Analisa dan Desain Berorientasi Objek

Use Case Diagram

Ridwan Rismanto

System requirement

Use case diagram adalah tool yang sangat powerful dalam UML

- Mendeskripsikan interaksi antara user dengan sistem
- Mendeskripsikan keseluruhan sistem yang akan dibuat

Sebelum membuat use case, harus dijabarkan terlebih dahulu system requirement, atau kebutuhan, atau fitur-fitur yang ada didalam sistem yang akan dibuat.

Use case

Use case dideskripsikan dengan kata kerja.

- “Pay Bills”, “Update Payroll”, “Create Account”

Didalam use case diagram juga terdapat deskripsi terhadap masing-masing interaksi antara user dengan sistem.

Notasi use case:

Actor

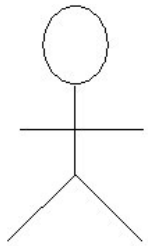
Use Case tidak dapat berdiri sendiri jika tidak ada **actor**

Actor adalah *sesuatu* yang dapat berinteraksi dengan use case

Contoh pada sistem perbankan

- Terdapat use case “Withdraw Money”
- Maka seharusnya terdapat actor “customer” yang dapat melakukan withdraw money (mengambil uang)

Notasi actor:



Customer

Actor tidak selalu *manusia*

Actor juga bisa berupa sesuatu diluar sistem yang berinteraksi dengan use case.

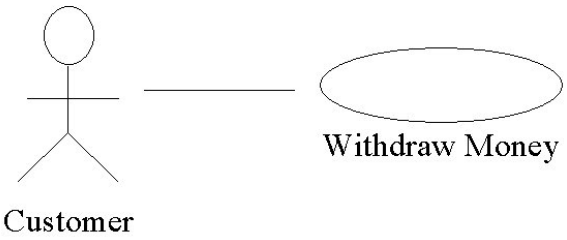
- Misal, sistem lain yang berinteraksi dengan sistem yang akan dibuat.

Actor juga bisa berupa sebuah konsep abstrak, seperti **waktu**, atau **tanggal**.

- Misal, terdapat use case “Hapus order kadaluwarsa” pada sistem informasi pengorderan barang, dan actor yang berinteraksi dengan use case tersebut adalah “Hari libur”

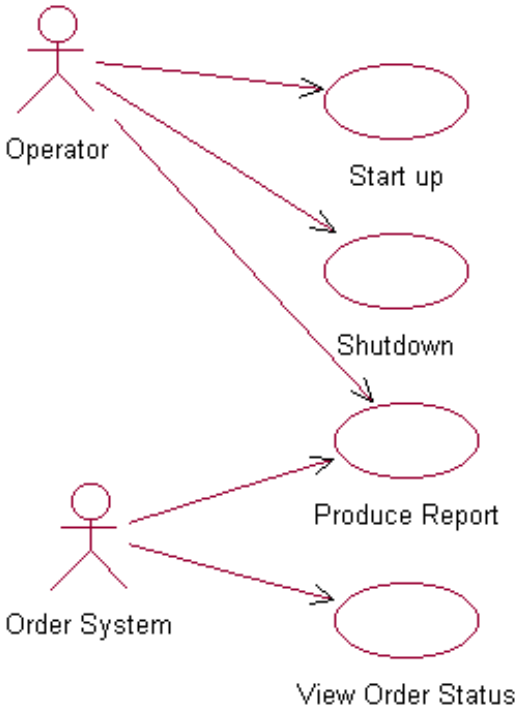
Interaksi actor dengan use case

Interaksi antara actor dengan use case digambarkan dengan garis lurus:



Sebuah actor dapat berinteraksi dengan

banyak use case, dan sebaliknya:



Tujuan use case diagram

Mendefinisikan scope dari sistem

- Memvisualisasikan size dan scope dari keseluruhan proses development

Use case sangat mirip dengan sistem requirement

- Akan tetapi lebih detail dan fokus

Use case adalah gambaran dari keseluruhan sistem

- Artinya, sesuatu yang tidak ada di use case adalah tidak termasuk dari sistem yang akan dibuat

Memungkinkan komunikasi antara user dan developer

- Karena diagramnya sederhana dan mudah dipahami

Use case dapat digunakan sebagai rujukan oleh tim developer

- Atau backbone dari proses development

Dapat dengan mudah melakukan planning dalam proses develop

Dapat digunakan sebagai basis untuk testing dan pengujian sistem

Dapat membantu pembuatan manual penggunaan sistem atau *user guides*

Menentukan use case

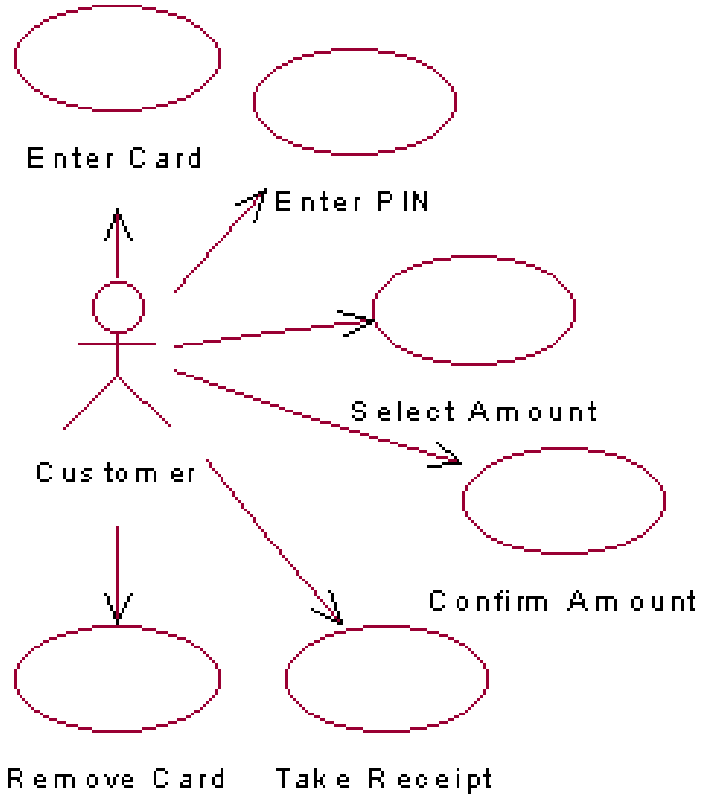
Kesalahan yang sering terjadi dalam pembuatan use case.

Misal pada mesin ATM yang memiliki fitur untuk mengambil uang. Skenarionya:

- Masukkan kartu ATM
- Masukkan PIN
- Pilih jumlah uang yang akan diambil
- Konfirmasi pengambilan uang
- Ambil kartu ATM
- Ambil resi

Jika kita terjemahkan setiap interaksi diatas kedalam use case, maka kira-kira hasilnya seperti ini:

Menentukan use case (lanjutan)



Menentukan use case (lanjutan)

Use case diatas sepertinya sah-sah saja.

Namun jika seperti itu, maka dalam sebuah sistem yang kompleks, use case akan membengkak dan terlalu berlebihan.

Kunci dalam membuat use case:

Use case haruslah memenuhi tujuan dari aktor

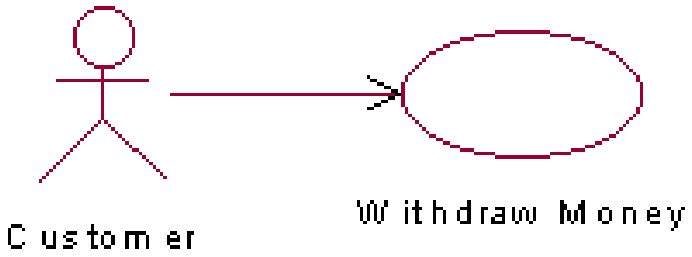
Dalam hal ini, apakah “Ambil resi” termasuk tujuan dari pengguna mesin ATM?

- Apakah dunia akan kiamat jika resi tidak keluar?

Berdasarkan hal ini, maka sebenarnya tujuan utama dari si actor tersebut adalah “withdraw money” atau ambil uang.

Sehingga:

Menentukan use case (lanjutan)

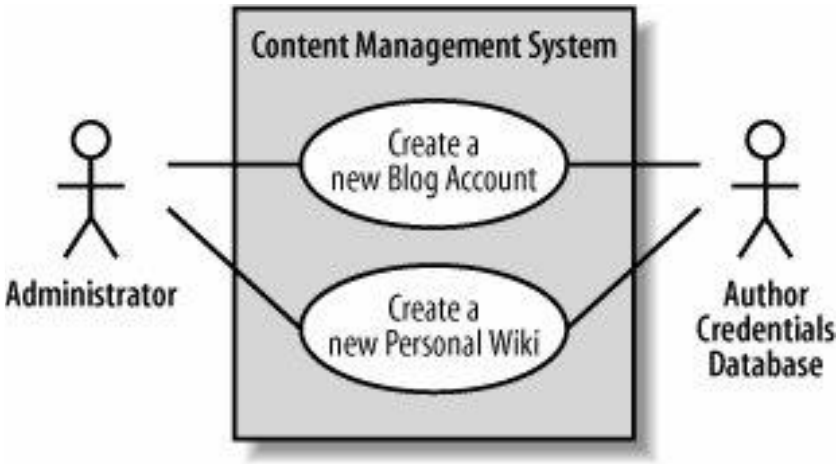


System boundaries

Digunakan untuk menggambarkan batas dari sebuah sistem.

Digambarkan dengan kotak, dengan nama sistem, dan didalamnya adalah use case. Namun actor digambarkan diluar kotak tersebut.

Contoh:



Relasi pada use case

Relationship pada use case terdiri dari beberapa macam:

- Generalization
- Extend
- Include

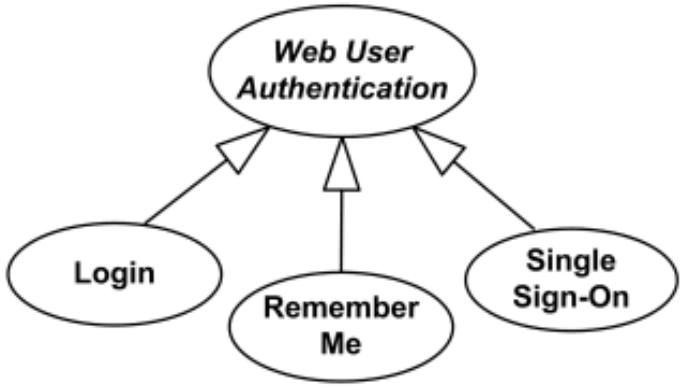
Relasi: Generalization

Relasi use case yang mirip seperti *inheritance.*

Digunakan jika ada use case yang tujuan utamanya sama, tapi lebih spesifik.

Digambarkan dengan tanda panah mengarah ke use case induknya.

Contoh pada use case “web user authentication”, terdapat beberapa metode yang berbeda, namun tujuan akhirnya sama, yaitu autentikasi user:



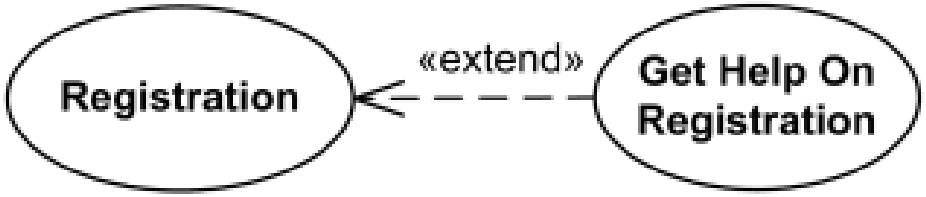
Relasi: Extend

Relasi extend, adalah jika ada use case yang dapat memanggil use case lain, tapi sifatnya opsional.

Digambarkan dengan panah putus-putus, dengan caption <<extend>>, mengarah **dari use case optional ke use case utama**.

Misal, pada use case “registration” maka ada opsi use case “get help on registration” yang sifatnya tidak wajib (optional).

Maka use case utamanya adalah “registration”, dan opsionalnya atau extend-nya adalah “get help on registration”.

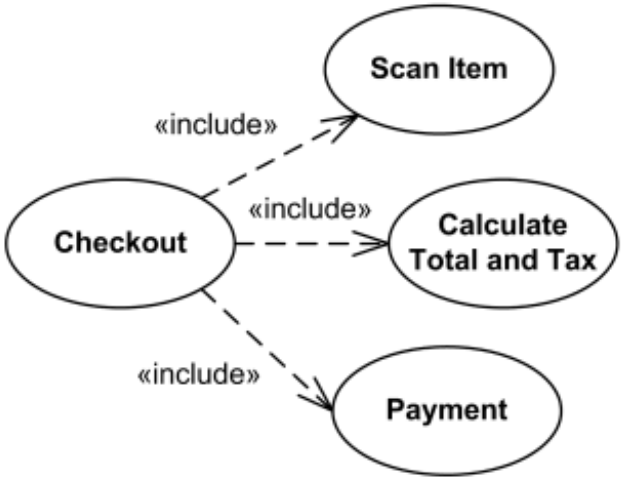


Relasi: Include

Relasi include adalah jika sebuah use case membutuhkan pemanggilan ke use case lain dan sifatnya **harus** atau *mandatory*.

Digambarkan dengan panah putus-putus, dengan caption <<include>>, mengarah dari **use case utama ke use case included**

Misal pada use case “checkout” pada sebuah sistem kasir supermarket, harus melibatkan use case “scan item”, “calculate total and tax” dan “payment”. Maka:



Deskripsi use case

Sebuah use case selalu disertai dengan penjelasan detail.

Template deskripsi use case:

Use Case:	Nama use case
Deskripsi:	Deskripsi singkat dari use case
Pre-conditions:	Kondisi/syarat tertentu agar use case ini dapat dijalankan
Post-conditions:	Kondisi setelah use case ini dijalankan
Main flow:	Interaksi apa saja yang dilakukan untuk menjalankan use case ini, misal untuk use case “withdraw money” maka main flow nya adalah “masukkan kartu, masukkan pin, dsb…”
Alternate flow:	Jika ada interaksi alternative
Exception flow:	Jika ada skenario untuk menghandle kejadian-kejadian tidak terduga, misal jika saldo tidak cukup, dsb…

Use case dalam fase analisis

Mengidentifikasi use case sebanyak mungkin, tetapi tidak perlu terlalu detail.

- Ingat *kunci dalam membuat use case*

Setelah mengidentifikasi use case yang ada, maka dapat dilakukan cross reference dengan sistem requirements.

Tidak perlu memaksakan sesuatu agar menjadi use case atau actor.

- Biasanya akan muncul use case tambahan dalam proses desain lebih lanjut nantinya. Dan ini normal.

Bagaimana membuat use case

Brainstorming dengan user

- Mengidentifikasi actor.
- Mengidentifikasi use case.
- Tulis deskripsi/notes singkat yang menjelaskan tentang masing-masing use case.
- Gambarkan diagram use case, dan deskripsi use case.

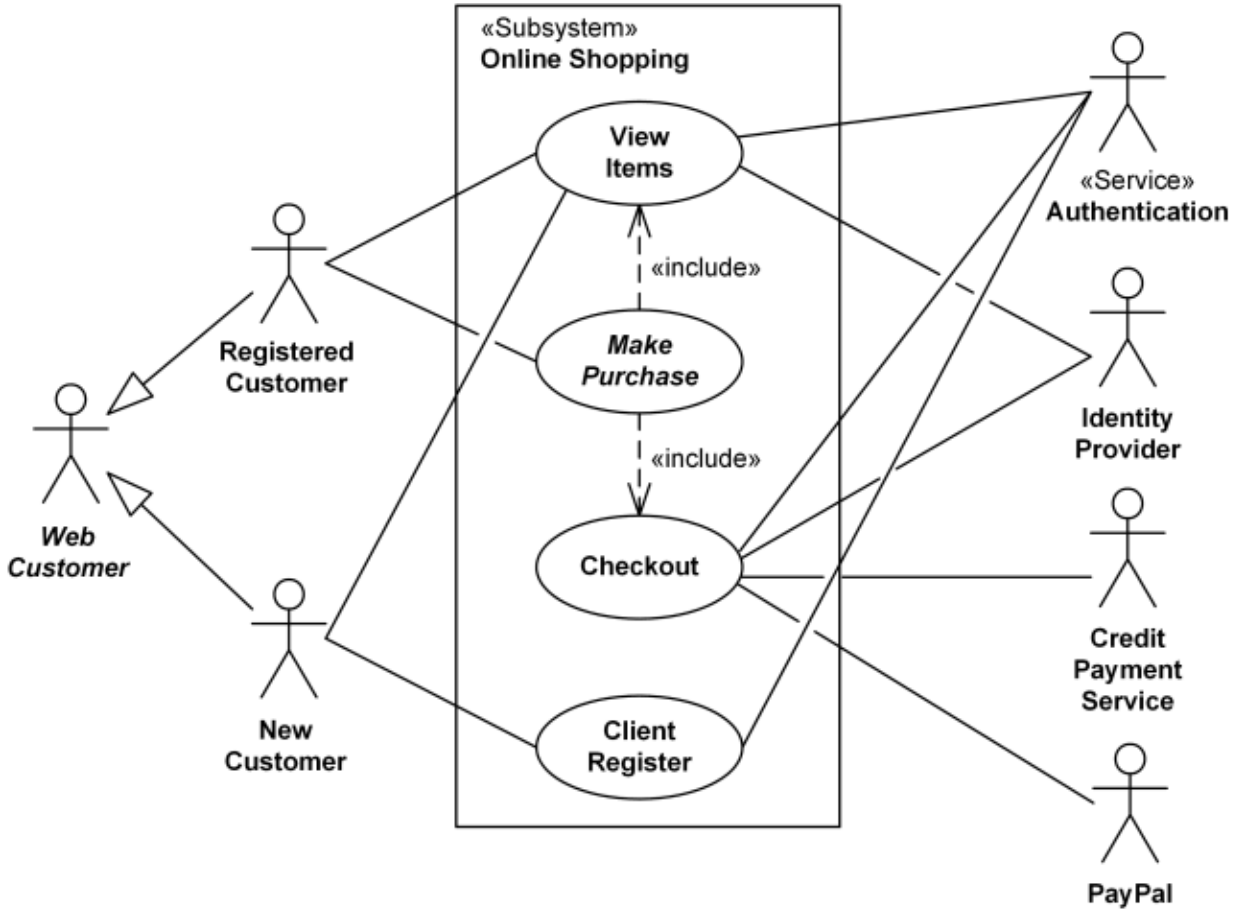
Contoh kasus: online shopping

Requirement:

- Aplikasi berbasis web dengan autentikasi user.
- Customer yang sudah registrasi dapat melihat item dan melakukan pembelian serta pembayaran.
- Customer yang belum registrasi dapat melihat item dan melakukan registrasi.
- Terdapat fasilitas:
  - Pencarian item
  - Browse item
  - Recommended item
  - Shopping cart
  - Wishlist

Contoh kasus: online shopping

Top level use case:



Contoh kasus: online shopping

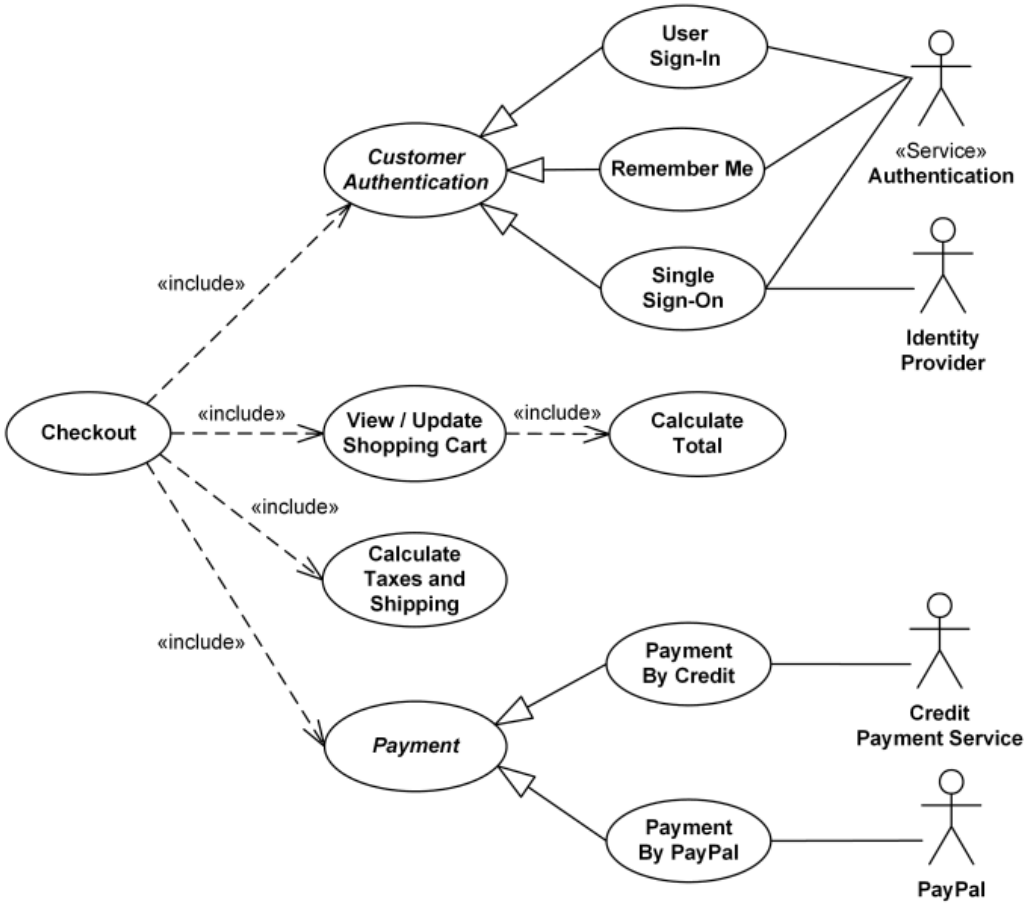
Use case “view items” dapat di-extend dengan use case lain yang sifatnya opsional.

Use case “customer authentication” di-include-kan dengan use case “view recommended items” dan “add to wishlist” karena customer memang harus dalam kondisi login.

Namun di lain sisi, untuk memasukkan item ke shopping cart, customer tidak perlu login.

Contoh kasus: online shopping

Use case “checkout” di-include-kan dengan use case lain yang sifatnya harus/mandatory.



Latihan

Buat system requirement dan use case diagram (pilih salah satu):

1. Pemesanan tiket kereta
2. Booking hotel
3. Koperasi simpan pinjam

Thanks!

Any questions?

You can find me at:
rismanto@polinema.ac.id