Requirements Engineering: How to deal with Quality Requirements

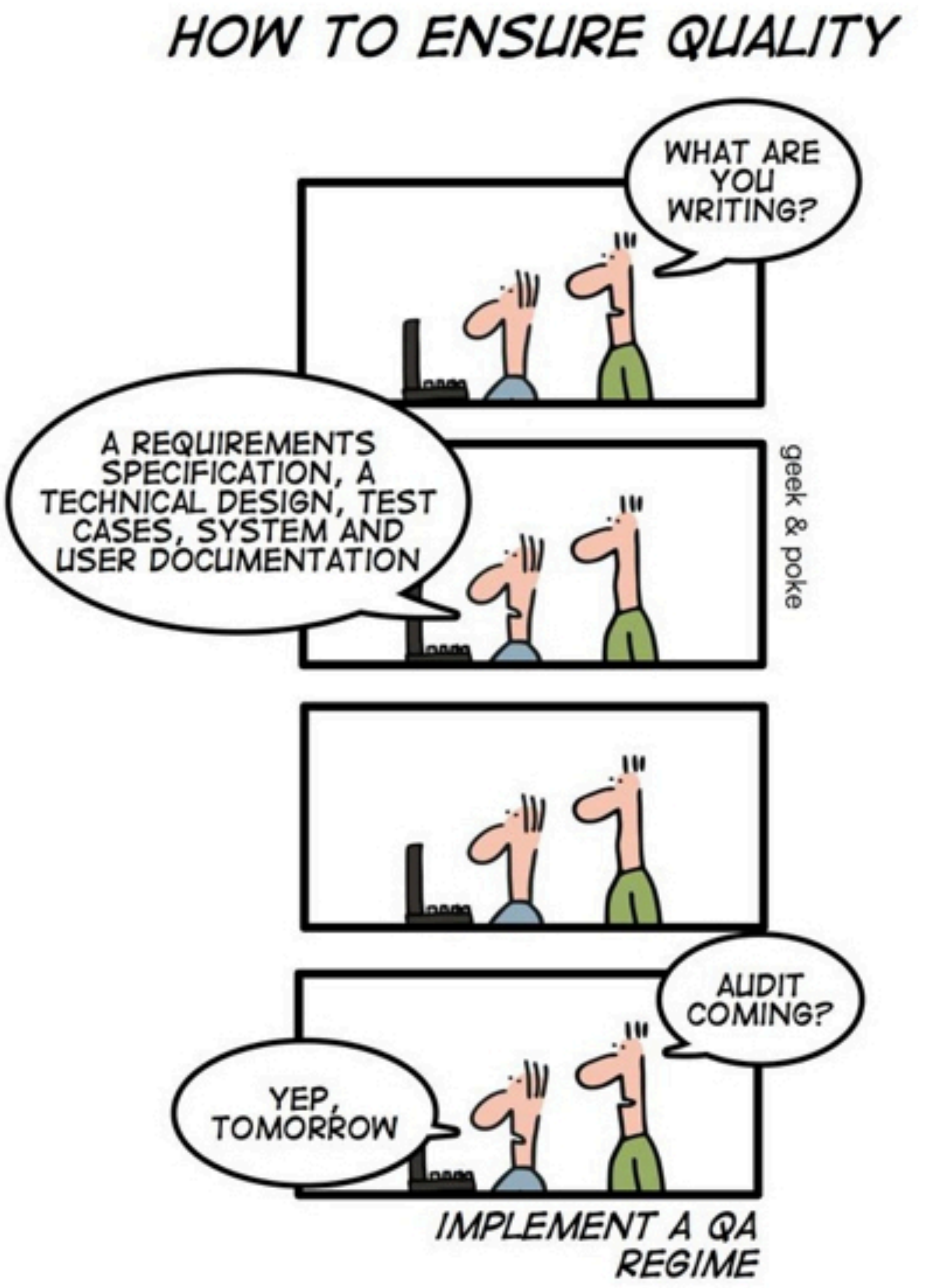
CECS 590

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?

Recap time! Quality Models

- What are quality models?
- What examples did we look at?
- What do we do with them?



Quality models and dealing with NFRs

- Usage of Quality models in RE
- Exemplary quality models
- Dealing with NFRs in AMDiRE

K Rayker, stock.xchng

Philosophy

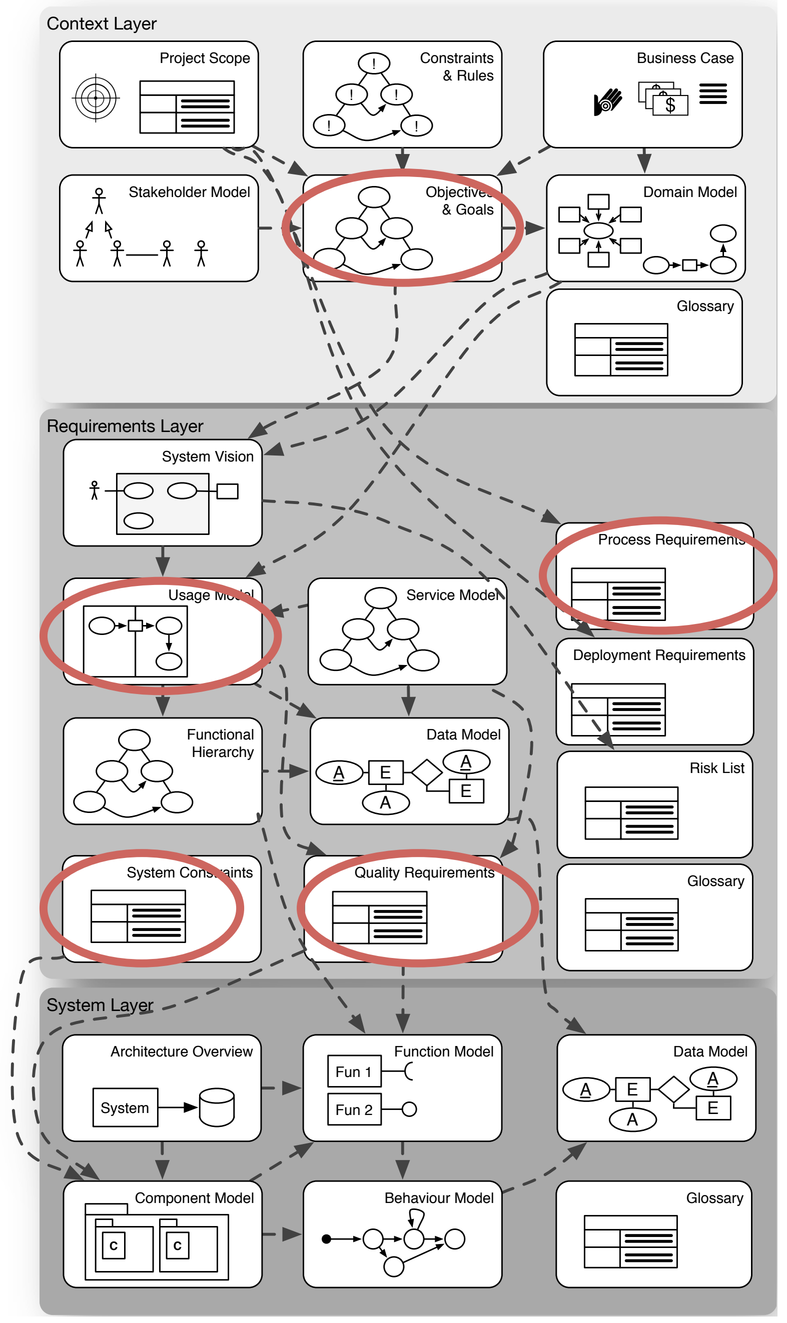
- AMDiRE concept model based on system model and quality model
- Behavior models are in the center for functional requirements and quality requirements

Classification of NFRs

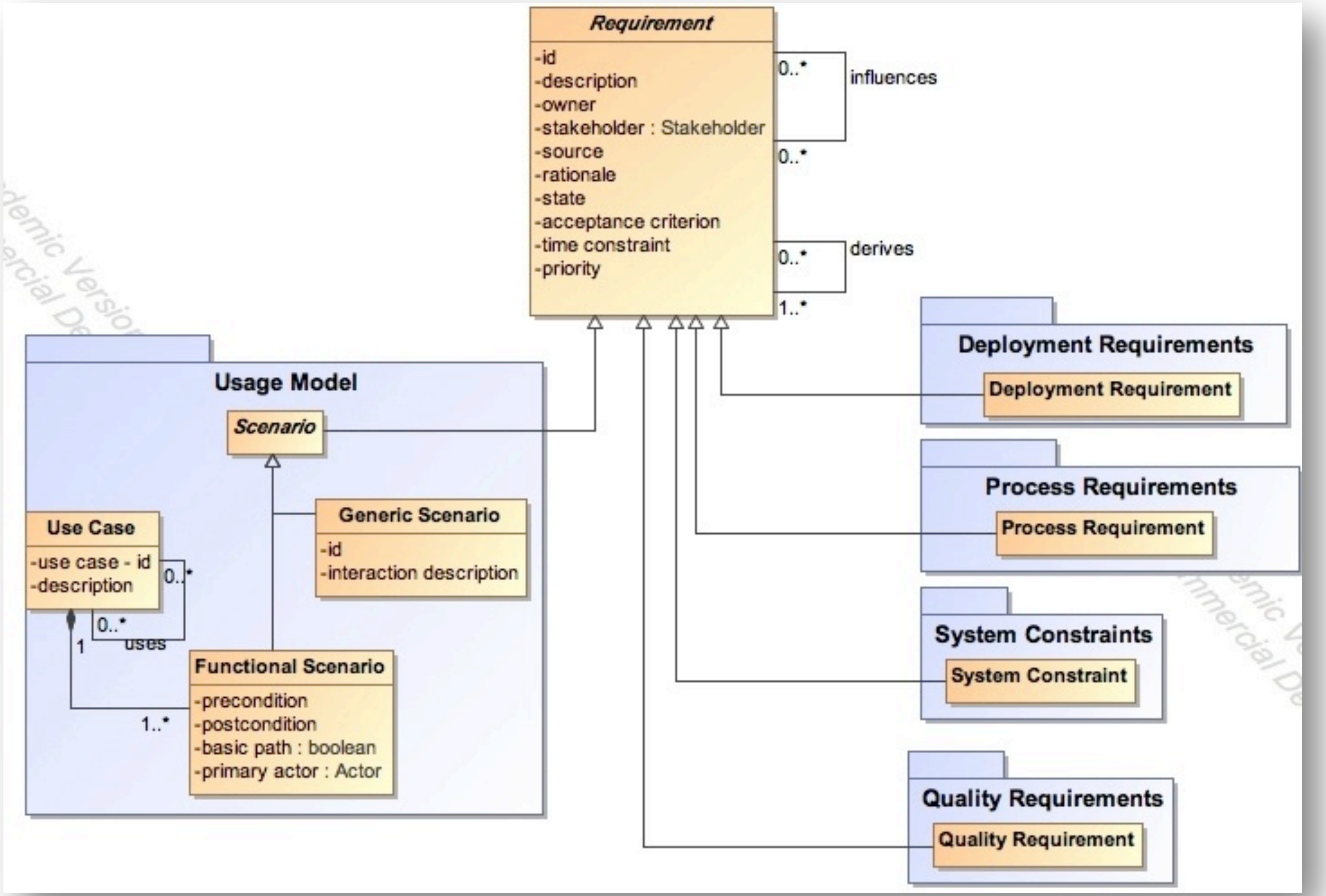
- Process Requirements
- Deployment Requirements
- System Constraints
- Quality Requirements

Structured elicitation of quality requirements

- From system goals
- To scenarios
- To quality requirements



Classification in AMDiRE (Excerpt)



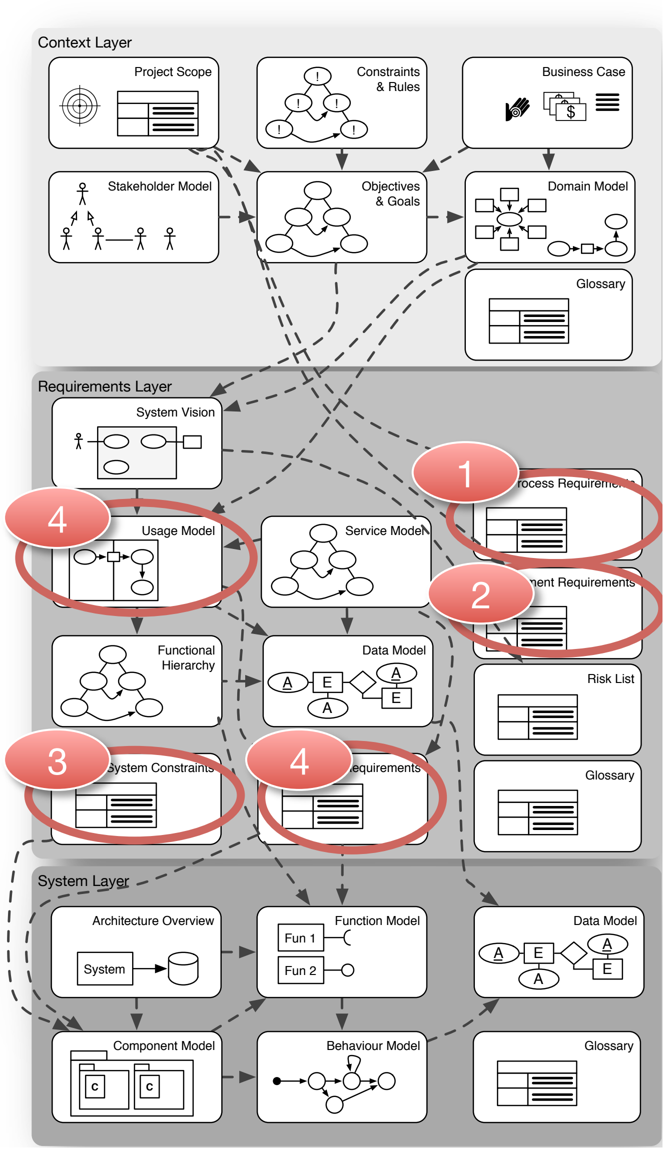
Overview of relevant Content Items

1. Process Requirements: Required characteristics of the process/ project
   e.g.: Use RUP as process model

2. Deployment Requirements: Demands for deployment
   e.g.: strategy to be followed for data migration

3. System Constraints: System-related restrictions that don‘t necessarily results from functional goal.
   E.g.: usage of specific technologies

4. Quality Requirements: desired quality characteristics of the system
   examples following



Quality principle in AMDiRE

Quality requirements across 3 levels of abstraction

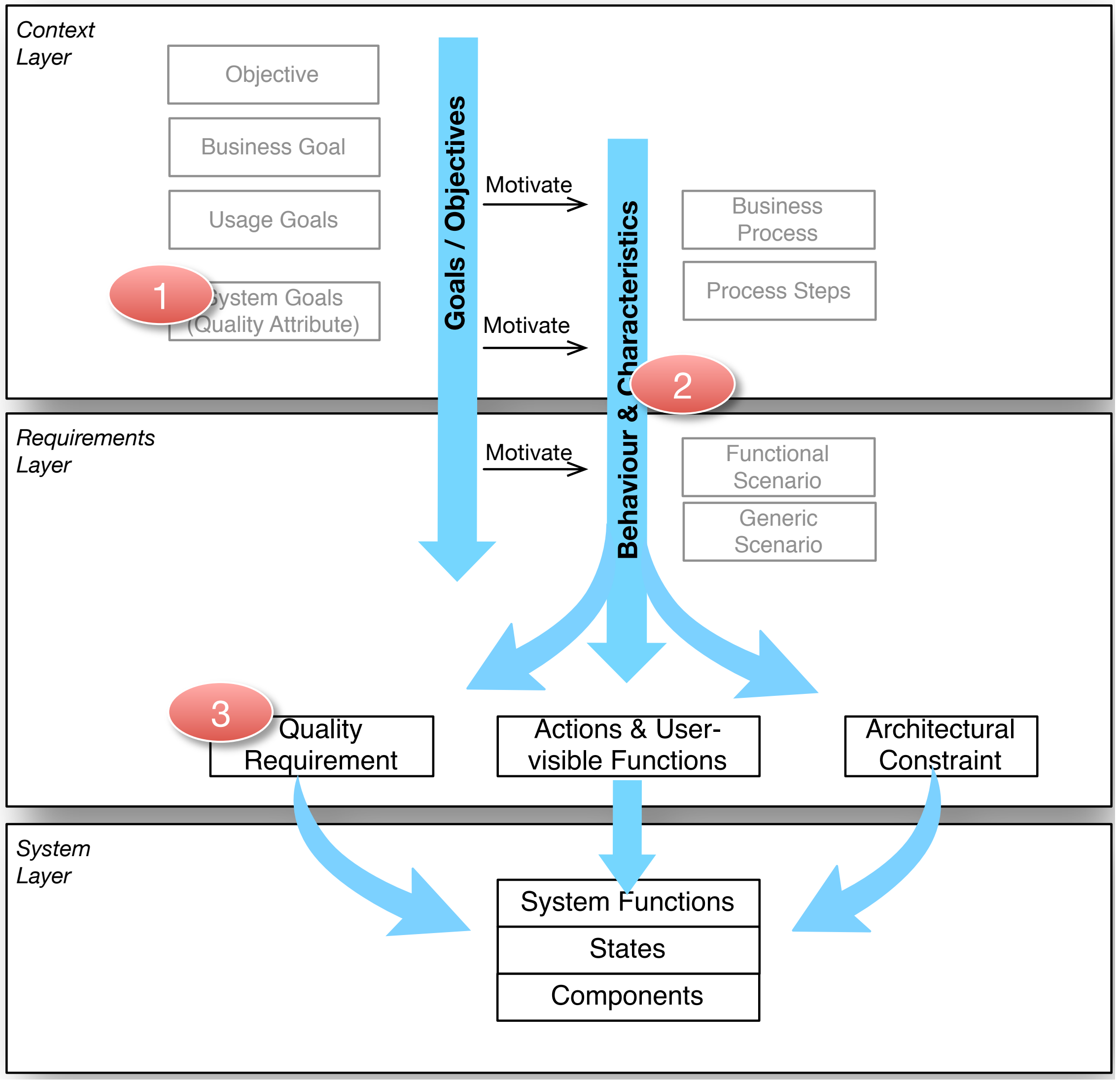
1. Goals: declaration of intent
2. Usage Model: definition of interaction scenarios that
   - Shall eb supported (e.g. maintenance).
   - Shall be avoided (e.g. hacker attack).

   → Compare goal/anti-goal

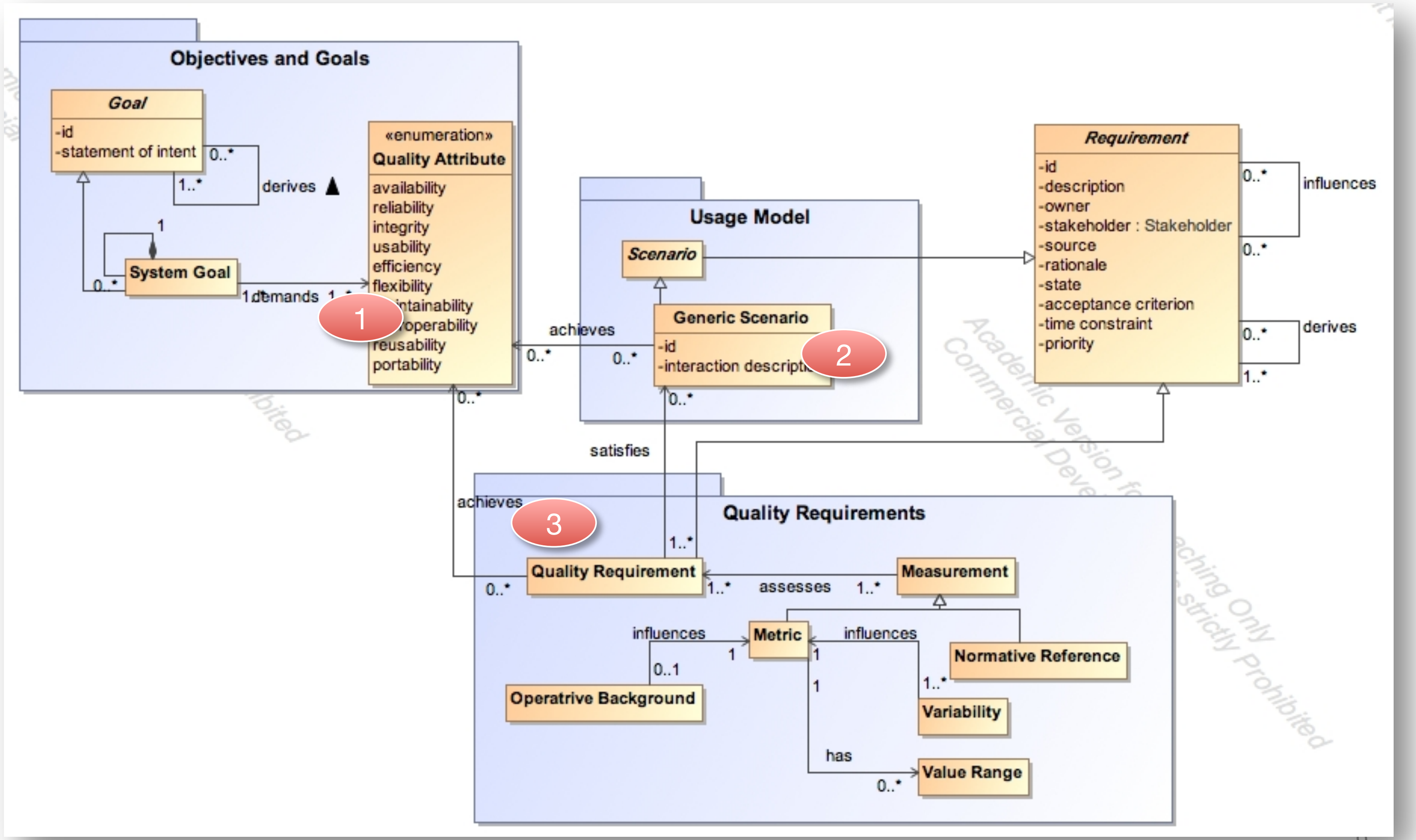
→ Allows for stepwise refinement of quality requirements

→ Allows to make abstract requirements measurable

3. Quality Requirements: measurable/quantified quality requirements



Quality principle in AMDiRE

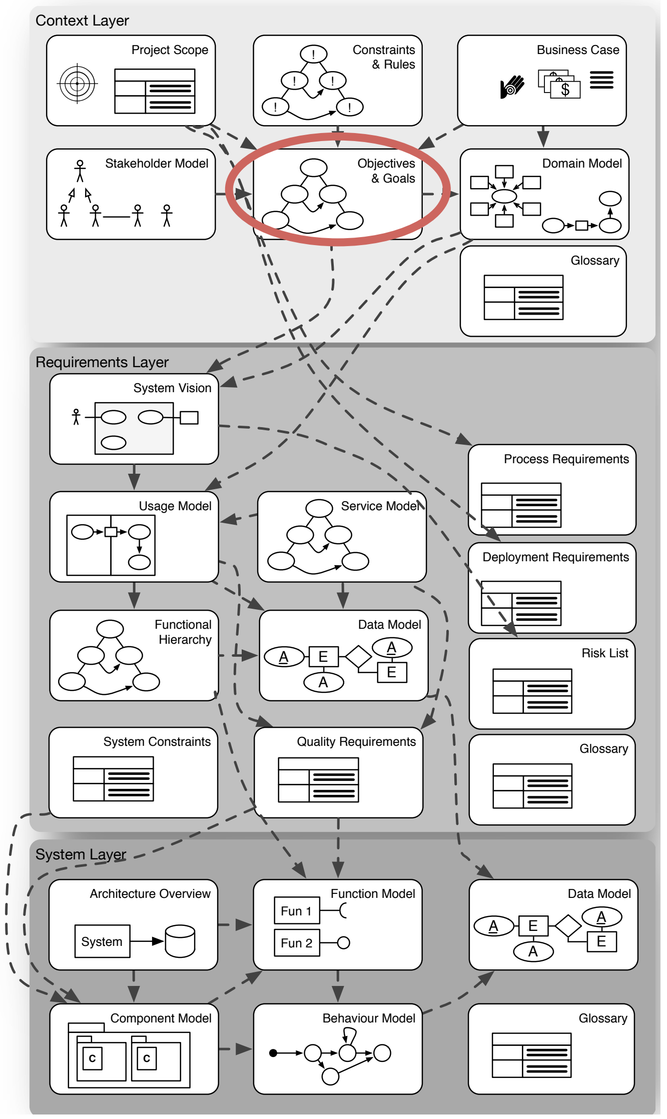


Quality requirements – goal modeling

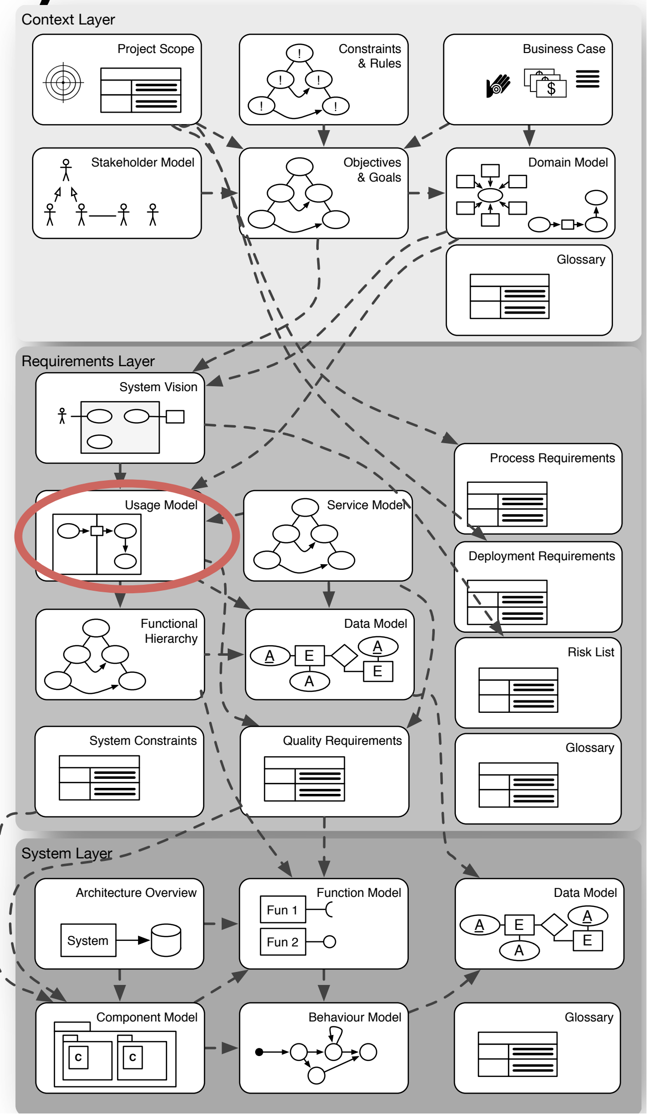
Goal modeling recap:

- Goals are declarations of intent
- Usage of goals as rationale for requirements
- System Goals: System-related goals that target system characteristics, e.g.
  - Maintainability
  - Usability
  - ...

→ Usage of goals as basis for behavioral characteristics/ quality requirements



Quality requirements – Generic Scenarios (1/3)



Usage Model recap:

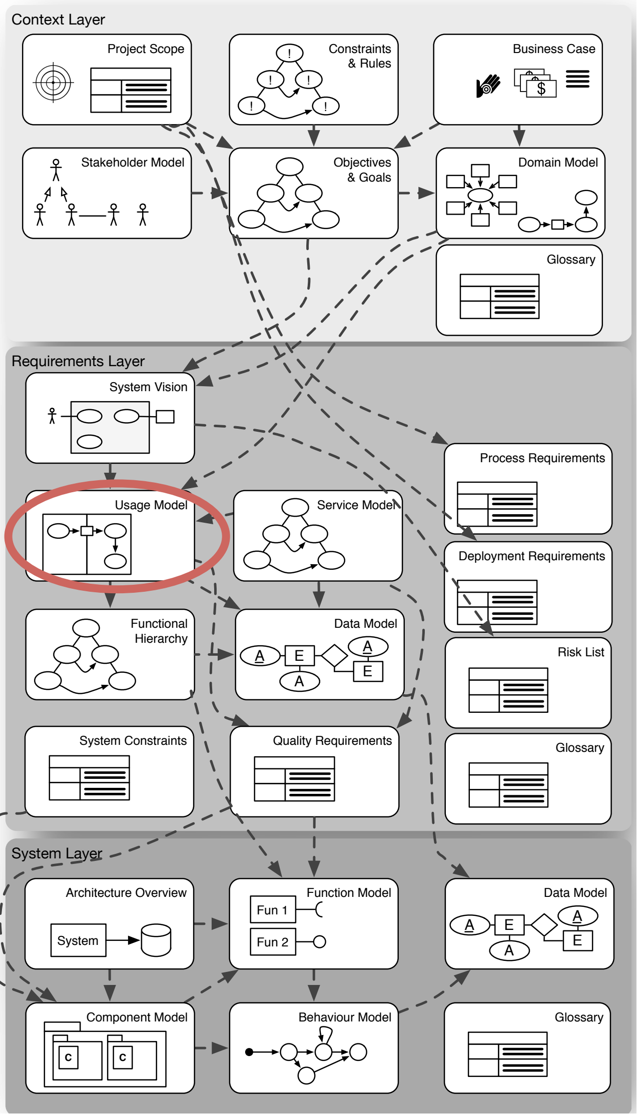
- Interaction scnearios for modeling usage of the system by external actors (user, external system).
- Differentiation:
  - Use Cases for modeling usage and external functional behavior, e.g. the system-user interaction for a business processes
  - Generic Scenarios for modeling of *quality characteristics* that are perceivable in the external behavior

Generic Scenarios (simplified)

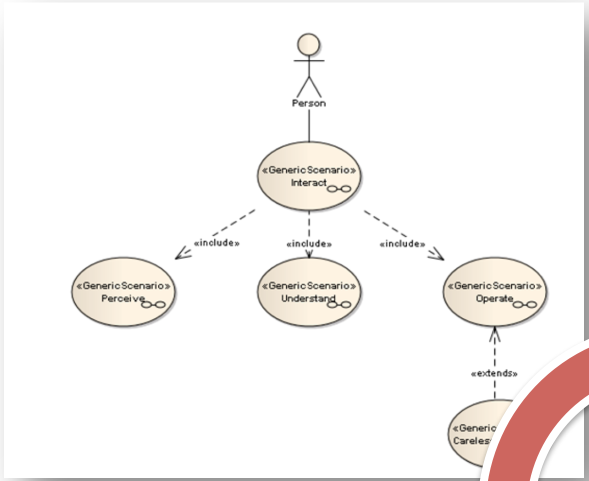
- Which non-fctl. motivated characteristics shall be *enabled*?
  E.g.: Which maintenance activities shall be enabled?
- Which non-ftcl. characteristics shall be *avoided*?
  E.g.: Which hacker attacks shall be avoided?

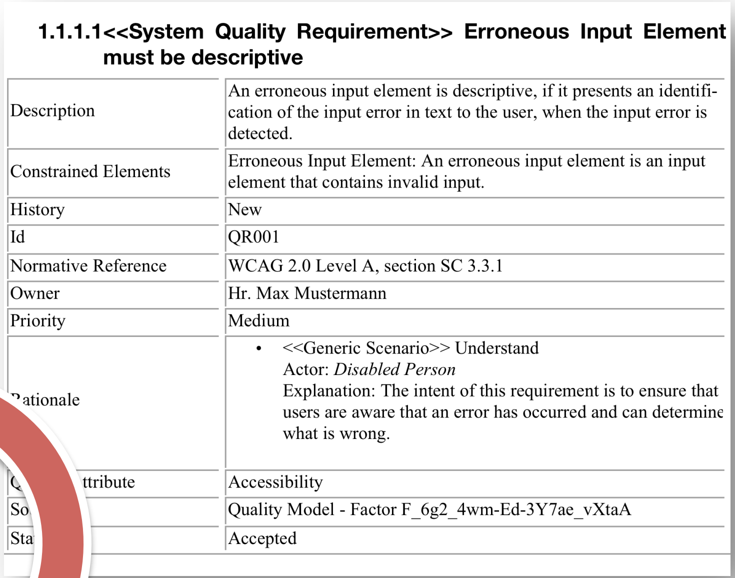
Quality requirements – Generic Scenarios (2/3)

- Allows to reduce quality to (non-functionally motivated) activities that shall be *enabled* or *avoided*.
- Generic scenarios support structured…
  - Elicitation of quality characteristics without having to immediately quantify
    - → *„System shall be easy to maintain“ (goal)*
    - → *„Which maintenance activities shall be supported?“ (Generic Scenario)*
  - Assessment of quality characteristics w.r.t. costs
    - → *Execution of activities can be related to costs.*
  - Evaluation whether non-fctl. Characteristics have been implemented
    - → *Usage scenarios can be tested.*

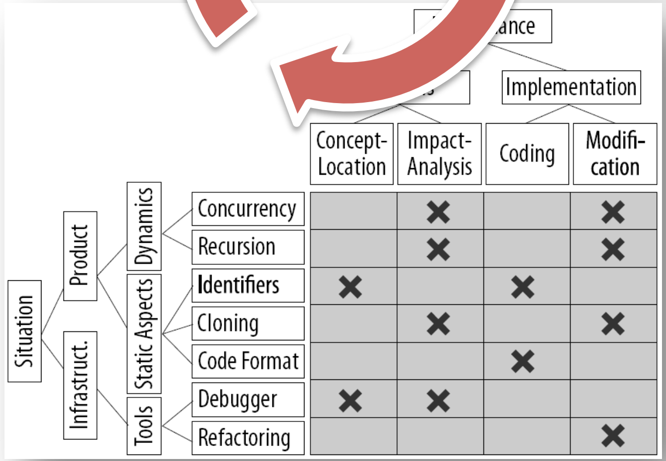


Quality requirements – Generic Scenarios (3/3)


Person
«GenericScenario» Interact
«include»
«include»
«include»
«GenericScenario» Perceive
«GenericScenario» Understand
«GenericScenario» Operate
«extends»
«Generic
Careles


1.1.1.1<<System Quality Requirement>> Erroneous Input Element must be descriptive

Description	An erroneous input element is descriptive, if it presents an identification of the input error in text to the user, when the input error is detected.
Constrained Elements	Erroneous Input Element: An erroneous input element is an input element that contains invalid input.
History	New
Id	QR001
Normative Reference	WCAG 2.0 Level A, section SC 3.3.1
Owner	Hr. Max Mustermann
Priority	Medium
Rationale	• <<Generic Scenario>> Understand Actor: *Disabled Person* Explanation: The intent of this requirement is to ensure that users are aware that an error has occurred and can determine what is wrong.
Q ttribute	Accessibility
So	Quality Model - Factor F_6g2_4wm-Ed-3Y7ae_vXtaA
Sta	Accepted



				ance			
				s		Implementation	
				Concept-Location	Impact-Analysis	Coding	Modifi-cation
Situation	Product	Dynamics	Concurrency		✖		✖
			Recursion		✖		✖
		Static Aspects	Identifiers	✖		✖	
			Cloning		✖		✖
			Code Format			✖	
	Infrastruct.	Tools	Debugger	✖	✖		
			Refactoring				✖

- System characteristic (positively/ negatively) influences system usage

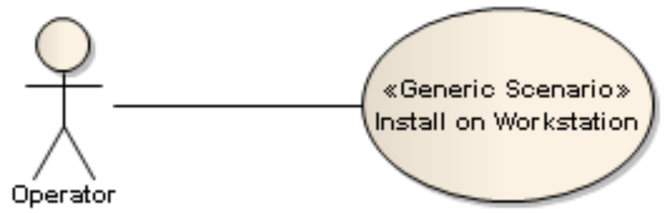
→ Use for structured eicitation of scenarios

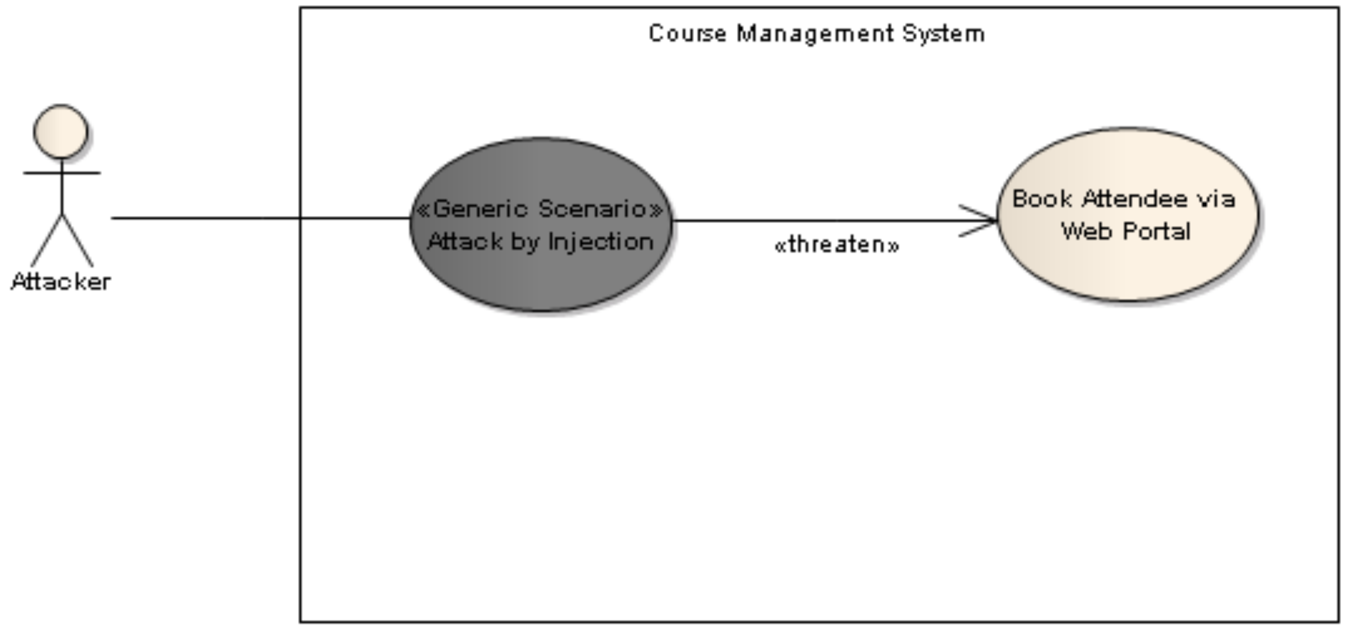
→ Use for deducing demanded system characteristics

Example Generic Scenario

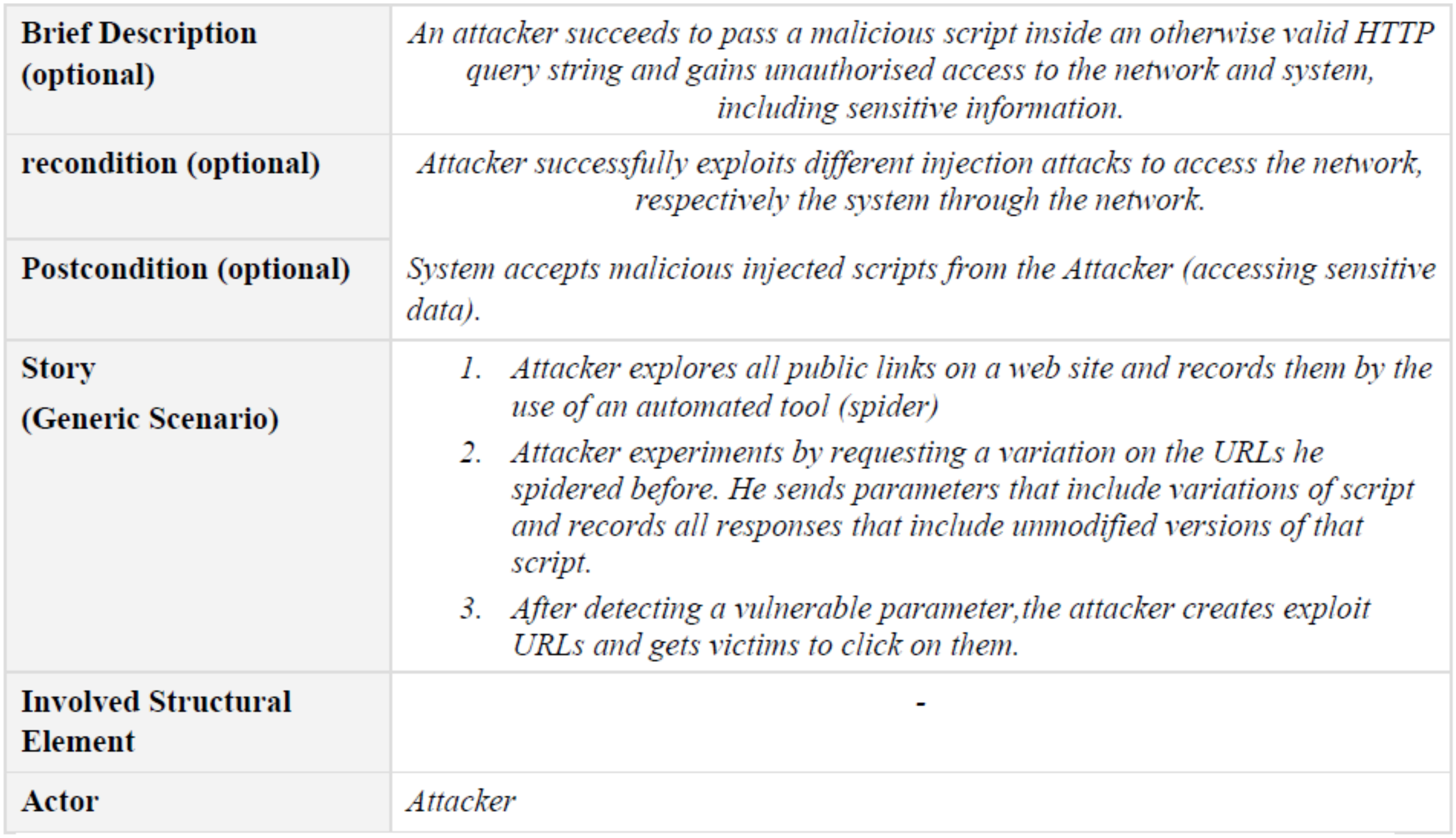
- Use the Cockburn template
- Specify interaction that is exemplary of how the system should behave for a quality characteristic tied to a functionality
- For example:
  - Interaction of the ATM with a visually impaired user
  - Alpine Adventure Tours example

Generic Scenarios: AAT Example





<<GenericScenario>> Attack by Injection (AAT Example)



Brief Description (optional)	*An attacker succeeds to pass a malicious script inside an otherwise valid HTTP query string and gains unauthorised access to the network and system, including sensitive information.*
recondition (optional)	*Attacker successfully exploits different injection attacks to access the network, respectively the system through the network.*
Postcondition (optional)	*System accepts malicious injected scripts from the Attacker (accessing sensitive data).*
Story (Generic Scenario)	1. *Attacker explores all public links on a web site and records them by the use of an automated tool (spider)* 2. *Attacker experiments by requesting a variation on the URLs he spidered before. He sends parameters that include variations of script and records all responses that include unmodified versions of that script.* 3. *After detecting a vulnerable parameter,the attacker creates exploit URLs and gets victims to click on them.*
Involved Structural Element	-
Actor	*Attacker*

Exercise

- Which technique for quality requirements do you find more helpful?
- Refine one ATM quality goal
  1. as textual quality requirement with metric and evaluation and
  2. as generic scenario.

Evaluation with regard to challenges

1. Crosscutting Concerns

→ Interdependency with behavioral models clearly defined

2. Classification and structuring

→ Structured in Content Items

3. Elicitation, assessment and evaluation

→ Refinement of NFRs over 3 abstraction levels

→ Assessment of NFRs:

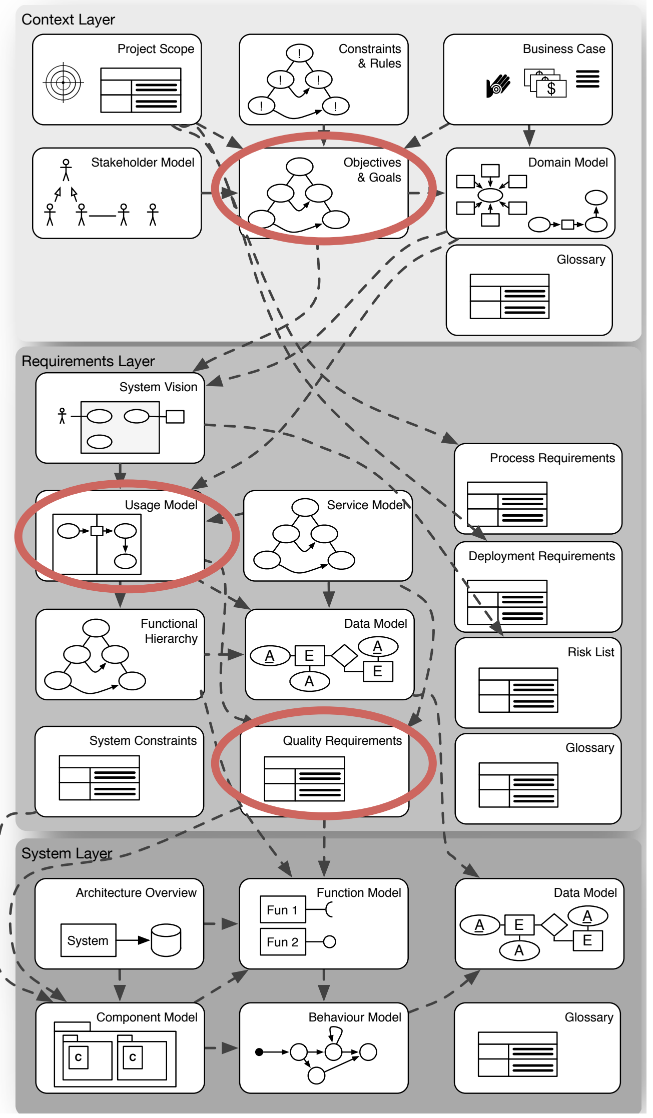
  – Interdependency with other requirements modeled in behavioral models

  – Implications for implementation/costs representable

→ Specification of NFRs can be validated

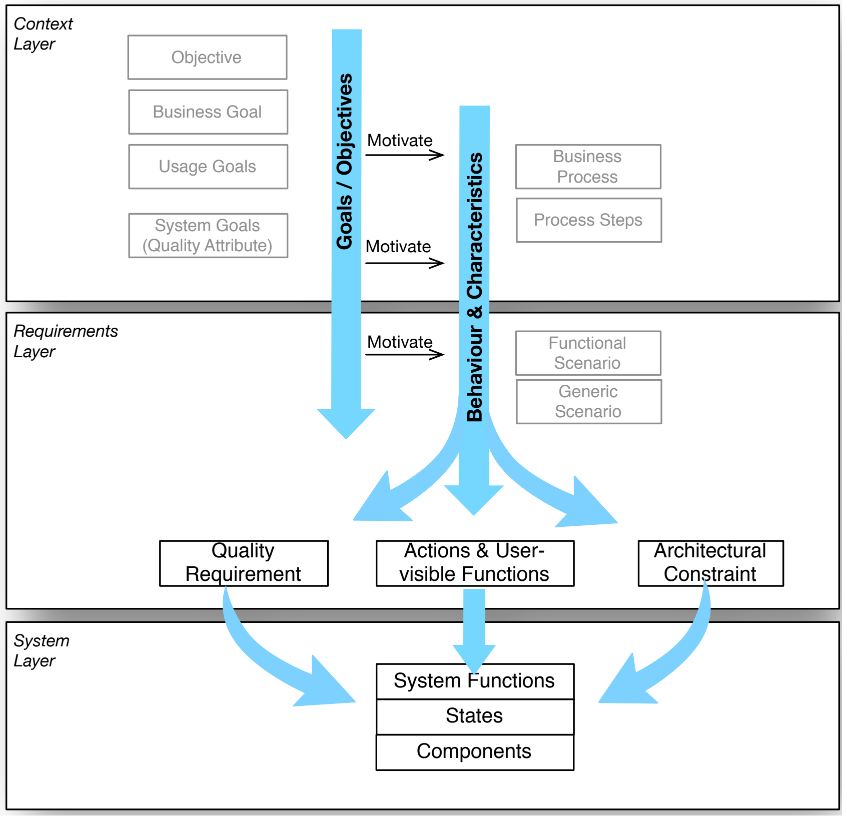
Critical assessment

- What are the costs and what are the benefits?
- Application of quality models can be cost-intensive (time-consuming)



Summary

- **Quality** and **Non-functional requirements**
  - Are important for RE
  - Are difficult to elicit, assess, and evaluate
- Application of quality models
  - Defines terminology
  - Supports definition of modeling concepts
  - Is basis for taxonomies


Context Layer
Objective
Business Goal
Usage Goals
System Goals (Quality Attribute)
Goals / Objectives
Motivate
Motivate
Behaviour & Characteristics
Business Process
Process Steps
Requirements Layer
Motivate
Functional Scenario
Generic Scenario
Quality Requirement
Actions & User-visible Functions
Architectural Constraint
System Layer
System Functions
States
Components