USE CASE DIAGRAM

Rekayasa Perangkat Lunak

D3 Teknik Informatika

Universitas Sebelas Maret

MATERI

- Pendahuluan
- Manfaat Use Case Diagram
- Include dan Extend
- Komponen Use Case Diagram
- MenemukanAktor dan Use Case
- Do and Dont's
- Contoh Use Case Diagram
- Chapter Exercise

MENDEFINISIKAN KEBUTUHAN

- Dalam membuat sebuah sistem, langkah awal yang perlu dilakukan adalah menentukan kebutuhan
- Terdapat dua jenis kebutuhan :

1. Kebutuhan fungsional adalah kebutuhan pengguna dan *stakeholder* sehari-hari yang akan dimiliki oleh sistem, dimana kebutuhan ini akan digunakan oleh pengguna dan *stakeholder*.
2. Kebutuhan nonfungsional adalah kebutuhan yang memperhatikan hal-hal berikut yaitu performansi, kemudahan dalam menggunakan sistem, kehandalan sistem, keamanan sistem, keuangan, legalitas, dan operasional.

- Kebutuhan fungsional akan digambarkan melalui sebuah diagram yang dinamakan diagram use case.

USE CASE DIAGRAM

- Diagram Use Case menggambarkan apa saja aktifitas yang dilakukan oleh suatu sistem dari sudut pandang pengamatan luar. Yang menjadi persoalan itu ***apa*** *yang dilakukan* bukan ***bagaimana*** *melakukannya*.
- Diagram Use Case dekat kaitannya dengan kejadian-kejadian. Kejadian (scenario) merupakan contoh apa yang terjadi ketika seseorang berinteraksi dengan sistem.
- Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng- *create* sebuah daftar belanja, dan sebagainya.
- Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.

MANFAAT USE CASE DIAGRAM

- Singkatnya, diagram use case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut
- Diagram Use Case berguna dalam tiga hal :
  - **Menjelaskan fasilitas yang ada (*requirements*).** Use Case baru selalu menghasilkan fasilitas baru ketika sistem dianalisa, dan *design* menjadi lebih jelas.
  - **Komunikasi dengan klien.** Penggunaan notasi dan simbol dalam diagram Use Case membuat pengembang lebih mudah berkomunikasi dengan klien- nya.
  - **Membuat test dari kasus-kasus secara umum.** Kumpulan dari kejadian- kejadian untuk Use Case bisa dilakukan test kasus layak untuk kejadian- kejadian tersebut.

- Untuk lebih memperjelas lihat gambaran suatu peristiwa untuk sebuah klinik kesehatan di bawah ini :

"Pasien menghubungi klinik untuk membuat janji (appointment) dalam pemeriksaan tahunan. Receptionist mendapatkan waktu yang luang pada buku jadwal dan memasukkan janji tersebut ke dalam waktu luang itu."

Gambar 1. Contoh kegiatan pasien yang membuat janji

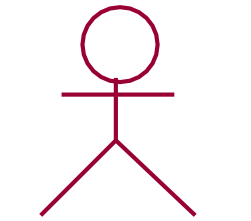
INCLUDE DAN EXTEND

- Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal.
- Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*.
- Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri.
- Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

KOMPONEN USE CASE

1. **Aktor** adalah seseorang atau apa saja yang berhubungan dengan sistem yang sedang dibangun.

- Aktor sebaiknya diberi nama dengan kata benda.
- Ketika memberi nama actor, gunakan nama **peranan** dan **jangan nama posisi.** Seorang individu dapat memainkan beberapa peranan.
- Dalam UML direpresentasikan dengan notasi berikut ini:



PASIEN

TIPE AKTOR

- Ada 3 tipe aktor :
  1. Pengguna sistem. Merupakan actor secara fisik atau seorang pengguna, gambaran secara umum dan selalu ada pada setiap sistem
  2. Sistem yang lain dan berhubungan dengan sistem yang dibangun. Misalkan pada sebuah sistem Informasi Puskesmas memerlukan koneksi dengan aplikasi sistem yang lain, semisal SIM rumah sakit. Maka dalam kasus ini, SIM rumah sakit adalah actor.
  3. Waktu. Dapat menjadi actor jika seiring perjalan waktu dapat memicu event/kejadian dalam sistem.

KOMPONEN USE CASE (2)

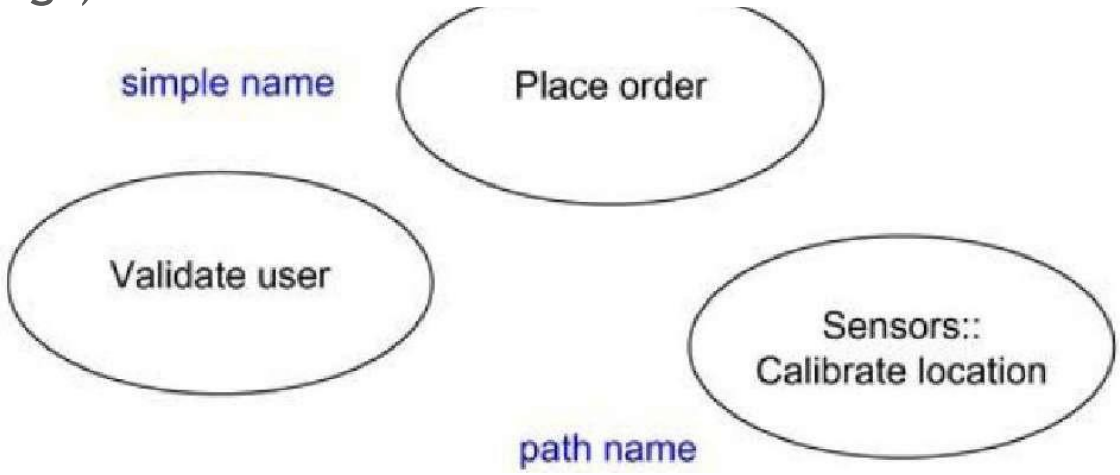
2. **Use Case.**Adalah bagian fungsionalitas tingkat tinggi yang disediakan oleh sistem.

- Dengan kata lain, use case menggambarkan bagaimana seseorang menggunakan sistem.
- Use Case dalam UML dinotasikan dengan simbol

PENDAFTARAN PASIEN

PEMBERIAN NAMA USE CASE

- Pemberian nama use case sebaiknya :
- Simple name. Biasanya berupa kata kerja + kata benda
- Path name. Merupakan nama di bagian depan yang menyatakan paket (package) dimana use case tersebut berada



KOMPONEN USE CASE (3)

3. **Relasi** yang menghubungkan antara actor dan use case

- Arah panah menunjukkan siapa yang mengawali komunikasi.
- Dengan mengecualikan use case dalam relasi include dan relasi extend, setiap use case harus diinisialisasi oleh actor
- Relasi dinotasikan seperti gambar berikut :

JENIS RELASI

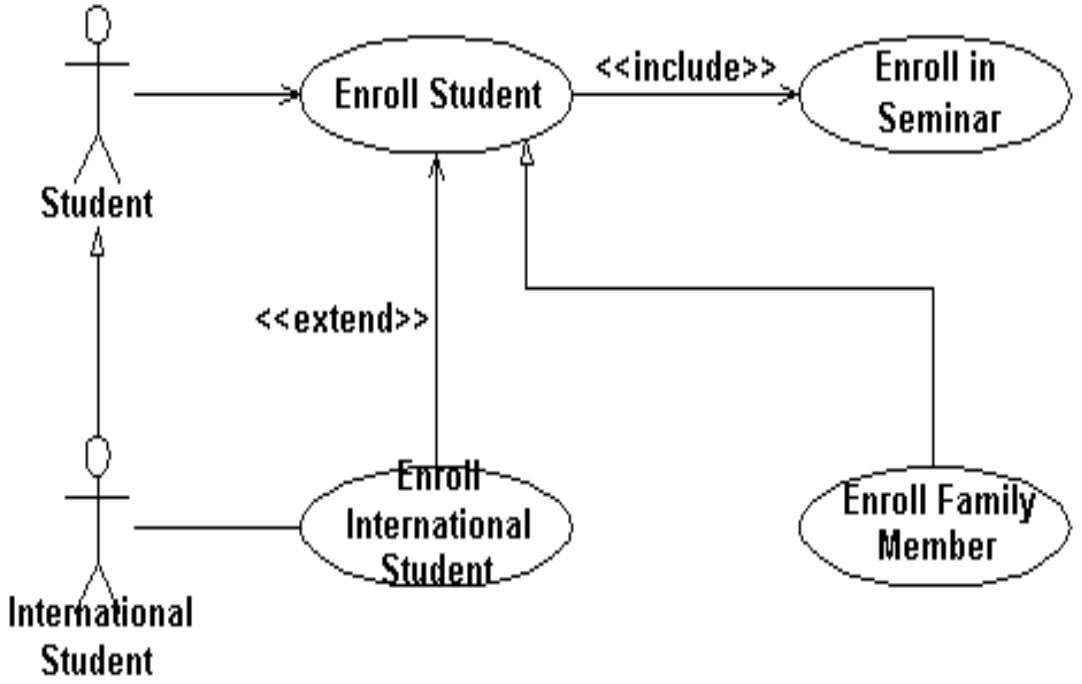
- Ada 3 jenis relasi
  - Generalization
  - Include
  - Extends

GENERALIZATION

- Hubungan antara induk dan anak
- Anak mewarisi sifat dan method dari induk
- Induk disebut root / base
- Class yang tidak memiliki anak disebut leaf
- Gambarkan generalization/inheritance antara use case secara vertical dengan inheriting use case dibawah base/parent use case
- Generalization/inheritance dipakai ketika ada sebuah keadaan yang lain sendiri/perlakuan khusus *(single condition)*
- Terbagi menjadi 2 :
  - Actor Generalization
  - Use Case Generalization

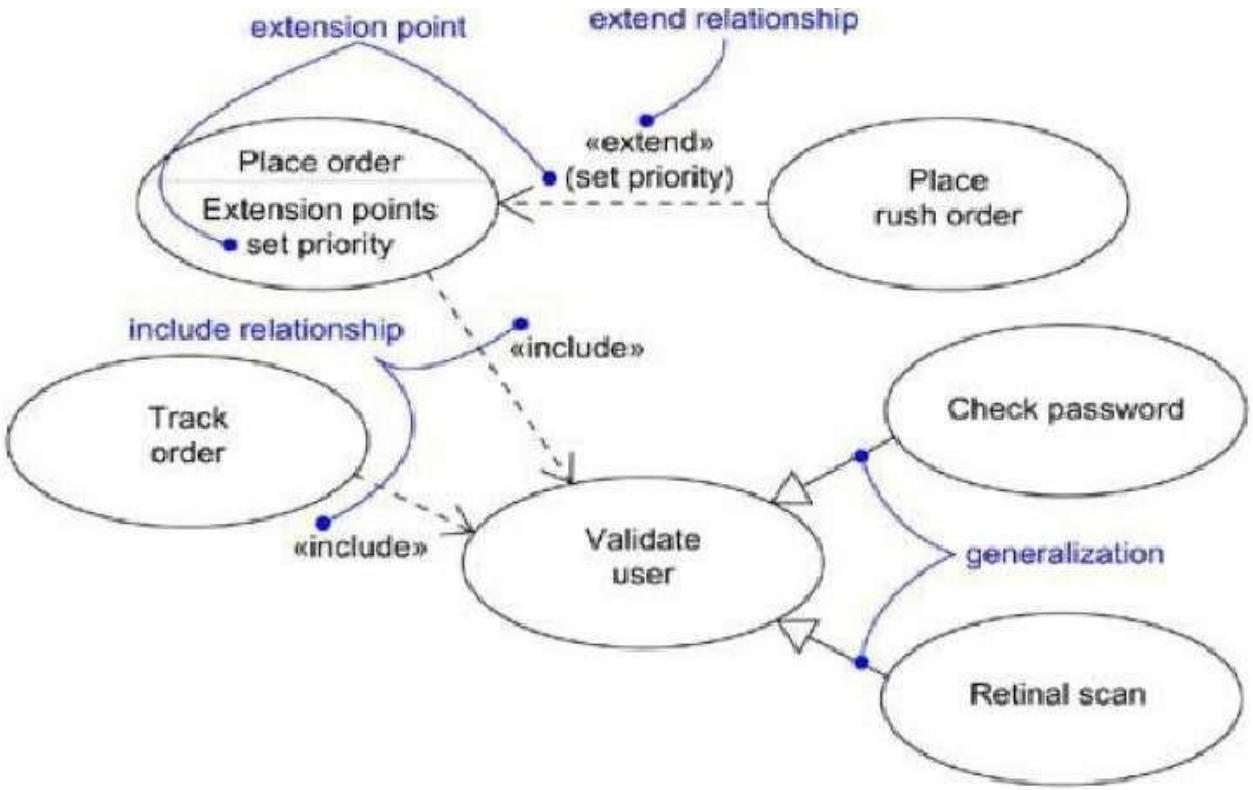
ACTOR GENERALIZATION

- Aktor bisa umum atau spesifik
- Gambarkan generalization/inheritance antara actors secara vertical dengan inheriting actor di bawah base/parent use case



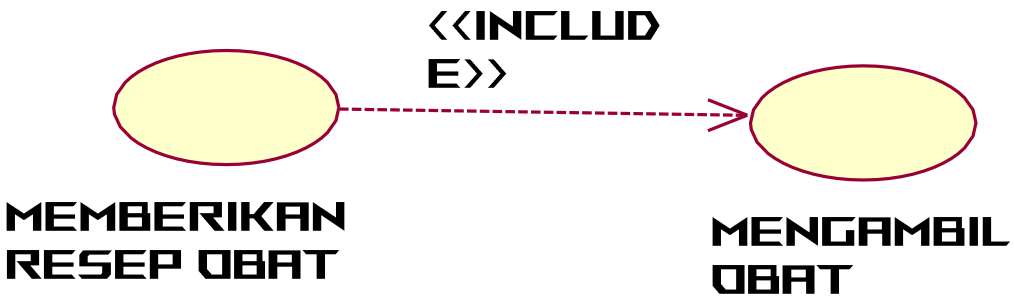
USE CASE GENERALIZATION

- Use case anak mewarisi arti dari use case induk sambil menambahkan/memodifikasi behaviour dari induk



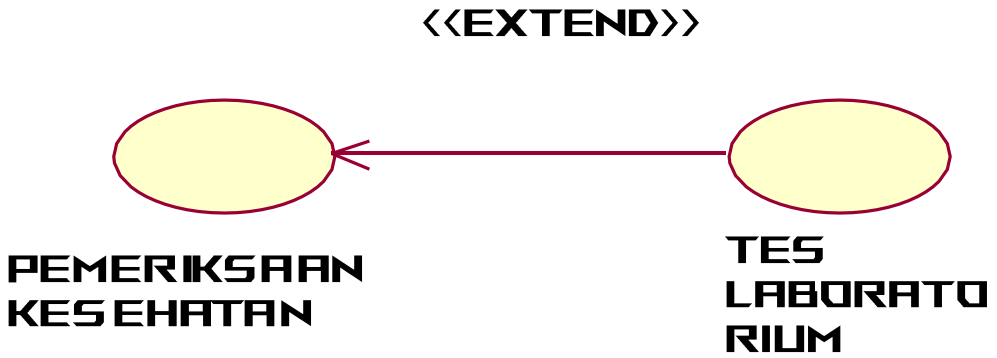
INCLUDE

- Memungkinkan (required/harus) satu use case menggunakan fungsionalitas yang disediakan oleh use case lainnya.
- Tanda panah terbuka harus terarah ke sub use case
- Gambarkan association include secara horizontal



EXTEND

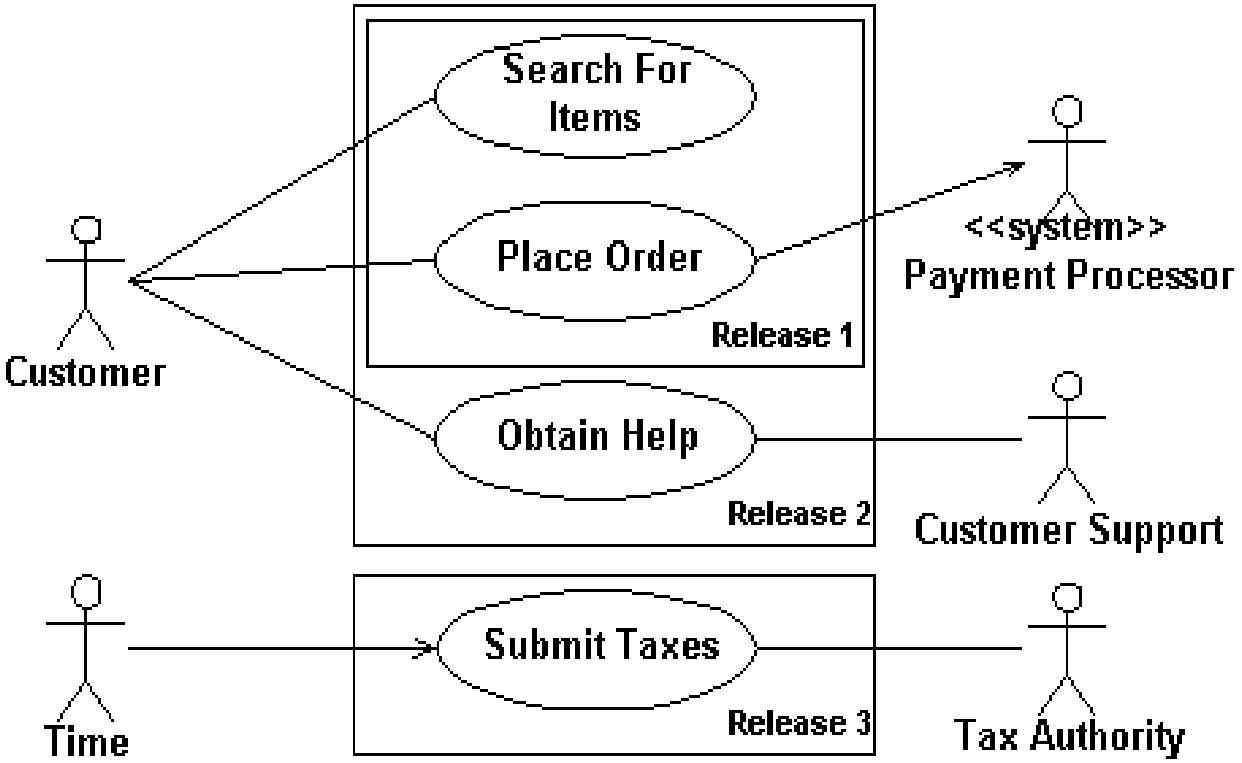
- Memungkinkan suatu use case secara **optional** menggunakan fungsionalitas yang disediakan oleh use case lainnya.
- Kurangi penggunaan association Extend ini, terlalu banyak pemakaian association membuat diagram sulit dipahami.
- Tanda panah terbuka harus terarah ke parent/base use case
- Gambarkan association extend secara vertical
- Contoh :
  - Use case pemeriksaan kesehatan suatu saat memerlukan tes laboratorium, tapi pada saat lain tidak. Tergantung pada kondisi pasien yang diperiksa.



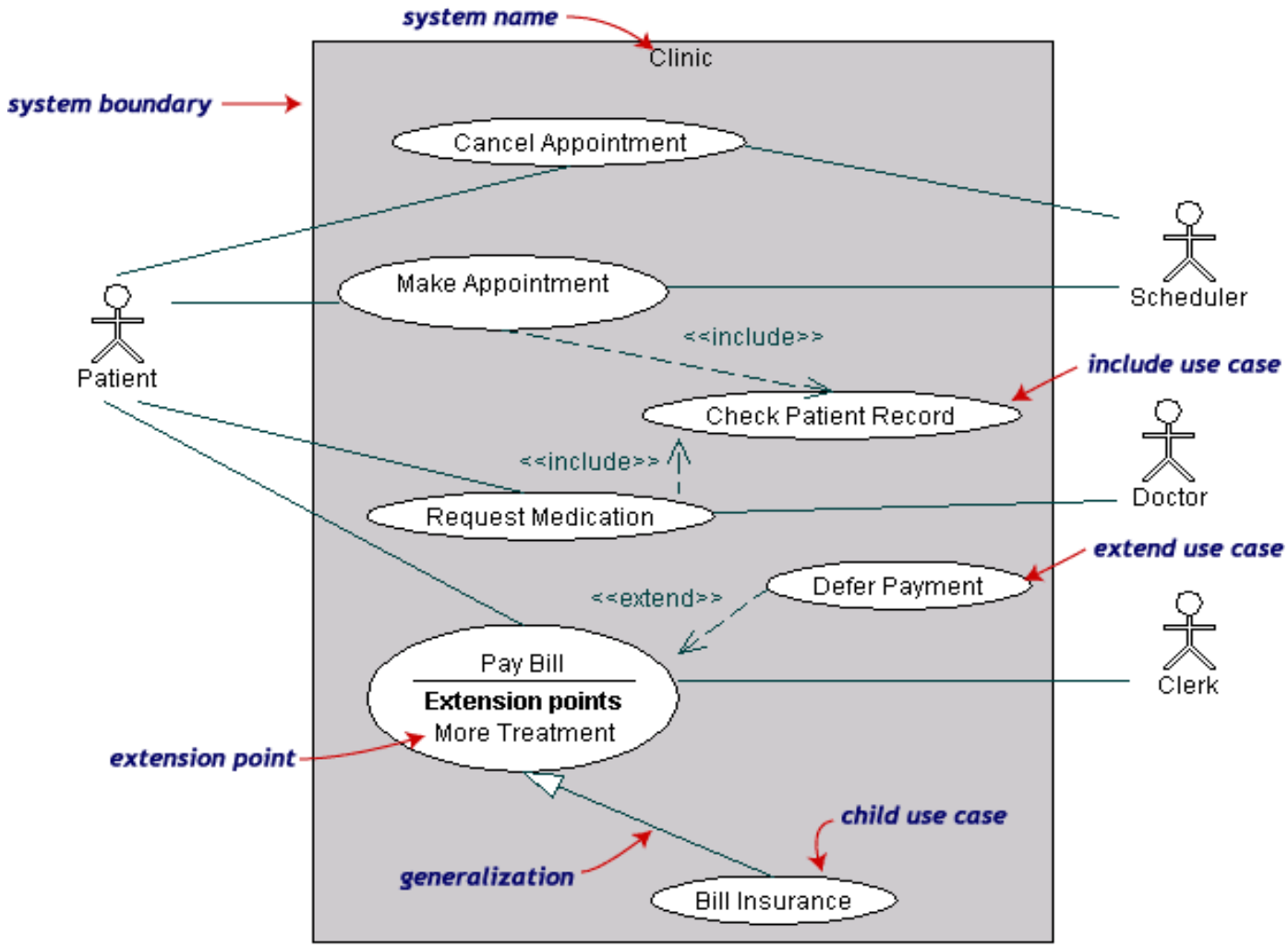
SYSTEM BOUNDARY

- Untuk memperlihatkan batasan sistem dalam diagram use case,Anda dapat menggambarkan sebuah kotak yang melingkupi semua use case, namun actor tetap berada di luar kotak.
- System boundary boxes dalam penggunaannya optional

- Contoh system boundaries yang lain



CONTOH USE CASE DIAGRAM

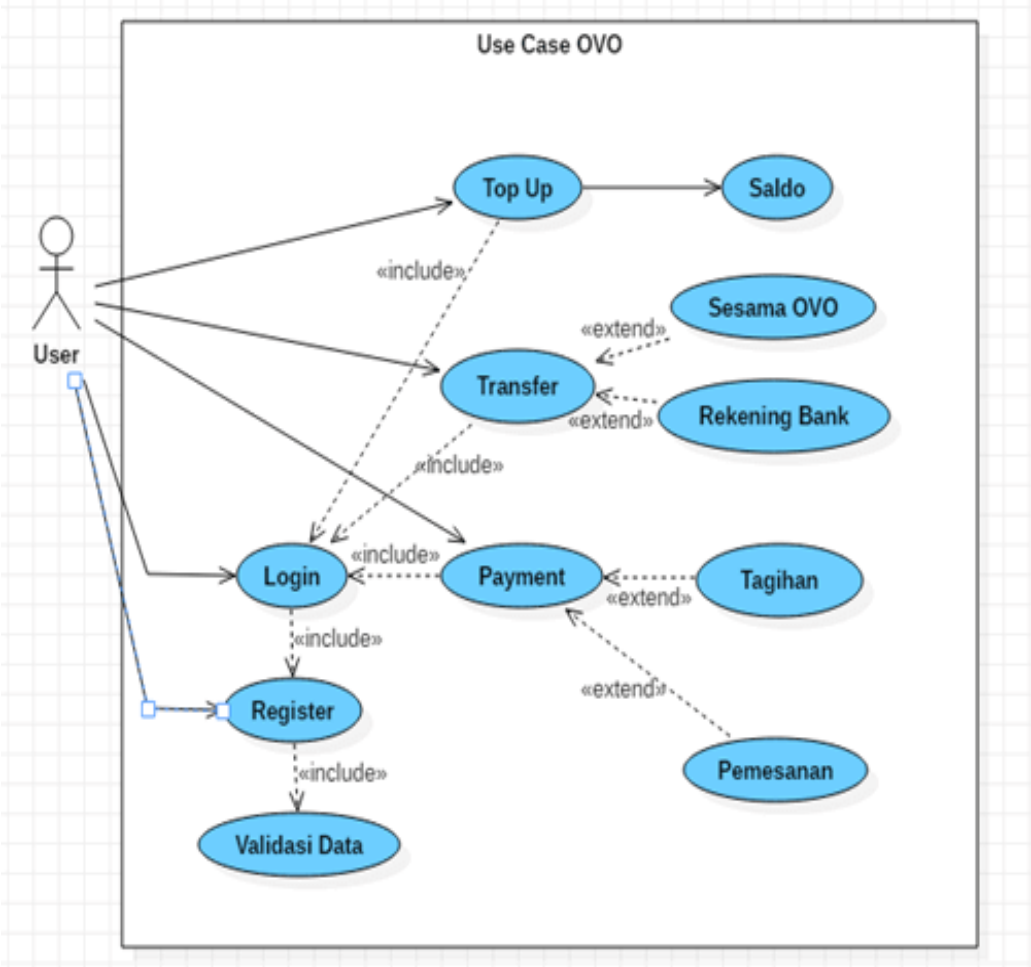


Tools Membuat Usecase Diagram

Untuk membuat sebuah use case diagram, tentunya kamu membutuhkan *tools* atau aplikasi. Berikut ini adalah beberapa aplikasi yang bisa kamu gunakan:

1. Draw.io
2. Star UML
3. Visio
4. UMLet

Contoh Use Case OVO



OVO adalah aplikasi pintar yang memberikan kemudahan dalam bertransaksi (OVO Cash), serta berfungsi untuk mengumpulkan poin di banyak tempat (OVO Points). Inilah contoh use case diagram studi kasus pada OVO

Penjelasan Usecase OVO

1. *User*: Orang yang dapat mengakses atau menggunakan aplikasi OVO, mulai dari login ke aplikasi hingga melakukan aksi terhadap aplikasi seperti top up saldo, transfer, dan payment.
2. *Register*: Register merupakan langkah pertama yang dilakukan user ketika ia tidak mempunyai akses pada aplikasi OVO. Mendaftarkan data diri ke dalam aplikasi agar dikenali.
3. Login: Setelah mendapatkan akun, user harus melakukan login agar dapat mengakses berbagai fitur aplikasi OVO.
4. *Top up*: Suatu kegiatan yang dilakukan user untuk mengisi ulang saldo OVO. Terdapat 2 pilihan alternatif untuk melakukan top up saldo, yaitu melalui ATM dan internet banking.
5. *Transfer*: Transfer berfungsi untuk mengirim atau membagikan saldo dalam aplikasi OVO ke pengguna lain, baik sesama OVO atau ke rekening tertentu.
6. *Payment*: Ketika user memilih menu payment, maka user dapat melakukan pembayaran lewat aplikasi.

Tugas Mandiri

1. Kenapa dalam proses rekayasa perangkat lunak perlu menggunakan use case diagram?
2. Dalam pengembangan software kapan kita perlu mendesain usecase diagram?
3. Jelaskan perbedaan relasi include, generalisasi dan extend.
4. Buat usecase diagram tentang ;
   a) Pesan tiket pesawat di traveloka (NIM Genap)
   b) Pesan makanan di Go Food (NIM Ganjil)

DEADLINE DIAKHIR PERKULIAHAN

REFERENSI

- **Catur Iswahyudi + Edhy Sutanta**, UML – Use Case Diagram