Notasi dalam UML

Actor

Gambar 1. Notasi Actor

Actor menggambarkan segala pengguna software aplikasi (user). Actor memberikan suatu gambaran jelas tentang apa yang harus dikerjakan software aplikasi. Sebagai contoh sebuah actor dapat memberikan input kedalam dan menerima informasi dari software aplikasi, perlu dicatat bahwa sebuah actor berinteraksi dengan use case, tetapi tidak memiliki kontrol atas use case. Sebuah actor mungkin seorang manusia, satu device, hardware atau sistem informasi lainnya.

Use Case

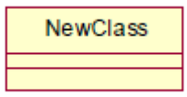
Gambar 2. Notasi Use Case

Use case menjelaskan urutan kegiatan yang dilakukan actor dan sistem untuk mencapai suatu tujuan tertentu. Walaupun menjelaskan kegiatan, namun use case hanya menjelaskan apa yang dilakukan oleh actor dan sistem bukan bagaimana actor dan sistem melakukan kegiatan tersebut.

- ✓ Use-case Konkret adalah use case yang dibuat langsung karena keperluan actor. Actor dapat melihat dan berinisiatif terhadapnya
- ✓ Use-case Abstrak adalah use case yang tidak pernah berdiri sendiri. Use case abstrak senantiasa termasuk didalam (include), diperluas dari (extend) atau memperumum (generalize) use case lainnya.

Untuk menggambarkannya dalam use case model biasanya digunakan association relationship yang memiliki stereotype include, extend atau generalization relationship. Hubungan include menggambarkan bahwa suatu use case seluruhnya meliputi fungsionalitas dari use case lainnya. Hubungan extend antar use case berarti bahwa satu use case merupakan tambahan fungsionalitas dari use case yang lain jika kondisi atau syarat tertentu terpenuhi.

Class



Gambar 3. Notasi Class

Class merupakan pembentuk utama dari sistem berorientasi obyek, karena class menunjukkan kumpulan obyek yang memiliki atribut dan operasi yang sama. Class digunakan untuk mengimplementasikan interface.

Class digunakan untuk mengabstraksikan elemen-elemen dari sistem yang sedang dibangun. Class bisa merepresentasikan baik perangkat lunak maupun perangkat keras, baik konsep maupun benda nyata.

Notasi class berbentuk persegi panjang berisi 3 bagian: persegi panjang paling atas untuk nama class, persegi panjang paling bawah untuk operasi, dan persegi panjang ditengah untuk atribut.

Atribut digunakan untuk menyimpan informasi. Nama atribut menggunakan kata benda yang bisa dengan jelas merepresentasikan informasi yang tersimpan didalamnya. Operasi menunjukkan sesuatu yang bisa dilakukan oleh obyek dan menggunakan kata kerja.

Interface

Gambar 4. Notasi Interface

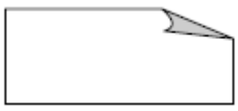
Interface merupakan kumpulan operasi tanpa implementasi dari suatu class. Implementasi operasi dalam interface dijabarkan oleh operasi didalam class. Oleh karena itu keberadaan interface selalu disertai oleh class yang mengimplementasikan operasinya. Interface ini merupakan salah satu cara mewujudkan prinsip enkapsulasi dalam obyek.

Interaction

Gambar 5. Notasi Interaction

Interaction digunakan untuk menunjukkan baik aliran pesan atau informasi antar obyek maupun hubungan antar obyek. Biasanya interaction ini dilengkapi juga dengan teks bernama operation signature yang tersusun dari nama operasi, parameter yang dikirim dan tipe parameter yang dikembalikan.

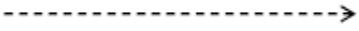
Note



Gambar 6. Notasi Note

Note digunakan untuk memberikan keterangan atau komentar tambahan dari suatu elemen sehingga bisa langsung terlampir dalam model. Note ini bisa disertakan ke semua elemen notasi yang lain.

Dependency



Gambar 7. Notasi Dependency

Dependency merupakan relasi yang menunjukan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain. Elemen yang ada di

bagian tanda panah adalah elemen yang tergantung pada elemen yang ada dibagian tanpa tanda panah.

Terdapat 2 stereotype dari dependency, yaitu include dan extend. Include menunjukkan bahwa suatu bagian dari elemen (yang ada digaris tanpa panah) memicu eksekusi bagian dari elemen lain (yang ada di garis dengan panah).

Extend menunjukkan bahwa suatu bagian dari elemen di garis tanpa panah bisa disisipkan kedalam elemen yang ada di garis dengan panah.

Association

Gambar 8. Notasi Asociation

Association menggambarkan navigasi antar class (navigation), berapa banyak obyek lain yang bisa berhubungan dengan satu obyek (multiplicity antar class) dan apakah suatu class menjadi bagian dari class lainnya (aggregation).

Navigation dilambangkan dengan penambahan tanda panah di akhir garis. Bidirectional navigation menunjukkan bahwa dengan mengetahui salah satu class bisa didapatkan informasi dari class lainnya. Sementara UniDirectional navigation hanya dengan mengetahui class diujung garis association tanpa panah kita bisa mendapatkan informasi dari class di ujung dengan panah, tetapi tidak sebaliknya.

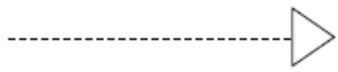
Aggregation mengacu pada hubungan "has-a", yaitu bahwa suatu class memiliki class lain, misalnya Rumah memiliki class Kamar.

Generalization

Gambar 9. Notasi Generalization

Generalization menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan generalization, class yang lebih spesifik (subclass) akan menurunkan atribut dan operasi dari class yang lebih umum (superclass) atau "subclass is superclass". Dengan menggunakan notasi generalization ini, konsep inheritance dari prinsip hirarki dapat dimodelkan.

Realization



Gambar 10. Notasi Realization

Realization menunjukkan hubungan bahwa elemen yang ada di bagian tanpa panah akan merealisasikan apa yang dinyatakan oleh elemen yang ada di bagian dengan panah. Misalnya class merealisasikan package, component merealisasikan class atau interface.

Use-Case Modelling

Use-case Modelling adalah teknik paling sederhana dan paling efektif untuk memodelkan kebutuhan sistem berdasarkan pandangan user (Bennet, Simon, Steve Mc Robb dan Ray Farmer, 2000). Use-case modelling digunakan untuk menunjukkan bagaimana sistem atau kerja nyata dari suatu sistem atau bagaimana user ingin sistem itu bekerja. Use-case pada dasarnya adalah langkah awal dari analisis berdasarkan obyek dengan UML.

Use-case model terdiri dari actor dan use-case. Actor merepresentasikan user dan sistem lain yang berinteraksi dengan sistem. Use-case model sesungguhnya merepresentasikan tipe dari user, bukan suatu hal dari user. Use-case merepresentasikan karakteristik sistem, skenario dari tujuan sistem kedalam reaksi untuk menggerakkan actor.

Use-case adalah skenario untuk memahami kebutuhan user. Use-case model dapat menjadi kerangka dalam proyek pengembangan, perencanaan dan dokumentasi dari kebutuhan sistem. Use-case adalah interaksi antara user dan sistem, menggambarkan tujuan dari sistem dan tanggapan sistem untuk user. Use-case model mencoba untuk mensistematiskan identifikasi dari kegunaan sistem dan tanggapan dari sistem. Use-case model juga dapat mencakup kelas-kelas dan hubungan yang dimiliki oleh sub-sistem dari sistem.

Setiap use-case atau skenario merepresentasikan apa yang user ingin lakukan. Setiap use-case harus mempunyai nama dan deskripsi teks pendek, hanya sedikit paragraph.

Identifikasi Actor

Mengidentifikasikan actor adalah sama pentingnya dengan mengidentifikasikan kelas, struktur, hubungan, atribut dan karakteristik. Ketika menentukan actor, sangat penting untuk berfikir mengenai aturan rata-rata dari orang atau jenis pekerjaan. User mungkin memainkan lebih dari satu aturan. Actor harus merepresentasikan user tunggal.

Harus mengidentifikasi actor dan mengerti bagaimana mereka akan berguna dan berinteraksi dengan sistem. Kandidat untuk actor dapat ditemukan dengan menjawab pertanyaan-pertanyaan berikut :

- ✓ Siapa yang menggunakan sistem ?
  Kelompok mana yang membutuhkan pertolongan dari sistem untuk melakukan suatu pekerjaan.
- ✓ Kelompok pengguna mana yang membutuhkan penampilan fungsi sistem ?
  Fungsi ini dapat merupakan fungsi utama dan fungsi sekunder, seperti administrasi.
- ✓ Apa masalah dari penyelesaian aplikasi (untuk siapa) ?
- ✓ Dan yang terakhir, bagaimana user menggunakan sistem (use-case) ?
  Apa yang mereka lakukan dengan sistem ?

Menemukan Use-Case

Menurut Bennet, Simon, Steve Mc Robb dan Ray Farmer (2000), langkah-langkah untuk menemukan use-case adalah sebagai berikut:

- ✓ Untuk setiap actor, temukan tugas dan fungsi actor harus dapat melakukan apa yang sistem butuhkan dari actor untuk melakukannya. Use-case harus merepresentasikan kegiatan dari kejadian untuk menghasilkan tujuan.
- ✓ Beri nama untuk setiap use-case.
- ✓ Deskripsikan masing-masing use-case.

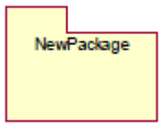
Berapa banyak use-case yang dibutuhkan ? Untuk sepuluh orang per tahun proyek, actor mungkin mendapatkan dua puluh use-case (tidak terhitung hubungan uses dan extend). Dalam penelitian lain, mungkin mencapai seratus use-case untuk sebuah proyek. Tidak ada formula tertentu, hanya dibutuhkan kedinamisan dan kerja yang menemukan kenyamanan.

Penamaan Use-Case

Penamaan use-case harus menguntungkan deskripsi global dari fungsi use-case. Nama harus mendeskripsikan apa yang terjadi ketika bagian dari kinerja use-case bekerja. Penamaan dalam kata kerja atau kata benda, harus dilakukan dengan hati-hati karena deskripsi dari use-case harus merupakan gambaran dan sifatnya tetap.

Use-case adalah alat utama dalam menggambarkan kebutuhan. Menggambarkan use-case adalah salah satu langkah pertama yang penting untuk melakukan sesuai dengan kebutuhan. Setiap use-case adalah kebutuhan yang berpotensi. Setiap use-case atau skenario merepresentasikan apa yang user ingin lakukan.

Package

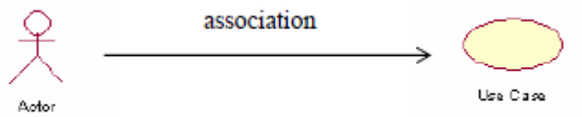


Gambar 11. Notasi Package (Paket)

Paket adalah mekanisme pengelompokkan yang dgunakan untuk menandakan pengelompokkan elemen-elemen model. Sebuah paket dapat mengandung beberapa paket lain didalamnya. Paket digunakan untuk memudahkan mengorganisasikan elemen-elemen model.

Relationship

Adalah koneksi antar model elemen.



Gambar 12. Association Relationship

Association Relationship, memodelkan koneksi antar obyek dari kelas yang berbeda.

Interaksi antara actor dan use-case dalam use-case model biasanya digunakan association relationship yaitu :

- <<uses>>

Hubungan uses menunjukkan bahwa prosedur dari use-case merupakan bagian dari prosedur yang menggunakan use-case. Tanda panah menunjukkan keadaan tidak mengakibatkan pemanggilan prosedur dalam menggunakan use-case. Relasi uses antara use-case ditunjukkan dengan panah generalisasi dari use-case. Use-case yang dilakukan secara berulang, digunakan untuk meminimalkan pekerjaan.

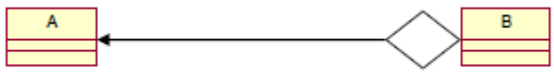
- <<extend>>

Hubungan extend antar use-case berarti bahwa suatu use-case merupakan tambahan kegunaan dari use-case yang lain jika kondisi atau syarat tertentu dipenuhi. Jika prosedur dari use-case merupakan alternatif untuk menjelaskan use-case lain. Use-case akan dikerjakan apabila salah satu syarat terpenuhi.

Hubungan generalisasi antar use-case menunjukkan bahwa use-case yang satu merupakan spesialisai dari yang lain.

- <<include>>

Hubungan include menggambarkan suatu use-case seluruhnya meliputi kegunaan dari use-case lainnya. 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 dieksekusi secara normal. Sebuah use-case dapat di-include oleh lebih dari use-case lain, sehingga duplikasi fungsional dapat dihindari.



Gambar 13. Aggregation Relationship

Aggregation Relationship, adalah bentuk khusus assosiasi yang memodelkan hubungan keanggotaan antara 2 kelas, yakni satu kelas disusun oleh kelas lainnya. Gambar diatas memperlihatkan bahwa B (kelas aggregate) yang secara fisik dibentuk oleh A atau secara logis B mengandung A.