Requirements Engineering: Frameworks & Templates

CECS 590

Recap time!

- System Models: Why and what?
- Refinement and Decomposition – what's that?
- What are Abstraction Levels?
  Why are they helpful?
- Which Modeling Views can we use?
- What is Rationale and why do we want it?
- What is Traceability and why does it matter?

Requirements Engineering – Outline

- WHY do we need Requirements Engineering and what is it?
- Principles: Definitions, process, roles, problem/solution view, artifact orientation
- System Models: Decomposition and abstraction, system views
- **Frameworks: What reference structures can I use for requirements?**
- Business Case Analysis: Why are we building this system?
- Stakeholders: Who are the people to talk to about requirements?
- Goals and Constraints: What are the major objectives for the system?
- System Vision: What exactly do we want to achieve?
- Domain Models: What are the surrounding systems ours interacts with?
- Usage Models: How will the system interact with the user?
- Software quality models: How to determine the quality characteristics?
- Quality requirements: How to specify which qualities need to be met?
- Process requirements: How to specify constraints for development?
- Towards a system specification: How to hand over to design?
- Quality assurance: How to ensure that RE is done in a good way?
- Change management: How to evolve requirements?

Frameworks

- What is a Framework?
- Standards (IEEE, Volere, ect.)
- Content & Artefact Models

What is a Framework?

- Def.: Framework
  A generic means of structuring and organising information.

Henk L, stock.xchng

Simple template for phrasing requirements:

[when?] [under what conditions?]
THE SYSTEM SHALL | SHOULD | WILL <process>
<thing to be processed> [<process details>*]

Exercise: Let‘s use the template

[when?] [under what conditions?]
THE SYSTEM SHALL | SHOULD | WILL <process>
<thing to be processed> [<process details>*]

Think of an ATM as example.

Standards: Volere

Requirement #: 75 Requirement Type: 9 Event/BUC/PUC #: 7, 9

Description: The product shall record all the roads that have been treated

Rationale: To be able to schedule untreated roads and highlight potential danger

Originator: Arnold Snow - Chief Engineer

Fit Criterion: The recorded treated roads shall agree with the drivers' road treatment logs and shall be up to date within 30 minutes of the completion of the road's treatment

Customer Satisfaction: 3 Customer Dissatisfaction: 5

Dependencies: All requirements using road and scheduling data Conflicts: 105

Supporting Materials: Work context diagram, terms definitions in section 5

History: Created February 29,2010

Volere

Discussion

- What kind of frameworks have you used?
- How were they organized?
- What did they provide?

Standards: IEEE 830

Table of Contents of an SRS

1. Introduction
   - 1.1 Purpose
   - 1.2 Scope
   - 1.3 Definitions, acronyms, and abbreviations
   - 1.4 References
   - 1.5 Overview

2. Overall description
   - 2.1 Product perspective
   - 2.2 Product functions
   - 2.3 User characteristics
   - 2.4 Constraints
   - 2.5 Assumptions and dependencies

3. Specific requirements *(the largest & most important part of an SRS, organized in different manner)*

Supporting informations
- Table of contents and Index
- Appendixes *(e.g. Samples I/O formats, background, description of problem to be solved by the software)*

IEEE Standard 830-1998:
Document template for Software Requirements Specifications

EARS: Easy Approach to Requirements Syntax


©Alistair Mavin

- Top-level system requirements are typically written in *Natural Language* (NL) by individuals who are not requirements experts.
- Unconstrained NL can cause problems.
- There is a need for simple, easy to apply guidance.
- EARS Templates are based on industry best practice and many years of experience at Rolls Royce [Alistair Mavin, EARS Tutorial at RE 2010].
- Provides simple, general templates that can be used for all NL requirements:
  - Four that specify normal operation (wanted behavior)
  - One that specifies unwanted behavior
- Exposes lack of understanding
  - Prevents ambiguity and vagueness
  - Consideration of the desired system behavior

EARS – Concepts (1 of 3)

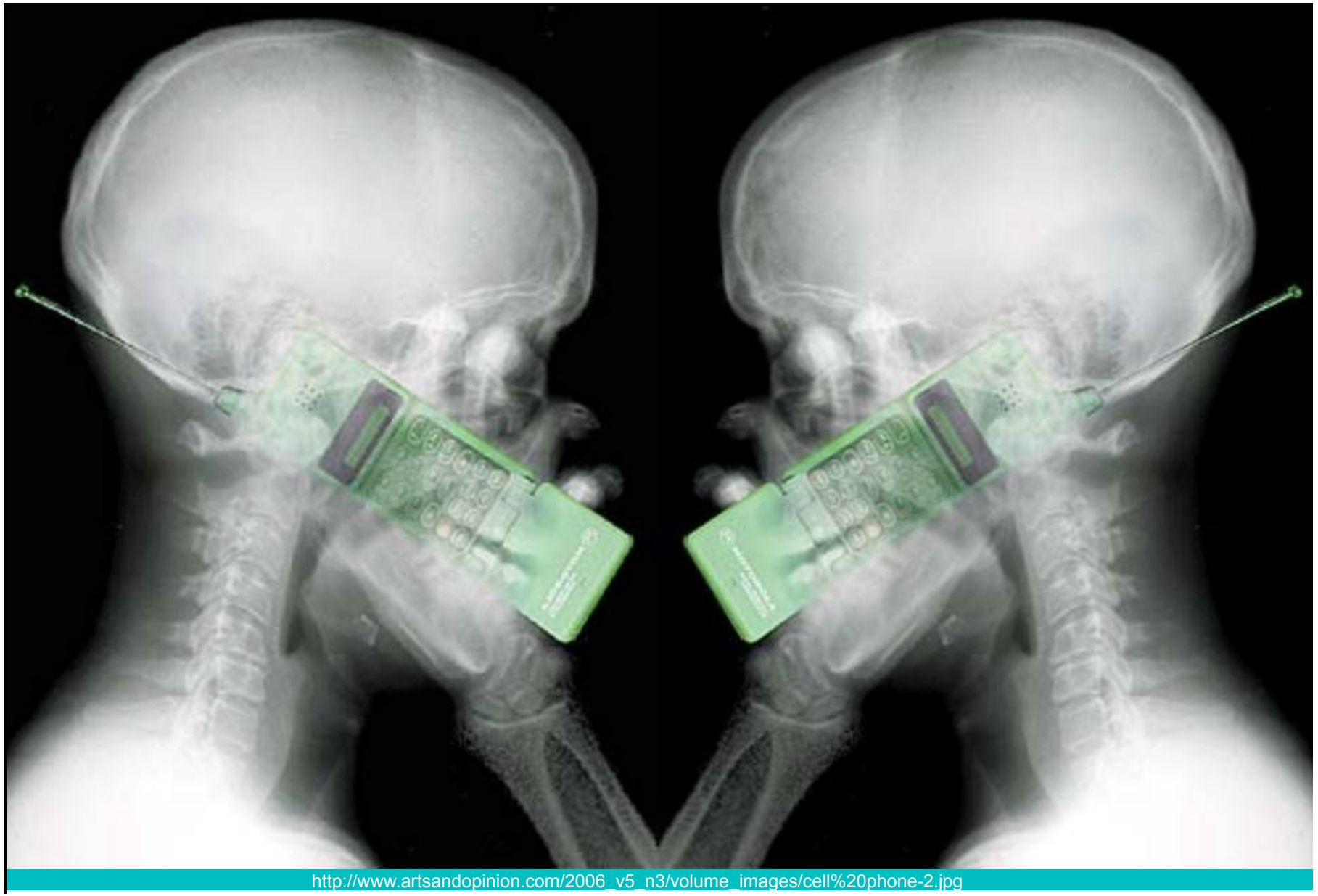
- There are two *classes* of requirement
  - *Normal operation*
  - *Unwanted behavior*
- All NL requirements can be defined using one of 5 simple templates
  - 4 *normal operation* templates
  - 1 *unwanted behavior* template

EARS – Concepts (2 of 3)

- *Normal operation* requirements
  - Define the required system behavior during *sunny day* operation
  - All users and all interacting systems behave as expected to meet the goals of the user

EARS – Concepts (3 of 3)

- *Unwanted behavior* requirements
  - A general term used to cover all deviations from *sunny day* operation
  - Define the *required response* of the system to
- Failures and disturbances
- Deviations from desired user behavior
- Unexpected behavior of interacting systems



http://www.artsandopinion.com/2006_v5_n3/volume_images/cell%20phone-2.jpg

EARS – Normal operation

- Generic syntax is

<optional preconditions> <optional trigger> the <system name> shall <system response>

- Simple structure adds rigor & clarity
- System response describes what the system must actually do that is *visible* at the boundary of the system

EARS – Normal operation

- Ubiquitous
  - Requirement is always active
- Event-driven (keyword *When*)
  - Required response to a triggering event
- State-driven (keyword *While*)
  - Required response in a specified state
- Option (keyword *Where*)
  - Applicable only if *feature* is included
- *Complex*
  - *can use combinations of When, While and Where for requirements with complex conditional clauses*

EARS – Normal operation

- Ubiquitous:
  - *The <system name> shall <system response>*
  - Used to define system behavior that must be active at all times: *"continuous"*
  - No preconditions or trigger: *"unconditional"*
- Examples:
  - Car
    - *The car shall have a maximum retail sale price of XXX*
    - *The car shall be compliant with the safety requirements defined in XXX*
  - Laptop
    - *The laptop shall have a mass of no more than XXX grams*
    - *The laptop shall have a minimum battery life of XXX hours*

EARS – Ubiquitous

Exercise: write some ubiquitous requirements for a mobile phone

http://www.celtnet.org.uk/mobile-phone/img/mobile-evolution.gif

EARS – Normal operation

- Event-driven (keyword When):
  - *When <trigger> the <system name> shall <system response>*
  - Initiated only when a triggering event is detected at the system boundary
  - The trigger must be something that the system itself can detect
    - *This often helps clarify the system boundary.*

EARS – Event-driven

- Example: Car
  - *When the clutch pedal is depressed, the car shall disengage the driving force*
  - *When the "turn indicator" command is received, the car shall operate the indicator lights on the front, side and rear of the vehicle, and provide audible and visual confirmation to the driver*
- Example: Laptop
  - *When the laptop is off and the power button is pressed, the laptop shall boot up*
  - *When the laptop is running and the laptop is closed, the laptop shall enter "powersave" mode*

EARS – Event-driven

Exercise: write some event-driven requirements for a mobile phone

http://www.healthspablog.org/wp-content/uploads/2009/12/Effects-of-mobile-phone-on-children-brains.jpg

EARS – Normal operation

- State-driven (keyword While):
- *While <in a specific state> the <system name> shall <system response>*
- Requirement is active while the system is in a defined state
  - Requirement is “continuous”, but only while the system is in the specified state

EARS – State-driven

- Examples Car
  - *While the ignition is on, the car shall display the fuel level and the oil level to the driver*
  - *While the key is in the ignition, the car alarm shall be inhibited*
  - *While the handbrake is applied, the wheels shall be locked*
- *Examples* Laptop
  - *While the laptop is running on the battery and the battery is below XXX % charge, the laptop shall display "low battery"*
  - *While an external audio output device is connected, the laptop shall mute the built-in speaker and send the audio output signal to the external audio output device*

EARS – State-driven

Exercise: write some state-driven requirements for a mobile phone

http://www.whatsgottago.com/wp-content/uploads/2007/12/angry_phone.jpg

EARS – normal operation

- Option (keyword Where):
  - *Where <feature is included> the <system name> shall <system response>*
- Applicable only in systems that include a particular *feature*
  - *The requirement will often be "ubiquitous", but only for systems that include the specified feature*

EARS – Option (Where)

- Examples Car
  - *Where the car has electric windows, the electric window controls shall be on the driver's door panel*
  - *Where the car includes automatic windscreen wipers, the car shall sense moisture on the windscreen and operate the windscreen wipers without driver commands*
- Examples Laptop
  - *Where a "long life" battery is fitted, the laptop shall have a minimum battery life of XXX hours*
  - *Where the laptop is a "lightweight" model, the laptop shall have a mass of no more than XXX grams*

EARS – Option (Where)

Exercise: write some option requirements for a mobile phone

http://www.maclife.com/files/u32/0708_contract_200.jpg

EARS – Unwanted behavior

Unwanted behavior: IF <optional preconditions> <trigger>, THEN the <system name> shall <system response>

- A variation of e*vent-driven* requirement.
- *If <optional preconditions> <trigger>, then the <system name> shall <system response>*
- This format forces the separation of
  - Circumstances in which the requirement can be invoked (preconditions)
  - The initiating event (trigger)
  - The expected system behavior (response)

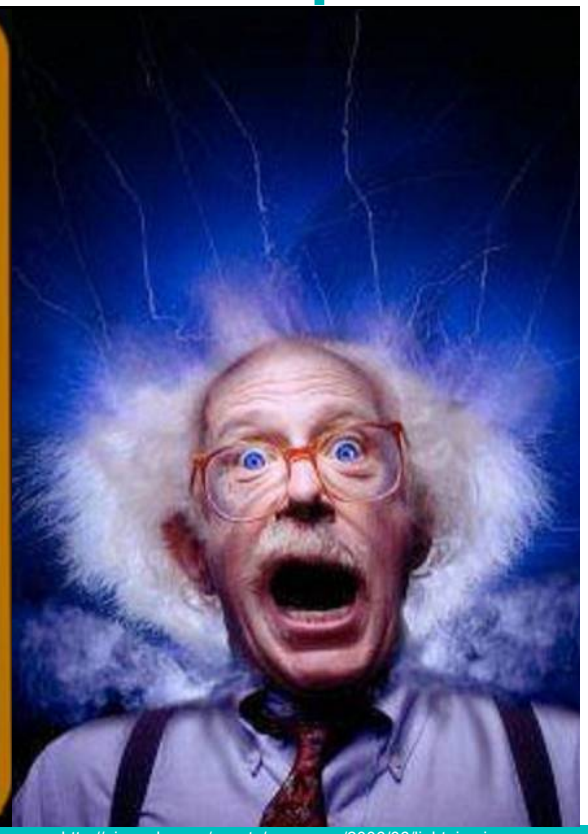
EARS – Unwanted behavior

- Examples Car
  - *If the car detects attempted intrusion, then the car shall operate the car alarm*
  - *If the car detects low oil pressure, then the car shall display a "low oil pressure" warning*
- Examples Laptop
  - *If the incorrect password is entered, then the laptop shall display XXX warning message*
  - *If the laptop is connected to a non-compatible device, then the laptop shall prevent transfer of data, prevent transfer of charge, display XXX warning message and not be damaged*

EARS – Unwanted behavior

Exercise: write some unwanted behaviour requirements for a mobile phone

http://www.newlaunches.com/entry_images/0708/21/cellphone-lightning-strike.jpg



http://gizmodo.com/assets/resources/2006/06/lightning.jpg

EARS - Complex requirements

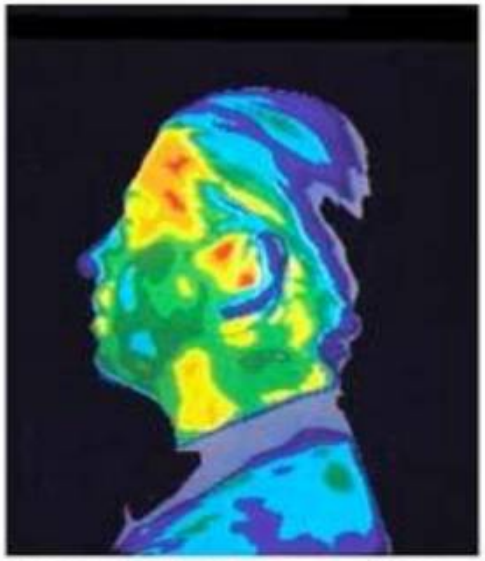
- Requirements with complex conditional clauses are defined using combinations of *When*, *While, Where and If-Then*
- The keywords can be built into more complex expressions to specify richer system behaviors
- For instance, the same event may trigger different system behavior depending on the state of the system when the event is detected

EARS - Complex requirements

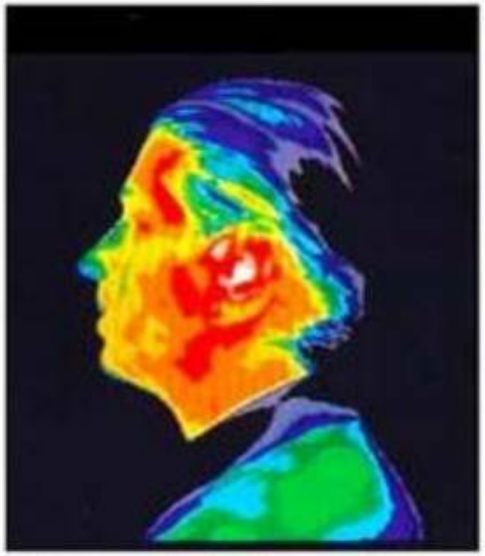
- Examples Car
  - *Where the car includes an "owner alert" system, if the car detects attempted intrusion, then the car shall send a message to the owner and activate the car alarm*
  - *While the car is being being driven forwards above a speed of XXX, if the driver attempts to engage reverse gear, then the car shall prevent engagement of reverse gear*
- *Examples Laptop*
  - *Where the laptop includes "voice input" option, while the voice input option is selected, the laptop shall accept voice input commands*
  - *While the laptop is running on mains electrical power, if the power cable is disconnected, then the laptop shall display a warning message*

EARS - Complex requirements

Exercise: write some complex requirements for a mobile phone



Thermographic Image of the head with no exposure to harmful cell phone radiation.



Thermographic Image of the head after a 15-minute phone call. Yellow and red areas indicate thermal (heating) effects that can cause negative health effects.

http://www.passenlaw.com/blog/wp-content/uploads/2010/01/cell-phone-brain-cancer-radiation.jpg

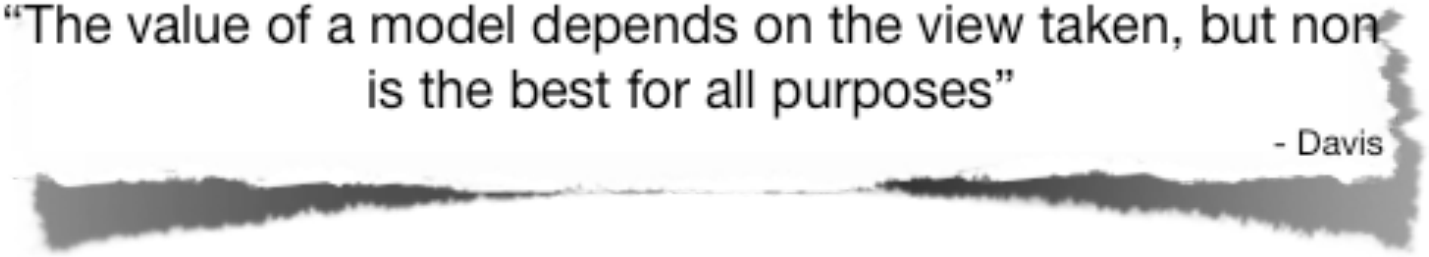
EARS - Summary

- Provides simple, general guidance
  - Templates support complex requirements
- Strengths
  - Provides rigor and consistency
  - Easy to learn and apply
  - No tools needed
  - Common form of requirements communication
- Weaknesses
  - Limited inter-requirement coupling
  - Unsuitable for very complex requirements (consider using truth tables or other non-textual notation)

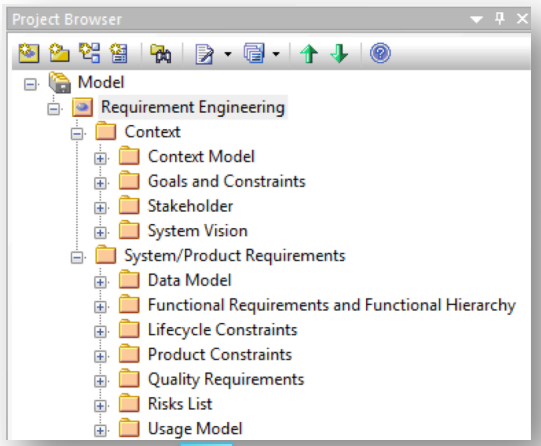
What exactly is an Artefact?

What is an Artefact/Artefact Model?

- **Artefact:**
  - Documents (intermediate) results of development process steps
  - Has structure, contents, and a representation
  - Has means as an out- and input, subject to version control
  - Examples: documents, data objects, and models
- **Artefact model:** All artefacts that are relevant throughout the development process plus their dependencies
- Careful:
  - In literature there exists a plethora of definitions and terms that are used synonymously (Artefact contents are represented in the form of: meta models, ontologies, product models, ...)
  - Artefact models can be structured and represented differently!

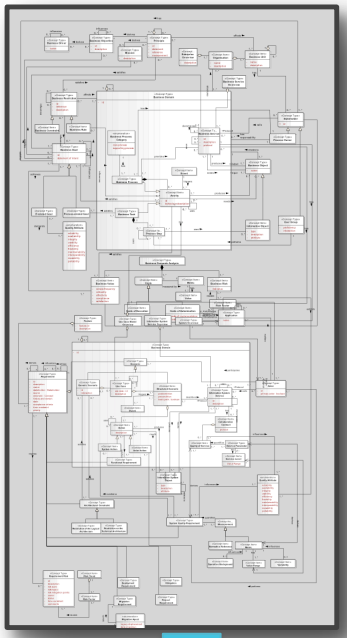


Frameworks: Artefact model (views)



Structure

- Packets / Document hierarchies

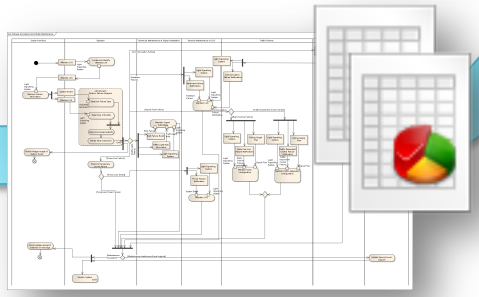


Content

- Concept models
- Ontologies
- Checklists
- …

Artefacts (Materialisation)

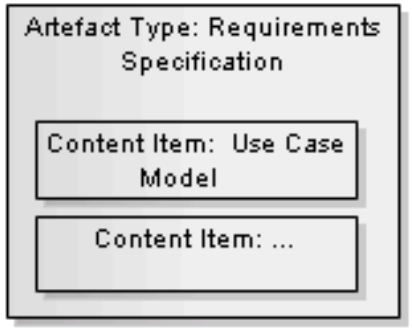
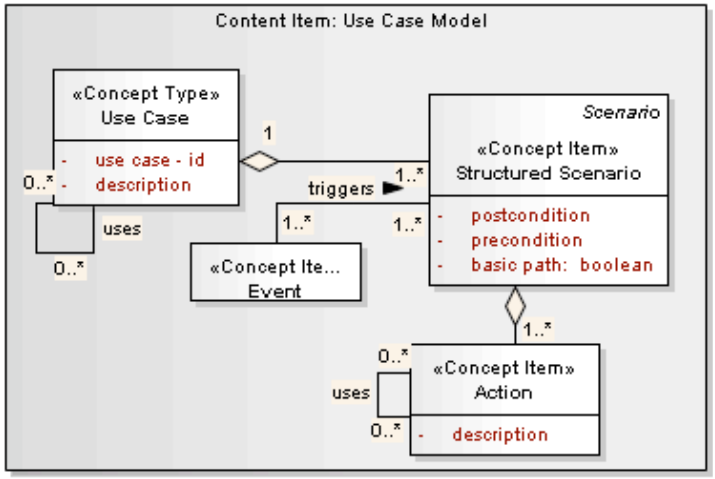
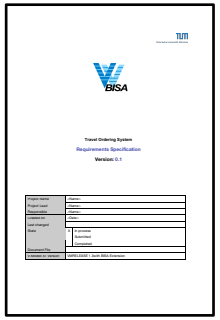
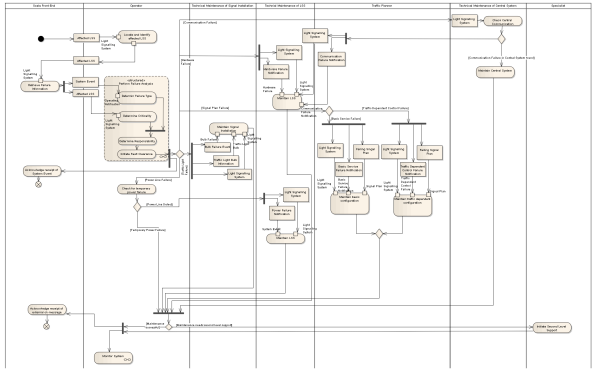
- Documents
- Data models



Representation

- Diagrams / Models
- Natural language text
- …

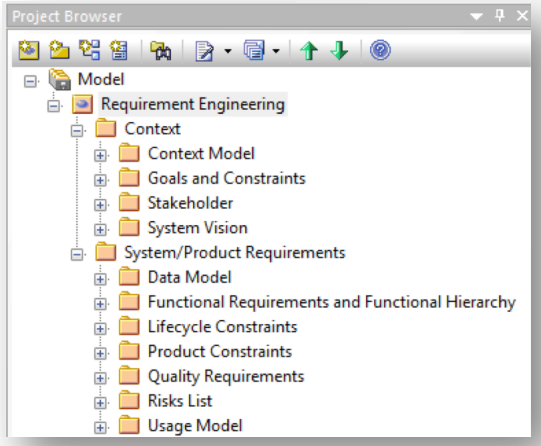
Example for Structure and Content Model

	Structure Model	Content Model
Reference model (M1)		
Project instance (M0)		

* Mendez, Kuhrmann: Artefact-based RE and its Integration into a Process Framework, Technical Report, 2009

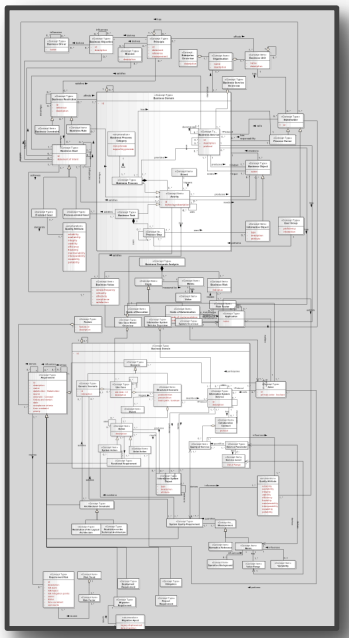
CONTINUE HERE

Structure and content allow for flexibility and precision



Structure

- Packets / Document hierarchy
- "Content Items" structure content



Content

- Modeling concepts
- Ontologies
- Checklists
- …

Structure Model

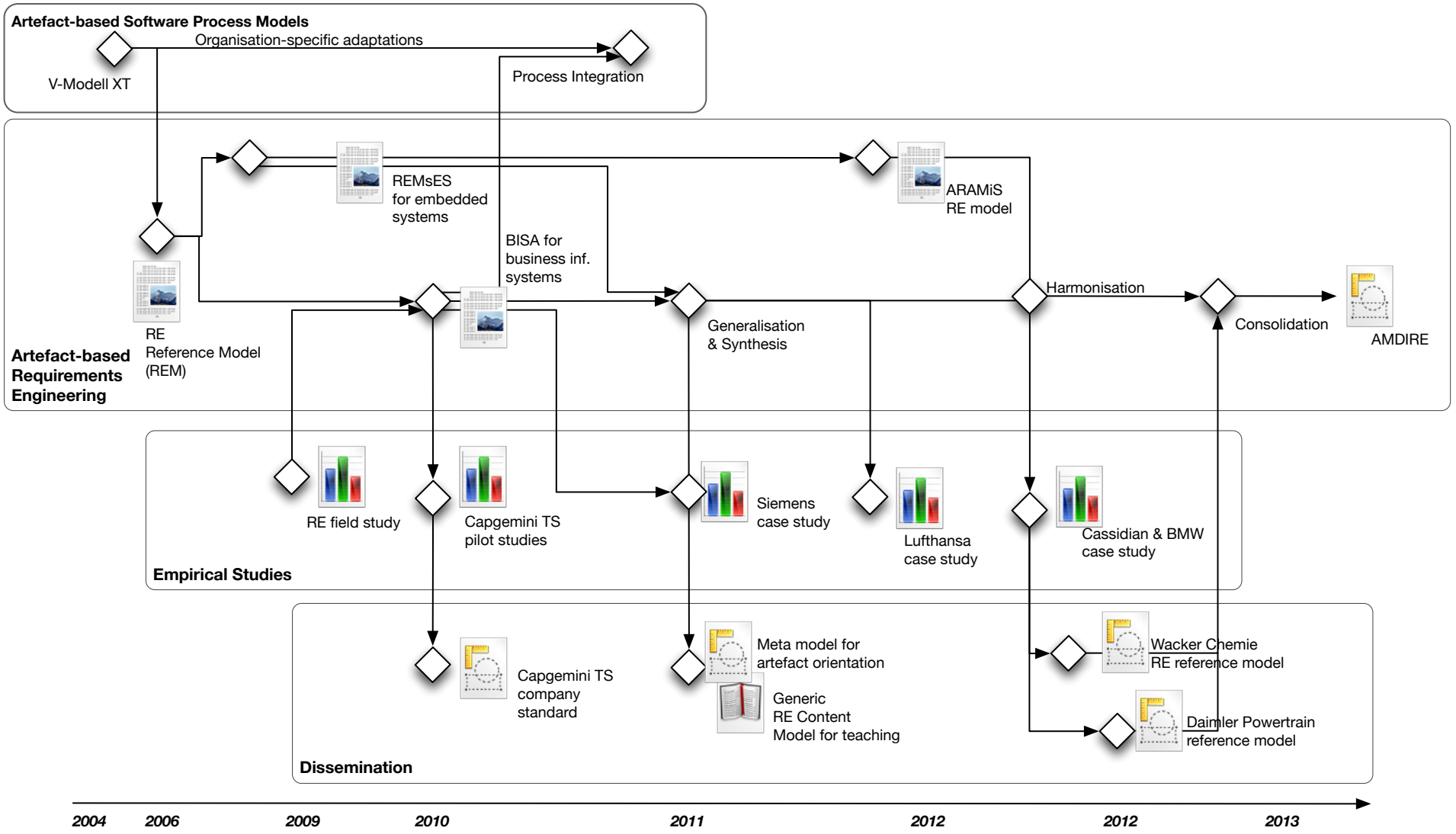
- Outline
- Dependencies
- Basis for adaptation

Content Model

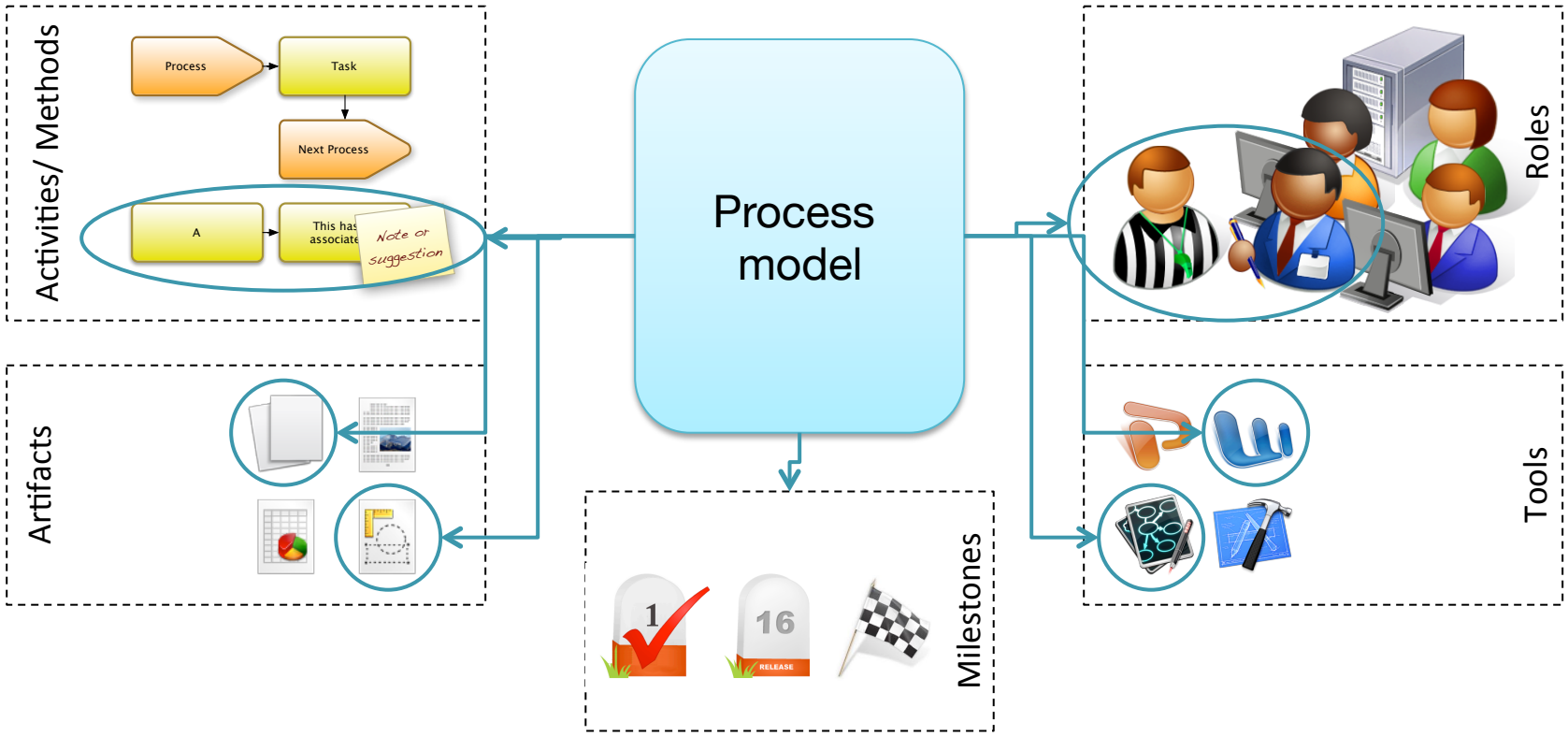
- Required type of information
- Modeling concepts / contents

→ Conscious handling of content and depedencies

Background: 6 years of research cooperations

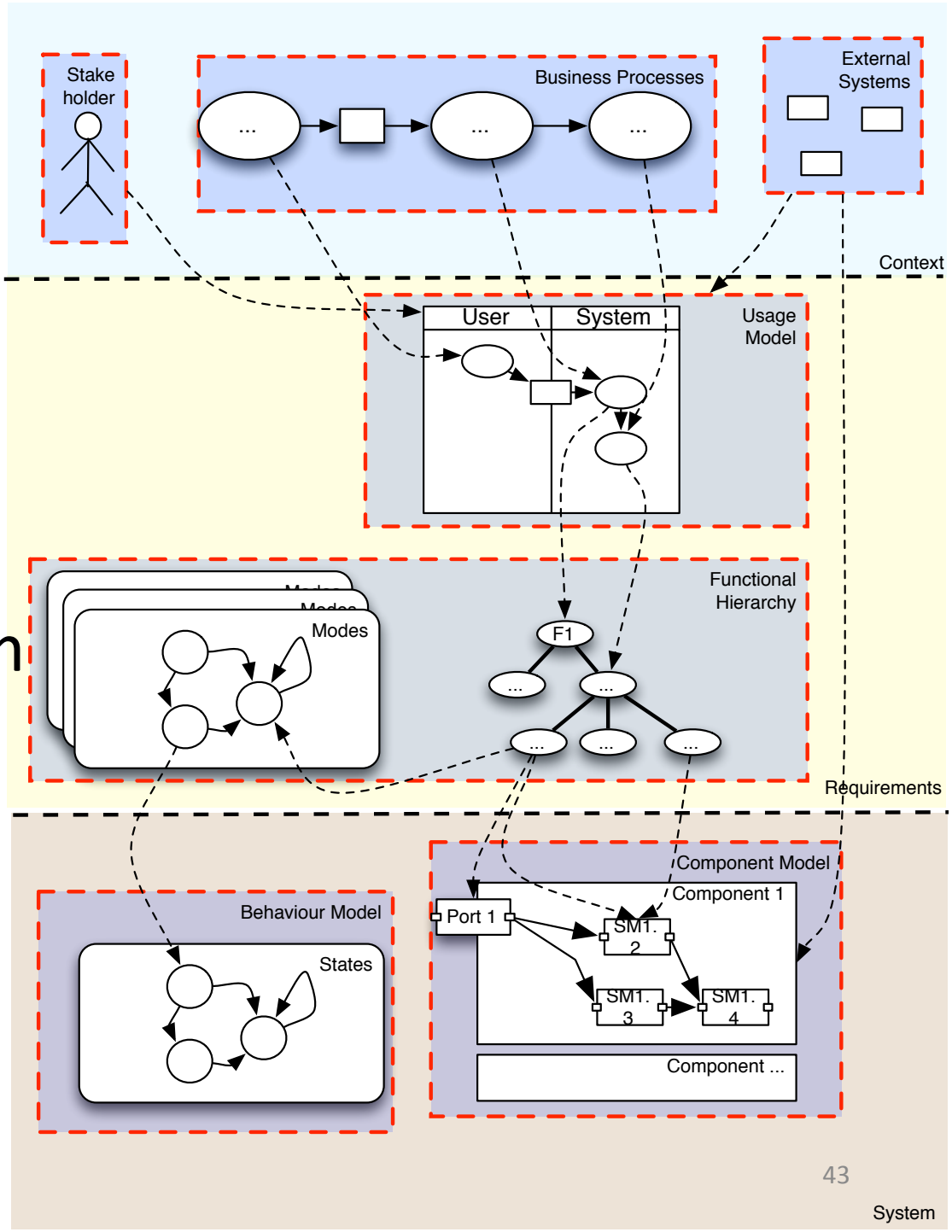


Use general process model

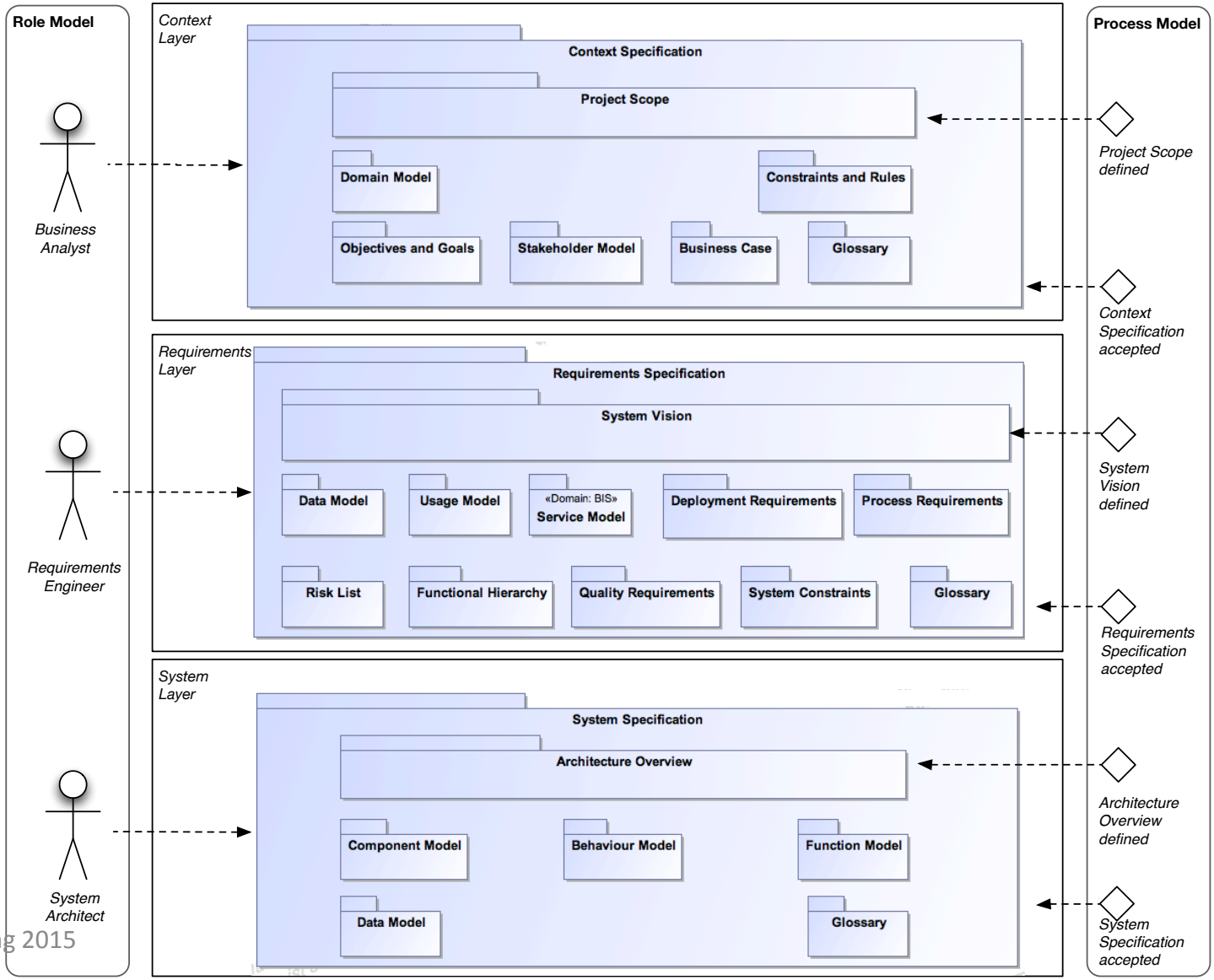


Use abstraction layers

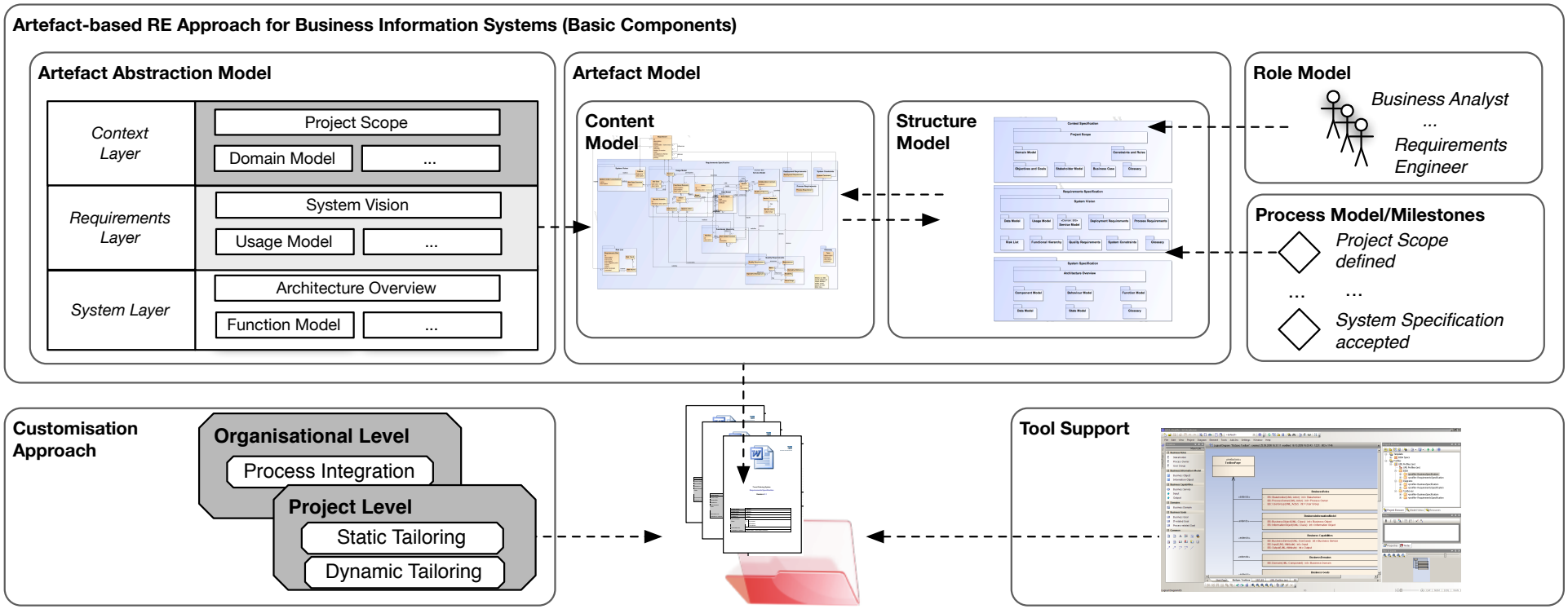
- Structuring of contents using **System Model:** views onto system
  - Context
  - Interface
  - Behavior
  - Components



Use roles and milestones

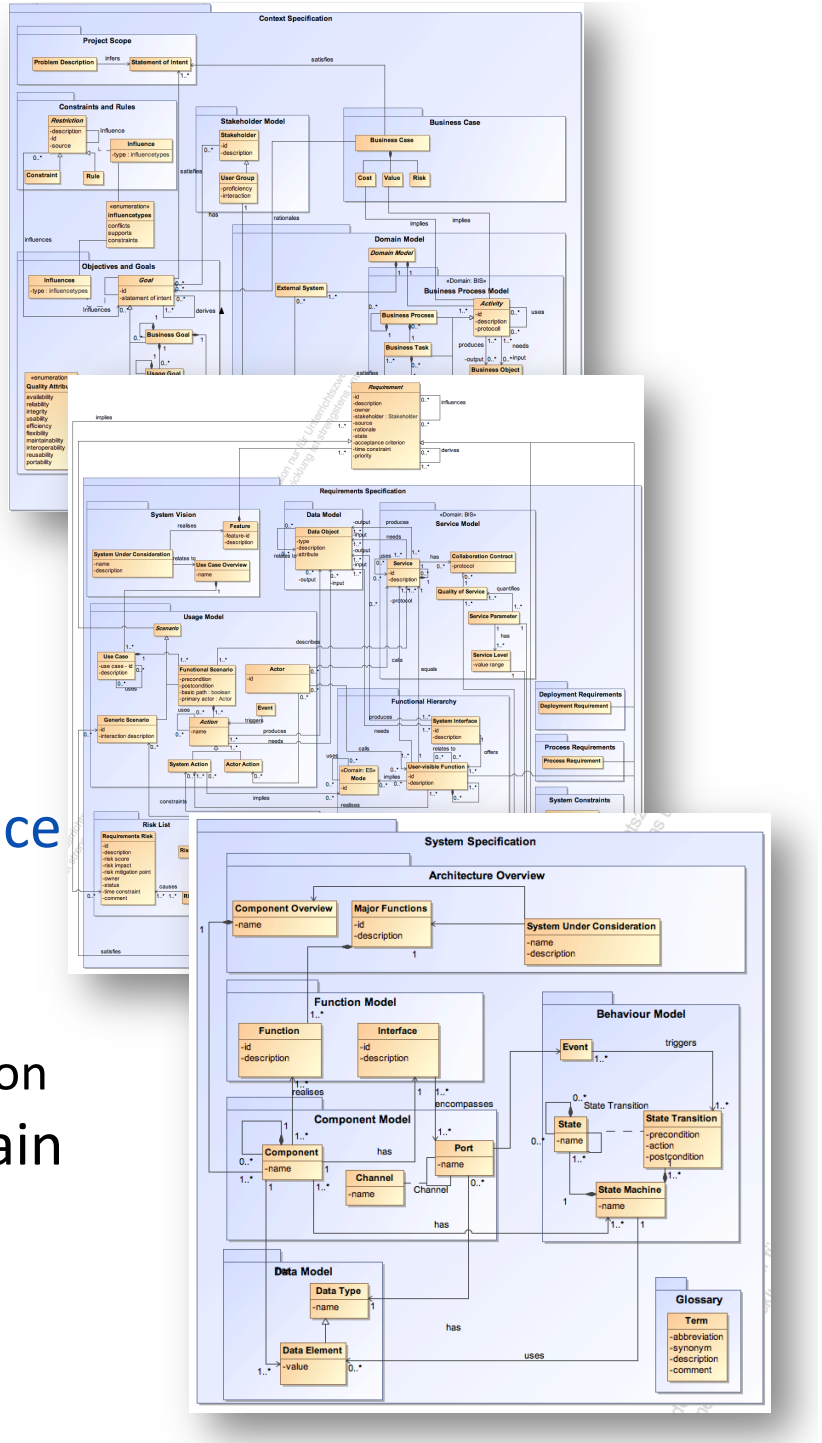


AMDiRE Overview



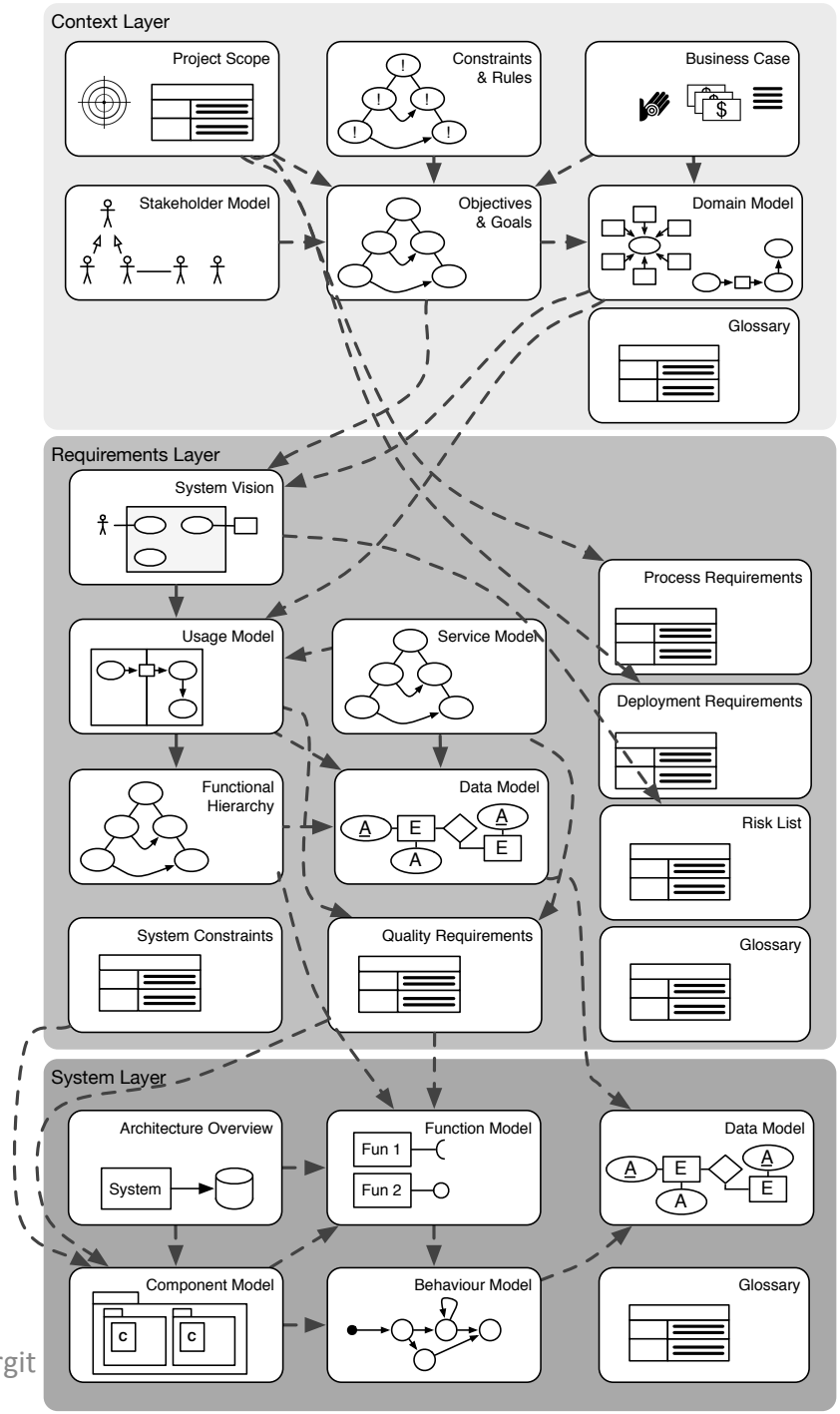
AMDiRE
Artefact model

- 3 Artefacts
  - Context Specification
  - Requirements Specification
  - System Specification
- Content Model contains:
  - Problem space and solution space as well as interdependencies
    - Problem space: Context and Requirements Specification
    - Solution space: System Specification
  - Domain specific as well as domain independent content

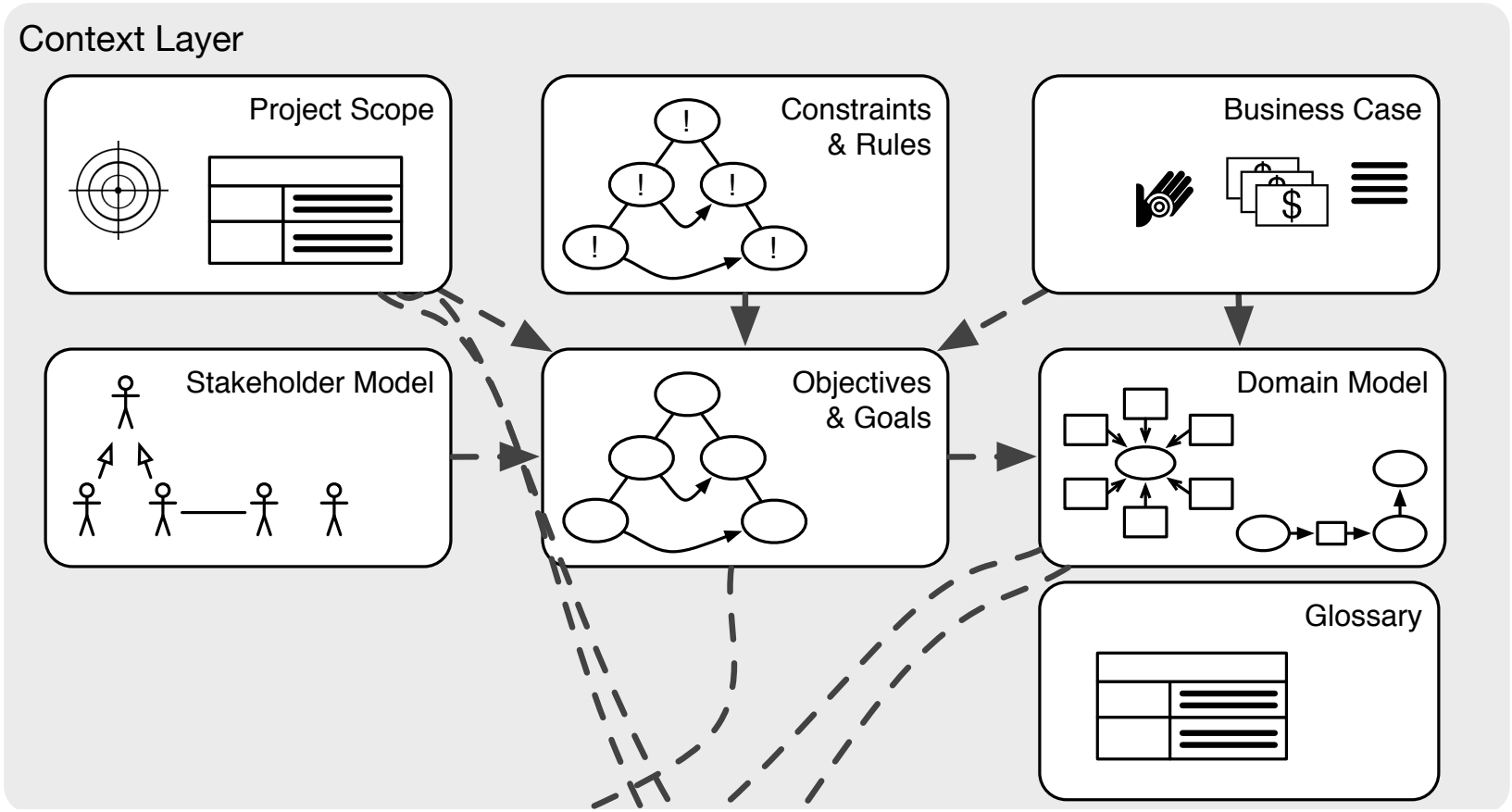


Frameworks: AMDiRE Artefact Model

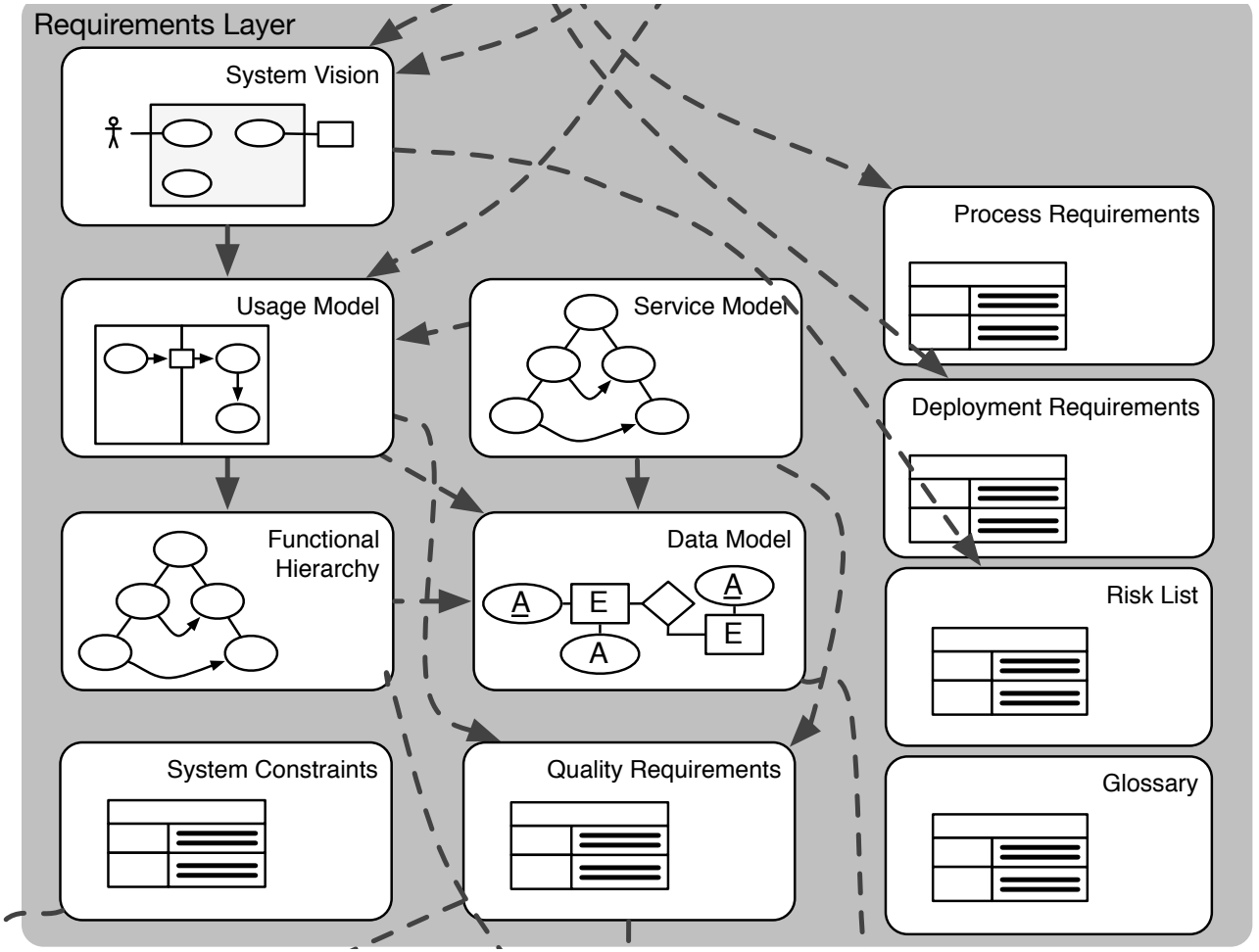
- Content model
- With concepts
- Plus structure
- Abstraction layers
- Common system model behind



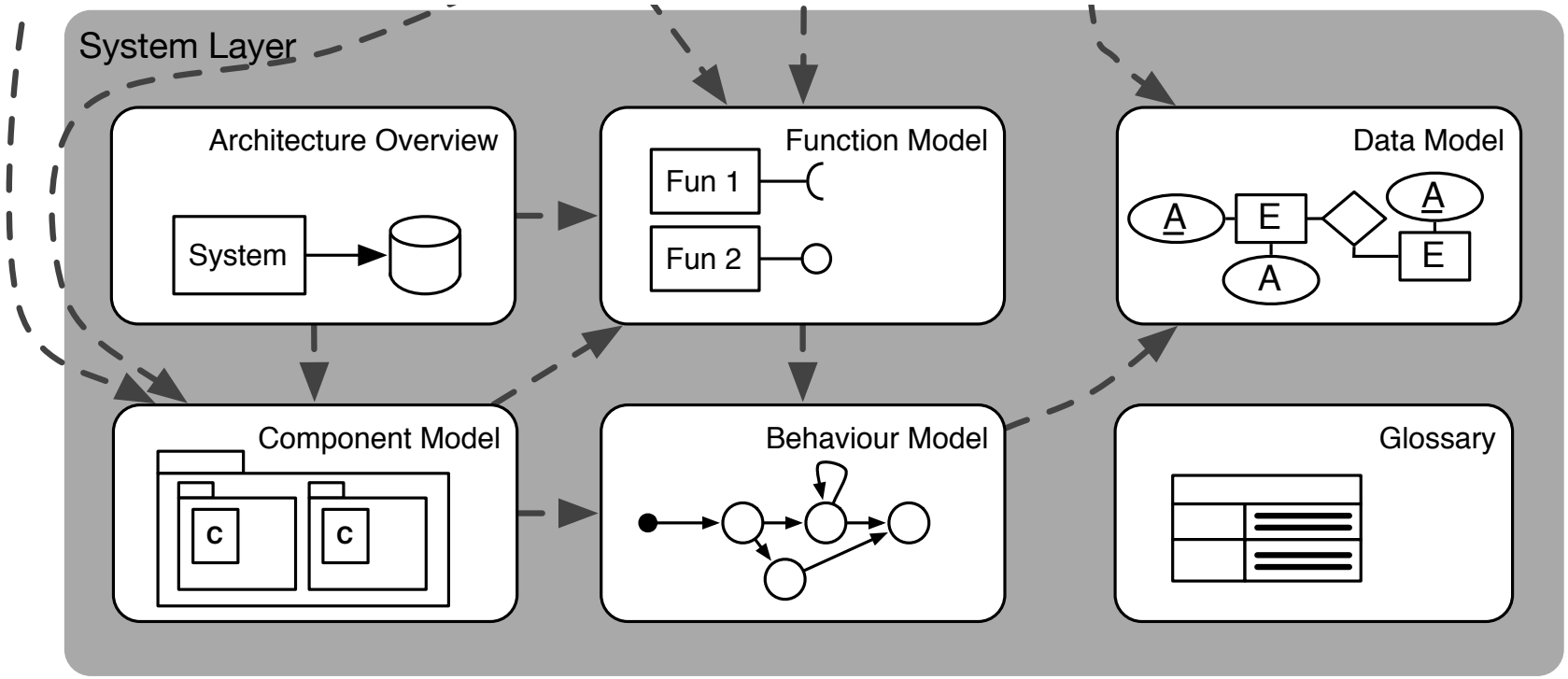
Frameworks: AMDiRE Artefact Model – Context Layer



Frameworks: AMDiRE Artefact Model – Requirements Layer

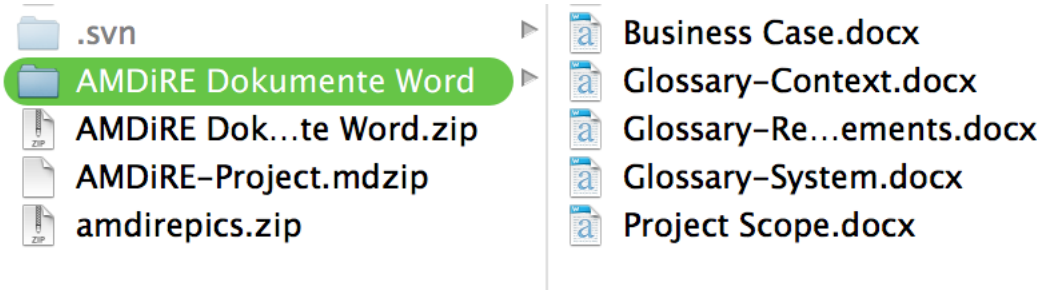


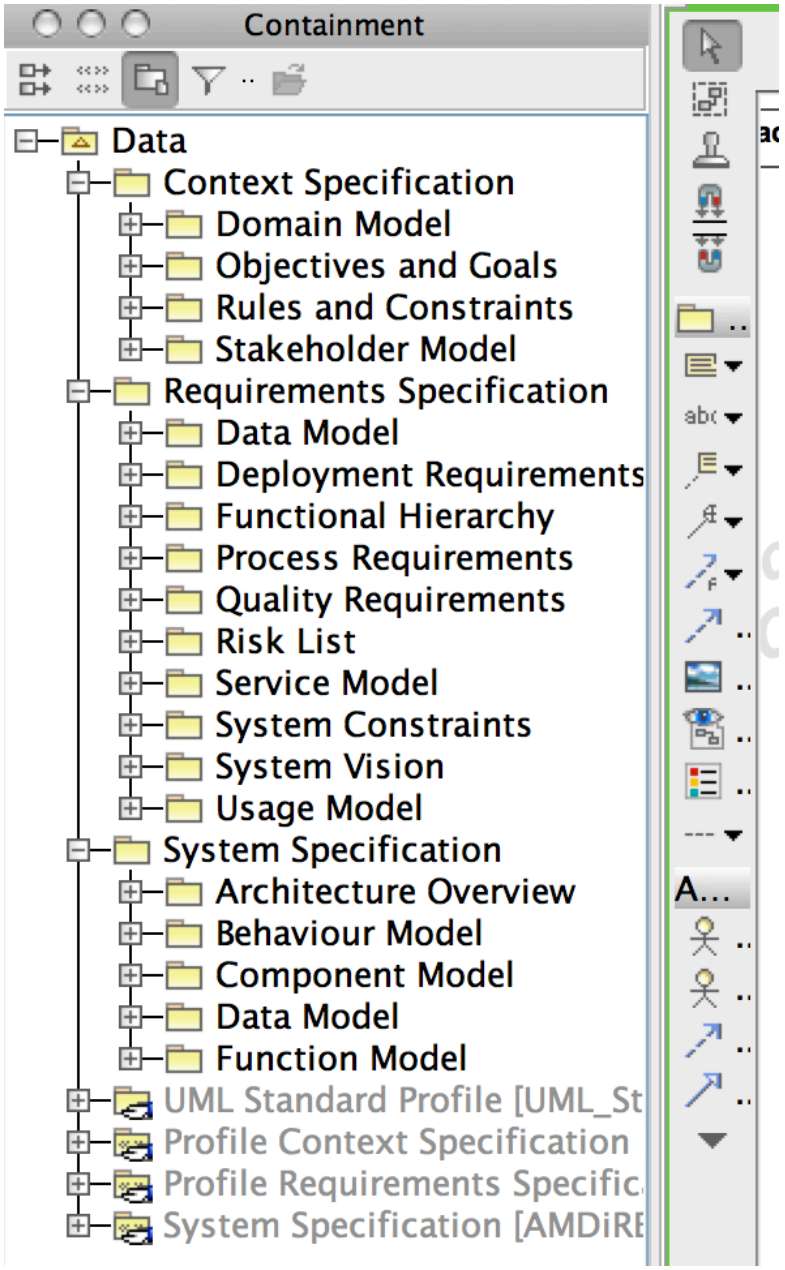
Frameworks: AMDiRE Artefact Model – System Layer

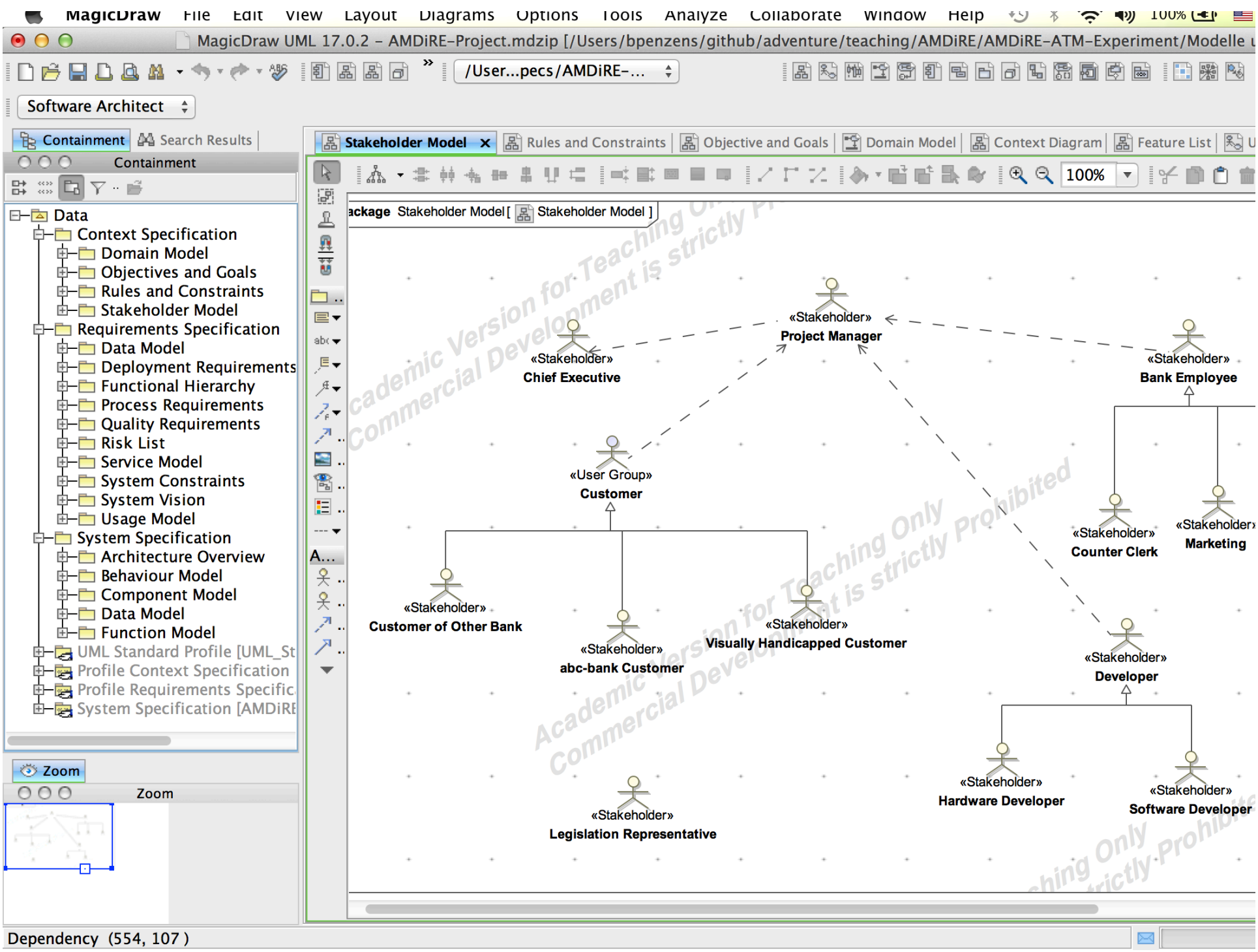


AMDiRE Artefact Model – in real life

Specifications in text and various types of diagrams






MagicDraw File Edit View Layout Diagrams Options Tools Analyze Collaborate Window Help
MagicDraw UML 17.0.2 - AMDiRE-Project.mdzip [/Users/bpenzens/github/adventure/teaching/AMDiRE/AMDiRE-ATM-Experiment/Modelle
Software Architect
Containment
Search Results
Containment
Data
Context Specification
Domain Model
Objectives and Goals
Rules and Constraints
Stakeholder Model
Requirements Specification
Data Model
Deployment Requirements
Functional Hierarchy
Process Requirements
Quality Requirements
Risk List
Service Model
System Constraints
System Vision
Usage Model
System Specification
Architecture Overview
Behaviour Model
Component Model
Data Model
Function Model
UML Standard Profile [UML_St
Profile Context Specification
Profile Requirements Specific
System Specification [AMDiRE
Zoom
Zoom
Stakeholder Model
Rules and Constraints
Objective and Goals
Domain Model
Context Diagram
Feature List
100%
ackage Stakeholder Model [Stakeholder Model]
«Stakeholder»
Project Manager
«Stakeholder»
Chief Executive
«Stakeholder»
Bank Employee
«User Group»
Customer
«Stakeholder»
Counter Clerk
«Stakeholder»
Marketing
«Stakeholder»
Customer of Other Bank
«Stakeholder»
abc-bank Customer
«Stakeholder»
Visually Handicapped Customer
«Stakeholder»
Developer
«Stakeholder»
Hardware Developer
«Stakeholder»
Software Developer
«Stakeholder»
Legislation Representative
Academic Version for Teaching Only
Commercial Development is strictly Prohibited
Dependency (554, 107)

Discussion

- How can we put that into practice?
- We have the artifact model.
  What else do we need to give it to a team of requirements engineers and let them use it?

AMDiRE additional resources

Please check out the online resources:

- Cheat sheet
- Tool extension (UML Magic Draw plugin)
- Example case study (ATM)
- Journal article

http://www4.in.tum.de/~mendezfe/openspace.shtml

The Case Study – your project

You will make a reduced version of the AMDiRE model for your class project.

And because I don't have a project that I want to make you develop requirements for, I want you to decide which project you want to do them for. Write that up in one paragraph and send it to me via email (Birgit.Penzenstadler@csulb.edu) until February 16 before class:

- Who forms the team? Team up in pairs or teams of three.
- What is your system idea? Develop your vision.
- Who are your competitors? What is your added value? Research on what competitors are out there (systems closest to what you are trying to develop), and point out how you differ from them.

The Chick's New Coat

- Once, there was a young chick. Freshly hatched out of the egg in the chilly April, she longed for a nice coat. So she went to see Rabbit, the tailor...
- Disappointed and still cold, Chick went to see Turtle, the other tailor in town…
- ***Moral: …?***

Special thanks to Vicky Sauter