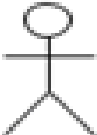







DAFTAR SIMBOL




- Daftar Simbol Pada *Use Case Diagram*




No	Simbol	Nama	Keterangan
1.		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
2.		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang bergantung pada elemen yang tidak mandiri (<i>independent</i>).
3.		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada diatas objek induk (<i>ancestor</i>).
4		<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> adalah sumber secara eksplisit.
5		<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
6		<i>Associaton</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.

7		<i>System</i>	Menspesifikasian paket yang menampilkan sistem secara terbatas.
8		<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
9		<i>Collaboration</i>	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		<i>Note</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

(Sumber : Modul *Workshop UML* Bab 2)

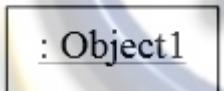



• **Daftar Simbol Pada *Activity Diagram***

No	Simbol	Nama	Keterangan
1.		<i>Initial</i>	Titik awal, untuk memulai suatu aktivitas.
2.		<i>Dependency</i>	Titik akhir, untuk mengakhiri aktivitas.
3.		<i>Activity</i>	Menandakan sebuah aktivitas.

4		<i>Decision</i>	Pilihan untuk mengambil keputusan.
5		<i>Fork / Join</i>	Menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu.
6		<i>Rake</i>	Menunjukkan adanya dekomposisi.

(Sumber : *User Guide Enterprise Architect 7.0*, Modul Workshop UML Bab 2)

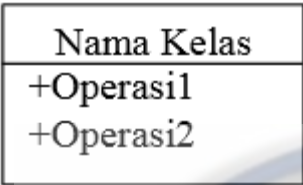
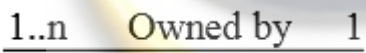

• **Daftar Simbol *Sequence Diagram***

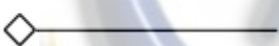
No	Simbol	Nama	Keterangan
1.		<i>Object</i> (Partisipan)	Merupakan instance dari sebuah class dan dituliskan tersusun secara horizontal.
2.		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
3.		<i>Lifeline</i>	Mengindikasikan keberadaan sebuah object dalam basis waktu.
4		<i>Activation</i>	Mengindikasikan sebuah objek yang akan melakukan sebuah aksi.

5		<i>Boundary</i>	Terletak diantara sistem dengan duni sekelilingnya. Semua form, laporan, antarmuka ke perangkat keras seperti printer atau scanner dan antarmuka ke sistem lainnya adalah termasuk dalam kategori.
6		<i>Control</i>	Berhubungan dengan fungsionalitas seperti pemanfaatan sumber daya, pemrosesan distribusi, atau penanganan kesalahan.
7		<i>Entity</i>	Digunakan untuk menangani informasi yang mungkin akan disimpan secara permanen.
8		<i>Message</i>	Mengindikasikan komunikasi antara object.
9		<i>Self-Message</i>	Mengindikasikan komunikasi kembali kedalam sebuah objek itu sendiri.
10		<i>Loop</i>	Mengeksekusi berulang kali dan penjaga menunjukkan dasar iterasi.

(Sumber : *Martin Fowler UML Distilled* 2005, Sholiq Pemodelan Sistem Informasi Berorientasi Objek Dengan UML 2006, Modul Workshop UML Bab 2)

• **Daftar Simbol *Class Diagram***

No	Simbol	Nama	Keterangan
1.		<i>Class</i>	<p>Blok pembangun pada pemrograman berorientasi objek.</p> <p>Bagian atas adalah bagian dari class. Bagian tengah mendefinisikan property/atribut class. Bagian akhir mendefinisikan method-method dari sebuah class.</p>
2.		<i>Assosiation</i>	<p>Relationship paling umum antara 2 class, dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 class. Garis ini bisa melambangkan tipe-tipe relationship dan juga dapat menampilkan hukum-hukum multiplisitas pada sebuah relationship.</p>
3.		<i>Composition</i>	<p>Jika sebuah class tidak bisa berdiri sendiri dan harus merupakan bagian dari class yang lain,</p>

			maka class tersebut memiliki relasi Composition terhadap class tempat bergantung tersebut.
4		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang bergantung pada elemen yang tidak mandiri (<i>independent</i>).
5		<i>Aggregation</i>	Menindikasikan keseluruhan bagian relationship dan biasanya disebut sebagai relasi.
6		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada diatas objek induk (<i>ancestor</i>).

(Sumber : Modul *Workshop UML* Bab 2)