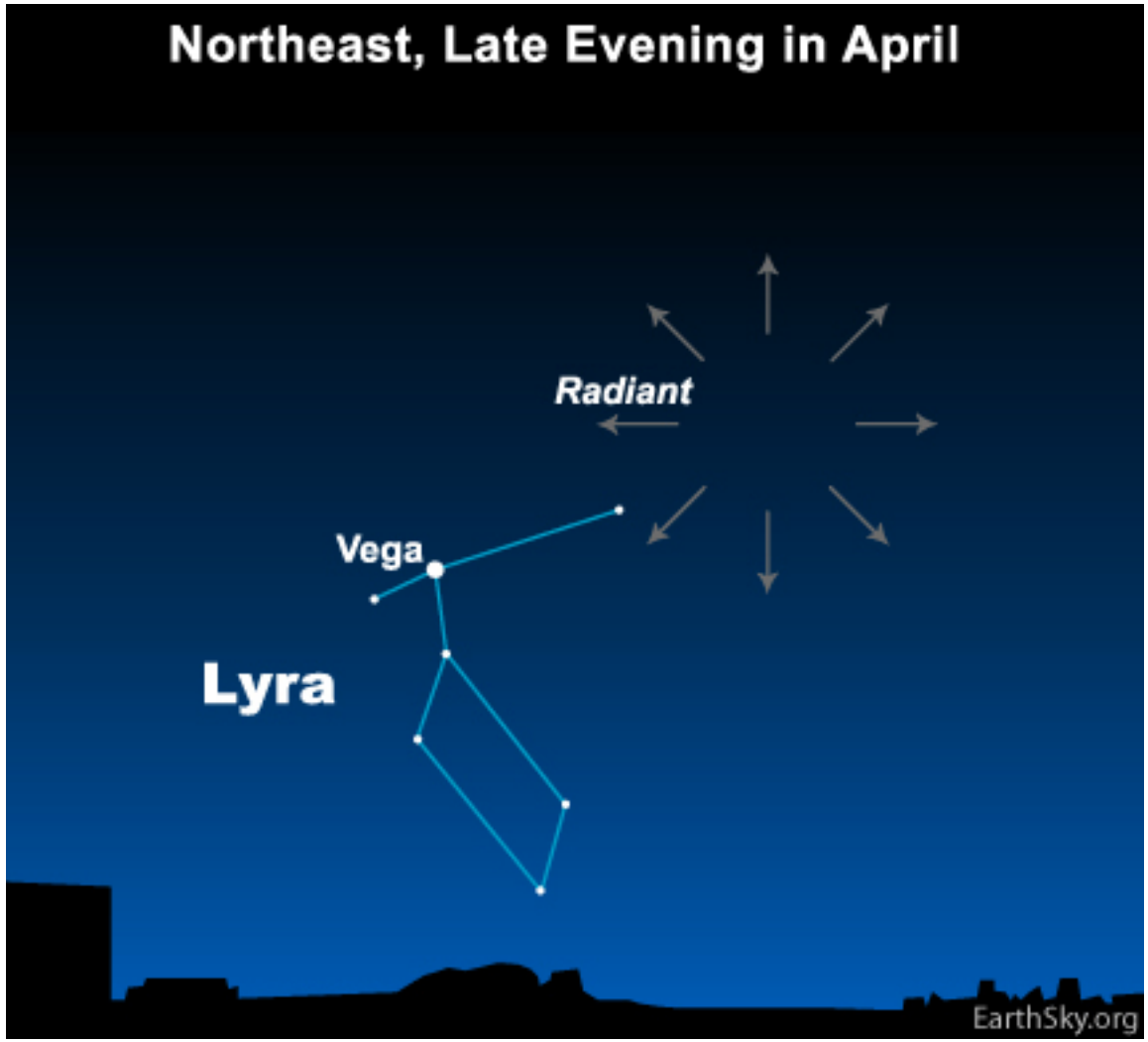
Requirements Engineering: Function Hierarchies and System Specification

CECS 590

Random side information



Lyrid meteor shower peaks tonight.

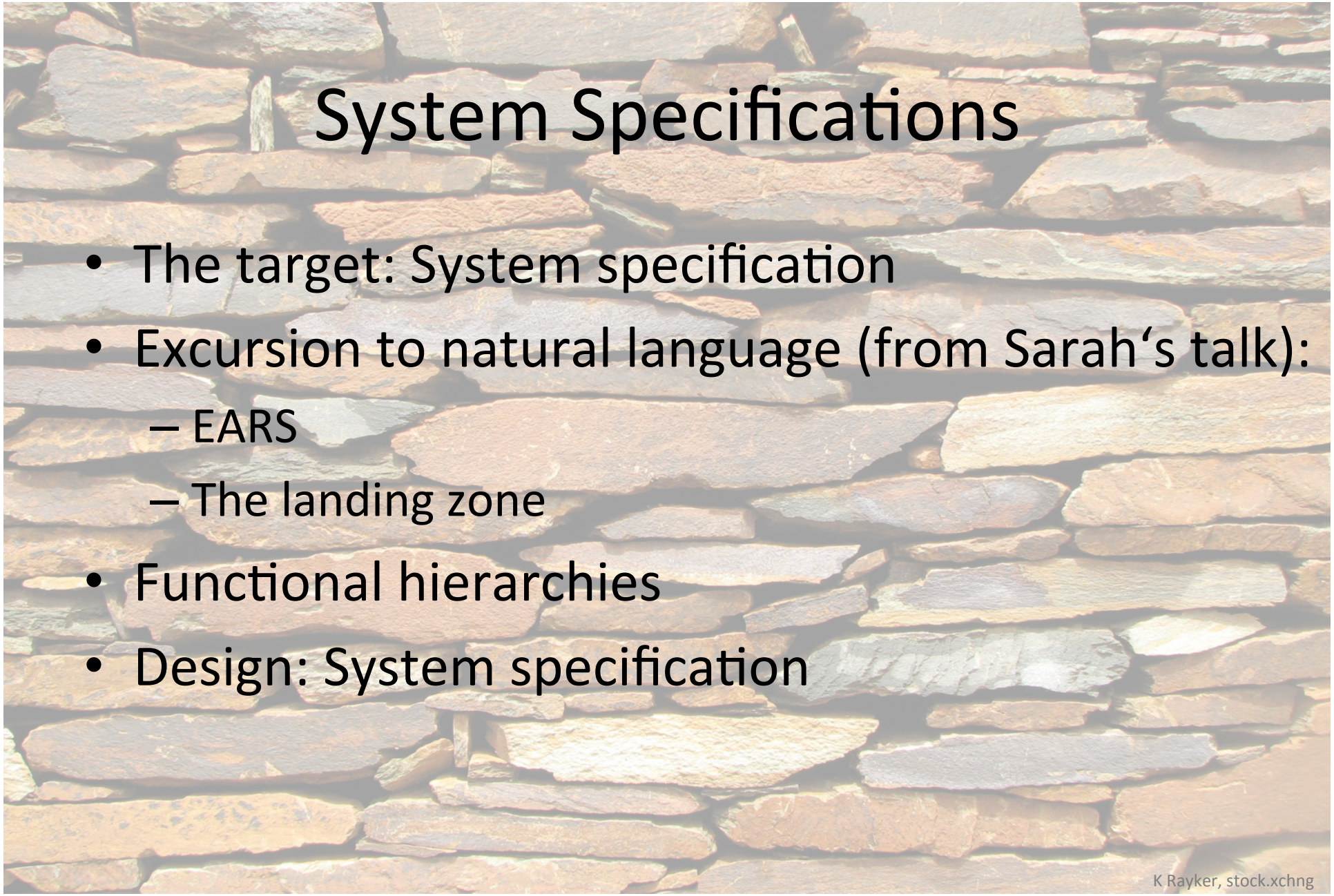
If you trace the paths of Lyrid meteors backward on the sky's dome, you'll find that they appear to originate from a point in the sky near the the star Vega, the heavens' 5th brightest star. This is the shower's radiant point. The radiant point for the Lyrid shower sits just to the right of Vega, which is the brightest star in the constellation Lyra the Harp. It's from Vega's constellation Lyra that the Lyrid meteor shower takes its name.

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?

System Specifications

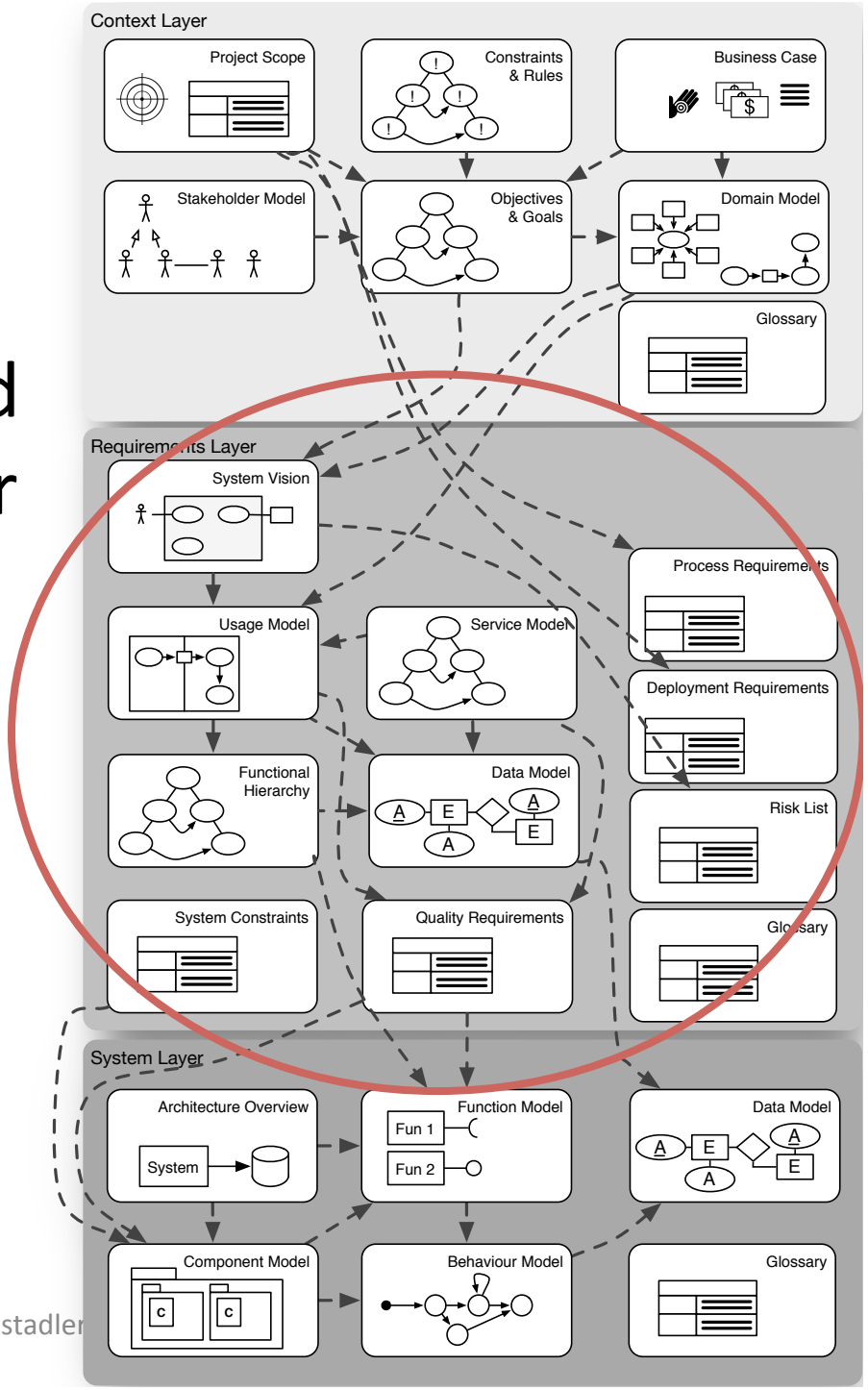
- The target: System specification
- Excursion to natural language (from Sarah‘s talk):
  - EARS
  - The landing zone
- Functional hierarchies
- Design: System specification



K Rayker, stock.xchng

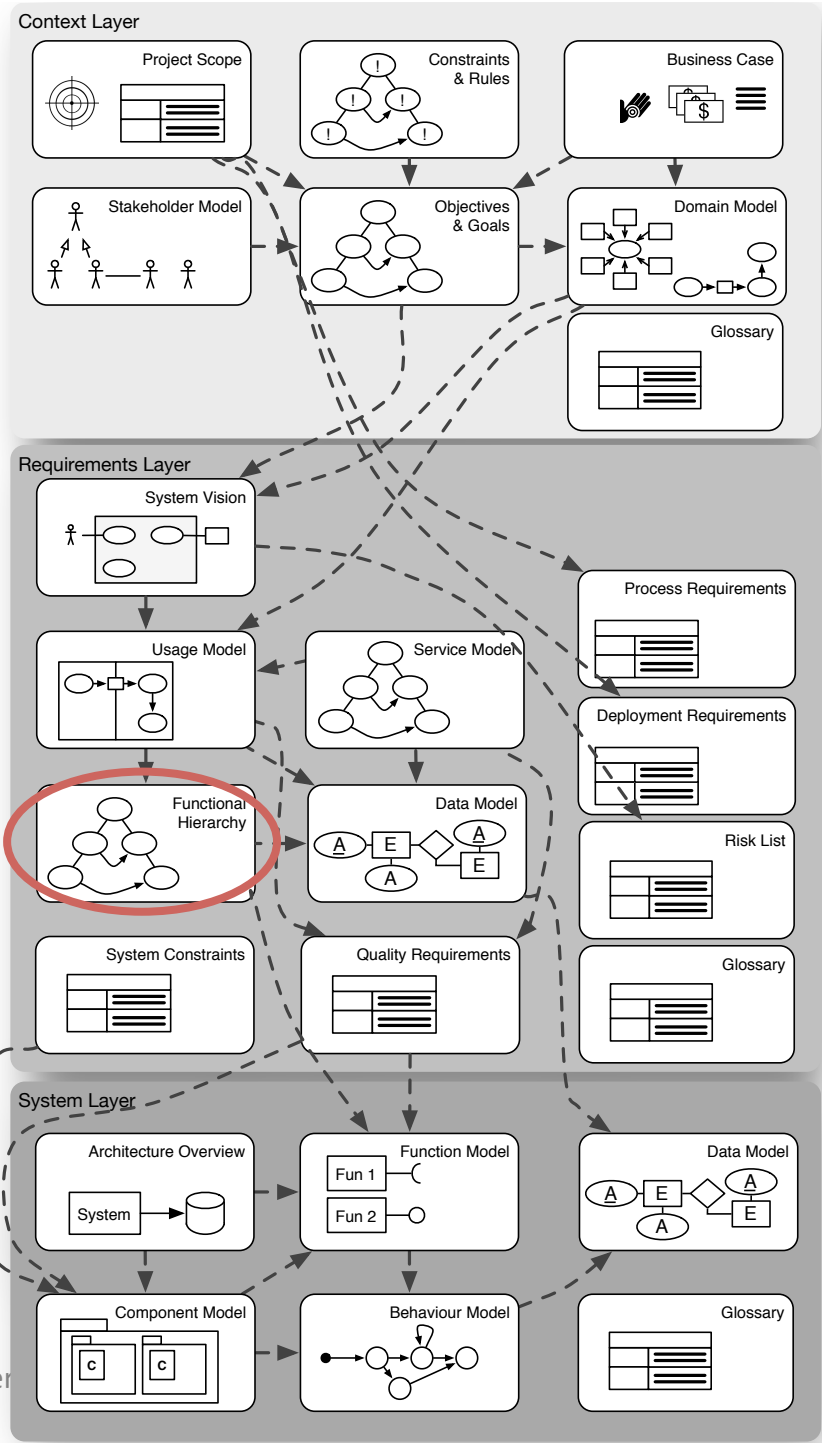
The target: System specification

- So far, we have modeled functionality from a user perspective.
  - Model-based
  - Natural language



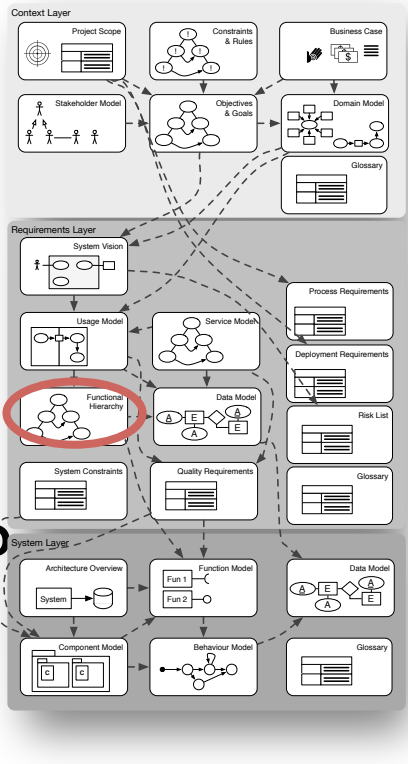
Next step: Function hierarchies

- So far, we have modeled functionality from a user perspective.
  - Model-based
  - Natural language
- When to use which one?
- How to move on?



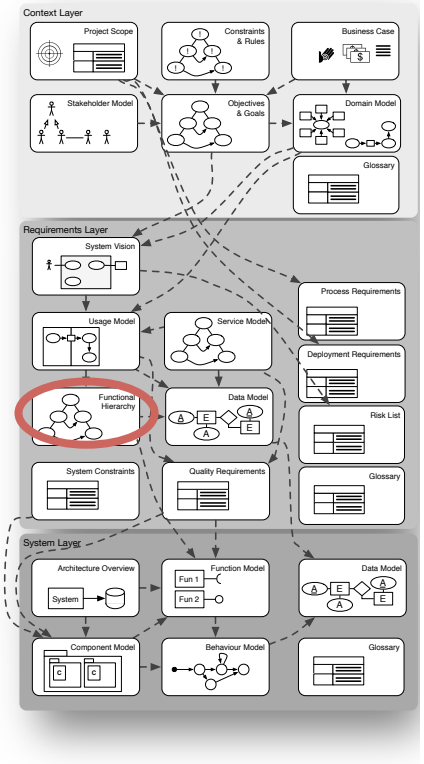
Functional hierarchies: Rationale

- Overview of functions serves as foundation for system analysis
  - Identification in Business Processes
  - Analysis in Use Cases and Scenarios
  - Structuring in Function Hierarchy
- Why do we need a function hierarchy?
  - Syntactic interface
  - Determination of logical components
  - Specification of behavior models



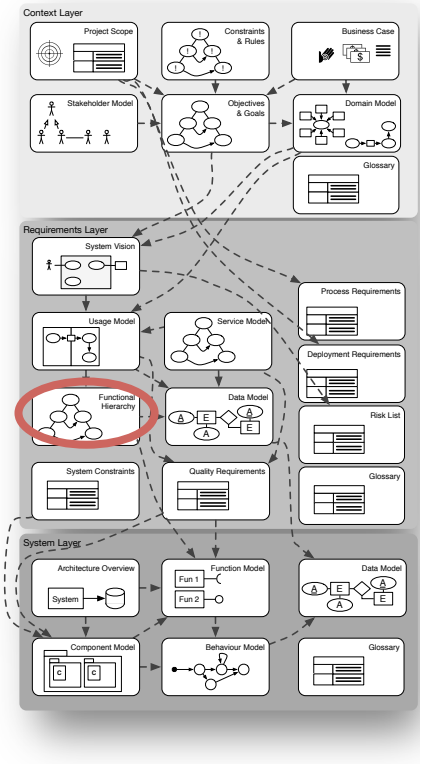
Functional hierarchies: Definition

- Def.: A (usage) function is a piece of behavior of the system that can be perceived at the system border and has a purpose.
- Def.: A Function Hierarchy is a decomposition of functions and subfunctions and their dependencies (feature interaction) for an analysis of their behavior according to different principles, for example:
  - Task and subtask
  - Causal / time order
  - Communication relation
  - Distribution onto system components
  - Domain-specific architecture principles



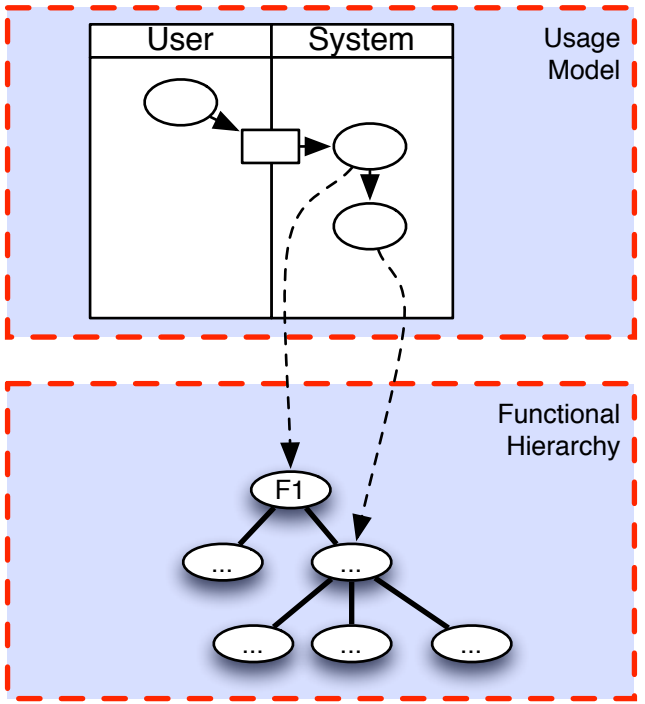
Functional hierarchies: Elements

- Function
- Subfunction
- Relations
  - Task and subtask (relation: *subfunction*)
  - Causal / time order (relation: *precedes/ follows, activates/deactivates*)
  - Communication relation (relation: <*data*>)
  - Distribution onto system components
  - Domain-specific architecture principles



From scenarios to functions

1. Identify System functions on basis of Use Cases and Scenarios
2. Structure and refine System Functions using a hierarchy
3. Identify dependencies between functions (e.g. using the state of the system)
4. Define external interface (Dialogues, Data, ...)



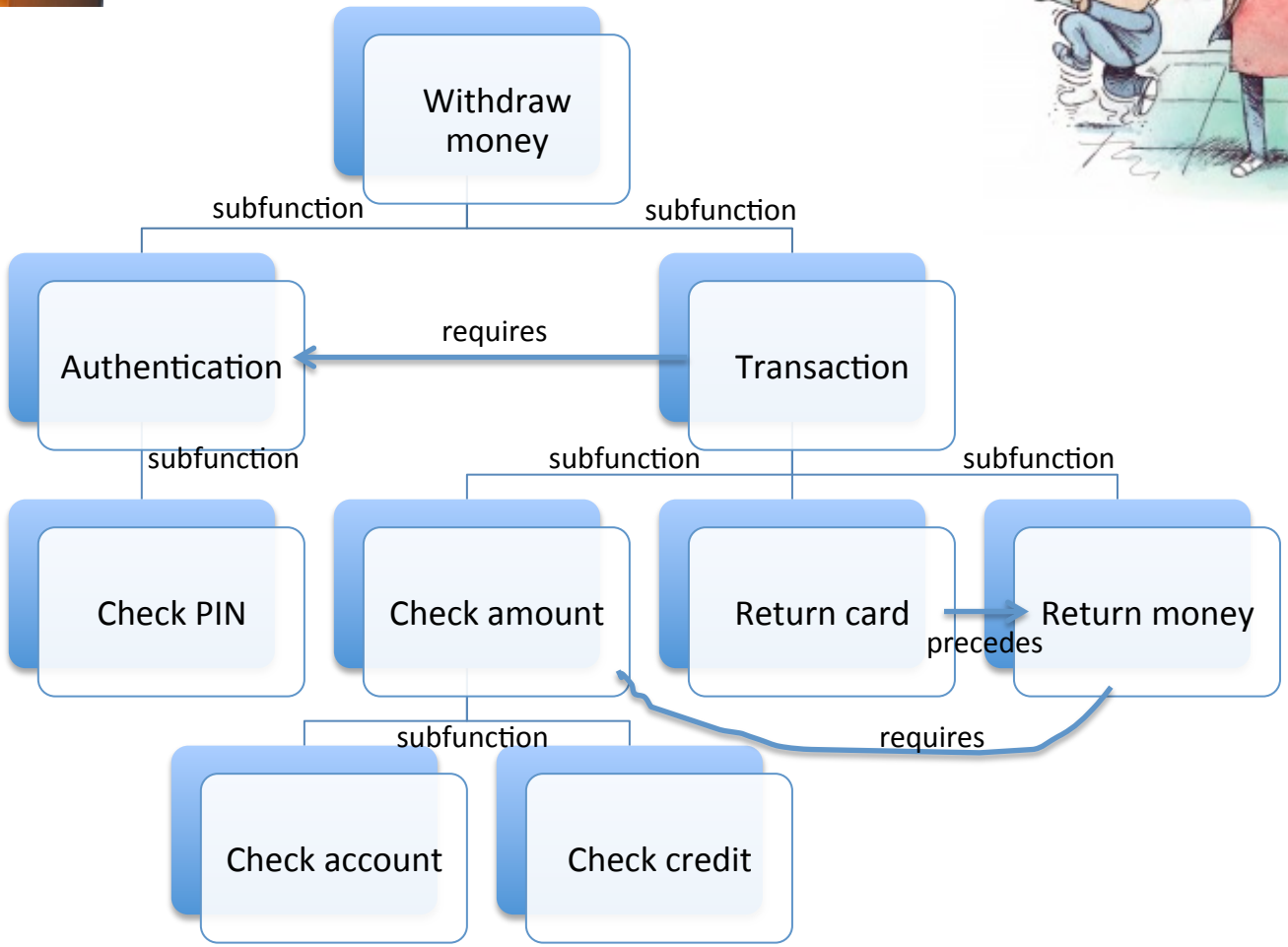
Function hierarchies allow a seamless transition to design

- Definition of a logical component architecture
- Specification of behavior with automata
- Iterative data modeling

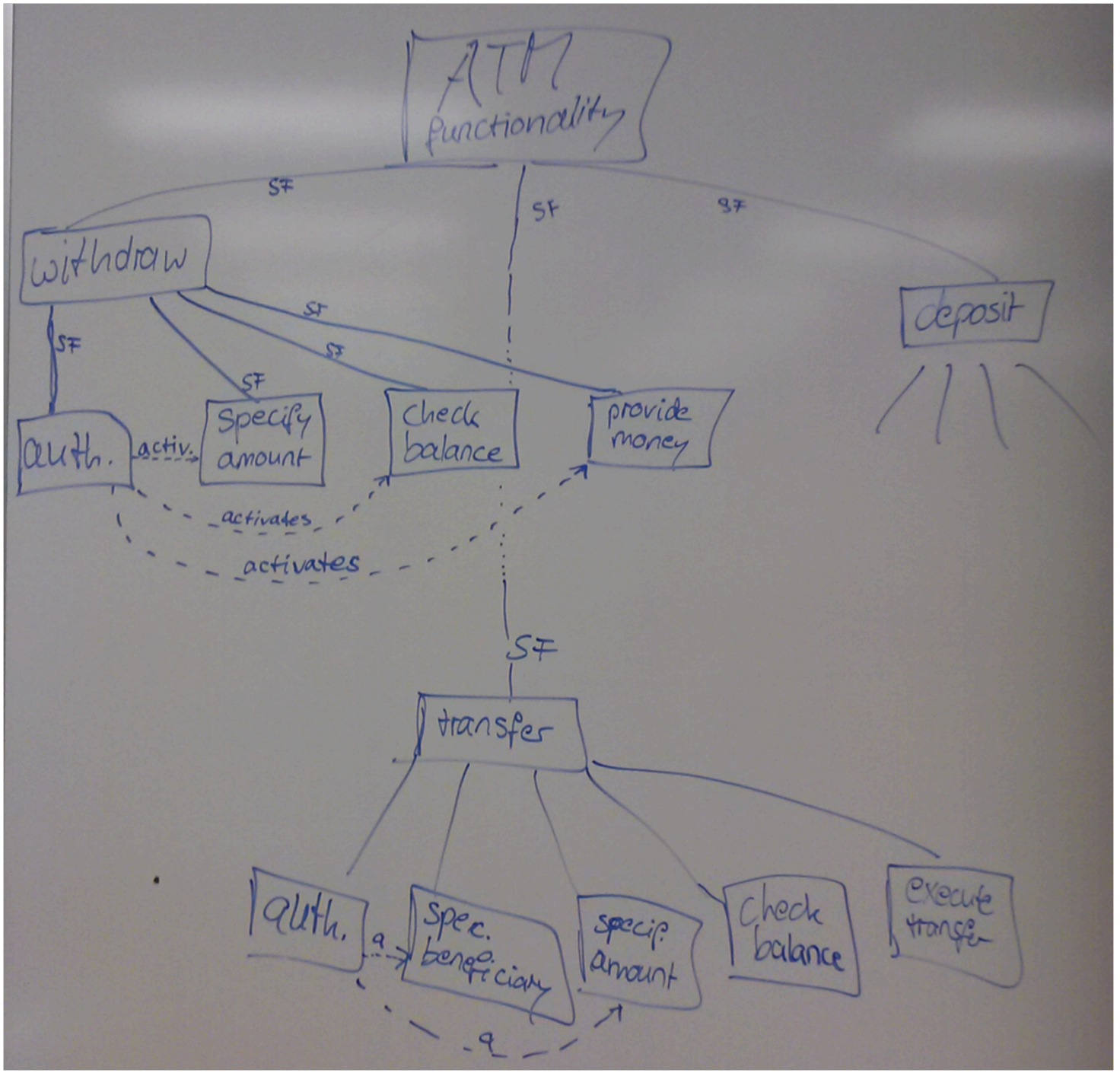
Exercise: ATM

- Function hierarchy
- Take the use cases
- Restructure in hierarchy
  - Task/subtask
  - Causal relations
  - Communication relations

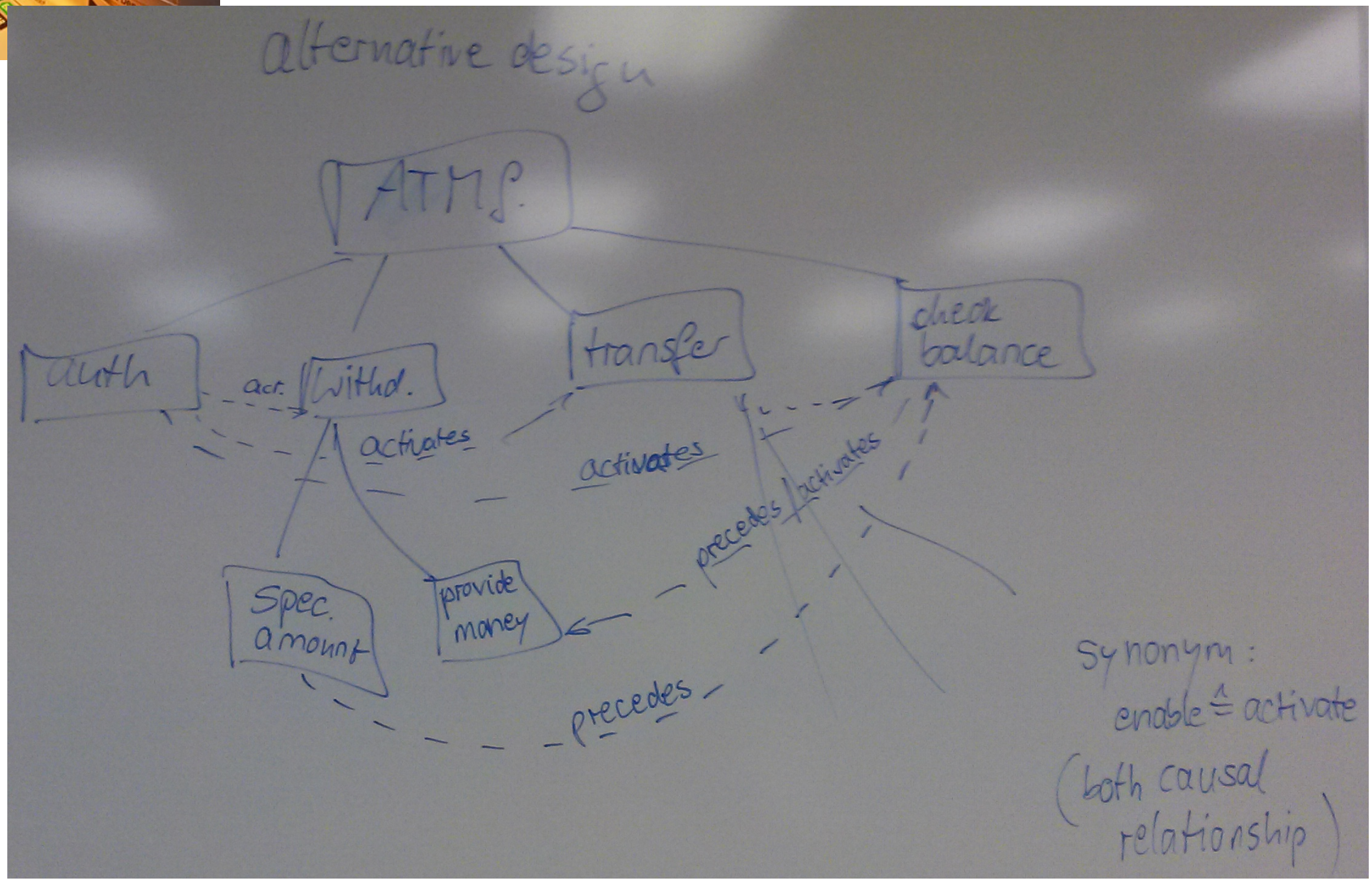
Exercise: ATM



Card
Deposit

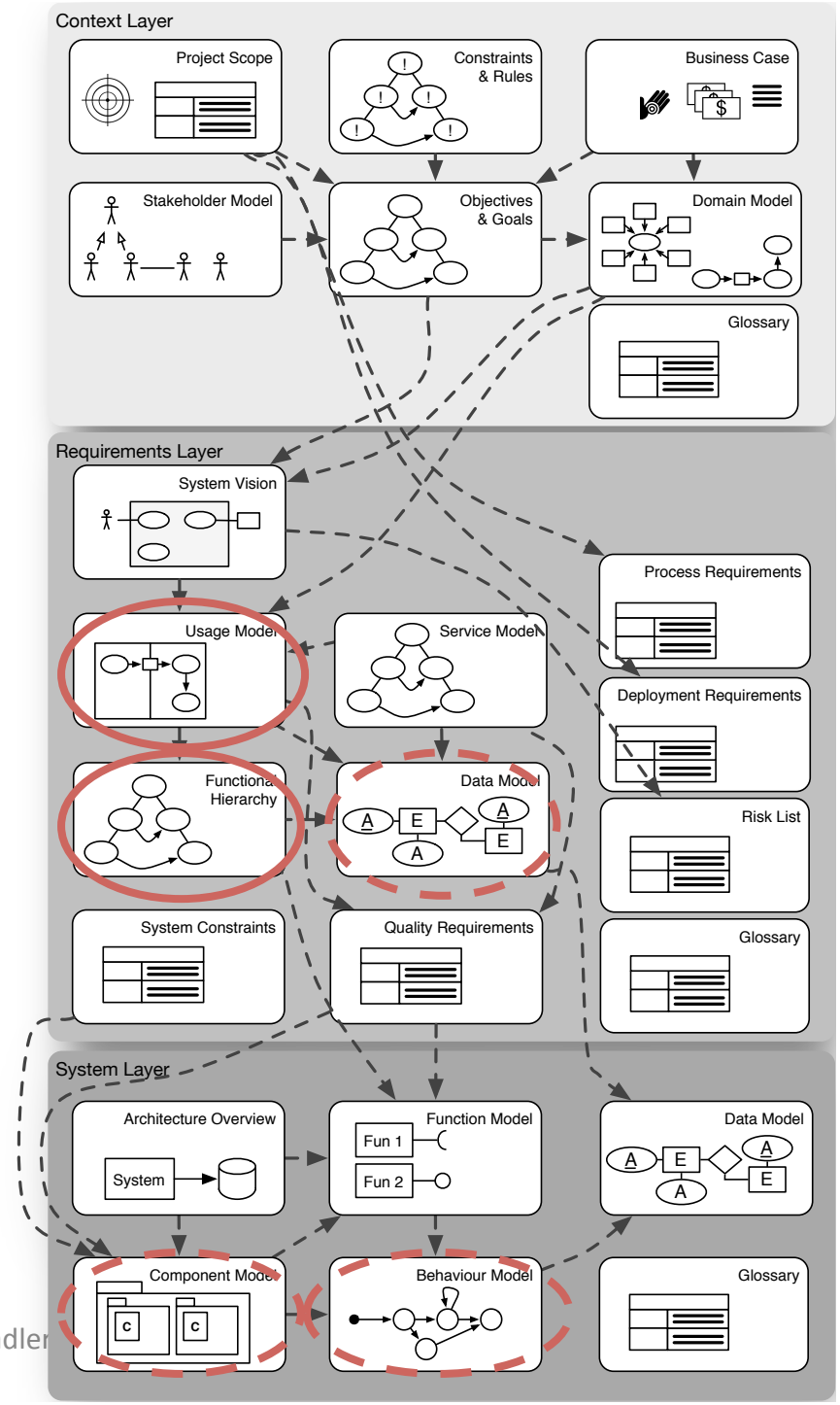

ATM functionality
withdraw
deposit
auth.
activ.
Specify amount
Check balance
provide money
activates
activates
transfer
auth.
Spec. beneficiary
specif. amount
Check balance
execute transfer

Exercise: ATM



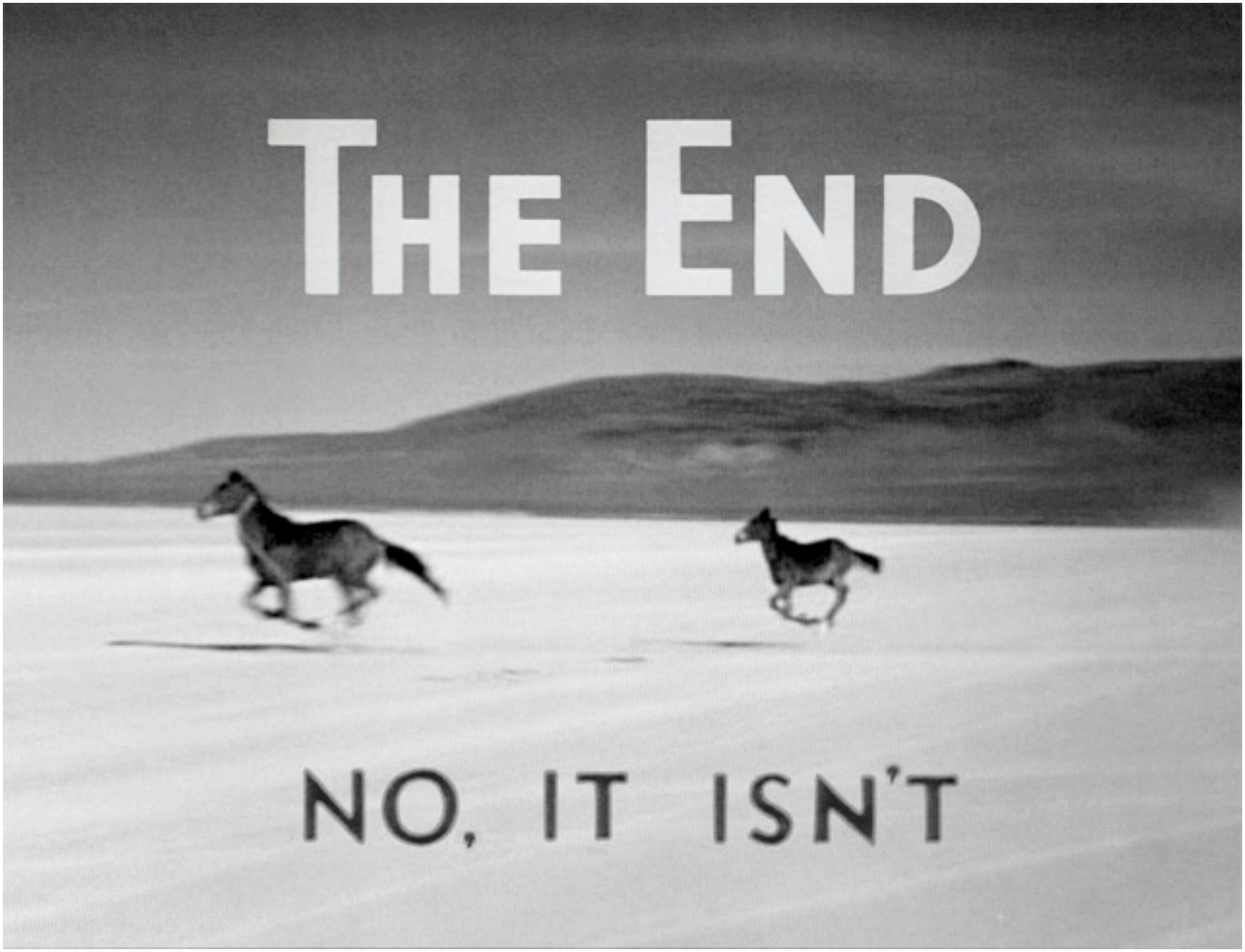
Moving twds. Design: System specification

- Use scenarios and function hierarchies
- Stepwise completion
  - Component design
  - Behavior specification
  - Data specification



Moving twds. Design: System specification

- Stepwise completion
  - Component design:
    - Class diagrams
  - Behavior specification:
    - Activity diagrams
    - State diagrams
  - Data specification:
    - Data models


THE END
NO, IT ISN'T