BAB 2

LANDASAN TEORI

2.1 Teori Umum

Teori umum secara garis besar berisi tentang teori-teori umum dan teori-teori dasar yang ikut berperan dalam perancangan sistem, meliputi informasi berkenaan dengan *E-Commerce*, *Internet*, *World Wide Web*, *Dreamweaver*, PHP, HTML, MySQL, JQuery, Database, DBMS, *Entity Relationship Diagram* (ERD), Interaksi Manusia dan Komputer (IMK), dan Rekayasa Piranti Lunak (RPL).

2.1.1 Pengertian E-Commerce

Menurut Turban, Rainer, Potter (2006, p181-182) Electronic *commerce*, disingkat sebagai Ecm atau *e-commerce* adalah suatu proses pembelian, penjualan, transfer, atau pertukaran produk, jasa atau informasi melalui jaringan komputer, termasuk internet. *E-commerce* dapat berupa berbagai bentuk tergantung pada tingkat digitalisasi-transformasi dari fisik ke digital yang dilibatkan. Tingkat digitalisasi dapat berhubungan dengan produk (jasa) yang dijual, proses, atau pelaku pengirimnya (perantaranya).

Transaksi e-commerce dapat dilakukan antara berbagai pihak. Menurut Turban, Rainer, Potter (2006, p183) Jenis umum dari transaksi e-commerce dijelaskan dibawah ini:

- Bisnis ke bisnis (*business-to-business*-B2B): Dalam transaksi B2B, baik penjual maupun pembeli adalah organisasi bisnis.

- Perdagangan kolaboratif (*collaborative commerce-c-commerce*): Dalam c-commerce para mitra bisnis berkolaborasi (alih-alih membeli atau menjual) secara elektronik.
- Bisnis ke komsumen (*business-to-consumer-B2C*): Dalam B2C, penjual adalah perusahaan dan pembeli adalah perorangan. B2C juga disebut e-tailing.
- Konsumen ke konsumen (*consumer-to-consumer*-C2C): Dalam C2C, seseoarang menjual produk atau jasa ke orang lain. (anda juga dapat melihat istilah C2C digunakan sebagai "*consumer-to-consumer*" (pelanggan-ke-pelanggan).
- Konsumen-ke-bisnis (*consumer-to-business-C2B*): Dalam C2B konsumer memberitahukan kebutuhan atas suatu produk atau jasa tertentu, dan para pemasok bersaing untuk menyediakan produk atau jasa tersebut ke konsumen.
- Perdagangan intrabisnis (*intraorganisasional)*: Dalam situasi ini perusahaan menggunakan EC secara internal untuk memperbaiki operasinya.
- Pemerintah ke warga (*government-to-citizen*-G2C) dan ke pihak lain: Dalam kondisi ini sebuah entitas (unit) pemerintah menyediakan layanan ke para warganya melalui teknologi EC.
- Perdagangan mobile (*mobile commerce-m-commerce*): Ketika e-commerce dilakukan dalam lingkungan nirkabel, seperti dengan menggunakan telepon seluler untuk mengakses internet dan berbelanja, maka hal ini disebut m-mobile.

2.1.2 Pengertian Interconnected-Networking (Internet)

Menurut Turban, Rainer, Potter (2006, p674) *Internet* tumbuh dari proyek eksperimental pada *Advanced Research Project Agency* (ARPA) dari Departement Pertahanan Amerika Serikat. Proyek ini dimulai pada tahun 1969, dengan nama *ARPAnet* , untuk menguji kelayakan jaringan komputer area luas dimana peneliti,

pendidik, militer, dan lemabaga pemerintahan dapat saling berbagi data, saling bertukar pesan, dan mentransfer *file.* Kini *ARPAnet* telah mengubah namanya menjadi *Internet* pada tahun 1993.

Turban, Rainer, Potter (2006, p674) menyimpulkan bahwa *Internet* ("*the Net*") adalah jaringan yang menghubungkan sekitar satu juta jaringan komputer organisasional internasional dilebih dari 200 negara disemua benua, termasuk Antarika. Sedangkan secara harfiah, internet (kependekan dari interconnected-networking) ialah rangkaian komputer yang terhubung didalam beberapa rangkaian.

2.1.3 World Wide Web (WWW)

Menurut Turban, Rainer, dan Potter (2006, p680) *World Wide Web* adalah sistem dengan standart yang diterima secara universal untuk menyimpan, menelusuri, memformat, dan menampilkan informasi melalui arsitektur klien/server, menggunakan fungsi-fungsi transport dari internet. Teknologi *World Wide Web* diciptakan oleh Timothy Berners-Lee, yang pada tahun 1989 mengusulkan jaringan global dari dokument hiperteks yang akan memungkinkan para peneliti fisika bekerjasama.

2.1.4 Hypertext Markup Language (HTML)

Menurut Turban, Rainer, dan Potter (2006, p680) *Hypertext Markup Language (HTML)* adalah bahasa pemograman yang digunakan di *web,* yang memformat dokumen dan memadukan link hiperteks dinamis ke dokumen-dokumen lainnya yang disimpan didalam komputer. HTML berasal dari *Standard Generalized Markup Language (SGML)* yang lebih rumit, bahasa berbasis teks untuk mendeskripsikan isi dan struktur dari dokument digital. HTML adalah subset yang lebih sederhana dari

SGML dan mencakup tabel, *applet*, aliran teks dalam gambar, superskrip, dan subskrip.

2.1.5 Adobe Dreamweaver

Menurut Madcoms (2008, p1) Dreamweaver adalah sebuah HTML editor profesional untuk mendesain web secara visual dengan mengelola situs atau halaman web. Kelebihan dreamweaver menurut Madcoms (2008, p15) diantaranya :

- Dapat digunakan pada dua sistem operasi, yaitu *Macintosh* atau *Windows*.
- Dapat dilihat tampilan *website preview* pada komputer, handphone/PDA, dan printer.
- Membangun dengan web CSS dengan menggunakan *CSS layout*, *CSS panel*, dan *CSS visualization*.
- Fasilitas lengkap pendukung CSS.
- Kemudahan pengelolaan data dengan *form* ataupun dengan *Spry* data object.

2.1.6 *Personal Home Page* (PHP)

Menurut Aditya Nur Alan (2010, p1) PHP (*hypertext Preprocessor*) adalah bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP banyak dipakai untuk memprogram situs web dinamis. PHP dapat digunakan untuk membangun sebuah CMS (*Content Management System)*.

Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (Situs Personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama *Form Interpreted* (FI), yang wujudnya berupa sekumpulan

skrip yang digunakan untuk mengolah data formulir dari *web*. Beberapa kelebihan PHP antara lain:

- Bahasa pemrograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaannya.
- *Web server* yang mendukung PHP dapat ditemukan dimana-mana dari mulai apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.
- Dalam sisi pengembangan lebih mudah, karena banyaknya milis-milis dan *developer* yang siap membantu dalam pengembangan.
- Dalam sisi pemahamannya, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
- PHP adalah bahasa *open source* yang dapat digunakan diberbagai mesim (*Linux, Unix, Macintosh, Windows*) dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem.

2.1.7 JQuery

Menurut Aloysius Sigit W (2011, p1) jquery adalah *library* atau kumpulan kode JavaScript siap pakai. Jquery pertama kali dirilis tahun 2006 oleh John Resig. Jquery menjadi sangat popular hingga telah digunakan pada banyak website kelas dunia seperti Google, Amazon, Twitter, ESPN, dan lain-lain. Kelebihan jquery diantaranya adalah sebagai berikut :

- Jquery kompatibel dengan banyak browser.
- Jquery mendukung semua versi CSS.
- Ukuran jquery sangat kecil, sekitar 20KB.

- Dokumentasi jquery yang lengkap.
- Dukungan komunitas terhadap jquery.
- Tersedianya *plugin* jquery yang sangat beragam.

2.1.8 My Structure Query Language (MySql)

Menurut Aditya Nur Alan (2010, p62) MySQL (*My Structure Query Language*) adalah sebuah implementasi dari sistem manajemen basis data relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public Licensi*). MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basis data yang telah ada sebelumnya yaitu SQL (*Structure Query Language*). SQL adalah sebuah konsep pengoperasian basis data, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. Kelebihan MySQL diantaranya adalah :

- Portabilitas: MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, dan masih banyak lagi.
- Perangkat lunak sumber terbuka: MySQL didistribusikan sebagai perangkat lunak sumber terbuka, dibawah lisensi GPL sehingga dapat digunakan secara gratis.
- *Multi-user*: MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah.
- *Performance tuning*: MySQL memeliki kecepatan yang menakjubkan dalam menangani *query* sederhana.
- Ragam tipe data: MySQL memiliki ragam tipe data yang sangat kaya, seperti *signed/usignied integer, float, double, char, text, date, timestamp,* dan lain-lain.

- Perintah dan fungsi: MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah *Select* dan *Where* dalam perintah (*query*).
- Keamanan: MySQL memiliki beberapa lapisan keamanan seperti level *subnetmask*, nama *host*, dan izin akses *user* dengan sistem perizinan yang mendetail.

2.1.9 Database

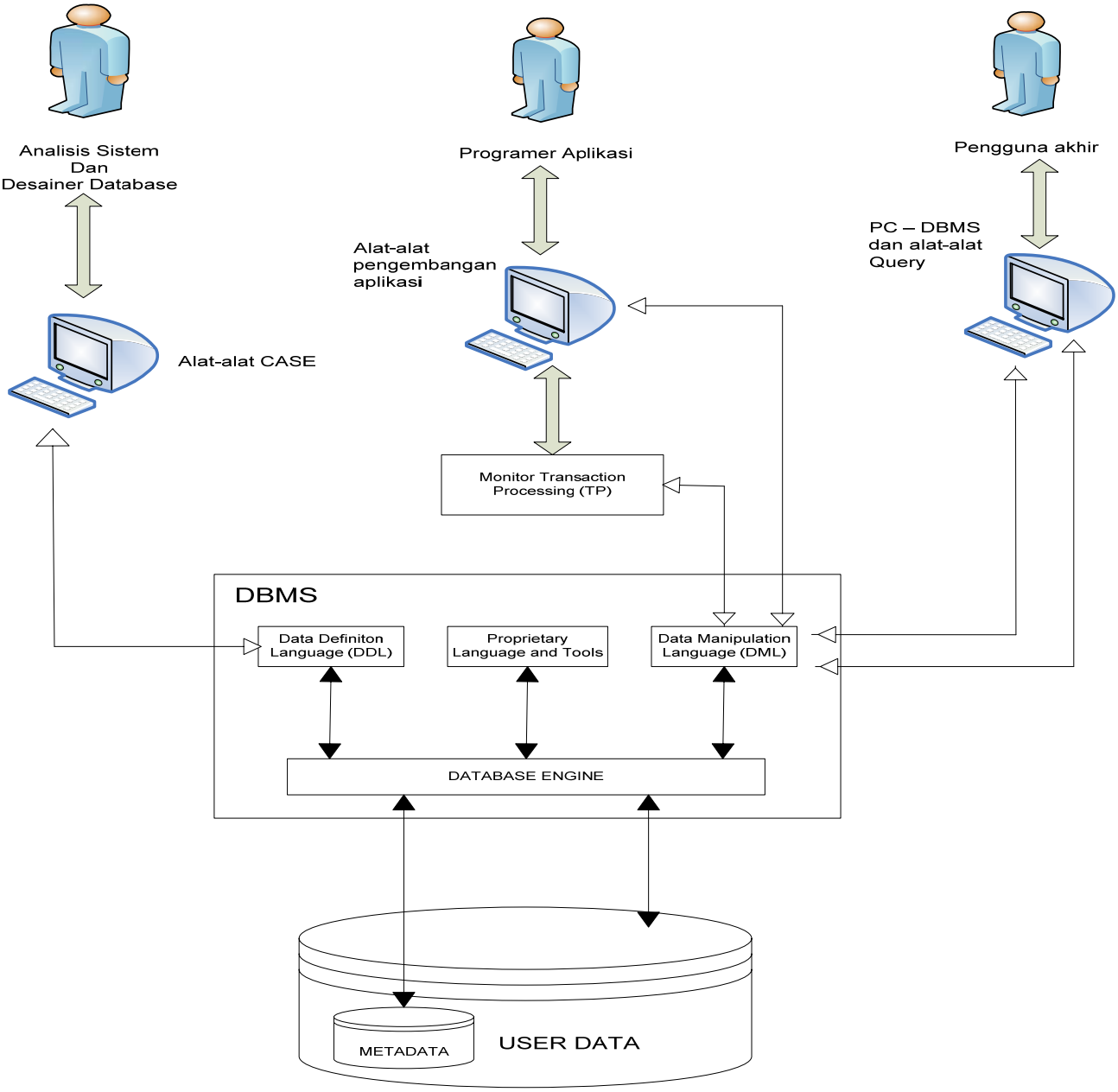
Menurut Brian K.Williams, dan Stacey C. Sawyer (2007, p464), *Database* adalah kumpulan data yang saling berhubungan, yang diatur secara logis, yang dirancang dan dibangun untuk tujuan khusus.

Keuntungan dan kelamahan pada database menurut Jeffery L. Whitten, Lonnie D. Bentley, dan Kevin C. Dittman (2004, p519) kelemahan yang umum mengenai pendekatan *database* adalah bahwa anda dapat membangun sebuah *super database* yang terdiri dari semua item data yang diperlukan oleh sebuah organisasi. Hal tersebut justru akan membangun sebuah *database* yang kompleks. Karena pada kenyataannya sebagian besar organisasi membangun beberapa *database*, masing-masing berbagi pakai data dengan beberapa sistem informasi. Jadi, akan ada beberapa redudansi diantara *database*. Sedangkan keuntungan teknologi *database* menawarkan penyimpanan data dalam format yang fleksibel. Hal ini memungkinkan karena *database* didefinisikan secara terpisah dari sistem informasi dan program-program aplikasi yang akan menggunakan *database*.

2.1.10 Database Management Sistem (DBMS)

Menurut Jeffery L. Whitten, Lonnie D. Bentley, dan Kevin C. Dittman (2004, p524) *Database Management System* (DBMS) adalah perangkat lunak komputer khusus yang disediakan dari vendor-vendor komputer yang digunakan untuk membuat, mengakses, mengontrol, dan mengelola *database*. Menurut Brian K. Williams dan Stacey C. Sawyer (2007, p422) keuntungan dari DBMS adalah:

1. Pengulangan Data Berkurang
2. Integritas Data Meningkat
3. Keamanan Meningkat
4. Kemudahan Memelihara Data



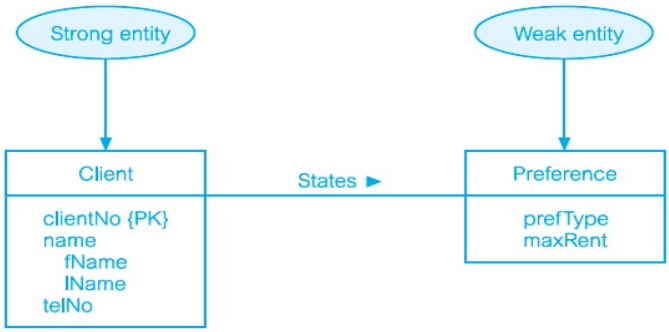
Gambar 2.1 Arsitektur Sistem Manajemen *Database*

2.1.11 *Entity Relationship Diagram*

Menurut Conolly dan Begg (2005) Model ER adalah suatu pendekatan *top-down* ke *design database* yang dimulai dengan mengidentifikasi data yang penting disebut *entity* dan relasi antara data yang harus mempresentasikan model kemudian menambahkan lebih detail seperti informasi yang ingin menjaga tentang *entity* dan relasi disebut atribut

2.1.11.1 Entitas

- ***Strong entity*** (entitas kuat) : Entitas yang mandiri, yang keberadaannya tidak bergantung pada keberadaan entitas yang lainnya. Instansiasi entitas kuat selalu memiliki karakteristik yang unik disebut identifier (sebuah atribut tunggal atau gabungan atribut-atribut yang secara unik dapat digunakan untuk membedakannya dari entitas kuat yang lain). Sering juga disebut parent, owner, dominan.
- ***Weak entity*** (entitas lemah) : Entitas yang keberadaannya sangat bergantung pada keberadaan entitas yang lainnya. Entitas lemah tidak memiliki arti apa-apa dan tidak dikehendaki kehadirannya dalam diagram ER tanpa kehadiran entitas di mana mereka bergantung. Sering juga disebut child, dependent, subordinat.



Gambar 2.2 Entitas

2.1.11.2 Atribut

Atribut adalah sebuah properti dari suatu entitas atau tipe hubungan. Atribut dapat diklasifikasikan sebagai berikut:

1. ***Simple Attributes***

   Sebuah atribut yang terdiri dari komponen tunggal dengan keberadaan independen.

2. ***Composite Attributes***

   Sebuah atribut yang terdiri dari beberapa komponen, masing-masing dengan keberadaan independen.

3. ***Single-Valued Attributes***

   Sebuah atribut yang memegang nilai tunggal untuk setiap kemunculan suatu entity.

4. ***Multi-Valued Attributes***

   Sebuah atribut yang memegang beberapa nilai untuk setiap terjadinya suatu tipe entitas.

5. ***Derived Attributes***

   Sebuah atribut yang mewakili nilai yang diturunkan dari nilai atribut terkait atau sekumpulan atribut, belum tentu dalam jenis entitas yang sama.

2.1.11.3 Hubungan atau Relasi

Hubungan adalah satu set asosiasi yang bermakna antara jenis entitas. Relasi menunjukkan adanya hubungan diantara sejumlah entitas yang berasal dari himpunan entitas yang berbeda.

2.1.12 Interaksi Manusia dan Komputer (IMK)

Interaksi manusia dan komputer (IMK) atau *Human Computer Interaction* (HCI) secara umum adalah mempelajari suatu interaksi yang terjadi antara manusia dan komputer (*E-book Human Computer Interaction, p4*).

2.1.13 Delapan Aturan Emas

Berdasarkan, peraturan ini didapat dari tulisan Designing the *user interface* oleh Ben Shneiderman (2008, p75-77). Shneiderman mengusulkan koleksi ini berdasar dari prinsip yang sudah diturunkan dari pengalaman di kebanyakan sistem interaktif setelah diperbaiki, ditambahkan dan diterjemahkan dengan benar. Untuk meningkatkan kegunaan dari aplikasi, adalah penting untuk memiliki desain antarmuka yang baik. *Eight Golden Rules Of Interface Design* (Delapan Aturan Emas dari Desain Antarmuka) dari Ben Shneiderman adalah sebuah petunjuk untuk desain antar muka.

a. Berusaha untuk konsisten. Rangkaian aksi yang konsisten dibutuhkan dalam situasi-situasi yang mirip, terminologi yang indetik harus digunakan pada prompts, menu dan layar pertolongan, dan perintah yang konsisten harus digunakan secara keseluruhan.
b. Memungkinkan pengguna yang sering untuk menggunakan shortcut (jalan pintas) sejalan dengan peningkatan penggunaan, pengguna juga menginginkan pengurangan jumlah interaksi dan meningkatkan kecepatan interaksi. Singkatan, fungsi, perintah tersembunyi, dan fasilitas makro sangat membantu untuk pengguna yang sudah mahir.
c. Menawarkan balasan yang informatif untuk setiap operator dari aksi. Seharusnya ada balasan dari sistem. Untuk aksi yang sering dan kecil, balasan seharusnya sederhana,

dimana untuk aksi yang jarang dan besar, respon harus lebih lengkap.

d. Mendesain dialog untuk menyatakan penutupan rangkaian dari aksi dapat diatur ke dalam kumpulan dengan awal, tengah dan akhir. Informasi balasan pada penyelesaian dari kumpulan aksi memberikan *operator* kepuasan dari penyelesain, perasaan lega, sinyal untuk menghilangkan kemungkinan dan pilihan dari pemikirannya, dan sebuah indikasi bahwa sudah selesai dan dipersiapkan untuk kumpulan aksi berikutnya.
e. Menawarkan penanganan kesalahan yang sederhana sebanyak mungkin, desain sistem yang membuat pengguna tidak dapat melakukan kesalahan yang fatal. Jika kesalahan terjadi, sistem harus dapat mendeteksi kesalahan dan menawarkan mekanisme yang sederhana dan luas untuk mengatasi kesalahan.
f. Mengizinkan pengembalian aksi yang mudah. Keistimewaan ini menghilangkan kegelisahan, karena pengguna tahu kesalahan dapat dihindari, dengan begitu meningkatkan keinginan untuk menelusuri pilihan-pilihan yang asing.
g. Mendukung tempat untuk kendali operator yang mahir, sangat menginginkan perasaan bahwa mereka sedang berada dalam sistem dan sistem merespon aksinya. Desain sistem yang membuat pengguna sebagai pengontrol aksi dari pada perespon.
h. Kurangi bahan ingatan jangka pendek. Keterbatasan manusia untuk memproses ingatan jangka pendek membutuhkan tampilan tetap sederhana, banyak halaman tampilan digabungkan, gerakan *window* yang banyak dikurangi, dan pelatihan yang cukup dibagi untuk kode-kode, menghafal, dan rangkaian aksi.

2.1.14 Rekayasa Perangkat Lunak (RPL)

Menurut Roger S. Pressman, Ph. D (2001, p10) rekaya piranti lunak adalah pemahaman tentang perangkat lunak (serta pemahaman tentang *software engineering*). Beberapa karakteristik perangkat lunak diantaranya :

- Perangkat lunak dibangun dan dikembangkan, tidak dibuat dalam bentuk yang klasik.
- Perangkat lunak tidak pernah usang.
- Sebagian besar perangkat lunak dibuat secara *custom-built*, serta tidak dapat dirakit dari kompenen yang sudah ada.

2.1.15 Tahap-Tahap Perancangan Piranti Lunak

Menurut Turban, Rainer, Potter (2006, p690) siklus hidup pengembangan sistem (*system development life cycle -SDLC*) adalah metode pengembangan sistem tradisional yang digunakan oleh berbagai perusahaan untuk proyek TI besar seperti instruktur TI. SDLC adalah kerangka kerja terstruktur yang terdiri atas berbagai proses berurutan untuk mengembangkan sistem informasi. Proses tersebut meliputi penelitian, analisis, desain, pemograman, penguji, implementasi, operasi, dan pemeliharaan. Berbagai proses tersebut terdiri atas pekerjaan-pekerjaan tertentu. Berikut beberapa tahapan pada SDLC :

a. Penelitian Sistem

   Karena *software* merupakan bagian dari suatu sistem, maka dimulai dengan penetapan semua sistem elemen dan mengalokasikan beberapa bagiannya ke dalam usulan pada software kemudian menggabungkan semua *level* sistem dengan melakukan pengkajian dari *level* atas dalam pendesainan dan analisis.

b. Analisis Kebutuhan *Software*

Merupakan proses mengerti tentang domain informasi, fungsi, kinerja dan tatap muka pada *software*.

c. Desain

Pada desain, prinsipnya adalah mengubah kebutuhan menjadi *software* yang layak dari segi kualitas sebelum proses pengkodean.

d. Pengkodean

Proses pengkodean yaitu mengubah ke dalam bentuk yang dapat dibaca oleh mesin.

e. Pengetesan

Proses yang memastikan semua kalimat dalam program telah dilakukan pengetesan sehingga memberikan input sesuai dengan yang diinginkan.

f. Implementasi

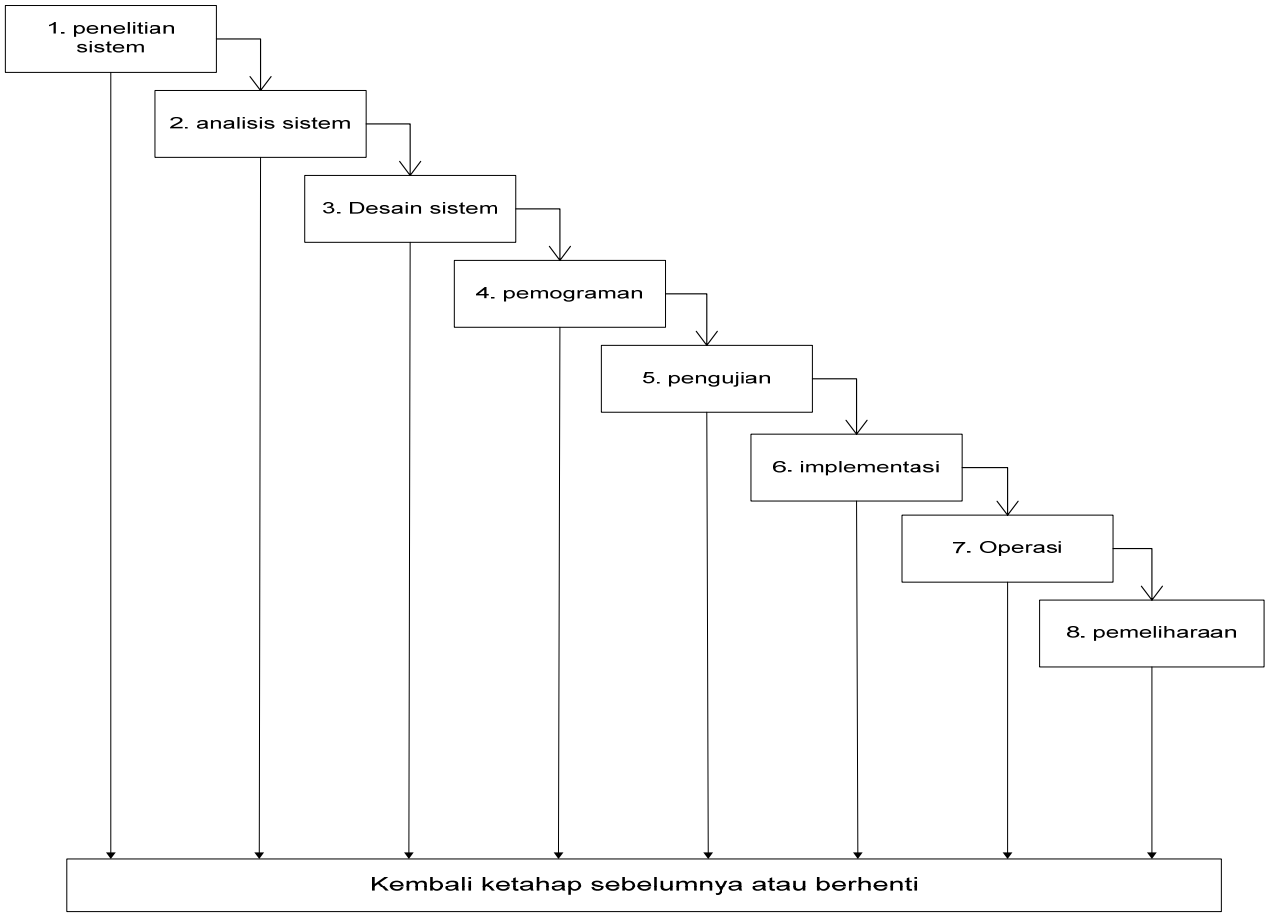
Software yang telah diuji dan siap diimplementasikan kedalam sistem pengguna/ sudah siap diterapkan.

g. Operasi

Pada tahap ini, sistem telah berjalan sebagaimana mestinya, akan tetapi secara secara berkala sistem membutuhkan modifikasi, penambahan peralatan baik perangkat keras maupun perangkat lunak pendukung, perubahan tenaga pendukung operasi, perbaikan kebijakan maupun prosedur dari suatu organisasi.

h. Pemeliharaan

Software akan mengalami perubahan setelah dikirim ke pengguna maka proses pemeliharaan dilakukan dengan menerapkan setiap langkah daur hidup sebelumnya disertai dengan perbaikan.



Gambar 2.3 proses (tahapan) SDLC

2.2 Teori Khusus

Teori khusus ini berisi mengenai perancangan system yang akan digunakan.

2.2.1 Unified Modeling Language (UML)

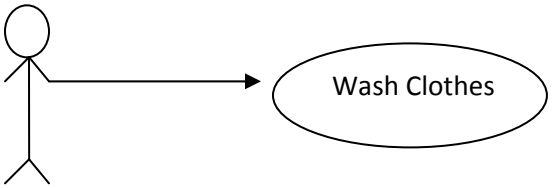
Menurut Grady Booch (2005, p7), UML adalah bahan baku untuk menulis rancangan/ *blueprint software*. UML dapat digunakan untuk menggambarkan, mendefinisikan, membentuk dan mendokumentasikan bagian-bagian dari sistem intensif dari *software*.

Bahasa pemodelan adalah bahasa yang pembendaharaan kata dan aturannya berfokus pada representasi sistem secara konseptual maupun fisik. UML membantu memodelkan sesuatu yang lebih mudah dipahami secara grafik. UML tidak hanya sekedar gambar dan simbol, karena tiap simbol dalam UML memiliki semantik yang terdefinisi dengan baik. Dalam hal ini, seorang pengembang dapat menuliskan model dalam UML, lalu pengembang lain dapat memahami model tersebut dengan tepat tanpa perbedaan persepsi.

UML mendefinisikan suatu model dengan tepat, tidak ambigu, dan lengkap. UML bukan bahasa pemrograman visual, namun model yang dibuat dengan UML dapat secara langsung terhubung dengan bermacam-macam bahasa pemrograman.

2.2.2 Use Case

Menurut Joseph schmuller (2001, p10) *Use Case Diagram*, adalah sebuah gambaran dari fungsi sistem yang dipandang dari sudut pandang pemakai. Contoh *Use Case Diagram* dapat dilihat pada gambar dibawah ini.



Gambar 2.4 Contoh Use Case Diagram Menurut Joseph Schmuller

2.2.3 Class Diagram

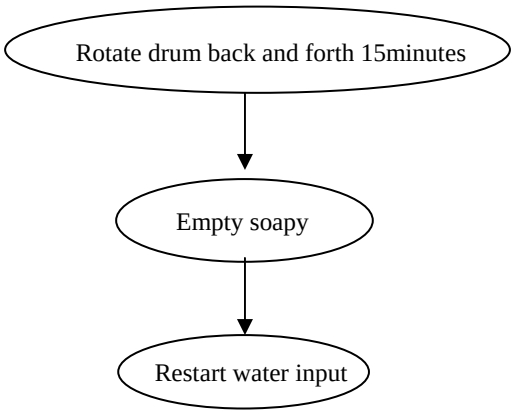
Menurut Joseph schmuller (2001, p8-9) *Class Diagram* adalah sebuah kategori atau pengelompokan dari hal – hal yang mempunyai atribut dan fungsi yang sama. *Class Diagram* adalah sebuah grafik presentasi dari gambaran statis yang menunjukkan sekumpulan model elemen yang terdeklarasi (statis), seperti kelas, tipe, dan isinya serta hubungannya. Contoh *Class diagram* dapat dilihat pada gambar dibawah ini.

Washing Machine
Brand Name Serial Name Capacity
Add Clothes() Add detergen() Remove Clothes()

Gambar 2.5 Contoh Class Diagram Menurut Joseph Scmuller

2.2.4 Activity Diagram

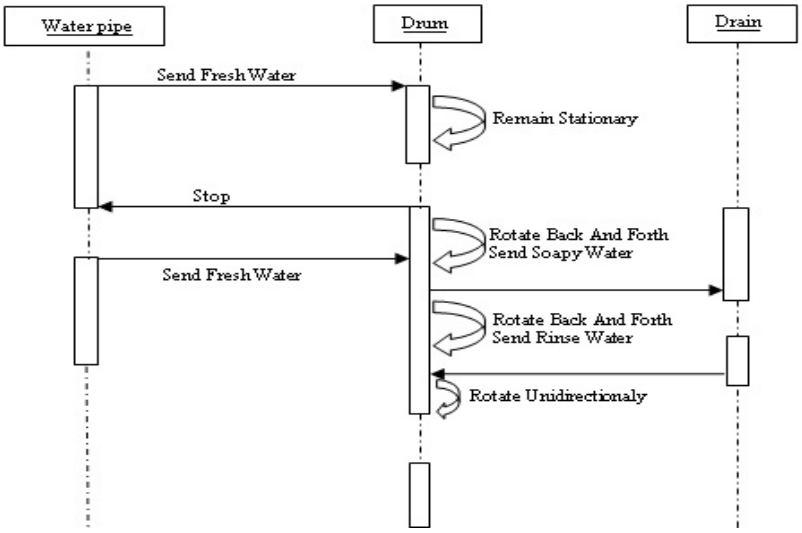
Menurut Joseph schmuller (2001, p12-13) Sebuah *Activity Diagram*, digunakan untuk mengetahui aliran kerja yang dilakukan oleh sebuah objek atau komponen. Contoh *Activity Diagram* dapat dilihat pada Gambar 2.6



Gambar 2.6 Contoh Activity Diagram Menurut Joseph Schmuller

2.2.5 Sequence Diagram

Menurut Joseph schmuller (2001, p11) *Sequence Diagram*, menunjukkan urutan pertukaran pesan yang dilakukan oleh sekumpulan objek atau *actor* yang mengerjakan pekerjaan tertentu. Contoh *Sequence Diagram* dapat dilihat pada gambar dibawah ini.



Gambar 2.7 Contoh Sequence Diagram Menurut Joseph Schmuller