

DAFTAR NOTASI UML

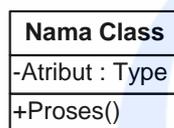
1. Use case diagram



Actor adalah karakter yang akan menjalankan setiap proses pada sistem.

Use case adalah proses yang akan dilakukan oleh si actor.

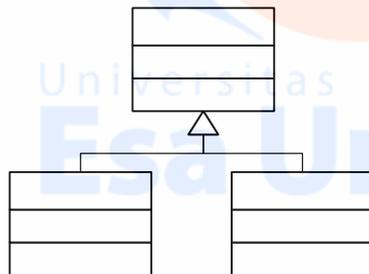
2. Class diagram



Class

Actor

Use case

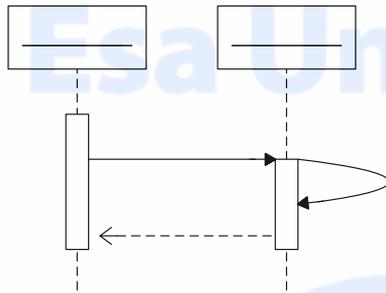


Proses Multiplicity pada Class diagram

- 1 menjelaskan bahwa sebuah transaksi terjadi apabila memiliki 1 objek.
- 0..1 menjelaskan bahwa sebuah transaksi terjadi apabila memiliki 1 objek atau bahkan tidak memiliki objek.

- * menjelaskan bahwa sebuah transaksi terjadi tanpa perlu ada batasan objek yang dimiliki.

3. Sequence diagram



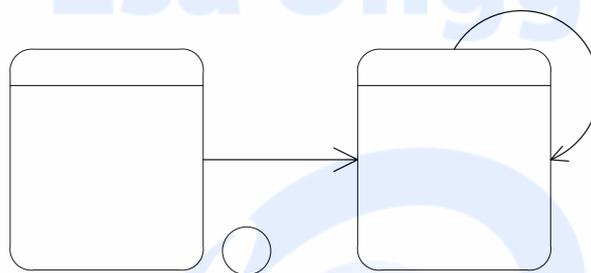
Prose self call yaitu aktivitas yang dilakukan pada class yang bertujuan sebagai proses dalam menjalankan aktivitas pada class itu sendiri.

Proses looping bisa terjadi pada sequence diagram apabila dalam menjalankan sebuah proses terjadi pengulangan secara terus menerus.

Nama Class

Nama Class

4. State machine diagram



Message

Return message

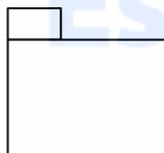
Keterangan proses pada statemachine diagram antara lain:

1. On entry / action

On entry dimaksudkan bahwa setiap kali memasuki state penjualan action akan tampil secara otomatis.

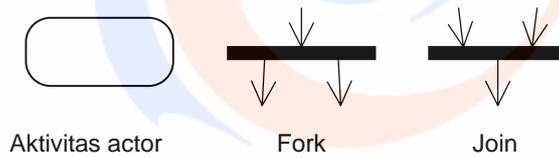
2. Do / action
Do / action adalah aktivitas yang dilakukan oleh state itu sendiri.
3. On exit / action
On exit / action adalah aktivitas yang dilakukan setelah keluar dari sebuah state.
4. History
History digambarkan dengan inisial \textcircled{H} , history dibuat dengan tujuan agar setiap data penjualan terdapat database seperti arsip, sehingga data yang ada tidak terus bertumpuk.
5. Self call
Self call yaitu aktivitas yang dilakukan pada state yang bertujuan sebagai proses dalam menjalankan aktivitas pada state itu sendiri.
6. Event
Event adalah pesan atau perintah kepada state.

5. Package diagram



Pada package diagram, setiap package memiliki ketergantungan terhadap package yang lainnya. Package diagram sangat berguna untuk desain sistem yang berskala besar supaya ketergantungan antar package dapat terlihat dengan jelas.

6. Activity diagram



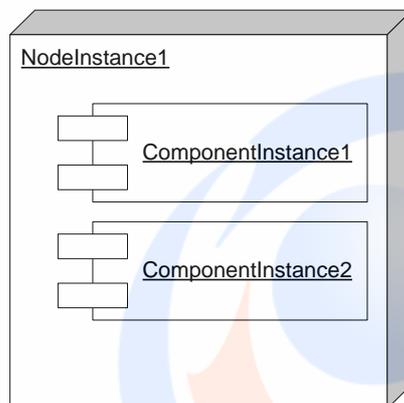
Pada activity diagram, apabila actor akan melakukan 2 aktivitas yang berbeda setelah melakukan sebuah aktivitas, maka diperlukan fork. Fork merupakan percabangan aktivitas menjadi dua. Sedangkan apabila actor setelah melakukan 2 aktivitas berbeda dibutuhkan join untuk melakukan satu aktivitas selanjutnya. Join merupakan penggabungan dua aktivitas berbeda menjadi satu.

7. Component diagram



Pada component diagram, component yang disajikan disesuaikan dengan program aplikasi yang dipakai dalam pembuatan sistem informasi.

8. Deployment diagram



Deployment diagram bertujuan untuk menggambarkan tampilan fisik dari sistem. Deployment diagram dibuat berdasarkan component diagram sebelumnya. Komponen yang terdapat pada deployment diagram harus menjadi satu kesatuan yang tidak boleh dipisahkan dan harus tetap berada dalam satu folder atau hard disk user, jika tidak maka program yang dibuat tidak akan berfungsi sama sekali.

9. Communication diagram



Communication diagram menggambarkan hubungan antar objek seperti yang digambarkan dalam sequence diagram.

Nama Objek

Proses CRUD

KELAS / ATRIBUT	Mutasi	Mutasi detail	Item	Supplier	Video cd & cd ps	Accessoris game	Kategori
Tgl_mutasi	<i>C</i>	--	--	--	--	--	--
Kode_mutasi	<i>R</i>	--	--	--	--	--	--
Jenis_mutasi	<i>R</i>	--	--	--	--	--	--
Kode_item	--	--	<i>CRUD</i>	--	--	--	--
Jenis_item	--	<i>R</i>	<i>CRUD</i>	--	--	--	--
Kode_supplier	--	--	<i>R</i>	<i>CRUD</i>	--	--	--
Nama_supplier	--	<i>R</i>	<i>R</i>	<i>CRUD</i>	--	--	--
Kode_kategori	--	--	--	--	--	--	<i>CRUD</i>
Jenis_kategori	--	<i>R</i>	--	--	<i>R</i>	<i>R</i>	<i>CRUD</i>
Kode_unit	--	<i>R</i>	--	--	<i>CRUD</i>	<i>CRUD</i>	--
Title	--	<i>R</i>	--	--	<i>CRUD</i>	<i>CRUD</i>	--
Jumlah_unit	--	<i>R</i>	--	--	<i>CRUD</i>	<i>CRUD</i>	--
Harga_satuan	--	<i>R</i>	--	--	<i>CRUD</i>	<i>CRUD</i>	--

Ket : C (create), R (read), U (update), D (delete).