Requirements Analysis

Lecture 7

Requirements Elicitation Activities

1. Identifying actors

   (user group.. system support, execute main function, perform secondary function, external hardware software)

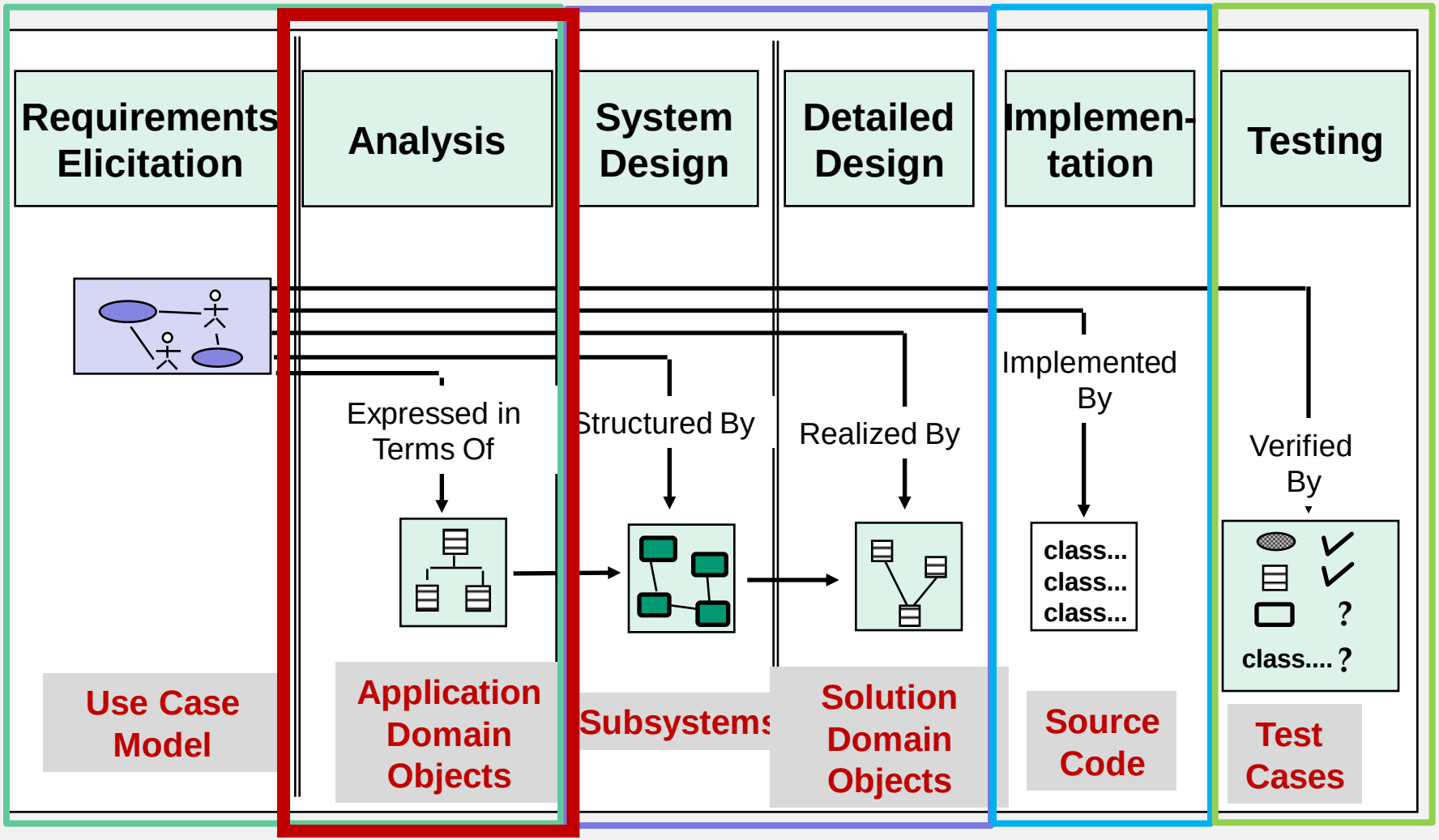
2. Identifying scenarios

   (primary task, data that actors create, store, change, remove, external changes system need to know, event actor need to know).

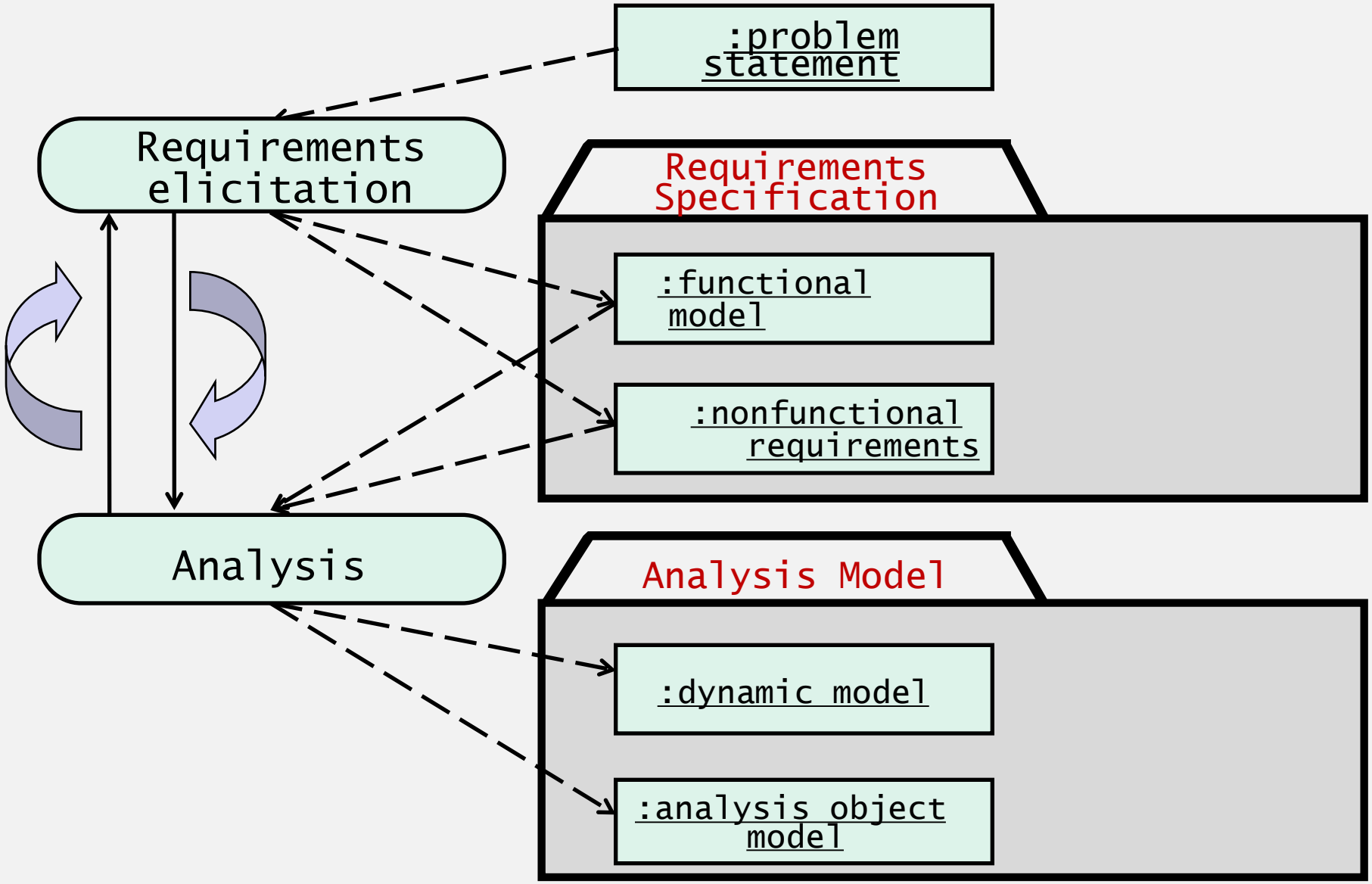
3. Identifying use cases

   (all possible scenario for a given functionality)

Software Lifecycle Activities



Requirements Process



Where are we?

We are in the beginning of Analysis

From use cases to class diagrams

1. Identify participatory objects in flow of events descriptions
2. Identify entity, control and boundary objects

Analysis Activities: From Use Cases to Objects

1	Identifying Entity Objects
2	Identifying boundary Objects
3	Identifying Control Objects
4	Mapping use case to Object with Seq Diagram
5	Modeling interaction among object with CRC
6	Identifying Association
7	Identifying Aggregation
8	Identifying Attributes

Object Identification

Approaches

- Application domain approach
  - Ask application domain experts to identify relevant abstractions
- Syntactic approach
  - Start with use cases
  - Analyze the text to identify the objects
  - Extract participating objects from flow of events
- Design patterns approach
  - Use reusable design patterns
- Component-based approach
  - Identify existing solution classes.

Object identification is a Hard Problem

Problem 1: Definition of the system boundary:

Which abstractions are outside, which abstractions are inside the system boundary?

- Actors are outside the system
- Classes/Objects are inside the system.

Problem 2: Objects are not just found by taking a picture of a scene or domain

The application domain has to be analyzed

Depending on the purpose of the system different objects might be found

- How can we identify the purpose of a system?
- Scenarios and use cases => Functional model

Example: SatWatch

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.

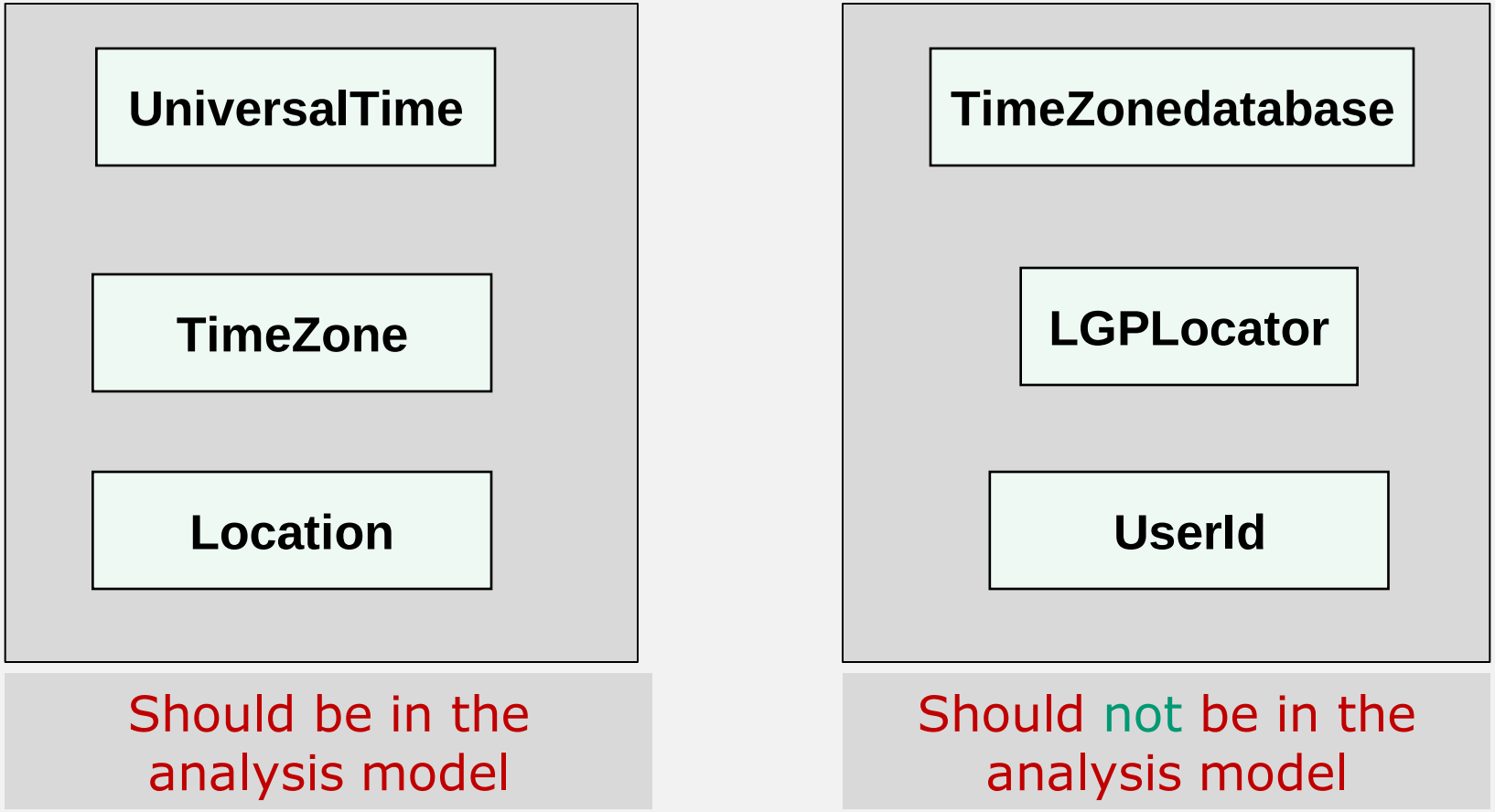
The information stored in SatWatch and its accuracy measuring time is such that the watch owner never needs to reset the time. SatWatch adjusts the time and date displayed as the watch owner crosses time zones and political boundaries. For this reason, SatWatch has no buttons or controls available to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

Example: SatWatch



There are different types of Objects

1. Entity Objects

   Represent the persistent information tracked by the system (Application domain objects, also called “Business objects”)

2. Boundary Objects

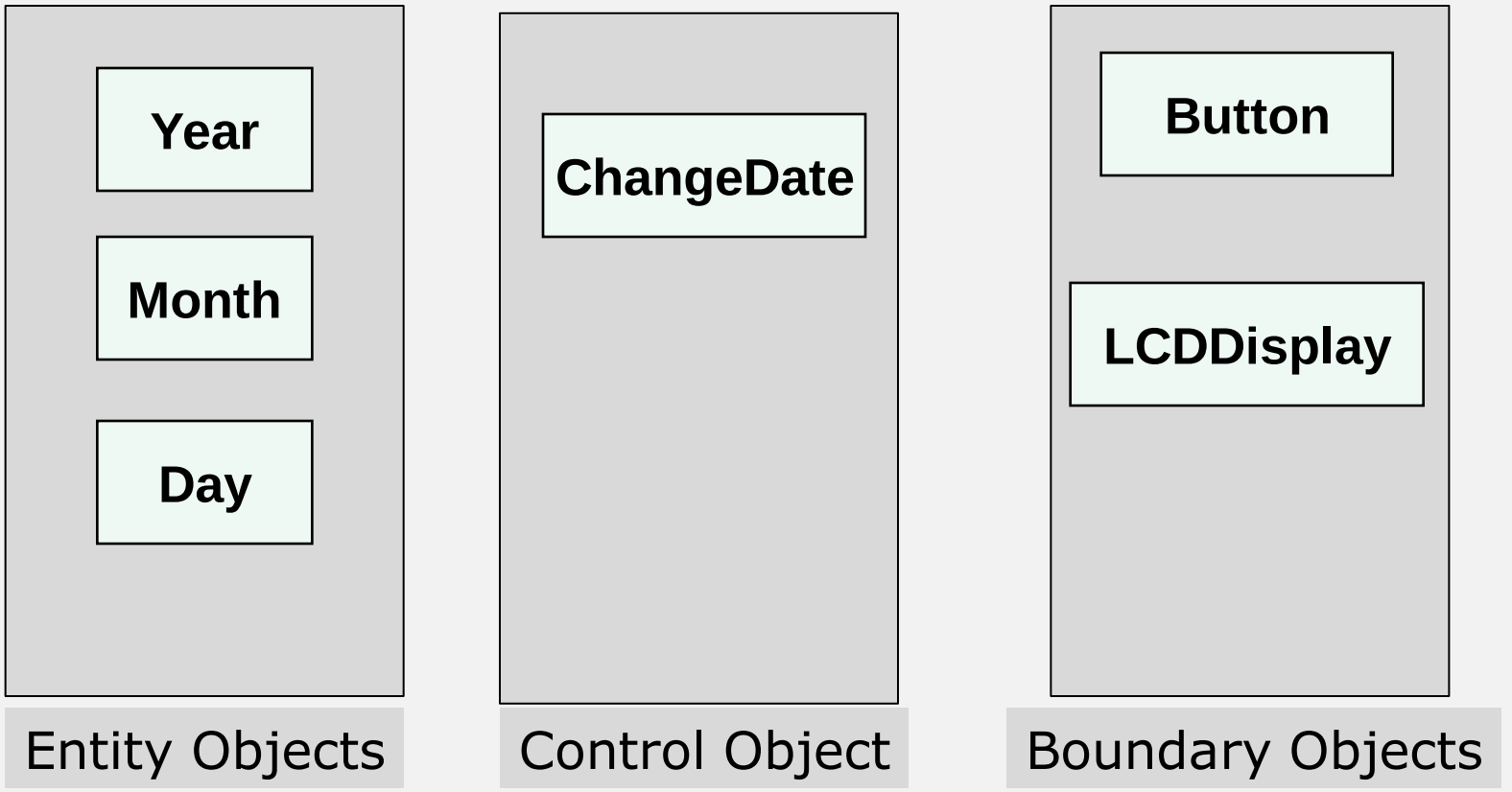
   Represent the interaction between the user and the system

3. Control Objects

   Represent the control tasks performed by the system.

Example: 2BWatch Modeling

To distinguish different object types in a model we can use the UML Stereotype mechanism

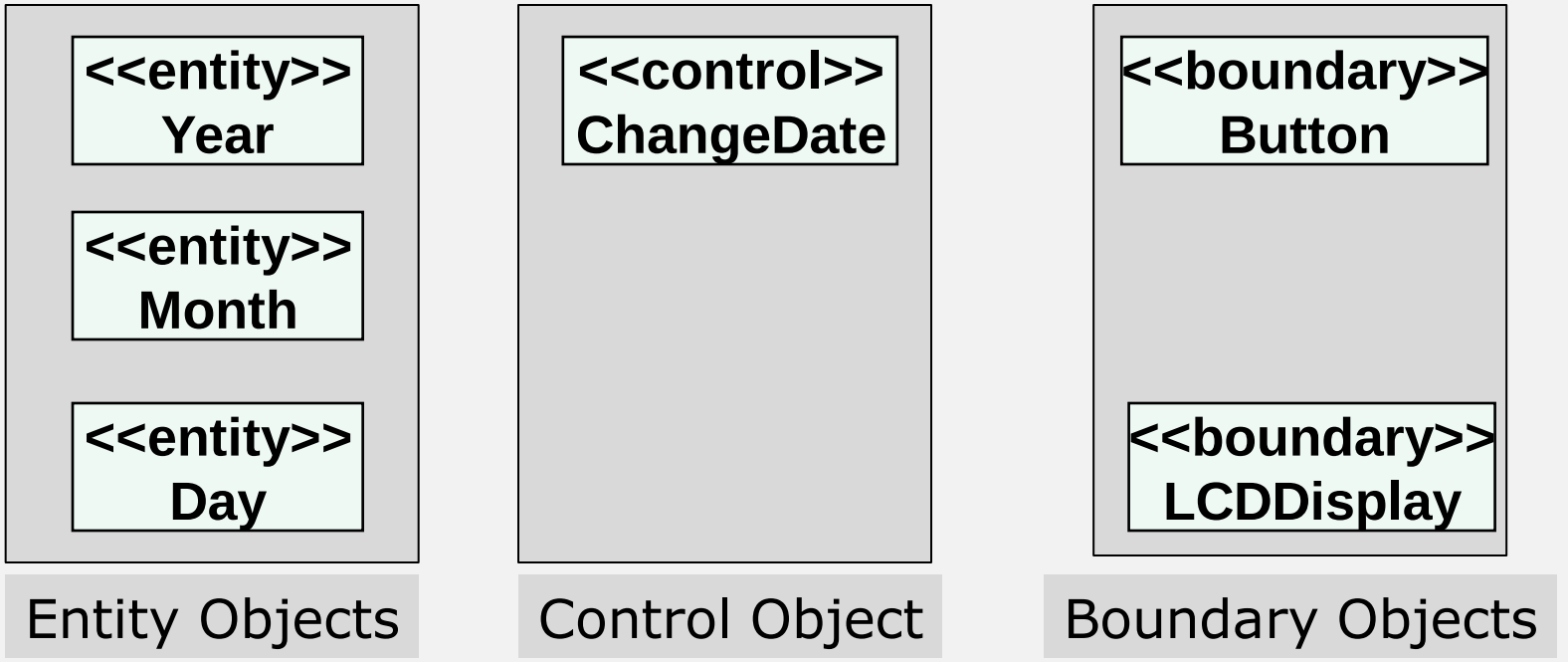


Example: 2BWatch Modeling

UML provides the stereotype mechanism to introduce new types of modeling elements

A stereotype is drawn as a name enclosed by angled double-quotes (“guillemets”) (<<, >>) and placed before the name of a UML element (class, method, attribute, ….)

Notation: <<String>>Name

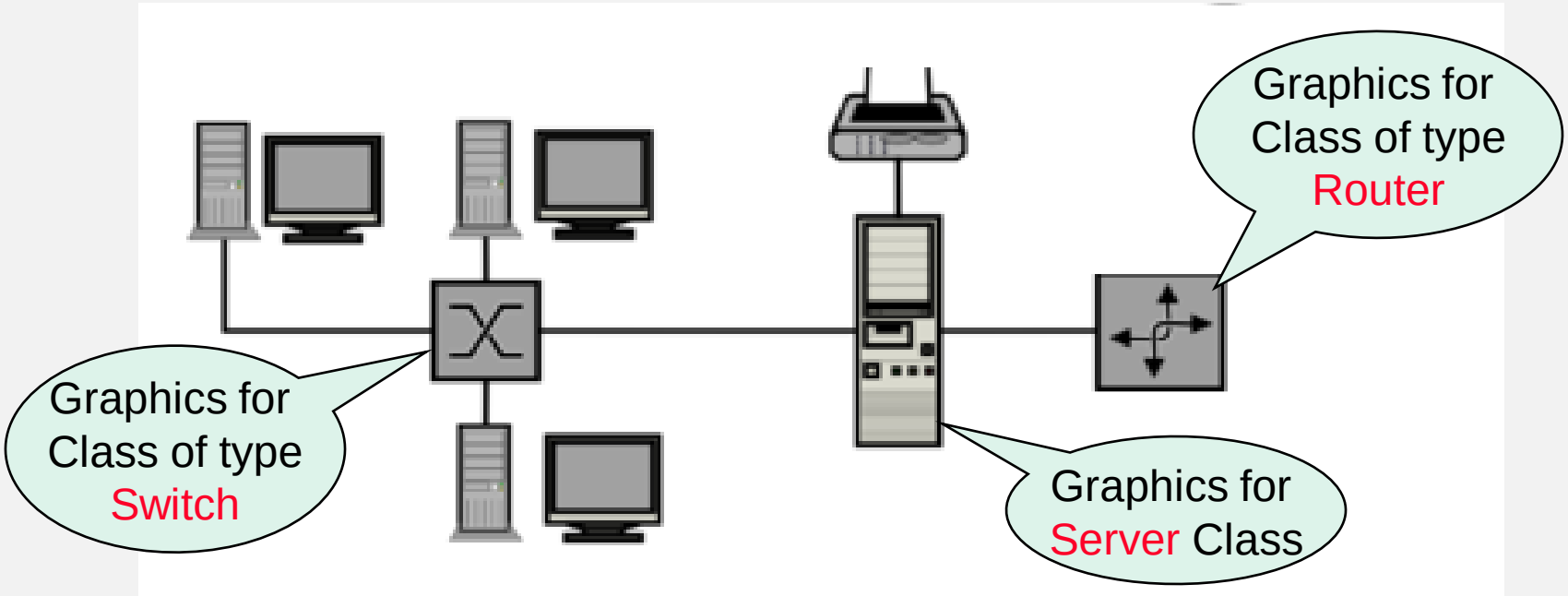


Graphics for Stereotypes

One can also use graphical symbols to identify a stereotype

When the stereotype is applied to a UML model element, the graphic replaces the default graphic for the diagram element.

Example: When modeling a network, define graphics for representing classes of type Switch, Server, Router, Printer,etc.



Object Types allow us to deal with Change

Having three types of object leads to models that are more resilient to change

- The interface of a system changes more likely than the control
- The way the system is controlled changes more likely than entities in the application domain

Finding Participating Objects in Use Cases

Pick a use case and look at flow of events

Do a textual analysis (noun-verb analysis)

- Nouns are candidates for objects
- Verbs are candidates for operations

Heuristics for finding entity object

- Terms that the developers/users need to clarify
- Recurring noun
- Real world entity that the system need to track
- Real world activities that the system need to track
- Data source/sink

Finding Participating Entity Objects

Abotts Heuristic

Maps part of speech to model component

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.

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.

Abotts Heuristic

Part of Speech	Model component	Examples
Proper Noun	Instance	Alice
Common Noun	Class	Field Officer
Doing Verb	Operation	Create, Submit, Selects
Being Verb	Inheritance	Is a kind of, is one of either
Having Verb	Aggregation	Has, consist of, includes
Modal Verb	Constraint	Must be
Adjective	Attribute	Incident description

Finding Participating entity objects

Dispatcher	Police officer who manages Incidents. A Dispatcher opens, documents, and closes Incidents in response to Emergency Reports and other communication with FieldOfficers. Dispatchers are identified by badge numbers.
EmergencyReport	Initial report about an Incident from a FieldOfficer to a Dispatcher. An EmergencyReport usually triggers the creation of an Incident by the Dispatcher. An EmergencyReport is composed of an emergency level, a type (fire, road accident, other), a location, and a description.
FieldOfficer	Police or fire officer on duty. A FieldOfficer can be allocated to, at most, one Incident at a time. FieldOfficers are identified by badge numbers.
Incident	Situation requiring attention from a FieldOfficer. An Incident may be reported in the system by a FieldOfficer or anybody else external to the system. An Incident is composed of a description, a response, a status (open, closed, documented), a location, and a number of FieldOfficers.

Finding Participating Boundary Objects

Heuristics for Funding Boundary Objects	
	Identify user interface controls that the user needs to initiate the use case
	Identify forms the users needs to enter data into the system
	Identify notices and messages the system uses to respond to the user
	When multiple actors are involved in a use case, identify actor terminals to refer to the user interface under consideration
	Do not model the visual aspects of the interface with boundary objects (user mock-ups are better suited for that)
	Always use the end user's terms for describing interfaces; do not use terms from the solution or implementation domains.

Finding Participating Boundary objects

FieldOfficerStation	Mobile computer used by the FieldOfficer.
ReportEmergencyButton	Button used by a FieldOfficer to initiate the ReportEmergency use case.
EmergencyReportForm	Form used for the input of the ReportEmergency. This form is presented to the FieldOfficer on the FieldOfficerStation when the “Report Emergency” function is selected. The EmergencyReportForm contains fields for specifying all attributes of an emergency report and a button (or other control) for submitting the completed form.
IncidentForm	Form used for the creation of Incidents. This form is presented to the Dispatcher on the DispatcherStation when the EmergencyReport is received. The Dispatcher also uses this form to allocate resources and to acknowledge the FieldOfficer’s report.
AcknowledgmentNotice	Notice used for displaying the Dispatcher’s acknowledgment to the FieldOfficer.

Finding Participating Control Objects

Heuristics for Funding Control Objects	
	Identify one control object per use case.
	Identify one control object per actor in the use case.
	The life span of a control object should cover the extent of the use case or the extent of a user session.

Finding Participating Control objects

ReportEmergencyControl	Mobile computer used by the FieldOfficer.
ManageEmergencyControl	Button used by a FieldOfficer to initiate the ReportEmergency use case.

Example for using the Technique

Flow of Events:

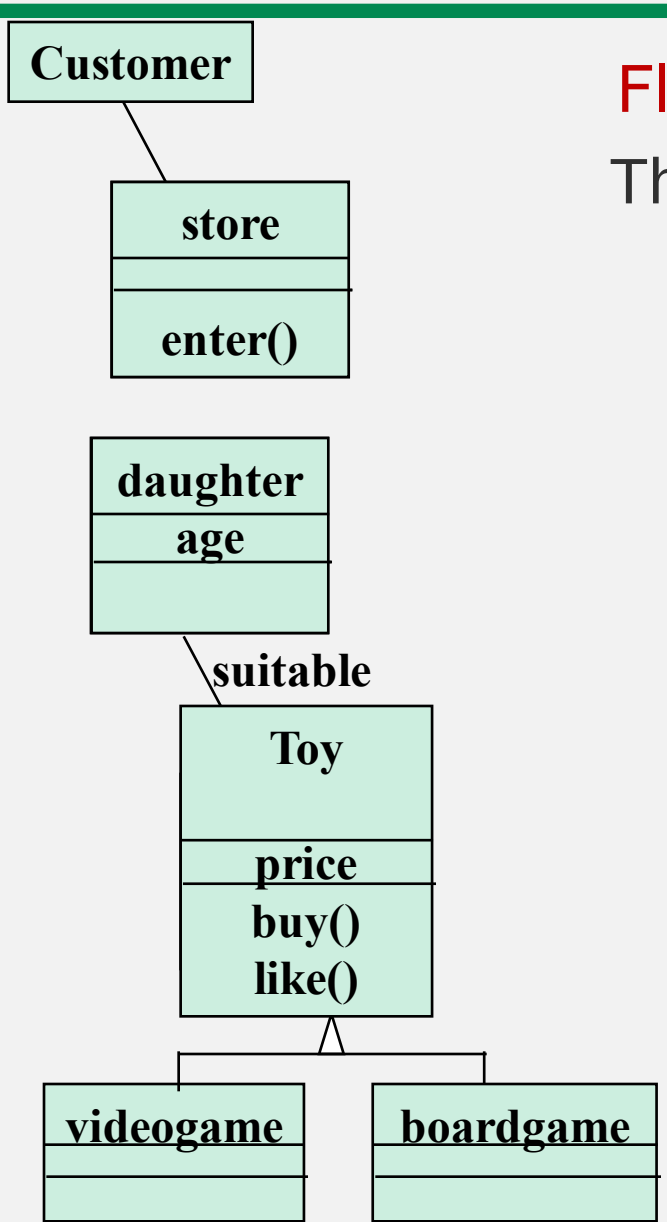
The customer enters the store to buy a toy. It has to be a toy that his daughter likes and it must cost less than 50 Euro. He tries a videogame, which uses a data glove and a head-mounted display. He likes it.

An assistant helps him. The suitability of the game depends on the age of the child. His daughter is only 3 years old. The assistant recommends another type of toy, namely the boardgame “Monopoly".

Mapping parts of speech to model components

Example	Part of Speech	Model component
Monopoly	Proper noun	Object
Toy	Improper noun	Class
Buy, recommend	Doing verb	Operation
Is a	Being verb	Inheritance
Has an	Having verb	Aggregation
Must be	Modal verb	Constraint
Dangerous	Adjective	Attribute
Enter	Transitive verb	Operation
depends	Intransitive verb	Association, constraint

Generating a Class Diagram from Flow of Events



Flow of events:

The customer enters the store to buy a toy. It has to be a toy that his daughter likes and it must cost less than 50 Euro. He tries a videogame, which uses a data glove and a head-mounted display. He likes it.

An assistant helps him. The suitability of the game depends on the age of the child. His daughter is only 3 years old. The assistant recommends another type of toy, namely a boardgame. The customer buy the game and leaves the store

Who uses Class Diagrams?

Purpose of class diagrams

The description of the static properties of a system

The main users of class diagrams:

The application domain expert

- uses class diagrams to model the application domain (including taxonomies)
  - during requirements elicitation and analysis

The developer

- uses class diagrams during the development of a system
  - during analysis, system design, object design and implementation.

Who does not use Class Diagrams?

- The client and the end user are usually not interested in class diagrams
  - Clients focus more on project management issues
  - End users are more interested in the functionality of the system.

Summary

System modeling

- Functional modeling+object modeling+dynamic modeling

Functional modeling

- From scenarios to use cases to objects

Object modeling is the central activity

- Class identification is a major activity of object modeling
- Easy syntactic rules to find classes and objects

- Abbot's Technique

Class diagrams are the "center of the universe" for the object-oriented developer

- The end user focuses more on the functional model and and usability.

Thank You