Rekayasa Perangkat Lunak
OOAD dengan UML (1)

Teknik Informatika
UNIKOM

OOAD dengan UML (1)

1. OOAD
2. Pengenalan UML
3. CRC cards
4. Tipe Diagram UML
5. Structural Diagram
6. Behavioral Diagram
7. Relasi pada diagram UML
8. Use case dan Use case skenario

OOAD (Object Oriented Analysis and Design)

1. Salah satu pendekatan analisis dan desain yang bisa digunakan selain terstruktur
2. OOAD bukan dipilih berdasarkan bahasa pemrograman yang digunakan.
3. Pola pikir yang menitik beratkan pada perekayasaan objek beserta relasinya.

OOAD (Object Oriented Analysis and Design)

1. **Analysis** — understanding, finding and describing concepts in the problem domain.
2. **Design** — understanding and defining software solution/objects that *represent* the analysis concepts and will eventually be implemented in code.
3. **OOAD** — Analysis is object-oriented and design is object-oriented. A software development approach that emphasizes a logical solution based on objects.

Objek

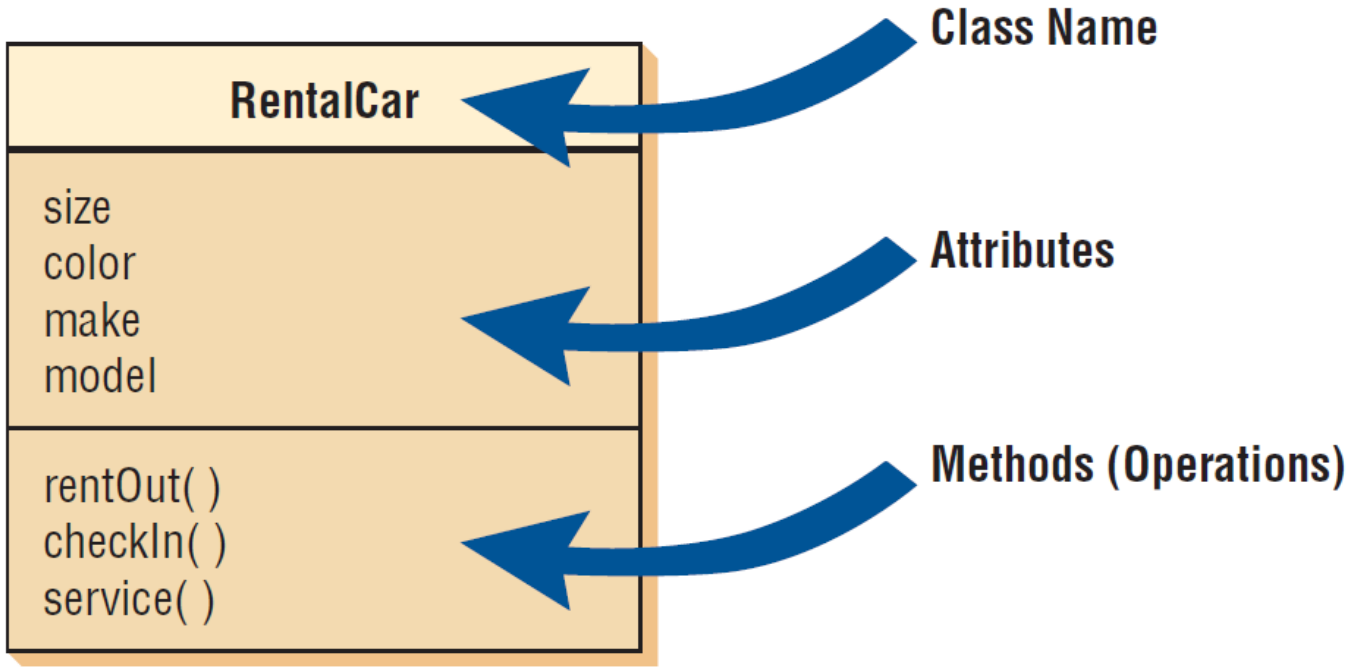
1. Objek adalah konsepsi atau benda di dunia nyata yang bisa dibedakan satu dengan yang lainnya.
2. Objek dapat dibentuk dari domain permasalahan yang diambil.
3. Objek mempunyai identitas, properti, dan tingkah laku.
4. Merupakan hasil instansiasi dari kelas.

Objek

Kelas

1. Sekumpulan objek yang memiliki kemiripan dalam hal properti, atribut, behaviour, dan semantik.
2. Proses klasifikasi dilakukan untuk membentuk kelompok dari beberapa objek yang memiliki kemiripan.

Kelas


RentalCar
size
color
make
model
rentOut( )
checkIn( )
service( )
Class Name
Attributes
Methods (Operations)

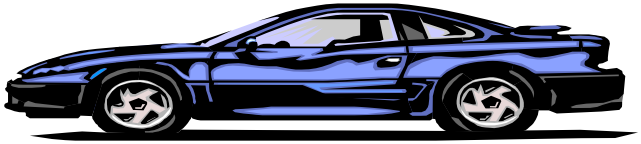
Abstraksi

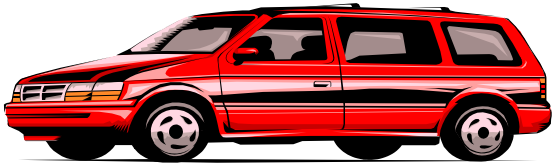
1. Fokus terhadap esensi.
2. Menghilangkan sejumlah detail.
3. Fokus terhadap “is and does” dari sebuah objek.

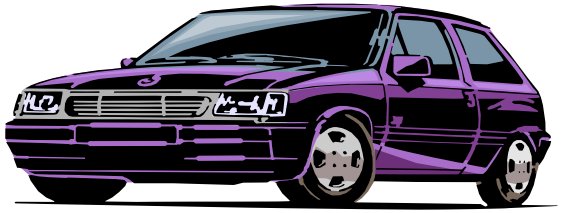
Encapsulation

1. Dikenal sebagai information hiding.
2. Pembungkusan pada objek:
    a. Property
    b. Behaviour.

Contoh Abstraksi dan Encapsulation







<<instanceOf>>

<<instanceOf>>

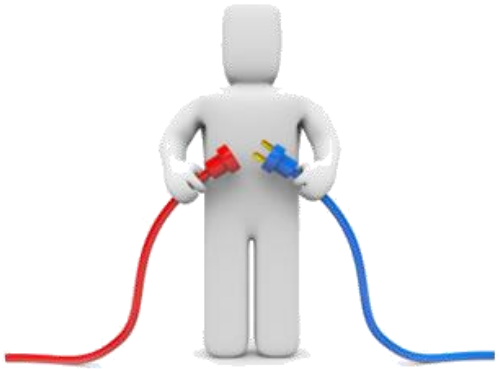
<<instanceOf>>

Class Car

Attributes
- Model
- Location
- #Wheels = 4

Operations
- Start
- Accelerate

OOAD



UNIFIED MODELING LANGUAGE™

Sejarah UML

1. OO languages muncul pada pertengahan tahun 70 sampai 80.
2. Antara tahun 89 sampai 94, metode OO meningkat dari 10% sampai 50 %.
3. Dicetuskan oleh Three Amigos:
    a. Grady Booch - Fusion
    b. James Rumbaugh – Object Modeling Technique (OMT)
    c. Ivar Jacobson – Object-oriented Software Engineering: A Use Case Approach (Objectory)
    d. (And David Harel - StateChart)

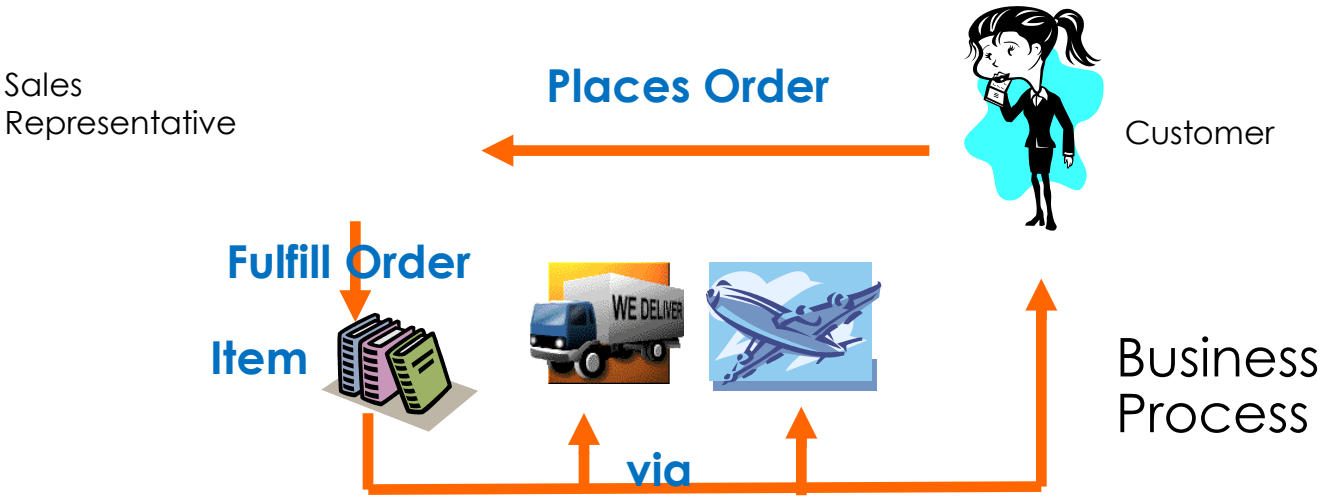
Sejarah UML

Unification of ideas began in mid 90's.
Rumbaugh joins Booch at Rational '94
v0.8 draft Unified Method '95
Jacobson joins Rational '95
UML v0.9 in June '96

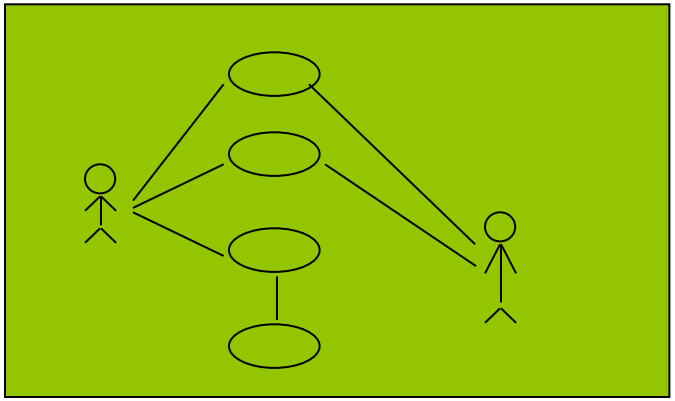
UML 1.0 offered to OMG in January '97
UML 1.1 offered to OMG in July '97
Maintenance through OMG RTF
UML 1.2 in June '98
UML 1.3 in fall '99
UML 1.5 http://www.omg.org/technology/documents/formal/uml.htm

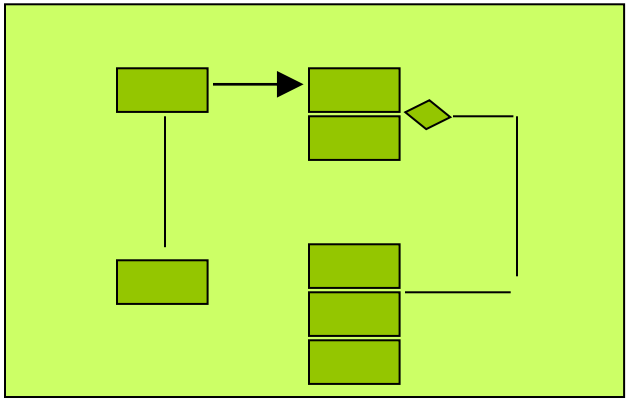
UML 2.0 underway http://www.uml.org/

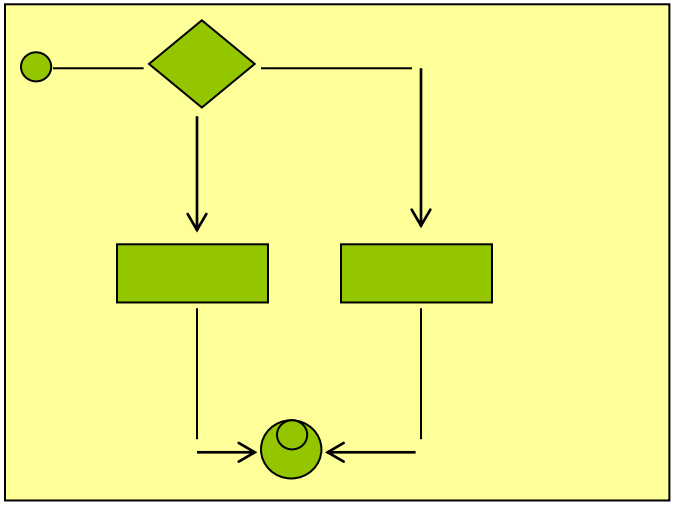
UML untuk Visual Modelling

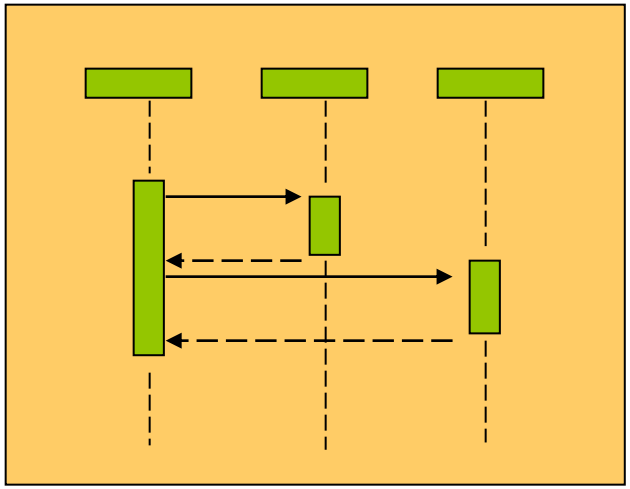


UML untuk Visual Modelling









Building Blocks of UML

1. Things - important modelling concepts
2. Relationships – tying individual things
3. Diagram – grouping interrelated collections of things and relationships

Building Blocks of UML

UML Category	UML Elements	Specific UML Details
Things	Structural Things	Classes Interfaces Collaborations Use Cases Active Classes Components Nodes
	Behavioral Things	Interactions State Machines
	Grouping Things	Packages
	Annotational Things	Notes
Relationships	Structural Relationships	Dependencies Aggregations Associations Generalizations
	Behavioral Relationships	Communicates Includes Extends Generalizes
Diagrams	Structural Diagrams	Class Diagrams Component Diagrams Deployment Diagrams
	Behavioral Diagrams	Use Case Diagrams Sequence Diagrams Communication Diagrams Statechart Diagrams Activity Diagrams

UML 1.x VS UML 2.0

UML 1.x: 9 diagram types.

Structural Diagrams

Represent the static aspects of a system.

- Class;
  Object
- Component
- Deployment

Behavioral Diagrams

Represent the dynamic aspects.

- Use case
- Sequence;
  Collaboration
- Statechart
- Activity

UML 2.0: 12 diagram types

Structural Diagrams

- Class;
  Object
- Component
- Deployment
- Composite Structure
- *Package*

Behavioral Diagrams

- Use case
- Statechart
- Activity

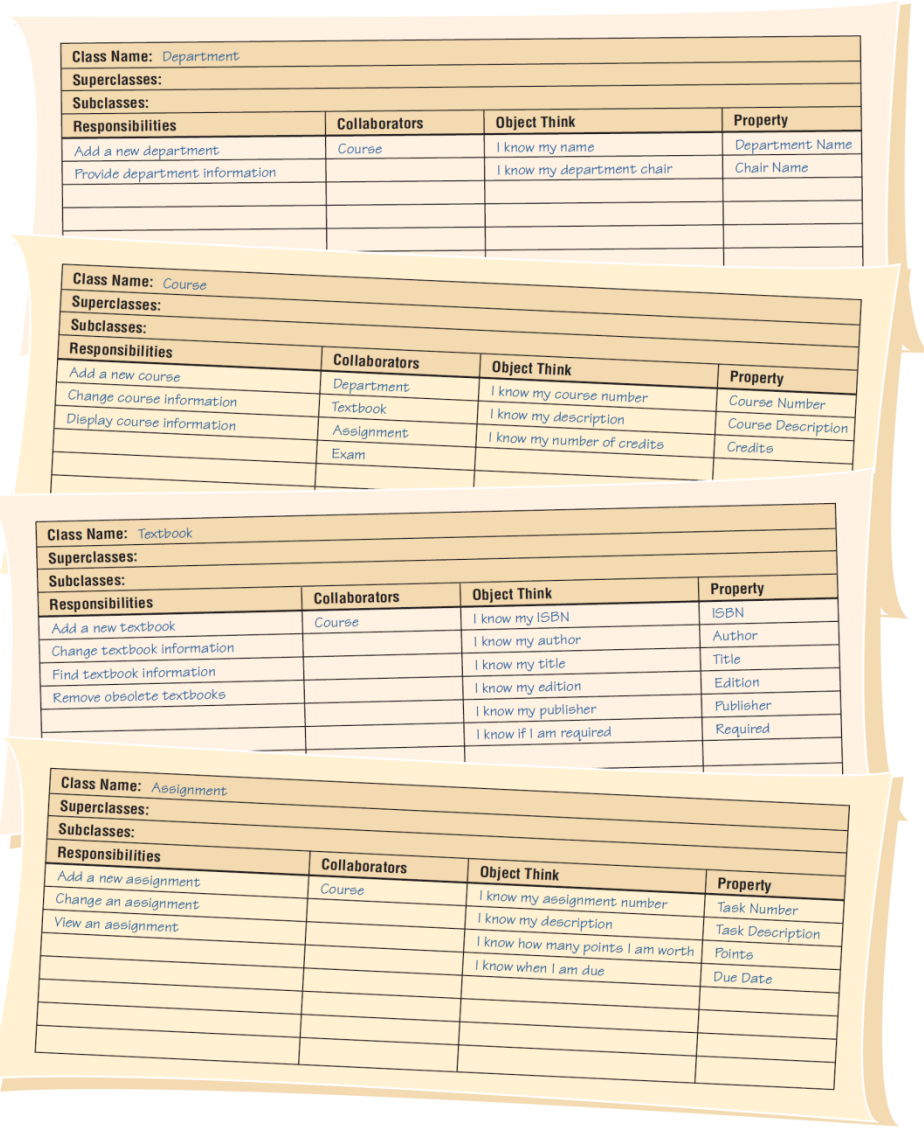
Interaction Diagrams

- Sequence;
  Communication
- Interaction Overview
- Timing

CRC Cards

1. CRC
    a. Class
    b. Responsibilities
    c. Collaborators
2. CRC cards are used to represent the responsibilities of classes and the interaction between the classes

CRC Cards



Class Name: Department			
Superclasses:			
Subclasses:			
Responsibilities	Collaborators	Object Think	Property
Add a new department	Course	I know my name	Department Name
Provide department information		I know my department chair	Chair Name

Class Name: Course			
Superclasses:			
Subclasses:			
Responsibilities	Collaborators	Object Think	Property
Add a new course	Department	I know my course number	Course Number
Change course information	Textbook	I know my description	Course Description
Display course information	Assignment	I know my number of credits	Credits
	Exam		

Class Name: Textbook			
Superclasses:			
Subclasses:			
Responsibilities	Collaborators	Object Think	Property
Add a new textbook	Course	I know my ISBN	ISBN
Change textbook information		I know my author	Author
Find textbook information		I know my title	Title
Remove obsolete textbooks		I know my edition	Edition
		I know my publisher	Publisher
		I know if I am required	Required

Class Name: Assignment			
Superclasses:			
Subclasses:			
Responsibilities	Collaborators	Object Think	Property
Add a new assignment	Course	I know my assignment number	Task Number
Change an assignment		I know my description	Task Description
View an assignment		I know how many points I am worth	Points
		I know when I am due	Due Date

Tipe Diagram UML

1. Structural Diagrams – Digunakan untuk mendeskripsikan relasi antar kelas.
2. Behavior Diagrams – Digunakan untuk mendeskripsikan interaksi antara aktor dan sebuah use case (bagaimana seorang aktor menggunakan sistem).

Structural Diagram

1. Class diagram
2. Object diagram
3. Component diagram
4. Deployment diagrams

Behavioral Diagram

1. Use case diagram
2. Sequence diagram
3. Collaboration diagram
4. Statechart diagram
5. Activity diagram

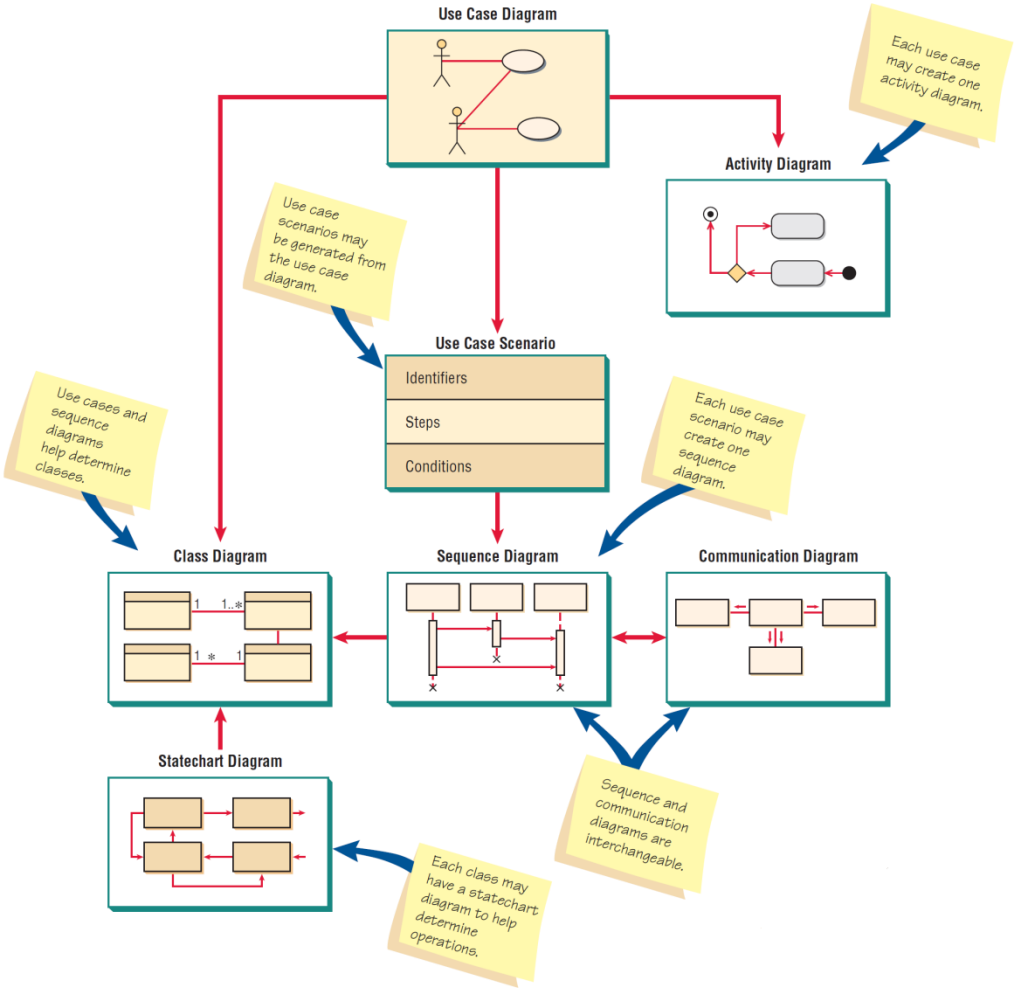
Commonly Used Diagrams

- **Use case diagram**
  - Describing how the system is used
  - The starting point for UML modeling
- **Use case scenario**
  - A verbal articulation of exceptions to the main behavior described by the primary use case
- **Activity diagram**
  - Illustrates the overall flow of activities

Commonly Used Diagrams

- **Sequence diagrams**
  - Show the sequence of activities and class relationships
- **Class diagrams**
  - Show classes and relationships
- **Statechart diagrams**
  - Show the state transitions

Relasi Antar Diagram UML

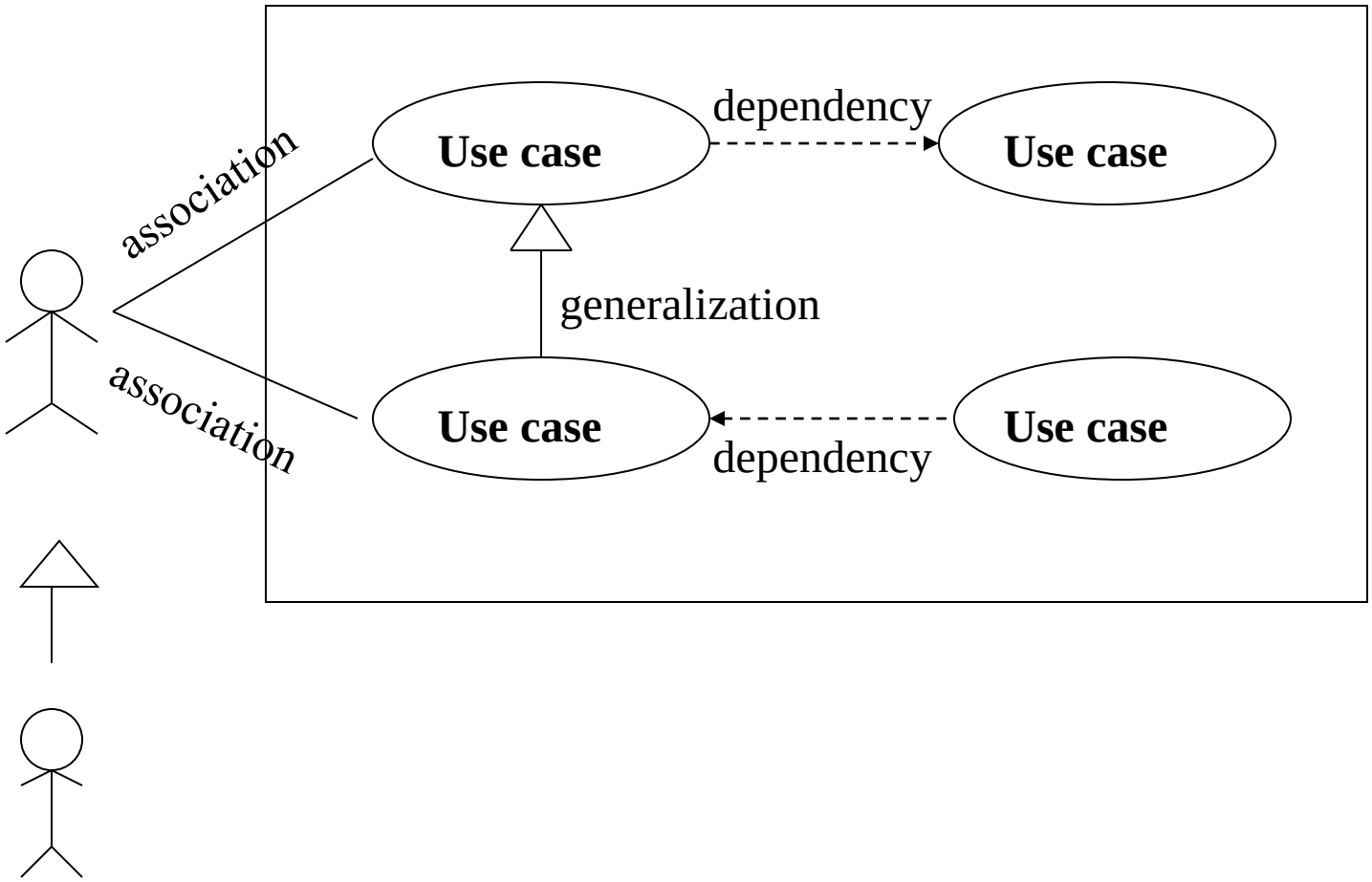


USE CASE

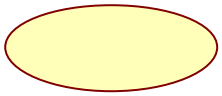
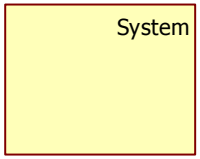
Deskripsi Use Case

1. Mendeskripsikan apa yang sistem lakukan tanpa mendeskripsikan bagaimana sistem menyelesaikannya.
2. Dibuat berdasarkan interaksi dan relasi dari individual use case.
3. Berisi aktor, event, dan use case.

Overview Use Case

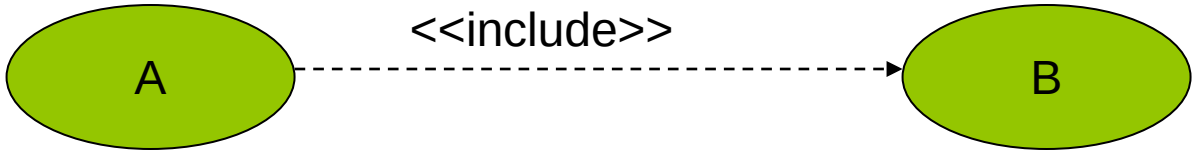


Simbol Use Case

SIMBOL	NAMA SIMBOL	FUNGSI
	Aktor	Pihak yang mengakses use case
	Use Case	Mewakili apa yang sistem bisa lakukan
	Association	Merelasikan aktor dengan use case
	System Boundary	Menggambarkan batasan sistem terhadap lingkungannya

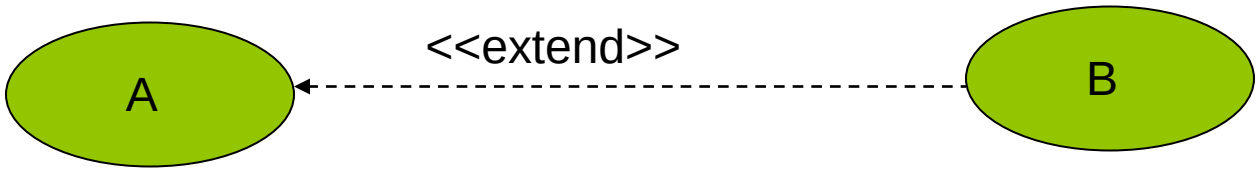
Relasi Pada Use Case

- **“Include”** dependency
  - Satu use case bisa meng-include use case lainnya
  - Jika use case A meng-include use case B maka use case B harus diimplementasi setiap kali use case A dipanggil
  - Direpresentasikan dengan garis panah putus-putus bertuliskan <<include>> ke arah use case yang diinclude



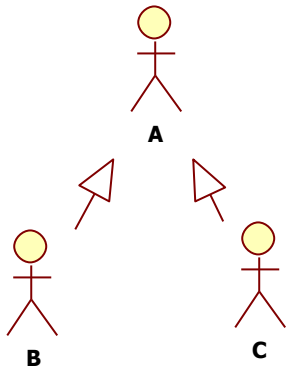
Relasi Pada Use Case

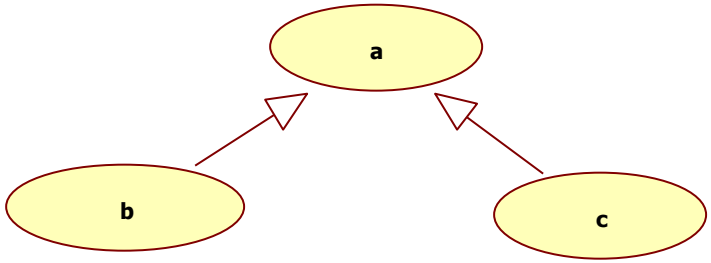
- "**extend**" dependency
  - Satu use case bisa di-extend oleh use case lain
  - Jika use case A di-extend oleh use case B maka antara A dan B dapat diimplementasi dan digunakan secara bebas.
  - Direpresentasikan dengan garis panah putus-putus bertuliskan <<extend>>
  - A tidak harus selalu memanggil B dalam beberapa batasan.



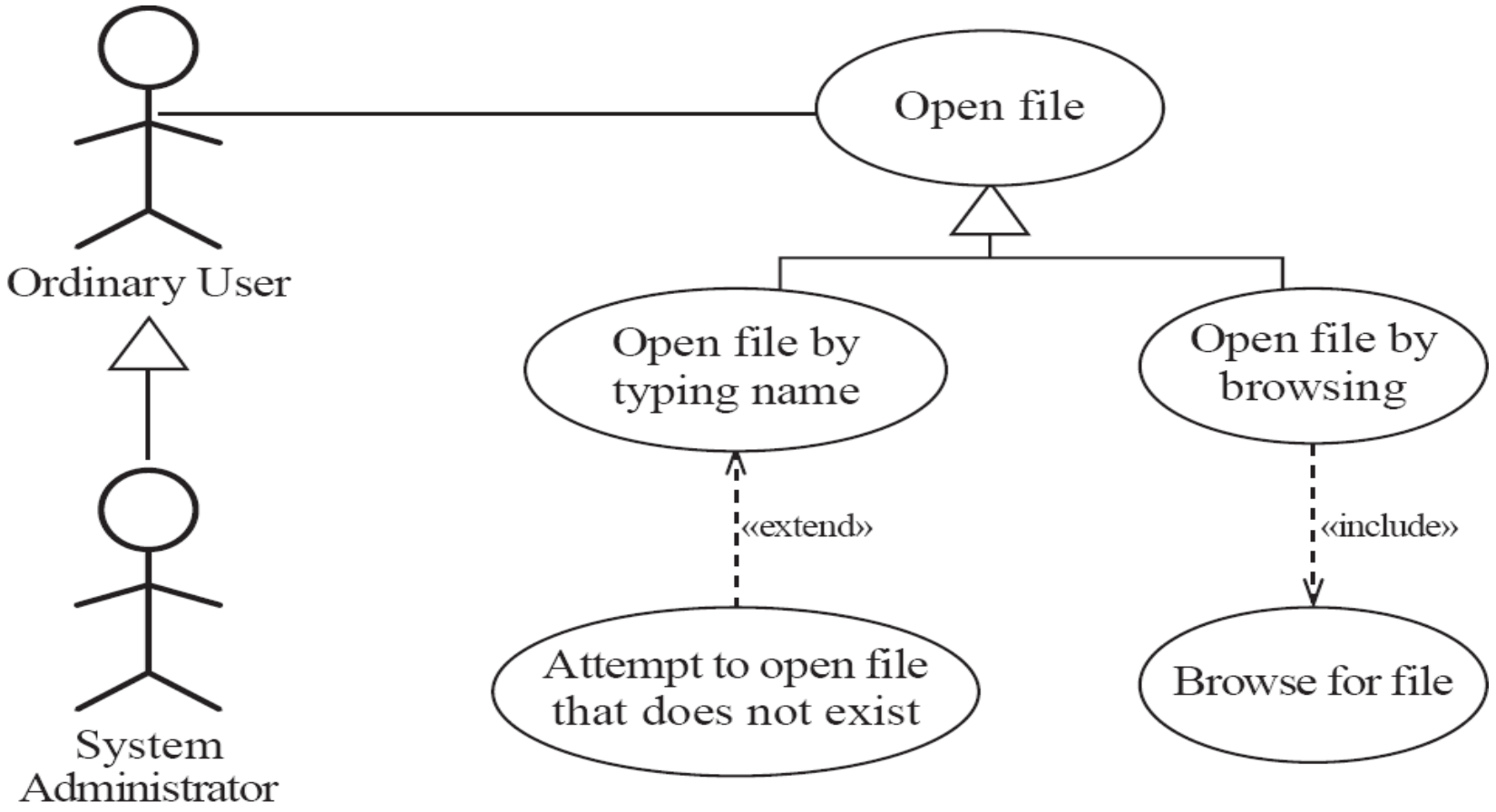
Relasi Pada Use Case

- "**generalization**" dependency
  - Aktor dan use case bisa digeneralisasi
  - Generalisasi digunakan untuk membuat aktor atau use case yang lebih spesifik dari suatu aktor dan use case.





Generalization



Use Case Skenario

1. Use case skenario merupakan hasil instansiasi dari setiap use case.
2. Terbagi menjadi tiga bagian, yaitu:
    a. identifikasi dan inisiasi
    b. step perfomed
    c. Kondisi, asumsi dan pertanyaan

Use case name:	Change Student Information	UniqueID: Student UC 005
Area:	Student System	
Actor(s):	Student	
Description:	Allow student to change his or her own information, such as name, home address, home telephone, campus address, campus telephone, cell phone, and other information using a secure Web site.	
Triggering Event:	Student uses Change Student Information Web site, enters student ID and password, and clicks the **Submit** button.	
Trigger type:	☒ External ☐ Temporal	
Steps Performed (Main Path)		**Information for Steps**
1. Student Logons on to the secure Web server.		Student ID, Password
2. Student record is read and password is verified.		Student Record, StudentID, Password
3. Current student personal information is displayed on the Change Student Web page.		Student Record
4. Student enters changes on the Change Student Web form and clicks **Submit** button.		Change Student Web Form
5. Changes are validated on the Web server.		Change Student Web Form
6. Change Student Journal record is written.		Change Student Web Form
7. Student record is updated on the Student Master.		Change Student Web Form, Student Record
8. Confirmation Web page is sent to the student.		Confirmation Page
Preconditions:	Student is on the Change Student Information Web page.	
Postconditions:	Student has successfully changed personal information.	
Assumptions:	Student has a browser and a valid user ID and password.	
Requirements Met:	Allow students to be able to change personal information using a secure Web site.	
Outstanding Issues:	Should the number of times a student is allowed to logon be controlled?	
Priority:	Medium	
Risk:	Medium	