



## **TUGAS II REKAYASA PERANGKAT LUNAK**

*Software Requirements Analysis and Specification*

Disusun oleh:

Aditya Tri Ananda                      09021181924019

Ferza Reyaldi                          09021281924060

**UNIVERSITAS SRIWIJAYA**

**INDRALAYA**

**2021**

## Software Quality Attributes

- **Correctness**  
Sejauh mana sebuah program memenuhi spesifikasinya dan tujuan dari pengguna.
- **Reliability**  
Sejauh mana sebuah program dapat diharapkan untuk melaksanakan fungsi yang diinginkan sesuai presisi yang diharapkan.
- **Efficiency**  
Jumlah sumber daya komputasi yang dibutuhkan oleh program untuk melaksanakan suatu fungsi.
- **Integrity**  
Sejauh mana akses ke perangkat lunak atau data oleh orang yang tidak berwenang dapat dikontrol.
- **Usability**  
Usaha yang diperlukan untuk mempelajari, mengoperasikan, mempersiapkan masukan, dan menginterpretasikan keluaran dari suatu program.
- **Survivability**  
Mengacu pada kontinuitas layanan. Program yang dibangun harus ditentukan waktu minimal yang diizinkan antara kegagalan sistem dan waktu maksimal yang diizinkan untuk pemulihan layanan program.
- **Maintainability**  
Usaha yang diperlukan untuk menemukan dan memperbaiki galat (*error*) dalam operasional program.
- **Verifiability**  
Menentukan fitur desain dan pemrograman yang memungkinkan verifikasi yang efisien.
- **Flexibility**  
Usaha yang diperlukan untuk memodifikasi operasional program.
- **Portability**  
Usaha yang diperlukan untuk mentransfer program dari suatu lingkungan sistem perangkat lunak ke lingkungan sistem perangkat lunak yang lain.
- **Reusability**  
Sejauh mana program dapat digunakan dalam aplikasi lain yang terkait dengan pengemasan dan ruang lingkup dari fungsi yang program jalankan.
- **Interoperability**

Usaha yang diperlukan dalam penggantian suatu sistem dengan sistem yang lain.

- **Expandability**

Mengacu pada usaha di masa depan yang dibutuhkan untuk melayani populasi-populasi yang lebih besar, meningkatkan layanan, atau menambah aplikasi baru yang bertujuan untuk meningkatkan *usability*.

## **Usecase Diagram**

*Usecase diagram* ini diperlukan sebagai langkah awal dalam menentukan kebutuhan fungsional. Kebutuhan fungsional adalah kebutuhan pengguna dan stakeholder dalam kesehariannya. Artinya *Usecase Diagram* ini berguna sebagai penggambaran kelakuan sistem yang akan dibuat nantinya.

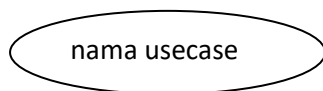
Didalam *UsecaseDiagram* haruslah mendeskripsikan sebuah interaksi antara satu atau lebih orang, proses, ataupun sistem lain dengan sistem informasi yang akan dibuat. Dengan begitu, *Usecase Diagram* digunakan sebagai informasi fungsi apa saja yang ada didalam sebuah sistem dan untuk siapa fungsi-fungsi tersebut seharusnya digunakan.

Hal yang perlu ditekankan pada *Usecase Diagram* adalah *Usecase Diagram* tidak menggambarkan *user interface*, arsitektur dari sistem, serta kebutuhan nonfungsional melainkan menggambarkan interaksi saja antara orang, proses atau sistem lain (aktor) dengan sistem yang akan dibuat.

Dalam *Usecase Diagram* terdapat beberapa simbol :

1. *Usecase*

Fungsionalitas yang disediakan sistem. Biasanya dinyatakan dengan menggunakan kata kerja di awal frasa.



2. Aktor

Orang, proses, atau sistem lain yang berinteraksi dengan sistem yang telah dibuat. Biasanya dinyatakan menggunakan kata benda di awal frasa nama aktor.



3. Asosiasi

Komunikasi antara aktor dan *Usecase*.



#### 4. *Extend*

Relasi *Usecase* tambahan dimana *Usecase* yang ditambahkan dapat berdiri sendiri.

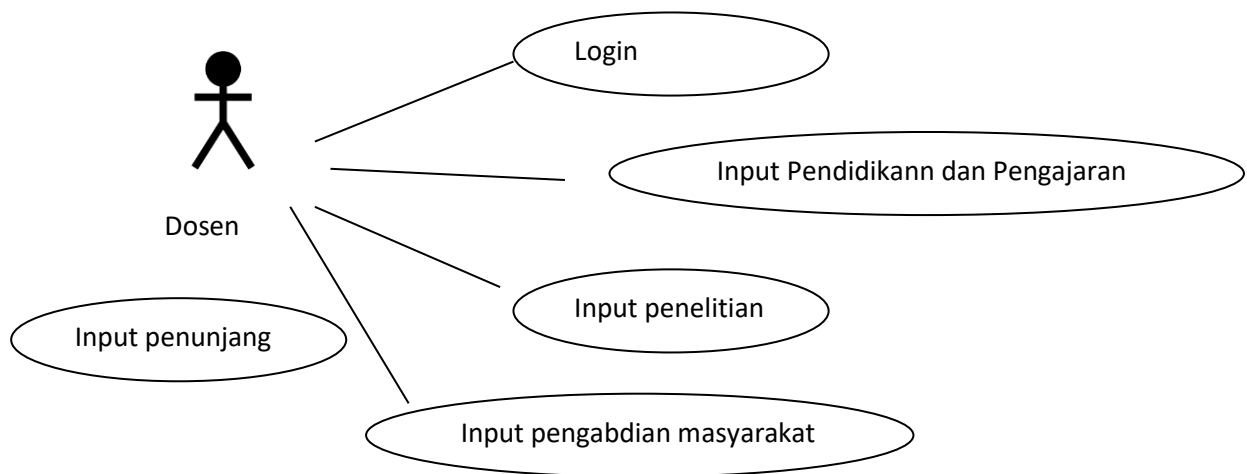
←-----  
<<extend>>

#### 5. *Include*

Relasi *Usecase* tambahan dimana *Usecase* yang ditambahkan tidak dapat berdiri sendiri.

<<include>>  
----->

Contoh *Usecase Diagram* Sederhana



### Data Flow Diagram (DFD)

*Data Flow Diagram* atau biasa disebut DFD merupakan diagram untuk merepresentasikan aliran data secara grafis dalam sistem informasi. Menurut Bruza (1997), DFD sering digunakan dalam tahap desain awal untuk memberikan gambaran umum dari suatu sistem. DFD secara grafis merepresentasikan fungsi, atau proses, yang menangkap, memanipulasi, menyimpan, dan mendistribusikan data antara suatu sistem dan lingkungannya dan antar komponen suatu sistem.

DFD dapat dibagi menjadi 2 macam, yaitu: *Logical Data Flow Diagram* dan *Physical Data Flow Diagram*.


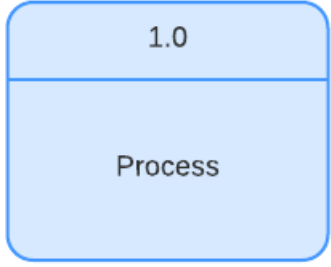
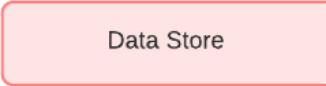
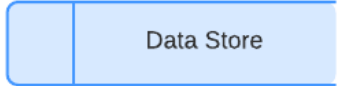

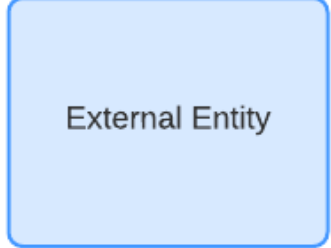


- *Logical DFD* berfokus pada bisnis dan bagaimana bisnis beroperasi. Tidak peduli bagaimana sistem akan dibangun. Kita dapat mengabaikan spesifikasi implementasi seperti, konfigurasi komputer, teknologi penyimpanan data, komunikasi atau metode penyampaian pesan dengan berfokus pada fungsi yang dilakukan oleh sistem, seperti, pengumpulan data, transformasi data ke informasi dan pelaporan informasi.
- *Physical DFD* menunjukkan bagaimana sistem akan diimplementasikan, termasuk perangkat keras, perangkat lunak, file, dan orang-orang dalam sistem. Ini dikembangkan sedemikian rupa sehingga proses yang dijelaskan dalam *Logical DFD* diimplementasikan dengan benar untuk mencapai tujuan bisnis.

Pada dasarnya, terdapat 2 jenis notasi DFD, yaitu:

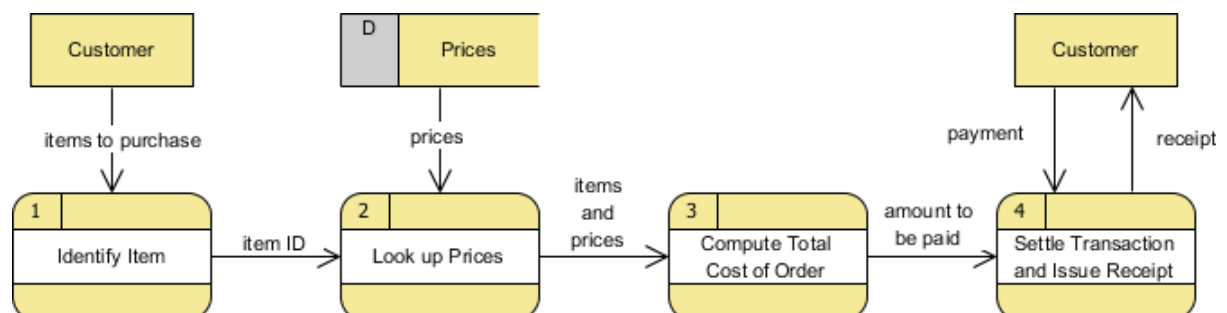
- Yourdon & Coad, tipe yang biasa digunakan untuk desain dan analisis sistem;
- Gane & Sarson, tipe yang digunakan untuk visualisasi sistem informasi secara umum.

Namun, ide dasar dari kedua notasi DFD sama. Terdapat 4 elemen dasar dari DFD yaitu sebagai berikut.

Tabel Elemen-Elemen *Data Flow Diagram*

Elemen	Yourdon & Coad	Gane & Sarson
<b>1. <i>Process</i></b> Suatu proses menerima masukan dan mengubahnya menjadi keluaran yang memiliki bentuk yang berbeda dari masukan. Setiap <i>process</i> diberi nama dengan <b>kata kerja</b> yang mengidentifikasi fungsi yang dijalankan.		
<b>2. <i>Data Store</i></b> Tempat penyimpanan data dalam suatu sistem. Pada <i>data store</i> tidak melakukan operasi apapun tetapi hanya data yang disimpan untuk diakses nanti.		
<b>3. <i>External Entity</i></b> Objek di luar sistem yang berkomunikasi dengan sistem. <i>External Entity</i> merupakan sumber dan tujuan dari masukan dan keluaran sistem.		
<b>4. <i>Data Flow</i></b> Jalur yang dilalui oleh paket arus informasi antara <i>process</i> , <i>data store</i> , dan <i>external entity</i> .		

Contoh penggunaan DFD sederhana:



## Data Dictionary

*Data Dictionary* atau bisa disebut sebagai Kamus Data merupakan kumpulan nama, definisi, dan atribut untuk elemen dan model data. Elemen-elemen ini kemudian digunakan sebagai bagian dari database, proyek penelitian, ataupun sistem informasi.

Beberapa elemen yang biasanya ada didalam di Kamus Data yaitu nama atribut, tipe atribut, ER, data refrensi, peraturan untuk validasi, skema ataupun kualitas data, rincian perlengkapan elemen data, dan informasi tentang dimana data disimpan.

Setidaknya ada 2 tipe dari Kamus Data yaitu aktif dan pasif. Kamus Data aktif ialah terikat ke database tertentu yang membuat pemindahan data menjadi sebuah tantangan tersendiri, tetapi diperbarui secara otomatis dengan sistem manajemen data. Sedangkan Kamus Data pasif ialah tidak terikat dengan database atau server tertentu, tetapi juga harus dijaga secara manual untuk mencegah metadata tidak sinkron.

Pentingnya sebuah Kamus Data :

1. untuk mendokumentasikan dan berbagi struktur data dan informasi lain untuk semua yang terlibat dengan proyek atau *database*.
2. menentukan konvensi untuk proyek dan konsistensi di seluruh kumpulan data
3. Mencegah risiko kehilangan informasi penting dalam terjemahan dan transisi
4. membantu tim menganalisis data dengan lebih mudah di lain waktu.

Untuk membuat kamus data aktif bisa menggunakan sistem manajemen basis data (DBMS) seperti SQL, Server, Oracle, atau mySQL. Untuk membuat kamus data pasif, perlu membuatnya secara terpisah dari DBMS karena kamus pasif tidak dikelola oleh sistem manajemen.

Pada tahap perencanaan sistem, kamus data digunakan untuk merancang input, merancang laporan-laporan dan *database*.

Tabel Simbol-Symbol Kamus Data

Simbol	Deskripsi
=	Terdiri dari, terbentuk dari, sama dengan
+	Penggabungan elemen data dengan elemen data lain
{ }	Pengulangan elemen data
[ ] dan	Memilih salah satu dari beberapa alternatif
( )	Data tambahan, boleh ada boleh tidak
*...*	Penjelasan atau keterangan tentang suatu data

Sumber : (Rosa, 2013)

## Sumber Pustaka

1. K. Henningsson and C. Wohlin, "Understanding the Relations between Software Quality Attributes - A Survey Approach", Proceedings 12th International Conference for Software Quality, Ottawa, Canada, Proceedings on CD, October 2002.
2. Sugiarti, Yuni. 2018. *Dasar-Dasar Pemrograman Java Netbeans Database, UML, dan Interface*. Bandung. PT Remaja Rosdakarya.
3. <https://www.visual-paradigm.com/guide/data-flow-diagram/what-is-data-flow-diagram/>
4. <https://www.lucidchart.com/pages/data-flow-diagram>
5. <https://www.smartdraw.com/data-flow-diagram/>
6. <https://www.trifacta.com/blog/data-dictionary/>