Systems Engineering and Project Management (Modeling process)

Prof. Dr. Franz Wotawa

Institute of Software Engineering and Artificial Intelligence

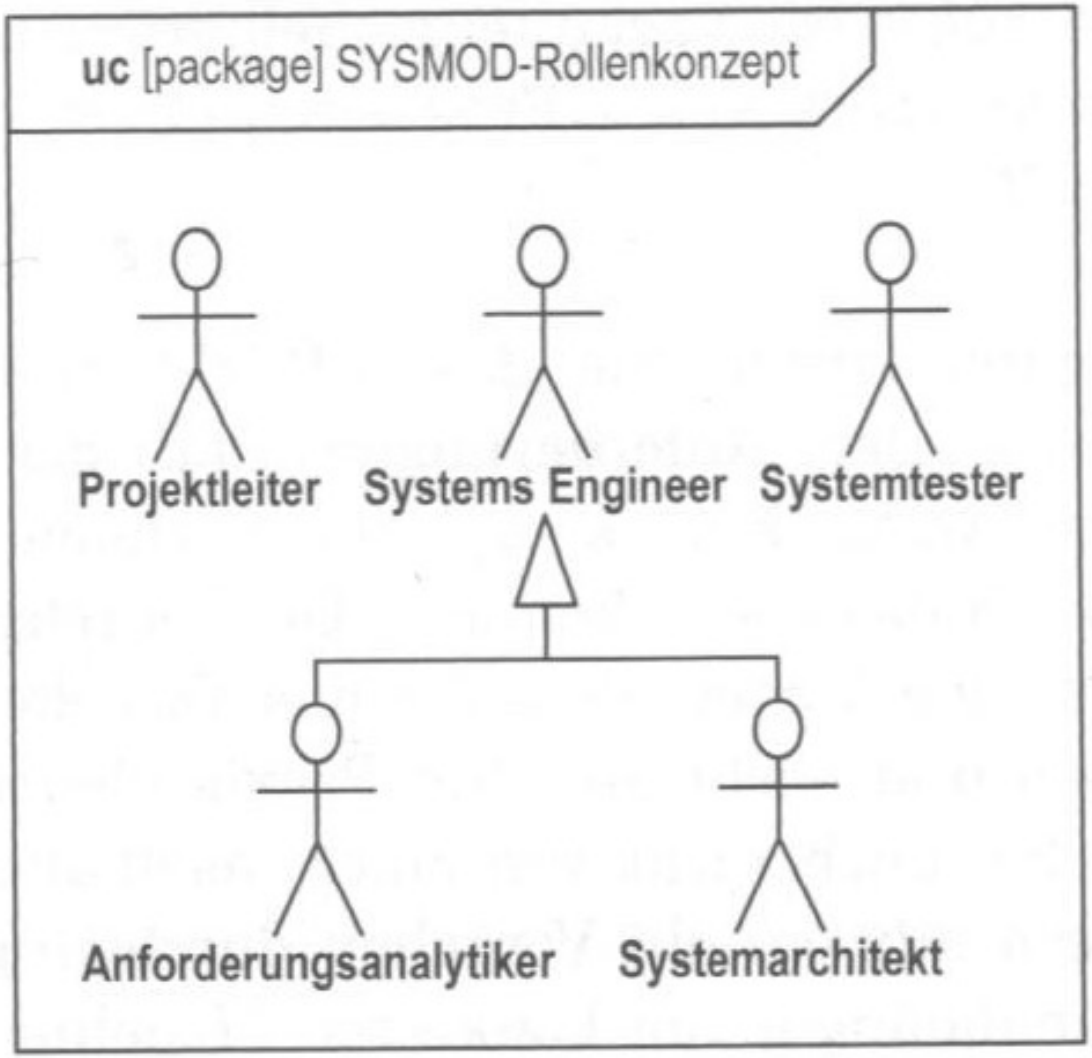
wotawa@tugraz.at

Modeling process

- SYSMOD from Tim Weilkiens Book: ”Systems Engineering mit SysML/UML, dpunkt.verlag GmbH, 3. Auflage, 2014”.

- Foundational idea (“Blackbox view on systems”):
  - Identify elements,
  - Describe the context (of the system),
  - Further develop the internals (of the system)

SYSMOD Role Concepts



SYSMOD Role Concepts

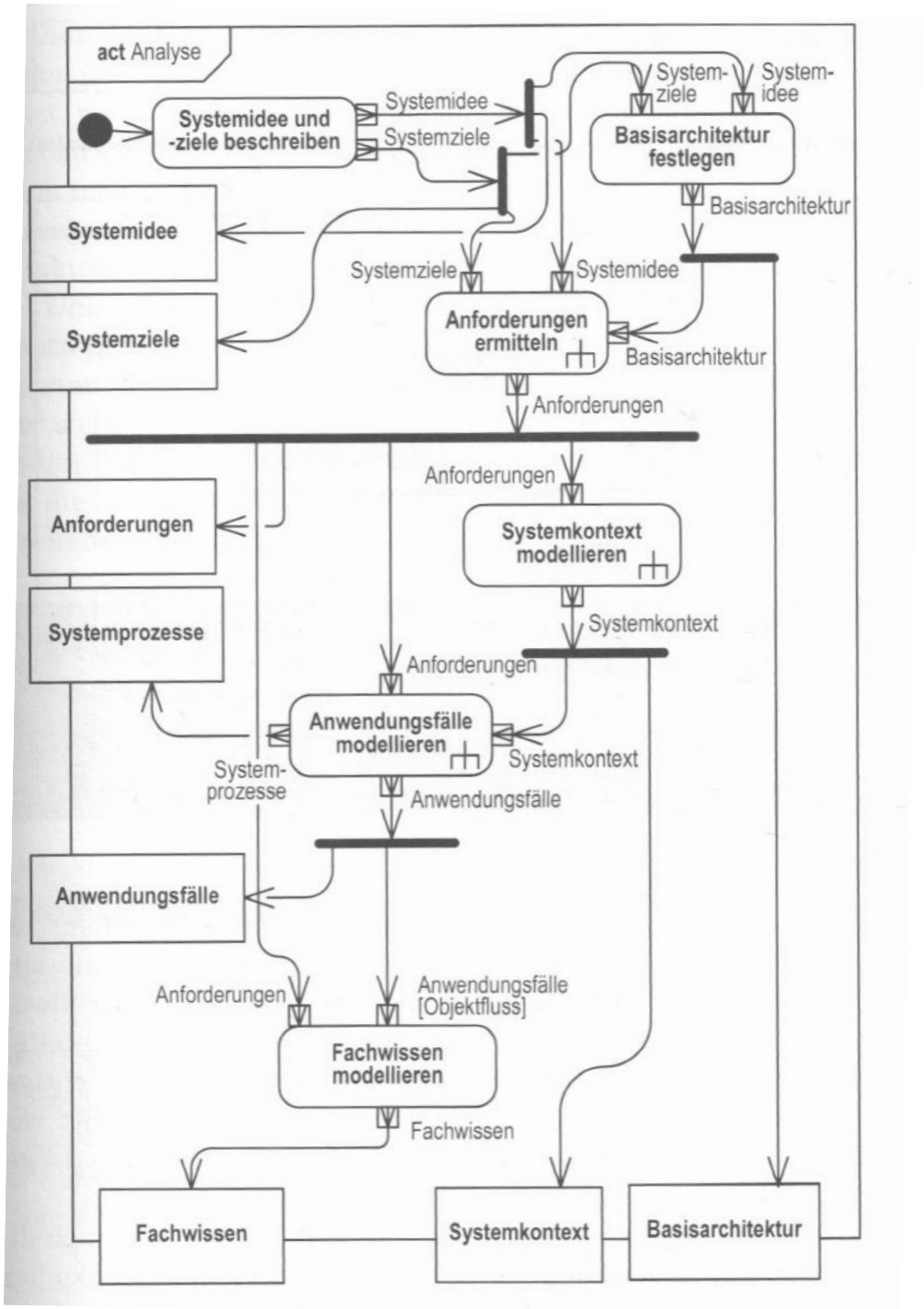
- **Systems engineer**
  - Requirements analyst: Obtains and maintains the system requirements.
  - System architect: Derives the solution architecture from the requirements
- **Project leader**
- **System tester**

SYSMOD - Overview

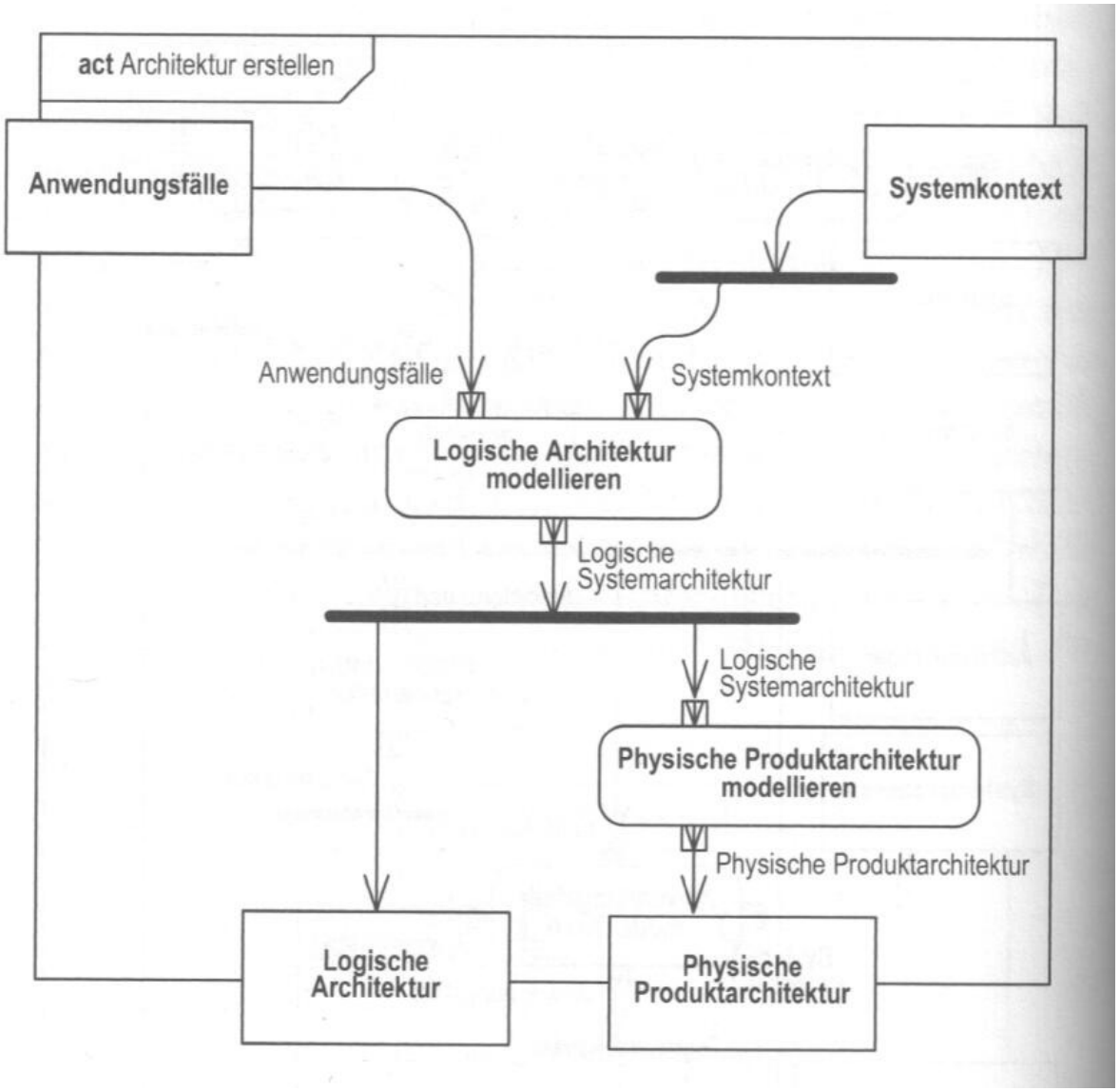
- **Tasks**:
  1. Describe the system's ideas and goals.
  2. Find and fix the base architecture.
  3. Obtain the requirements.
  4. Model the system context.
  5. Model the Use Cases.
  6. Model the Domain Knowledge.

- Attention: Apply the tasks iteratively (and NOT sequentially)!

SYSMOD - Overview



SYSMOD Architecture Process



Documents

- [Project diary: Collection of notes obtained during the project, including the protocols.]

- Meeting protocols (formal document):
  - Title
  - Place and date
  - List of participants
  - Description of schedule, content, events, and agreed decisions/results.

Start of a project is always a system idea!

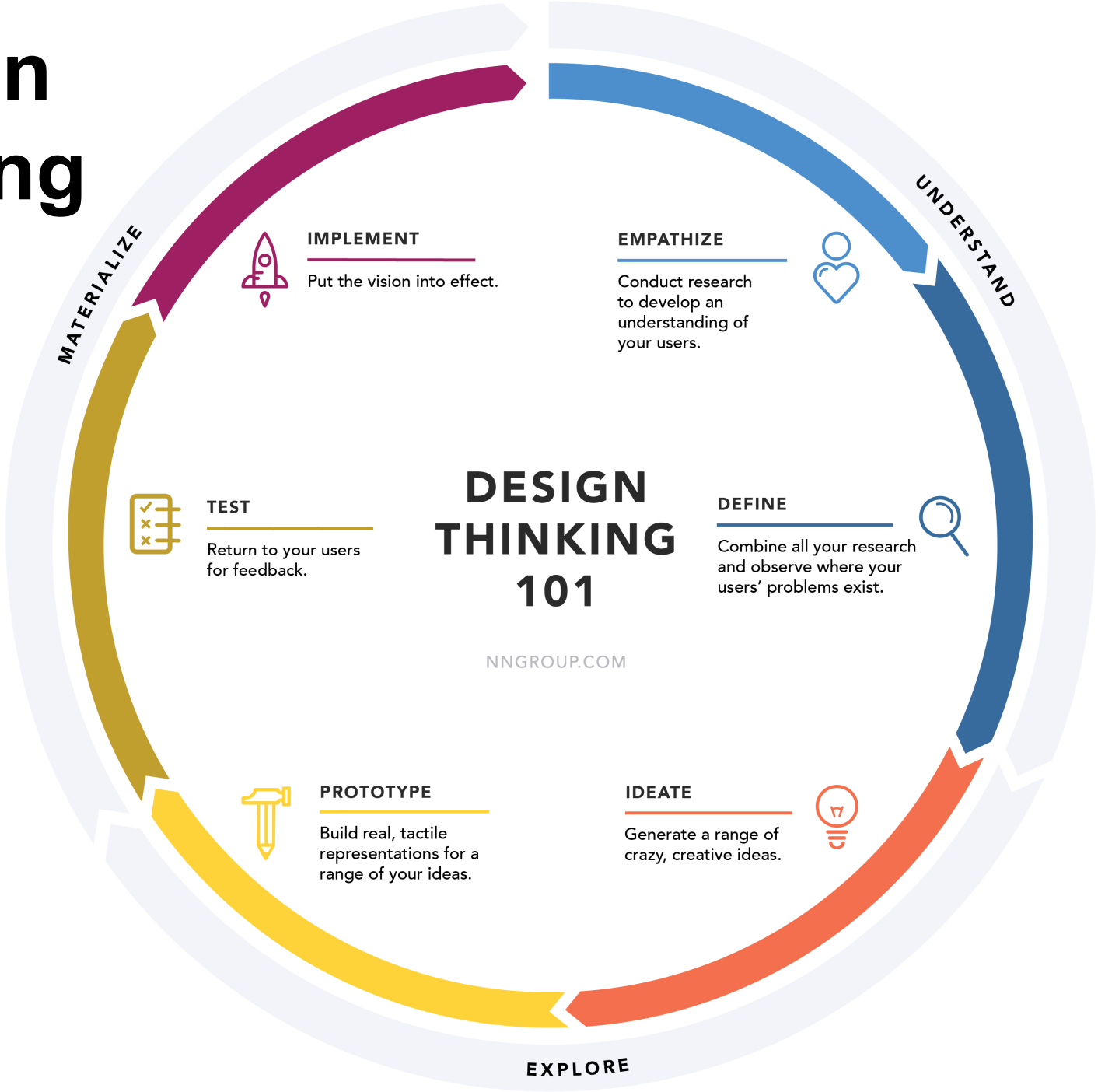
- Description of the system idea/concept
- What innovations are needed?
- Innovations are not inherently beneficial.
- Requirements can hinder innovation
- Requirements/ideas should not preempt the solution!
- In systems engineering, several potential solutions should be developed and compared.

System idea/concept

- Search for the ideal system
- Focus on simplicity!

- Method for finding innovative solutions: Design Thinking!
  - Disruptive innovations as a goal
  - Focus on human needs
  - Iterative process

Design Thinking



SYSMOD Process in Detail

- Description of the individual phases using a profile

(1) Describe the system idea and goals

- **Incoming and outgoing data:**
  - System concept
  - System objectives
- **Responsible role:**
  - Project manager
- **Roles involved:**
  - Requirements analyst

(1) Describe the system idea and goals

- **Motivation/Description:**
  - **Why?** All participants must be familiar with the system concept and objectives.
  - **What?** Describe the concept and objectives of the system in a concise and readable form.
  - **How?** The information is developed and documented in workshops.
  - **Where?** The system concept and objectives form the basis for all subsequent steps!

(1) Describe the system idea and goals

- **Key questions:**
  - How can the system be presented in 5 minutes?
  - What are the three most important goals of the system?
  - Are all project participants informed about the goals?
  - What goals does the project not pursue?
- **Model elements:**
  - **SysML:** Requirements diagram
  - **SYSMOD:** Goal

(2) Define basic architecture

- **Incoming and outgoing data:**
  - System concept
  - System objectives
  - Basic architecture
- **Responsible role:**
  - System architect
- **Roles involved:**
  - Requirements analyst

(2) Define basic architecture

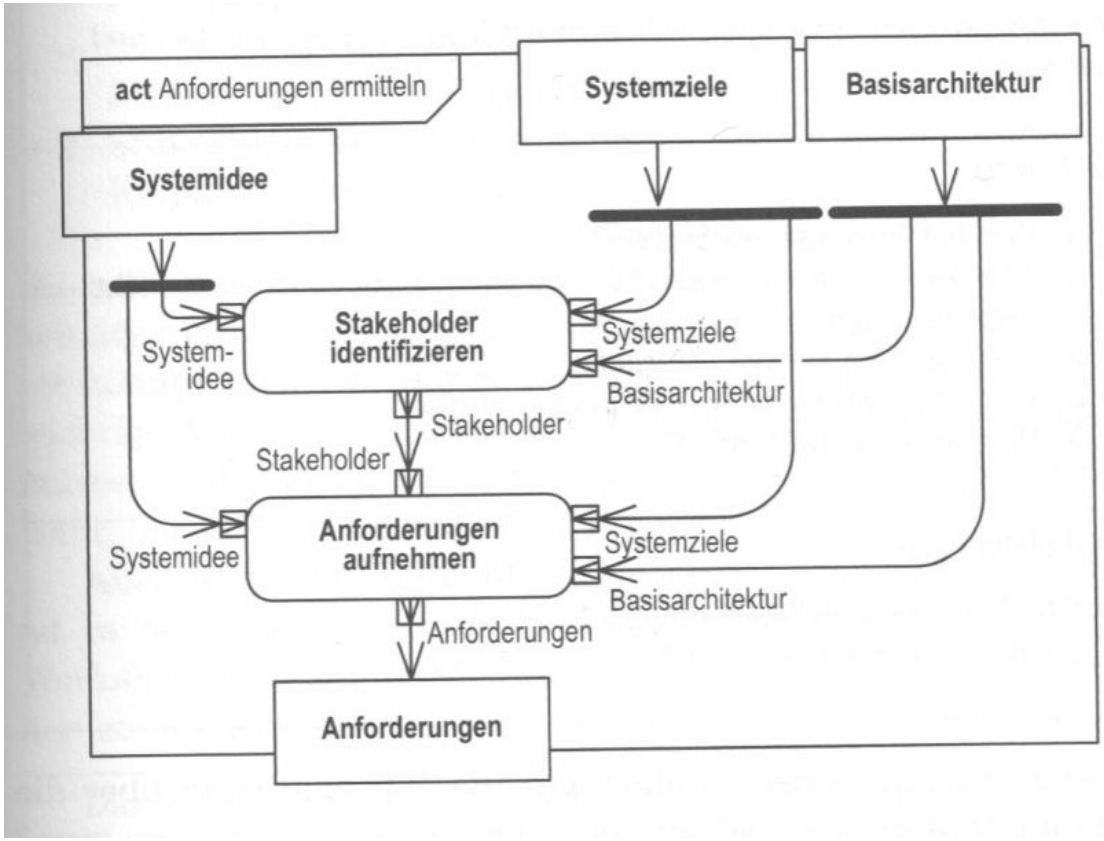
- **Motivation/Description:**
  - **Why?** The basic architecture specifies the level of abstraction of the requirements and explicitly separates the requirements from the technical solution.
  - **What?** Model and describe the architectural decisions that are specified as framework conditions.
  - **How?** The basic architecture is described using block diagrams, sketches, and textual explanations.
  - **Where?** Basis for project implementation and can be reused for other system developments.

(2) Define basic architecture

- **Key questions:**
  - Which technical components are specified?
  - How high is the desired degree of innovation?
  - Which architectural decisions are specified?
- **Model elements:**
  - **SysML:** Block definition diagram, internal block diagram, association, generalization, role, connector, port
  - **SYSMOD:** System, subsystem

(3) Determine requirements

- 2 Parts:
  - Identify stakeholder
  - Obtain requirements



(3.1) Identify stakeholder

- **Incoming and outgoing data:**
  - System concept
  - System objectives
  - Basic architecture
  - Stakeholders
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - None

(3.1) Identify stakeholder

- **Motivation/description:**
  - **Why?** Stakeholder needs are crucial to the success of the project.
  - **What?** Identification of all individuals and institutions that have an interest in or requirements for the system.
  - **How?** Compile a list of stakeholders in the workshop and add to it on an ongoing basis.
  - **Where?** Stakeholders are the source of requirements. Their interests are identified and analyzed.

(3.1) Identify stakeholder

- **Key questions:**
  - Who is interested in the system?
  - What would happen if we did not take these stakeholders and their interests into account?
  - Who will use the system?
  - Who will be affected if the system fails?
  - Who will dispose of the system?
- **Model elements:**
  - **SysML:** Use case diagram
  - **SYSMOD:** Extended stakeholders

(3.2) Obtain requirements

- **Incoming and outgoing data:**
  - Stakeholders
  - Basic architecture
  - Requirements
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - None

(3.2) Obtain requirements

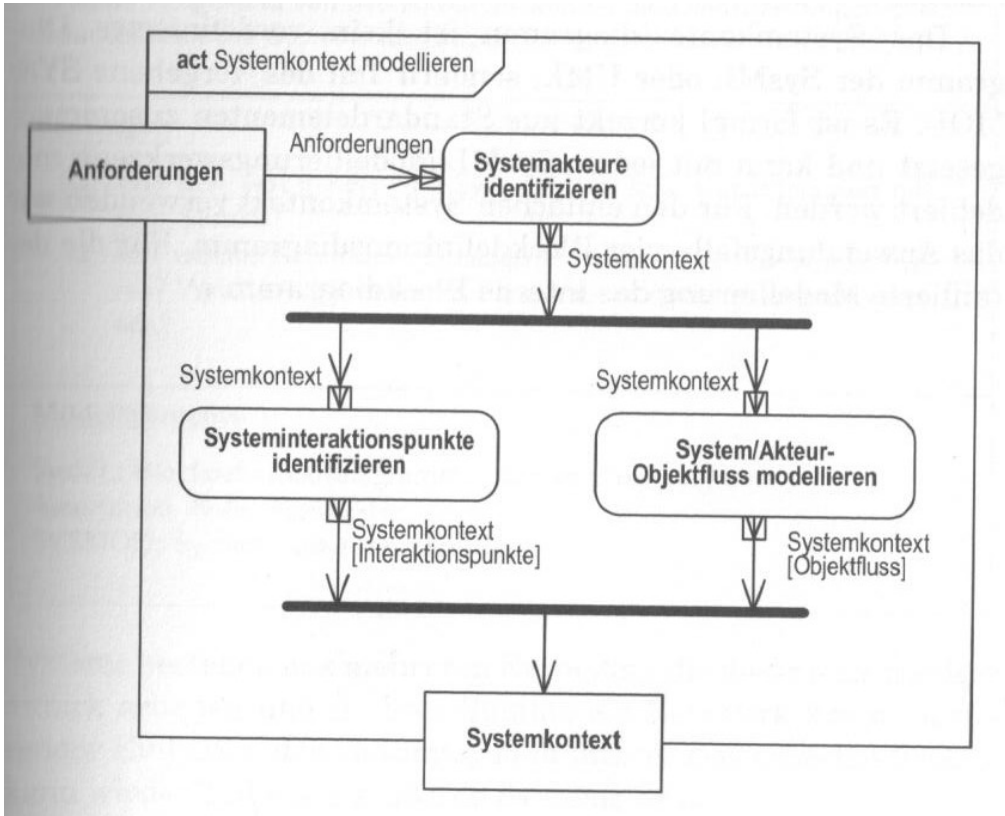
- **Motivation/Description:**
  - **Why?** Requirements are the fundamental basis of system development. Determine what the system should do.
  - **What?** Requirements are requested, documented, and structured by stakeholders.
  - **How?** Requirements are determined using survey techniques (e.g., interviews) and described using SysML.
  - **Where?** Requirements are input information for the entire system development process.

(3.2) Obtain requirements

- **Key questions:**
  - Are you asking the right people?
  - Are the answers official?
  - Are you asking the right questions?
  - Is the stakeholder requirement mandatory or optional?
  - How can you verify that the system meets the requirements?
- **Model elements:**
  - **SysML:** Use case diagram, contains relationship, refinement relationship, traceability relationship
  - **SYSMOD:** extended requirement

(4) Model System Context

- 3 Steps:
  - Identify system actors
  - Model system/actor object flow
  - Identify system interaction points



(4.1) Identify system actors

- **Incoming and outgoing data:**
  - Requirements
  - System context
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - System architect

(4.1) Identify system actors

- **Motivation/Description:**
  - **Why?** System actors are direct interaction partners for the services and interfaces being developed. Description of system boundaries.
  - **What?** All users and systems that interact with the system being developed.
  - **How?** System actors are primarily taken from the requirements and modeled in the system context diagram.
  - **Where?** Based on actors, the services and interfaces of the system are identified.

(4.1) Identify system actors

- **Key questions:**
  - Who or what belongs to the system?
  - Who or what interacts with the system?
  - Which communication partners do you want to focus on?
  - Which aspects do you want to emphasize with an actor category?
- **Model elements:**
  - **SysML:** Block definition diagram, internal block diagram, association, role, connector
  - **SYSMOD:** System, actor categories

(4.2) Model system/actor object flow

- **Incoming and outgoing data:**
  - System context
  - System context (object flow)
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - None

(4.2) Model system/actor object flow

- **Motivation/Description:**
  - **Why?** The object flow of the system with its environment is helpful for a clear understanding of the system embedding.
  - **What?** Describe the objects that the system exchanges with its environment.
  - **How?** For each actor, the relevant objects that it sends to the system or receives from the system are noted. The focus is on the object flow direction from the actor to the system.
  - **Where?** In the analysis, the object flows are used to identify and describe use cases. In the system architecture, the object flows must be reflected within the system and correspond to the system interfaces.

(4.2) Model system/actor object flow

- **Key questions:**
  - Which objects do the actors send to the system?
  - Are the objects relevant to our system in technical terms?
  - Which objects does the system send to its actors?
- **Model elements:**
  - **SysML:** Block definition diagram, internal block diagram, system module, actor, association, role, connector, object flow

(4.3) Identify system interaction points

- **Incoming and outgoing data:**
  - System context
  - System context (interaction points)
- **Responsible role:**
  - System architect
- **Roles involved:**
  - Requirements analyst

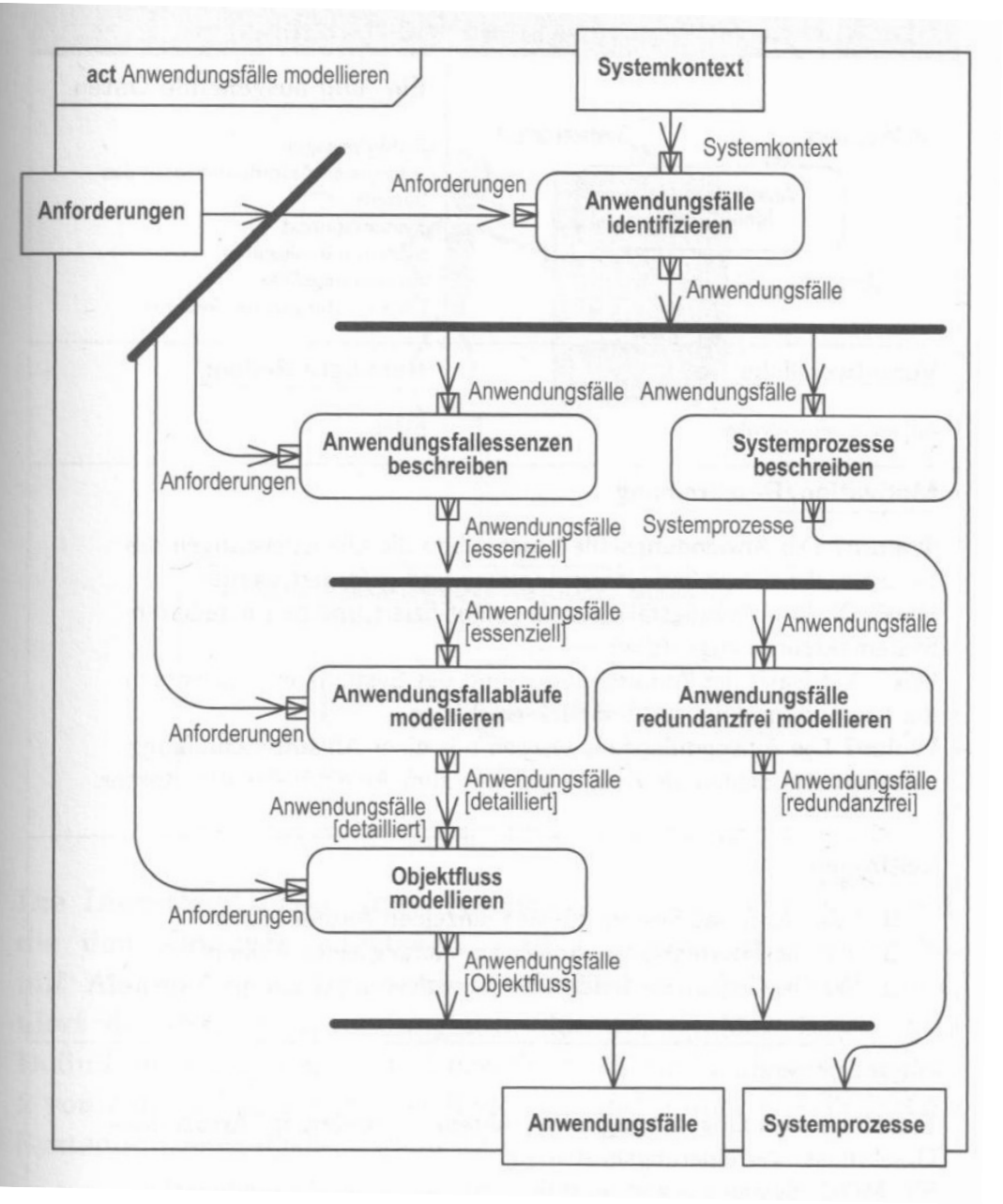
(4.3) Identify system interaction points

- **Motivation/Description:**
  - **Why?** The interaction points describe the interface to the system environment (system integration!).
  - **What?** Describe the points of the system through which interaction with the environment takes place.
  - **How?** Consider the associated interaction point for each actor and specify the services and objects that are offered or requested via that point.
  - **Where?** The interaction points are specified in more detail in the architecture and connected to the internal building blocks of the system.

(4.3) Identify system interaction points

- **Key questions:**
  - How does the system exchange information with its environment?
  - Which interaction channels with the actors can be combined?
- **Model elements:**
  - **SysML:** Block definition diagram, internal block diagram, system component, actor, association, role, connector, port

(5) Modeling use cases



(5.1) Identify use cases

- **Incoming and outgoing data:**
  - Requirements
  - System context
  - Use cases
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - None

(5.1) Identify use cases

- **Motivation/Description:**
  - **Why?** Use cases describe the services provided by the system.
  - **What?** Use cases are identified and assigned to the actors involved.
  - **How?** Use cases are systematically developed based on the requirements and the system context.
  - **Where?** Use cases are provided with a process description and are used to derive the architecture.

(5.1) Identify use cases

- **Key questions:**
  - What can the system do for the individual actor?
  - What information is available at the start of a process?
  - What events do the services deliver to the actors?
- **Model elements:**
  - **SysML:** Use case diagram, actor, requirements, association, object flow, refinement relationship
  - **SYSMOD:** System use case, continuous use case (use case that delivers continuous results)

(5.2) Describe essential use cases

- **Incoming and outgoing data:**
  - Requirements
  - Use cases
  - Use cases (essential)
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - None

(5.2) Describe essential use cases

- **Motivation/Description:**
  - **Why?** Need an overview of the system's services that can be determined quickly, and that is independent of technical solutions.
  - **What?** Description of the technical intention of the use case in the form of essential steps without considering technical details and specific processes.
  - **How?** Consider the steps of the standard process and abstract the technical details.
  - **Where?** The essential steps determine the structure of the next level of detail. The description is stable and can be reused for subsequent or related systems.

(5.2) Describe essential use cases

- **Key questions:**
  - What is the technical intention of the use case?
  - Does the essential description contain technical solutions?
  - Is an essential step the same as the use case?
  - Is the essential description understandable to an expert who is not familiar with the project?
  - Does the use case have 2-8 essential steps?
- **Model elements:**
  - **SysML:** Use case diagram, comment
  - **SYSMOD:** System use case

(5.3) Describe system processes

- **Incoming and outgoing data:**
  - Use cases
  - System processes (cross-use case process consisting of a set of use cases that have a logical sequence)
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - None

(5.3) Describe system processes

- **Motivation/Description:**
  - **Why?** Complex sequences of events between use cases must be explicitly addressed and modeled.
  - **What?** Description of the sequence dependencies between use cases and summary of sequences in system processes.
  - **How?** Investigation of sequence dependencies of related use cases based on their preconditions and postconditions, and description of activities using a state machine.
  - **Where?** System processes must be taken into account in the architecture and often also implemented.

(5.3) Describe system processes

- **Key questions:**
  - What process dependencies exist between the use cases?
  - Which use cases are closely related in terms of subject matter?
- **Model elements:**
  - **SysML:** Use case diagram, activity diagram, contains relationship, activity, action, edge, control node, composition
  - **SYSMOD:** System process

(5.4) Model use cases without redundancy

- **Incoming and outgoing data:**
  - Use cases
  - Use cases (redundancy-free)
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - None

(5.4) Model use cases without redundancy

- **Motivation/Description:**
  - **Why?** Redundant model information can lead to serious problems if its consistency is violated.
  - **What?** Identification of commonalities between the processes of the use cases. Isolated modeling of these areas to avoid redundancies.
  - **How?** Commonalities between use cases are described in secondary use cases and integrated into the primary use cases via containment relationships.
  - **Where?** The redundancy-free use case structure provides good starting points for a redundancy-free system architecture.

(5.4) Model use cases without redundancy

- **Key questions:**
  - Which use case steps are repeated in other use cases?
  - Which use cases are similar?
- **Model elements:**
  - **SysML:** Use case diagram, use case, contains relationship, generalization
  - **SYSMOD:** Secondary use case

(5.5) Model use case flows

- **Incoming and outgoing data:**
  - Requirements
  - Use cases (essential)
  - Use cases (detailed)
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - System architect

(5.5) Model use case flows

- **Motivation/Description:**
  - **Why?** The process descriptions of the use cases are part of the core information of the requirements analysis. They specify in detail the required behavior of the system.
  - **What?** Description of the processes of the use cases with all exceptions and variants.
  - **How?** Use case processes are described with SysML activities.
  - **Where?** The activities form the direct basis for the functional aspects of the system architecture. To involve the client in system development.

(5.5) Model use case flows

- **Key questions:**
  - What steps are necessary for the use case?
  - What exceptions and variants can occur during the process?
  - Is the use case described in sufficient detail and in an understandable way?
- **Model elements:**
  - **SysML**: Activity diagram, activity, action, edge, control node
  - **SYSMOD**: Essential/continuous activity

(5.6) Model object flow

- **Incoming and outgoing data:**
  - Requirements
  - Use cases (detailed)
  - Use cases (object flow)
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - None

(5.6) Model object flow

- **Motivation/Description:**
  - **Why?** Examining the incoming and outgoing data of the process steps sharpens their description and provides important information for the architecture (interfaces!).
  - **What?** Modeling the incoming and outgoing data of the individual application steps and their relationships.
  - **How?** Activity actions contain pins that describe the incoming and outgoing data.
  - **Where?** The object flow adds a level of detail to the model that can be used for simulations and automated consistency checks.

(5.6) Model object flow

- **Key questions:**
  - What data is required by an use case step?
  - What data is generated or modified by an use case step?
- **Model elements:**
  - **SysML**: Activity diagram, activity, activity parameter, action, pin, control node

(6) Modeling domain knowledge

- **Incoming and outgoing data:**
  - Requirements
  - Use cases (object flow)
  - Expertise/domain knowledge (structure of the system's technical terms)
- **Responsible role:**
  - Requirements analyst
- **Roles involved:**
  - None

(6) Modeling domain knowledge

- **Motivation/Description:**
  - **Why?** The technical objects used in the object flow of activities must be defined. A clear understanding by stakeholders and model consistency depend on this.
  - **What?** Modeling the structure of technically relevant terms from the perspective of the system.
  - **How?** Terms and structures are modeled as technical system components in a block definition diagram.
  - **Where?** The domain knowledge model reflects the static view of the domain logic and is well-suited for coordination with the client and as a basis for the architecture.

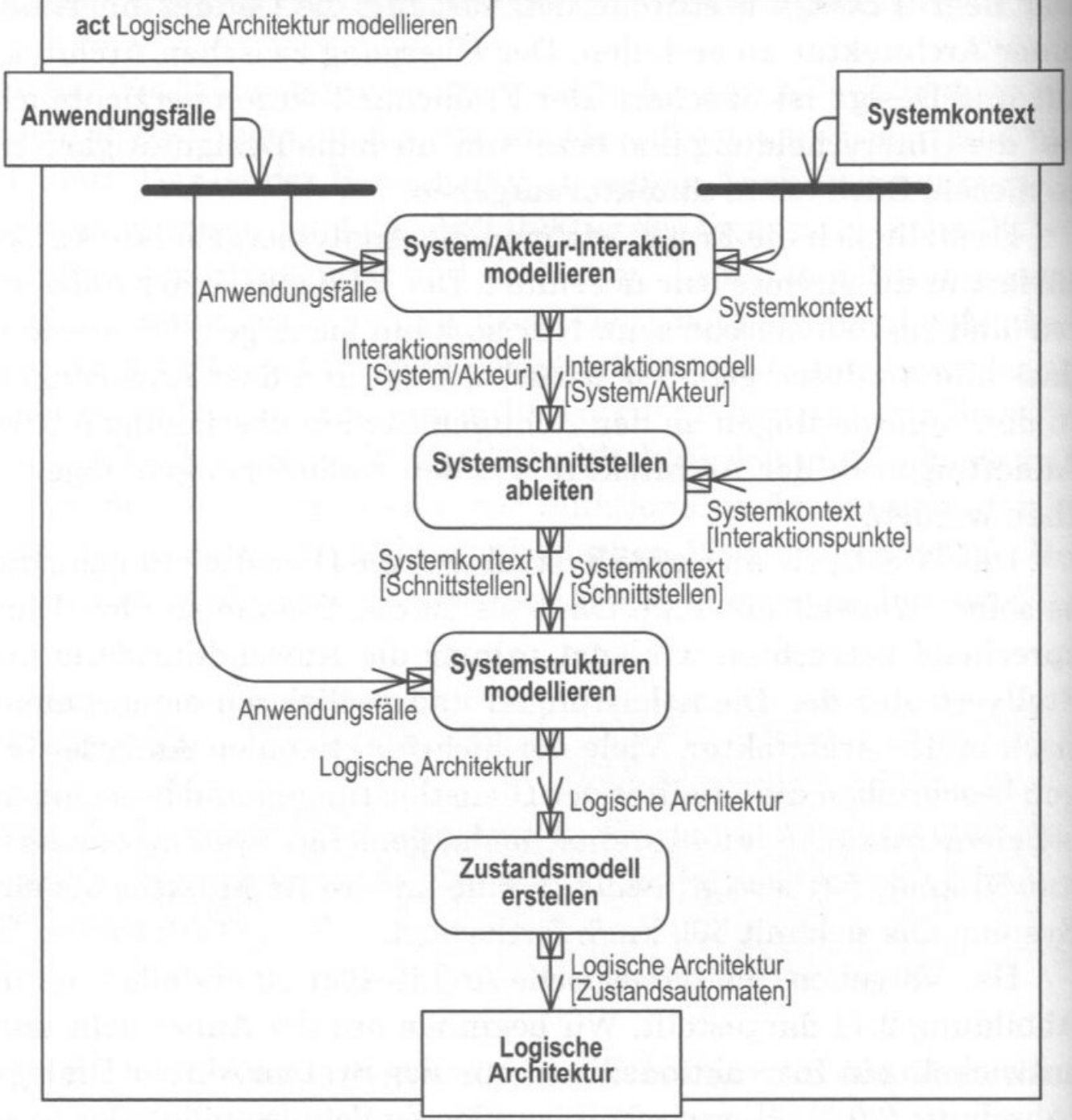
(6) Modeling domain knowledge

- **Key questions:**
  - Which technical terms does the system deal with?
  - Is the client familiar with the term?
  - Is the term important enough to be explicitly modeled?
- **Model elements:**
  - **SysML**: Block definition diagram, association, generalization
  - **SYSMOD**: Technical system component

(7) Modeling logical architecture

- **Architecture:** The sum of fundamental decisions about the structure of a system that are difficult to revise later on.

- **Logical architecture:** Description of the technical concepts and principles of the system.


act Logische Architektur modellieren
Anwendungsfälle
Systemkontext
System/Akteur-Interaktion modellieren
Anwendungsfälle
Systemkontext
Interaktionsmodell [System/Akteur]
Interaktionsmodell [System/Akteur]
Systemschnittstellen ableiten
Systemkontext [Interaktionspunkte]
Systemkontext [Schnittstellen]
Systemkontext [Schnittstellen]
Systemstrukturen modellieren
Anwendungsfälle
Logische Architektur
Logische Architektur
Zustandsmodell erstellen
Logische Architektur [Zustandsautomaten]
Logische Architektur

(7.1) Modeling system/actor interaction

- **Incoming and outgoing data:**
  - System context
  - Use cases
  - Interaction model (system/actor)
- **Responsible role:**
  - System architect
- **Roles involved:**
  - Requirements analyst

(7.1) Modeling system/actor interaction

- **Motivation/Description:**
  - **Why?** The system context and use cases do not consider the interaction between the system and actors in detail, which can lead to misjudgments of usability and system integration.
  - **What?** Description of the interactions between the system and actors in relation to a use case.
  - **How?** The interaction between the system and actors is described in a sequence diagram for selected process scenarios.
  - **Where?** The system/actor interaction provides a template for the design, specifically for defining the interfaces.

(7.1) Modeling system/actor interaction

- **Key questions:**
  - What requests do the actors send to the system during a use case sequence?
  - What requests does the system send to the actors during a use case sequence?
  - Who executes which requests and when?
- **Model elements:**
  - **SysML**: sequence diagram, lifeline, message, combined fragment

(7.2) Derive system interfaces

- **Incoming and outgoing data:**
  - System context (interaction points)
  - Interaction model (system/actor)
  - System context (interfaces)
- **Responsible role:**
  - System architect
- **Roles involved:**
  - None

(7.2) Derive system interfaces

- **Motivation/Description:**
  - **Why?** The interface specification is necessary in order to integrate the system into its environment.
  - **What?** Description of the interfaces between the system and actors in relation to the individual interaction points.
  - **How?** Derivation of the interfaces from the system/actor interaction points and modeling using ports.
  - **Where?** The system interfaces represent the “contracts” that the system concludes with its actors in order to integrate the system permanently and successfully into its context.

(7.2) Derive system interfaces

- **Key questions:**
  - What services are offered via the interaction points?
  - What services are requested?
  - What objects flow through the interaction points?
- **Model elements:**
  - **SysML**: Block definition diagram, port, interface
  - **SYSMOD**: User interface, document module

(7.3) Modeling system structures

- **Incoming and outgoing data:**
  - System context (interfaces)
  - Use cases
  - Logical architecture
- **Responsible role:**
  - System architect
- **Roles involved:**
  - None

(7.3) Modeling system structures

- **Motivation/Description:**
  - **Why?** System structures are the static building blocks of logical architecture.
  - **What?** Modeling of system building blocks and their relationships in sufficient detail as required for the overall system to meet project requirements.
  - **How?** The necessary building blocks and structures are determined for each use case or requirement and modeled using block diagrams.
  - **Where?** The building blocks of the logical architecture describe generic concepts and principles and can be reused.

(7.3) Modeling system structures

- **Key questions:**
  - Which building blocks are needed to implement the use cases/requirements?
  - How is a building block structured?
  - How are the building blocks connected to each other?
  - What interaction points and interfaces do the building blocks have?
- **Model elements:**
  - **SysML**: block definition diagram, internal block definition diagram, system building block, interface, association, connector, port, object flow
  - **SYSMOD**: discipline-specific elements

(7.4) Create state model

- **Incoming and outgoing data:**
  - Logical architecture
  - Use cases
  - Logical architecture (state machines)
- **Responsible role:**
  - System architect
- **Roles involved:**
  - None

(7.4) Create state model

- **Motivation/Description:**
  - **Why?** System behavior is described by the states of the system components.
  - **What?** Modeling of finite state machines for all relevant system components.
  - **How?** Description of finite state machines using state diagrams. The starting point is interactions between system components and use case sequences.
  - **Where?** State machines can be easily executed and used to simulate the system.

(7.4) Create state model

- **Key questions:**
  - Are all paths of the system processes taken into account in the state machines?
  - Are different aspects modeled in separate regions?
- **Model elements:**
  - **SysML**: State diagram, state machine, state, transition, pseudo states

(7.5) Modeling physical product architecture

- **Incoming and outgoing data:**
  - Logical architecture
  - Physical product architecture
- **Responsible role:**
  - System architect
- **Roles involved:**
  - None

(7.5) Modeling physical product architecture

- **Motivation/Description:**
  - **Why?** The physical product architecture specifies the specific product details that are not included in the logical architecture.
  - **What?** Modeling of system components, their relationships, and their behavior at the level of very specific components.
  - **How?** Modeling takes place at a concrete level similar to that of logical architecture.
  - **Where?** Physical product architecture is used for detailed, concrete, and interdisciplinary product development.

(7.5) Modeling physical product architecture

- **Key questions:**
  - What added value does the high level of detail in the physical product architecture offer compared to the logical architecture? Does the added value justify the effort?
  - Do you need a loose or tight coupling between the physical product architecture and the logical architecture?
- **Model elements:**
  - **SysML**: block definition diagram, internal block diagram, system block, interface, association, connector, port, object flow
  - **SYSMOD**: discipline-specific elements

Conclusions

- Process for modeling systems
- Abstract system descriptions
- Alignment with stakeholders
- Focus on requirements and use cases