Design Patterns and UML Modeling Practice

Dániel Varró / Kristian Sandahl

LINKÖPING UNIVERSITY

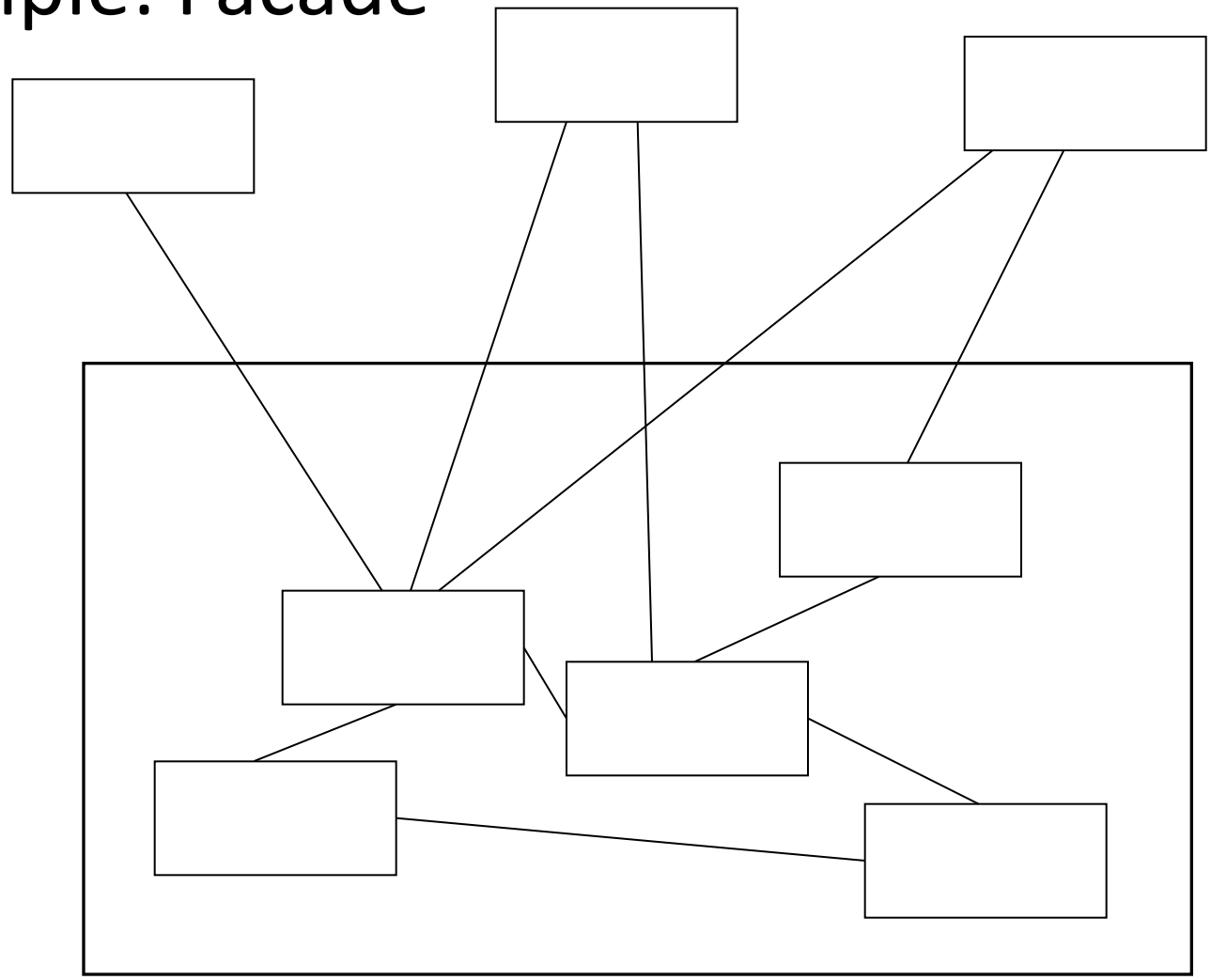
Design Patterns

Software Design Patterns

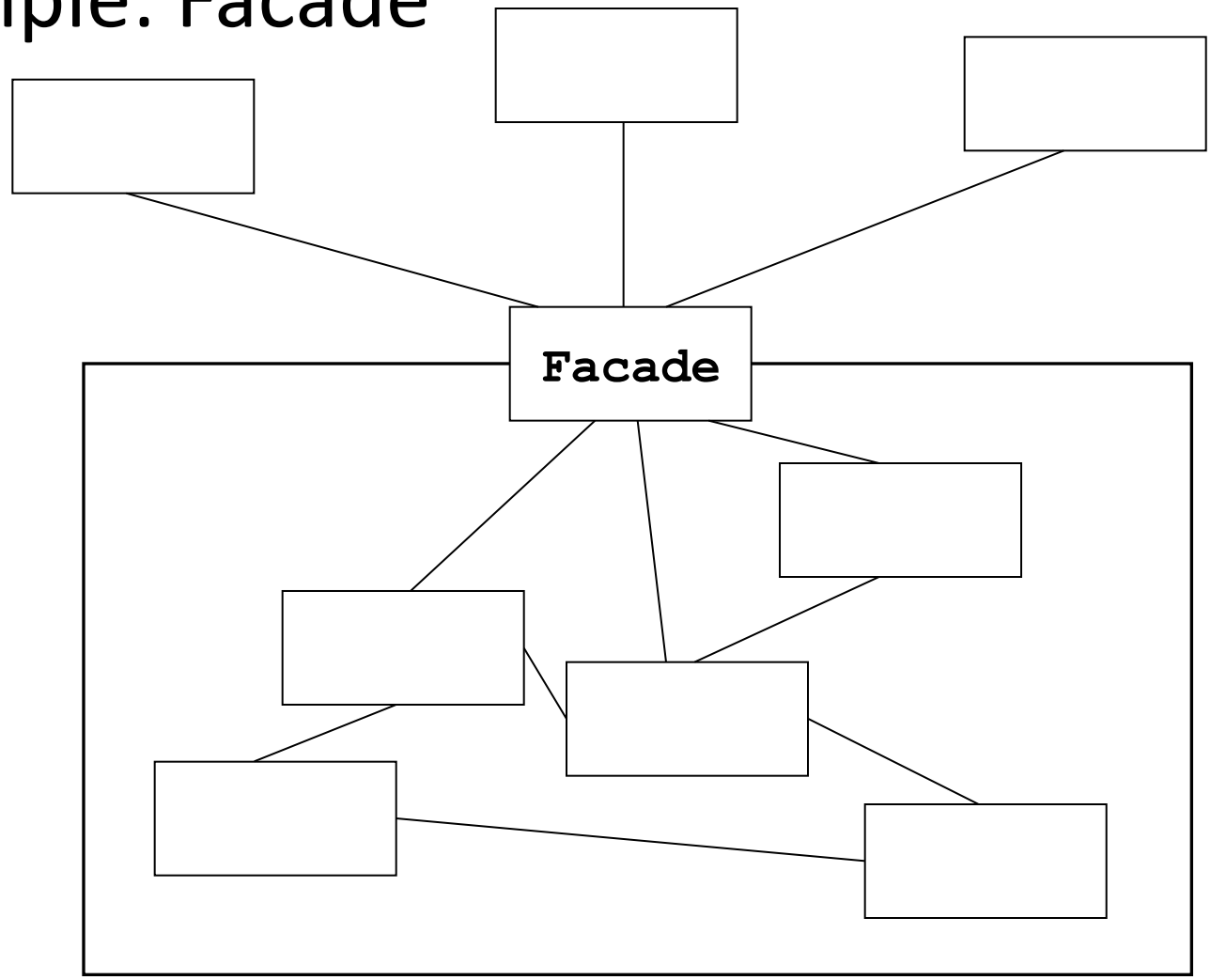
A Design Pattern is a standard solution for a standard design problem in a certain context.

Goal: reuse design information

Example: Facade

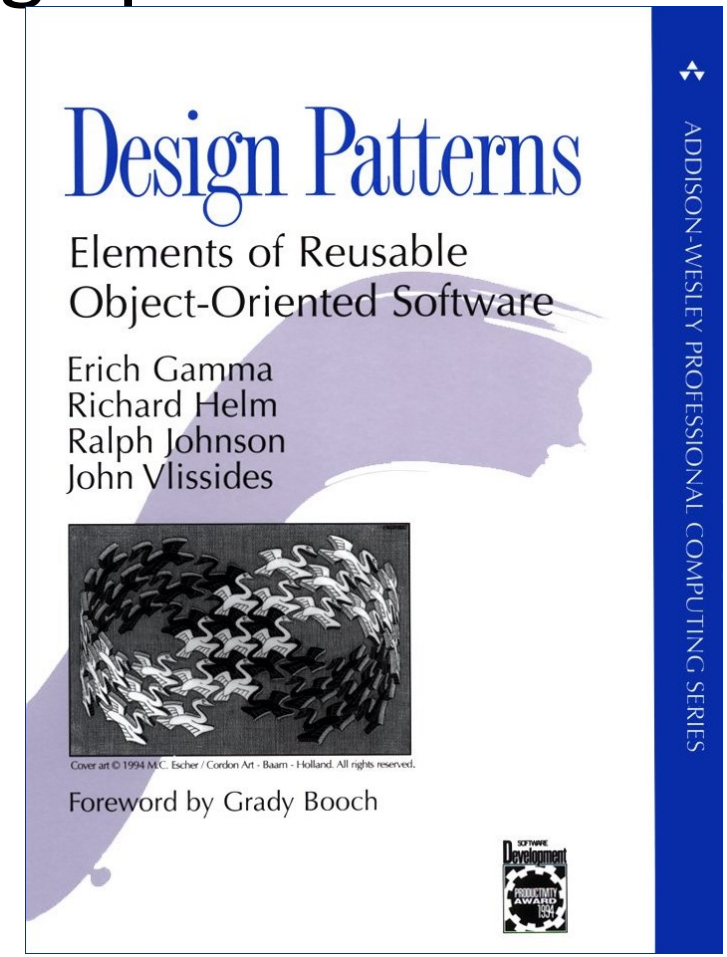


Example: Facade



How to describe design patterns?

- Gangs of the Four (GoF) book



Facade

Intent

Provide a **unified interface** to a set of interfaces in a subsystem. Facade defines a higher-level interface that makes the **subsystem easier to use**.

Motivation

Structuring a system into subsystems helps **reduce complexity**. A common design goal is to **minimize the communication and dependencies** between subsystems. … example …

Facade

Applicability

Use the Facade pattern when:

- you want to provide a **simple interface** to a complex subsystem. This makes subsystems more **reusable** and easier to **customize**.
- there are **many dependencies** between clients and the implementation classes of an abstraction. Introduce a facade to **decouple** the subsystem from other subsystems, thereby promoting subsystem **independence and portability**.
- you want to **layer** your subsystems. Use a facade to define an entry point to each subsystem level.

Facade

Consequences

The Facade pattern offers the following benefits:

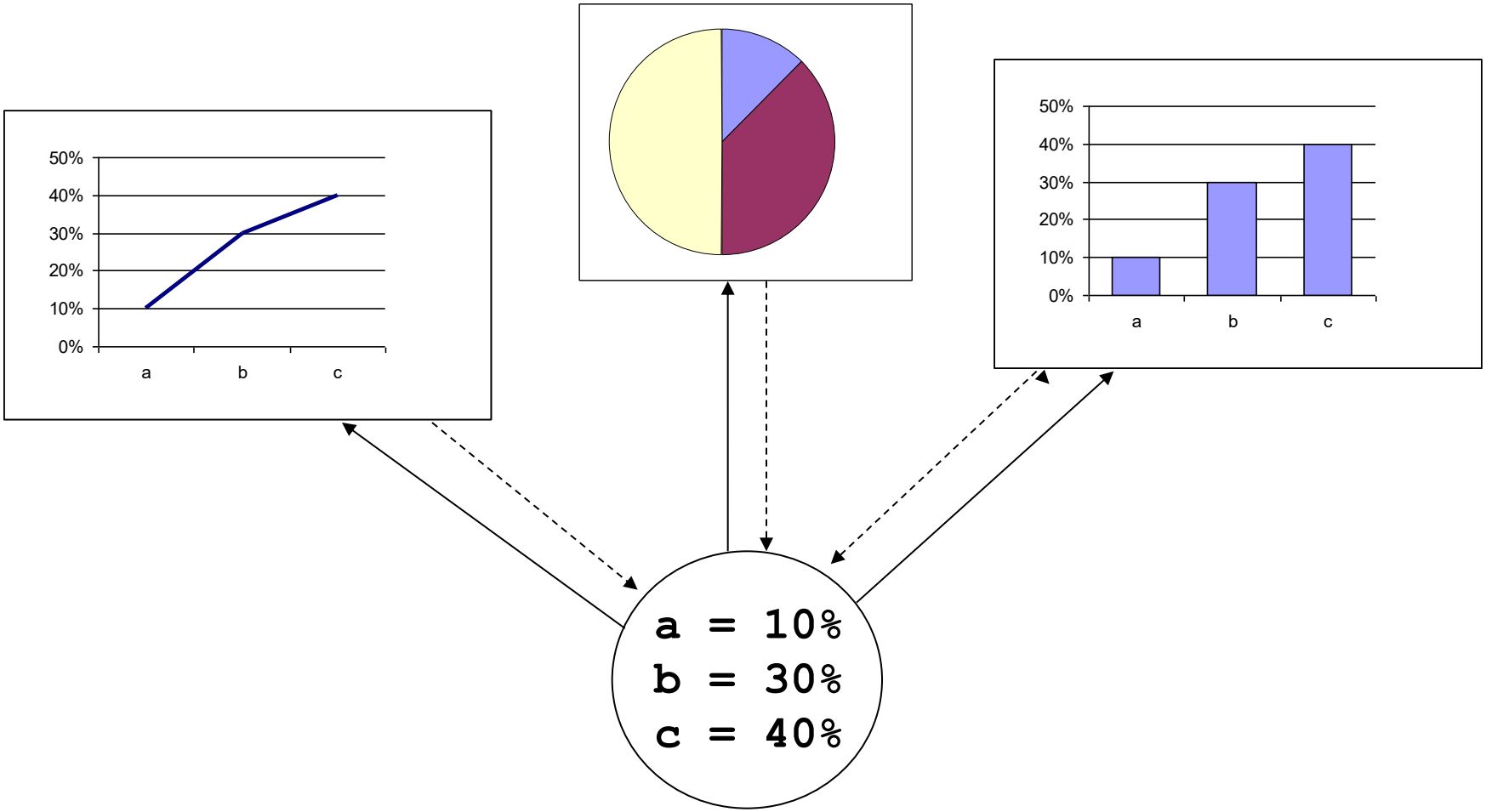
1. It **shields clients from subsystem** components, thereby reducing the number of objects that clients deal with and making subsystem easier to use.
2. It promotes **weak coupling** between subsystem and its clients. Weak coupling lets you vary the components of the subsystem without affecting its clients.
3. It doesn't prevent applications from using subsystem classes if they need to.

Facade

- **Structure**
- **Participants**
- **Collaborations**
- **Implementation**
- **Sample Code**
- **Known Uses**
- **Related Patterns**

Design Patterns
Observer

Observer

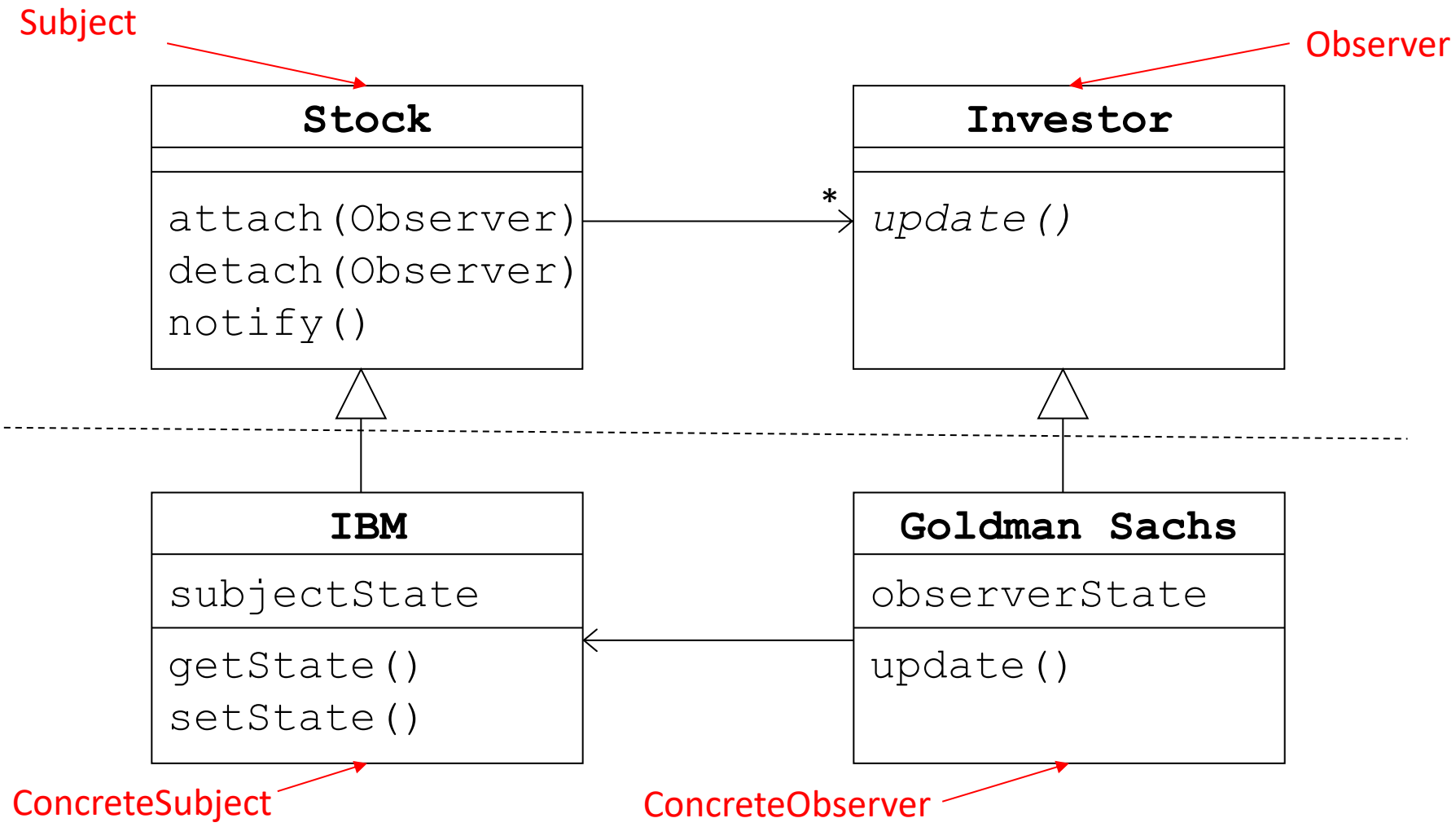


Observer

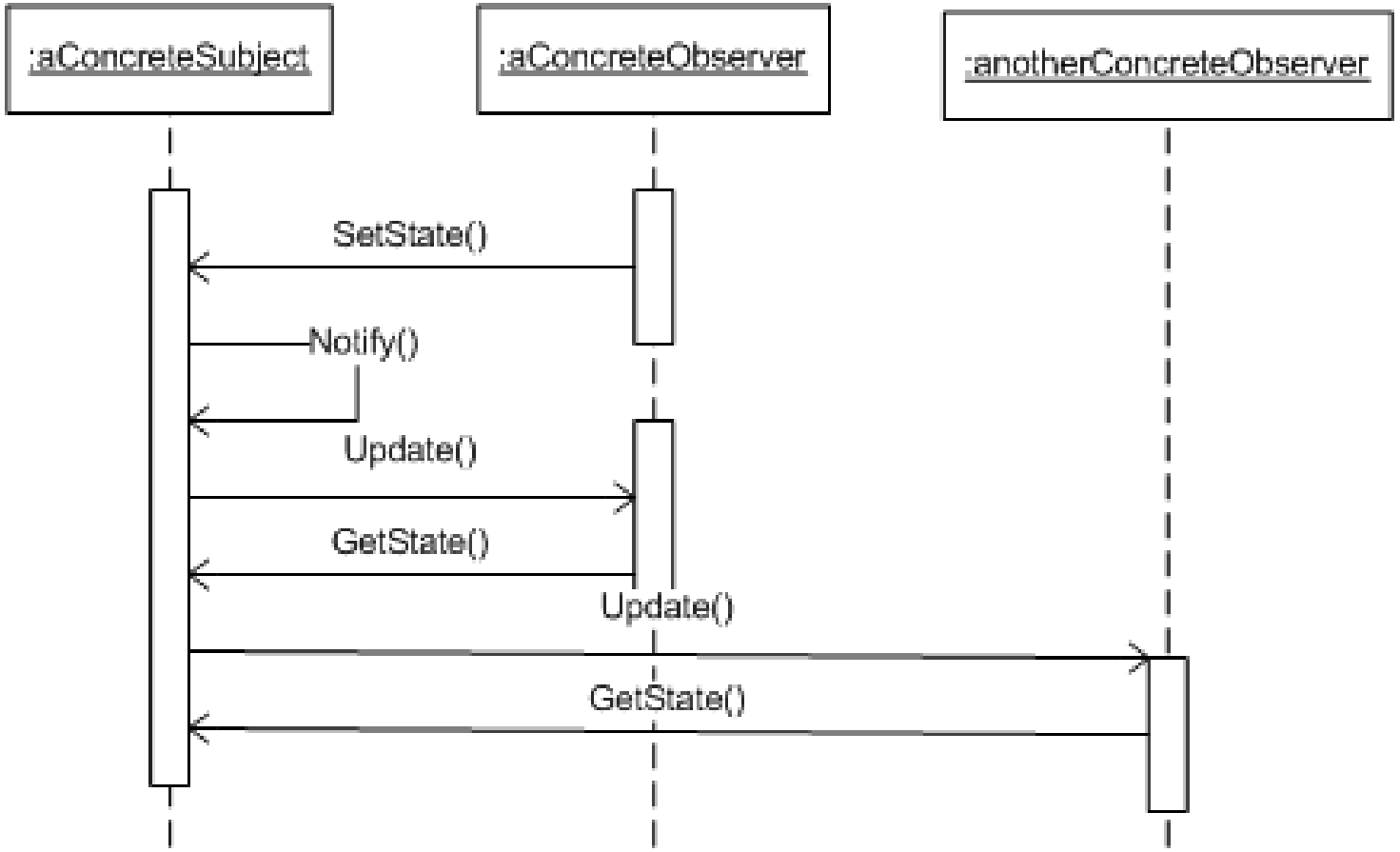
Applicability

- When an abstraction has **two aspects**, one dependent on the other.
- When a **change to one object requires changing others**.
- When an object should be able to notify other objects **without making assumptions** about who these objects are.

Observer, structure



Observer, collaborations



Observer, consequences

- Abstract coupling between Subject and Observer
- Support for broadcast communication
- Unexpected updates

Design Patterns
Strategy

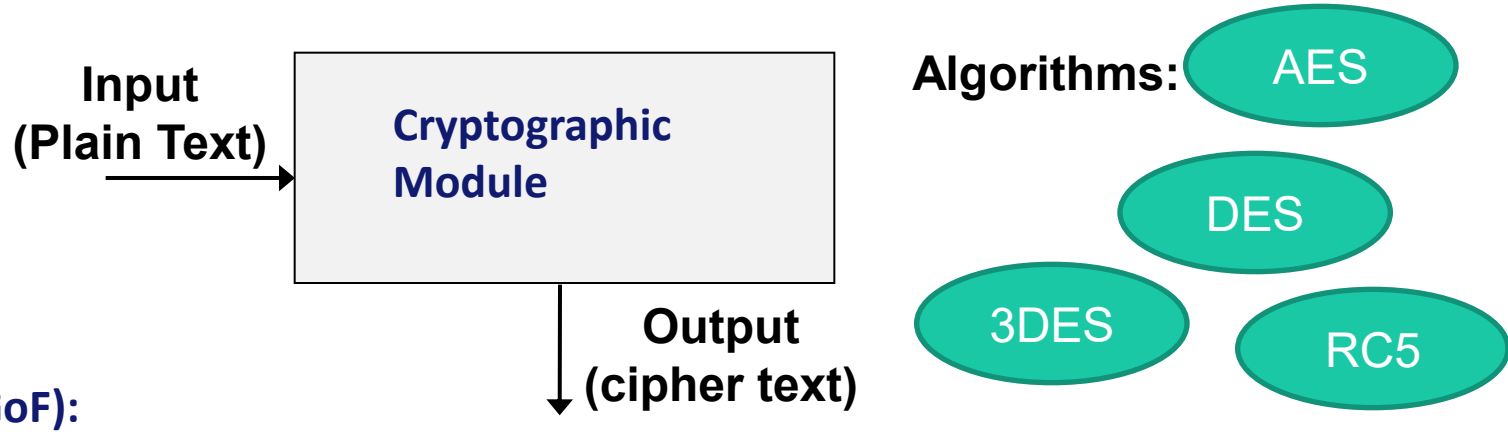
Strategy

Name: Strategy **Also known as:** Policy

Problem:

- Need to use different variants of the same algorithm in a class
- Different algorithms will be appropriate at different time.
- It is hard to add new algorithms and to change existing ones.

Example:

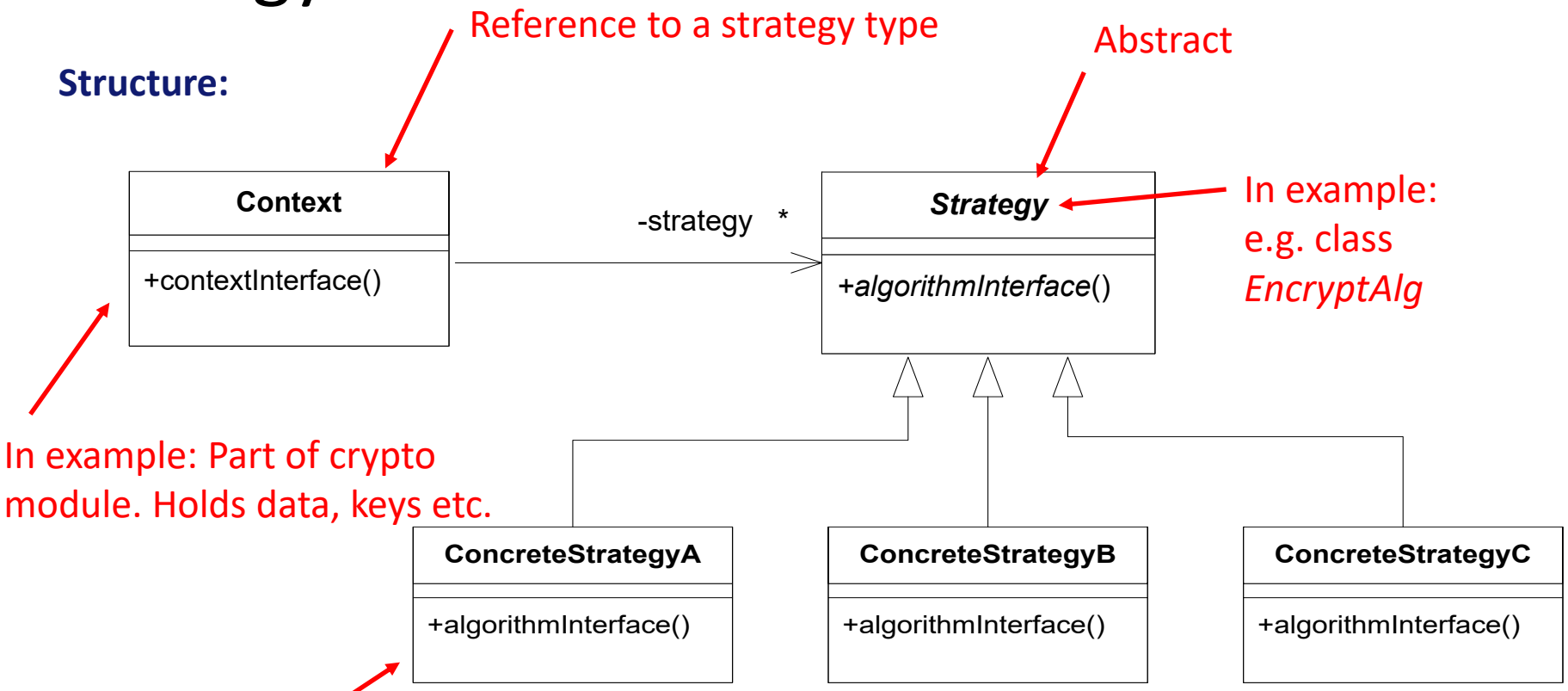


Intent (from GoF):

"Define a family of algorithms, encapsulate each one and make them interchangeable. Strategy lets the algorithm vary independently from clients that use it."

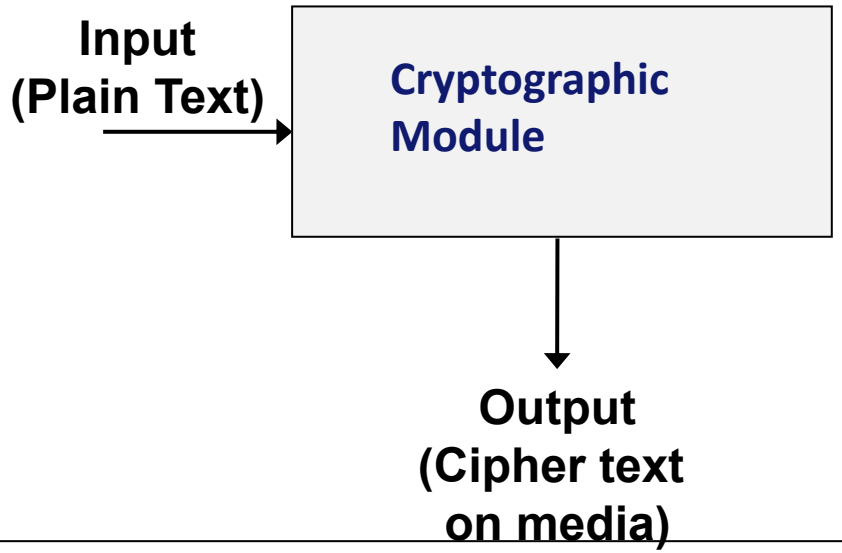
Strategy

Structure:

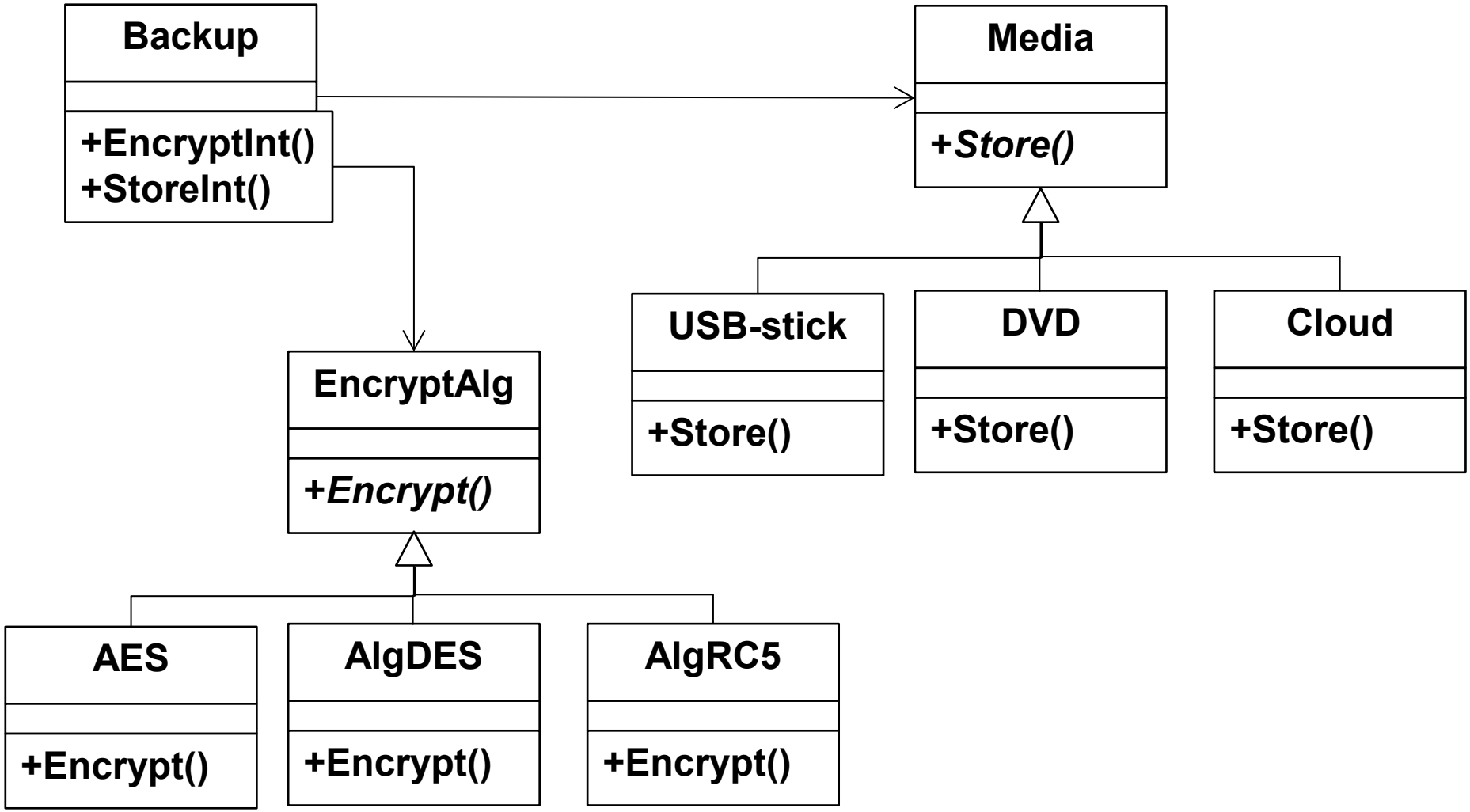


Strategy

- Suppose we add a new strategy:
- Storage media:
  - Disc
  - USB-stick
  - DVD
  - Cloud
    - ….



Two strategies



Summary

- Design patterns: Façade, observer, strategy
- More of this in course:
- TDDE45 Software Design and Construction

UML Modeling Practice /
Dániel Varró & Kristian Sandahl

www.liu.se

LINKÖPING UNIVERSITY

A Domain Modeling Challenge

REMS: Requirements

A review management system (REMS) help the review of scientific journal papers submitted by researchers. Authors submit a paper by using a form to specify a title, an abstract, a list of keywords and a first version as PDF document. They may also suggest names for excluded reviewers. When a new submission is received, REMS assigns a qualified editor to manage its review process by matching the keywords of the paper with editors' expertise. An editor sends invitation to several reviewers (not excluded by the authors) who either accept or decline this invitation. When two reviewers agree to review the paper, no further reviewers will be invited. A reviewer needs to complete a review which includes a textual critic and a recommendation: accept, minor revision, major revision or reject. Based upon the recommendations of the reviews, the editor makes a decision on the paper (which is also one of accept, minor revision, major revision and reject). If the decision is major revision, the authors need to resubmit a revised version of the paper, and the editor initiates a 2nd round of review, which is identical with the 1st round, except for excluding major revision as a possible outcome.

Write a **functional requirement** to capture that *only qualified editors will handle any paper*.
Write an **non-functional requirement** on *the availability* of the REMS system.

Draw a ***use case diagram*** for the REMS system highlighting key actors, use cases and their relations.

Draw a UML Class Diagram as domain model for the REMS system showing the domain concepts, their relationships and potential generalizations. Specify multiplicities for your associations and arrange all objects into a containment hierarchy by appropriate composition relations between classes.

Describe the **high-level workflow** of the *paper review process* using a UML Activity Diagram. You may assume that the successful invitation of a reviewer is separated into an activity called *Invite-and-Accept-Review* which you may use in your diagram. Your actions should have direct traceability to use cases!

Describe the **state-based behavior** of the "*Paper*" class by a UML Statechart Diagram. Use operations derived from use cases as triggering events of transitions. (The Paper class represents a submission that is handled by REMS for review.)

REMS: Requirements

A review management system (REMS) help the review of scientific journal papers submitted by researchers. Authors submit a paper by using a form to specify a title, an abstract, a list of keywords and a first version as PDF document. They may also suggest names for excluded reviewers. When a new submission is received, REMS assigns a qualified editor to manage its review process by matching the keywords of the paper with editors' expertise. An editor sends invitation to several reviewers (not excluded by the authors) who either accept or decline this invitation. When two reviewers agree to review the paper, no further reviewers will be invited. A reviewer needs to complete a review which includes a textual critic and a recommendation: accept, minor revision, major revision or reject. Based upon the recommendations of the reviews, the editor makes a decision on the paper (which is also one of accept, minor revision, major revision and reject). If the decision is major revision, the authors need to resubmit a revised version of the paper, and the editor initiates a 2nd round of review, which is identical with the 1st round, except for excluding major revision as a possible outcome.

Write a **functional requirement** to capture that *only qualified editors will handle any paper*.
Write an **non-functional requirement** on *the availability* of the REMS system.

REMS: Requirements

- Functional Requirement:
  - The **REMS system** ***shall*** automatically **select a reviewer** whose **expertise matches the keywords provided for a paper**.
- Non-functional Requirement:
  - The down-time of the REMS system should be less than 24 hours a year.
  - The availability of the REMS system should be 99.99%.

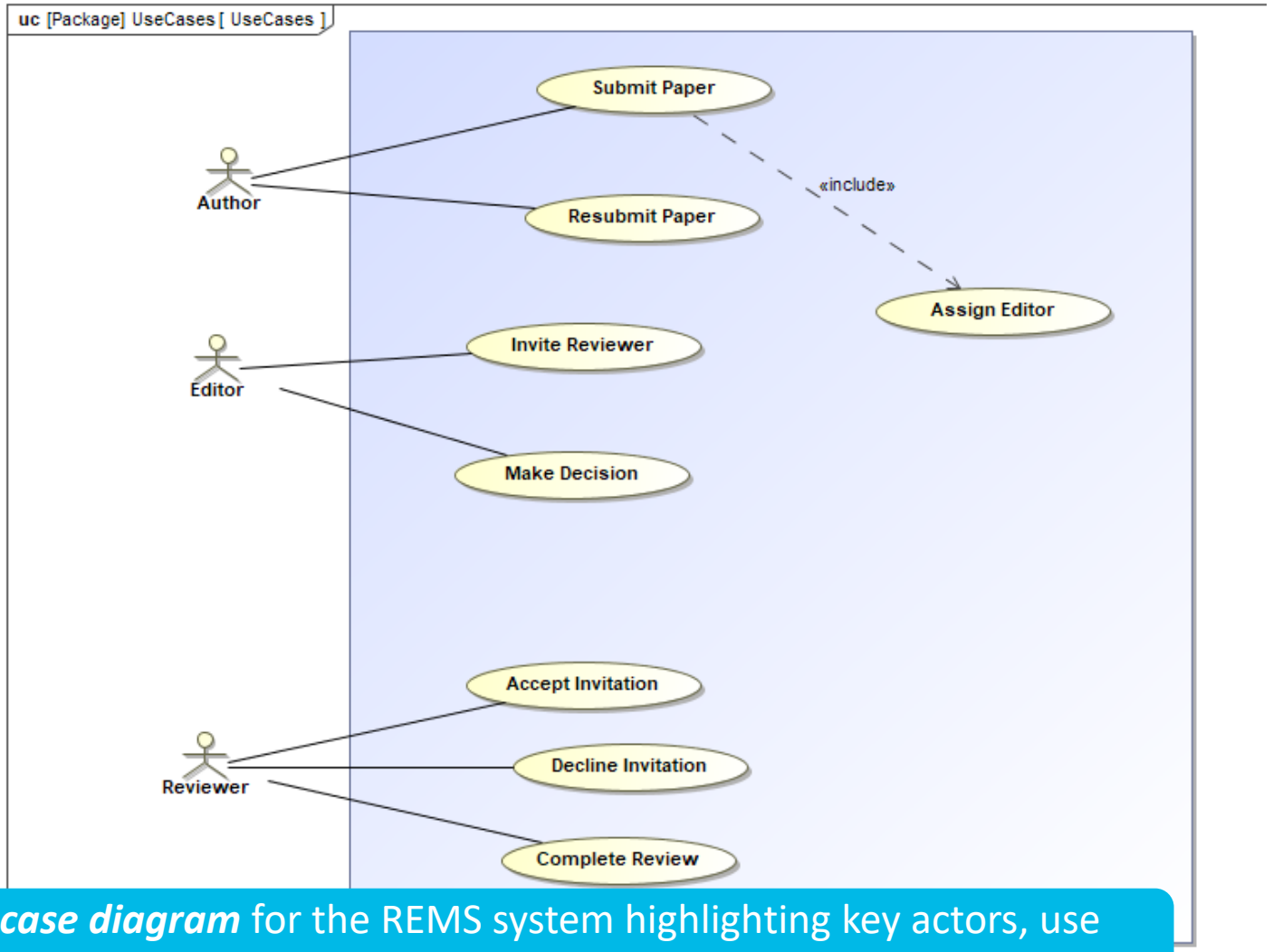
Write a **functional requirement** to capture that *only qualified editors will handle any paper*.
Write an **non-functional requirement** on *the availability* of the REMS system.

REMS: Use Cases

A review management system (REMS) help the review of scientific journal papers submitted by researchers. Authors submit a paper by using a form to specify a title, an abstract, a list of keywords and a first version as PDF document. They may also suggest names for excluded reviewers. When a new submission is received, REMS assigns a qualified editor to manage its review process by matching the keywords of the paper with editors' expertise. An editor sends invitation to several reviewers (not excluded by the authors) who either accept or decline this invitation. When two reviewers agree to review the paper, no further reviewers will be invited. A reviewer needs to complete a review which includes a textual critic and a recommendation: accept, minor revision, major revision or reject. Based upon the recommendations of the reviews, the editor makes a decision on the paper (which is also one of accept, minor revision, major revision and reject). If the decision is major revision, the authors need to resubmit a revised version of the paper, and the editor initiates a 2nd round of review, which is identical with the 1st round, except for excluding major revision as a possible outcome.

Draw a ***use case diagram*** for the REMS system highlighting key actors, use cases and their relations.

REMS: U



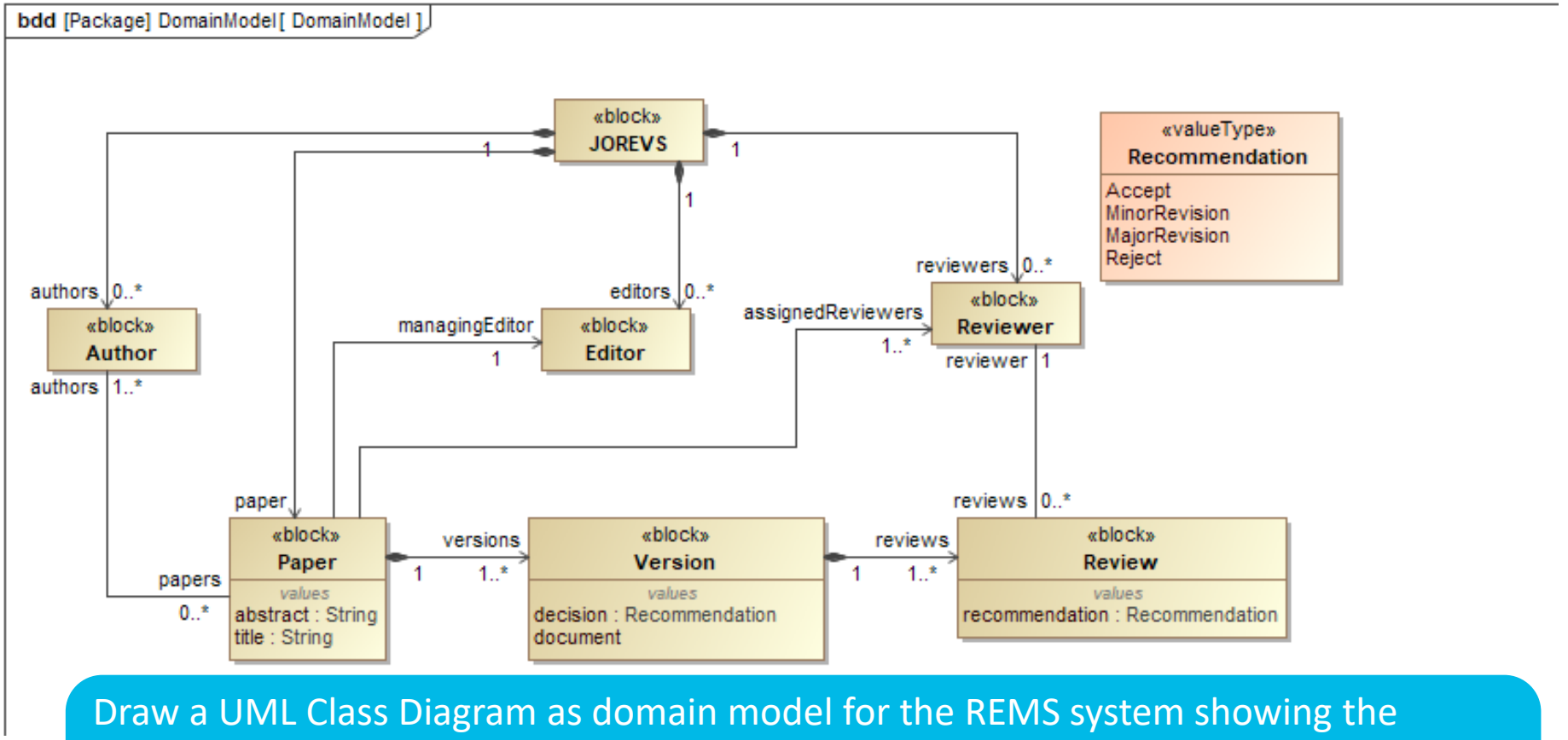
Draw a ***use case diagram*** for the REMS system highlighting key actors, use cases and their relations.

REMS: Domain Model / Class Diagram

A review management system (REMS) help the review of scientific journal papers submitted by researchers. Authors submit a paper by using a form to specify a title, an abstract, a list of keywords and a first version as PDF document. They may also suggest names for excluded reviewers. When a new submission is received, REMS assigns a qualified editor to manage its review process by matching the keywords of the paper with editors' expertise. An editor sends invitation to several reviewers (not excluded by the authors) who either accept or decline this invitation. When two reviewers agree to review the paper, no further reviewers will be invited. A reviewer needs to complete a review which includes a textual critic and a recommendation: accept, minor revision, major revision or reject. Based upon the recommendations of the reviews, the editor makes a decision on the paper (which is also one of accept, minor revision, major revision and reject). If the decision is major revision, the authors need to resubmit a revised version of the paper, and the editor initiates a 2nd round of review, which is identical with the 1st round, except for excluding major revision as a possible outcome.

Draw a UML Class Diagram as domain model for the REMS system showing the domain concepts and their relationships as well as potential generalizations. Specify multiplicities for your associations and arrange all objects into a containment hierarchy by appropriate composition relations between classes.

REMS: Domain Model / Class Diagram



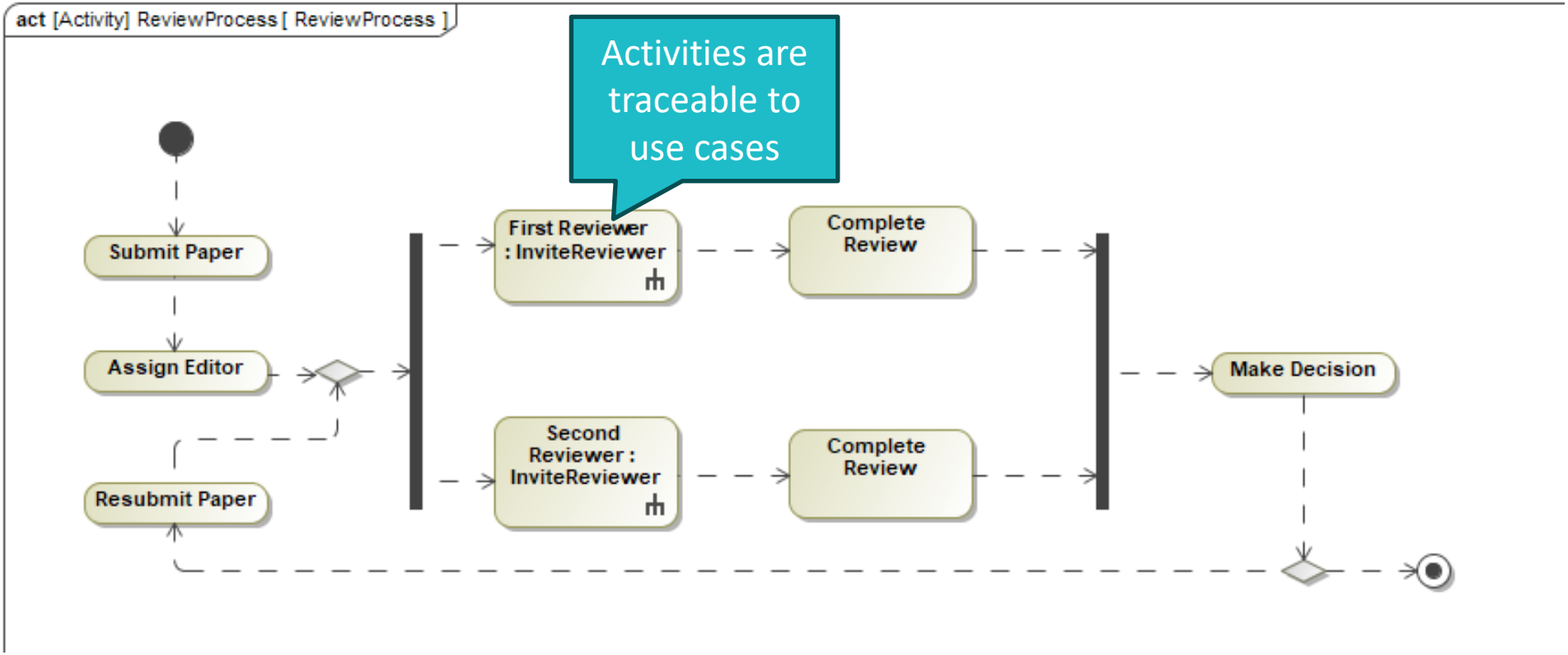
Draw a UML Class Diagram as domain model for the REMS system showing the domain concepts and their relationships as well as potential generalizations. Specify multiplicities for your associations and arrange all objects into a containment hierarchy by appropriate composition relations between classes.

REMS: Workflow by Activity Diagram

A review management system (REMS) help the review of scientific journal papers submitted by researchers. Authors submit a paper by using a form to specify a title, an abstract, a list of keywords and a first version as PDF document. They may also suggest names for excluded reviewers. When a new submission is received, REMS assigns a qualified editor to manage its review process by matching the keywords of the paper with editors' expertise. An editor sends invitation to several reviewers (not excluded by the authors) who either accept or decline this invitation. When two reviewers agree to review the paper, no further reviewers will be invited. A reviewer needs to complete a review which includes a textual critic and a recommendation: accept, minor revision, major revision or reject. Based upon the recommendations of the reviews, the editor makes a decision on the paper (which is also one of accept, minor revision, major revision and reject). If the decision is major revision, the authors need to resubmit a revised version of the paper, and the editor initiates a 2nd round of review, which is identical with the 1st round, except for excluding major revision as a possible outcome.

Describe the **high-level workflow** of the *paper review process* using a UML Activity Diagram. You may assume that the successful invitation of a reviewer is separated into an activity called *Invite-and-Accept-Review* which you may use in your diagram. Your actions should have direct traceability to use cases!

REMS: Workflow by Activity Diagram



Describe the **high-level workflow** of the *paper review process* using a UML Activity Diagram. You may assume that the successful invitation of a reviewer is separated into an activity called *Invite-and-Accept-Review* which you may use in your diagram. Your actions should have direct traceability to use cases!

REMS: State machine diagrams

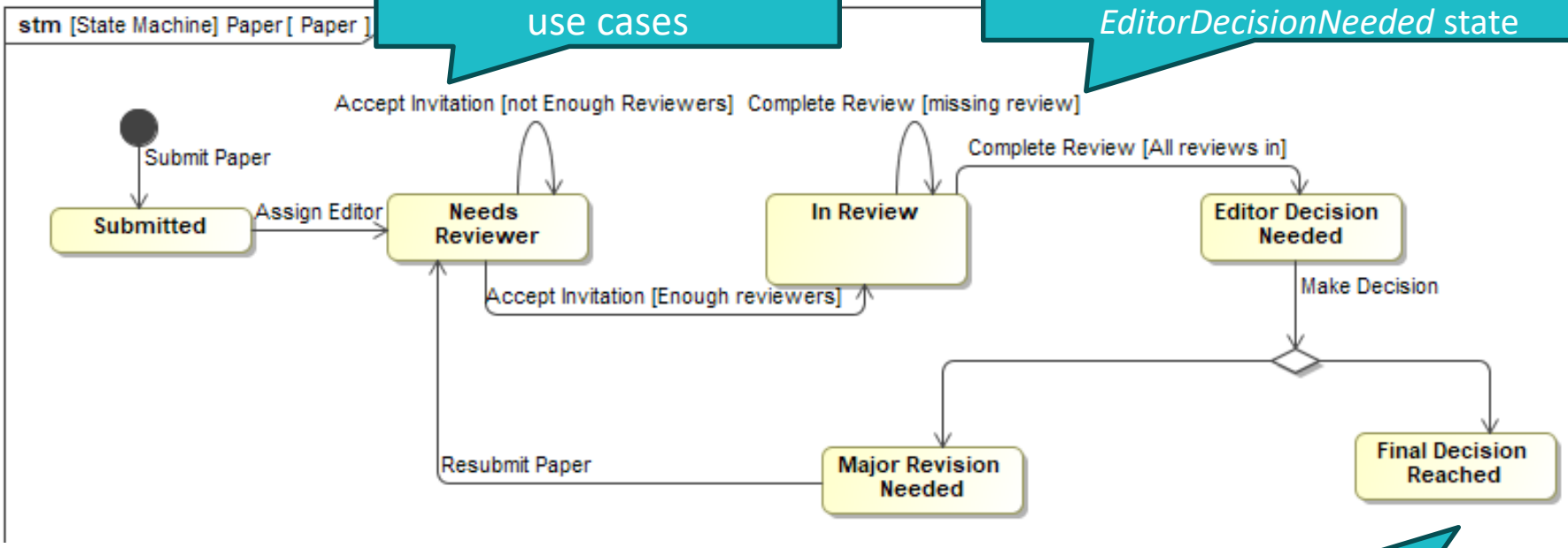
A review management system (REMS) help the review of scientific journal papers submitted by researchers. Authors submit a paper by using a form to specify a title, an abstract, a list of keywords and a first version as PDF document. They may also suggest names for excluded reviewers. When a new submission is received, REMS assigns a qualified editor to manage its review process by matching the keywords of the paper with editors' expertise. An editor sends invitation to several reviewers (not excluded by the authors) who either accept or decline this invitation. When two reviewers agree to review the paper, no further reviewers will be invited. A reviewer needs to complete a review which includes a textual critic and a recommendation: accept, minor revision, major revision or reject. Based upon the recommendations of the reviews, the editor makes a decision on the paper (which is also one of accept, minor revision, major revision and reject). If the decision is major revision, the authors need to resubmit a revised version of the paper, and the editor initiates a 2nd round of review, which is identical with the 1st round, except for excluding major revision as a possible outcome.

Describe the **state-based behavior** of the "*Paper*" class by a UML Statechart Diagram. Use operations derived from use cases as triggering events of transitions.
(The Paper class represents a submission that is handled by REMS for review.)

REMS: State machine diag

Triggering events are often traceable to use cases

When a Complete Review trigger arrives, if there are missing reviews for the paper than it stays in the *InReview* state otherwise, it moves into the *EditorDecisionNeeded* state



When a paper is rejected or accepted, the state can be the same (as this information is reflected in the Class Diagram)