

BAB II

LANDASAN TEORI

2.1 Sistem

Terdapat dua kelompok pendekatan di dalam mendefinisikan sistem, yaitu yang menekankan pada prosedurnya dan yang menekankan pada komponen atau elemennya. Pendekatan sistem adalah *“Suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan satu sasaran tertentu”*[9] (Tata Sutabri, 2014).

Pendekatan sistem yang merupakan jaringan kerja dari prosedur lebih menekankan urutan-urutan operasi didalam sistem. Sedangkan pendekatan sistem yang lebih menekankan pada elemen atau komponennya, menurut Gordon B.Davis sebagai berikut : *“Sistem adalah kumpulan dari elemen yang berinteraksi untuk mencapai suatu tujuan tertentu”*[9] (Tata Sutabri, 2014).

2.2 Informasi

Informasi merupakan proses lebih lanjut dari data yang sudah memiliki nilai tambah. Informasi dapat dikelompokkan menjadi 3 bagian, yaitu:

1. Informasi Strategis. Informasi ini digunakan untuk mengambil keputusan jangka panjang, yang mencakup informasi eksternal, rencana perluasan perusahaan, dan sebagainya.
2. Informasi Taktis. Informasi ini dibutuhkan untuk mengambil keputusan jangka menengah, seperti informasi tren penjualan yang dapat dimanfaatkan untuk menyusun rencana penjualan.
3. Informasi Teknis. Informasi ini dibutuhkan untuk keperluan operasional sehari-hari, seperti informasi persediaan stock, retur penjualan, dan laporan kas harian. Informasi adalah data yang telah diklasifikasikan atau diolah untuk digunakan dalam proses pengambilan keputusan[9] (Tata Sutabri, 2014).

2.3 Sistem Informasi

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan[9]. (Tata Sutabri, 2014).

2.4 Basis Data

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. Sistem informasi tidak dapat dipisahkan dengan kebutuhan akan basis data apapun bentuknya, entah berupa *file* teks ataupun *Data Management System (DBMS)* [8] (Rosa A.S, M.Shalahuddin,2016).

Kebutuhan basis data dalam sistem informasi meliputi:

- 1.Memasukkan, menyimpan, dan mengambil data
- 2.Membuat laporan berdasarkan data yang telah disimpan

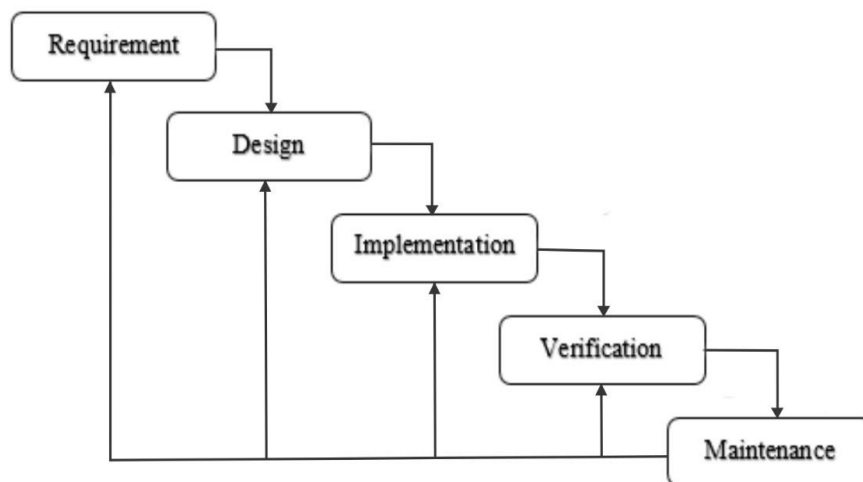
Tujuan dari dibuatnya table-tabel disini adalah untuk menyimpan data ke dalam tabel-tabel agar mudah diakses. Oleh karena itu, untuk merancang tabel-tabel yang akan dibuat maka dibutuhkan pola pikir penyimpanan data nantinya jika dalam bentuk baris-baris data (*record*) dimana setiap baris terdiri dari beberapa kolom, [8] (Rosa A.S, M.Shalahuddin, 2016).

2.5 Metode Pengembangan Sistem

Metode pengembangan sistem sangat dibutuhkan dalam perancangan sebuah sistem karena sebelum memulai pembuatan koding – koding hendaknya merancang terlebih dahulu metode pemodelan seperti apa yang harus digunakan dengan memprioritaskan ketepatan waktu selesai dan efektifitas dalam perancangan sebuah sistem.

(Rosa A.S, M.Shalahuddin, 2016) Model air terjun (*waterfall*) sering juga disebut model sekuensial linier (*sequential linear*) atau alur hidup klasik (*classic life cycle*). Model *waterfall* menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengodean, pengujian, dan tahap pendukung (*support*) [8].

Berikut adalah gambar model *waterfall*:



Gambar 2.1 model *waterfall*

1. *Requirement system*

Proses pengumpulan kebutuhan dilakukan secara intensif untuk memesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

2. *System design*

Desain perangkat lunak adalah proses multi langkah yang focus pada desain pembuatan program perangkat lunak, representasi antarmuka, dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain

perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

3. *Implementation*

Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain.

4. *Integration & testing*

Pengujian focus pada perangkat lunak secara dari logik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

5. *Operation and Maintenance*

Dalam tahapan ini, sistem diinstal dan mulai digunakan. Selain itu juga memperbaiki *error* yang tidak ditemukan pada tahap pembuatan. Dalam tahap ini juga dilakukan pengembangan sistem seperti penambahan fitur dan fungsi baru.

2.6 *Unified Modeling Language (UML)*

Bahasa pemrograman berorientasi objek yang pertama dikembangkan dikenal dengan nama Simula-67 yang dikembangkan pada tahun 1967. Bahasa pemrograman ini kurang berkembang dan dikembangkan lebih lanjut, namun dengan kemunculannya telah memberikan sumbangan yang sebesar pada developer pengembangan bahasa pemrograman berorientasi objek selanjutnya[8], (Rosa A.S, M.Shalahuddin, 2016).

Secara fisik, UML adalah sekumpulan spesifikasi yang dikeluarkan oleh OMG. UML terbaru adalah UML 2.3 yang terdiri dari 4 macam spesifikasi, yaitu *Diagram Interchange Specification*, *UML Infrastructure*, *UML Superstructure*, dan *Object Constraint Language (OCL)*. Seluruh

spesifikasic tersebut dapat diakses diwebsite <http://www.omg.org>[8], (Rosa A.S, M.Shalahuddin, 2016).

2.6.1 Diagram UML

Untuk mendapatkan banyak pandangan terhadap sistem informasi yang akan dibangun, UML menyediakan beberapa diagram visual yang menunjukkan berbagai aspek dalam sistem. Ada beberapa diagram yang disediakan dalam UML antara lain :

1. Use Case Diagram




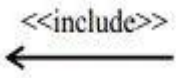
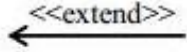
Use case atau diagram *Use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendefinisikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu[8], (Rosa A.S, M.Shalahuddin, 2016).

syarat penamaan pada use care adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada use case yaitu pendefinisian apa yang disebut aktor dan *use case*.

1. Aktor merupakan orang, proses, atau sistem laen yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar kesan antar unit atau aktor.

Berikut adalah tabel simbol-simbol yang ada diagram *use case*:

Tabel 2.1 Simbol *Use Case Diagram*

No	Simbol	Nama	Keterangan
1		Actor	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan use case.
2		Use case	Deskripsi dari urutan aksi – aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor.
3		Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.
4		Include	Menspesifikasikan bahwa use case sumber secara eksplisit.
5		Extend	Menspesifikasikan bahwa use case target memperluas perilaku dari use case sumber pada suatu titik yang diberikan.

2. Activity Diagram


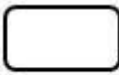

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh system[8], (Rosa A.S, M.Shalahuddin, 2016).

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut:




1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem / *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antar muka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak.

Berikut adalah tabel simbol-simbol yang ada pada diagram aktivitas:

Tabel 2.2 Simbol *Activity Diagram*

No	Simbol	Nama	Keterangan
1		Swimlane	Menunjukkan siapa yang bertanggung jawab dalam melakukan aktivitas dalam suatu diagram.
2		Action	Langkah-langkah dalam sebuah activity. Action bias terjadi saat memasuki activity, meninggalkan activity, atau pada event yang spesifik.
3		Initial State	Menunjukkan dimana aliran kerja dimulai.

Tabel 2.3 Simbol *Activity Diagram* (lanjutan)

4		Activity Final Node	Menunjukkan dimana aliran kerja diakhiri.
5		Decision Node	Menunjukkan suatu keputusan yang mempunyai satu atau lebih transisi sesuai dengan suatu kondisi.
6		Control Flow	Menunjukkan bagaimana kendali suatu aktivitas terjadi pada aliran kerja dalam tindakan tertentu.


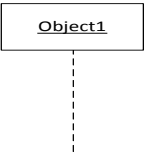

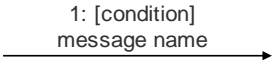
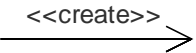
3. Sequence Diagram

Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendefinisikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstalasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*.

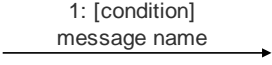
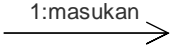
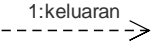
Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak[8], (Rosa A.S, M.Shalahuddin, 2016).

Berikutnya adalah table simbol-simbol yang ada pada Sequence Diagram:

Tabel 2.4 Simbol *Sequence Diagram*

Simbol	Deskripsi
<p>Aktor</p> 	<p>Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang dibuat itu sendiri</p>
<p>Objek</p> 	<p>Menyatakan objek yang berinteraksi pesan.</p>
<p>Waktu aktif</p> 	<p>Menyatakan objek dalam keadaan aktif dan berinteraksi pesan.</p>
<p>Pesan tipe <i>call</i></p> 	<p>Arah panah mengarah pada objek yang memiliki operasi atau metode karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.</p>
<p>Pesan tipe <i>create</i></p> 	<p>Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.</p>

Tabel 2.5 Simbol *Sequence Diagram* (lanjutan)

<p>Pesan tipe <i>call</i></p> 	<p>Arah panah mengarah pada objek yang memiliki operasi atau metode karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.</p>
<p>Pesan tipe <i>send</i></p> 	<p>Menyatakan bahwa suatu objek mengirimkan data /masukan / informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim.</p>
<p>Pesan tipe <i>return</i></p> 	<p>Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode yang menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.</p>

4. Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi, (Rosa A.S, M.Shalahuddin, 2016).

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.
2. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

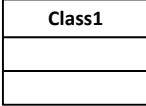
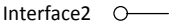



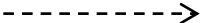

Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan didalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. Banyak berbagai kasus, perancangan kelas yang dibuat tidak sesuai dengan kelas-kelas yang dibuat pada perangkat lunak, sehingga tidaklah ada gunanya lagi sebuah perancangan karena apa yang dirancangan dan hasil jadinya tidak sesuai.

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem sehingga pembuat perangkat lunak atau *programmer* dapat membuat kelas-kelas didalam program perangkat lunak sesuai dengan perancangan diagram kelas. Susunan struktur kelas yang baik pada digram kelas sebaiknya memiliki jenis-jenis kelas berikut:

1. Kelas main
Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.
2. Kelas yang menangani tampilan sistem (*view*)
Kelas yang mendefinisikan dan mengatur tampilan ke pemakai.
3. Kelas yang diambil dari pendefinisian *use case* (*controller*)
Kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use care*, kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.
4. Kelas yang diambil dari pendefinisian data (*model*)
Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data. Semua tabel yang dibuat dibasis data dapat dijadikan kelas, namun untuk tabel dari hasil relasi atau atribut mutivalue pada ERD dapat dijadikan kelas tersendiri

dapat juga tidak asalkan pengaksesannya dapat dipertanggung jawabkan atau tetap ada didalam perancangan kelas.

Tabel 2.6 Simbol *Class Diagram*

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem.
Natarmuka/ <i>interface</i> 	Sama dengan konsep <i>interface</i> dalam pemograman berorientasi objek.
Asosiasi 	Relasi antar kelas dalam makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
Asosiasi berarah 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus).
Kebergantungan 	Relasi antar kelas dengan makna kebergantungan antar kelas.
Agregasi 	Relasi antar kelas dengan maknasemua bagian (<i>whole-part</i>).

2.7 Sejarah

Sejarah adalah rekrontuksi masa lalu yang telah dipikirkan, dikatakan, dikerjakan, dirasakan dan dialami oleh seseorang. Ilmu sejarah mencoba

untuk membangun kepastian dan objektivitas yang berdasarkan pada analisis dan klarifikasi mengenai tingkah laku manusia sehingga dapat diterima oleh akal budi sehingga struktur peristiwa yang kompleks dapat dimengerti untuk kemajuan dimasa sekarang[7]. (Novita Mujiyati, 2016)

2.8 Android

Menurut (Nadia Firly, 2018) Android adalah sistem operasi yang berfungsi sebagai penghubung (*device*) antara pengguna dan perangkat keras pada *smarthphone* atau alat elektronik tertentu. Sehingga, hal tersebut memungkinkan pengguna dapat berinteraksi dengan *device* dan menjalankan berbagai macam aplikasi *mobile*[6].

Android adalah sebuah perangkat lunak untuk perangkat *mobile* berbasis *linux*, Android sendiri ini menggunakan bahasa pemrograman *java*[1]. (Alanus Raro Rega, 2017)

2.8.1 Android SDK

Menurut (Nadia Firly, 2018) Android SDK adalah merupakan sebuah kit yang berfungsi untuk mengembangkan berbagai aplikasi berbasis Android oleh para *developer*. Didalam SDK telah terdapat berbagai tools yang bertujuan untuk proses pengembangan aplikasi seperti proses *debugger*, *emulator*, *software libraries*, dan dokumentasi[6].

2.9 PHP (*Hypertext Preprocessor*)

Menurut (Jubilee Enterprise, 2018) PHP merupakan Bahasa pemrograman yang digunakan untuk membuat website dinamis dan interaktif. Dinamis artinya, website tersebut bisa berubah-ubah tampilan dan kontennya sesuai kondisi tertentu. Sebagai contoh, PHP bisa menampilkan tanggal dan hari saat ini secara berganti-ganti di dalam sebuah website. Interaktif artinya, PHP dapat memberi *feedback* bagi *user* (misalnya menampilkan hasil pencarian produk) [4].

2.10 Java

Java menurut definisi dari *Sun Microsystem* adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *standalone* ataupun pada lingkungan jaringan. *Java 2* adalah generasi kedua dari *Java Platform*. *Java* berdiri atas sebuah mesin *interpreter* yang diberi nama *Java Virtual Machine (JVM)*. *JVM* inilah yang akan membaca *bytecode* dalam *file .class* dari suatu program sebagai representasi langsung program yang berisi bahasa mesin. Oleh karena itu bahasa *Java* disebut sebagai bahasa pemrograman yang *portable* karena dapat dijalankan pada berbagai sistem operasi, asalkan pada sistem operasi tersebut terdapat *JVM*. *Java* merupakan bahasa pemrograman objek murni karena semua kode programnya dibungkus dalam kelas. Saat ini *Sun Microsystem* sudah diakuisisi *Oracle Corporation* sehingga pengembangan *Java* diteruskan oleh *Oracle Corporation* [8] (Rosa A.S, M.Shalahuddin, 2016).

2.11 MySQL

MySQL adalah sebuah program *database server* yang mampu menerima dan mengirimkan datanya dengan sangat cepat, *multi user* serta menggunakan perintah standar SQL (*Structured Query Language*). *MySQL* juga telah mendukung bahasa pemrograman berfitur API seperti *Java* sehingga memudahkan para programmer *java* untuk berkoneksi dengan menggunakan *MySQL* [8]. (Rosa A.S, M.Shalahuddin, 2016).

2.12 GIS (*Geographic Information Sistem*)

SIG merupakan sebuah sistem informasi yang didesain untuk bekerja dengan sumber data spasial. *SIG* merupakan suatu media yang sangat handal untuk mempresentasikan data *remote Sensing (RS)* menjadi informasi yang berguna bagi banyak pihak untuk berbagai keperluan. Data spasial adalah data-data yang memiliki sistem koordinat geografis. Dengan kata lain *SIG* merupakan suatu sistem database yang memiliki kemampuan

spesifik untuk melakukan operasi tertentu pada data. Teknologi SIG biasanya telah terintegrasi dengan teknologi database seperti *query* dan analisa statistic dengan tampilan yang unik, serta analisis geografis yang menguntungkan dengan peta. Kemampuan ini yang membuat SIG berbeda dengan sistem informasi lainnya, sehingga SIG menjadi barang berharga bagi masyarakat umum dan perusahaan untuk menjelaskan peristiwa, memprediksi pendapatan dan perencanaan strategis. SIG meliputi pemahaman tentang: pola, ruang, dan proses (metodologi) yang dibutuhkan untuk penyelesaian masalah. Selain obyektif, SIG juga dapat dijadikan sebagai alat tindakan yang sangat cepat dan efisien. SIG akan sangat berguna bila pemakai memahami konsep spasial secara utuh [3] (Indarto, 2013).