BAB II
LANDASAN TEORI

2.1 Pengertian Rekayasa Perangkat Lunak

Istilah Rekayasa Perangkat Lunak (RPL) secara umum disepakati sebagai terjemahan dari istilah *Software Engineering*. Istilah *Software Engineering* mulai dipopulerkan tahun 1968 pada *Software Engineering Conference* yang diselenggarakan oleh NATO. Sebagian orang mengartikan RPL hanya sebatas pada bagaimana membuat program komputer. Padahal ada perbedaan yang mendasar antara perangkat lunak (*software*) dan program komputer[1].

Agar komputer dapat membaca, mengingat, membuat keputusan (membandingkan), menghitung, menyortir, dan menghasilkan *output* berupa informasi, komputer harus dapat membaca serta memasukkan program ke dalam memori utamanya. Program adalah instruksi dalam bahasa mesin atau yang dapat dibaca oleh komputer yang dirancang untuk tujuan tertentu sehingga apabila operator menjalankan komputer dan menekan tombol tertentu (misalnya untuk memproses data akuntansi) disebut dengan program aplikasi (*aplication program*). Pengertian perangkat lunak menunjuk pada program dan alat bantu lain yang bersifat menambah kemampuan komputer sebagai alat untuk melaksanakan tugas atau operasi tertentu. Program aplikasi dapat dibuat secara khusus untuk memenuhi kebutuhan khusus pula (*tailor-made*) atau berupa paket yang mempunyai aplikasi umum. Dapat disimpulkan perangkat lunak yaitu :

1. Merupakan kumpulan beberapa perintah yang dieksekusi oleh mesin komputer dalam menjalankan pekerjaannya. perangkat lunak ini merupakan catatan bagi mesin komputer untuk menyimpan perintah, maupun dokumen serta arsip lainnya.
2. Merupakan data elektronik yang disimpan sedemikian rupa oleh komputer itu sendiri, data yang disimpan ini dapat berupa program atau instruksi yang akan dijalankan oleh perintah, maupun catatan-catatan yang diperlukan oleh komputer untuk menjalankan perintah yang dijalankannya.

3. Untuk mencapai keinginannya tersebut dirancanglah suatu susunan logika, logika yang disusun ini diolah melalui perangkat lunak, yang disebut juga dengan program beserta data-data yang diolahnya. Pengeloahan pada software ini melibatkan beberapa hal, diantaranya adalah sistem operasi, program, dan data. Software ini mengatur sedemikian rupa sehingga logika yang ada dapat dimengerti oleh mesin komputer.

2.2 Sistem Teknologi Informasi

Seringkali terdapat penggunaan istilah data dan informasi secara bersamaan dengan maksud yang sama, padahal data dan informasi merupakan dua hal yang berbeda. Walaupun demikian keduanya berkaitan erat dengan fakta. Data adalah bahan informasi, dirumuskan sebagai kumpulan dari simbol-simbol yang teratur yang menyatakan jumlah, tindakan-tindakan, hal-hal dan sebagainya. Data dibentuk dari lambang grafis, alfabetis, numerik, atau lambang khusus. Sedangkan informasi adalah data yang telah diolah ke dalam bentuk yang berarti bagi si pemakai, mempunyai nilai guna dan manfaat dalam proses pengambilan keputusan pemakainya[2].

Hubungan data dan informasi didefinisikan sebagai bahan baku dan produk jadi. Data sebagai bahan baku, diolah melalui suatu proses transformasi atau pengolahan data menjadi informasi. Atau dapat dikatakan bahwa informasi merupakan keluaran-keluaran (output) dari proses transformasi, dimana data berfungsi sebagai masukan-masukannya (input). Jika ditinjau sebagai suatu sistem, maka sistem informasi akan menerima masukan-masukan yang berupa data dan instruksi, mengolah data sesuai dengan instruksi-instruksi, dan mengeluarkan hasilnya berupa informasi-informasi. Sistem informasi adalah suatu sistem didalam organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung informasi, bersifat manajerial, dan kegiatan dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan.

Berdasarkan uraian pengertian dari kedua kata yang membentuknya, maka dapat dijelaskan sistem informasi adalah sebuah sistem manusia/mesin yang

terpadu untuk menyajikan informasi guna mendukung fungsi operasi, manajemen, dan pengambilan keputusan dalam sebuah organisasi. Sistem ini menggunakan perangkat keras dan perangkat lunak komputer, prosedur pedoman, model manajemen dan keputusan, dan sebuah basis data[2].

Istilah Teknologi Informasi yang populer saat ini adalah bagian dari mata rantai panjang dari perkembangan istilah dalam dunia SI (Sistem Informasi) atau IS (Information System). Istilah TI memang lebih merujuk pada teknologi yang digunakan dalam menyampaikan maupun mengolah informasi, namun pada dasarnya masih merupakan bagian dari sebuah sistem informasi itu sendiri. TI memang secara nota bene lebih mudah dipahami secara umum sebagai pengolahan informasi yang berbasis pada teknologi komputer yang tengah terus berkembang pesat.

Sebuah Sistem TI atau selanjutnya akan disebut STI, pada dasarnya dibangun diatas lima tingkatan dalam sebuah piramida STI. Berurutan dari dasar yaitu konsep dasar, teknologi, aplikasi, pengembangan dan pengelolaan. STi pula memilki beberapa tuijuan yang diantaranya adalah :

1. **Performance,** peningkatan yterhadap kinerja,
2. **Information,** peningkatan terhadap mutu dan informasi yang disajikan,
3. **Economy,** peningkatan terhadap nilai manfaat/penurunan biaya yang terjadi
4. **Control,** peningkatan terhadap pengendalian untuk mendeteksi dan memperbaiki kesalahan/kecurangan yang akan terjadi.
5. **Efficiency,** peningkatan terhadap efisiensi operasi,
6. **Services,** peningkatan pelayanan yang diberikan oleh system[3].

2.2.1 Konsep dasar

Konsep memberikan pemahaman yang penting dan menyeluruh dari sebuah STI yang tengah dibangun. Setidaknya ada 4 (empat) konsep dasar dari sebuah STI yang harus dipahami secara umum, yaitu :

1. **Konsep tentang sistem yang tengah berlangsung atau berlaku.** Ini penting karena STI itu sendiri adalah sebuah sistem dan merupakan bagian dari sistem pula, misalnya dalam sebua
2. **Konsep tentang informasi.** Informasi tentu saja adalah produk yang diharapkan dapat dihasilkan dari sebuah STI dan informasi adalah sebuah fokus yang harus mendapatkan pemahaman serius secara umum dan merata. Sudah menjadi sebuah permasalahan yang sering kali muncul manakala sering kali didapati sebuah kenyataan bahwa terkadang sebuah STI tidak selalu menghasilkan informasi, bahwa banyak dari STI dapat dinilai gagal karena ternyata bukan informasi yang dihasilkan, meskipun didukung teknologi yang cukup memadai.
3. **Konsep yang menyangkut komponen-komponen pembentuk STI itu sendiri.** Pemahaman akan hal tersebut akan berguna saat proses penerapan STI dengan aplikasi – aplikasi berbeda sambil tetap mempertahankan STI tersebut sebagai satu kesatuan yang utuh. Aplikasi STI untuk Bagian Penjualan sudah tentu akan berbeda dengan aplikasi yang digunakan di Bagian Keuangan dan pasti berbeda dengan yang diterapkan di Bagian Personalia, namun ketiganya merupakan bagian dari sebuah STI yang lebih luas dan besar dan dibangun atas dasar yang sama. Konteks penerapannyalah yang membuat ketiganya memiliki perbedaan.
4. **Konsep tentang pemanfaatan informasi yang dihasilkan dari STI yang dikembangkan.** Dengan memahami tipe-tipe/jenis-jenis pemanfaatan informasi, maka dapat diketahui karakteristik/macam ragam informasi yang relevan untuk dihasilkan oleh sebuah STI[3].

2.2.2 Teknologi

Teknologi merupakan penerapan keilmuan yang mempelajari dan mengembangkan kemampuan dari suatu rekayasa dengan langkah dan teknik tertentu dalam suatu bidang. Di atas konsep dasar dapat ditentukan teknologi yang akan digunakan dalam STI yang akan dikembangkan. Dapat berupa teknologi komputer, telekomunikasi atau teknologi apapun yang dapat memberi nilai tambah dalam proses STI[3].

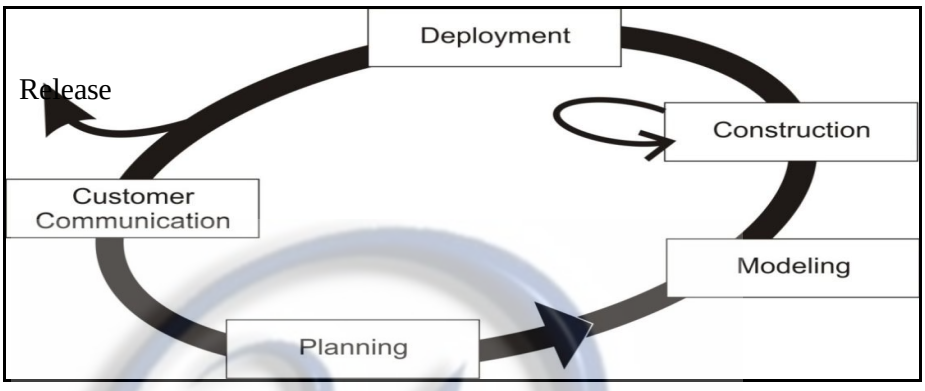
2.2.3 Aplikasi

Pengaplikasian dari STI dapat diterapkan dengan berbagai cara. dapat diterapkan mengikuti fungsi-fungsi organisasi atau tingkatan manajemen dimana STI tersebut akan diaplikasikan. Beberapa contoh STI yang diaplikasikan mengikuti fungsi-fungsi organisasi yang ada misalnya, MIS (*Marketing Information System*) untuk Bagian Penjualan, HRIS (*Human Resources Information System*) untuk Bagian Personalia, atau FIS (*Financial Information System*) untuk Bagian Keuangan. Sedangkan beberapa contoh STI yang diaplikasikan mengikuti fungsi-fungsi manajemen yang ada misalnya, TP (*Transaction Processing*) dan PCS (*Process Control System*) untuk manajemen level bawah, DSS (*Decision Support System*) atau sistem penunjang keputusan, ES (*Expert System*) atau sistem pakar, kemudian ada EIS (*Executive Information System*) untuk manajemen tingkat menengah dan atas[3].

2.2.4 Pengembangan

Dalam membangun sebuah sistem berbasis komputer, perlu dilakukan tahapan-tahapan pengembangan. Metodologi yang digunakan adalah *data oriented*. Metode pengembangan sistem yang digunakan adalah metode *WebE Proses*, sistem yang dikembangkan dipecah menjadi beberapa modul, dengan menggunakan metode ini dokumentasi menjadi lebih terkontrol yang berguna untuk pengembangan sistem selanjutnya.

Adapun tahapan-tahapan yang dilakukan dalam melakukan pengembangan sistem adalah *Costomer Communication, Planning, Modeling, Construction, Deployment. G*ambar 3.1 halaman 2-6 menggambarkan metode *WebE Proses.*



Gambar 2.1 *Web*E Proses[4]

Penjelasan dari gambar 3.1 halaman 3-5 adalah sebagai berikut:

1. *Costomer Communication*

   Costomer Communication dibagi menjadi dua yaitu:

   a. Business analysis yaitu mendefinisikan konteks bisnis untuk membangun suatu aplikasi *web*

   b. Formulation yaitu aktivitas yang dilakukan oleh semua pihak yang berkepentingan untuk mendefinisikan semua kebutuhan yang diperlukan membangun aplikasi *web*

2. *Planning*

   Merencanakan proyek untuk membuat aplikasi berbasis *Web*

3. *Modeling*

   Analisa dan desain perangkat lunak yang disesuaikan untuk pengembangan aplikasi berbasis *web*

4. *Construction*

   Peralatan dan teknologi yang digunakan untuk mengkonstruksi aplikasi *web* yang sudah dimodelkan

5. *Deployment*

   Aplikasi *web* yang telah dikonfigurasi untuk lingkungan operasionalnya dan kemudian dikirimkan kepada *user*[5].

2.2.5 Pengelolaan

Tahap paling tinggi dari pengembangan STI adalah pengelolaan STI itu sendiri yang telah beroperasi. Ada 2 (dua) isu penting tentang pengelolaan STI, yaitu sebagai berikut :

1. **Pertama**, pengendalian dan kontrol terhadap STI itu sendiri. Kontrol yang tidak dikelola dengan baik akan menyebabkan STI tidak dapat mencapai tujuannya. Informasi yang diinginkan dari STI mungkin bisa menjadi tidak akurat. Kontrol dan pengendalian di sini termasuk di dalamnya isu-isu seputar kemanan STI.
2. **Kedua**, etika dan politik informasi yang juga harus diberikan perhatian yang cukup. Pengelolaan di bidang ini yang dilakukan dengan tidak tepat mungkin akan menurunkan kinerja. Demikian juga dengan pengelolaan politik informasi. Banyak STI yang secara teknis bagus, tetapi mengalami kegagalan dalam penerapannya karena adanya politik informasi yang menggagalkan STI tersebut. Salah satu diantaranya adalah adanya *resistance to change* atau keengganan berubah karena STI yang diterapkan ini akan menurunkan kekuasaan atau kesempatan seseorang yang menyebabkan yang bersangkutan enggan menerima STI yang ada[3].

2.3 Internet [7]

Internet berasal dari kata Interconnection Networking yang mempunyai arti hubungan komputer dengan berbagai tipe yang membentuk sistem jaringan yang mencakup seluruh dunia (jaringan komputer global) dengan melalui jalur telekomunikasi seperti telepon, radio link, satelit dan lainnya. Dalam mengatur integrasi dan komunikasi jaringan komputer ini digunakan protokol yaitu TCP/IP. TCP (Transmission Control Protocol) bertugas

memastikan bahwa semua hubungan bekerja dengan benar, sedangkan IP (Internet Protocol) yang mentransmisikan data dari satu komputer ke komputer lain. TPC/IP secara umum berfungsi memilih rute terbaik transmisi data, memilih rute alternatif jika suatu rute tidak dapat di gunakan, mengatur dan mengirimkan data.

2.4 WWW (*World Wide Web*)

WWW (*World Wide Web*) adalah jaringan beribu-ribu komputer yang dikategori menjadi dua : *Client* dan *server* dengan menggunakan *software* khusus membentuk sebuah jaringan yang disebut jaringan *client-server*. Dalam cara kerja dari www ada dua hal yang terpenting yaitu *software web server* dan *software web browser*. [7]

Server menyimpan/menyediakan informasi dan memproses permintaan dari *client,* apabila ada *client* yang meminta informasi maka *server* mengirimkannya. Informasi yang diakses dapat berupa teks, gambar, suara. *Server* juga mengirimkan perintah-perintah ke *client* tentang bagaimana cara menampilkan semua informasi tersebut. Intalasi tersebut dalam bentuk HTML (*Hypertext Markup Language*). Client membuat permintaan informasi dan kemudian menangani pengaksesan informasi tersebut kepada *end user* (pemakai akhir).

Komunikasi jaringan komputer diatur dengan bahasa/*software* standar yang disebut dengan protocol. Hal ini memungkinkan beragam jaringan komputer dan jenis komputer yang berbeda untuk berkomunikasi.Protokol ini secara resmi di kenal sebagai TCP/IP (*Transmission Control Protocol Internet Protocol*) merupakan cara standar untuk mempaketkan dan menyelamatkan data komputer (sinyal elektronik) sehingga data tersebut dapat dikirim ke komputer yang lain[7].

2.5 Pendekatan Berorientasi Objek

Sistem-sistem berbasis komputer harus merupakan hasil dari suatu analisis dan dirancang sebaik mungkin. Hasil analisis dan rancangan tersebut harus merupakan hasil yang terbaik, sebab akan menentukan keseluruhan hasil yang

akan di dapat di akhir. Apabila analisis atau rancangan tidak baik atau bermasalah, maka akan berdampak buruk pula pada hasil akhir yang akan di dapat.

Rancangan yang bermasalah dapat menuntun sistem menjadi tidak berfungsi atau bahkan membahayakan kehidupan. Rancangan merupakan hasil aktivitas. Aktivitas-aktivitas yang dilakukan harus berdasarkan suatu pendekatan yang beralasan. Pendekatan konvensional (aliran data atau terstruktur) tidak berdasarkan pada entitas-entitas di dunia eksternal dan hal ini mempersulit dalam mengelola dan mengadaptasi ketika terjadi perubahan kebutuhan.

Pendekatan berorientasi objek merupakan pendekatan yang cukup efektif karena objek-objek dapat merepresentasikan bagian-bagian dari dunia ekternal, mempersempit kesenjangan (gap) konseptual antara dunia eksternal dan komponen-komponen perangkat lunak.

Pendekatan berorientasi objek memiliki beberapa keunggulan. Keunggulan pendekatan berorientasi objek diantaranya adalah sebagai berikut :

1. Bekerja yang mendekati keadaan lingkungan manusia.
2. Menghasilkan sistem yang dibangun di atas bentuk-bentuk antara yang stabil dan dengan demikian lebih mampu untuk mengikuti perubahan.
3. Dapat digunakan tidak hanya pada perancangan perangkat lunak tapi juga seluruh proses pengembangan perangkat lunak.
4. Pendekatan ini membantu mengeksploitasi keampuhan bahasa pemrograman berbasis objek atau berorientasi objek.

2.6 *Unified Modeling Language* (UML)

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem perangkati lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi perangkat lunak, dimana aplikasi tersebut dapat berjalan pada perangkat keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun.

UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram perangkat lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntax* mendefinisikan bagaimana bentuk – bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (*Object-Oriented Design*), Jim Rumbaugh OMT *(Object Modeling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*).

Dimulai pada bulan Oktober 1994 Booch, Rumbaugh dan Jacobson, yang merupakan tiga tokoh yang boleh dikata metodologinya banyak digunakan mempelopori usaha untuk penyatuan metodologi pendesainan berorientasi objek. Pada tahun 1995 *direlease draft* pertama dari UML (versi 0.8). Sejak tahun 1996 pengembangan tersebut dikoordinasikan oleh ***Object Management Group*** (OMG – http://www.omg.org).

Pada tahun 1997 UML versi 1.1 muncul, dan saat ini versi terbaru adalah versi 1.5 yang dirilis bulan Maret 2003. Booch, Rumbaugh dan Jacobson menyusun tiga buku serial tentang UML pada tahun 1999. Sejak saat itulah UML telah menjelma menjadi standar bahasa pemodelan untuk aplikasi berorientasi objek.

2.6.1 Konsepsi Dasar UML

Dari berbagai penjelasan rumit yang terdapat di dokumen dan buku-buku UML. Sebenarnya konsepsi dasar UML bisa kita rangkumkan dalam tabel 2.1 halaman 2-11.

Tabel 2.1 Konsepsi UML[8]

Major Area	*Views*	*Diagrams*	*Main Concepts*
Structural	*Static View*	*class Diagram*	*class, association, generalization, Dependency, realization, interface*
	Use Case View	*Use Case Diagram*	*use case, actor, association, extend, Include, use case generalization*
	Implementation View	*Component Diagram*	*component, interface, dependency, realization*
	Deployment View	*Deployment Diagram*	*node, component, dependency, location*
Dynamic	*State Machine View*	*State Chart Diagram*	*state, event, transition, action*
	Activity View	*Activity Diagram*	*state, activity, completion transition, fork, join*
	Interaction View	*Sequence Diagram*	*interaction, object, message, activation*
		Collaboration Diagram	*collaboration, interaction, collaboration role, message*

Major Area	*Views*	*Diagrams*	*Main Concepts*
Model Management	*Model Management View*	*Class Diagram*	*package, subsystem, model*
extensibility	*All*	*All*	*constraint, stereotype, tagged value*

Seperti juga tercantum pada gambar di-atas UML mendefinisikan diagram-diagram sebagai berikut:

- *use case diagram*
- *class diagram*
- *sequence diagram*
- *use case Model*
- *Object Robustness Diagram*

2.6.2 *Use Case Diagram*

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.

Use case diagram dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang kasus uji untuk semua bagian yang ada pada sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya.

Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang sama. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

2.6.3 *Class Diagram*

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

Class memiliki tiga area pokok :

- Nama (dan stereotype)
- Atribut
- Metode

Atribut dan metoda dapat memiliki salah satu sifat berikut :

- *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.
- *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya.
- *Public*, dapat dipanggil oleh siapa saja.

Hubungan Antar *Class*

1. Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau

class yang harus mengetahui eksistensi *class* lain. Panah *navigability* menunjukkan arah *query* antar *class*.

2. Agregasi, yaitu hubungan yang menyatakan bagian ("terdiri atas..").
3. Pewarisan, yaitu hubungan hirarkis antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.
4. Hubungan dinamis, yaitu rangkaian pesan (*message*) yang di-*passing* dari satu *class* kepada *class* lain. Hubungan dinamis dapat digambarkan dengan menggunakan ***sequence diagram*** yang akan dijelaskan kemudian.

2.6.4 *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan.

Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class*. *Activation bar* menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message*.

2.6.5 *Use Case Model*

Use case model menggambarkan hubungan antara *actor* dengan proses, ataupun sebaliknya proses engan *actor. Use case model* merupakan pemodelan dari proses-prose yang dilakukan *actor* dalam perancangan aplikasi.

2.6.6 *Collaboration Diagram*

Collaboration diagram menggambarkan struktur dan hubungan antar komponen piranti lunak dengan *actor* yang melakukan proses, termasuk ketergantungan (*dependency*).Komponen piranti lunak adalah modul berisi *code*, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time*, *link time*, maupun *run time*. Umumnya komponen terbentuk dari beberapa *class* dan/atau *package*, tapi dapat juga dari komponen-komponen yang lebih kecil. Komponen dapat juga berupa *interface*, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain.

2.7 Analisis Berorientasi Objek

Sebutan lengkap analisis adalah analisis kebutuhan perangkat lunak (*software requirement analysis)*. Analisis kebutuhan adalah mendaftarkan apa-apa yang harus dipenuhi oleh sistem perangkat lunak, bukan mengenai bagaimana sistem perangkat lunak melakukannya.

Analisis merupakan bidang yang menarik, melibatkan studi interaksi antar manusia, kelompok-kelompok orang, komputer dan organisasi. Seperti dinyatakan oleh Tom DeMarco:

"Analisis (sistem) adalah hal dapat membuat frustasi, penuh dengan hubungan antar manusia yang kompleks, tak tentu dan sulit. Dalam satu kata, ini menakjubkan. Sekali anda terlibat, kesenangan mudah dari pembangunan sistem yang lama tidak lagi memadai untuk memuaskan anda" [9].

Pemodelan Penggunaan Sistem

Pemodelan penggunaan sistem dibagi ke dalam tiga bagian, yaitu pemodelan *use-case,* pemodelan struktur, pemodelan perilaku kelas objek. Kedua pemodelan tersebut akan dijelaskan pada sub bab halaman 2-22.

2.7.1 Pemodelan *Use-Case*

Pemodelan *use-case* dibagi ke dalam beberapa tahap, yaitu :

a. Identifikasi aktor
b. Identifikasi *use-case*
c. Pembuatan diagram *use-case*
d. Pembuatan diagram sekuen atau diagram kolaborasi untuk memperjelas masing-masing *use-case*
e. Pembuatan digram aktivitas

2.7.2 Pemodelan Struktur

Pemodelan struktur dibagi ke dalam beberapa tahap, yaitu :

a. Pemodelan High Level *Use-Case*
b. Pemodelan struktur dan hirarki
c. Identifikasi atribut-atribut
d. Identifikasi operasi-operasi
e. Pemodelan keterhubungan kelas

2.7.3 Pemodelan Perilaku Kelas Objek

Pemodelan perilaku kelas objek dibagi ke dalam beberapa tahap, yaitu :

a. Evaluasi semua *use-case* agar dapat memahami sepenuhnya sekuen interaksi di dalam sistem.
b. Identifikasi kejadian-kejadian yang menuntun sekuen interaksi dan pahami bagaimana kejadian-kejadian ini berhubungan dengan objek-objek tertentu.
c. Pembuatan *diagram sekuen* untuk masing-masing *use-case*.
d. Pembuatan *diagram kolaborasi* untuk masing-masing kelas.

e. Pembuatan *diagram aktivitas* untuk memperjelas masing-masing kelas atau operasi.
f. Pembuatan *diagram statechart* untuk sistem.
g. Lakukan *review* model perilaku objek yang diperoleh untuk verifikasi akurasi dan konsistensi.

2.8 Perancangan Berorientasi Objek

Pada setiap disiplin rekayasa, perancangan merupakan pendekatan berdisiplin untuk menemukan solusi masalah. Perancangan merupakan penghubung antara spesifikasi kebutuhan dan implementasi. Perancangan menekankan pada solusi logik mengenai cara sistem memenuhi kebutuhan.

Terdapat banyak metode perancangan berorientasi objek. Perbedaan-perbedaan pada metode perancangan berorientasi objek bukan pada langkah-langkah esensi, hanya rincian-rincian, sehingga dapat disimpulkan langkah-langkah perancangan berorientasi objek adalah sebagai berikut :

1. Perancangan sistem meliputi arsitektur sistem dan pendeskripsian subsistem-subsistem dan alokasinya di pemroses dan proses.
2. Pemilihan strategi perancangan untuk implementasi manajemen data, dukungan antarmuka dan manajemen proses/memori, penanganan kesalahan.
3. Perancangan mekanisme kendali yang cocok untuk sistem.
4. Perancangan rinci kelas objek dalam hal struktur data dan algoritmanya.
5. Perancangan pertukaran pesan menggunakan kolaborasi antar objek dan hubungan objek.
6. Penciptaan model pertukaran pesan.
7. Melakukan *review* atas model rancangan dan melakukan iterasi bila perlu untuk perbaikan model rancangan yang sebelumnya.

2.9 Implementasi

Pemrograman diperuntukkan bagi implementasi untuk memenuhi kebutuhan-kebutuhan yang dispesifikasikan dengan rancangan rinci. Untuk

mengimplementasikannya dibutuhkan beberapa program pendukung yang akan dijelaskan pada sub bab berikut :

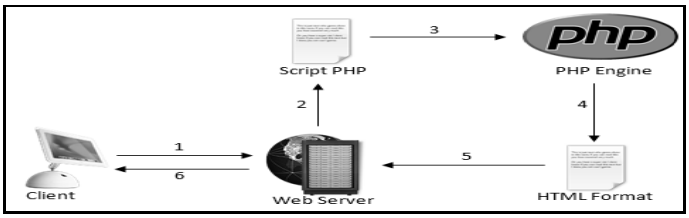
2.9.1 PHP

PHP adalah kependekan dari *Hypertext Preprocessor* atau Profesional *Homepage*, yaitu sebuah bahasa *scripting* yang dieksekusi di sisi *server* (*Server-side Scripting Language*). Fungsinya adalah membuat sebuah *web* yang interaktif dan dinamis. PHP dibuat pertama kali pada tahun 1995 oleh Rasmus Lerdorf, seorang *software engineer* anggota tim pengembangan *web server Apache*. Pada tahun 1996 Rasmos menulis kode script Perl untuk diterapkan pada hal yang lebih kompleks dengan bahasa pemrograman C.

Pada bulan Oktober 2000 PHP merilis versi 4.0 dengan mengintegrasikan Zend Engine, dan merupakan versi pengembangan dari PHP 3 *Script Engine*. Keunggulannya dari sifatnya yang *server-side* tersebut antara lain :

1. Tidak diperlukan kompatibilitas *browser* atau harus menggunakan *browser* tertentu, karena *server* yang akan mengerjakan skrip PHP. Hasil yang dikirimkan kembali ke *browser* apapun.
2. Dapat memanfaatkan sumber-sumber aplikasi yang dimiliki oleh *server*, misalnya koneksi ke basis data.
3. Skrip tidak dapat dilihat dengan menggunakan fasilitas *view* HTML *source.*

Interpreter PHP dalam mengeksekusi kode PHP pada sisi server (server slide), sedangkan tanpa adanya interpreter PHP, maka semua skrip dan aplikasi PHP yang dibuat tidak dapat dijalankan. Proses eksekusi kode PHP yang dilakukan oleh Apache *Web Server* dan *interpreter* secara diagram dapat dilihat pada gambar 3.3 halaman 3-18 berikut :



Gambar 2.2 Struktur Pembacaan *Web server*[10]

Adapun kelebihan dari PHP yaitu dapat "*melakukan*" semua aplikasi program CGI, seperti mengambil nilai *form*, menghasilkan halaman *web* yang dinamis, mengirim dan menerima *cookie*. PHP juga dapat berkomunikasi dengan layanan-layanan yang menggunakan protokol IMAP, SNMP, NNTP, POP3, HTTP, dan lain-lain.

Namun tampaknya kelebihan PHP yang paling signifikan adalah kemampuannya untuk melakukan koneksi dengan berbagai macam-macam basis data. Saat ini, basis data yang didukung PHP adalah : *Adabas D, InterBase, PostgreSQL, dBase, FrontBase, Solid, Empress, mSQL, Sybase, FilePro(read-only), Direct MS-SQL, Velocis, IBM DB2, MySQL, Unix dbm, Informix*, Semua basis data yang mempunyai *provider ODBC, Ingres, Oracle* (*OCI7 and OCI8*).

2.9.2 Basis Data MySQL

Isi *website* yang dinamis tentu memerlukan dukungan basis data yang handal. MySQL adalah basis data server yang cukup handal untuk memenuhi kebutuhan sebuah website portal. MySQL dikembangkan oleh sebuah perusahaan Swedia bernama MySQL AB, yang pada saat itu bernama TcX Data Konsult AB, sejak sekitar 1994 – 1995. MySQL versi 1.0 dirilis Mei 1996 secara terbatas kepada empat orang. Baru di bulan Oktober versi 3.11.0 dilepas ke publik. Versi pertama ini hanya berjalan di Linux dan Solaris serta sebagian besar masih belum terdokumentasi itu berangsur-angsur diperbaiki dan ditambah fitur demi fiturnya.

Barulah di versi-versi akhir 3.22 sepanjang 1998–1999 *MySQL* menjadi semakin populer dan dilirik orang. Kalau di versi ini *MySQL* mulai diadopsi

banyak orang dan meningkat jumlah penggunanya, maka di versi 5.0.41-lah terjadi banyak peningkatan dari sisi teknologi.

Data-data yang disimpan ke dalam MySQL, atau data yang ingin ditampilkan dari MySQL dapat menggunakan *Application Programing Interface* (API) yang ada pada PHP. Ketika seseorang mengirimkan *request* ke MySQL dengan *script* PHP melalui *Web browser*, maka *Web server Apache* akan memerintahkan *engine* PHP untuk *query* ke MySQL. Setelah MySQL memberikan data yang diminta, PHP akan memformatnya mengirimkan kepada Apache untuk selanjutnya diteruskan kepada member yang meminta data tersebut melalui HTTP. MySQL memiliki beberapa tipe data, tetapi yang banyak digunakan dalam basis data *website* portal adalah tipe data *int, varchar, double, date* dan *time*.