Unified Process

Architecture centric

Anders Kalhauge - Rolf-Helge Pfeiffer

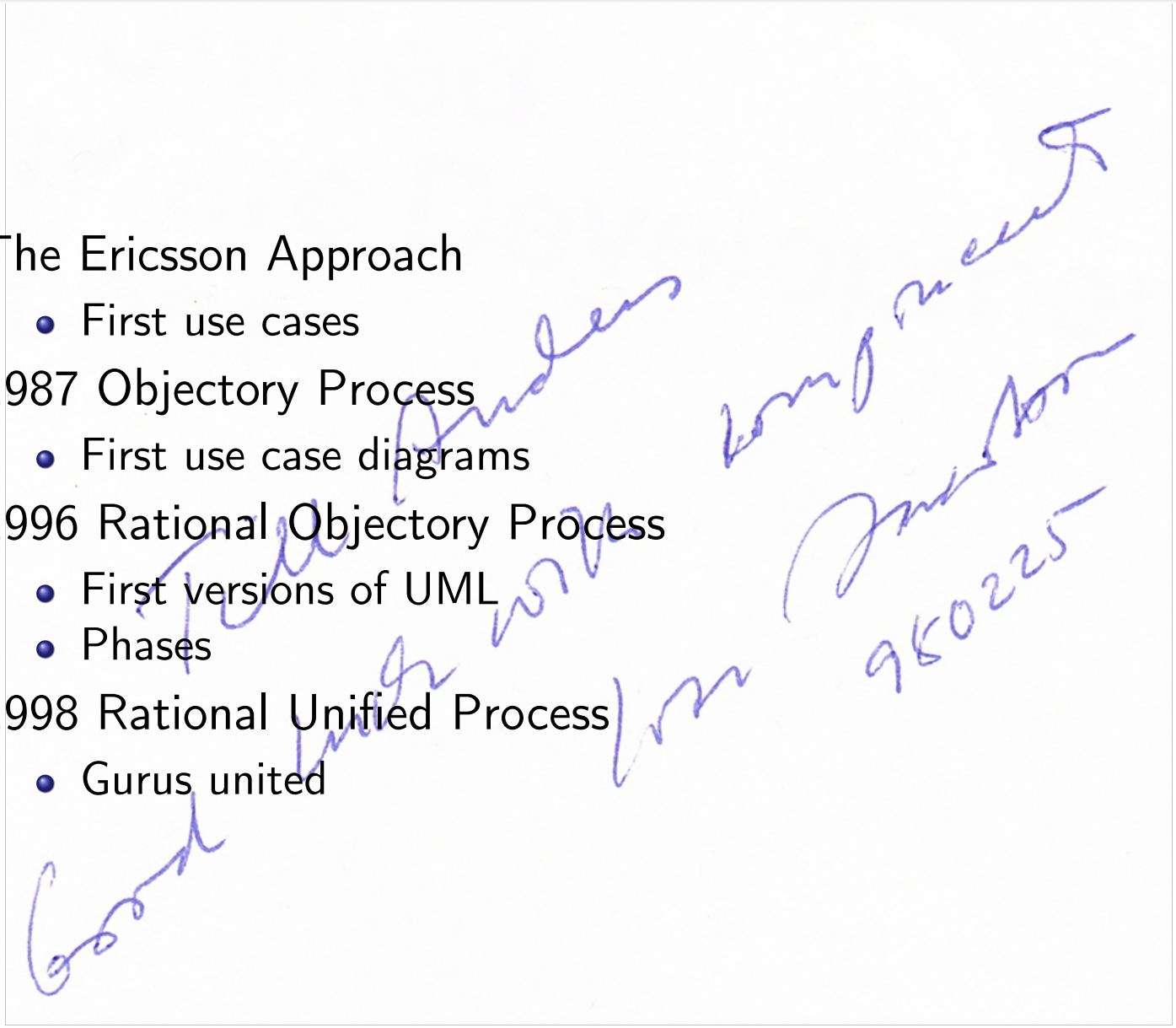
Fall 2018

Agenda

- Unified Process review
- Architecture
- Unified Modeling Language

History of the Unified Process

- The Ericsson Approach
  - First use cases
- 1987 Objectory Process
  - First use case diagrams
- 1996 Rational Objectory Process
  - First versions of UML
  - Phases
- 1998 Rational Unified Process
  - Gurus united

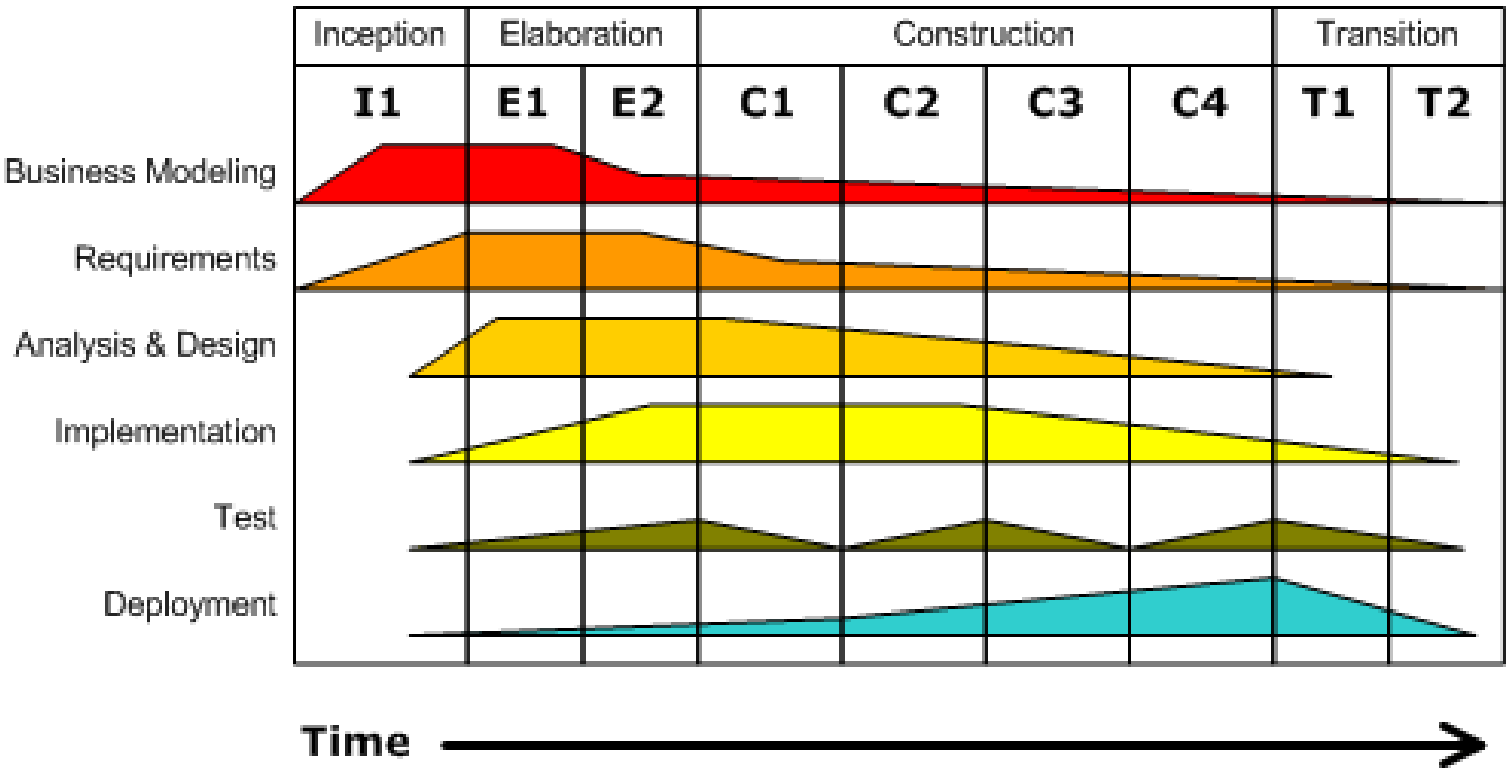


The Unified Process is

- **Use-Case Driven**
- Architecture-Centric
- Iterative and Incremental

Iterative Development

Business value is delivered incrementally in time-boxed cross-discipline iterations.



- Business modeling
- Requirements Capture: From Vision to Requirements
- Capturing the Requirements as Use Cases
- Analysis
- Design
- Implementation
- Test

The Phases

- Inception
  Launches the Project
- Elaboration
  Makes the Architectural Baseline
- Construction
  Leads to Initial Operational Capability
- Transistion
  Completes Product Release

The goal in the inception phase is to make the business case to the extend necessary to justify lauching the project.

- Cost/benefit
  Will the gains accruing from the use or sale of the software product more than offset the cost of developing it?
- Time to market
  Will it reach the market (or internal application) in time to obtain these gains?
- Resolve system scope
- Resolve ambiguities in the requirements needed in this phase
- Establish a candidate architecture
- Mitigate the critical risks

- A feature list.
- A first version of a business (or domain) model that describes the context of the system.
- A first cut of the models representing a first version of the use-case model, the analysis model, the design model. Of the implementation model and test model, there may be something rudimentary. There is also a first version of the supplementary requirements.
- A first draft of a candidate architecture description with outlines of views of the use case, analysis, design, and implementation models.
- Possibly a proof-of-concept exploratory prototype, demonstrating the use of the new system.
- An initial risk list and a use-case ranking list.
- The beginnings of a plan for the entire project, including a general plan for the phases.
- A first draft of the business case, which includes: business context and success criteria (revenue projection, market recognition, project estimate).

The goal in the elaboration phase is to capture most of the remaining requirements, formulating the functional requirements as use cases.
Also a sound architectural foundation - the architectural baseline - must be established.

- Monitor remaining risks
- Fill in further details of the project plan.
- Judge the Worth of the Business Case

- Preferably a complete business (or domain) model which describes the context of the system.
- A new version of all models: use cases, analysis, design, deployment, and implementation. (At the end of the elaboration phase these models will be complete to less than 10% apart from the use case and analysis model that may include more (in some cases up to 80%) use cases to ascertain that the requirements have been understood. The majority of all use cases have been understood to make sure that no architecturally important use cases are left aside and that we can estimate the costs of introducing them.)
- An executable architectural baseline.
- An architecture description, including views of the use case, analysis, design, deployment, and implementation models.
- Updated risk list.
- Project plan for the construction and transition phases.
- A preliminary user manual (optional).
- Completed business case, including business bid.

The team working in the construction phase, starting from an executable architecture baseline and working through a series of iterations and increments, develops a software product ready for initial operation in the user environment.

- Project plan for the transition phase.
- The executable software itself—the initial-operational-capability release. This is the final build from construction.
- All artifacts, including models of the system.
- Maintained and minimally updated architecture description.
- Preliminary user manual in enough detail to guide beta users.
- Business case, reflecting situation at end of phase.

This phase focuses on establishing the product in the operational environment.

- Find out whether the systme really does what the business and its users request.
- Discover unanticipated risks.
- Note unresolved problems.
- Find failures.
- Fix ambiguities and gaps in the user documentation.
- Focus on areas where users appear to be deficient and in need of information or training.

- The executable software itself, including installation software.
- Legal documents such as contracts, license documents, waivers, and warranties.
- Completed and corrected product release baseline including all models of the system.
- Completed and updated architecture description.
- Final user, operator, and system administrator manuals and training materials.
- Customer support references and web references on where to find more information, to report defects, and to find information on fixes and upgrades.

Consider the following classification of prototypes by Christiane Floyd!

3. Approaches to Prototyping

Depending on the goals we wish to achieve, we can distinguish three broad classes of prototyping:

- prototyping for exploration, where the emphasis is on clarifying requirements and desirable features of the target system and where alternative possibilities for solutions are discussed,
- prototyping for experimentation, where the emphasis is on determining the adequacy of a proposed solution before investing in large-scale implementation of the target system,
- prototyping for evolution, where the emphasis is on adapting the system gradually to changing requirements, which cannot reliably be determined in one early phase.

Can you in any way relate them to the four UP phases?

A user story describes functionality that will be valuable to either a user or purchaser of a system or software.

- User stories are written by users.
- User stories focuses on details.
- User stories cannot be guaranteed to be complete.
- User stories does not define system boundaries.

A use case is a generalised description of a set of interactions between the system and one or more actors, where an actor is either a user or another system.

- Use cases are written by developers.
- A use case model will describe the complete (sub)system.
- A use case defines the system boundary.
- Any use case scenario will bring the system from one consistent state to another.

Alistair Cockburn has proposed a template for use cases, which can be found at:

`http://alistair.cockburn.us/Basic+use+case+template`

The template is used in the example later in this presentation.

Use cases can be at different detail levels:

- Brief use case
  The use case is merely more than the title of an action
- Casual use case
  Typically casual use cases are described in more detail
- Fully dressed use case
  Includes all entries from the template

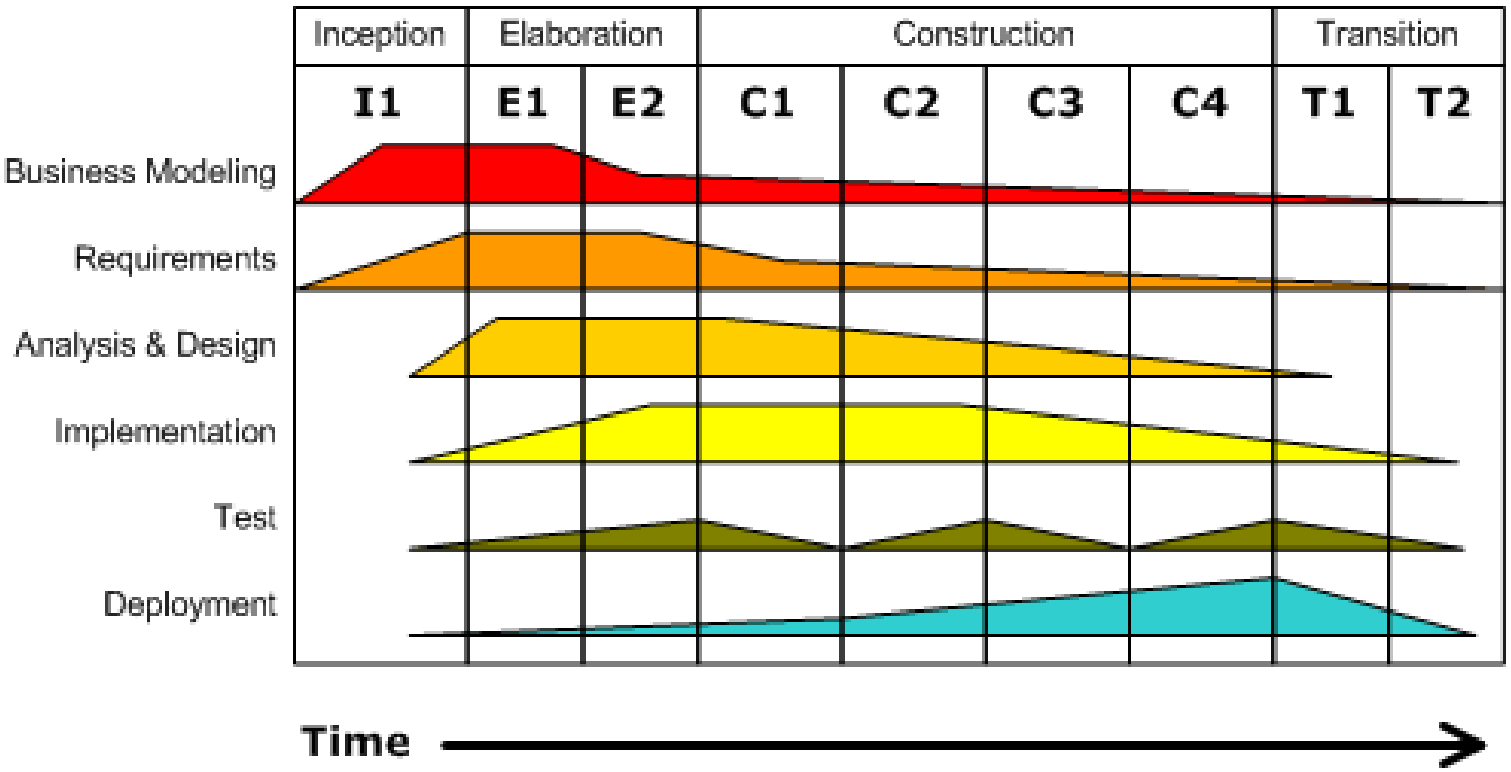
Actors are outside the system. For this reason, it is important to describe their responsibilities in detail. Actor resposibilities are not and should not be covered by the system.
Actors are not concrete people or systems, rather abstract roles of these users or systems.

The Unified Process is

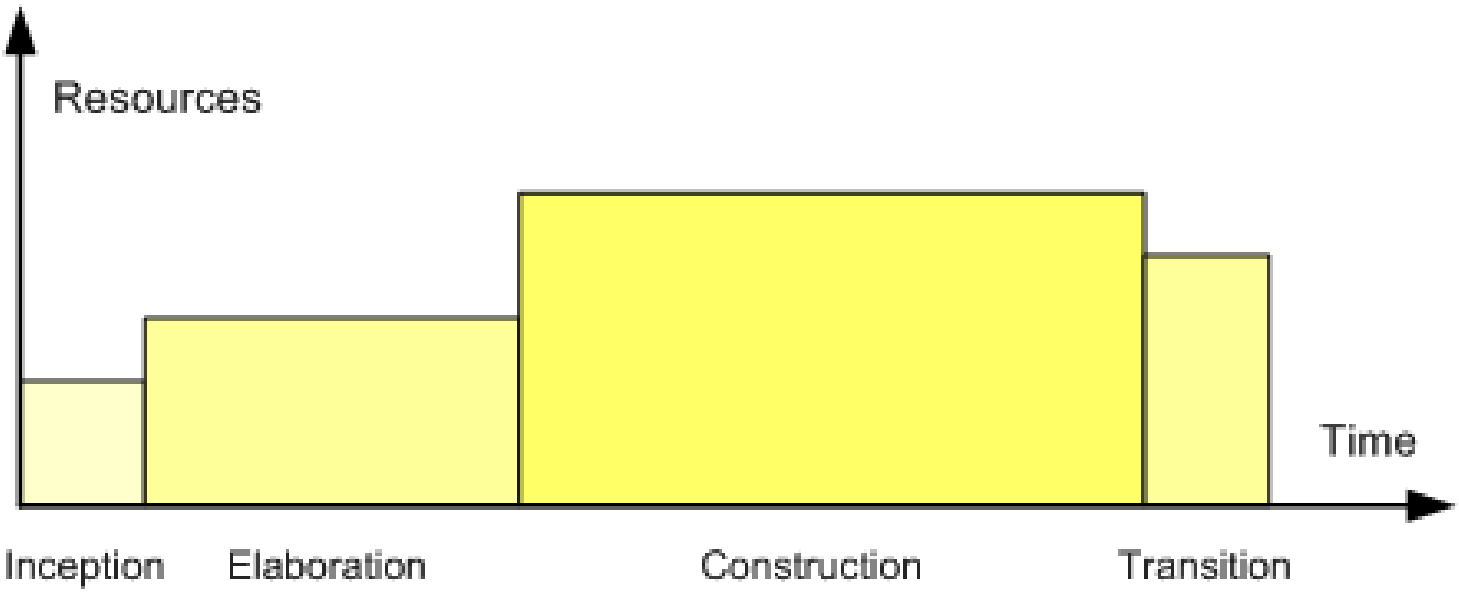
- Use-Case Driven
- **Architecture-Centric**
- Iterative and Incremental

Iterative Development

Business value is delivered incrementally in time-boxed cross-discipline iterations.



Phases and resources



The goal in the inception phase is to make the business case to the extend necessary to justify lauching the project.

- Cost/benefit
  Will the gains accruing from the use or sale of the software product more than offset the cost of developing it?
- Time to market
  Will it reach the market (or internal application) in time to obtain these gains?
- Resolve system scope
- Resolve ambiguities in the requirements needed in this phase
- **Establish a candidate architecture**
- Mitigate the critical risks

- A feature list.
- A first version of a business (or domain) model that describes the context of the system.
- A first cut of the models representing a first version of the use-case model, the analysis model, the design model. Of the implementation model and test model, there may be something rudimentary. There is also a first version of the supplementary requirements.
- A first draft of a candidate architecture description with outlines of views of the use case, analysis, design, and implementation models.
- Possibly a proof-of-concept exploratory prototype, demonstrating the use of the new system.
- An initial risk list and a use-case ranking list.
- The beginnings of a plan for the entire project, including a general plan for the phases.
- A first draft of the business case, which includes: business context and success criteria (revenue projection, market recognition, project estimate).

The goal in the elaboration phase is to capture most of the remaining requirements, formulating the functional requirements as use cases. Also a sound architectural foundation - the **architectural baseline** - must be established.

- Monitor remaining risks
- Fill in further details of the project plan.
- Judge the Worth of the Business Case

- Preferably a complete business (or domain) model which describes the context of the system.
- A new version of all models: use cases, analysis, design, deployment, and implementation. (At the end of the elaboration phase these models will be complete to less than 10% apart from the use case and analysis model that may include more (in some cases up to 80%) use cases to ascertain that the requirements have been understood. The majority of all use cases have been understood to make sure that no architecturally important use cases are left aside and that we can estimate the costs of introducing them.)
- An executable architectural baseline.
- An architecture description, including views of the use case, analysis, design, deployment, and implementation models.
- Updated risk list.
- Project plan for the construction and transition phases.
- A preliminary user manual (optional).
- Completed business case, including business bid.

The team working in the construction phase, starting from an executable **architecture baseline** and working through a series of iterations and increments, develops a software product ready for initial operation in the user environment.

- Project plan for the transition phase.
- The executable software itself—the initial-operational-capability release. This is the final build from construction.
- All artifacts, including models of the system.
- Maintained and minimally updated architecture description.
- Preliminary user manual in enough detail to guide beta users.
- Business case, reflecting situation at end of phase.

This phase focuses on establishing the product in the operational environment.

- Find out whether the systme really does what the business and its users request.
- Discover unanticipated risks.
- Note unresolved problems.
- Find failures.
- Fix ambiguities and gaps in the user documentation.
- Focus on areas where users appear to be deficient and in need of information or training.

- The executable software itself, including installation software.
- Legal documents such as contracts, license documents, waivers, and warranties.
- Completed and corrected product release baseline including all models of the system.
- Completed and updated architecture description.
- Final user, operator, and system administrator manuals and training materials.
- Customer support references and web references on where to find more information, to report defects, and to find information on fixes and upgrades.

Problem domain

- Functional requirements
- Use cases (verbs)
- Domain model (nouns)

Solution domain

- Non-functional requirements
- Class diagrams
- Sequence diagrams
- State machine

Architecture centric
Inception?

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Architecture centric
Elaboration?

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Architecture centric
Construction?

cphbusiness
COPENHAGEN BUSINESS ACADEMY

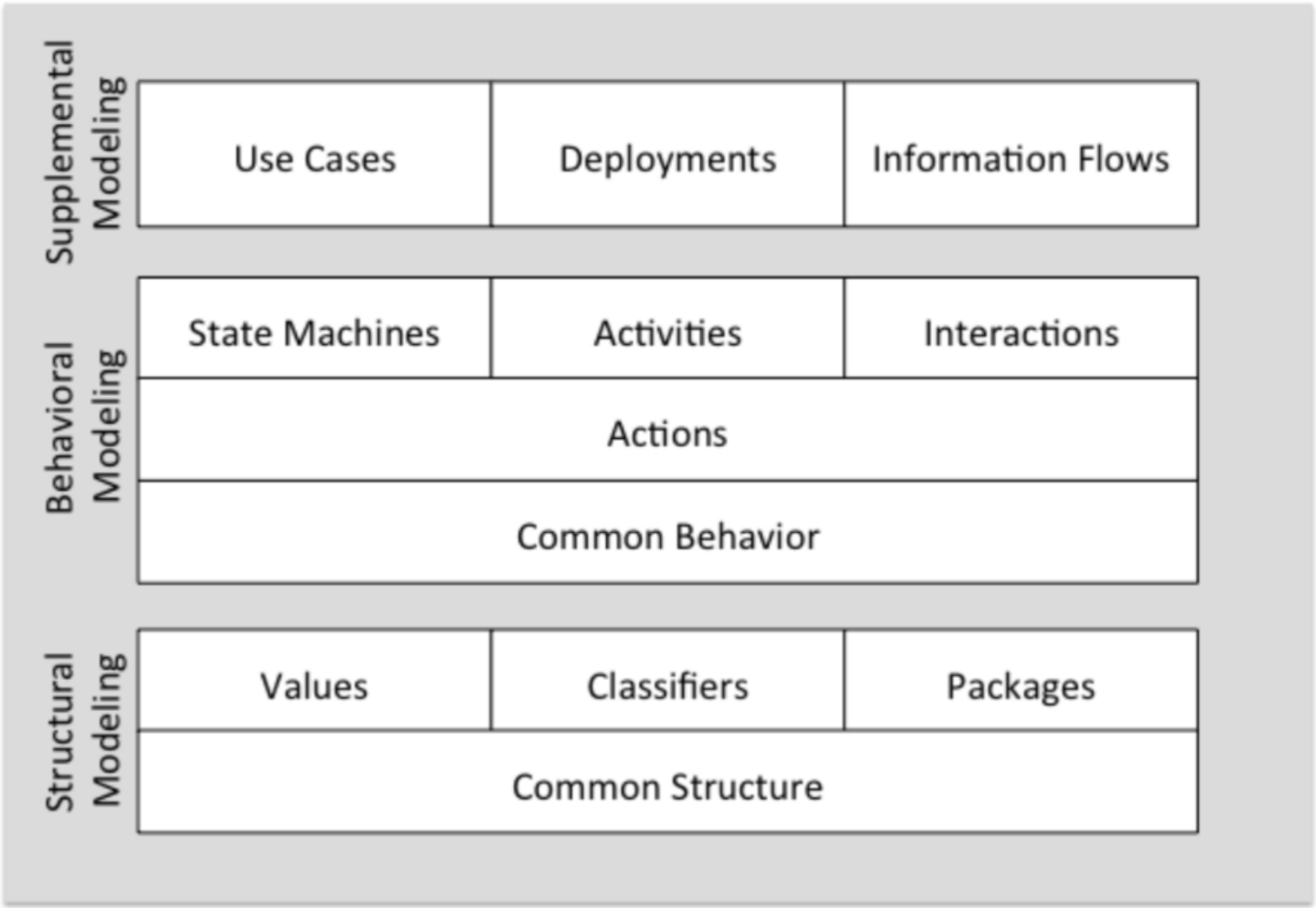
Architectural styles

- Client/server
- Peer-to-Peer
- Repository
- Model/View/Controller
- Three (or more) tier Architecture
- Service-Oriented Architecture (SOA)
- Pipes and filters

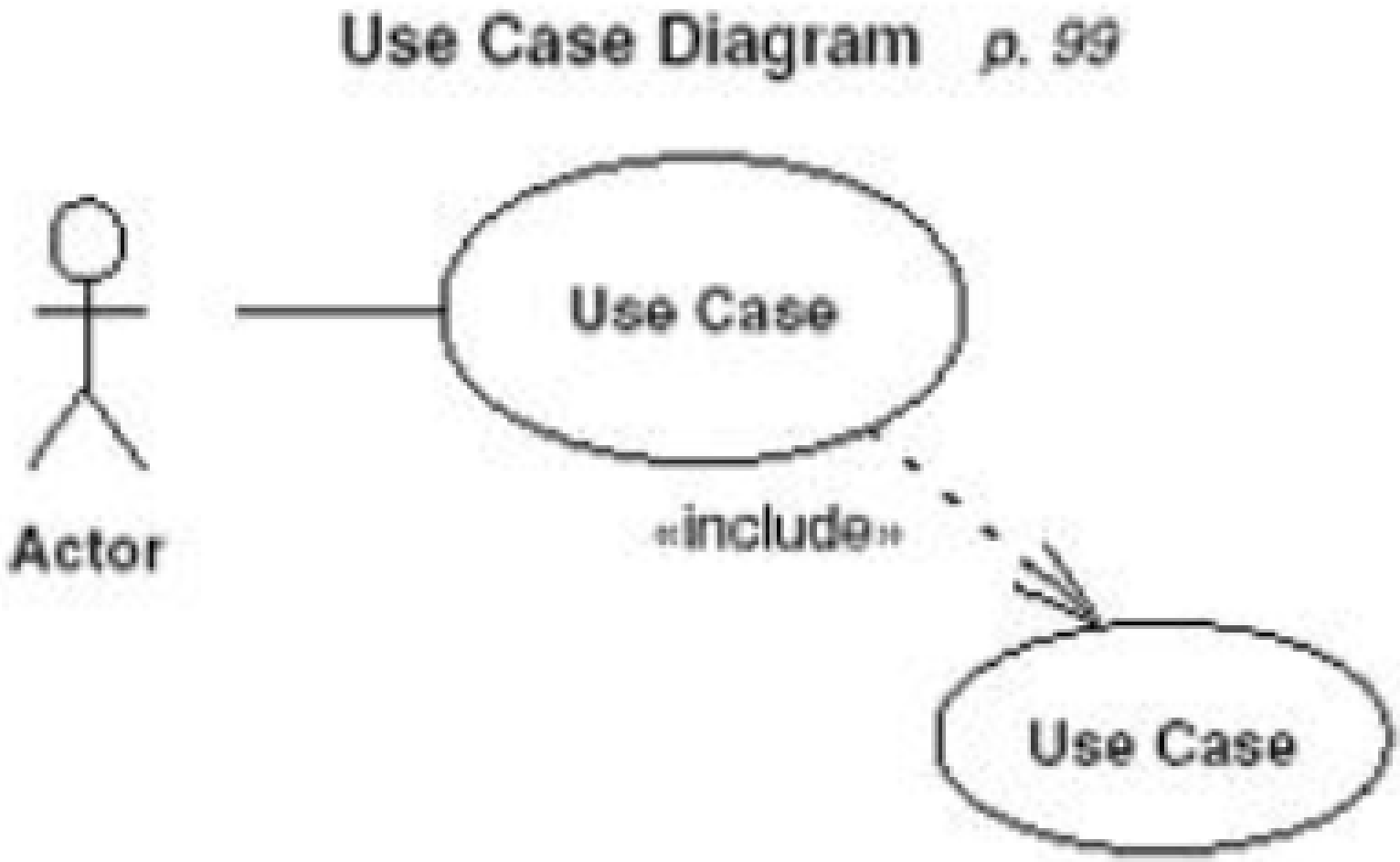
- Logical breakdown into subsystems
  - package diagrams
  - interfaces
- Dynamics of component interaction
  - system sequence diagrams
  - activity diagrams

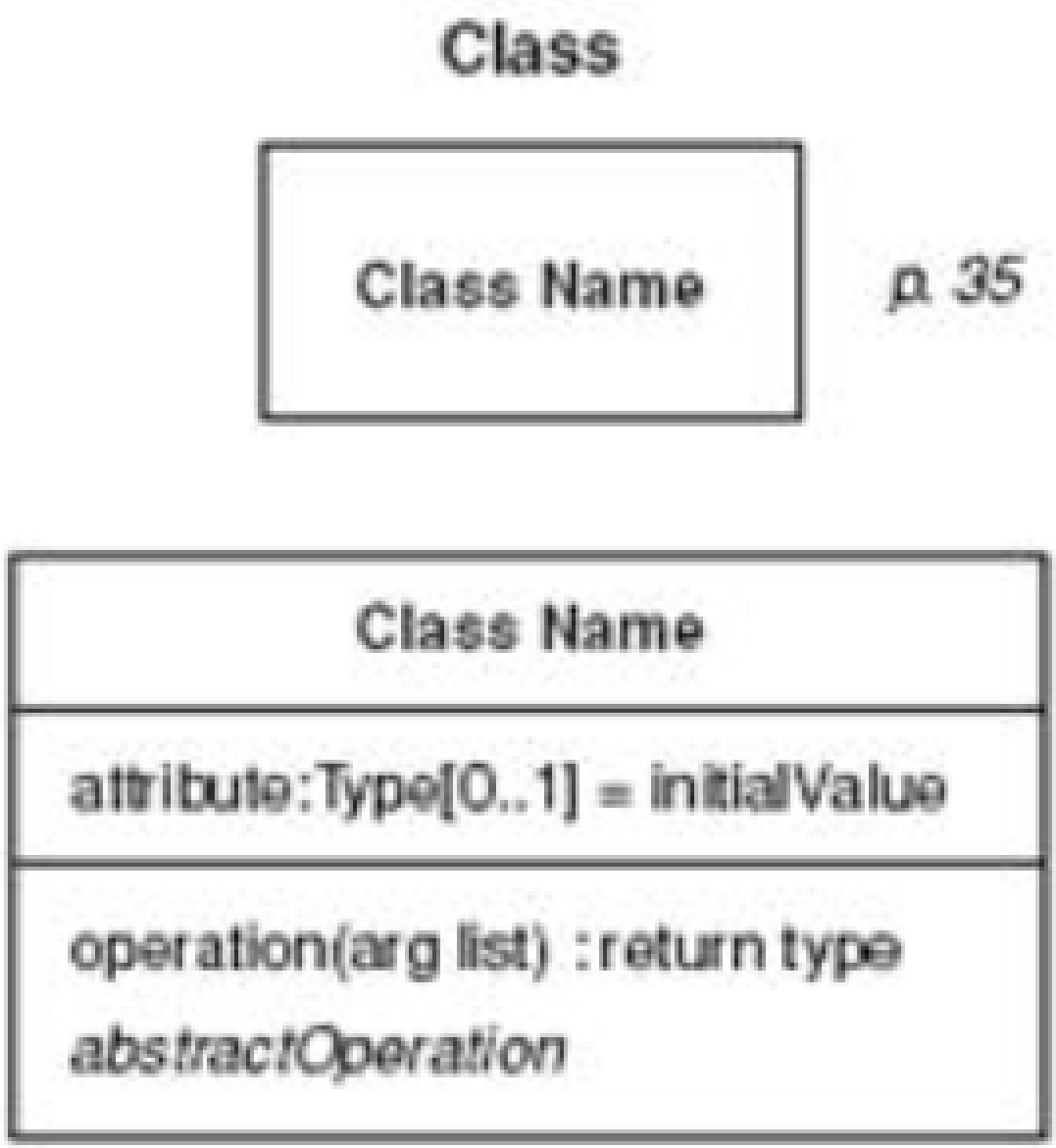
- Data shared among subsystems
  - class diagram
- Run time components
  - component diagrams
  - deployment diagrams

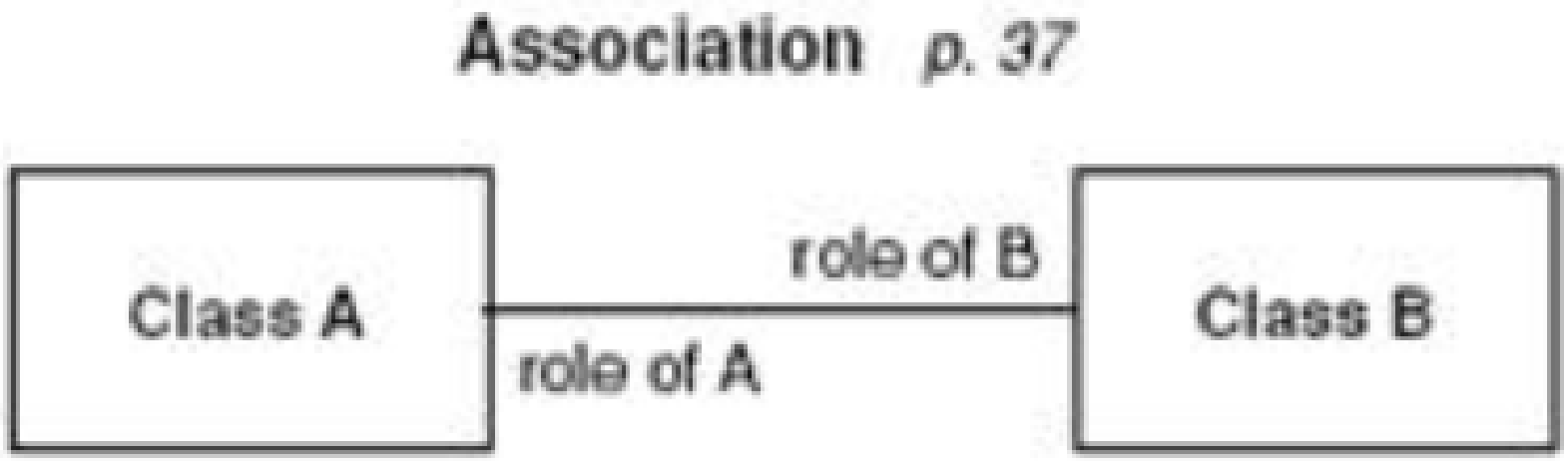

TM


Supplemental Modeling
Use Cases
Deployments
Information Flows
Behavioral Modeling
State Machines
Activities
Interactions
Actions
Common Behavior
Structural Modeling
Values
Classifiers
Packages
Common Structure

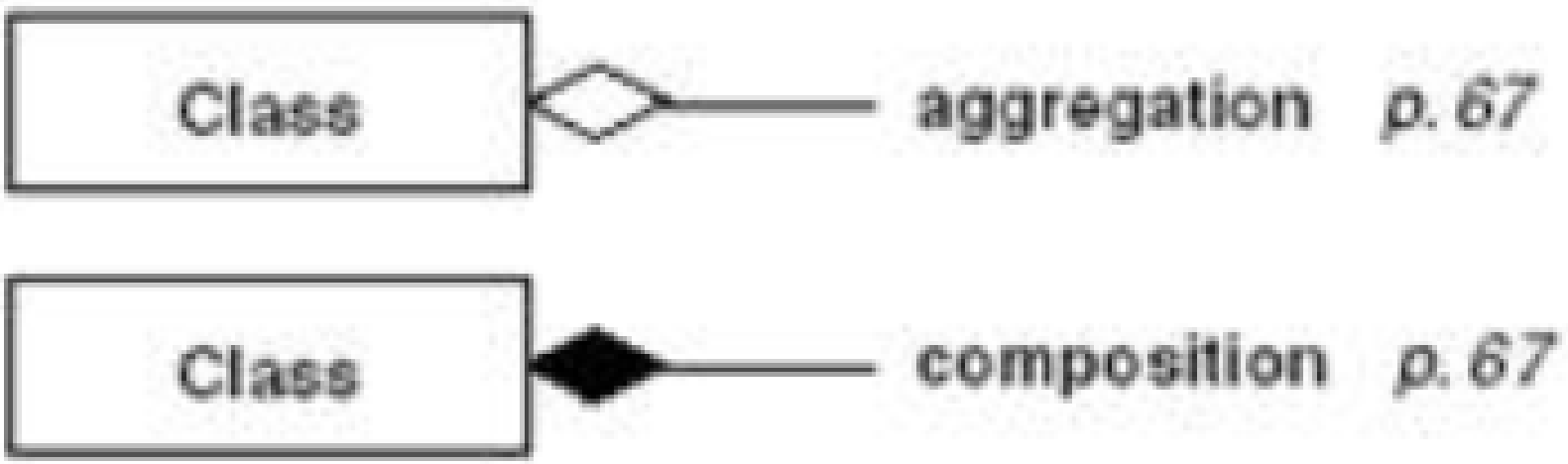
Use case diagram
From UML distilled

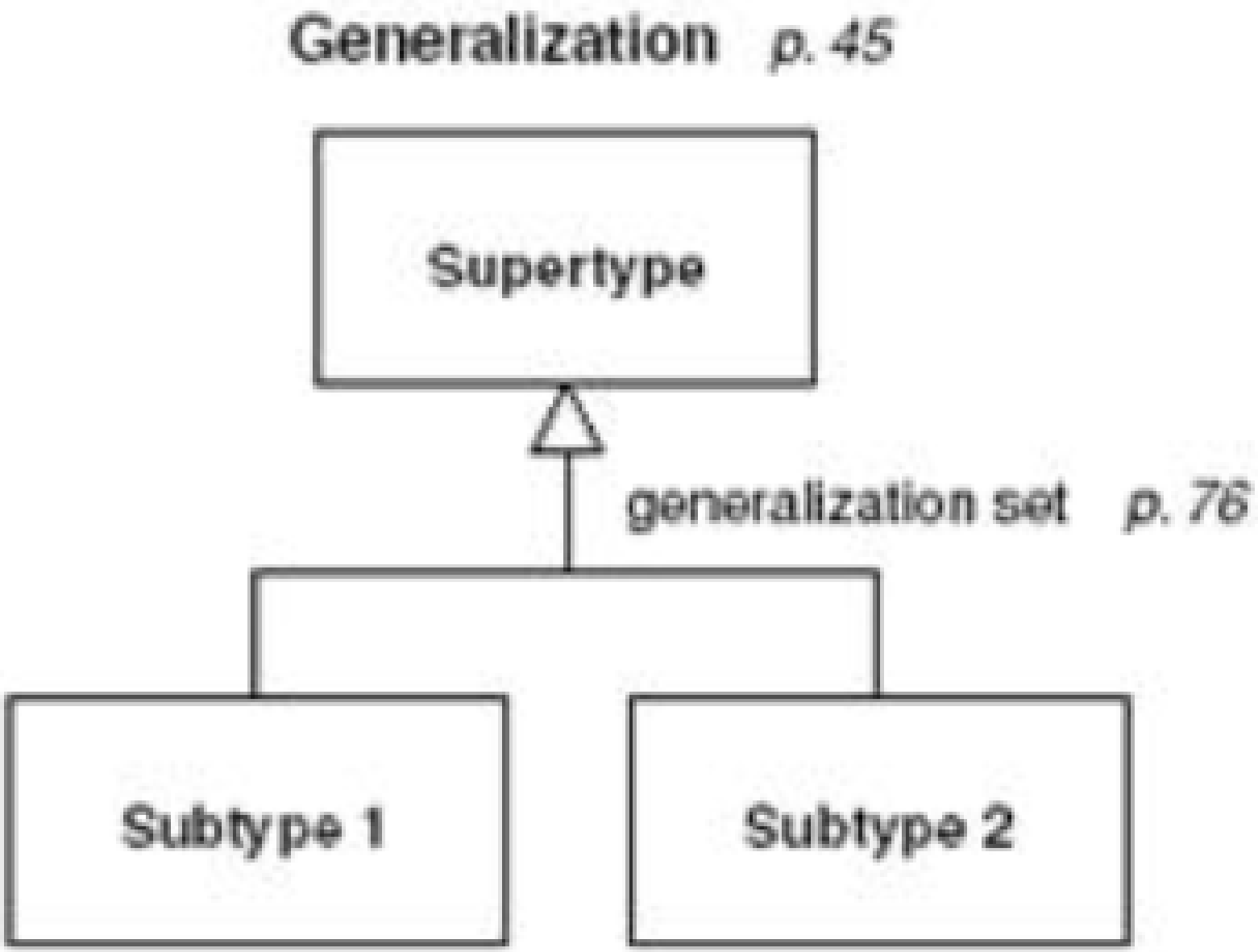


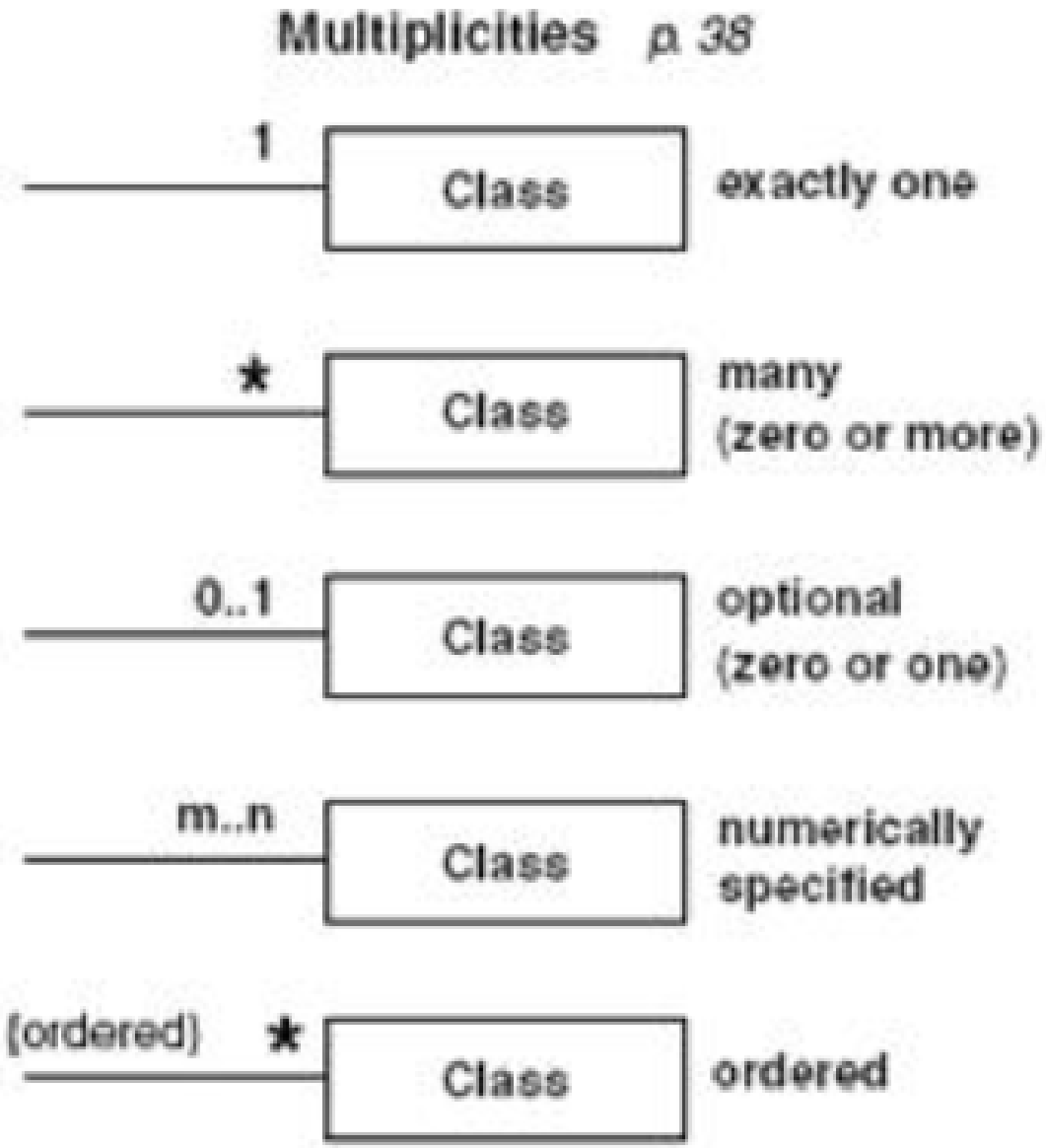

Class
Class Name
p. 35
Class Name
attribute:Type[0..1] = initialValue
operation(arg list) :return type
abstractOperation

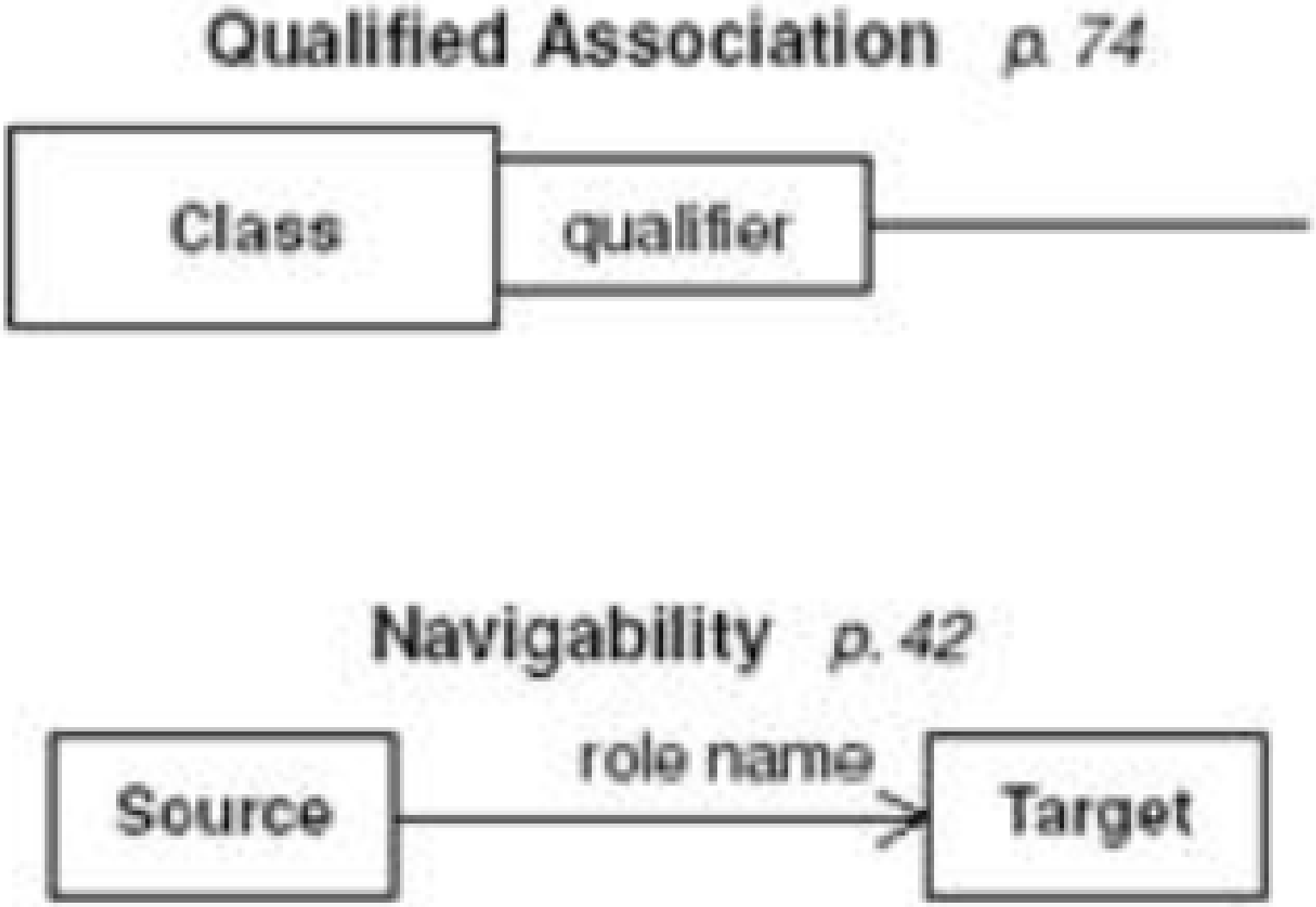

Association p. 37
Class A
role of B
role of A
Class B

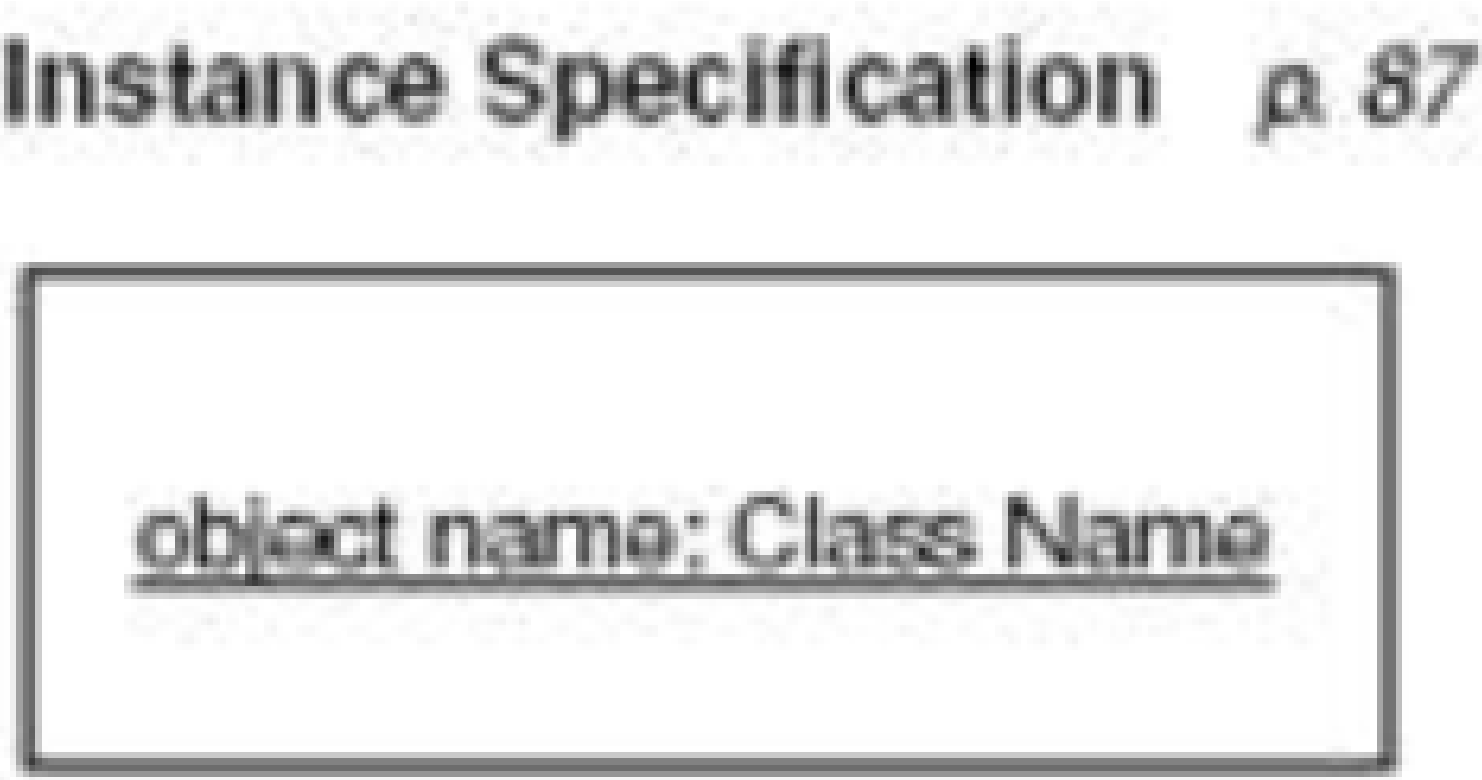
Class diagram 3
From UML distilled



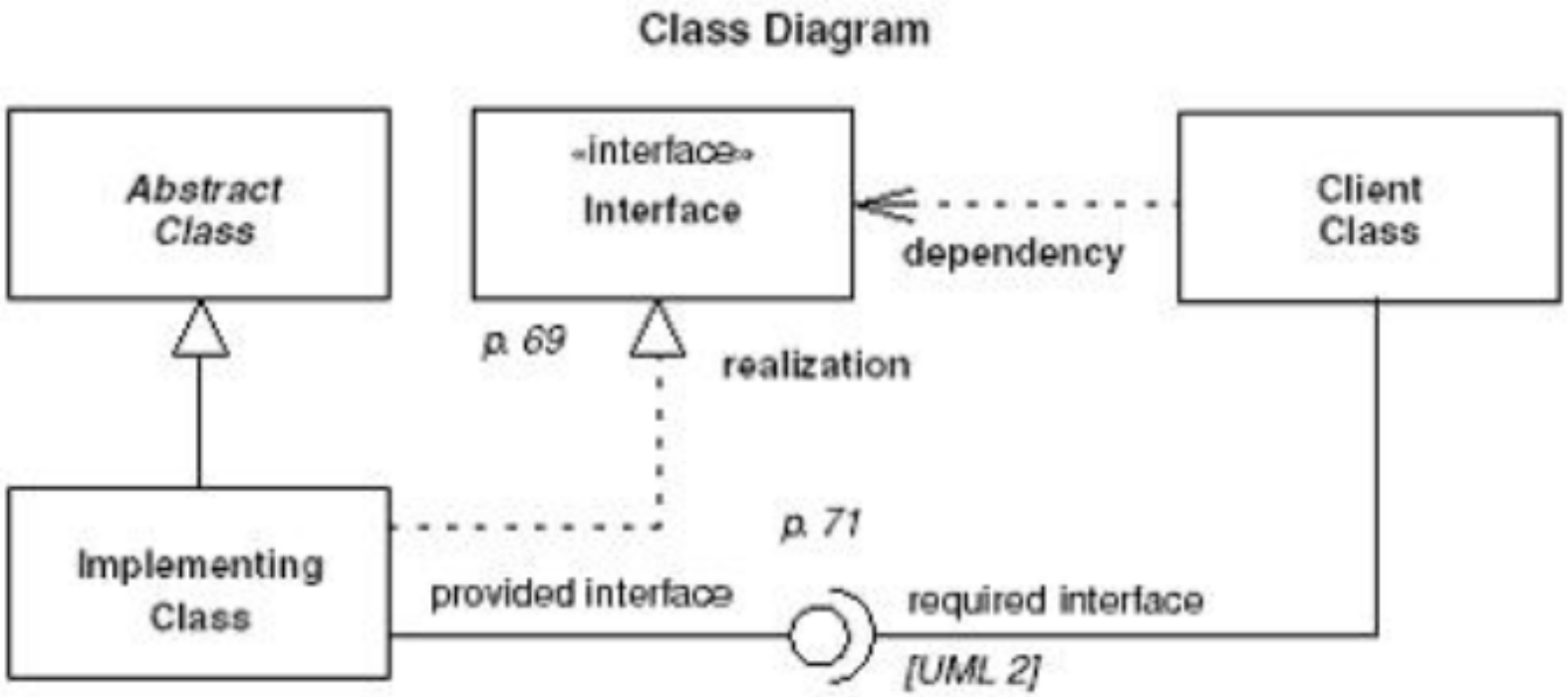

Generalization p. 45
Supertype
generalization set p. 76
Subtype 1
Subtype 2


Multiplicities p. 38
1
Class
exactly one
*
Class
many
(zero or more)
0..1
Class
optional
(zero or one)
m..n
Class
numerically
specified
{ordered}
*
Class
ordered

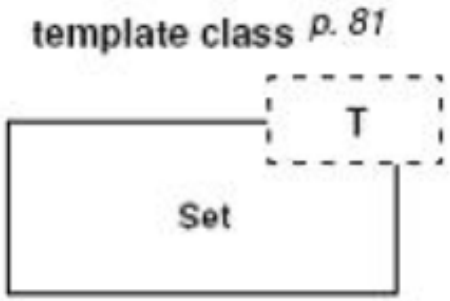

Qualified Association p. 74
Class
qualifier
Navigability p. 42
role name
Source
Target

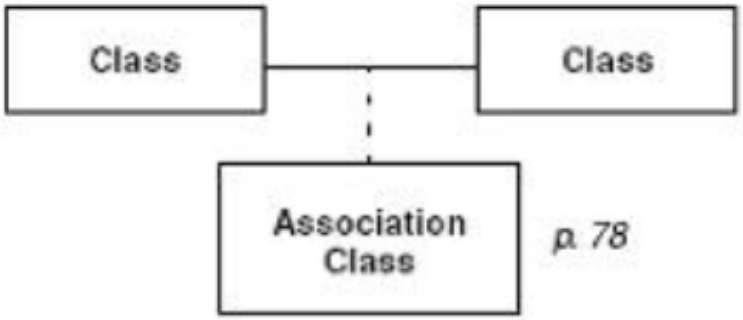

Instance Specification p. 87
object name: Class Name

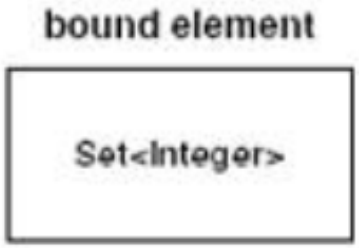
Class diagram 8
From UML distilled

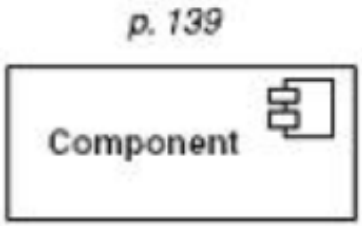


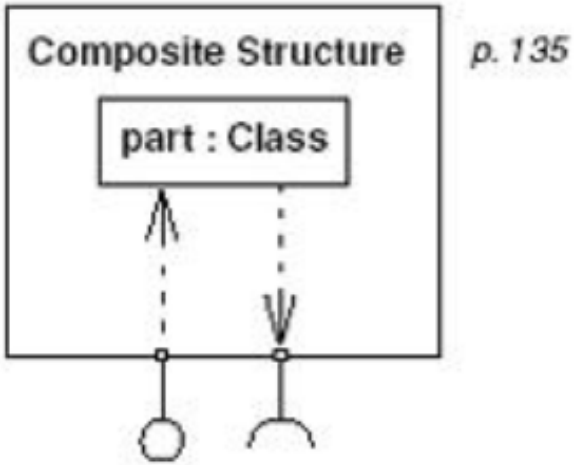
Class diagram 9
From UML distilled



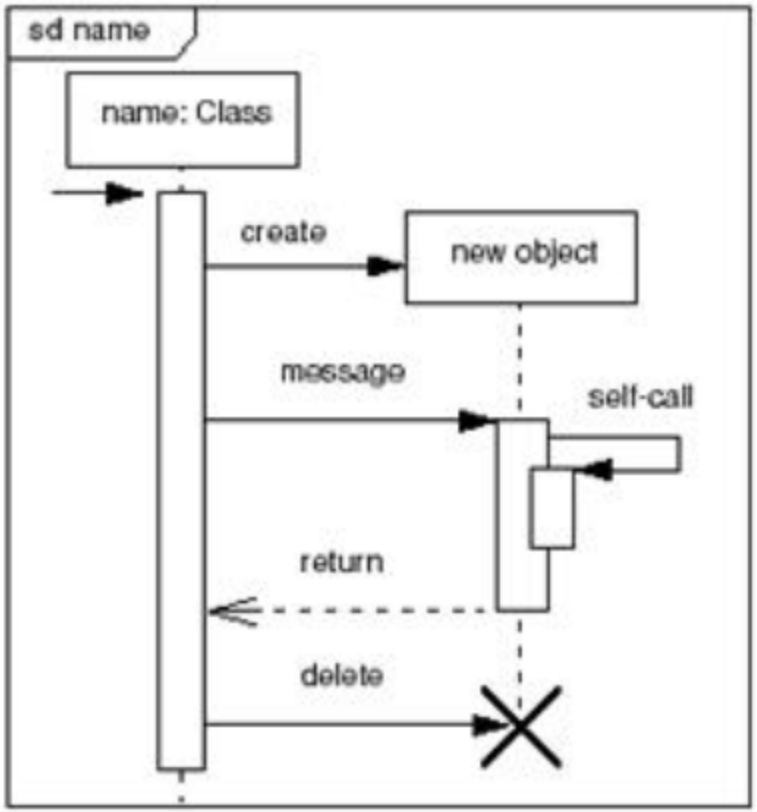


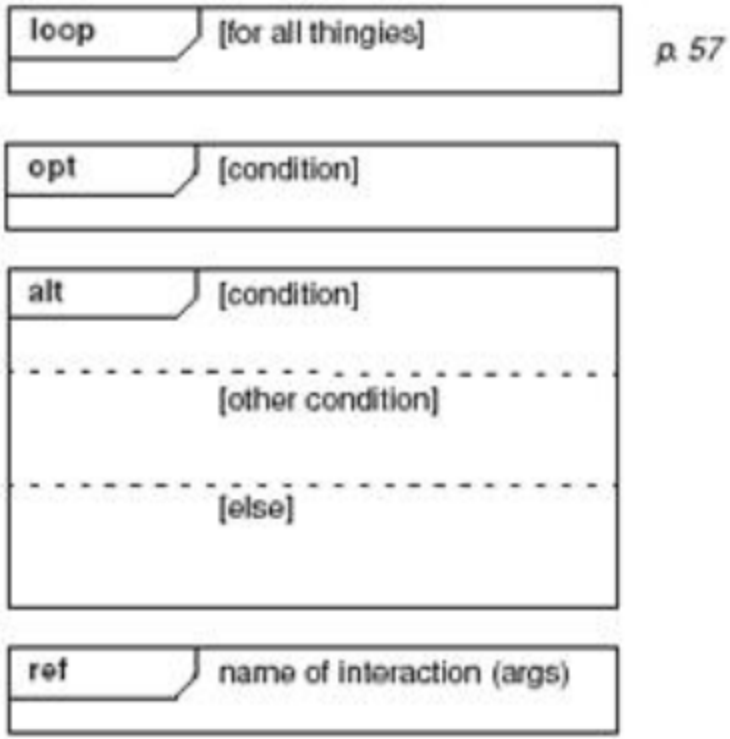


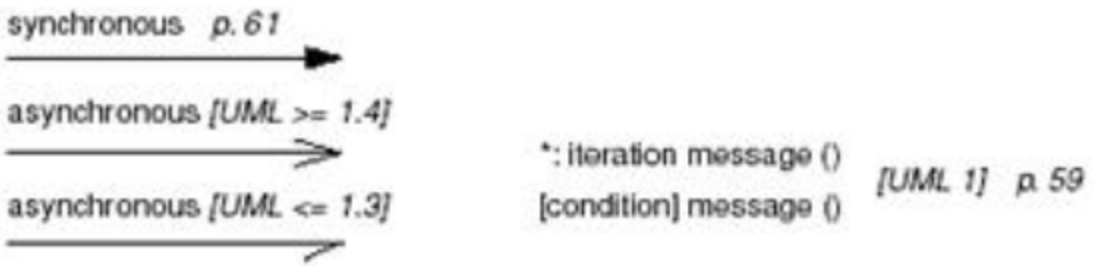




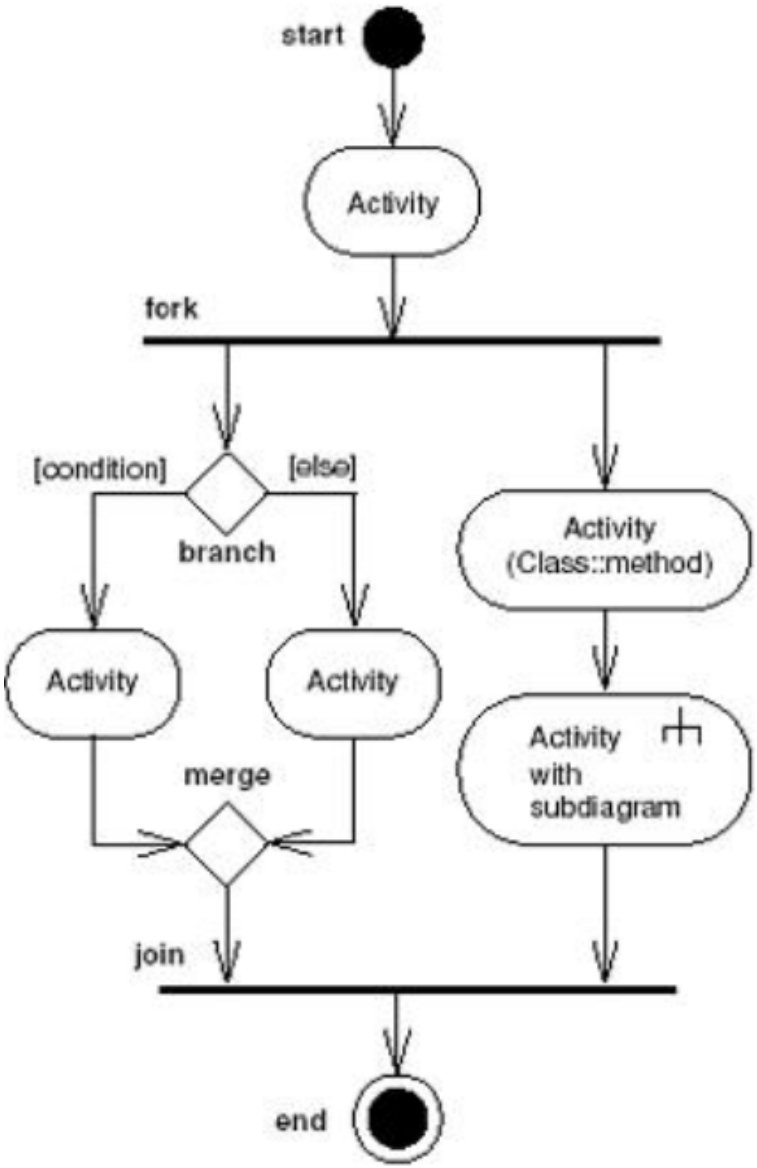
Sequence diagram
From UML distilled

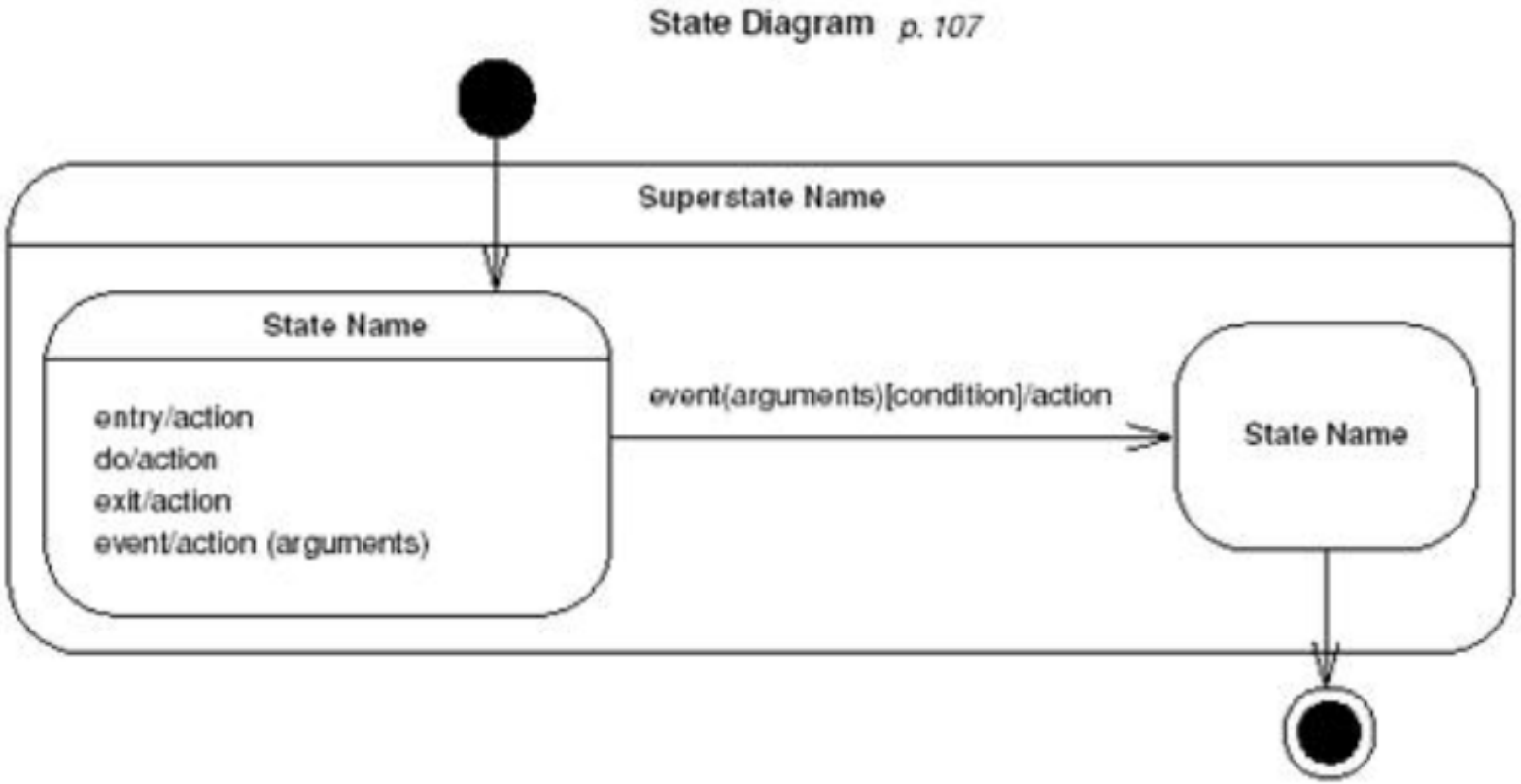


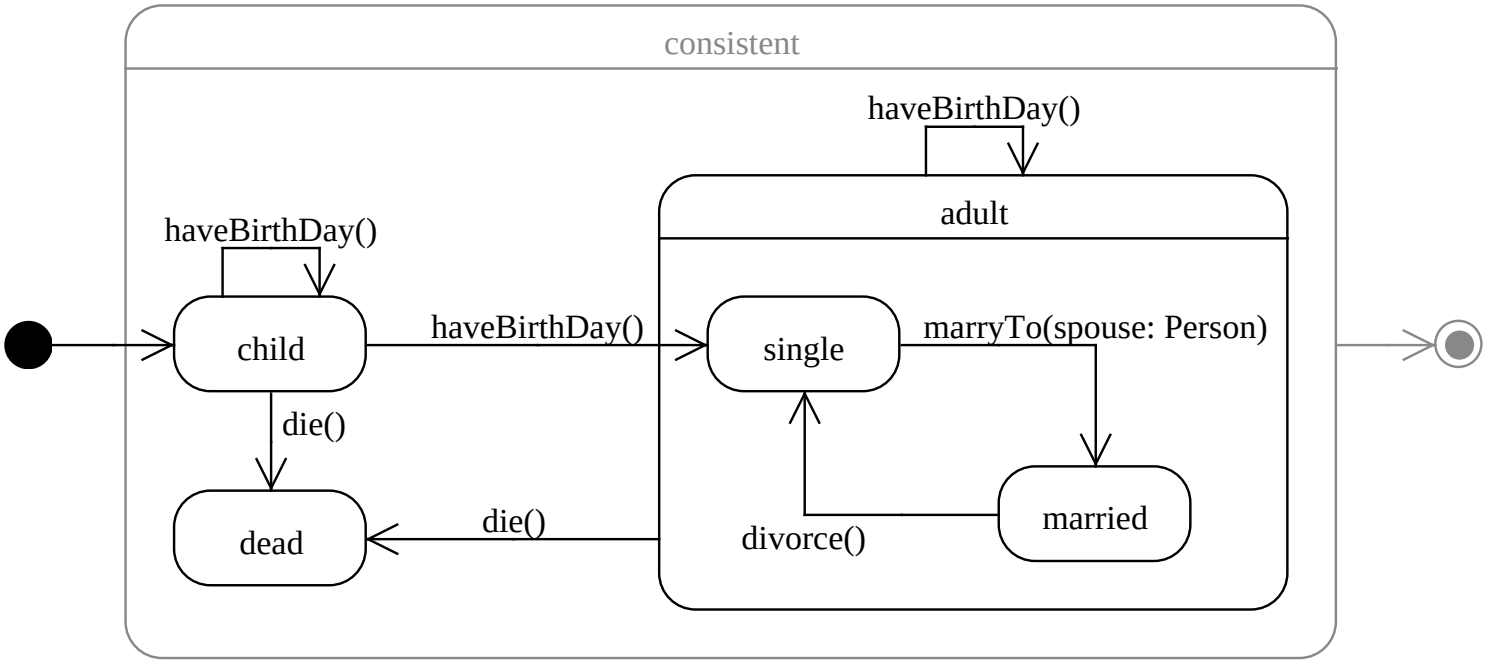




Activity diagram
From UML distilled



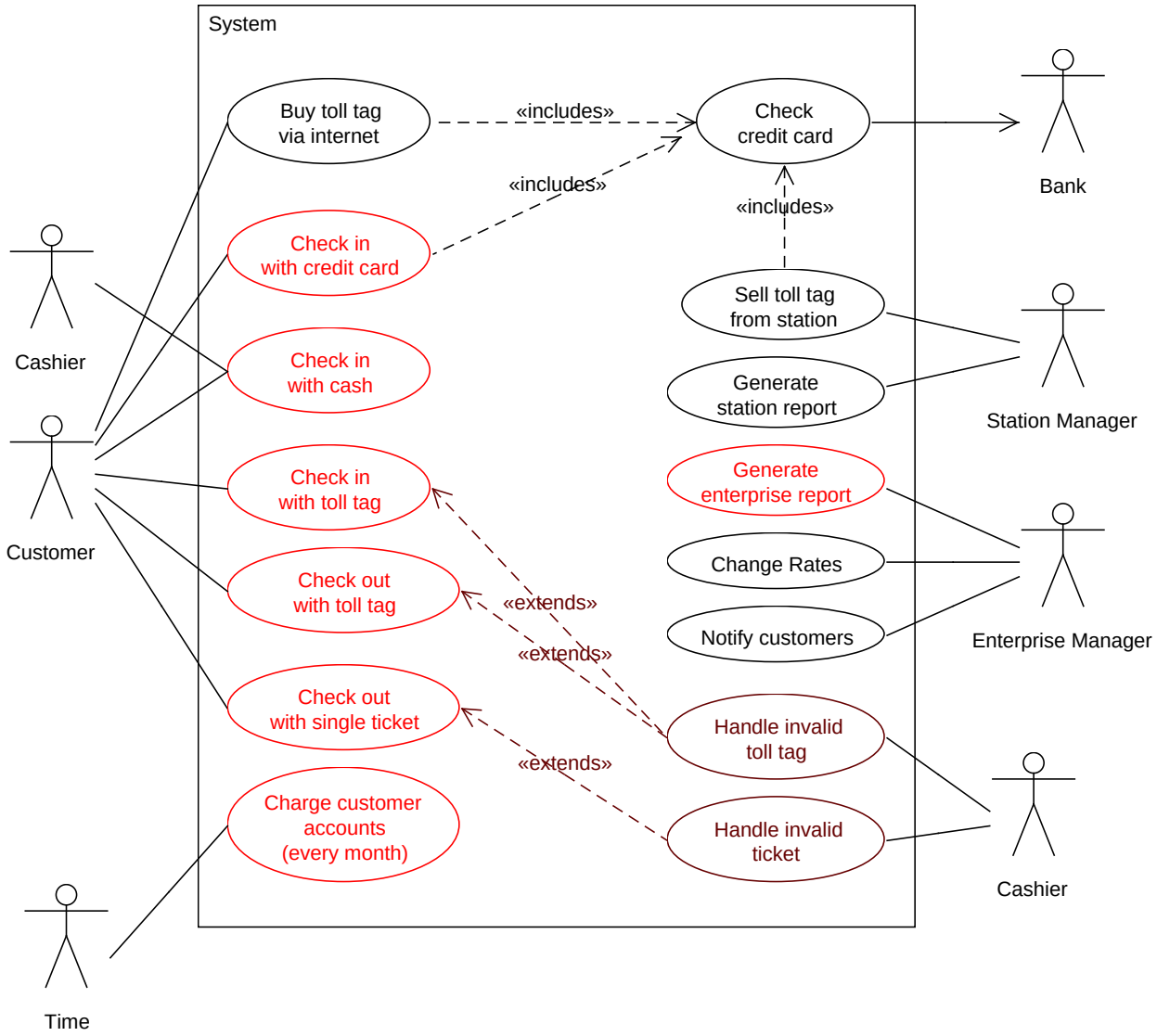

State Diagram p. 107
Superstate Name
State Name
entry/action
do/action
exit/action
event/action (arguments)
event(arguments)[condition]/action
State Name


consistent
haveBirthDay()
adult
haveBirthDay()
child
haveBirthDay()
single
marryTo(spouse: Person)
die()
dead
die()
divorce()
married

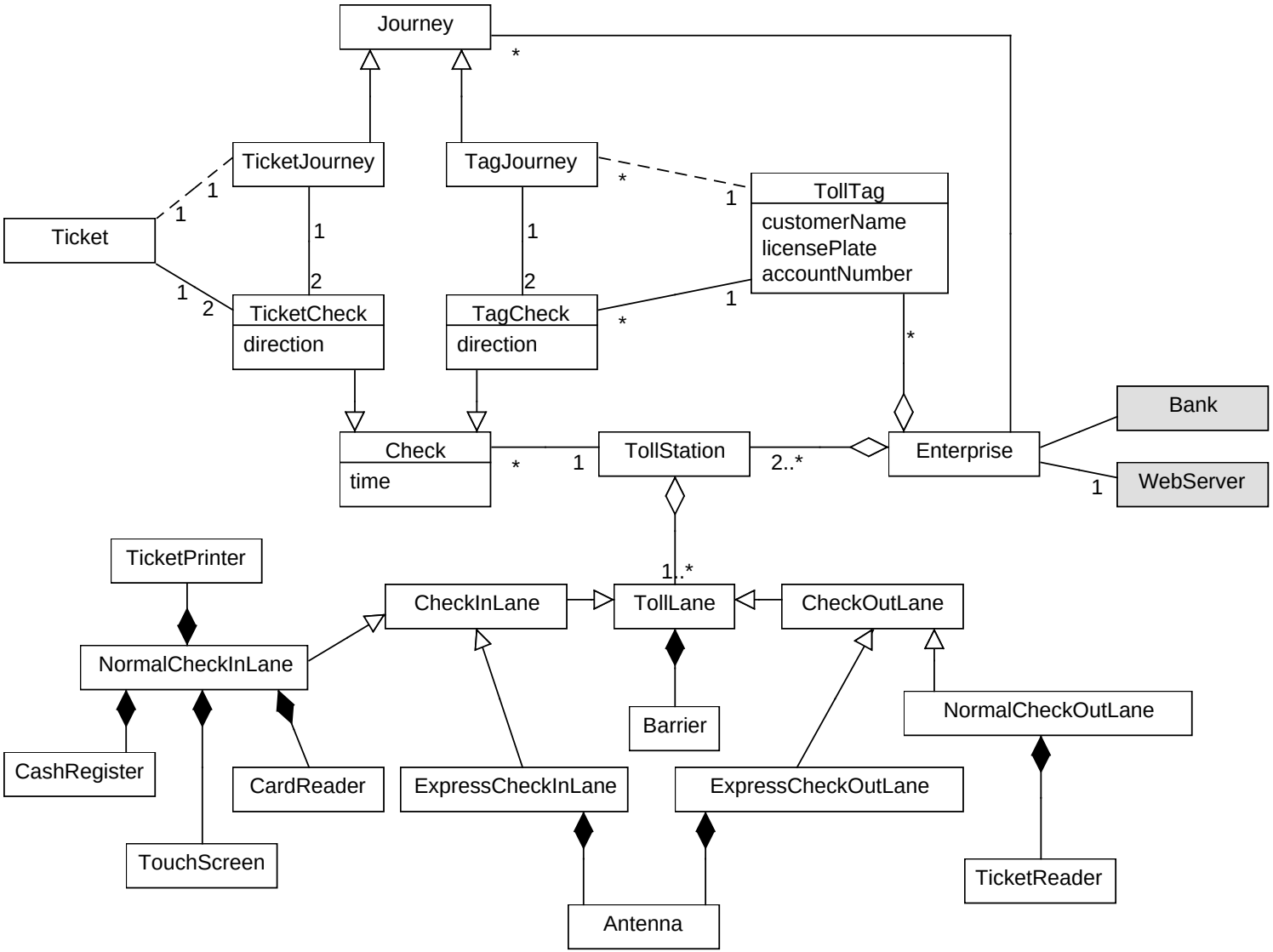
Example
Highway toll system

cphbusiness
COPENHAGEN BUSINESS ACADEMY

Use Case Diagram



Domain model

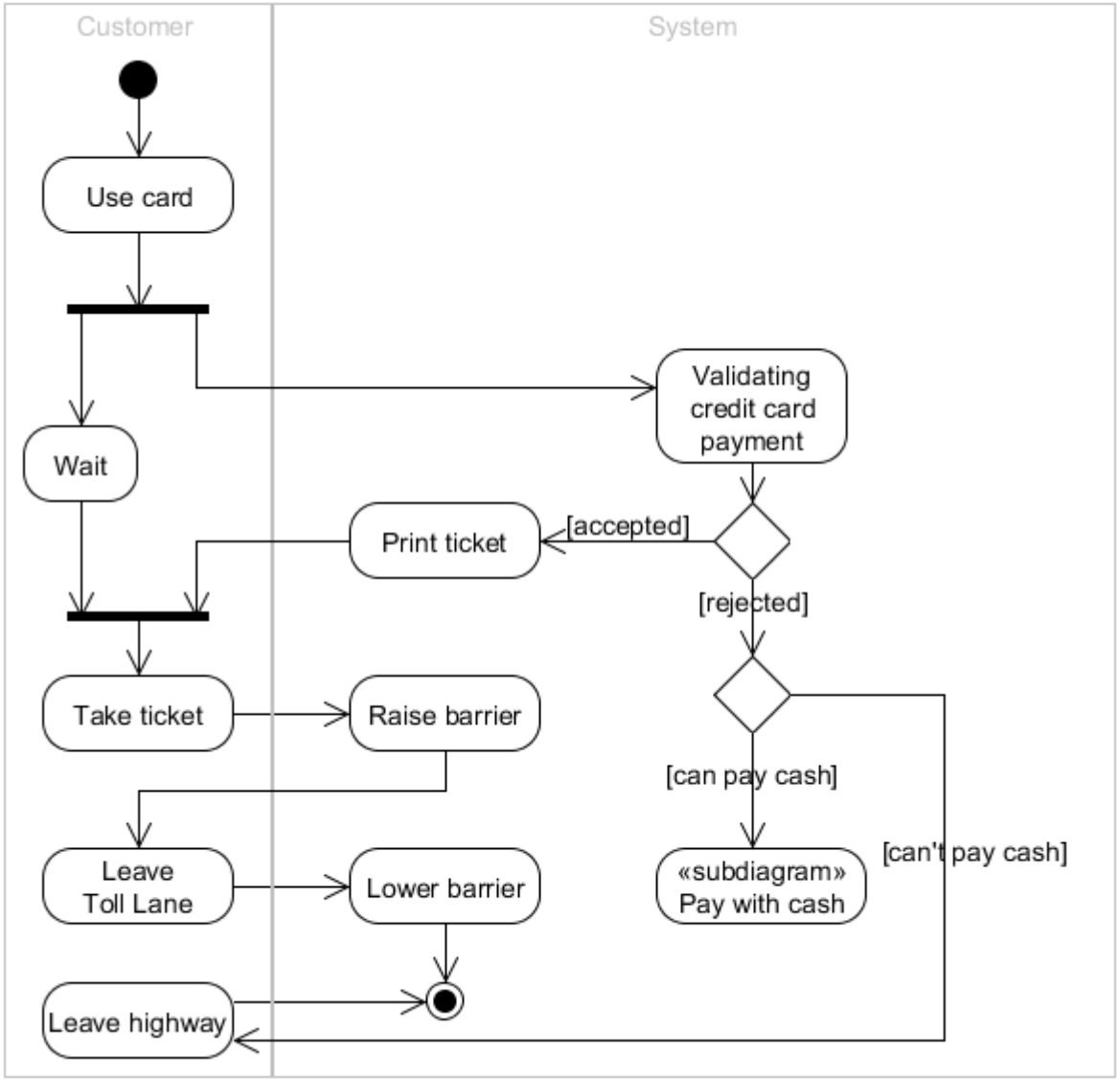


- **Use Case Name** Check in with credit card
- **Summary** The customer wants to enter the highway. To do so, he needs to pay a toll and he wants to pay using a credit card. The customer uses the computer to set the type of vehicle and uses the card reader to pay.
- **Actors** Customer
- **Preconditions**
  - The customer is in possession of a credit card
  - The customer is at a normal check-in lane

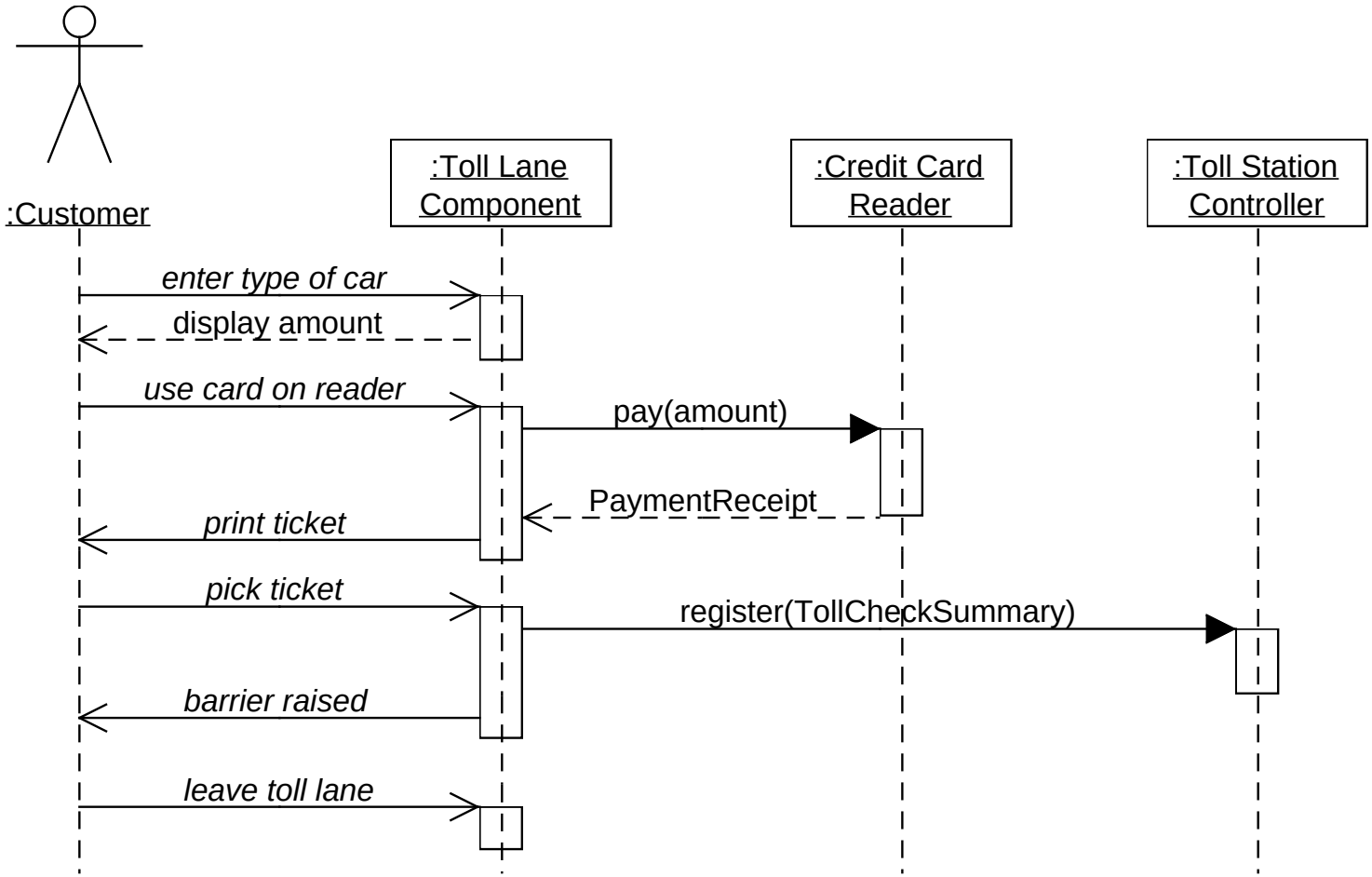
- **Main scenario** Check in with credit card
  1. The customer enters the type of vehicle into the system.
  2. The customer verifies his choice and continues, or reselects (back to step 1).
  3. The amount of money needed to be paid is shown to the customer
  4. The customer uses his card on the card reader
  5. The device validates the card payment
  6. Ticket is printed.
  7. The customer picks the ticket.
  8. The barrier is raised.
  9. The customer leaves the toll lane and the barrier is lowered.

- **Alternative** Check in with credit card
  - 2a. The customer continues with wrong type of vehicle.
    The use case continues as normal
  - 5a. The credit card is not valid for payment
    1. A cashier is summoned
    2. The cashier decides how to check in the customer (e.g. by receiving cash or not letting the customer on the motorway)
- **Post conditions**
  - Information about the check-in has been registered at the toll station.

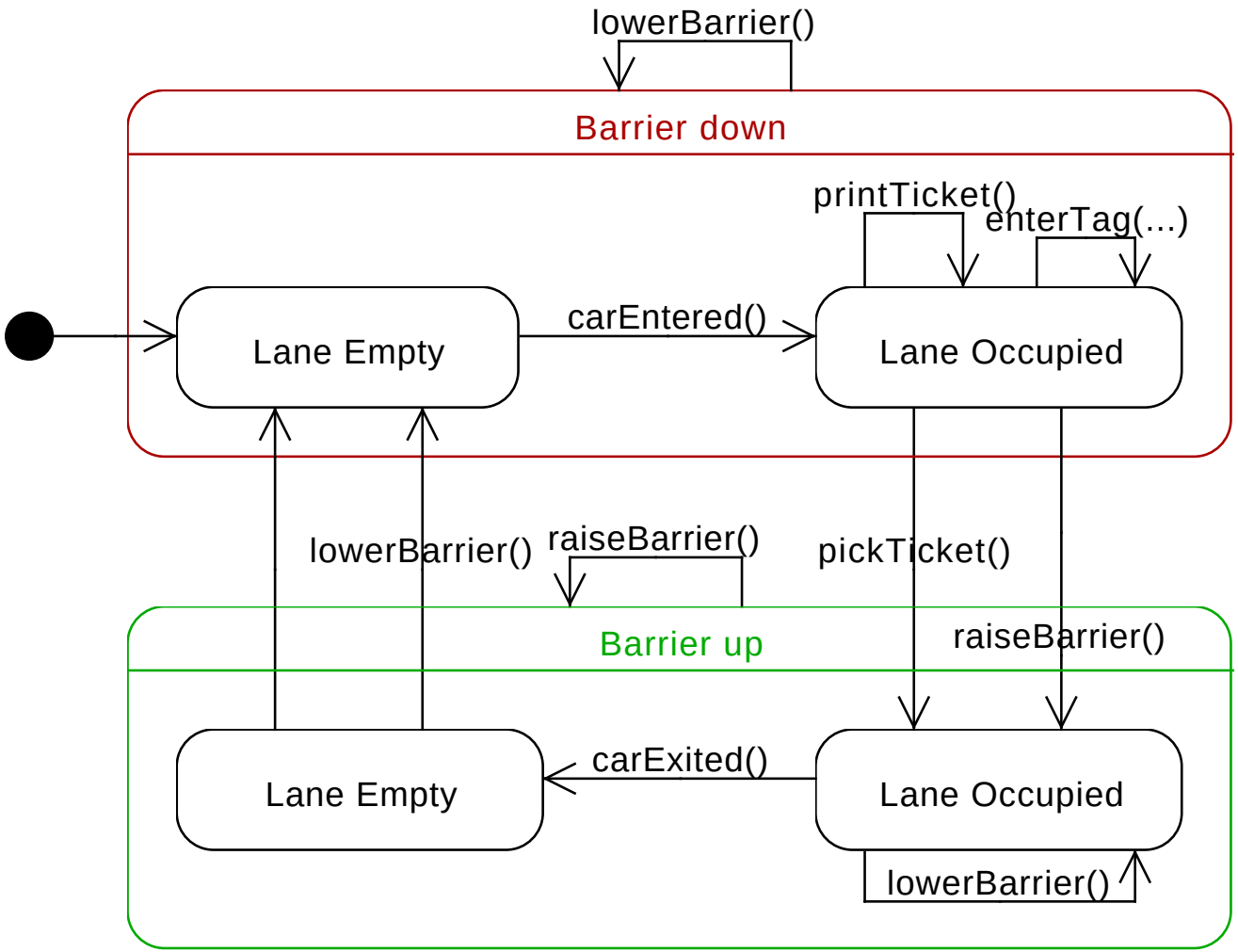
Activity Diagram
Check in with credit card

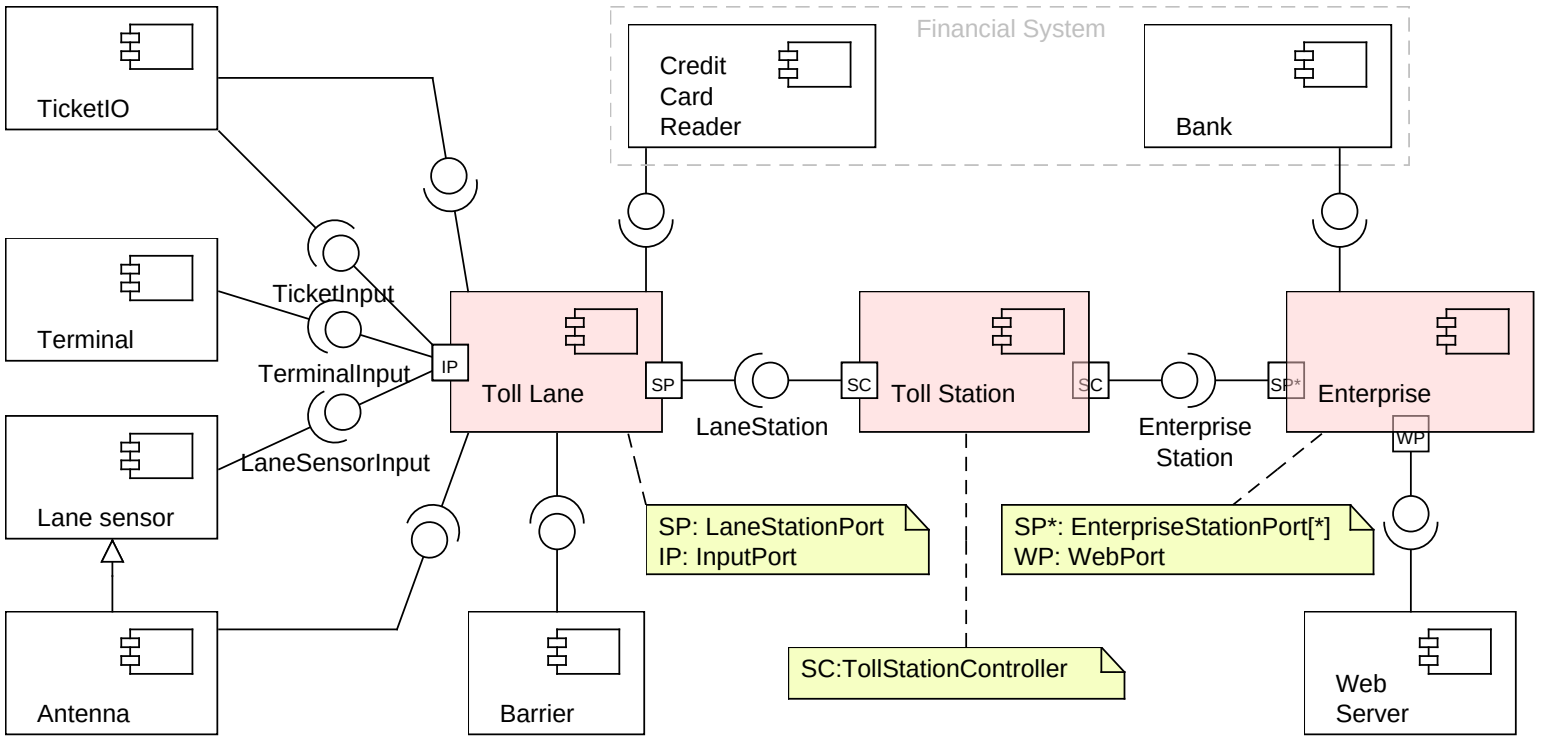


Sequence Diagram
Check in with credit card



State Diagram
Toll lane input port




TicketIO
Terminal
Lane sensor
Antenna
Barrier
Financial System
Credit Card Reader
Bank
TicketInput
TerminalInput
LaneSensorInput
IP
Toll Lane
SP
LaneStation
SC
Toll Station
SC
Enterprise Station
SP*
Enterprise
WP
SP: LaneStationPort
IP: InputPort
SP*: EnterpriseStationPort[*]
WP: WebPort
SC:TollStationController
Web Server

Design Class Diagram
Toll lane component