**Deployment Diagram**

Deployment Diagram menggunakan gambaran kotak tiga dimensi untuk menggambarkan

masing-masing platform. Diagram ini akan dihubungkan dengan garis-garis yang menggambarkan

koneksi jaringan antar platform yang memungkinkan komponen suatu sistem berkomunikasi.

Garis tersebut dapat di beri label dengan protokol jaringan yang digunakan untuk berkomunikasi.

Tanda Multiplicity yang sama dengan diagram kelas dapat digunakan untuk menggambarkan

jumlah mesin yang di ijinkan berkomunikasi dengan sistem.

Fungsi dari Deployment Diagram adalah :

1. Menunjukkan dimana setiap komponen-komponen perangkat keras dan perangkat lunak
2. Menunjukkan hubungan komunikasi antara komponen perangkat keras;
3. Menunjukkan struktur dari sistem run-time.

Diagram deployment mewakili pandangan pengembangan sistem sehingga hanya akan ada satu

diagram deployment untuk satu sistem. Diagram deployment terdiri dari node dan node merupakan

perangkat keras fisik yang digunakan untuk menyebarkan aplikasi. Diagram deployment banyak digunakan oleh system engineer. Tiap node pada diagram

deployment mewakili satu unit komputasi sistem yang dalam banyak hal merupakan bagian dari

perangkat keras.Diagram deployment umumnya memiliki node dan hubungan kebergantungan. Memungkinkan juga dalam diagram deployment terdapat komponen.

Sequence Diagram digunakan di UML untuk menggambarkan deskripsi setiap

skenario di dalam Use Case. Sequence Diagrammemodelkan obyek-obyek yang mengambil

peranan di dalam skenario dan menggambarkan interaksinya. Selama fase analisis,

sequence diagram mencerminkan suatu interkasi. Selama fase perancangan,setiap

interaksi dikonversi ke dalam pemanggilan method sesuai dengan bahasa pemroraman yang

dipilihnya.

Setiap sequence diagram :

1. mendeskripsikan use case atau satu skenario di dalam sebuah use case secara

diagramatik

1. mengidentifikasi obyek-obyek yang terlibat di setiap proses
2. mengidentifikasi aksi atau peristiwa yang terjadi di setiap proses
3. mengidentifikasi informasi apa saja yang harus dikirim di sekitarnya untuk setiap

proses

1. mengidentifikasi respon-respon apa saja yang diperlukan setelah setiap aksi atau

peristiwa

System analyst harus membuat kurang lebih satu sequence diagram untuk setiap use case. Use casedapat memiliki lebih dari satu alur atau cabang yang muncul ketika keputusan dibuat di dalam aliran informasi. Proses recallsetiap jalur berbeda yang ada diuse case, yaitu satu jalur tanpa cabang, dinamakan sebagai skenario. Satu sequence diagramumumnya digambarkan untuk setiap skenario di dalam use case. Sequence diagram menggambarkan aliran informasi dalam periode waktu tertentu.

**Class Diagram**

Class diagram adalah diagram yang digunakan untuk membentuk model konseptual sistem aplikasi danmemberikan detail model untuk diterjemahkan ke dalam kode program.

Class diagram memiliki 3 blok utama, yaitu :

1. Blok pertama adalah nama kelas.
2. Blok kedua adalah atribut atau field kelas.
3. Blok ketiga adalah method atau operasi yang dapat dilakukan kelas.

Beberapa aturan dalam class diagram adalah tentang visibilitas dan scope.

* Visibilitas pada class diagram digambarkan dengan simbol dibawah ini :
* Scope dapat berarti jangkauan sebuah variabel atau methoddapat diakses, namun pada class

diagram,scope adalah cara mengakses sebuah method atau variabel.

Scope dapat dibagi menjadi dua, yaitu Penulisan classifier member/method pada class diagram adalah dengan memberikan garis bawah (underline) pada member atau method statis. Jika tidak ada garis bawah pada member atau method, maka diasumsikan merupakan instance Member

* Nama Class

Digunakan untuk membedakan antara satu kelas dan kelas yang lain.

Contohnya : Kendaraan, Pegawai

* Attribute

Digunakan untuk menyimpan state, pada bahasa pemrograman ini berupa field. Bisa juga

diartikan apa yang dimiliki oleh sebuah objek.

- Contohnya : nama, alamat, usia, nim, warna

- Aturan penggunaan :modifier nama\_attribute : tipedata

- Contoh penggunaan : nama : String dibaca attribute nama memiliki modifier private

dengan tipe dataString

* Mengakses Atribut yang Terenkapsulasi

Karena atribut yang dienkapsulasi tidak dapat diakses langsung, maka diperlukan

method untuk mengaksesnya. Untuk melakukan ini, perlu dibangun operasi‐operasi

berikut ini :

- Method untuk menyeting nilai atribut.

Contoh : setName, setAddress,setDateOfBirth, setAccountNumber;

- Method untuk mengakses nilai dari atribut, mencakup derived attribute.

Contoh : getName, getAddress, getDateOfBirth, getAccountNumber;

- Method untuk menanyakan pertanyaan dengan jawaban berupa true atau false.

Contohm : isConnected, isSave.

* Kelas Eventdan Event Handler

Pada teknologi Java, human interface merupakan ekstensi dari kelas Swing/AWT.

EventHandler mendaftarkan sebuah event, dengan obyek‐obyek Swing/AWT, dan method

spesifik yang akan dipanggil ketika ada event yang terjadi (sebagai contoh, tombol ditekan).

Teknologi Java menyediakan framework untuk Swing/AWT event.Namun, programmer dapat

membuat sendiri struktur event untuk event non GUI.Obyek event Handler diinstansiasi

dengan link interaksi ke obyek-obyek bisnis dan obyek-obyek human interface. Sebagai contoh,

Obyek event handler dapat didaftarkan ke obyek MenuItem. Ketika obyek MenuItem dipilih,

method actionPerformed pada obyek handler dipanggil.Event Handler dapat menyalin data dari

obyek-obyek bisnis ke obyek‐obyek GUI atau melakukan prosedur sebaliknya, yaitu menyalin data

dari GUI ke obyek‐obyek bisnis. Prosedur dapat dilakukan dengan obyek yang sama atau obyek

terpisah di mana setiap obyek melakukan satu tugas spesifik.

**Actifity Diagram**

Diagram aktivitas atau dalam bahasa inggris [activity diagram](https://en.wikipedia.org/wiki/Activity_diagram) adalah representasi grafis dari seluruh tahapan alur kerja. Diagram ini mengandung aktivitas, pilihan tindakan, perulangan dan hasil dari aktivitas tersebut. Pada pemodelan [UML](http://id.wikipedia.org/wiki/Unified_Modeling_Language), diagram ini dapat digunakan untuk menjelaskan proses bisnis dan alur kerja operasional secara langkah demi langkah dari komponen suatu sistem.

Pengertian Activity Diagram

Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Activity diagram merupakan state diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya state sebelumnya (internal processing). Oleh karena itu activity diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. Menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Dipakai pada business modeling untuk memperlihatkan urutan aktifitas proses bisnis. Struktur diagram ini mirip flowchart atau Data Flow Diagram pada perancangan terstruktur. Sangat bermanfaat apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan. Activity diagram dibuat berdasarkan sebuah atau beberapa use case pada use case diagram.

**Usecase Diagram**

Usecase diagram merupakan gambaran fungsionalitas yang diharapkan dari sebuah sistem yang ditekankan adalah “apa” yang diperbuat sistem bukan “bagaimana” sistem mengerjakannya. Usecase menggambarkan kebutuhan sistem dari sudut pandang user, menggambarkan hubungan antara usescase dan actor.Secara umum usecase mempunyai beberapa bagian penting seperti:

* Usecase

Merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja. Dinotasikan dengan gambar horizontal elipse dan tidak boleh ada 2 atau lebih nama usecase yang sama. Simbolnya seperti ini:

* Actors

Berikut ini merupakan ciri-ciri actor :

o Menggambarkan orang, sistem,atau external entitas yang menyediakan atau menerrima informasi dari system.

o Aktor menggambarkan sebuah tugas atau peran dan bukan jabatan.

o Aktor memberikan input atau menerima informasi.

o Aktor biasayan menggunakan kata benda.

o Tidak boleh ada komunikasi langsung antar actor.

o Meletakkan aktor utama di pojok kiri atas.

o Simbolnya seperti dibawah ini:

System Boundary

Digambarkan dengan kotak disekitar usecase, untuk menggambarkan jangkauan sistem. Digunakan apabila memberikan beberapa alternatif sistem yang dapat dijadikan pilihan. System boundary dalam penggunaanya bersifat optional. 5

* Asosiasi antara aktor dan usecase.

o Ujung panah pada asosiasi mengindikasin siapa / apa yang meminta interaksi dan bukannya mengindikasin aliran data.

o Menggunakan garis tanpa panah.

o Jika asosiasi mengguankan panah terbuka untuk mengindikasi bila aktor berinteraksi secara pasif dengan sistem.

* Asosiasi antar usecase.

o <<Include>>

- Tanda panah terbuka harus terarah ke sub usecaseTermasuk didalam usecase lain / required / diharuskan ada.

- Cara penggambaran association include secara horizontal.

o <<extends>>

- Perluasan dari usecase lain jika kondisi atau syarat terpenuhi.

- Tanda panah terbuka harus terarah ke parent / base usecase.

- Gambaran association extend secara vertikal.

 Generalisasi/ inheritance antara usecase.

o Digambarkan dengan garis berpanah tertutup pada salah satu ujungnya yang menunjukkan lebih umum.

o Cara penggambarannya secara vertical dengan inheriting usecase dibawah parent usecasenya.

 Generalisasi / inheritance antara actor.

o Gambarkan secara vertikal dengan inheriting aktor dibawah parent usecase.

* Relationship

Asosiasi bukan menggambarkan aliran data / informasi. Asosiasi digunakan untuk menggambarkan bagaimana aktor terlibat dalam usecase. Terdapat 4 jenis relasi yang timbul pada usecase diagram yaitu: