Tim Pengajar IF2150

IF2250 – Rekayasa Perangkat Lunak

Scenario-based Modeling (1)

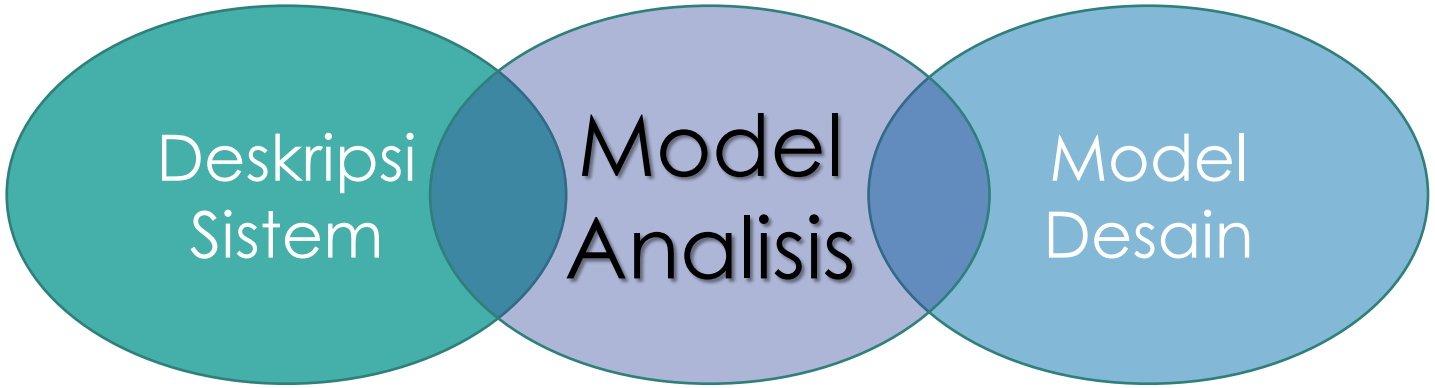
SEMESTER I TAHUN AJARAN 2024/2025

KNOWLEDGE & SOFTWARE ENGINEERING

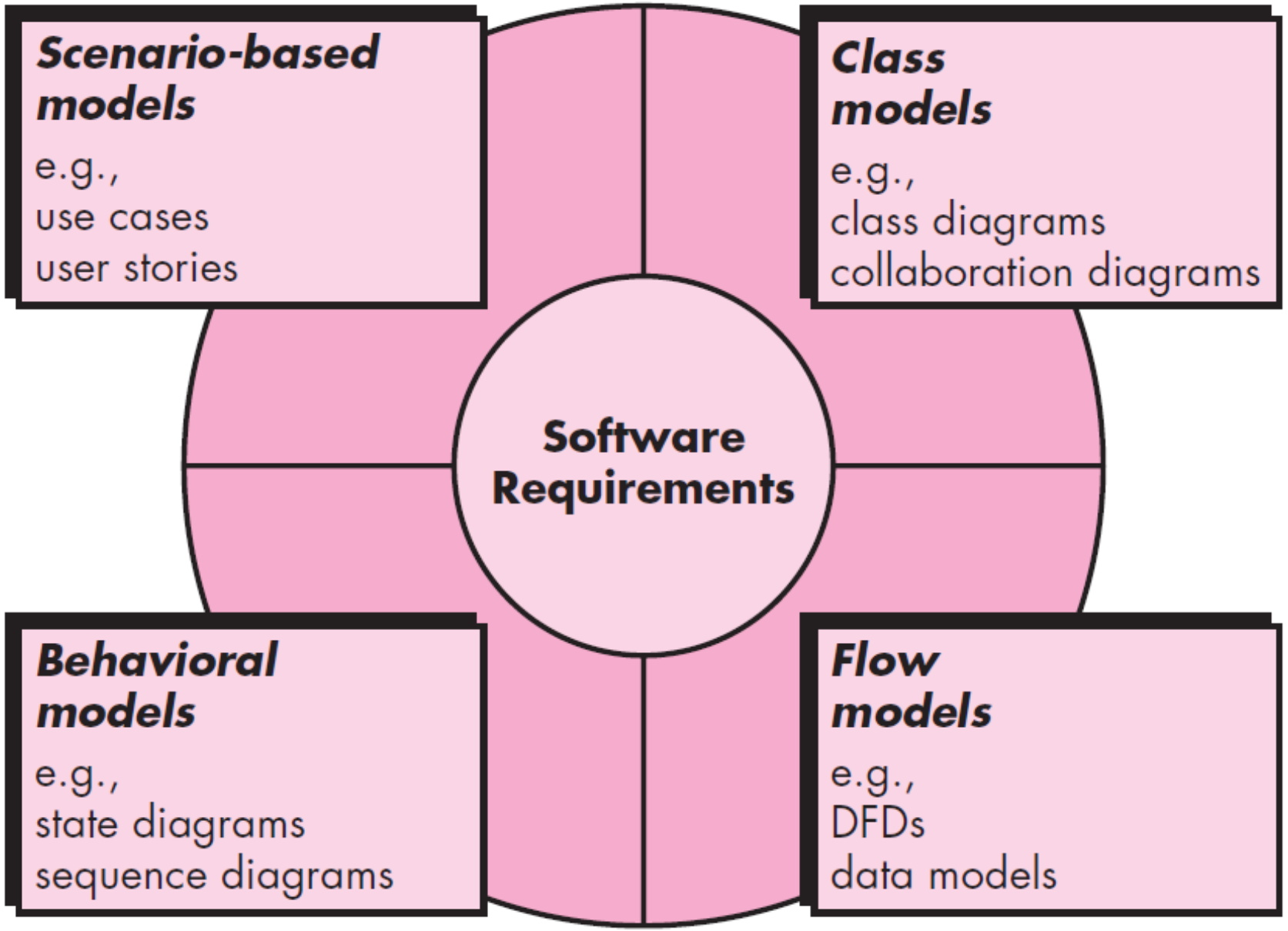
Model Analisis

KNOWLEDGE & SOFTWARE ENGINEERING

Model Analisis adalah "jembatan" antara Deskripsi Sistem dan Model Desain



- Model harus difokuskan pada kebutuhan yang ada dalam domain masalah/bisnis
- Tiap elemen digunakan untuk mengerti kebutuhan sistem
- Model dibuat sederhana


Scenario-based models
e.g.,
use cases
user stories
Class models
e.g.,
class diagrams
collaboration diagrams
Software Requirements
Behavioral models
e.g.,
state diagrams
sequence diagrams
Flow models
e.g.,
DFDs
data models

Model Analisis Kebutuhan PL

- **Model Berbasis Skenario (*Scenario-Based Models*)**
  - **Fungsional – pemrosesan cerita untuk suatu fungsi perangkat lunak**
  - ***Use-Case* – deskripsi hubungan interaksi antara aktor dan sistem**
- Model Berbasis Kelas (*Class-Based Models*)
  - Dikembangkan berdasarkan entitas yang ditemukan saat melakukan pemodelan skenario dan/atau fungsional
- Model Perilaku (*Behavioral Models*)
  - Contoh: *State Transition Diagram* atau *State Diagram*
- Model Berorientasi Aliran (*Flow-Oriented Models*)
  - Contoh: DFD

Pemodelan Berbasis Skenario (Scenario Based Modeling)

Scenario-based Modeling

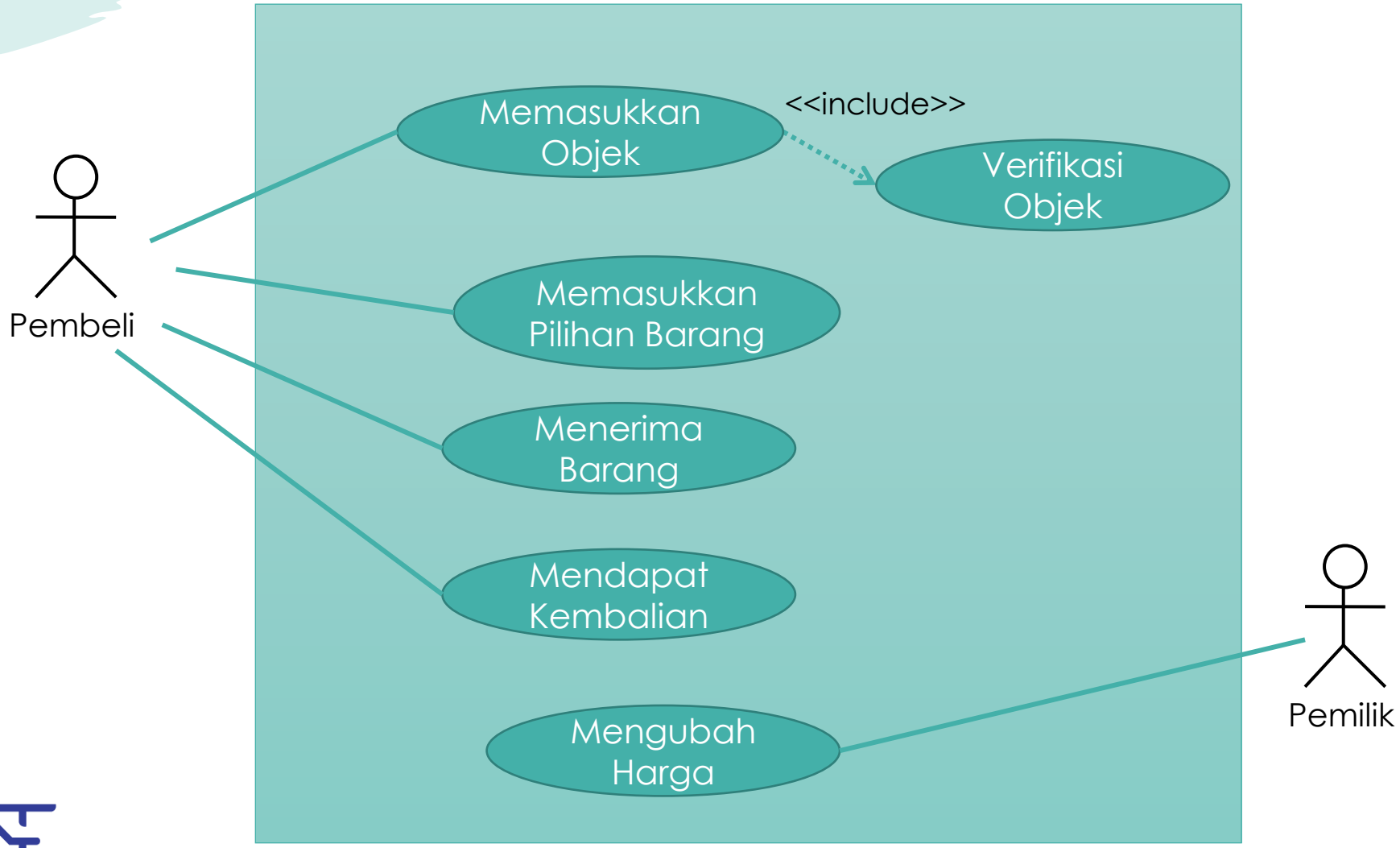
- Apa yang dimodelkan?
  - Cara pengguna **berinteraksi** dengan perangkat lunak
  - Cara perangkat lunak **berinteraksi** dengan sistem lain
- Seperti apa modelnya?
  - Diagram *use-case*
  - Diagram aktivitas (dengan dan tanpa *swimlane*)
  - Diagram interaksi (mis. *diagram sequence*)

Diagram Use-case

- *Use-case*
- Actor
- Scenario

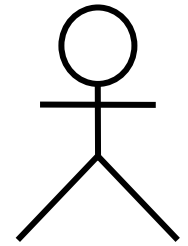
Ivar Jacobson: "*[**Use-cases**] are simply an aid to defining what exists outside the system (**actors**) and what should be performed by the system (**use-cases**).*"

Use Case untuk Vending Machine

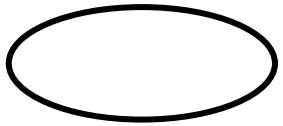


Konsep Pemodelan use-case

- Aktor mewakili semua yang **berinteraksi** dengan sistem
  - Aktor adalah unsur '**eksternal**'
- *Use-case* adalah ***urutan aksi-aksi*** dalam sistem yang melakukan suatu pekerjaan yang memberikan suatu hasil untuk aktor
  - use-case bertindak sebagai **penghubung** antara pengguna dengan pengembang
  - use-case berguna sebagai **alat komunikasi** antara pengguna dan pengembang



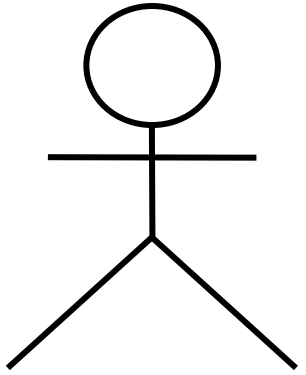
Actor



Use-Case

Actor

- Aktor bukan bagian dari sistem
- Aktor mungkin secara aktif bertukar informasi dengan sistem
- Aktor mungkin berfungsi pasif sebagai penerima informasi
- Aktor bisa merepresentasikan
  - Manusia,
  - Mesin,
  - Sistem lain



Actor

Aktor mewakili unsur Eksternal

Seorang pengguna mungkin memiliki peran berbeda

Ali

Ali sebagai mahasiswa

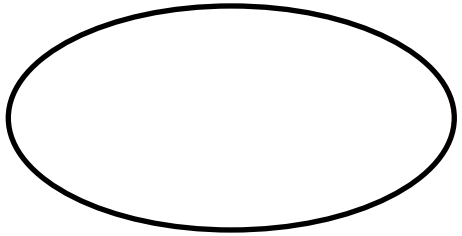
Mahasiswa

Ali sebagai Dosen

Dosen

Dengan *use-case* peran ini harus digambarkan berbeda walau dalam kenyataannya orangnya mungkin hanya satu

Use-Case



Use-Case

- *Use-case* merepresentasikan **dialog** antara aktor dengan sistem
  - *use-case* di'**inisiasi**' oleh **aktor** untuk melakukan suatu fungsi tertentu dalam sistem
- *Use-case* merepresentasikan dialog antara satu atau lebih aktor
  - sistem akan **mengembalikan** suatu **nilai** ke aktor
- *Use-case* perlu menggambarkan ***event*** yang lengkap dan memiliki makna bagi sistem
- *Use case* juga dapat dilihat sebagai tujuan sistem secara umum yang mungkin melibatkan satu atau lebih aktor
- Semua *use-case* mengarahkan ke semua penggunaan sistem

Penjelasan Use-case

- Gambar *use-case* (termasuk aktornya) perlu disertai dengan **keterangan** yang akan membantu **menjelaskan** gambaran yang diberikan
  - Keterangan ini tidak perlu panjang lebar, untuk setiap *use-case* lebih kurang **dua baris** saja.
  - Deskripsi langkah-demi langkah apa yang perlu dikerjakan sistem ketika berinteraksi dengan aktor
- Perlu dijelaskan bagaimana antar *use-case* **saling terkait**
  - Atau suatu *use case* yang terlibat pada suatu *use-case* lain
  - Bagaimana *use case* itu dilakukan oleh actor

→ **Deskripsi Use Case**

Deskripsi Use Case (1)

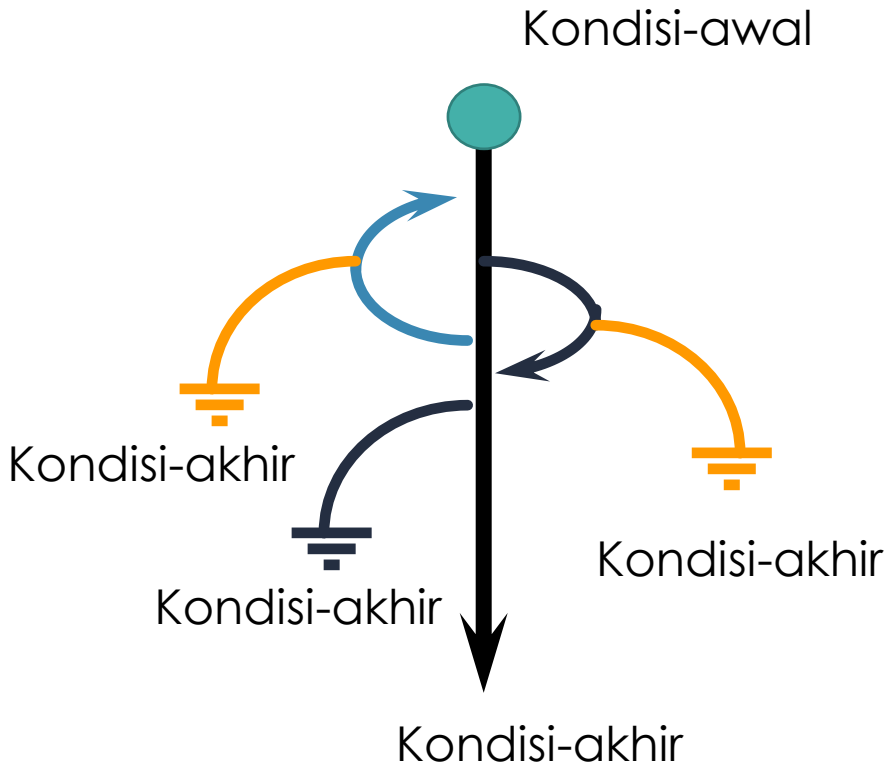
- Penjelasan **Kondisi awal**
- Bagaimana dan kapan *use-case* **dimulai**
- Bagaimana urutan aksinya
- Bagaimana dan kapan *use-case* selesai
- Kemungkinan *post*-kondisi alternatif
- Jalur eksekusi yang tidak diperbolehkan
- Jalur alternatif (diambil dari jalur dasar)
- Interaksi sistem dengan aktor
- Penggunaan objek, sumber daya dalam sistem
- Penjelasan apa yang dilakukan oleh sistem

Aliran kejadian (flow of event) (1)

- Aliran kejadian pada use case:
  - Step-by-step pelaksanaan aksi pada use case
  - Aliran yang '**normal**', atau aliran 'dasar' atau jalur dasar (*basic path*) atau "*Happy path*"
  - Aliran **alternatif**
    - Varian yang regular
    - Penanganan kasus khusus
    - Aliran yang khusus menangani terjadinya '*error*'

Contoh:

- Basic Path untuk *use case* 'Memasukkan Barang' adalah ketika pelanggan memasukkan suatu item barang dan meminta tanda terima
- Alternatif path adalah *use case* ketika barang yang dimasukkan terjepit di slot pemasukan barang



Aliran kejadian (2)

- Aliran Basic (***basic flow***) atau **normal** menjelaskan
  - Apa yang terjadi pada '**umum**'nya kalau suatu *use-case* dilakukan
  - Sering disebut sebagai '***happy path***' atau skenario '*happy path*'
- Aliran aksi dalam *use-case* bisa dibagi menjadi beberapa '***sub flow***'
  - Beberapa bagian dari aliran bisa dipecah dan pecahannya dapat diberikan penjelasan secara terpisah; untuk meningkatkan kemudahan pembacaan (*readability*), dan menjadi perbaikan dari struktur model *use-case*
  - Pemisahan ini juga membuat aliran dasar menjadi lebih jelas
  - Jika suatu *subflow* melibatkan hanya sebagian kecil dari *event flow* lengkapnya, mungkin cukup dijelaskan dalam teks saja.
  - *Sub flow* kadang sebaiknya dipisahkan sebagai suplemen terpisah

Aliran kejadian (3)

- *Use-case* harus mencakup **semua flow**, baik yang **normal**, **alternatif** maupun *flow* pengecualian (***exceptional***).
  - *use-case* harus digambarkan sedemikian rupa sehingga mudah untuk melihat *flow* dan mudah dimengerti apa yang terjadi pada satu saat
- Contoh *subflow*
  - Bila menempati bagian yang cukup besar dalam aliran aksi
  - Jika merupakan *flow* varian, atau *exception*
  - Bisa dieksekusi pada beberapa interval yang dalam *flow* yang sama
  - *Sub flow* mungkin dilakukan pada waktu yang sama, dan mungkin bersifat *optional*

Deskripsi Use Case (2)

- Untuk jumlah *use-case* yang **besar** dengan berbagai **alternatif**, maka penulisan teks menjadi tidak praktis, jadi kadang digunakan **diagram**
  - **Diagram *Statechart***
    - Untuk menggambarkan *use-case* yang kompleks
    - Berisi penjelasan *state* dan transisi dalam *use-case*
  - **Diagram Aktivitas**
    - Menggambarkan transisi antar *state* dalam bentuk urutan aksi
    - Bentuk yang lebih umum dari *State Transition Diagram*
  - **Diagram Interaksi**
    - Menjelaskan interaksi antar instansiasi dari aktor dan instansiasi dari *use-case*

Deskripsi Use Case - contoh

Use Case: Mengajukan Usulan Kuliah
Iteration : ke-2,
Modifikasi terakhir 1 Maret 2018
Primary Actor: Mahasiswa
Goal in Context: Untuk pengajuan usulan kuliah oleh mahasiswa
Preconditions: Mahasiswa sudah terdaftar dan mahasiswa sudah memasukkan nama user dan password sebelumnya
Trigger: Jika Mahasiswa memutuskan untuk mengambil kuliah di awal semester

Scenario:

1. Mahasiswa memilih menu entri usulan kuliah
2. Sistem menampilkan form entri FRS
3. Mahasiswa mengisikan kode kuliah
4. Sistem menampilkan informasi detil matakuliah (nama, sks)
5. Mahasiswa menekan tombol SIMPAN
6. Sistem menyimpan data usulan ke dalam basisdata

Exception:

1. Mahasiswa memilih menu entri usulan kuliah
2. Mahasiswa memilih untuk melihat daftar kelas
3. Sistem menampilkan daftar kelas yang dibuka
4. Mahasiswa memilih matakuliah dari daftar
5. Mahasiswa menekan tombol SIMPAN
6. Sistem menyimpan data usulan ke dalam basisdata

Priority: Prioritas sedang
When available: Iterasi ketiga
Secondary Actor: tidak ada
Open Issues:

1. Bagaimana mekanisme mendeteksi, jika ada suatu kuliah memiliki prerequisite kuliah lain
2. Bagaimana jika mahasiwa ingin membatalkan usulan

Langkah pemodelan berbasis skenario

1. Identifikasi aktor
2. Identifikasi *use case*
3. Gambarkan diagram *use case*
4. Buat skenario tiap *use case*

1. Identifikasi aktor

- Siapa '**pengguna**' sistem atau yang terkait dengan sistem
  - Ada aktor yang **menggunaka**n sistem
  - Ada aktor yang bertugas melakukan perawatan (***maintenance***)
- **Peran** aktor harus **berbeda**
  - Mungkin bisa terjadi peran yang saling tumpang tindih
  - Perlu **nama** yang 'relevan' dengan makna semantik dari peran
    - **Pengguna sistem** dengan **pelanggan sistem**, harus jelas apa yang dimaksud dengan peran ini.
    - Istilah **pengguna sistem** mungkin juga adalah **sistem lain** (bukan orang).
  - Juga harus jelas apa '**kebutuhan**' dan '**tanggung jawab**' si aktor!

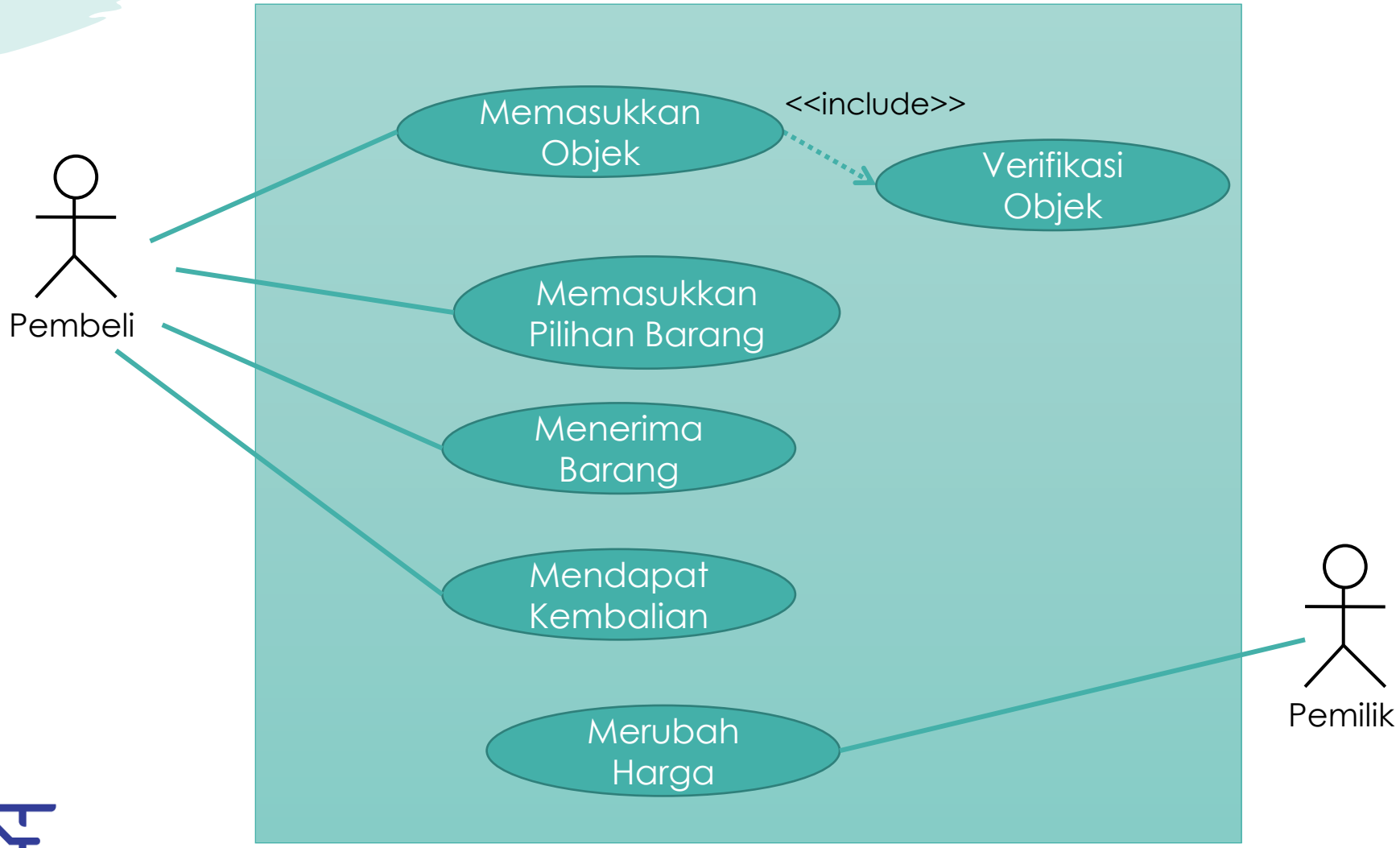
2. Identifikasi *use-case*

- Daftarkan **aktivitas** yang dilakukan oleh **aktor** untuk melakukan suatu **fungsi/kegiatan**
- Beri **nama**
  - Nama menggambarkan sekumpulan aksi
  - Biasanya dimulai dengan '**kata kerja**'
- *Use-case* biasanya harus '**lengkap**' atau dapat '**berdiri sendiri**'
- *Use-case* ini memberikan suatu '**hasil**' untuk **aktor** ini
- *Use-case* biasanya ditulis dalam bentuk **cerita** yang kemudian dipetakan ke suatu template
- Setiap skenario utama harus dikaji ulang (*review*)

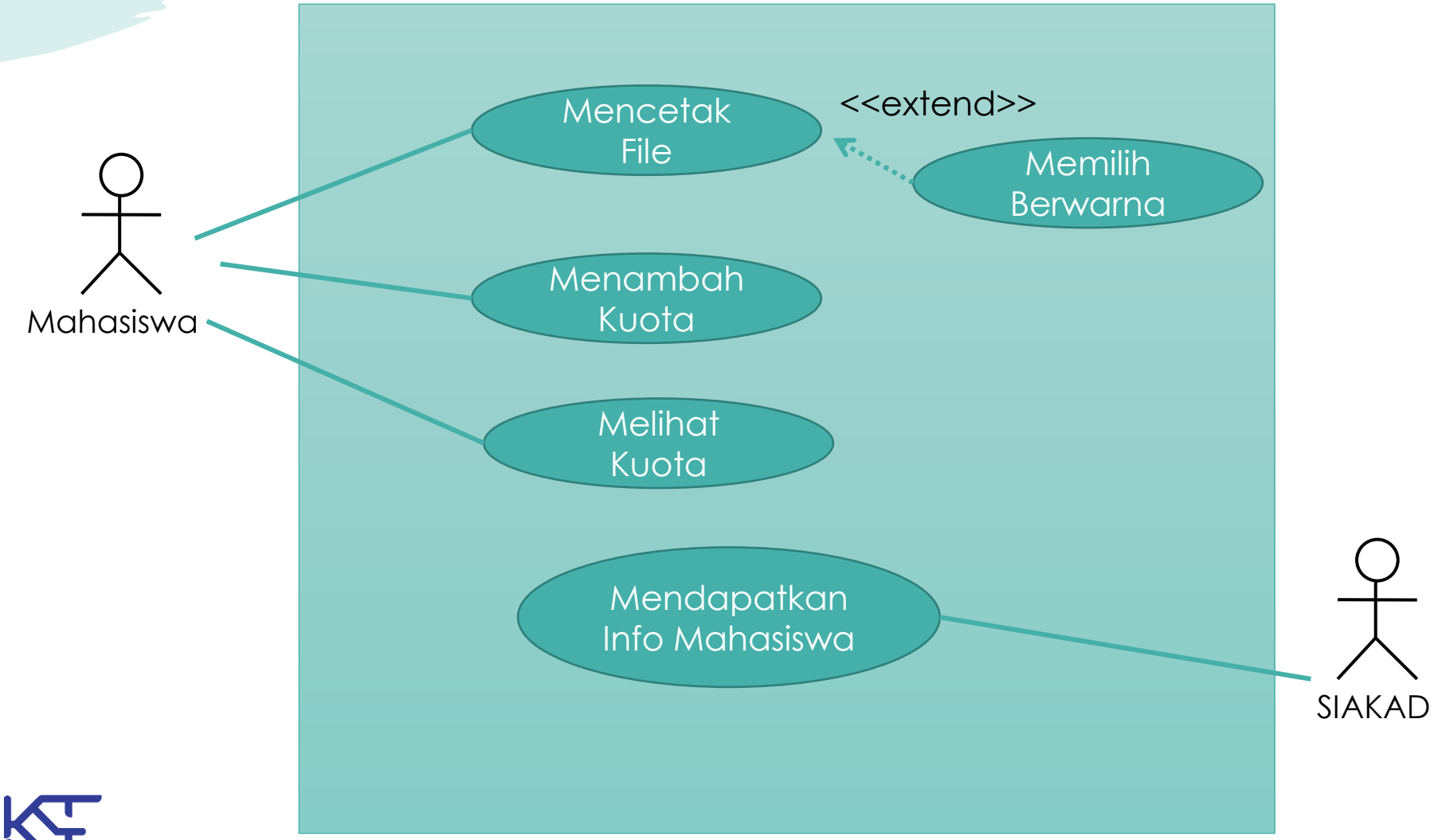
3. Gambarkan diagram use case

- Setiap ***use case*** harus **terhubung** dengan minimal **satu aktor**
- Jika *use case* terhubung dengan **lebih** dari satu aktor, maka harus diperjelas aktor yang men-***trigger*** *use case* pertama kali (gunakan **tanda panah**)
- Mungkin ada hubungan antar *use case*: ***include*** (*uses*) atau ***extends*** (*insert*)

Use Case untuk Vending Machine



Use Case Layanan Pencetakan Mahasiswa

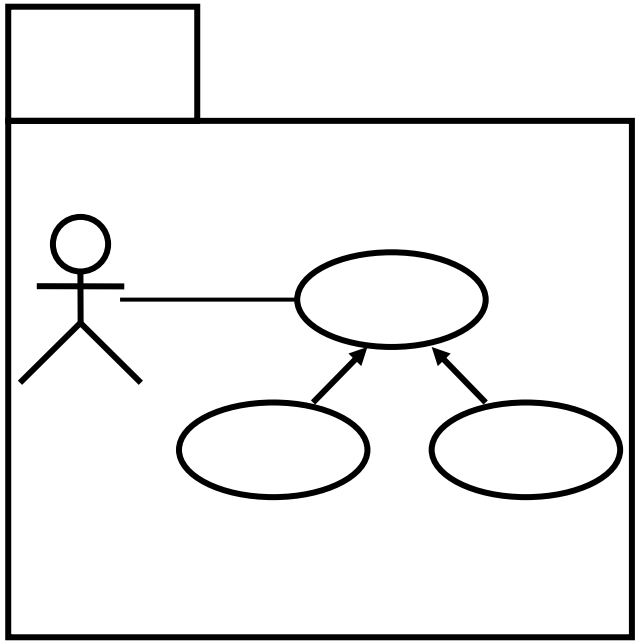


4. Membuat skenario

- Skenario digunakan untuk **menjelaskan pemakaian sistem**
- Setiap skenario **dilihat dari sudut pandang 'aktor'**
- Setiap skenario **menjawab pertanyaan** berikut:
  - Siapa aktor utama, siapa aktor pendukung
  - Apa tujuan si aktor
  - Kondisi awal apa yang harus ada sebelum suatu cerita atau '*story*' dimulai
  - Apa tugas/fungsi utama yang dilakukan oleh si aktor
  - Apa tugas/fungsi tambahan yang dapat diberikan
  - Apa variasi yang memungkinkan dalam interaksi si aktor
  - Informasi dari sistem apa yang dibutuhkan, diproduksi atau diubah dari/oleh si aktor
  - Apakah si aktor harus memberitahukan sistem bila terjadi perubahan pada lingkungan eksternal?
  - Informasi apa yang diinginkan oleh si aktor dari sistem
  - Apakah si aktor ingin diberitahu bila ada perubahan yang di luar rencana?

Paket (Packages) dalam model Use-case

- **Paket** digunakan untuk **mengelompokkan** elemen-elemen yang terkait secara **semantik**
- Kegunaan:
  - *Use-case* lebih terstruktur
  - **Batasan** lingkup dari satu atau beberapa *use-case*
  - Paket dalam *use-case* juga bisa digunakan untuk
    - Menunjukkan urutan sistem
    - Konfigurasi sistem
    - *Delivery unit*
  - Memudahkan pembagian pekerjaan dalam tim



Contoh: Mesin Recycle

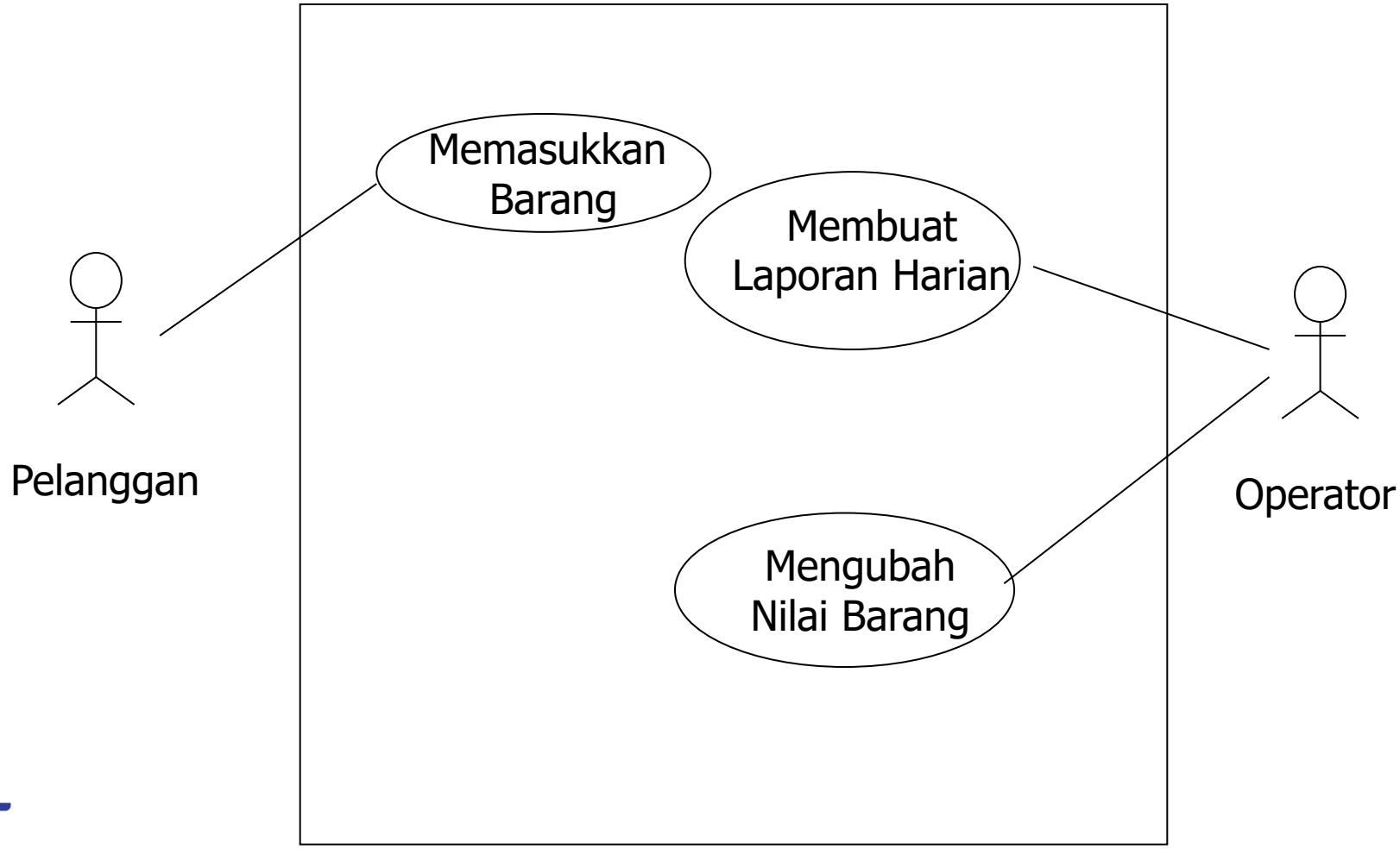
- Suatu mesin melakukan *recycle* terhadap botol dan kaleng
- Mesin dapat digunakan oleh pengguna yang berbeda-beda
- Sistem akan mencatat tipe dan jumlah item yang dimasukkan
- Sistem akan mencetak tanda terima agar dapat diganti dengan uang oleh pengguna
- Operator dapat meminta laporan harian
- Operator dapat merubah nilai dari item (benda) yang dikembalikan
- (Operator akan diinformasikan bila ditemukan *malfunction*)

Contoh: Mesin Recycle

Mesin Recycle

- Aktor?
  - Pelanggan
  - Operator
- Use case?
  - Pelanggan dapat memasukkan barang
  - Operator dapat mengubah nilai barang
  - Operator dapat membuat laporan harian
- Gambarkan diagram *use case* (slide berikutnya)
- Detilkan skenarionya

Contoh: Mesin Recycle



Skenario Use Case Memasukkan Barang

- Kondisi Awal (*Pre-condition/Initial State*):
  Pelanggan mau meletakkan botol atau kaleng ke mesin
- Aksi:
  - Pelanggan meletakkan setiap barang ke mesin
  - Sistem akan menaikkan jumlah barang yang diterima, juga mencatat jumlah total barang secara keseluruhan
  - Pelanggan menekan tombol selesai untuk mendapatkan tanda terima
  - Sistem akan mencetak tanda terima berisi jumlah barang yang dimasukkan.
- Aksi alternatif
  - Barang mungkin terjepit di slot mesin
  - Pelanggan harus diberi informasi, operator diberi notifikasi, dan mesin mencetak tanda terima barang yang berhasil dimasukkan
- Kondisi Akhir (*Post-condition/final state*):
  Use-case selesai setelah tombol tanda terima (tombol selesai) ditekan

Skenario Use Case Membuat Laporan harian

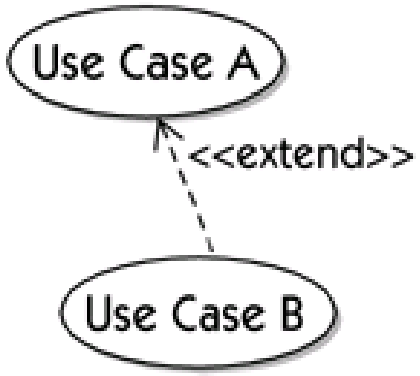
- Kondisi-awal:
  Operator mau mencetak laporan harian untuk semua barang yang dimasukkan
- Aksi:
  - Sistem akan mencetak jumlah barang yang diterima untuk setiap tipe
  - Sistem akan mencetak jumlah barang
  - Sistem akan di-*reset* kembali nol
- Kondisi-akhir:
  Jumlah harian di-set nol setelah *use-case* selesai

Skenario Use Case Mengubah Nilai Barang

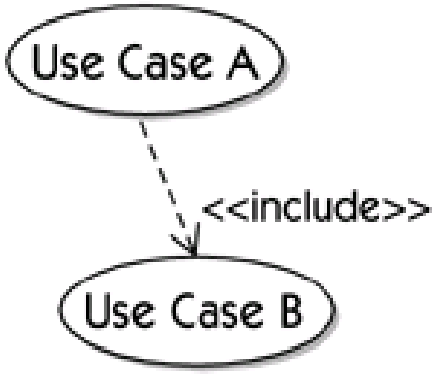
- Kondisi-awal:
  Operator mau mengubah nilai keterangan barang
- Aksi:
  - Nilai jumlah setiap barang yang akan diubah
  - Ukuran barang yang akan dirubah
  - Tipe barang baru dapat ditambahkan
- Kondisi-akhir:
  Nilai keterangan barang baru berubah

Ketergantungan antar use-case

- Hubungan ***extend*** menunjukkan kemungkinan adanya perilaku (*behaviour*) **tambahan** (optional)
- Hubungan ***include*** mendefinisikan hubungan **langsung** dua *use-case (wajib dilibatkan)*



use-case B **mungkin** dilakukan bersamaan dengan *use-case* A



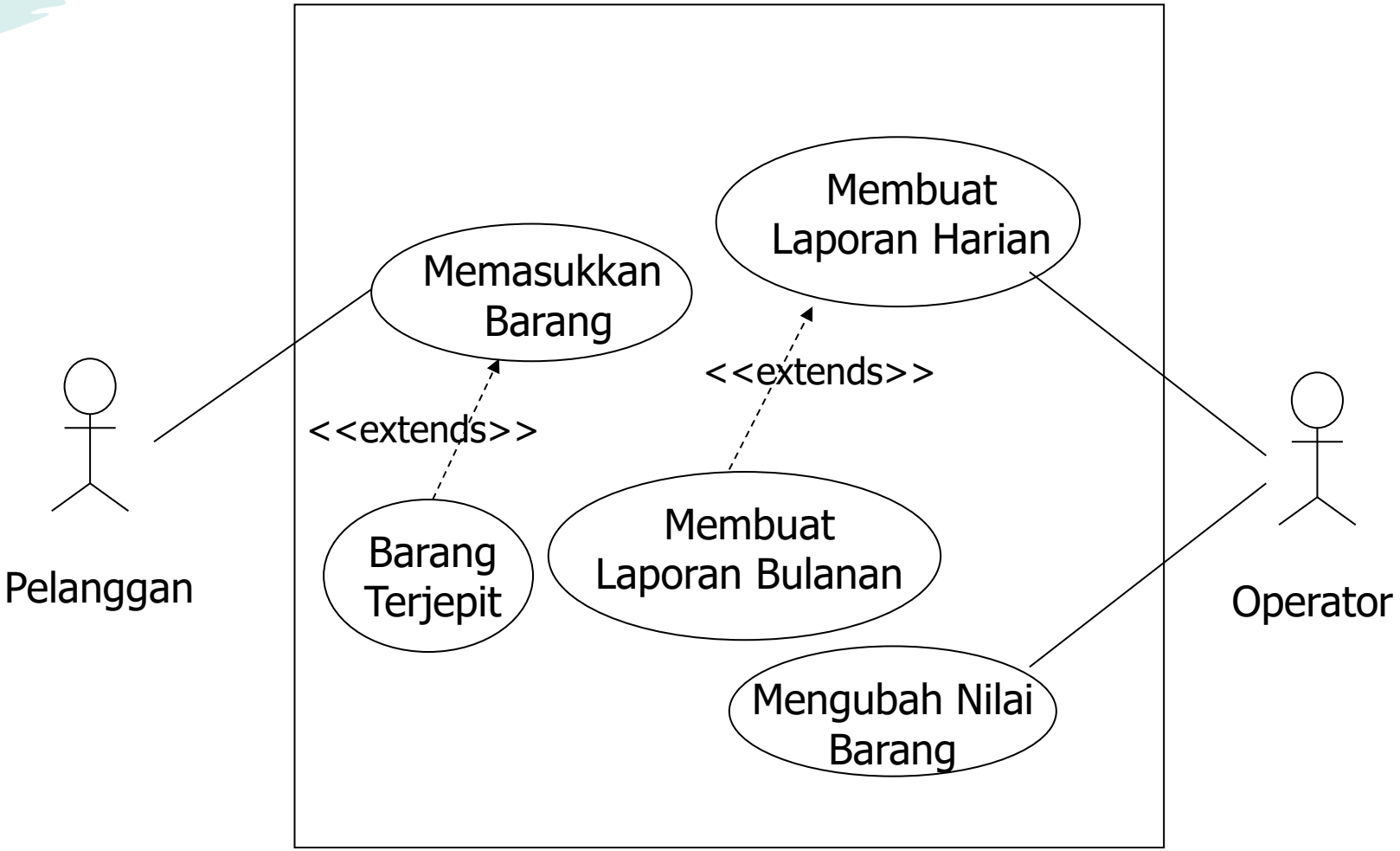
use-case B **dilibatkan** ketika *use-case* A dilakukan

KNOWLEDGE & SOFTWARE ENGINEERING

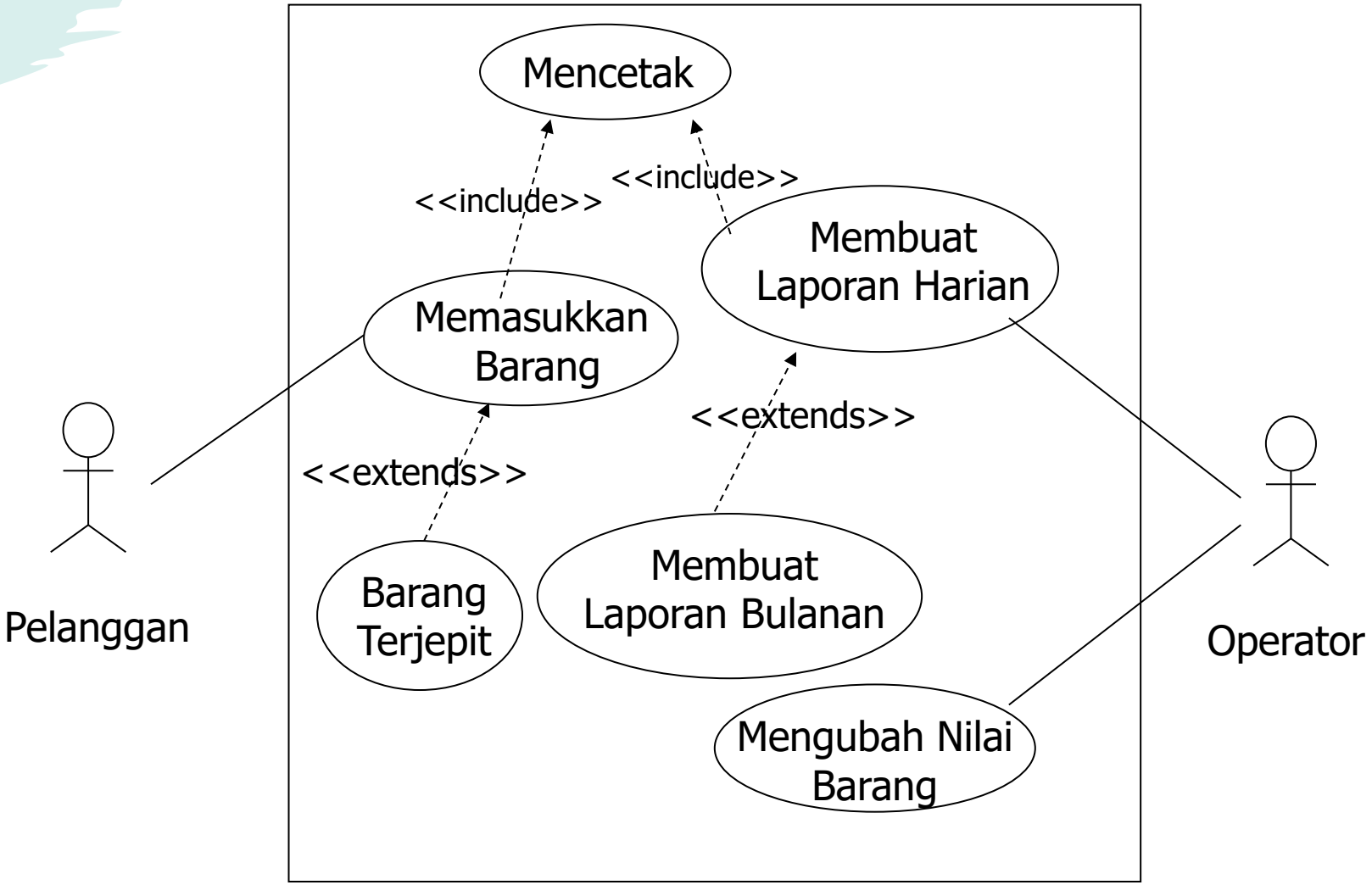
Extend/Include

- Dalam model *use-case*, dimungkin adanya pilihan *event* alternatif ataupun ada *use-case* yang harus dilakukan jika suatu *use-case* dieksekusi
  - **Alternatif** *use-case* ditunjukkan sebagai **<<extends>>**
    - Bagian yang **optional**
  - *Use-case* yang **harus** dilakukan ditunjukkan sebagai **<<include>>**
    - **Harus dilakukan**
- Walau jarang terjadi, tetapi *use-case extends* dan *include* bisa terjadi.
- Kenapa ada *extends/include*?
  - Karena harus ditambahkan fitur/ fungsionalitas baru
  - Untuk mendapatkan aksi tambahan dalam suatu kondisi lain

Contoh penggunaan <<extends>>

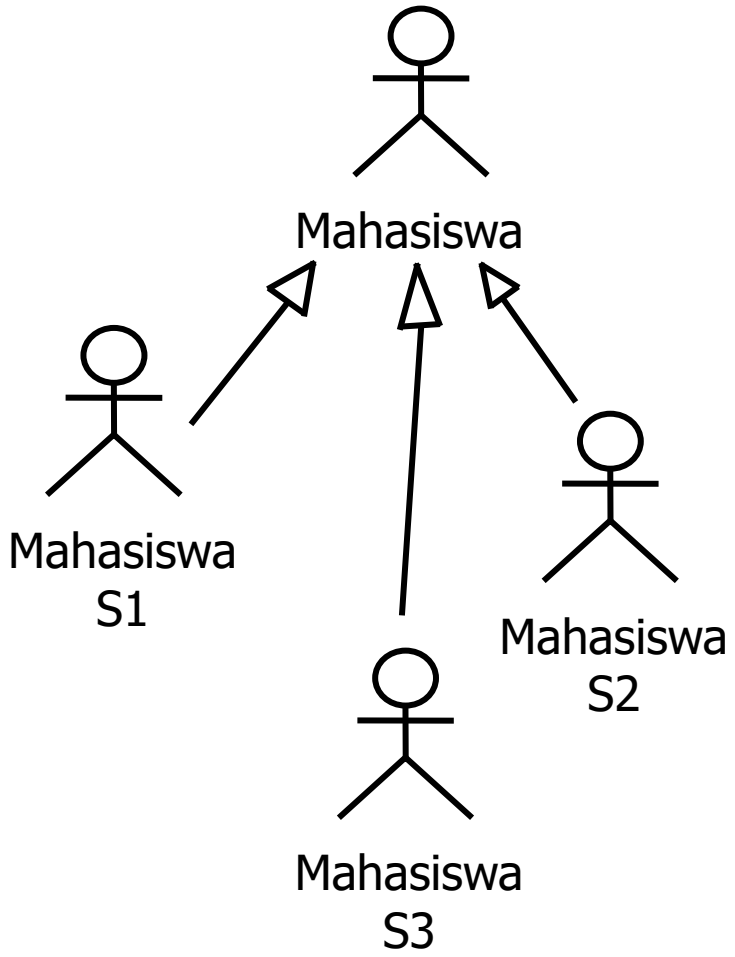


Contoh penggunaan <<include>>

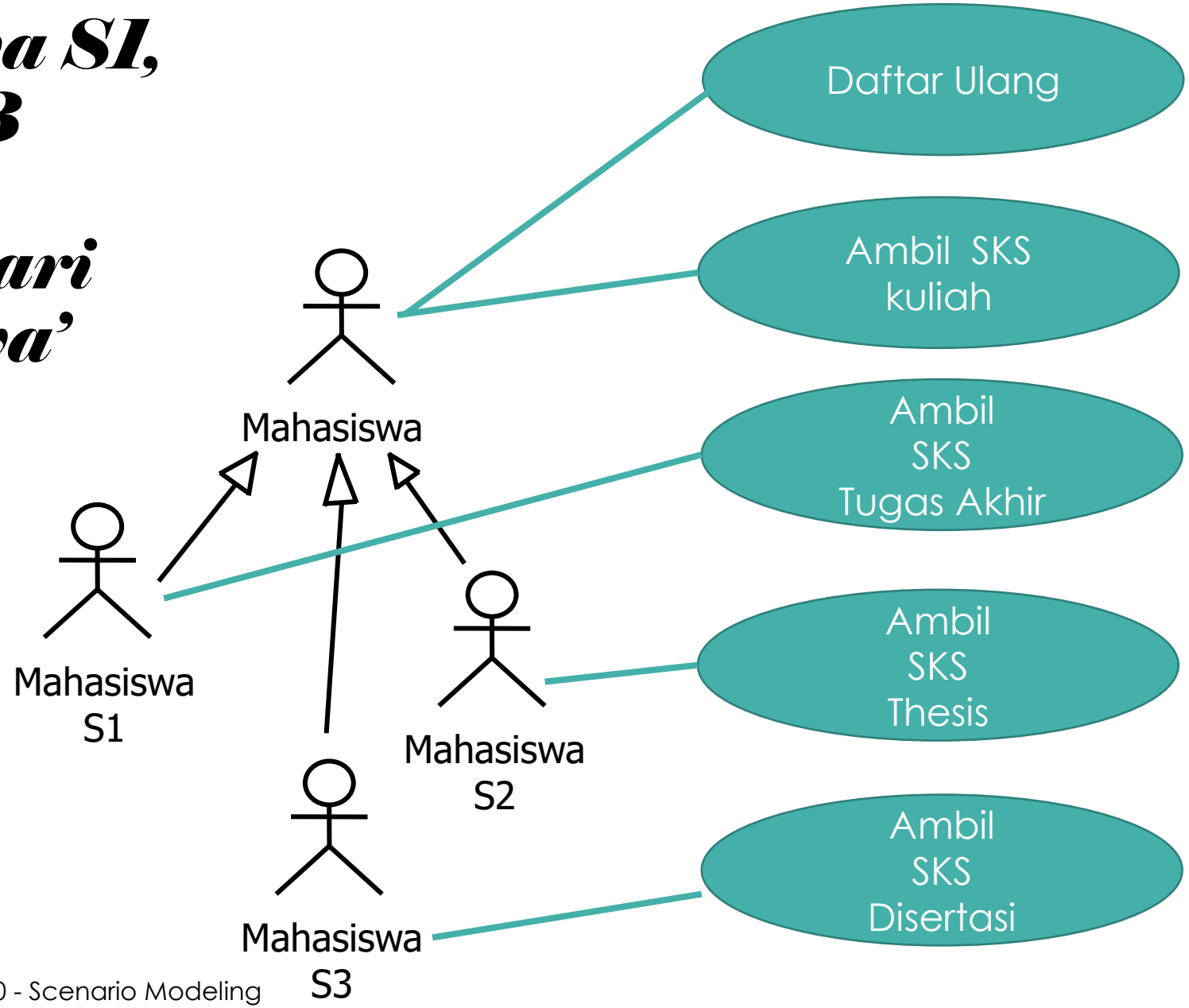


Generalisasi Aktor (Actor Generalization/Actor Inheritance)

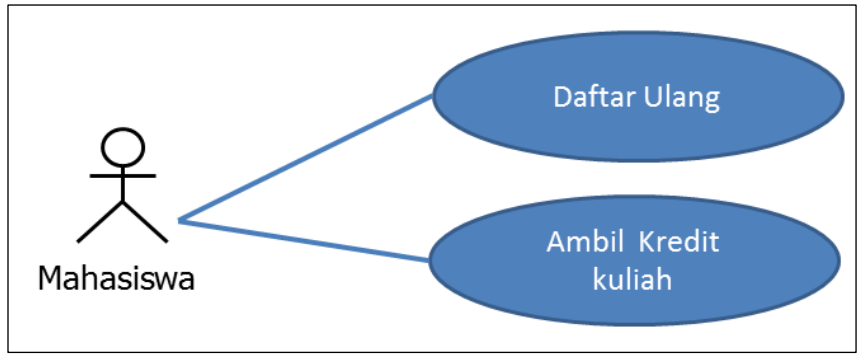
- Beberapa aktor dapat memiliki **peran** yang **sama** pada suatu *use-case*
- Contoh:
  - Ada mahasiswa S1, S2 dan S3 yang ketiganya terdaftar suatu kuliah
  - Ketiga akan terlihat sebagai entitas eksternal oleh *use-case* 'Daftar Ulang' atau 'Ambil mata kuliah'
  - Mahasiswa S1, S2 atau S3 dapat dimodelkan sebagai 'Mahasiswa' saja, karena ketiganya memiliki banyak kesamaan
- Hal ini disebut '**generalisasi aktor**'

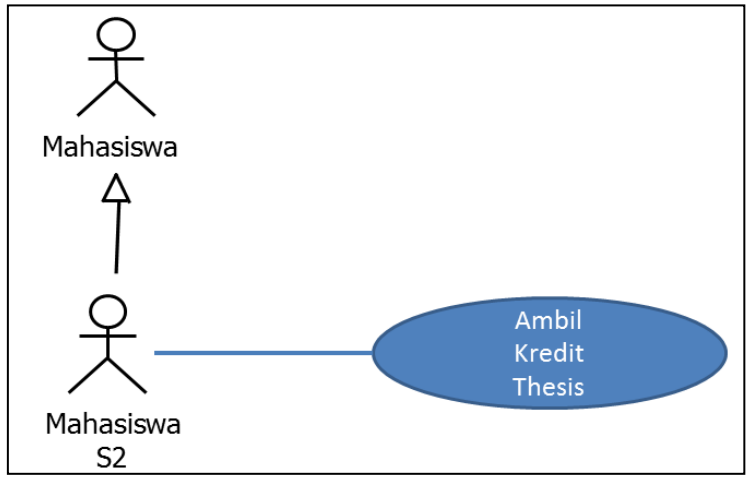


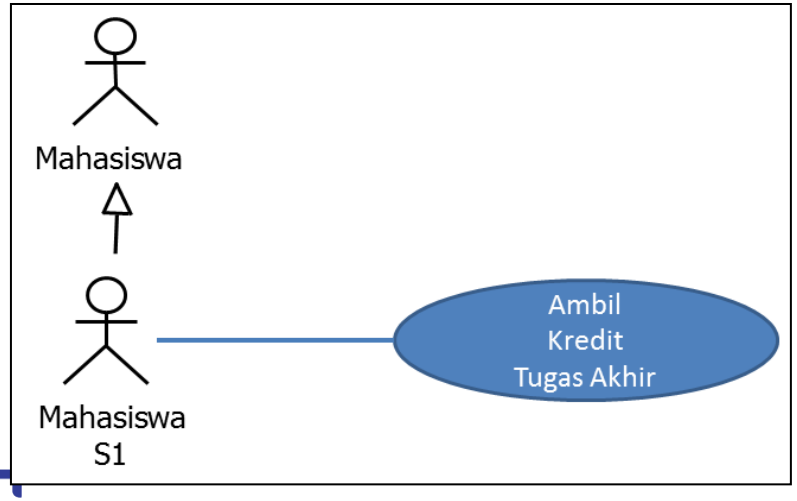
Mahasiswa S1, S2, dan S3 adalah turunan dari 'Mahasiswa'

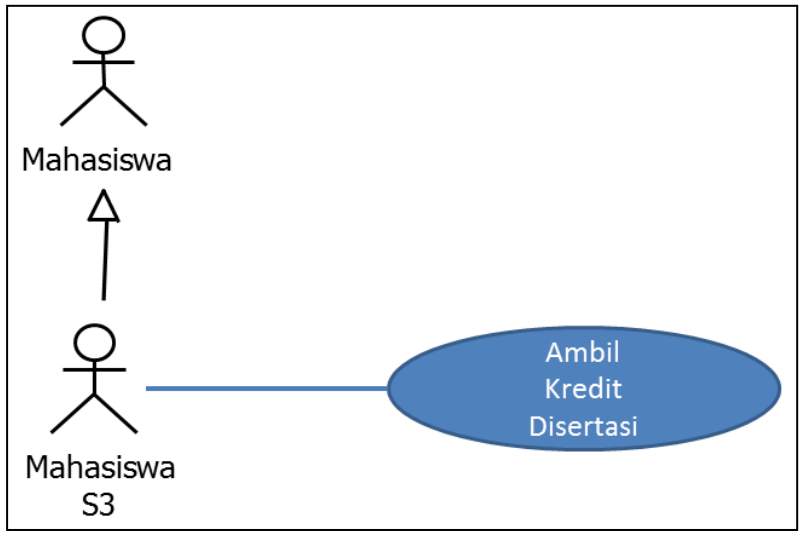


Generalisasi Aktor dapat dipisahkan penggambarannya





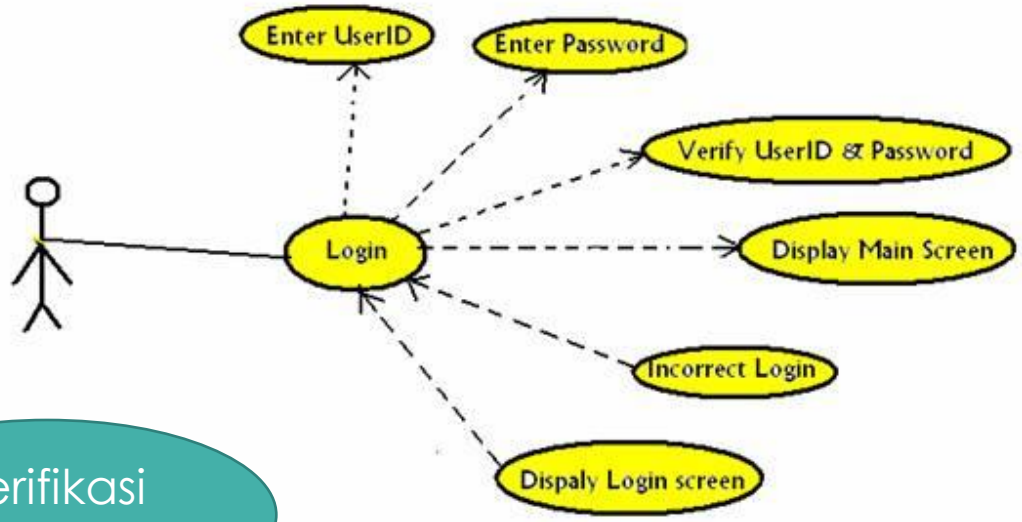


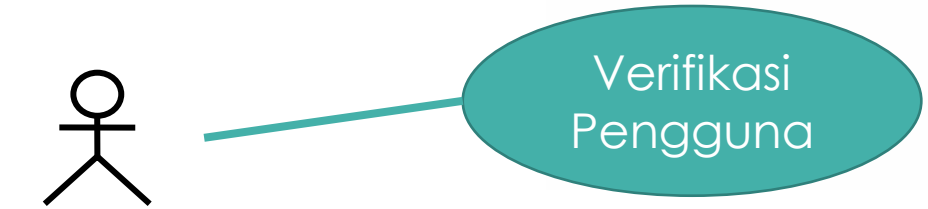


Pemakaian Use-case

- Dalam pengembangan perangkat lunak, *Use-case* akan berperan:
  - Membantu proses iterasi untuk mengerti masalah
  - Membantu mencari prioritas pengembangan

Apa yang salah?





Mahasiswa

Skenario:

Display Login Screen
Enter User Id
Enter Password
Verify User Id dan Password
If LoginError then Incorrect Login
Else display Main Screen

Latihan membuat diagram use-case

Akan dibangun sebuah perangkat lunak untuk mendukung proses pendaftaran ulang mahasiswa secara online. Melalui aplikasi tersebut, mahasiswa dapat mengajukan usulan pengambilan matakuliah.

Selanjutnya, dosen wali dapat melihat usulan pengambilan matakuliah untuk disetujui/ditolak. Usulan yang ditolak dapat direvisi kembali oleh mahasiswa.

Usulan yang telah disetujui wali dapat langsung diproses oleh Petugas Administrasi untuk pencetakan KSM. KSM hanya bisa dicetak apabila status pembayaran SPP mahasiswa sudah beres. Informasi status pembayaran SPP diperoleh dari perangkat lunak lain yaitu SISKEU (Sistem Informasi Keuangan). Perangkat lunak ini juga akan berhubungan dengan perangkat lunak SIKAD (Sistem Informasi Akademik) untuk mendapatkan informasi tentang matakuliah yang ditawarkan pada semester tersebut, serta informasi transkrip nilai mahasiswa, agar dosen wali mendapatkan referensi untuk menyetujui/menolak usulan pengambilan matakuliah.

Cari Aktor!

Akan dibangun sebuah perangkat lunak untuk mendukung proses pendaftaran ulang mahasiswa secara online. Melalui aplikasi tersebut, mahasiswa dapat mengajukan usulan pengambilan matakuliah.

Selanjutnya, dosen wali dapat melihat usulan pengambilan matakuliah untuk disetujui/ditolak. Usulan yang ditolak dapat direvisi kembali oleh mahasiswa.

Usulan yang telah disetujui wali dapat langsung diproses oleh Petugas Administrasi untuk pencetakan KSM. KSM hanya bisa dicetak apabila status pembayaran SPP mahasiswa sudah beres. Informasi status pembayaran SPP diperoleh dari perangkat lunak lain yaitu SISKEU (Sistem Informasi Keuangan). Perangkat lunak ini juga akan berhubungan dengan perangkat lunak SIKAD (Sistem Informasi Akademik) untuk mendapatkan informasi tentang matakuliah yang ditawarkan pada semester tersebut, serta informasi transkrip nilai mahasiswa, agar dosen wali mendapatkan referensi untuk menyetujui/menolak usulan pengambilan matakuliah.

Aktor

Akan dibangun sebuah perangkat lunak untuk mendukung proses pendaftaran ulang **mahasiswa** secara online. Melalui aplikasi tersebut, mahasiswa dapat mengajukan usulan pengambilan matakuliah.

Selanjutnya, **dosen wali** dapat melihat usulan pengambilan matakuliah untuk disetujui/ditolak. Usulan yang ditolak dapat direvisi kembali oleh mahasiswa.

Usulan yang telah disetujui wali dapat langsung diproses oleh **Petugas Administrasi** untuk pencetakan KSM. KSM hanya bisa dicetak apabila status pembayaran SPP mahasiswa sudah beres. Informasi status pembayaran SPP diperoleh dari perangkat lunak lain yaitu **SISKEU** (Sistem Informasi Keuangan). Perangkat lunak ini juga akan berhubungan dengan perangkat lunak **SIKAD** (Sistem Informasi Akademik) untuk mendapatkan informasi tentang matakuliah yang ditawarkan pada semester tersebut, serta informasi transkrip nilai mahasiswa, agar dosen wali mendapatkan referensi untuk menyetujui/menolak usulan pengambilan matakuliah.

Aktor

- Mahasiswa
- Dosen Wali
- Petugas Administrasi
- SISKEU
- SIKAD

Cari Use-case!

Akan dibangun sebuah perangkat lunak untuk mendukung proses pendaftaran ulang mahasiswa secara online. Melalui aplikasi tersebut, mahasiswa dapat mengajukan usulan pengambilan matakuliah.

Selanjutnya, dosen wali dapat melihat usulan pengambilan matakuliah untuk disetujui/ditolak. Usulan yang ditolak dapat direvisi kembali oleh mahasiswa.

Usulan yang telah disetujui wali dapat langsung diproses oleh Petugas Administrasi untuk pencetakan KSM. KSM hanya bisa dicetak apabila status pembayaran SPP mahasiswa sudah beres. Informasi status pembayaran SPP diperoleh dari perangkat lunak lain yaitu SISKEU (Sistem Informasi Keuangan). Perangkat lunak ini juga akan berhubungan dengan perangkat lunak SIKAD (Sistem Informasi Akademik) untuk mendapatkan informasi tentang matakuliah yang ditawarkan pada semester tersebut, serta informasi transkrip nilai mahasiswa, agar dosen wali mendapatkan referensi untuk menyetujui/menolak usulan pengambilan matakuliah.

Use-case

Akan dibangun sebuah perangkat lunak untuk mendukung proses pendaftaran ulang mahasiswa secara online. Melalui aplikasi tersebut, mahasiswa dapat **mengajukan usulan** pengambilan matakuliah.

Selanjutnya, dosen wali dapat **melihat usulan** pengambilan matakuliah untuk **disetujui/ditolak**. Usulan yang ditolak dapat **direvisi** kembali oleh mahasiswa.

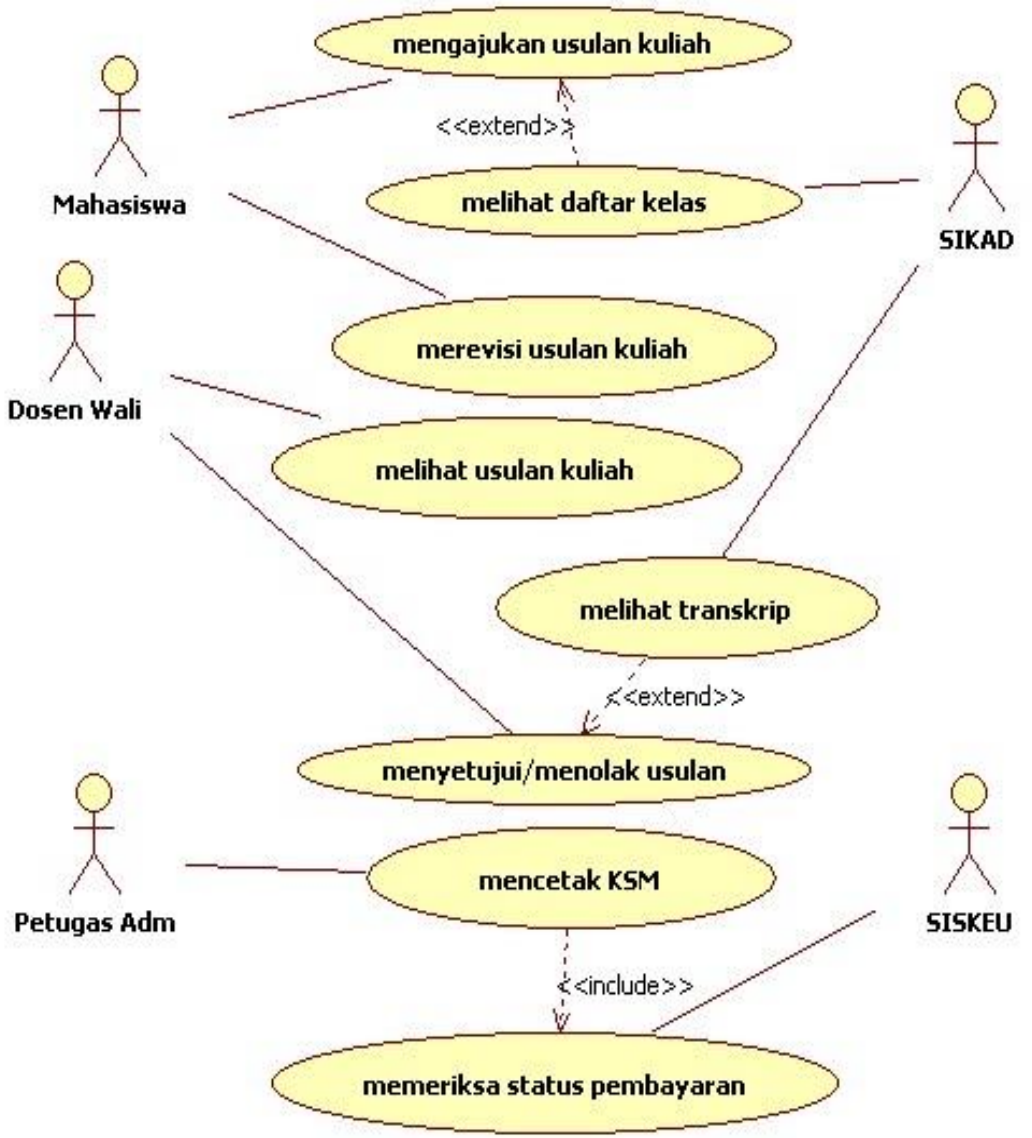
Usulan yang telah disetujui wali dapat langsung diproses oleh Petugas Administrasi untuk **pencetakan** KSM. KSM hanya bisa dicetak apabila **status pembayaran** SPP mahasiswa sudah beres. Informasi status pembayaran SPP diperoleh dari perangkat lunak lain yaitu SISKEU (Sistem Informasi Keuangan). Perangkat lunak ini juga akan berhubungan dengan perangkat lunak SIKAD (Sistem Informasi Akademik) untuk **mendapatkan informasi tentang matakuliah** yang ditawarkan pada semester tersebut, serta **informasi transkrip** nilai mahasiswa, agar dosen wali mendapatkan referensi untuk menyetujui/menolak usulan pengambilan matakuliah.

Use-case

- Mengajukan Usulan
- Melihat Usulan
- Menyetujui Usulan
- Menolak Usulan
- Merevisi Usulan
- Memeriksa Status Pembayaran
- Melihat Daftar Kelas
- Melihat Transkrip
- Mencetak KSM

KNOWLEDGE & SOFTWARE ENGINEERING

Use-case Diagram



Acknowledgement

- Pengembangan dari slide: "Scenario Based Modeling" IF2036 Sem II - 2012/2013
- Pengembangan dari slide: "Pemodelan Berbasis Skenario" oleh Bayu Hendradjaya, IF2250 Sem II - 2015/2016