BAB III

OBJEK DAN METODE PENELITIAN

3.1. Objek Penelitian

Menurut Sugiyono menyatakan bahwa, definisi objek penelitian adalah sebagai berikut :

> "Objek penelitian merupakan Suatu atribut atau sifat atau nilai dari orang, objek atau kegiatan yang mempunyai variasi tertentu yang ditetapkan oleh peneliti untuk di pelajari dan kemudian ditarik kesimpulannya."
>
> (2009:38) .

Berdasarkan definisi diatas, dapat ditarik kesimpulan bahwa objek penelitian merupakan sesuatu hal yang akan diteliti dengan mendapatkan data untuk tujuan tertentu dan kemudian dapat ditarik kesimpulan penulis mengambil objek penelitian dengan objek *e-rental*.

3.1.1. Sejarah Perusahaan

PT. Musi Car Rental di jln. Pasantren 2 no 3 cigadung bandung sejak tahun 1995, pada mulanya perusahaan tersebut memiliki 4 unit kendaraan. Maksud dan tujuan perusaahaan tersebut untuk membantu masyarakat yang memerlukan bantuan jasa peminjaman kendaraan mobil untuk kebutuhan pribadi atau keluarga dengan melalui suatu proses atau prosedur penyewaan yang mudah, nyaman, aman dan tidak berbelit-belit.

PT. Musi Car Rental sebagai salah satu usaha yang bergerak dibidang penyewaan kendaraan khususnya mobil, dalam melaksanakan kegiatannya masih

menggunakan cara manual, seperti mencatat pendaftaran, membuat laporan keluar – masuk kendaraan yang di sewa maupun yang sudah di kembalikanserta masalah yang bersangkutan dengan hal keuangan.

Dalam penyewaan mobil biasanya penyewa langsung datang ke tempat usaha atau penyewaan, disana mereka dapat memilih jenis atau merk mobil serta dapat langsung menyewa dan membawa mobil yang di suka. Data-data yang diperlukan perusahaan dari para penyewa adalah jenis mobil yang di sewa, nama dan alamat penyewa.

Setiapa petugas penyewaan mempunyai data penyewa, jika terdapat penyewa baru maka akan di masukan kedalam data penyewa, data-data yang di perlukan adalah nama, alamat, no telepon serta kode penyewa.

Tiap transaksi yang terjadi penyewa menerima bukti pembayaran berupa bukti penyewaan untuk pihak Toko dapat berupa salinan (untuk back up) serta laporan transaksi penyewaan tiap bulannya.

3.1.2. Visi dan Misi

Visi dan Misi Perusahaan

Visi PT. Musi Car Rental adalah Menjadikan perusahaan transportasi terkemuka di tingkat nasional, regional dan internasional.

Misi PT. Musi car rental adalah:

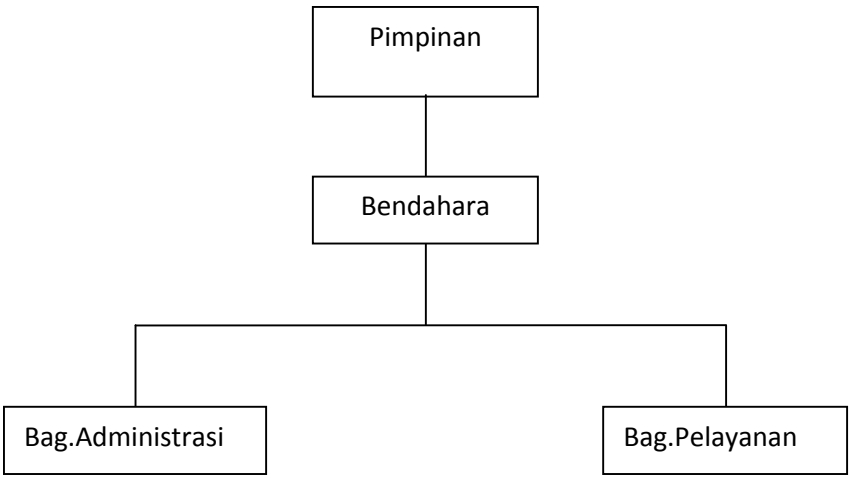
a. Mengantisipasi kecenderungan pasar dan kebutuhan pelanggan.
b. Mengutamakan manajemen yang professional dan berintegritas.
c. Mengembangkan sumber daya manusia serta memberikan penghargaan atas prestasi kerja.

d. Mengembangkan budaya inovatif.

e. Menerapkan tata kelola perusahaan yang baik.

3.1.3. Struktur Organisasi

Struktur organisasi PT. Musi car rental adalah sebagai berikut:



Gambar 3.1. Struktur Organisasi

3.1.4. Deskripsi Tugas

Dalam Striktur Organisasi ini akan di jelaskan pula mengenai tugas dan tanggung jawab dari setiap jabatan atau kedudukan:

A. Pimpinan

Adapun tugas dan tanggung jawab seorang Pimpinan yaitu :

1. Bertugas mengawasi, mengamati dan mengatur setiap karyawannya dalam melakukan pekerjaan.
2. Bertanggung jawab terhadap maju dan mundurnya perusahaan yang di pimpinnya.

B. Bendahara

Sebagai bendahara didalam suatu perusahaan mempunyai tugas dan tanggung jawab sebagai berikut:

1. Menerima administrasi penyewaan mobil serta membuat laporan mengenai keuangan perusahaan.
2. Mengatur dan membuat pembukuan perusahaan yang harus dipertanggung jawabkan laporan kepada pimpinan.

C. Bagian Administrasi

Adapun tugas dan tanggung jawab seorang bagian administrasi sebagai berikut:

1. Memberikan informasi atau tata cara didalam penyewaan mobil
2. Menentukan tarip atau harga sewa dan denda serta peraturan atau ketentuan yang berlaku di dalam tata cara penyewaan mobil. Selain itu bagian administrasi menpunyai tanggung jawab terhadap tugasnya dan kepada pimpinan.

D. Bagian Pelayanan

Adapun tugas dan tanggung jawab seorang bagian pelayanan sebagai berikut:

1. Memberikan pelayanan atau service yang memuaskan bagi pelanggan (baik baru maupun lama).
2. Member informasi mengenai spesifikasi kendaraan atau mobil yang akan disewa oleh pelanggan

Mengawasi keluar masuknya kendaraan atau mobil yang telah disewakan oleh pelanggan serta mencetak kembali kondisi kendaraan atau mobil yang telah dipulangkan atau dikembalikan oleh pelanggan serta bertanggung jawab kepada pimpinan terhadap tugas dan laporan.

3.2.Metode Penelitian

Berikut ini merupakan metode penelitian yang dilakukan dalam melaksanakan penelitian.

3.2.1. Desain Penelitian

Dalam melakukan penelitian, desain penelitian merupakan kerangka atau perincian prosedur kerja yang akan dilakukan pada waktu meneliti, sehingga diharapkan dapat memberikan gambaran dan arah mana yang akan dilakukan dalam melaksanakan penelitian tersebut, desain penelitian yang baik dapat memudahkan kita dalam melakukan penelitian dan sesuai dengan tujuan yang telah ditetapkan sebelumnya.

Sesuai dengan tujuan yang ingin dicapai dalam penelitian ini, penulis mengumpulkan data sebagai bahan laporan dengan menggunakan metode deskriptif, yaitu suatu metode yang pada tahap pertama penulis melakukan dengan cara mengumpulkan data dan bahan yang diperlukan terlebih dahulu dan pada tahap berikutnya, penulis mengolah dan membahas sampai pada suatu kesimpulan yang pada akhirnya dapat dibuat suatu laporan.

3.2.2. Jenis & Metode Pengumpulan Data

Metode yang digunakan penulis dalam pengumpulan data adalah dengan cara mengumpulkan data dari sumber data primer dan sumber data sekunder, berikut ini adalah penjelasan singkat mengenai metode yang penulis gunakan.

3.2.2.1. Sumber Data Primer

Data Primer adalah data yang diperoleh langsung dari unit pengamatan / responden penelitian. Data primer bisa di dapatkan dengan cara berikut:

1. Wawancara (*Interview*)

Metode wawancara merupakan sesi tanya jawab yang dilaksanakan penulis untuk mewawancarai dengan pihak yang terkait. Dalam teknis wawancara ini, penulis berperan sebagai pewawancara dan mewawancarai pihak-pihak yang berhubungan dengan penelitian yang penulis lakukan. Penulis melakukan wawancara langsung kepada Pemimpin PT. Musi Car Rental mengenai permasalahan yang dibahas yaitu mengenai aplikasi-aplikasi yang terdapat di *mobile phone* android.

Wawancara ini dilakukan untuk mengetahui beberapa kekurangan tentang aplikasi *mobile phone* android, baik itu masalah layanan penyewaan maupun interface pada aplikasi-aplikasi yang sudah ada.

2. Observasi

Melakukan penelitian secara langsung pada mobile phone android untuk mengetahui prosedur pelayanan apa saja yang di butuhkan untuk membuat aplikasi *e-rental*.

3.2.2.2. Sumber Data Sekunder

Data Sekunder adalah data yang digunakan untuk mendukung data primer salah satunya dengan mengumpulkan dokumen-dokumen yang dianggap penting untuk penulisan laporan penelitian.

Studi Pustaka

Metode yang digunakan dengan cara mencari referensi buku-buku yang dapat diperlukan sebagai landasan teori maupun yang berhubungan dengan masalah pembuatan program aplikasi. Adapun buku referensi yang penulis gunakan untuk penyusunan landasan teori adalah buku-buku yang berasal dari perpustakaan, sebagai contoh adalah buku karangan Jogiyanto.Albahra.

3.2.3. Metode Pendekatan & Pengembangan Sistem

Metode pendekatan dan pengembangan sistem berisikan beberapa teknik yang digunakan dalam penelitian ini, diantaranya metode pengembangan sistem, metode pendekatan sistem, alat bantu analisis dan perancangan, dan pengujian software. Di bawah ini akan diuraikan tentang pemahaman dari metode tersebut.

3.2.3.1. Metode Pendekatan Sistem

Metode pendekatan sistem merupakan serangkaian langkah-langkah pemecahan masalah yang memastikan bahwa masalah dipahami, solusi alternative dipertimbangkan dan solusi yang dipilih bekerja. Proses pemecahan masalah secara sistematis bermula dari John Dewey, seorang professor filosofi di Columbia University pada awal abad ini. Dalam bukunya 1910 diidentifikasi ada 3 seri penilaian dalam memecahkan suatu kontroversi yang memadai :

1. Mengenali kontroversi.
2. Menimbang klaim alternatif.
3. Membentuk penilaian.

Metode pendekatan sistem yang akan digunakan dalam penelitian ini adalah pendekatan objek. Pendekatan Objek merupakan paradigma pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek, dimana setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya. Pendekatan objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem.

Pendekatan Objek memiliki beberapa keuntungan, antara lain:

1. Maintenance. Program lebih mudah dibaca dan dipahami.

2. Pengubahan program (berupa penambahan ataupun penghapusan fitur tertentu). Perubahan ini antara lain menyangkut penambahan dan penghapusan dalam suatu database program misalnya.

3. Dapat digunakannya objek-objek sesering yang diinginkan.

3.2.3.2 Metode Pengembangan Sistem

Dalam perancangan sebuah sistem hal yang sangat penting dan harus diperhatikan agar menghasilkan suatu sistem yang baik dan dapat melengkapi informasi yang dibutuhkan merupakan kajian teknis formal. Untuk itu digunakan metode-metode perancangan sistem. Pengembangan sistem yang penulis gunakan dalam penelitian ini adalah dengan menggunakan pendekatan model SDLC (*System Development Life Cycle*).

Siklus hidup pengembangan sistem (Systems Development Life Cycle) adalah proses pembuatan dan pengubahan sistem serta model dan metedologi yang digunakan untuk mengembangkan sistem-sistem tersebut. Konsep ini umumnya merujuk pada sistem komputer atau system informasi.

SDLC berfungsi untuk menggambarkan tahapan-tahapan utama dan langkah-langkah dari setiap tahapan yang secara garis besar terbagi dalam tiga kegiatan utama, yaitu:

a. *Analysis*

Digunakann untuk membuat keputusan, apabila sistem saat ini mempunyai masalah atau sudah tidak berfungsi secara baik, dan hasil analisisnya digunakan sebagai dasar untuk memperbaiki system.

b. *Design* (perancangan)

Tahapan perancangan / desain memiliki tujuan untuk mendesain sistem baru yang dapat menyelesaikan masalah-masalah yang dihadapi perusahaan yang diperoleh dari pemlihan alternatif sistem yang terbaik. Kegiatan yang dilakukan dalam tahapan perancangan ini meliputi perancangan output, input, dan file.

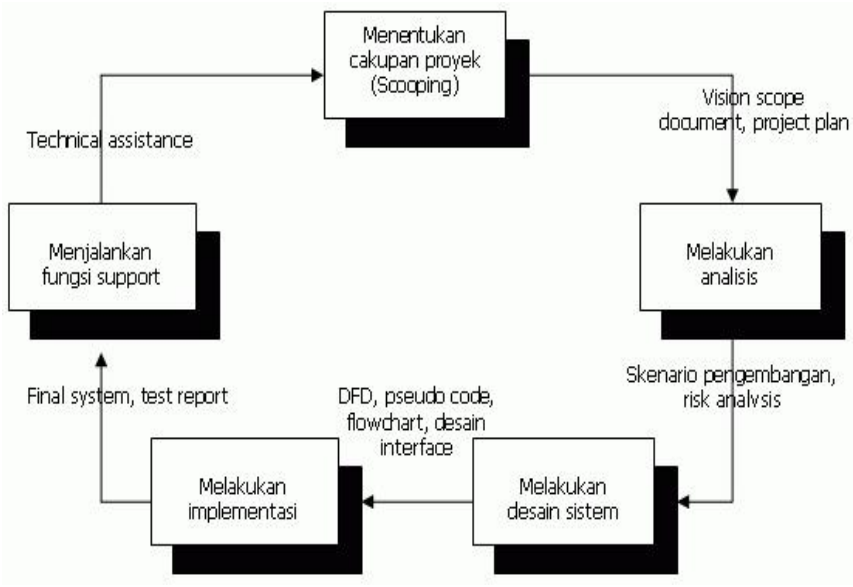
c. *Implementation*

Memiliki beberapa tujuan yaitu untuk melakukan kegiatan spesifikasi rancangan logikal ke dalam kegiatan yang sebenarnya dari sistem informasi yang akan dibangunnya atau dikembangkan nya, lalu mengimplementasikan sistem yang baru tersebut ke dalam salah satu bahasa pemrograman yang paling sesuai. Pada tahap ini juga harus menjamin bahwa sistem yang baru dapat berjalan secara

optimal. Kegiatan yang dilakukan dalam tahap implementasi ini adalah pembuatan program dan test data, pelatihan dan pergantian sistem.

Dalam penerapan tahapan pengembangan sistem informasi penulis menggunakan sistem pengembangan *prototype*. *Prototyping* merupakan salah satu metode pengembangan perangat lunak yang banyak digunakan. Dengan metode prototyping ini pengembang dan pelanggan dapat saling berinteraksi selama proses pembuatan sistem. Sering terjadi seorang pelanggan hanya mendefinisikan secara umum apa yang dikehendakinya tanpa menyebutkan secara detal output apa saja yang dibutuhkan, pemrosesan dan data-data apa saja yang dibutuhkan. Sebaliknya disisi pengembang kurang memperhatikan efesiensi algoritma, kemampuan sistem operasi dan *interface* yang menghubungkan manusia dan komputer.

Untuk mengatasi ketidakserasian antara pelanggan dan pengembang , maka harus dibutuhakan kerjasama yanga baik diantara keduanya sehingga pengembang akan mengetahui dengan benar apa yang diinginkan dengan tidak mengesampingkan segi-segi teknis dan pelanggan akan mengetahui proses-proses dalm menyelasaikan system yang diinginkan. Dengan demikian akan menghasilkan sistem sesuai dengan jadwal waktu penyelesaian yang telah ditentukan. Kunci agar model *prototype* ini berhasil dengan baik adalah dengan mendefinisikan aturan-aturan main pada saat awal, yaitu pelanggan dan pengembang harus setuju bahwa prototype dibangun untuk mendefinisikan kebutuhan. Prototype akan dihilangkan sebagian atau seluruhnya dan perangkat lunak aktual actual direkayasa dengan kualitas dan implementasi yang sudah ditentukan.



Gambar 3.2 Proses Prototype

a) Pengumpulan kebutuhan

Pelanggan dan pengembang bersama-sama mendefinisikan format seluruh perangkat lunak, mengidentifikasikan semua kebutuhan, dan garis besar sistem yang akan dibuat.

b) Membangun *prototyping*

Membangun prototyping dengan membuat perancangan sementara yang berfokuspada penyajian kepada pelanggan (misalnya dengan membuat input dan format output)

c) Evaluasi *protoptyping*

Evaluasi ini dilakukan oleh pelanggan apakah prototyping yang sudah dibangun sudah sesuai dengan keinginann pelanggan. Jika sudah sesuai maka langkah 4 akan diambil.Jika tidak prototyping direvisi dengan mengulang langkah 1, 2 , dan 3.

d) Mengkodekan Sistem

Dalam tahap ini prototyping yang sudah disepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai.

e) Menguji Sistem

Setelah sistem sudah menjadi suatu perangkat lunak yang siap pakai, harus dites dahulu sebelum digunakan. Pengujian ini dilakukan dengan White Box, Black Box.

f) Evaluasi Sistem

Pelanggan mengevaluasi apakah sistem yang sudah jadi sudah sesuai dengan yang diharapkan . Jika ya, langkah 7 dilakukan; jika tidak, ulangi langkah 4 dan 5.

g) Menggunakan sistem

Perangkat lunak yang telah diuji dan diterima pelanggan siap untuk digunakan .

Adapun alasan mengapa penulis menggunakan metode ini karena metode ini mempunyai beberapa keunggulan diantaranya :

Keunggulan *prototyping* adalah:

1. Adanya komunikasi yang baik antara pengembang dan pelanggan
2. Pengembang dapat bekerja lebih baik dalam menentukan kebutuhan pelanggan
3. Pelanggan berperan aktif dalam pengembangan sistem
4. Lebih menghemat waktu dalam pengembangan sistem
5. Penerapan menjadi lebih mudah karena pemakai mengetahui apa yang diharapkannya.

Kelemahan *prototyping* adalah :

1. Pelanggan kadang tidak melihat atau menyadari bahwa perangkat lunak yang ada belum mencantumkan kualitas perangkat lunak secara keseluruhan dan juga belum memikirkan kemampuan pemeliharaan untuk jangka waktu lama.
2. Pengembang biasanya ingin cepat menyelesaikan proyek. Sehingga menggunakan algoritma dan bahasa pemrograman yang sederhana untuk membuat prototyping lebih cepat selesai tanpa memikirkan lebih lanjut bahwa program tersebut hanya merupakan cetak biru sistem .
3. Hubungan pelanggan dengan komputer yang disediakan mungkin tidak mencerminkan teknik perancangan yang baik

Prototyping bekerja dengan baik pada penerapan-penerapan yang berciri sebagai berikut:

1. Resiko tinggi yaitu untuk masalah-masalah yang tidak terstruktur dengan baik, ada perubahan yang besar dari waktu ke waktu, dan adanya persyaratan data yang tidak menentu.
2. Perlunya penyelesaian yang cepat
3. Perilaku pemakai yang sulit ditebak
4. Sistem yang inovatif. Sistem tersebut membutuhkan cara penyelesaian masalah dan penggunaan perangkat keras yang mutakhir
5. Perkiraan tahap penggunaan sistem yang pendek

3.2.3.3. Alat Bantu Analisis dan Perancangan

Berikut ini adalah alat bantu analisis yang akan digunakan dalam merancang aplikasi penguncian file:

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C.

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti 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).

Konsepsi Dasar UML

Dari berbagai penjelasan rumit yang terdapat di dokumen dan buku-buku UML. Sebenarnya konsepsi dasar UML bisa kita rangkumkan dalam gambar dibawah.

Major Area	*View*	*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	statechart 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
model management	model management view	class diagram	package, subsystem, model
extensibility	all	all	constraint, stereotype, tagged values

Gambar 3.3. Konsepsi Dasar UML

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

A. *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 *test case* untuk semua *feature* 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 dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. 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.

B. Activity diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sama seperti *state*, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-

proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal. *Activity diagram* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

C. 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 atar 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.

Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan *icon* khusus untuk objek *boundary, controller* dan *persistent entity*.

D. Class Diagram

Class diagram 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.

Atribut dan metoda dapat memiliki salah satu sifat berikut :

a. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan
b. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya
c. *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.

E. *Deployment* Diagram

Deployment/physical diagram menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal Sebuah *node* adalah server, *workstation*, atau piranti keras lain yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya. Hubungan antar *node* (misalnya TCP/IP) dan *requirement* dapat juga didefinisikan dalam diagram ini.

Perangkat Yang Mendukung UML

Saat ini banyak sekali *tool* pendesainan yang mendukung UML, baik itu *tool* komersial maupun *opensource*. Beberapa diantaranya adalah:

1. *Rational Rose*
2. *Astah*
3. *Object Domain*
4. *Jvision*
5. *Objecteering*
6. *MagicDraw*
7. *Visual Object Modeller*

3.2.4. Pengujian Software

Software testing adalah investigasi yang dilakukan kepada software untuk mengetahui kualitas suatu produk. dilakukan untuk mengeliminasi *defect* yang terjadi setelah *software* diluncurkan, pengujian *software* sendiri tidak akan pernah secara penuh mengenali *defect* pada suatu *software.black box testing* dan *white box testing* adalah metode yang dikenal untuk menguji *software*.

Pengujian *black-box* sering disebut juga dengan pengujian tingkah laku *(behavoiur testing)* yang lebih terfokus kepada kebutuhan fungsional dari perangkat lunak, pengujian *black-box* memungkinkan pembuat perangkat lunak untuk menentukan kondisi yang terjadi untuk suatu masukan yang akan menjalankan semua kebutuhan fungsional (Pressman 2002:551).

Pengujian *black box* merupakan pendekatan komplementer dari teknik *white box*, karena pengujian *black box* diharapkan mampu mengungkap kelas kesalahan yang lebih luas dibandingkan teknik *white box.* Pengujian *black box* berfokus pada pengujian persyaratan fungsional perangkat lunak, untuk mendapatkan serangkaian kondisi input yang sesuai dengan persyaratan fungsional suatu program. Pengujian *black box* berusaha menemukan kesalahan dalam kategori :

1. fungsi-fungsi yang tidak benar atau hilang
2. kesalahan *interface*
3. kesalahan dalam struktur data atau akses database eksternal
4. kesalahan kinerja
5. inisialisasi dan kesalahan terminasi.