
UML Overview

Deskripsi:

- OBJECTIVE:

  Lebih memahami RPL dalam penerapannya menggunakan UML untuk pemodelan software systems.
- TARGET:

  Diharapkan mahasiswa mengenal pengetahuan tentang konsep modeling dan Object Oriented.
- PREREQUISITES:

___Lebih memahami konsep object.

___Mengetahui tentang bahasa berorientasi object.

Mengetahui software engineering process.

Pemahaman Modeling

- Apakah model itu?
  - Model adalah penyederhanaan sesuatu yg real
- Mengapa dilakukan modeling?
  - Membantu visualisasi
  - Memberikan specification
  - Memberikan sebuah template
  - dokumentasi

Keuntungan Utama OOAD:

- Object oriented pendekatan jalan berpikirnya menggunakan konsep dunia nyata sedangkan Struktur Analisa menggunakan konsep fungsi.

- Alasan menggunakan pendekatan OOAD :
  - Pemahaman yang lebih baik tentang kebutuhan user
  - Memberikan perancangan lebih jelas
  - Design lebih flexibel
  - Penguraian system lebih konsisten
  - Memudahkan abstraksi data dan informasi yang tersembunyi
  - Software reuse
  - Maintenance mudah
  - Implementasi flexibel

Apakah UML?

- Adalah Unified Modeling Language, yang mana berisi kumpulan notasi graphical digunakan untuk mengekpresikan analisa dan desain.
- UML adalah bahasa visualisasi, spesifikasi, konstruksi dan dokumentasi yang merupakan artifacts dari system PL.
- UML adalah bahasa visual modeling untuk memodelkan systems
- UML bukan bagian dari notasi Booch, OMT, OOSE tapi merupakan gabungan dari ketiganya.
- adalah suatu langkah yang evolusiner, yang mana lebih ekspresif dan lebih seragam dibanding notasi yang individu.

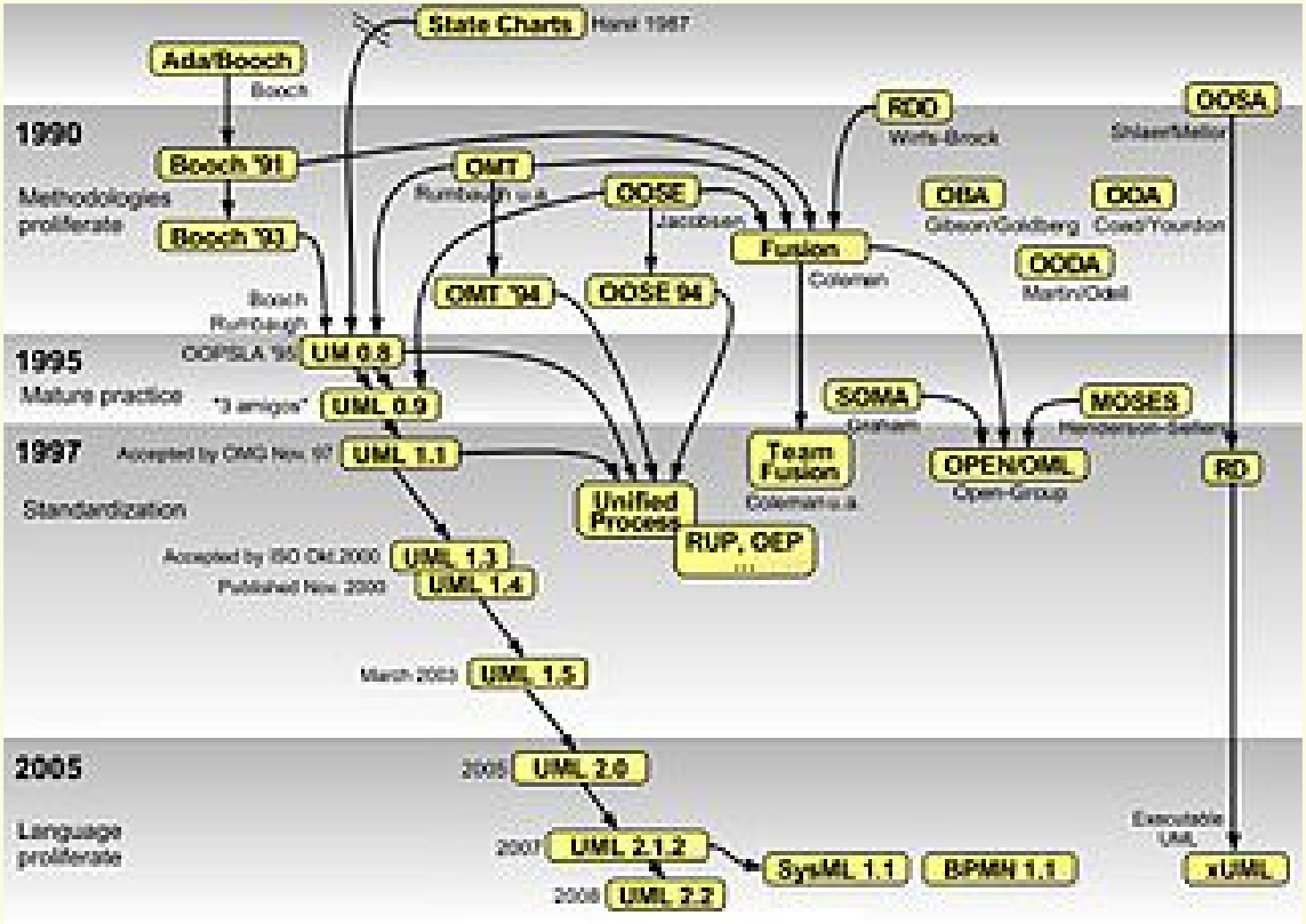
UML digunakan dimana?

- Di beberapa system, penggunaan UML berbeda beda,
- UML dapat digunakan dimana-mana
- Utamanya untuk :

  Systems software

  Business processes

Evolution UML:



Evolusi UML

- Sebelum UML 1.1
  - Menggabungkan 2 Model OO (1994)
    - Rumbaught (OMT) dan Grady Booch (OOD)
  - Ivar Jacobson (OOSE) bergabung (1995)
  - Dibentuk organisasi OMG (1996)
  - Semantic, Spesifikasi dan standarisasi (1997)
  - Release UML 1.1

Evolusi UML

- UML 1.x
  - Konsep dari berbagai metoda OO di integrasikan
  - Tony Wasserman dan peter pircher (OOSD) diintegrasikan
  - UML sebagai standard internasional
  - Standard ISO, UML version 1.4.2 (2000)

Evolusi UML

- UML 2.x
  - UML 2.0 tahun 2005 by OMG
  - UML 2.1.2 Tahun 2007
  - UML 2.2 tahun 2009
  - UML 2.3 tahun 2010
  - UML 2.4 tahun 2011
  - UML 2.5 tahun 2012

Evolusi UML

- 4 bagian Spek ditambahkan V 2.x
  - UML Diagram Interchange, perubahan layout diagram UML 2
  - Object Constraint Language (V. 2.3.1)
  - Infrastructure core metamodel (V. 2.4.1)
  - Superstructure mendefinisikan notasi dan semantic diagram (V. 2.5)

Keuntungan UML:

- Menangkap proses bisnis
- Menigkatkan komunikasi dan memastikan komunikasi yang benar
- Kemampuan untuk membuat arsitektur perangkat lunak yang logis tidak terikat pada bahasa implementasi
- Mengatur kompleksitas
- Membolehkan reuse design

Apakah tool?

- Suatu peralatan yang mendukung setiap phase software development life cycle.

Mengapa menggunakanTool?

- Membantu designer merancang sw lebih cepat.
- Mendukung validations seperti:
  - *Consistency checking*
  - *Completeness checking*
  - *Constrain checking.*
- Waktu yang dibutuhkan untuk operasi dapat diprediksi .
- Code generation
- Reverse engineering.
- Round trip engineering
- Conversion dari SSAD ke OOAD
- Dokumentasi lebih cepat…dll

UML 2.0 diagrams:

1. Use case diagram
2. Class Diagram
3. Behavioral diagrams
   - State chart diagrams
   - Object diagram
   - Activity diagrams
   - Interaction diagrams
     - Sequence diagrams
     - Collaboration diagrams
4. Implementation diagrams
   - Component diagram
   - Deployment diagram

UML 2.2

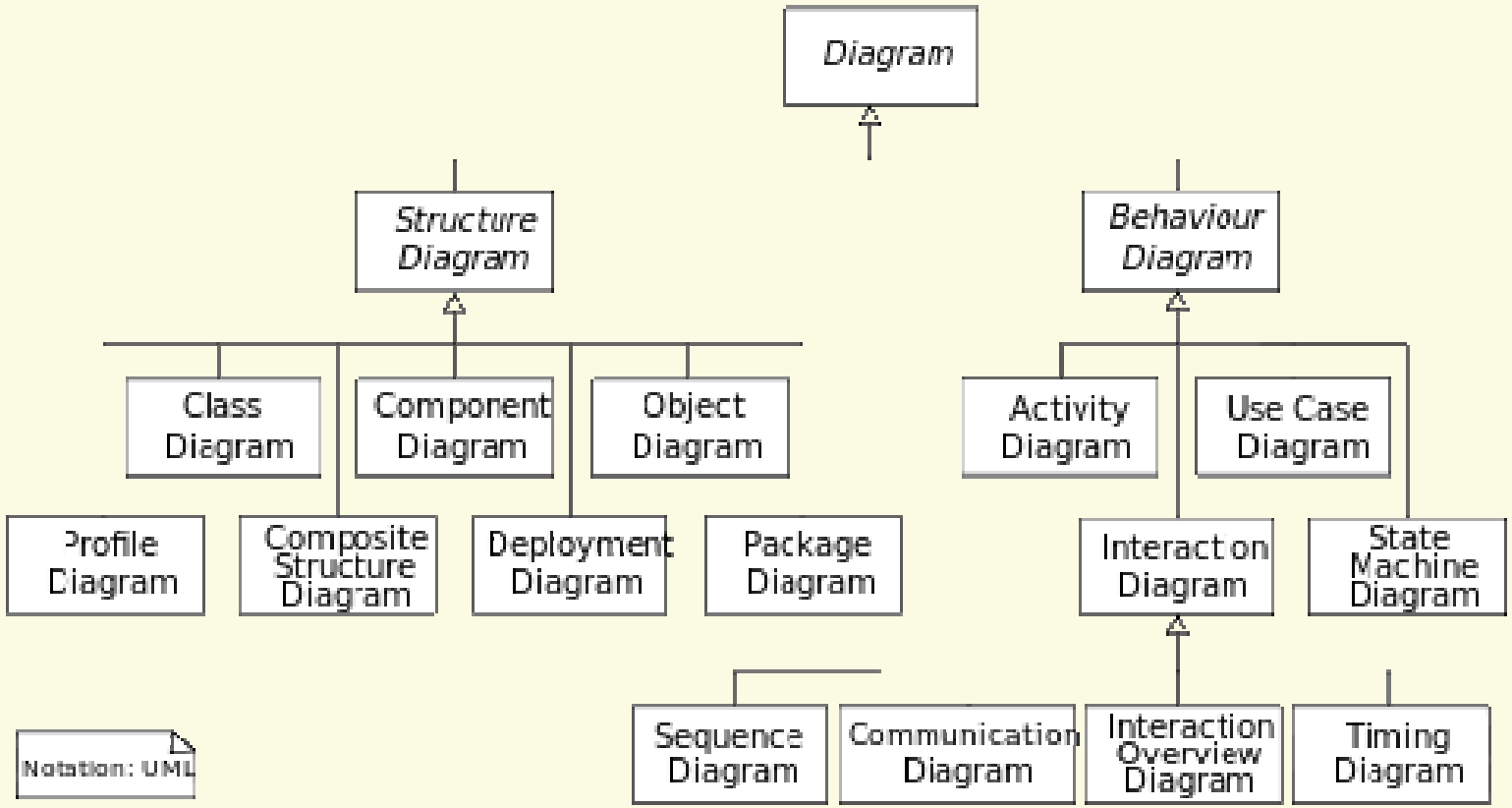


Diagram
Structure Diagram
Behaviour Diagram
Class Diagram
Component Diagram
Object Diagram
Activity Diagram
Use Case Diagram
Profile Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Interaction Diagram
State Machine Diagram
Sequence Diagram
Communication Diagram
Interaction Overview Diagram
Timing Diagram
Notation: UML

Semantics Diagrams:

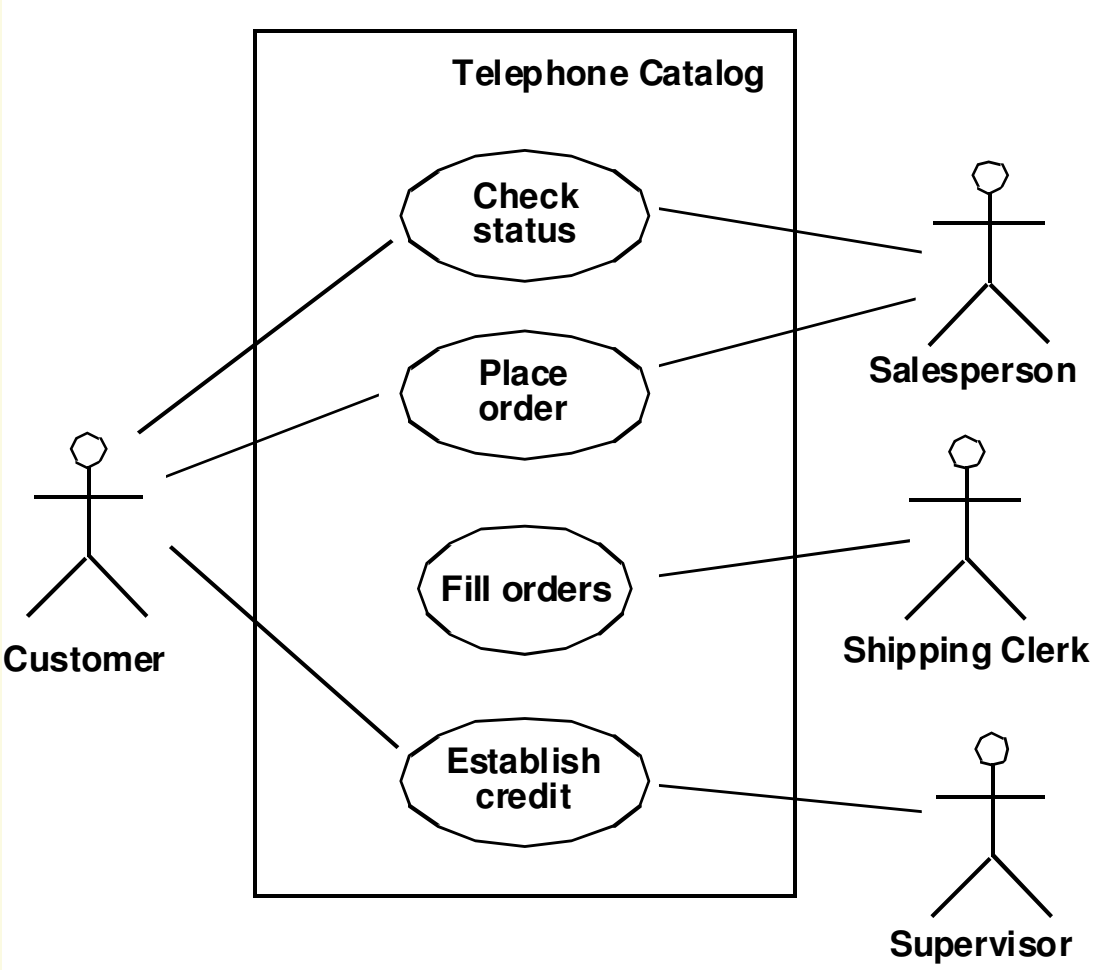
- Use case diagrams merepresentasikan fungsi system dari titik sudut pandang user.
- Class diagrams merepresentasikan struktur static dalam bentuk class class, atribut berikut relationshipnya.
- State chart diagrams merepresentasikan class dalam bentuk states
- Object diagrams merepresentasikan objects dan relationshipnya (model system terhadap waktu)
- Activity diagrams merepresentasikan secara bertahap workflow
- Sequence diagrams merepresentasikan objects dan interaksinya.
- Component diagrams merepresentasikan aplikasi komponen-komponen PL dan dependensi komponen.
- Deployment diagrams merepresentasikan Hardware yang digunakan dalam implementasi system dan environmentnya, deploy HW.

Semantics of Diagrams:

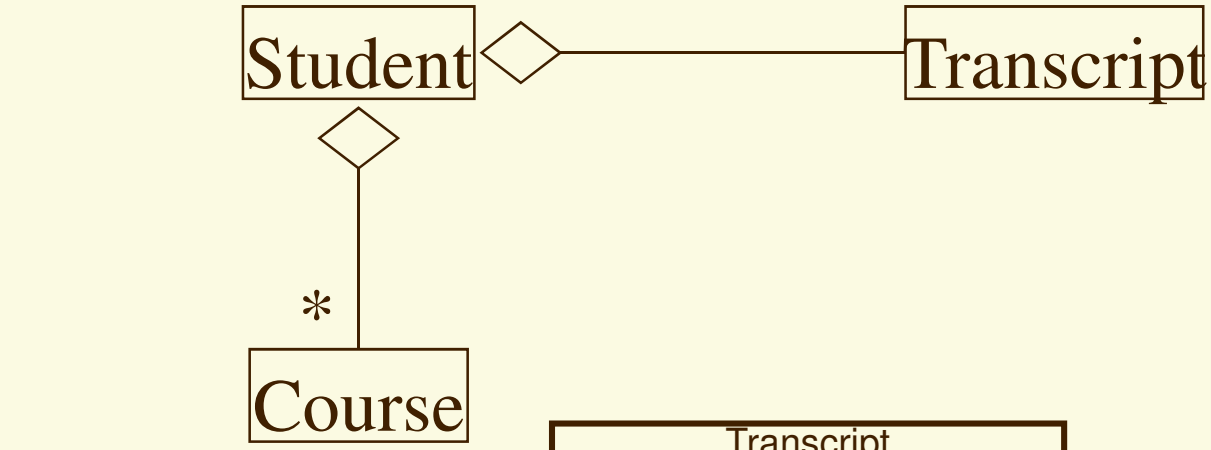
lanjutan...

- Composite Structure diagrams merepresentasikan struktur internal dari class dan kolaborasi
- Package Diagram merepresentasikan system dalam logical group
- Profile Diagram operasi pada level metamodel untuk menunjukkan stereotype sebagai class dan profile sebagai package, relasi yang mengindikasi sebagai elemen model
- Communication Diagram menunjukkan interaksi antar object
- Interaction diagram memberikan gambaran node-node yang merepresentasikan diagram interaksi
- Timing Diagram merupakan diagram interaksi yang fokus pada batasan waktu

USE CASE Diagram



CLASS Diagram

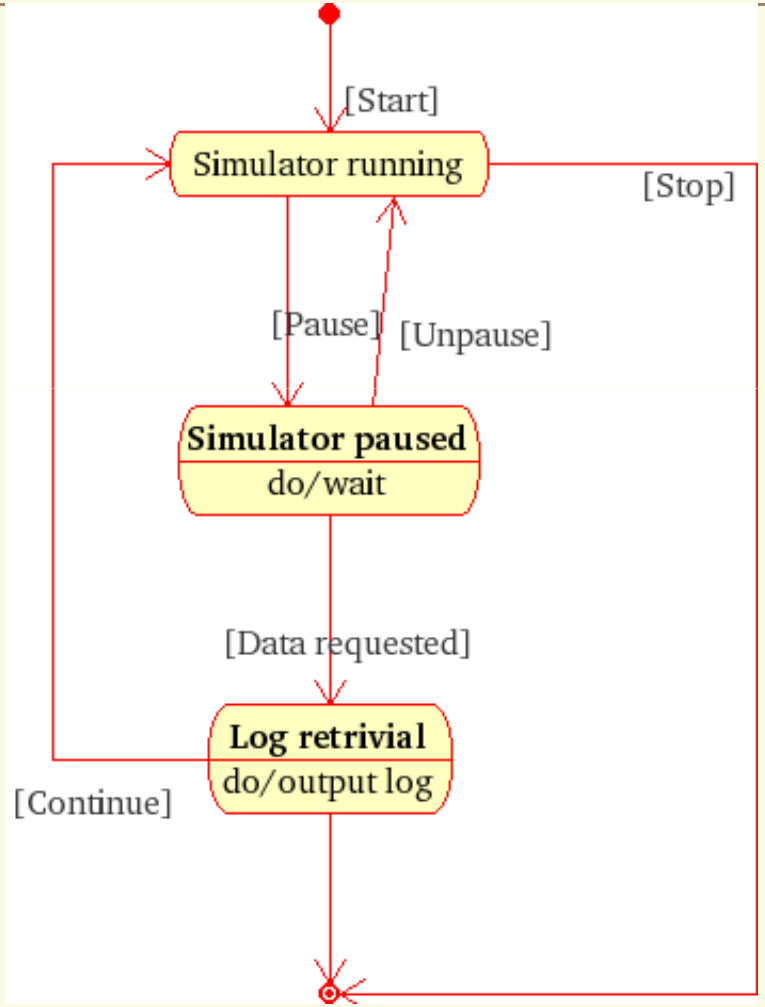


Student
studentName studentID listOfCoursesID transcript:Transcript
setStudentName() getStudentName() setStudentID() getStudentID() addToListofCourseIDs() makeTranscript()

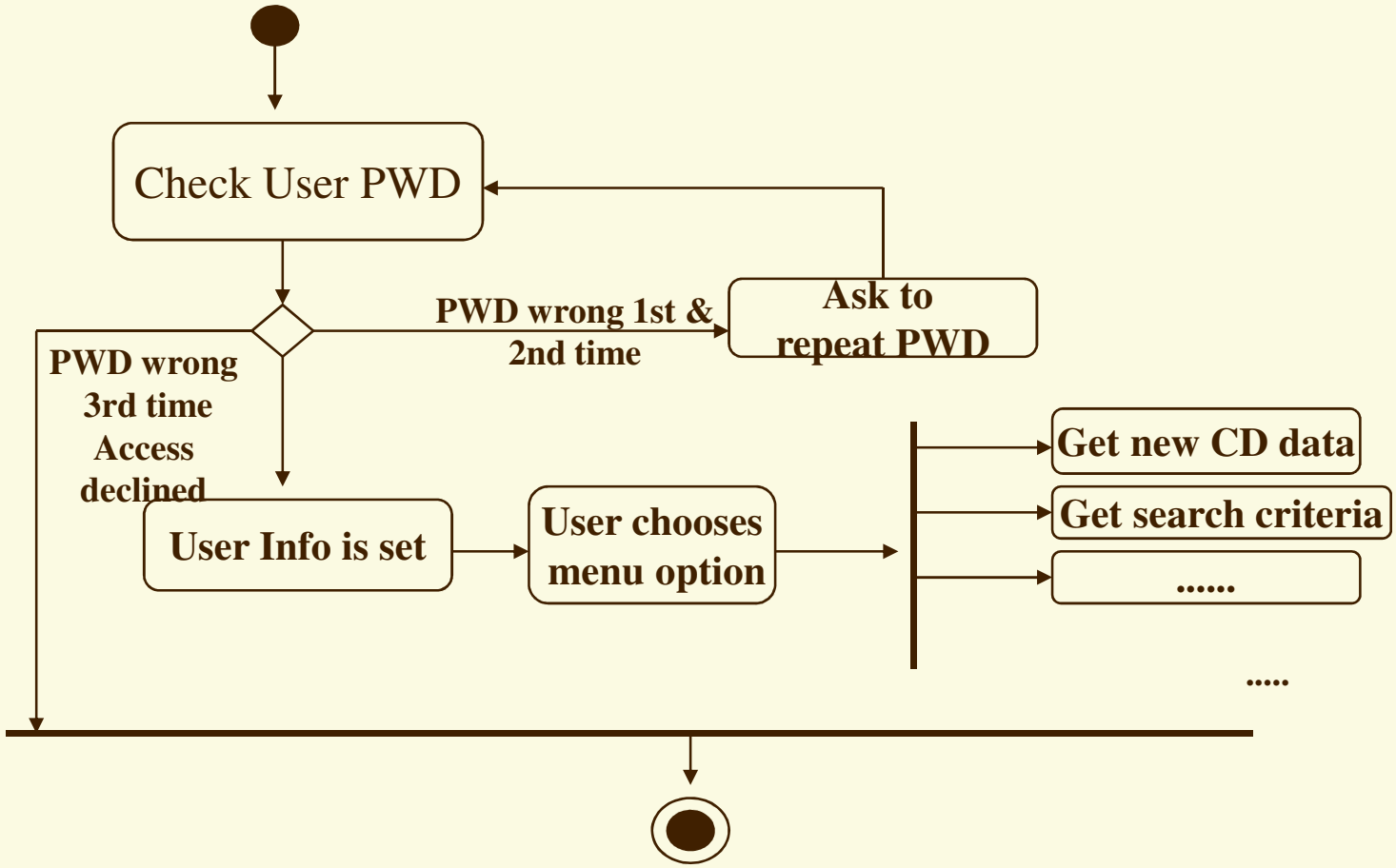
Transcript
studentName studentID listOfCourseNames lilstOfGrades average
setStudentName() setStudentID() addCourseName() addGrade() calculateAverage() show()

Course
courseName courseID grade
readCourseData() readCourseGrade() setCourseID()

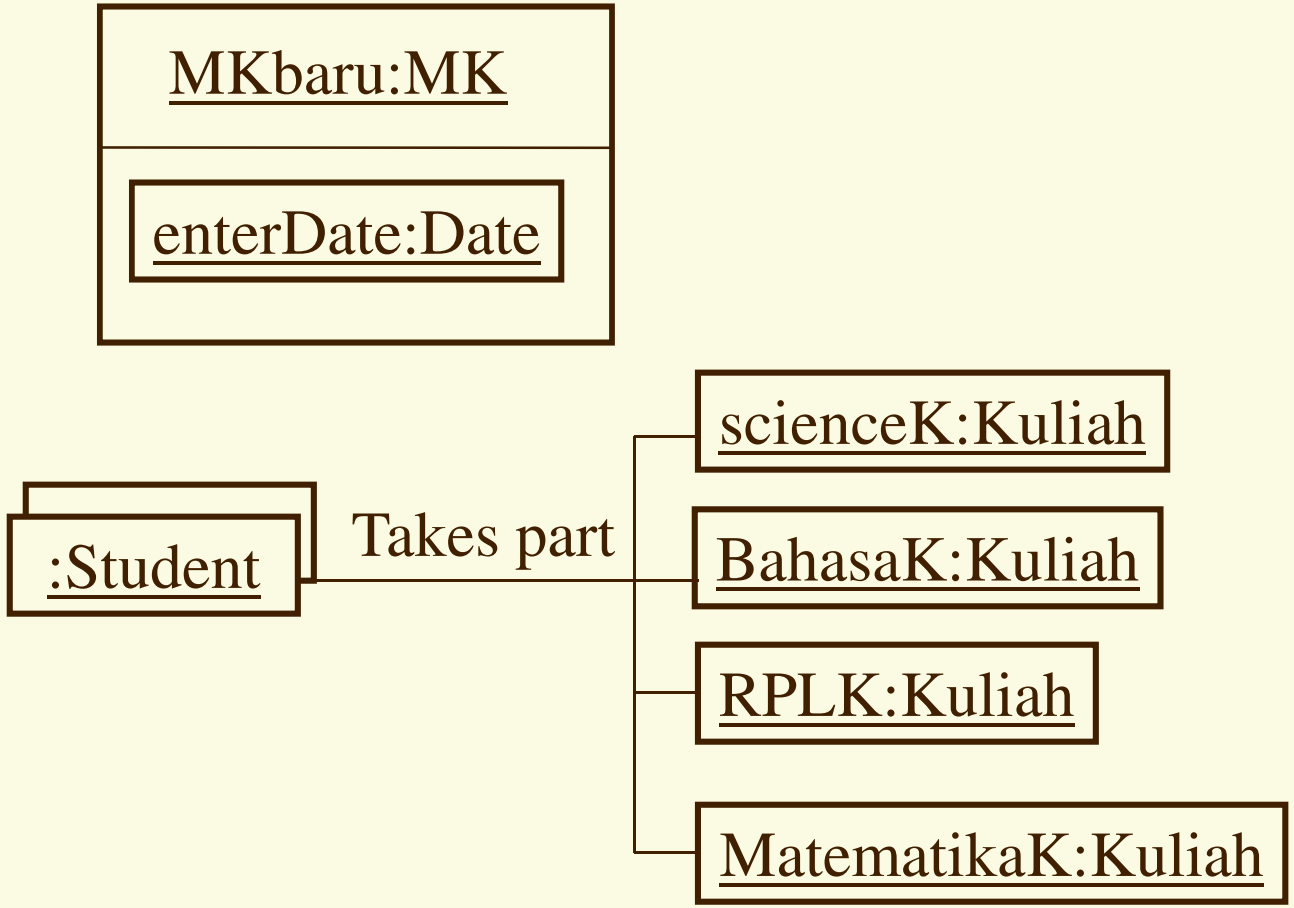
State Diagram



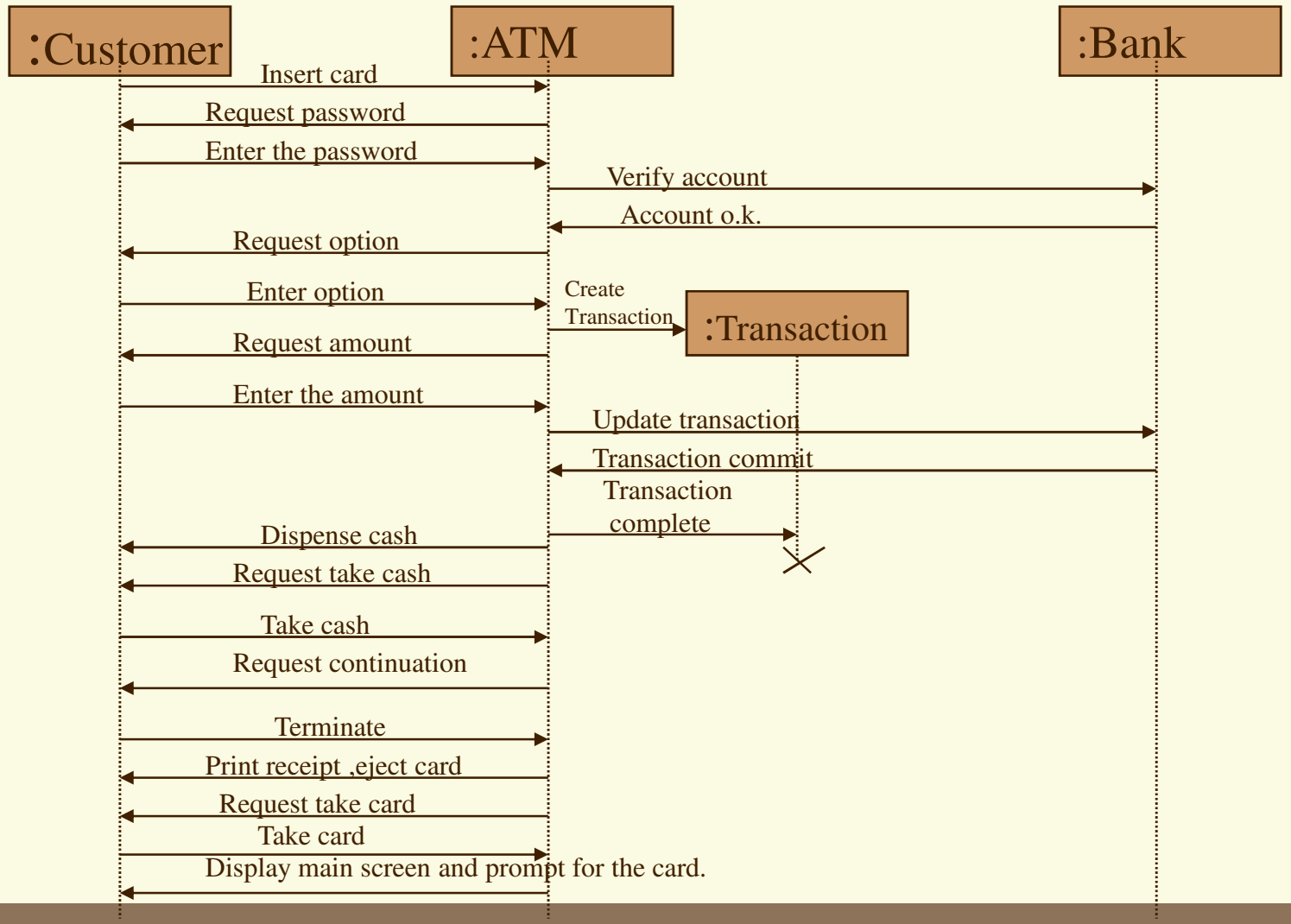
Activity Diagram



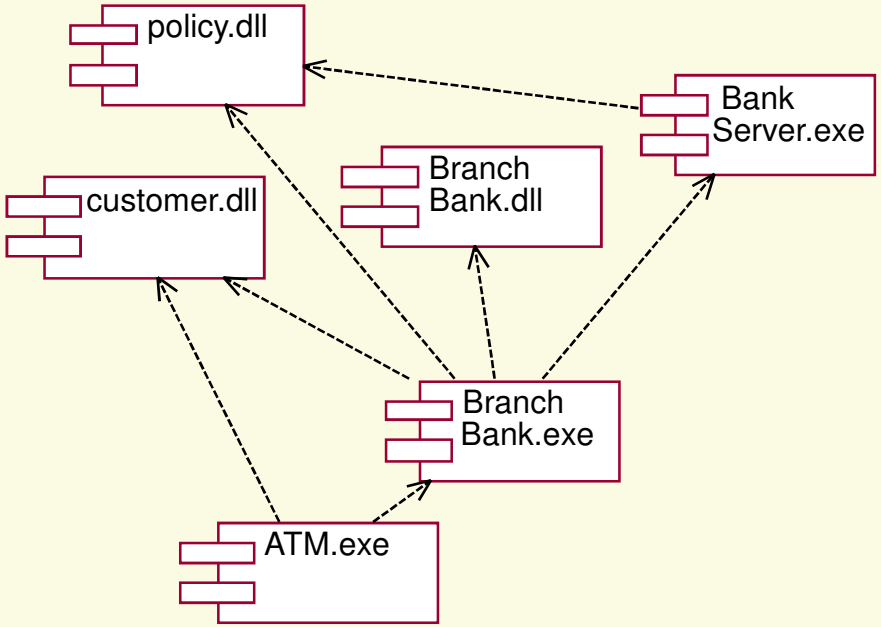
Object Diagram



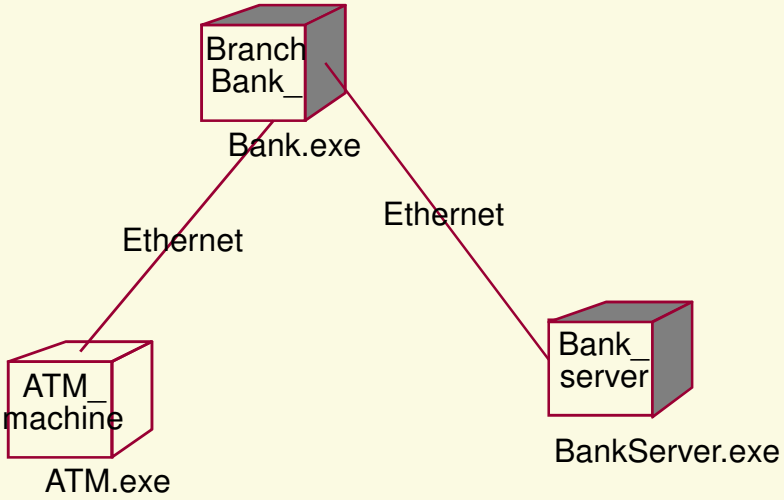
Sequence Diagram



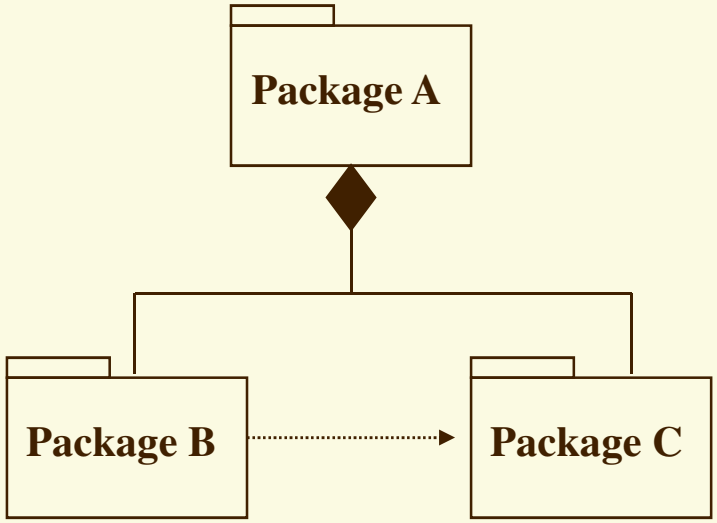
Komponen Diagram



Deployment Diagram



Package Diagram



Normal Flow Events:

Penarikan uang di ATM:

- 1.(SR) ATM meminta user memasukkan kartu.
- 2.(AA) User memasukkan kartu ATM.
- 3.(SR) ATM menerima kartu dan membaca serial number.
- 4.(SR) ATM meminta password.
- 5.(AA) user memasukkan passsword.
- 6.(SR) ATM memverifikasi serial number dan password dengan bank dan mendapatkan notifikasi.
- 7.(SR) ATM meminta user memilih macam-macam transaksi.
- 8.(AA) User memilih penarikan uang.

Normal Flow of Events:

Contd...

- 9.(SR) ATM meminta user memilih jumlah uang;
- 10. (AA) User memasukkan Rp. 300.000,-
- 11.(SR) ATM memverifikasi bahwa uang yang diminta masih didalam batas saldo dan meminta untuk memproses transaksi, mengkonfirms success dan menginformasikan saldo baru.
- 11a.(SR) ATM mengeluarkan uang dan meminta user untuk mengambilnya.
- 12.(AA) user mengambil uang.
- 13.(SR) ATM meminta user untuk melanjutkan transaksi atau tidak.
- 14.(AA) User memilih tidak

Normal Flow Events:

Contd...

- 15.(SR) ATM mengeluarkan kartu dan meminta user mengambilnya
- 16.(AA) User mengambil kartu.
- 17.(SR) ATM meminta user memasukkan kartu.

Alternative Flow Events:

Penarikan uang di ATM :

- 9. ATM meminta user memilih sejumlah uang yang akan ditarik; user tidak jadi ambil dan menekan “cancel”.

Exceptional Flow Events:

Penarikan uang di ATM :

- 3 Penggunaan kartu yang berbeda.
- 10 Mesin ATM tidak ada uang.

Mengapa flow events?

- Membantu memahami fungsional system untuk dikembangkan.
- Flow event membantu pencarian object system.
- Kejadian yang sangat penting dan langkah pertama menuju analysis dan design.