – Group work

- 7 (25/2): EMF and Eclipse – Group presentation – Business Model
- 8 (4/3): EMF and GMF – Group presentation

- 10(18/3): Walk through of Oblig 1
- 12(15/4): Walk through of Oblig 2
- 13(22/4): Group work, Oblig 3 – Group presentation
- 14(29/4): Group work, Oblig 3 – Group presentation
- 16(13/5): Walk trough of Oblig 3 - Preparation for Exam

- Exam: Monday June 3rd, 2013, (4 hours)
Obligs

- Partially individual, partially group - in 3 parts

- Oblig 1 – Group “Personal Concierge” - Business architecture and requirement models (March 11)

- Oblig 2 – Individual – Eclipse editor for “Value networks” (April 8)

- Oblig 3 – Group “Personal Concierge” – System architecture (May 6)
Student Groups

Group 1
- davletd@ifi.uio.no
- ulrikeja@student.matnat.uio.no
- Mathiare@ifi.uio.no
- bushran@ifi.uio.no

Group 2
- Karollu@ifi.uio.no
- jesperov@ifi.uio.no
- pcscheel@gmail.com

Group 3
- zahramaslavi@gmail.com
- guangyuh@ifi.uio.no
- yannickl@ifi.uio.no
- eunji.lee@sintef.no

Group 4
- animutdt@ifi.uio.no
- shakar70@gmail.com
- henrikvs@ifi.uio.no
- suthannaa@ifi.uio.no

Group 5
- aakamran@gmail.com
- mlnguyen@ifi.uio.no
Strategyzer (Osterwalder)
ServiceML Editor

- Web-based modelling editor
  - [http://tomcat.thingml.org/backend/poem/repository](http://tomcat.thingml.org/backend/poem/repository)

- User guide

- Currently being extended to support AT ONE Method (i.e., the Service Innovation practice)
Manifesto for Agile Software Development

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

http://agilemanifesto.org/
Specifying Services using the Service oriented architecture Modeling Language (SoaML): A baseline for Specification of Cloud-based Services

Brian Elvesæter¹, Arne-Jørgen Berre¹ and Andrey Sadovykh²

¹ SINTEF ICT, P. O. Box 124 Blindern, N-0314 Oslo, Norway
  brian.elvesater@sintef.no, arne.j.berre@sintef.no
² SOFTEAM, 21 avenue Victor Hugo, 75016 Paris, France
  andrey.sadovykh@softeam.fr
Outline

- Short presentation of the main part of the paper
  - What is SoaML?
  - Experiences and issues
  - Illustrative example
  - Two approaches

- Ongoing and future work
  - A baseline for Specification of Cloud-based Services
  - Towards a Cloud Modeling Language (CloudML)
What is SoaML?

- Service oriented architecture Modeling Language (SoaML)
  - Defines language constructs and extensions to UML2 to support service concepts (metamodel and UML profile)
  - Focuses on basic service modelling concepts and structure.
  - A foundation for further extensions and integration with BPMN, BMM and other metamodels.

- Key language constructs
  - Consumer
  - MessageType
  - Participant
  - Provider
  - ServiceContract
  - ServiceInterface
  - ServicesArchitecture
SoaML experiences

- Our experiences with SoaML
  - Tooling
  - Methods and practices
  - Application in industry projects

- Identified issues
  - Inconsistencies in the specification.
  - Two (three) main approaches to service modelling.
  - Examples illustrating the two approaches are not consistent.
  - No clear separation, the two approaches are somewhat intertwined.
  - Tool support lacking or worse wrongly implemented.

- Purpose of our paper
  - Clarify the differences and similarities between the different approaches.
  - Describe how to align the approaches.
  - Position SoaML as a baseline for specification of cloud-based services.
  - SoaML can be extended with new modelling constructs and integrated with other modelling languages.
Consolidated and extended example

Participant type (role type)

CollaborationUse

RoleBinding

Services architecture:
- High level description of how participants work together for a purpose by providing and using services expressed as service contracts.

Service contract:
- Service specifications that define the roles each participant plays in the service and the interfaces they implement to play that role.
Two approaches
Business-SoaML: Service Identification and Specification from a Business Perspective

T. Chang\textsuperscript{1-3}, A.-J. Berre\textsuperscript{4}, C. Carrez\textsuperscript{4} and B. Elvesæter\textsuperscript{4}

\textsuperscript{1} University of Bordeaux I, France \quad \textsuperscript{2} Harbin Institute of Technology, China
\textsuperscript{3} University of Stavanger, Norway \quad \textsuperscript{4} SINTEF ICT, Norway

I-ESA’12 – March 22\textsuperscript{nd}-23\textsuperscript{rd} – Valencia, Spain
Presentation outline

- Business-SoaML Motivation
- Vision, problem and solution
- Context: NEFFICS project
- VDML and SoaML
- Filling the gap between business and IT
- Business-SoaML as a simplified SoaML for business people
- Conclusion & Future Work
Motivation: Lack of understanding between Business and IT

- Business models:
  - Strategy planning
  - Identify sources of revenue
  - Identify new business opportunities
  - ...

- Business people don’t understand implementation models & code
  - How it should be done?

- Realisation by IT:
  - Detailed models, complex code

- Technical IT people don’t understand business
  - What we did is what was planned?
  - What we did generate revenue?
  - ...

Osterwalder's framework
Vision, Problem and Solution

- Business people should be able to communicate IT people:
  - The identified services, their specification (business requirements/…)
  - By using model-driven techniques starting from the business perspective down to the IT realisation

- SoaML was designed to model business and IT as a "service-oriented" system. However:
  - Business people use a different jargon and different models
  - Business people may talk about services, not about SOA which is perceived as more technical
  - SoaML is based on UML which is not typically used by business

- But:
  - SOA provides a good way to harmonise implementation with business reqs.

- We propose Business-SoaML:
  - Intermediate level that abstracts away technical terms
  - Linked to the up-coming VDML (Value Delivery Modelling Language; OMG standard)
NEFFICS: Networked Enterprise transFormation and resource management in Future internet enabled Innovation CloudS

- **Overview:**
  - STREP FP7-ICT-2009-1.3
  - 36 months, started Sept. 2010
  - [http://www.neffics.eu](http://www.neffics.eu)
  - SINTEF, Vlastuin, HSJD, Cordys, Induct, IC Focus, Aalborg University

- **Short description:**
  - Provide a cloud-based Software platform with associated methods and models for Networked Business operations, linked to relevant Innovation processes, for enabling Networked Enterprises to become Highly Innovative, Flexible and Agile.

- **Demonstrated in the following areas:**
  - Virtual extended factory (Vlastuin)
  - Virtual network of hospitals and companies (Hospital of Sant Juan de Deu)
Our aim (in the paper) is to align business and IT.
What is VDML

- Value Delivery Modeling Language
  - Answer to OMG RFP Value Delivery Metamodel (bmi/09-03-09)
  - Initial submission May 23 2011, Revised submission May 21 2012

- Goals
  - Model value exchanges between business partners & customers
  - Analysis of value exchanges in an enterprise ecosystem
  - Analysis of value requirements and sources of value from activities
  - Characterise Capabilities and map them to responsible organizations

- Scope: provide a business design abstraction that supports:
  - Organization Structure
  - Value Network Analysis
  - Value Stream Analysis
  - Capability Analysis
  - Business Model Analysis
  - Risk Analysis
  - REA (Resource Event Agent) Analysis
  - e3Value Analysis
VDML: views and example

- 7 views defined
- Value Network view

1. Collaboration view
2. Activity discovery view
3. Value network view
4. Activity network view
5. Collaboration hierarchy view
6. Role participation view
7. Value assessment view
What is SoaML

- Service oriented architecture Modeling Language (SoaML)
  - Specification for the UML Profile and Metamodel for Services (UPMS)
    - Adopted by OMG in March 2010

- Goals
  - Support for modelling services in UML:
    - bi-directional asynchronous services between multiple parties
    - Services Architectures where parties provide & use multiple services
  - Compatibility with UML, BPDM and BPMN
  - Top-down, bottom up or meet-in-the-middle modelling
  - Design by contract or dynamic adaptation of services

- Scope
  - Extensions to UML2.1 to support the following new modeling capabilities:
    - Identifying & specifying services
    - Defining service consumers and providers
    - Policies for using and providing services.
SoaML example: ServicesArchitecture & ServiceContract

Role in the service

<<ServicesArchitecture>>
TravelReservation

<<ServiceContract>>
TravelOrder

<<interface>>
TravelOrder

cancelOrder(in order : Order)
payOrder(in cardDetails : CardDetails)
createOrder(in travel : Travel, in client : Client):Order
Bridging the business-IT gap

VDML (Value Delivery Modelling Language)

CMMN (Cases)

BPMN (Processes)

Business-SoaML (Services)

Identify Cases

Has value

Identify Processes

Has value

Identify Services

Execute / Uses

Execute & uses

Deploy new services

WebServices

WS1 WS2 WS3

WSx WSx WSx

Business Strategist

Business Analyst

IT Architect
Bridging the business-IT gap

The paper concentrates on Business-SoaML, with a link to VDML
Our aims with Business-SoaML

- **Reduced complexity**
  - Straightforward functionalities,
  - Simple and business friendly
  - No unnecessary elements (e.g. service implementation details)

- **High Level Abstraction**: 
  - Specify how Participants interact with each other through services
  - Minimum amount of information in the models

- **Intuitive**: 
  - e.g. Service Contract and Services Architecture shall be easy to differentiate;

- **Relationship with value modelling**: 
  - Aligned notations
  - Relationships will be analysed in more details for future integration.
Business-SoaML: simplified metamodel
Business-SoaML: simplified notation

Services Architecture

- How Participants are connected and how they work together

Criticisms:
- Cluttered diagram
- Information overload
- Use of a container for Services Architecture is useful only if a higher level of architecture is needed
Business-SoaML: simplified notation

ServicesArchitecture

- How Participants are connected and how they work together

SoaML

<<ServicesArchitecture>>
Dealer network architecture

RoleA: buyer
- dealer: Dealer
- ps: Purchasing Service
- RoleC: enquire
- status: Ship Status
- RoleD: ship info
- ship: Ship

RoleB: seller
- mfg: Manufacturer
- RoleE: agent

RoleF: from

Business-SoaML

buyer
Purchasing Service
seller
Dealer
from
enquire
Manufacturer
agent
Ship Status
Ship info
Shipper
agent
Business-SoaML: simplified notation
Service Contract

- Specify interaction patterns based on services

**SoaML**

**Simple service contract**
- payer: Payer
- invoicer: Invoicer

**Compound service contract**
- buyer: Buyer
- seller: Seller
- orderer: Orderer
- Order processor

**Criticisms:**
- Roles and Participants should be differentiated by shape
- Compound Service Contracts look like Services Architectures while they represent different concepts
- Use of the ellipse as a container if not optimal
Business-SoaML: simplified notation

Service Contract

- Specify interaction patterns based on services
Business-SoaML editor in Oryx
Conclusion

- Business-SoaML is a simplified SoaML
  - Suitable for business
  - Initial link to VDML

- Editor
  - Current (open-source) editor implemented on Cordys (proprietary)

- We are currently validating the approach with Vlastuin

- AGR service
Future Work

- Comprehensive study with NEFFICS use cases (Manufacture / Healthcare)
  - Future improvements
  - Feedback on friendliness
- Improve the metamodel
  - Notably with Operations defined in Interface.
- Provide a mapping from VDML, as well as links between VDML and Business-SoaML
- Further Tooling support:
  - Prototype in Oryx - considered for Cordys etc.
    http://bpt.hpi.uni-potsdam.de/Oryx (open-source)
Requirements Engineering
RE Framework from K. Pohl’s book
Goal Modeling – approaches

With input from Prof. Dr. Shihong Huang
shihong@fau.edu
Florida Atlantic University
Goal Model

- A goal model is a conceptual model that documents goals, their decomposition into sub-goals, and existing goal dependencies.

- Common goal modeling language include different dialects:
  - AND/OR trees
  - The Goal-oriented Requirements Language (GRL) [GRL 2009]
  - i* [Yu 1993]
  - KAOS [Van Lamsweerde 2009]
RE-Tools (with StarUML)

- The NFR Framework for non-functional requirements (NFRs) modeling
- The i* Framework for agent-oriented modeling
- KAOS for formal goals modeling
- Problem Frames for business and system requirements and specifications modeling
- UML for object-oriented modeling

http://www.utdallas.edu/~supakkul/tools/RE-Tools/
**Main**

The RE-Tools is an open source requirements modeling toolkit that supports the following notations:

- The **NFR Framework** for non-functional requirements (NFRS) modeling
- The **{ Framework** for agent-oriented modeling
- **KAOS** for formal goals modeling
- **Problem Frames** for business and system requirements and specifications modeling
- **UML** for object-oriented modeling

These notations may be modeled separately or together in an integrated manner. Created diagrams are represented in-memory based on a meta-model and stored in text files based on an XML/XMI format, thus can be navigated in-memory via the StarUML COM-based API or parsed using a standard XML/XMI parser.

Below is a brief overview video clip.
i*(i-Star) Framework

- Comprehensive approach for documenting and analyzing goals and goal dependencies
- i* is based on the idea that an actor depends on other actors in order to achieve its goals
- In i*, the actors and their dependencies are documented in the strategic dependency model
- In addition, the goals, tasks, etc. of each actor are documented using the strategic rational model
i*(i-Star)

- Based on the modeling language GRL
- AND/OR trees for documenting goal decompositions
- GRL provides modeling constructs for quality aspects
- Basic concepts
  - Objects
  - Dependencies
  - Relationships, which is further subdivided into
    - Dependencies
    - links
Notation of the modeling constructs in the i* framework
StarUML with RE module
RE Modules – Goal modeling

- iTropos
- KAOS
- System Problem Frame
- NFR Framework
- Problem Interdependency Graph
KAOS

- KAOS modeling language is part of the KAOS framework for eliciting, specifying, and analyzing goals, requirements, scenarios, and responsibilities assignments.
- Framework with six complementary views or sub-models:
  - Goal model
  - Obstacle model
  - Object model
  - Agent model
  - Operation model
  - Behaviour model
- All sub-models are interrelated via traceability links.
KAOS - Keep All Objectives Satisfied


Creating a KAOS Goal Model
Achieve [Train Stopped At Block Entry If Stop Signal]

Stop Signal And Not Train Stopped At Block Entry

Signal Not Visible

Driver Unresponsive

Brake System Down

Achieve [Responsiveness Check Sent Regularly]
Constructs for goal modeling in KAOS

- Objects
  - Behavioural goal
  - Softgoal
  - Agent

- Relationships
  - AND-decomposition
  - Alternative decomposition
  - Potential Conflict
  - Responsibility assignment
Basic constructs of the KAOS framework for modeling goals and assigning responsibilities for goals to agents
In KAOS, a goal is either a behavioral goal or a softgoal, and is defined as a prescriptive statement of intent that the system should satisfy through the cooperation of its agents. Characterize:

- Behavioral goals
- Softgoals, and
- agents
KAOS Object: Behavioral goals

- Behavioral goal: Set of admissible system behaviors
- Can be defined in a clean-cut manner, i.e., one can verify whether the system satisfies a behavior goal or not
- Example: “The train doors shall remain closed while the train is moving”

Two types of behavior goals
- Achieve goal: the defined property must eventually hold
- Maintain goal: the defined property must always hold (possibly under some condition)
KAOS Object: Softgoals

- Document preferences among alternative system behaviors
- Like in i*, softgoal has no clear cut criterion for satisfaction
- Hence, softgoals are expected to be satisfied within acceptable limits rather than absolutely
- Example: “the system shall minimize the traveling time”
KAOS Object: Agent

- While i* focuses primarily on agents within organization structures.
- Agent in KAOS related to users and components of software-intensive systems.
- An agent is defined as an active system component with a specific role for satisfying a goal (e.g., user).
- An agent can be a human agent, a device (e.g., a sensor or actuator), or a software component.
KAOS - Relationships

- **AND-decomposition:**
  - Relates a super-goal to a set of sub-goals
  - The super-goal is satisfied if all sub-goals are satisfied

- **Alternative decomposition (OR-decomposition):**
  - Assignment of multiple AND-decompositions to the super-goal
  - Each alternative is a set of AND-decompositions (may be only one)
  - The super-goal is satisfied if one of the alternatives (one AND-decomposition assigned to the super-goal) is satisfied
KAOS - Relationships

- Potential Conflict
  - One goal may prevent satisfying another goal
  - This link does not correspond to the conflict dependency, which documents that when one goal is satisfied, the other goal cannot be satisfied

- Responsibility assignment (relation of goals to agents)
  - Means that the agent is responsible for satisfying the goal
  - Only terminal goals (goals that cannot be refined) can be assigned to an agent
  - Alternative responsibility assignments can be defined, i.e., a goal can be related to multiple agents using a responsibility assignment relationship
Modeling goals in KAOS

- The goal (sub-) model documents goals softgoals, and their dependencies, i.e.,
  - AND-decompositions
  - Alternative decompositions, and
  - Potential conflicts
Example of a goal model in KAOS

Decomposition of the goal “avoid traffic jams”

Conflict link

Contribute to two softgoals

AND-
Decomposed into two goals

OR-
Decomposed into two goals

Shihong Huang
Modeling goals in KAOS

- Goal sub-model
  - Documents goals, softgoals and their dependencies

- Agent sub-model
  - Defines agents

- Responsibilities
  - Link goal sub-model with agent sub-model
Modeling Responsibility Assignments in KAOS

- Are used to interrelate goals defined in the goal sub-model and agents defined in the agent sub-model
Example of responsibility assignment in KAOS
Deciding Which Goal Modeling Language to Use

- Extended AND/OR graphs
  - Document goals and dependencies, but no agents
  - Cannot be used to model actors and relationships between actors
- i* framework
  - Primarily targeted at actors and their goals within organization and organization structures
  - Main focus: the system environment
  - Sophisticated modeling language in terms of actors, dependencies between actors, and rationale structures of the individual actors
  - Requires some training before use
Deciding Which Goal Modeling Language to Use

KAOS

- Targets on agents that are system users or components of software systems
- The basic concepts for defining goals in KAOS framework are very similar to extended AND/OR goal graphs
- Support conflicts and softgoals
- Five other supplementary models, also for hardware components – thus support interrelating goal models with data, functional, and behavior models
- Very suitable for embedded systems
- Requires some training before use
Hint: Choosing a goal modeling language

- Use extended AND/OR graphs if your main focus is the documentation of goals, their decompositions, and their dependencies.
- Use i* models if you aim to document and analyze the relationships between different actors in an organization using an agent-oriented modeling paradigm.
- Use KAOS models if you aim to document the intended properties of the hardware and software components of a software-intensive paradigm and if you aim to relate the defined goals with solution-oriented requirements models.
References


Axel van Lamsweerde “Requirements Engineering from System Goals to UML Models to Software Specifications”, Wiley 2011
KAOS support

Tool:

Reading:
ISO standards, 250xx

- Quality Requirements Division 2503n
- Quality Model Division 2501n
- Quality Management Division 2500n
- Quality Measurement Division 2502n
- Quality Evaluation Division 2504n
- Extension Division 25050 - 25099
ISO 25010

Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models
Targets of Quality Models

Key:
- What is measured by the model:
- Some of the factors that influence quality in use
Quality in Use

- Effectiveness
  - Effectiveness
- Efficiency
  - Efficiency
- Satisfaction
  - Usefulness
  - Trust
  - Pleasure
  - Comfort
- Freedom from risk
  - Economic risk mitigation
  - Health and safety risk mitigation
  - Environmental risk mitigation
- Context coverage
  - Context completeness
  - Flexibility
Product Quality
Quality Measures

- Software Product Quality
  - Quality Characteristics
    - Quality Sub-Characteristics
  - Quality Measures
    - Measurement Function
      - Quality Measure Elements

- Composed of
- Indicate
- Generates
- Are applied to
Measurements
Tom and Kai Gilb
‘Quantifying All Quality Requirements’

https://www.dropbox.com/sh/w05apx9s1qbfpsz/IkQ78CZLfv

www.Gilb.com
See ‘Downloads’
Multiple Required Performance and Cost Attributes are the basis for architecture selection and evaluation.
User-Friendliness

**Scale:** average time in minutes, to learn how to program contact names and telephone numbers into the memory of the phone.

**Goal 5**

[within 1 year]

The challenge is to move from the Past level of 35 min to learn to the Goal level of 5 min to learn!

Within the time frame
Scale and Meter, Past and Goal

User-Friendliness. Learn. Contacts

Scale: average time in minutes, to learn how to program contact names and telephone numbers into the memory of the phone.

Meter: time 5 people who have not had a mobile telephone before, as well as 5 people who has, use the average.

Past 35 min.

Goal [within 1 year] 5 min.
Structuring concepts

Business Model

Context Business Model

Community model

Business Process & Roles Model

Business Resource Model

Vision for change

Scoping Statement

Context statement

Risk analysis

Goal model

Work analysis Refinement
BMM – Business Motivation Model (OMG)

2007 standard from OMG, based on input from the Business Rules Community

http://www.omg.org/technology/documents/br_pm_spec_catalog.htm

Example of a relevant metamodel, with a domain specific language, and editor support with EMF/GMF.
What BMM Offers

- Standardizes common Business terms and Business relationships
- Provide a open medium for communicating Business Plans
- Provides a “bridge” for relating the WHAT and the WHY to the HOW:
  - Business Processes
  - Business Rules
  - Organizational Structure
BMM (Metamodel)
BMM overview

executed by

Services Architecture/Community
BMM modeling
Concrete Syntax

- No mandated syntax.
- Which is best?

Textual Syntax

Structured Syntax

Graphical Syntax

EU-Fly is vulnerable to direct competition from low-cost airlines on many of its single-sector routes. Revenue from business travel is critical.
Graphical Syntax Option

- Implemented in Eclipse, using GMF (Graphical Modeling Framework)
Example

- Business travel takes EU-Fly beyond break even
  - quantified by: Business travel targets
- Economy travel delivers additional profit
  - quantified by: Profit targets
- EU-Fly is perceived as value for money
  - quantified by: Press reports