Requirement Engineering

Requirement Elicitation

Lecture 5

Software Lifecycle Definition

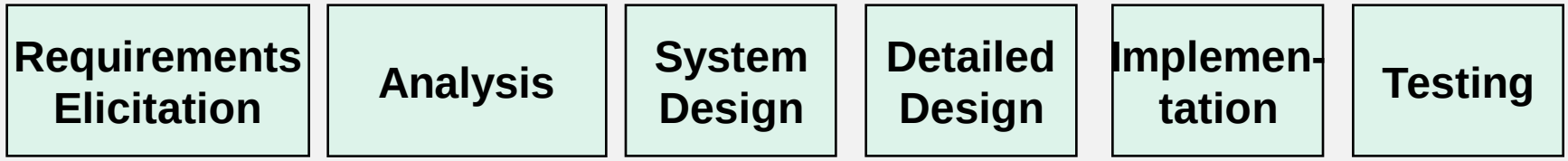
Software lifecycle

- Models for the development of software
  - Set of activities and their dependency relationships to each other to support the development of a software system
  - Examples:
    - Analysis, Design, Implementation, Testing

Typical Lifecycle questions:

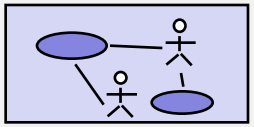
- Which activities should I select when I develop software?
- What are the dependencies between activities?
- How should I schedule the activities?

A Typical Example of Software Lifecycle Activities



Software Lifecycle Activities

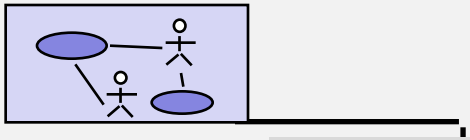
Requirements Elicitation	Analysis	System Design	Detailed Design	Implemen-tation	Testing



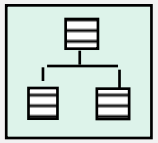
Use Case Model

Software Lifecycle Activities

Requirements Elicitation	Analysis	System Design	Detailed Design	Implemen-tation	Testing



Expressed in terms of

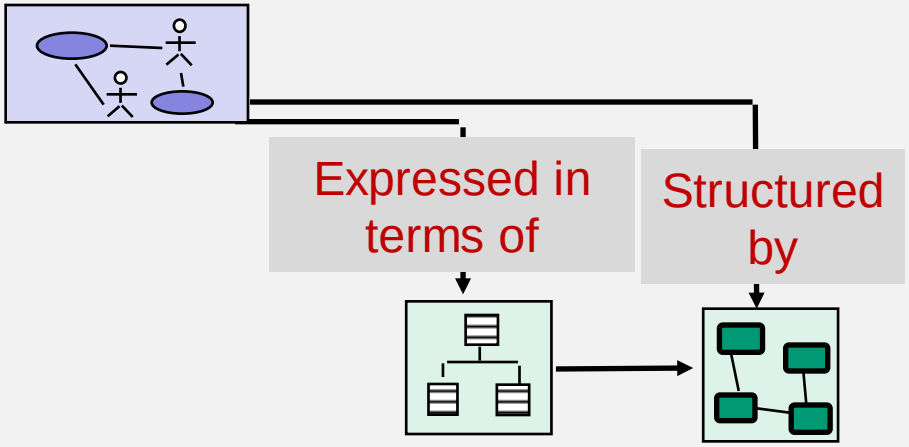


Use Case Model

Application Domain Objects

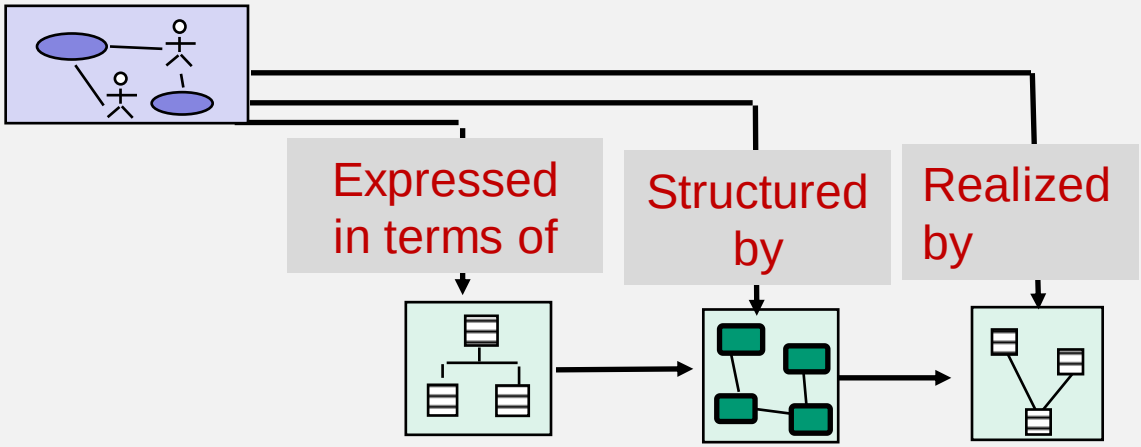
Software Lifecycle Activities

Requirements Elicitation	Analysis	System Design	Detailed Design	Implemen-tation	Testing



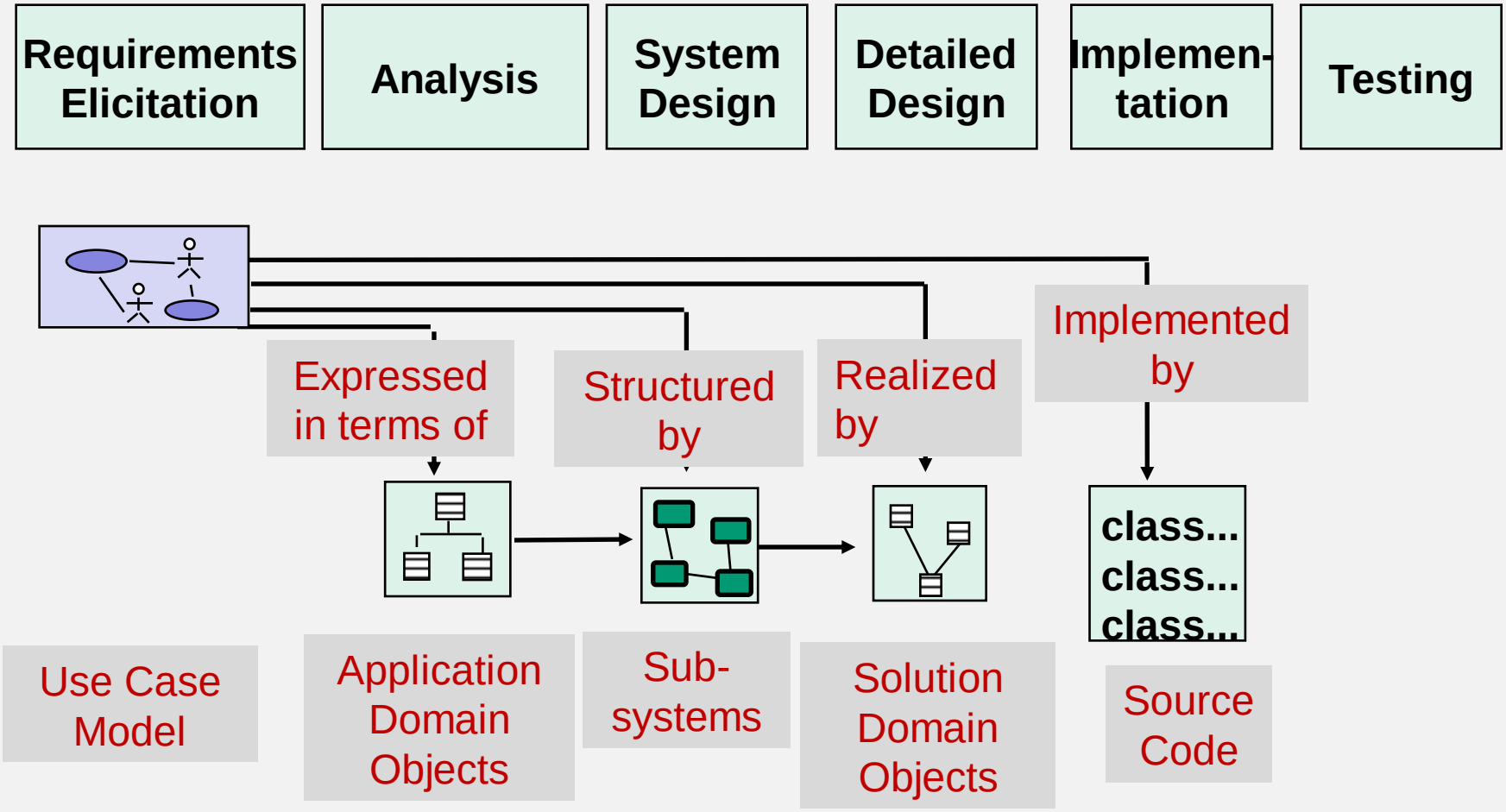
Software Lifecycle Activities

Requirements Elicitation	Analysis	System Design	Detailed Design	Implemen-tation	Testing

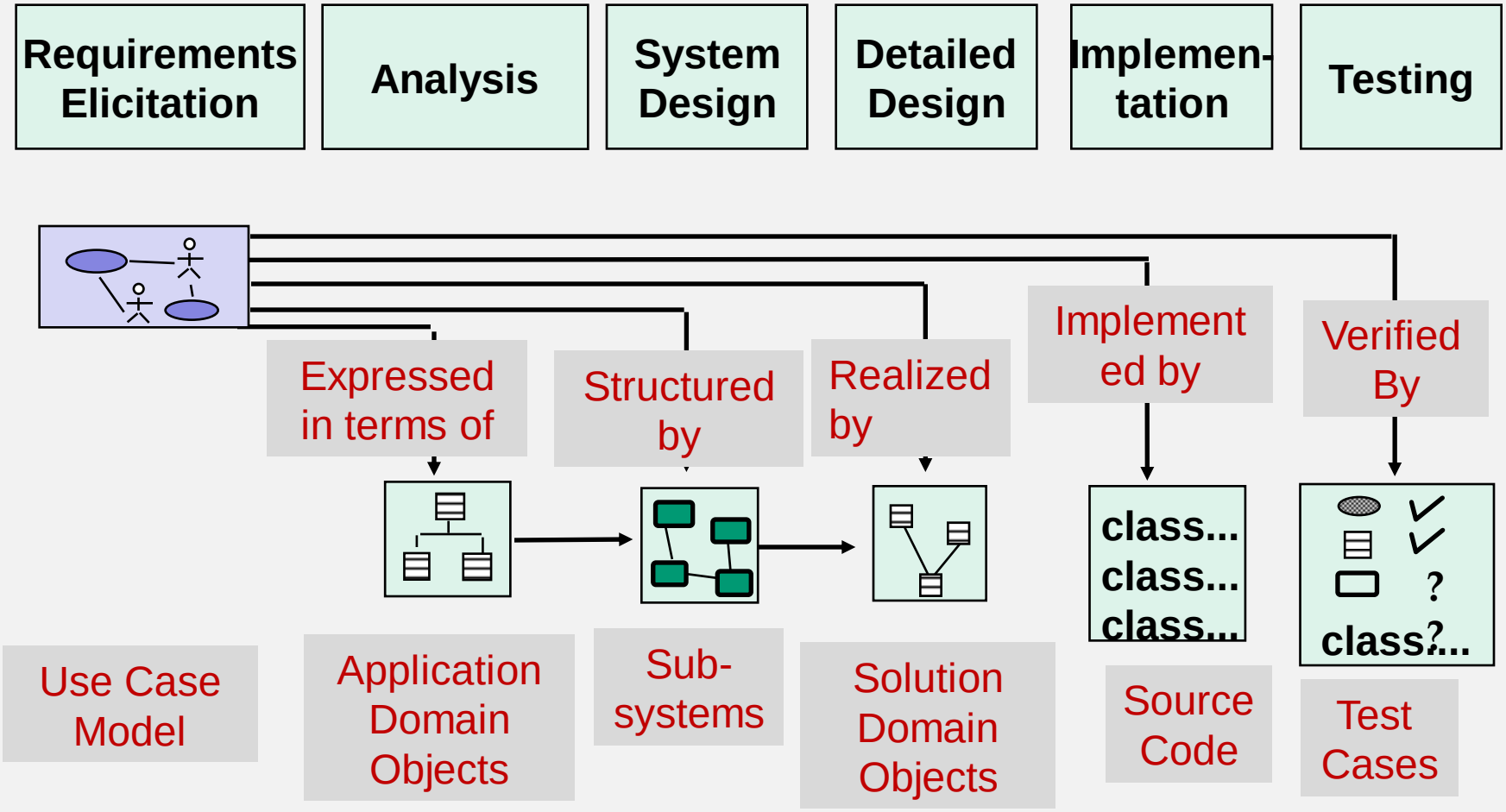


Use Case Model	Application Domain Objects	Sub-systems	Solution Domain Objects

Software Lifecycle Activities



Software Lifecycle Activities



What is the best Software Lifecycle?

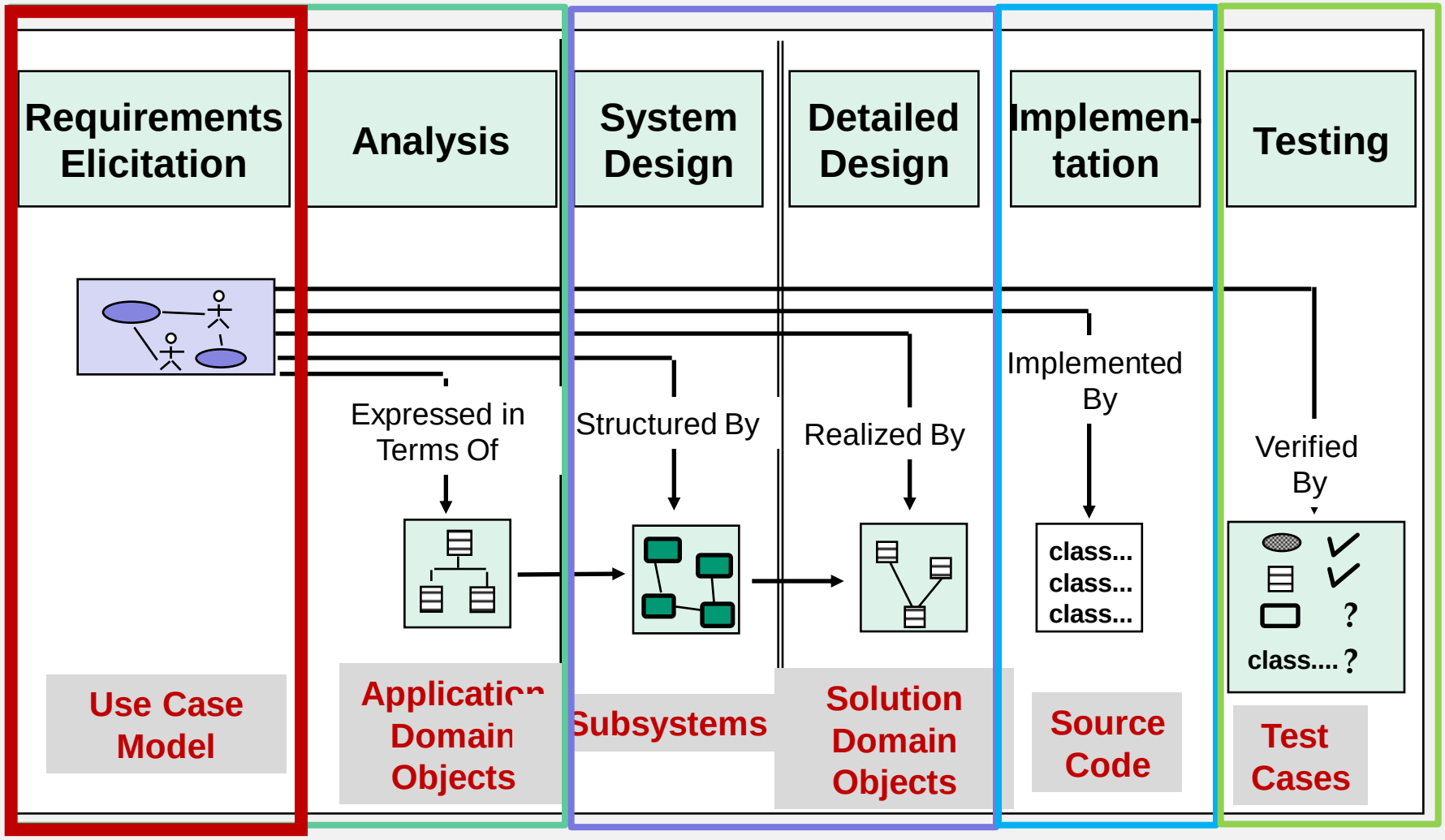
Answering this question is the topics of the lecture on software lifecycle modeling

For now we assume we have a set of predefined activities:

Today we focus on the activity

requirements elicitation

Software Lifecycle Activities



First step in identifying the Requirements: System identification

Two questions need to be answered:

1. How can we identify the purpose of a system?
2. What is inside, what is outside the system?

These two questions are answered during requirements elicitation and analysis

Requirements elicitation:

Definition of the system in terms understood by the customer (“Requirements specification”)

Analysis:

Definition of the system in terms understood by the developer (Technical specification, “Analysis model”)

Techniques to elicit Requirements

Bridging the gap between end user and developer:

- Questionnaires: Asking the end user a list of pre-selected questions
- Task Analysis: Observing end users in their operational environment
- Scenarios: Describe the use of the system as a series of interactions between a concrete end user and the system
- Use cases: Abstractions that describe a class of scenarios.

Scenarios

Scenario

A synthetic description of an event or series of actions and events.

A textual description of the usage of a system. The description is written from an end user's point of view.

A scenario can include text, video, pictures and story boards. It usually also contains details about the work place, social situations and resource constraints.

More Definitions

Scenario: “A narrative description of what people do and experience as they try to make use of computer systems and applications”

A concrete, focused, informal description of a single feature of the system used by a single actor.

Requirements Elicitation: Difficulties and Challenges

Communicate accurately about the domain and the system

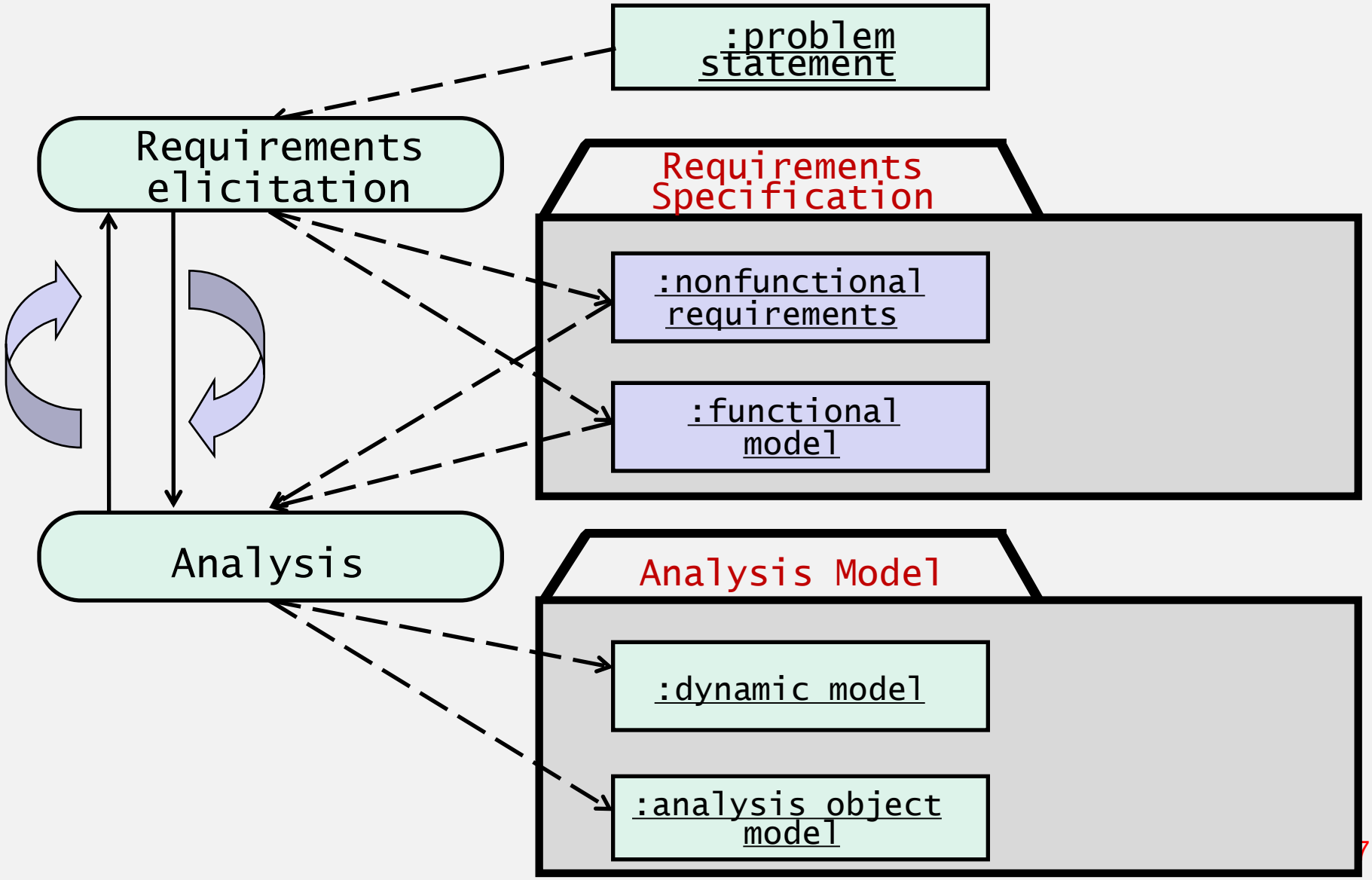
People with different backgrounds must collaborate to bridge the gap between end users and developers

- Client and end users have application domain knowledge
- Developers have solution domain knowledge

Identify an appropriate system (Defining the system boundary)

Provide an unambiguous specification

Requirements Process



Requirements Specification vs Analysis Model

Both focus on the requirements from the user's view of the system

The requirements specification uses natural language (derived from the problem statement)

The analysis model uses a formal or semi-formal notation

We use UML.

Requirements Elicitation Concepts

Main requirement elicitation concepts

1. Functional requirement
2. Nonfunctional requirement
3. Completeness, consistency, clarity and correctness
4. Realism, verifiability and traceability

Types of Requirements

Functional requirements

Describe the interactions between the system and its environment independent from the implementation

"An operator must be able to define a new game. "

Nonfunctional requirements

Aspects not directly related to functional behavior.

"The response time must be less than 1 second"

Functional vs. Nonfunctional Requirements

Functional Requirements	Nonfunctional Requirements
Describe user tasks that the system needs to support Interaction between the system and its environment independent of its implementation Phrased as actions “Advertise a new league” “Schedule tournament” “Notify an interest group”	Describe properties of the system or the domain Aspects that are not directly related to the functional behavior of the system Phrased as constraints or negative assertions “All user inputs should be acknowledged within 1 second” “A system crash should not result in data loss”.

Completeness, consistency, clarity and correctness

Complete

All features of interest are described by requirement

Consistent

No two requirements of the specification contradict each other

Unambiguous

A requirement cannot be interpreted in two mutually exclusive ways

Correct

Describe features of the system and environment of interest to the client. Do not describe unintended feature

Realism, verifiability and Traceability

Realism

Specification is realistic if the system can be implemented within constraint.

Verifiability

Specification is verifiable if once the system is built repeatable test can be designed that the system fulfills the requirements specification

Traceability

Specification is traceable if each requirement can be traced through the software to its function to requirement.

Requirements Elicitation Activities

1. Identifying actors
2. Identifying scenarios
3. Identifying use cases
4. Refining use cases
5. Identifying relationship among use cases
6. Identifying initial analysis objects
7. Identifying non functional requirement

Identifying Actors

Actors are external entities that interact with the system.

Actor can be human or an external system.

Actors are role abstractions and do not necessarily directly map to persons.

Actors are outside of the system boundary; they are external.

Questions for Identifying Actors

- Which user group are supported by the system to perform their work
- Which user group execute the system's main function
- Which user group perform secondary function's such as maintenance and administration
- With what external hardware or software system will the system interact

Identifying Actors

SatWatch is a wrist watch that displays the time based on its current location. SatWatch uses GPS satellites (Global Positioning System) to determine its location and internal data structures to convert this location into a time zone.

The information stored in SatWatch and its accuracy measuring time is such that the watch owner never needs to reset the time. SatWatch adjusts the time and date displayed as the watch owner crosses time zones and political boundaries. For this reason, SatWatch has no buttons or controls available to the user.

SatWatch determines its location using GPS satellites and, as such, suffers from the same limitations as all other GPS devices (e.g., inability to determine location at certain times of the day in mountainous regions). During blackout periods, SatWatch assumes that it does not cross a time zone or a politicalboundary. SatWatch corrects its time zone as soon as a blackout period ends.

SatWatch has a two-line display showing, on the top line, the time (hour, minute, second, time zone) and on the bottom line, the date (day, date, month, year). The display technology used is such that the watchowner can see the time and date even under poor light conditions.

When political boundaries change, the watch owner may upgrade the software of the watch using theWebifyWatch device (provided with the watch) and a personal computer connected to the Internet.

Identifying Actors

SatWatch is a wrist watch that displays the time based on its current location. SatWatch uses **GPS satellites** (Global Positioning System) to determine its location and internal data structures to convert this location into a time zone.

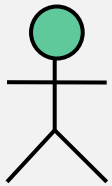
The information stored in SatWatch and its accuracy measuring time is such that the watch owner never needs to reset the time. SatWatch adjusts the time and date displayed as the watch owner crosses time zones and political boundaries. For this reason, SatWatch has no buttons or controls available to the user.

SatWatch determines its location using GPS satellites and, as such, suffers from the same limitations as all other GPS devices (e.g., inability to determine location at certain times of the day in mountainous regions). During blackout periods, SatWatch assumes that it does not cross a time zone or a politicalboundary. SatWatch corrects its time zone as soon as a blackout period ends.

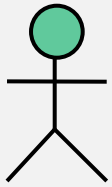
SatWatch has a two-line display showing, on the top line, the time (hour, minute, second, time zone) and on the bottom line, the date (day, date, month, year). The display technology used is such that the **watchowner** can see the time and date even under poor light conditions.

When political boundaries change, the watch owner may upgrade the software of the watch using the **WebifyWatch** device (provided with the watch) and a personal computer connected to the Internet.

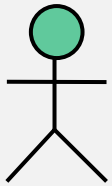
Identifying Actors



Watchowner



GPS



WebifyWatch

Identifying Actors

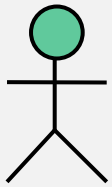
Your hairdresser salon has started a web-site for frequent customers. If you have signed up for the free membership you can login and book treatments. You can search for different employees, date, or time. The hairdressers use the system for planning, for instance, entering working hours and serving drop-in customers. The customer can erase bookings if more than 24 hours remain before the start of the treatment.

Identifying Actors

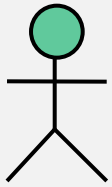
Your hairdresser salon has started a web-site for frequent customers. If you have signed up for the free membership you can login and book treatments. You can search for different employees, date, or time. The hairdressers use the system for planning, for instance, entering working hours and serving drop-in customers. The customer can erase bookings if more than 24 hours remain before the start of the treatment.

Identifying Actors

Your hairdresser salon has started a web-site for frequent customers. If you have signed up for the free membership you can login and book treatments. You can search for different employees, date, or time. The hairdressers use the system for planning, for instance, entering working hours and serving drop-in customers. The customer can erase bookings if more than 24 hours remain before the start of the treatment.



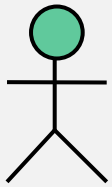
Hairdresser



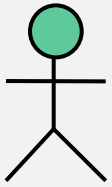
Customer

Identifying Actors

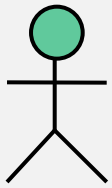
Your hairdresser salon has started a web-site for frequent customers. If you have signed up for the free membership you can login and book treatments. You can search for different employees, date, or time. The hairdressers use the system for planning, for instance, entering working hours and serving drop-in customers. The customer can erase bookings if more than 24 hours remain before the start of the treatment.



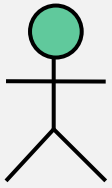
Hairdresser



Manager



Customer



Receptionist

Identifying Actors

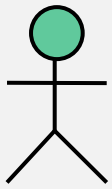
Consider normal operation of an ATM for withdrawing cash. The scenarios are like; a customer inserts the card, enters his/her PIN, enters the amount, takes the card, and takes the money. Identify the main actors and give the use-cases.

Identifying Actors

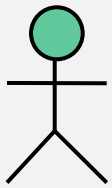
Consider normal operation of an ATM for withdrawing cash. The scenarios are like; a customer inserts the card, enters his/her PIN, enters the amount, takes the card, and takes the money. Identify the main actors and give the use-cases.

Identifying Actors

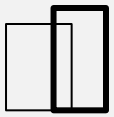
Consider normal operation of an ATM for withdrawing cash. The scenarios are like; a customer inserts the card, enters his/her PIN, enters the amount, takes the card, and takes the money. Identify the main actors and give the use-cases.



Customer



Technician



Bank

Identifying Actors

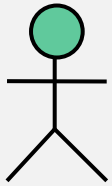
Consider a more complex example, FRIEND. A distributed information system for accident management. It includes many actors, such as FieldOfficer, who represents the police and fire officers who are responding to an incident, and Dispatcher, the police officer responsible for answering 911 calls and dispatching resources to an incident. FRIEND supports both actors by keeping track of incidents, resources, and task plans. It also has access to multiple databases, such as a hazardous materials database and emergency operations procedures. The FieldOfficer and the Dispatcher actors interact through different interfaces: FieldOfficers access FRIEND through a mobile personal assistant, Dispatchers access FRIEND through a workstation.

Identifying Actors

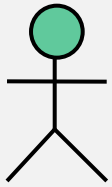
Consider a more complex example, FRIEND. A distributed information system for accident management. It includes many actors, such as FieldOfficer, who represents the police and fire officers who are responding to an incident, and Dispatcher, the police officer responsible for answering 911 calls and dispatching resources to an incident. FRIEND supports both actors by keeping track of incidents, resources, and task plans. It also has access to multiple databases, such as a hazardous materials database and emergency operations procedures. The FieldOfficer and the Dispatcher actors interact through different interfaces: FieldOfficers access FRIEND through a mobile personal assistant, Dispatchers access FRIEND through a workstation.

Identifying Actors

Consider a more complex example, FRIEND. A distributed information system for accident management. It includes many actors, such as FieldOfficer, who represents the police and fire officers who are responding to an incident, and Dispatcher, the police officer responsible for answering 911 calls and dispatching resources to an incident. FRIEND supports both actors by keeping track of incidents, resources, and task plans. It also has access to multiple databases, such as a hazardous materials database and emergency operations procedures. The FieldOfficer and the Dispatcher actors interact through different interfaces: FieldOfficers access FRIEND through a mobile personal assistant, Dispatchers access FRIEND through a workstation.



FieldOfficer



dispatcher

Identifying scenarios

Scenario: “A narrative description of what people do and experience as they try to make use of computer systems and applications”

A concrete, focused, informal description of a single feature of the system used by a single actor.

Types of Scenarios

As-is scenario:

Describes a current situation. Usually used in re-engineering projects. The user describes the system

Example: Description of Letter-Chess

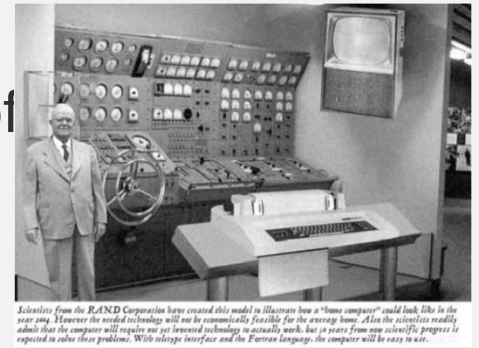
Visionary scenario:

Describes a future system. Usually used in greenfield engineering and reengineering projects

Can often not be done by the user or developer alone

Example: Description of an interactive internet-based Tic Tac Toe game tournament

Example: Description - in the year 1954 - of Home Computer of the Future.



Scientists from the RAND Corporation have created this model to illustrate how a "home computer" could look like in the year 1984. However the needed technology will not be economically feasible for the average home. Also the scientists readily admit that the computer will require not yet invented technology to actually work, but in years from now scientific progress is expected to solve these problems. With teletype interface and the Fortran language, the computer will be easy to use.

Additional Types of Scenarios (2)

Evaluation scenario:

Description of a user task against which the system is to be evaluated.

Example: Four users (two novice, two experts) play in a TicTac Toe tournament in ARENA.

Training scenario:

A description of the step by step instructions that guide a novice user through a system

Example: How to play Tic Tac Toe in the ARENA Game Framework.

How do we find scenarios?

Don’t expect the client to be verbal if the system does not exist

- Client understands problem domain, not the solution domain.

Don’t wait for information even if the system exists

- “What is obvious does not need to be said”

Engage in a dialectic approach

- You help the client to formulate the requirements
- The client helps you to understand the requirements
- The requirements evolve while the scenarios are being developed

Heuristics for finding scenarios

Ask yourself or the client the following questions:

1. What are the primary tasks that the system needs to perform?
2. What data will the actor create, store, change, remove or add in the system?
3. What external changes does the system need to know about?
4. What changes or events will the actor of the system need to be informed about?

Finding scenarios

Example scenario

Scenario name	Check Balance	
Participating actors	Customer ATM bank	
Flow of event	1	Insert chip card
	2	Provide PIN
	3	Select service (mini statement)
	4	Take the statement
	5	Ask for more service
	6	Take the card

Finding scenarios

Example scenario

Scenario name	Withdraw cash	
Participating actors	Customer ATM bank	
Flow of event	1	Insert chip card
	2	Provide PIN
	3	Select amount
	4	Take the card
	5	Take the cash
	6	Take the receipt

Finding scenarios

Example scenario

Scenario name	`warehouseOnFire`
Participating actor instances	`bob, alice:FieldOfficer` `john:Dispatcher`
Flow of events	1. Bob, driving down main street in his patrol car, notices smoke coming out of a warehouse. His partner, Alice, activates the "Report Emergency" function from her `FRIEND` laptop. 2. Alice enters the address of the building, a brief description of its location (i.e., northwest corner), and an emergency level. In addition to a fire unit, she requests several paramedic units on the scene, given that the area appears to be relatively busy. She confirms her input and waits for an acknowledgment. 3. John, the `Dispatcher`, is alerted to the emergency by a beep of his workstation. He reviews the information submitted by Alice and acknowledges the report. He allocates a fire unit and two paramedic units to the `Incident` site and sends their estimated arrival time (ETA) to Alice. 4. Alice receives the acknowledgment and the ETA.

Figure 4-6 `warehouseOnFire` scenario for the `ReportEmergency` use case.

Identifying Use Cases

A use case specifies all possible scenarios for a given piece of functionality.

A scenario is an instance of a use case

A use case is initiated by an actor. After its initiations a use case may interact with other actors.

Guidelines for Formulation of Use Cases (1)

Name

- Use a verb phrase to name the use case.
- The name should indicate what the user is trying to accomplish.
- Examples:
  - “Request Meeting”, “Schedule Meeting”, “Propose Alternate Date”

Length

- A use case description should not exceed 1-2 pages. If longer, use include relationships.
- A use case should describe a complete set of interactions.

Guidelines for Formulation of Use Cases (2)

Flow of events:

Use the active voice. Steps should start either with "The Actor" or "The System …".

The causal relationship between the steps should be clear.

All flow of events should be described (not only the main flow of event).

The boundaries of the system should be clear. Components external to the system should be described as such.

Define important terms in the glossary.

Example of a badly written Use Case

“The driver arrives at the parking gate, the driver receives a ticket from the distributor, the gate is opened, the driver drives through.”

Example of a badly written Use Case

> “The driver arrives at the parking gate, the driver receives a ticket from the distributor, the gate is opened, the driver drives through.”

It contains no actors

It is not clear which action triggers the ticket being issued
Because of the passive form, it is not clear who opens the gate (The driver? The computer? A gate keeper?)
It is not a complete transaction. A complete transaction would also describe the driver paying for the parking and driving out of the parking lot.

How to write a use case

Name of Use Case	
Actors	Description of Actors involved in use case
Entry condition	“This use case starts when…”
Flow of Events	Free form, informal natural language
Exit condition	“This use cases terminates when…”
Exceptions	Describe what happens if things go wrong
Special Requirements	Nonfunctional Requirements, Constraints

Finding scenarios

Use Case name	Withdraw cash	
Participating actors	Customer ATM bank	
Flow of event	1	Insert chip card
	2	Provide PIN
	3	Select amount
	4	Take the card
	5	Take the cash
	6	Take the receipt
Entry condition	Inserted the card	
Exit conditioner	Received cash	
Special requirement	none	

One use case

Use Case Name		Report Emergency
Entry Condition	1	The FieldOfficer activates the “Report Emergency” function of her terminal.
Flow of Event	2	FRIEND responds by presenting a form to the officer. The form includes an emergency type menu (general emergency, fire, transportation), a location, incident description, resource request, and hazardous material fields.
	3	The FieldOfficer completes the form by specifying minimally the emergency type and description fields. The FieldOfficer may also describe possible responses to the emergency situation and request specific resources. Once the form is completed, the FieldOfficer submits the form by pressing the “Send Report” button, at which point, the Dispatcher is notified.
	4	The Dispatcher reviews the information submitted by the FieldOfficer and creates an Incident in the database by invoking the OpenIncident use case. All the information contained in the FieldOfficer’s form is automatically included in the incident. The Dispatcher selects a response by allocating resources to the incident (with the AllocateResources use case) and acknowledges the emergency report by sending a FRIENDgram to the FieldOfficer.
Exit condition	5	The FieldOfficer receives the acknowledgment and the selected response.

Thank You