BAB III

OBJEK DAN METODE PENELITIAN

3.1. Objek Penelitian

Objek yang dijadikan penelitian didalam penulisan skripsi ini adalah PT. PRIMARINDO ASIA INFRASTRUCTURE, Tbk. Berikut adalah sejarah singkat perusahaan, visi dan misi, struktur organisasi dan deskripsi tugas yang penulis peroleh langsung dari tempat yang penulis teliti.

3.1.1. Sejarah Singkat Perusahaan

PT. PRIMARINDO ASIA INFRASTRUCTURE, Tbk. didirikan di Bandung berdasarkan Akta No. 7 tanggal 1 Juli 1988 dan Notaris Nany Sukarja, S. H. Akta Pendirian Perusahaan telah disahkan oleh Menteri Kehakiman Republik Indonesia dalam Surat Keputusan No. C2-9967-HT.01.01.TH 1988 tanggal 31 Oktober 1988 serta diumumkan dalam Berita Negara Republik Indonesia No. 53 tanggal 2 Juli 1991, tambahan No. 1851. Anggaran Dasar Perusahaan mengalami beberapa kali perubahan, terakhir dengan akta No. 16 tanggal 23 Juni 1999 dari Notaris Raharti Sudjardjati, SH, mengenai ketentuan jabatan komisaris dan direksi perusahaan. Akta perubahan ini telah disetujui oleh Menteri Kehakiman Republik Indonesia sesuai Surat keputusan No. C-1183-HT.01.04.TH.2000 tanggal 2 Februari 2000.

Sesuai dengan pasal 3 Anggaran Dasar Perusahaan, ruang lingkup kegiatan perusahaan meliputi bidang usaha infrastruktur dan industri.

Perusahaan mulai berproduksi secara komersial pada tanggal 1 Oktober 1989. Kegiatan perusahaan dari sejak pendirian sampai saat ini meliputi industri alas kaki, khususnya produksi sepatu olah raga dan yang berhubungan dengan pengolahan bahan-bahan dasar pembuatan sepatu olah raga tersebut. Perusahaan menerima pesanan dan memproduksi sepatu untuk merk seperti *YK, FOS International, Jason/Dockers* dan *Samji*. Perusahaan juga memproduksi sepatu lokal dengan merk *Tomkins.*

Perusahaan berdomisili di Jakarta dengan pabrik berlokasi di Bandung, Jawa Barat. Kantor pusat perusahaan beralamat di Gedung Dana Pensiun – Bank Mandiri Lt. 3A,Jl. Tanjung Karang No. 3-4A, Jakarta. Jumlah karyawan perusahaan sebanyak 3.624 karyawan pada tahun 2008 dan sebanyak 3.294 karyawan pada tahun 2009.

3.1.2. Visi dan Misi perusahaan

Adapun Visi dan Misi dari PT. PRIMARINDO ASIA INFRASTRUCTURE Tbk, yaitu :

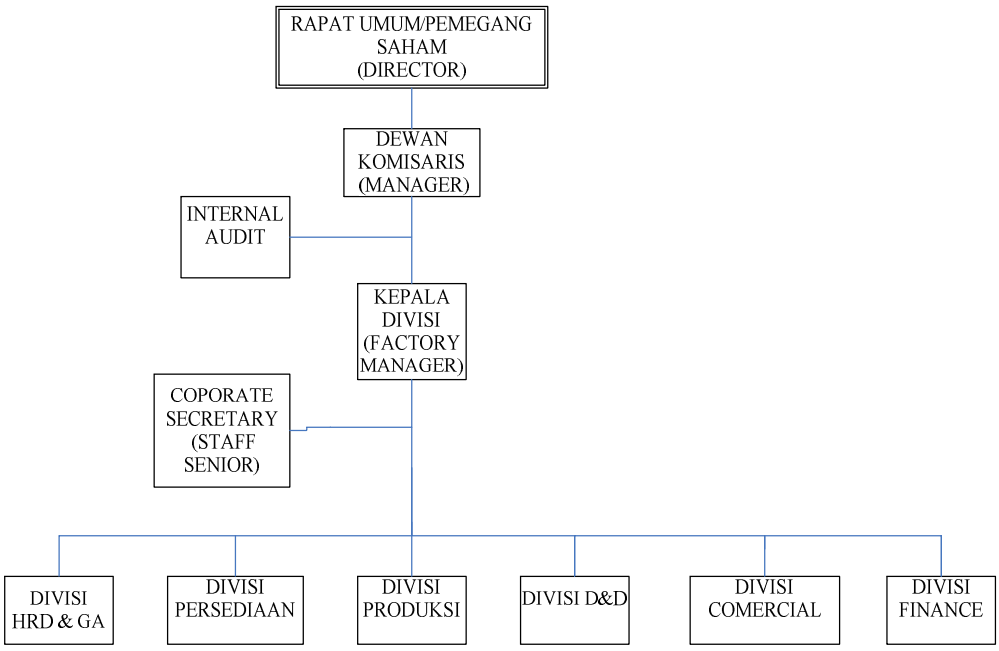
1. **Visi**

   " To Be Number One Footwear Industry In the World ".

2. ***2.*** **Misi**

   Menjadi industri pembuatan sepatu modern dengan kualitas terbaik dan menjadi rekan terpercaya, masa kini dan masa yang akan datang.

3.1.3. Struktur Organisasi Perusahaan



Gambar 3.1. Struktur Organisasi

3.1.4. Deskripsi Tugas

1. **Top Manajemen**

   Nama Jabatan : Direktur/Pemegang Saham

   Nama Pangkat : *Director*

   Gol : I

   Penjelasan singkat

   Pada Top Manajemen ini merupakan tingkatan manajemen dalam organisasi perusahaan yang bertanggung jawab atas kegiatan operasional, keuangan dan karyawan secara menyeluruh. Pada Top Manajemen merumuskan tujuan, rencana dan kebijaksanaan umum serta mengawasi seluruh kegiatan perusahaan. Top Manajemen

bertugas untuk pengambilan keputusan penting yang mempengaruhi perusahaan.

2. Divisi

Nama Jabatan **:** Kepala Divisi

Nama Pangkat **:** *Factory Manager*

Gol **:** IV

Penjelasan Singkat

Tingkatan manajemen ini membantu Top manajemen dalam merencanakan dan mendistribusikan pekerjaan kepada tingkatan manajemen yang berada di bawahnya sesuai dengan fungsi masing-masing dan mengawasi pelaksanaannya agar tidak menyimpang dari tujuan, rencana dan kebijakan yang telah ditetapkan perusahaan.

a) Divisi HRD & GA

Divisi ini melakukan pencarian dan penyediaan Sumber Daya Manusia, administrasi data karyawan, pengembangan kualitas Sumber Daya Manusia, memelihara gedung dan peralatan, menjaga keamanan, memelihara kendaraan, melayani kerumahtanggaan dan mengurus ijin-ijin perusahaan.

b) Divisi Persediaan

Divisi ini menyediakan bahan - bahan yang akan di produksi dan mengatur persediaan barang yang ada sehingga persediaan tidak habis.

c) Divisi Produksi

Divisi ini melaksanakan perencanaan produksi, kegiatan pabrikasi, evaluasi kinerja produksi, pemeliharaan dan perawatan mesin-mesin yang menunjang proses produksi.

d) Divisi D&D

Divisi ini melaksanakan tugasnya membuat sampel sepatu untuk produksi.

e) Divisi *Commercial*

Divisi ini bertugas untuk mempromosikan, memasarkan dan mendistribusikan produk serta menyiapkan material yang berhubungan dengan supplier.

f) Divisi *Finance*

Divisi ini melaksanakan perencanaan keuangan, mengawasi realisasi, pengumpulan, pengolahan data sehingga menghasilkan keuangan yang berguna bagi manajemen dalam mengendalikan kegiatan perusahaan dan pengambilan keputusan.

3. Dewan

Nama Jabatan : Dewan Komisaris

Nama Pangkat : *Manager*

Gol : II

Penjelasan Singkat

Tingkatan Manajemen ini merupakan pecahan dari Divisi yang membantu tingkatan manajemen di atasnya untuk mencapai tujuan perusahaan dengan melaksanakan kegiatan teknis yang khusus di bebankan oleh tingkatan Divisi.

4. Bagian

Nama Jabatan : *Internal Audit*

Nama Pangkat : *Internal Audit*

Gol : III

Penjelasan Singkat

Tingkatan manajemen ini berada satu tingkatan di bawah tingkatan Dewan Komisaris, tingkatan ini bertugas melakukan pengecekan dana masuk dan keluar, *cuting* barang dan mengatur keuangan.

5. Seketariat

Nama Jabatan : *Coporate Secretary*

Nama Pangkat : Staf Senior

Gol : II

Penjelasan Singkat

Tingkatan manajemen ini berada dibawah satu tingkatan Divisi, dan bertugas mencatat semua pekerjaan, mencatat dan mengatur jadwal.

3.2. Metode Penelitian

Metode penelitian merupakan cara ilmiah untuk mendapatkan data dengan tujuan dan kegunaan tertentu.

3.2.1. Desain Penelitian

Sesuai dengan tujuan yang ingin dicapai dalam penelitian ini, maka metode yang akan digunakan adalah metode *deskriptif* dan terapan. *Deskriptif* adalah metode penelitian yang memberikan gambaran tentang sifat individu, keadaan, gejala suatu objek, dimana dalam penelitian ini untuk memperoleh gambaran tentang perancangan sistem informasi berbasis *web* pada produk sepatu *Tomkins,* masalah yang dihadapi dan pemecahannya. Sedangkan metode terapan adalah menguji dan mengevaluasi kemampuan teori yang diterapkan dalam memecahkan masalah-masalah kehidupan praktis.

3.2.2. Jenis dan Metode Pengumpulan Data

Supaya data dan informasi yang diperoleh sesuai dengan PT. PRIMARINDO ASIA INFRASTRUCTURE, Tbk, maka penulis menggunakan metode penelitian lapangan yaitu penelitian yang langsung terjun ke tempat untuk dijadikan objek penelitian.

3.2.2.1. Jenis Data

Dalam penelitian ini, terdapat dua jenis data yang digunakan yaitu sebagai berikut :

1. Sumber Data Primer adalah data yang diperoleh langsung dari unit pengamatan atau responden penelitian.
2. Sumber Data Sekunder adalah data yang digunakan untuk mendukung data primer, merupakan jenis data yang sudah diolah terlebih dahulu oleh pihak pertama.

3.2.2.2. Metode Pengumpulan Data

Wawancara yaitu mengumpulkan data melalui tatap muka dan tanya jawab langsung antara pewawancara (pengumpul data) dengan responden (sumber data).

Melalui wawancara tujuan dilakukan penelitian menjadi mudah, untuk mengidentifikasi dan merumuskan masalah menjadi lebih terstruktur.

Observasi yang dilakukan di PT. PRIMARINDO ASIA INFRASTRUCTURE Tbk ternyata dalam pelaporan penjualan dari luar pulau jawa ke perusahaan sering terlambat dan konsumen atau pelanggan merasa *counter Tomkins* jauh. Untuk itu dibuatkan *website* untuk memperpercepat proses laporan penjualan atau pemesanan ke perusahaan dan mempromosikan secara menyeluruh sehingga mempermudah karyawan dalam bekerja dan konsumen yang melakukan transaksi pembelian.

Responden yang diwawancara yaitu HRD & GA sepatu *Tomkins* yang berada di PT. PRIMARINDO ASIA

INFRASTRUCTURE Tbk. Identitas yang dipaparkan hanya intinya saja yaitu :

a) Jenis Kelamin : Laki-laki
b) Umur : ± 29 tahun
c) Jabatan : HRD

Sedangkan pihak pewawancara identitasnya

a) Jenis Kelamin : Pria
b) Umur : 23 tahun
c) Jabatan : Pelajar/Mahasiswa Unikom

Sifat wawancara terbuka dikarenakan data yang diambil tidak merugikan perusahaan.

Merupakan kumpulan data yang terkait atau yang berhubungan dengan objek penelitian tersebut diantaranya :

1) Data produk
2) Data costumer
3) Data gender
4) Data size
5) Struktur Organisasi
6) Contoh Laporan Penjualan

3.2.3. Metode Pendekatan dan Pengembangan Sistem

3.2.3.1. Metode Pendekatan Sistem

1) Object Oriented Analysis and Design (OOAD)

Pendekatan baru untuk pengembangan sistem, sering disebut sebagai pendekatan ketiga setelah pendekatan yang berorientasi data dan berorientasi proses. *OOAD* adalah metode pengembangan sistem yang lebih menekankan pada objek dibandingkan dengan data atau proses. Ada beberapa ciri khas dari pendekatan ini yaitu *object, Inheritance* dan *object class*

Objek adalah benda secara fisik atau konseptual, yang dapat kita temui disekeliling kita. *Hardware, software,* dokumen atau manusia dan bahkan konsep semuanya adalah contoh obyek. Untuk kepentingan memodelkan perusahaannya, seorang kepala eksekutif akan melihat gedung, karyawan, divisi, dokumen dan keuntungan sebagai obyek. Seorang teknisi mesin akan melihat ban, pintu, mesin, laju tertinggi dan banyaknya bahan bakar sebagai sebuah obyek. Dan seorang *software engineer* akan memandang tumpukan, antrian, jendela dan *check box* sebagai sebuah obyek. ***Sumber: (Bettyudha. Materi UML Dan Use Case. Tesis Program Magister.P20).***

Sebuah obyek memiliki *keadaan (state)* dan *perilaku (behavior).* State dari sebuah obyek adalah kondisi obyek tersebut atau himpunan dari keadaan yang menggambarkan

obyek tersebut. Sebagai contoh, bola lampu adalah obyek, dan salah satu keadaan nyala atau tidak nyala adalah state dari obyek bola lampu tersebut. ***Sumber: (Bettyudha. Materi UML Dan Use Case. Tesis Program Magister.P20).***

State dinyatakan dengan nilai dari sebuah atribut obyeknya. Atribut adalah nilai internal suatu obyek yang mencerminkan antara lain karakteristik obyek, kondisi obyek, kondisi sesaat, koneksi dengan obyek lain dan identitas. Perubahan state dicerminkan oleh perilaku *(behavior*) obyek tersebut. ***Sumber: (Bettyudha. Materi UML Dan Use Case. Tesis Program Magister.P21).***

Behavior suatu obyek mendefinisikan bagaimana sebuah obyek bertindak (beraksi) dan memberikan reaksi. Behavior ditentukan oleh himpunan semua atau beberapa operation yang dapat dilakukan dalam obyek itu sendiri. *Behavior* dari sebuah obyek dicerminkan oleh *interface, service dan method* dari obyek tersebut. ***Sumber: (Bettyudha. Materi UML Dan Use Case. Tesis Program Magister.P21).***

Interface adalah pintu untuk mengakses *service* obyek. *Service* adalah fungsi yang bisa diemban obyek. *Method* adalah mekanisme internal obyek yang mencerminkan perilaku (*behavior*) obyek tersebut. Sebagai contoh, jika p*rinter* merupakan sebuah obyek maka perilaku (*behavior*) atau *service* nya mencetak apapun yang dia terima. ***Sumber: (Bettyudha. Materi UML Dan Use Case. Tesis Program Magister.P21).***

Object class adalah sekumpulan objek yang berbagi struktur yang sama dan perilaku yang sama. *Inheritance. Properti* yang muncul ketika tipe entitas atau *object class* disusun secara hirarki dan setiap tipe entitas atau *object class* menerima atau mewarisi atribut dan metode dari pendahulunya. ***Sumber: (Bettyudha. Materi UML Dan Use Case. Tesis Program Magister.P21).***

2) Unified Modelling language (UML)

Unified Modelling Language (*UML*) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk menentukan, visualisasi, merancang dan mendokumentasikan *artifact* dari sistem *software*, untuk memodelkan bisnis dan sistem *non software* lainnya. *UML* merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan sistem yang besar dan kompleks. ***Sumber: (Bettyudha. Materi UML Dan Use Case. Tesis Program Magister.P27).***

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, VB.NET*. Walaupun demikian, *UML* tetap dapat digunakan untuk modeling aplikasi prosedural dalam *VB*

atau C. ***Sumber: (Bettyudha. Materi UML Dan Use Case. Tesis Program Magister.P27).***

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).* ***Sumber: (Bettyudha. Materi UML Dan Use Case. Tesis Program Magister.P27).***

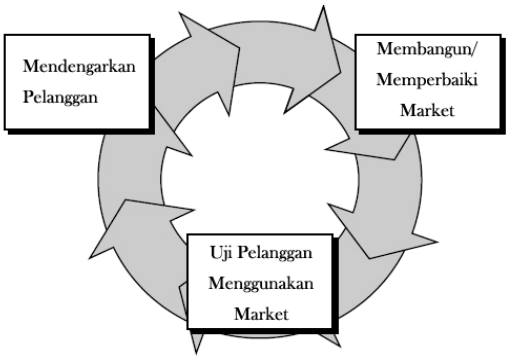
Sejarah *UML* sendiri cukup panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: metodologi *Booch*, metodologi *Coad* , metodologi *OOSE*, metodologi *OMT*, metodologi *Shlaer-Mellor*, metodologi *Wirfs-Brock*, dan sebagainya. Masa itu terkenal dengan masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila kita bekerja sama dengan group/perusahaan lain yang menggunakan

metodologi yang berlainan. ***Sumber: (Bettyudha. Materi UML Dan Use Case. Tesis Program Magister.P28).***

3.2.3.2. Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan adalah *prototype* paradigma yang merupakan metode pengembangan yang berfungsi sebagai sebuah mekanisme untuk mengidentifikasi kebutuhan perangkat lunak, *prototype* bisa menjadi paradigma yang efektif bagi rekayasa perangkat lunak.

Prototype paradigma dimulai dengan mengumpulkan kebutuhan. Pengembangan dan pelanggan bertemu dan mendefinisikan obyektif keseluruhan dari perangkat lunak, mengidentifikasi segala kebutuhan yang diketahui, dan area garis besar dimana definisi lebih jauh merupakan keharusan kemudian dilakukan perancangan kilat, perancangan kilat berfokus pada penyajian dari aspek-aspek perangkat lunak tersebut yang akan nampak bagi pelanggan/pemakai (contohnya pendekatan input dan format output).



Gambar 3.2. *Prototype*

Sumber : Roger S. Pressman. 2002 . *Rekayasa Perangkat Lunak*. Andi .Yogyakarta.

3.2.3.3. Alat Bantu Dan Analisis

1) Actor

Actor menggambarkan segala pengguna *software* aplikasi (*user*). Actor memberikan suatu gambaran jelas tentang apa yang harus dikerjakan *software* aplikasi. Sebagai contoh sebuah *actor* dapat memberikan input kedalam dan menerima informasi dari *software* aplikasi, perlu dicatat bahwa sebuah a*ctor* berinteraksi dengan *use case*, tetapi tidak memiliki kontrol atas *use case*. Sebuah *actor* mungkin seorang manusia, satu *device*, *hardware* atau sistem informasi lainnya. ***Sumber : (http://erwan2213.blogspot.com/2009/06/uml-unified-modelling-language-rpl.html).***

2) Use Case

Use case menjelaskan urutan kegiatan yang dilakukan *actor* dan sistem untuk mencapai suatu tujuan tertentu. Walaupun menjelaskan kegiatan, namun *use case* hanya menjelaskan apa yang dilakukan oleh *actor* dan sistem bukan bagaimana *actor* dan sistem melakukan kegiatan tersebut.

a) *Use-case Konkret* adalah *use case* yang dibuat langsung karena keperluan *actor*. *Actor* dapat melihat dan berinisiatif terhadapnya

b) *Use-case Abstrak* adalah *use case* yang tidak pernah berdiri sendiri. *Use case* abstrak senantiasa termasuk

didalam (*include*), diperluas dari (*extend*) atau memperumum (*generalize*) *use case* lainnya.

Untuk menggambarkannya dalam *use case* model biasanya digunakan *association relationship* yang memiliki *stereotype include, extend* atau *generalization relationship*. Hubungan *include* menggambarkan bahwa suatu *use case* seluruhnya meliputi fungsionalitas dari *use case* lainnya. Hubungan *extend* antar *use case* berarti bahwa satu *use case* merupakan tambahan fungsionalitas dari *use case* yang lain jika kondisi atau syarat tertentu terpenuhi. ***Sumber : (http://erwan2213.blogspot.com/2009/06/uml-unified-modelling-language-rpl.html).***

3) Class

Class merupakan pembentuk utama dari sistem berorientasi obyek, karena *class* menunjukkan kumpulan obyek yang memiliki atribut dan operasi yang sama. *Class* digunakan untuk mengimplementasikan *interface*.

Class digunakan untuk mengabstraksikan elemen-elemen dari sistem yang sedang dibangun. *Class* bisa merepresentasikan baik perangkat lunak maupun perangkat keras, baik konsep maupun benda nyata.

Notasi *class* berbentuk persegi panjang berisi 3 bagian: persegi panjang paling atas untuk *nama class*, persegi panjang

paling bawah untuk operasi, dan persegi panjang ditengah untuk *atribut.*

Atribut digunakan untuk menyimpan informasi. Nama atribut menggunakan kata benda yang bisa dengan jelas merepresentasikan informasi yang tersimpan didalamnya. Operasi menunjukkan sesuatu yang bisa dilakukan oleh obyek dan menggunakan kata kerja. ***Sumber : (http://erwan2213.blogspot.com/2009/06/uml-unified-modelling-language-rpl.html).***

4) *Statechart* Diagram

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu *state* ke *state* lainnya) suatu objek pada sistem sebagai akibat dari stimuli yang diterima. Pada umumnya *statechart* diagram menggambarkan *class* tertentu (satu *class* dapat memiliki lebih dari satu *statechart* diagram).

Dalam *UML*, *state* digambarkan berbentuk segiempat dengan sudut membulat dan memiliki nama sesuai kondisinya saat itu. Transisi antar *state* umumnya memiliki kondisi *guard* yang merupakan syarat terjadinya transisi yang bersangkutan, dituliskan dalam kurung siku. *Action* yang dilakukan sebagai akibat dari *event* tertentu dituliskan dengan diawali garis miring.

Titik awal dan akhir digambarkan berbentuk lingkaran berwarna penuh dan berwarna setengah. ***Sumber :***

(http://erwan2213.blogspot.com/2009/06/uml-unified-modelling-language-rpl.html).

5) *Activity* Diagram

Menggambarkan aliran fungsionalitas dalam suatu sistem. Dapat digunakan dalam pemodelan bisnis untuk menunjukkan *business workflow*. Atau juga digunakan dalam analisa kebutuhan untuk menggambarkan aliran kejadian melalui suatu *use case*. Mendefinisikan dimana *workflow* dimulai, dimana berhentinya, aktifitas apa yang terjadi selama *workflow*, bagaimana urutan kejadian aktifitas. Suatu aktifitas adalah suatu pekerjaan yang dilaksanakan selama *workflow*. ***Sumber : (http://erwan2213.blogspot.com/2009/06/uml-unified-modelling-language-rpl.html).***

6) *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 dariapa 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.

Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan icon khusus untuk *objek boundary*, *controller* dan *persistent entity*. ***Sumber : (http://erwan2213.blogspot.com/2009/06/uml-unified-modelling-language-rpl.html).***

7) *Collaboration* Diagram

Collaboration diagram juga menggambarkan interaksi antar objek seperti *sequence* diagram, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message*. Setiap *message* memiliki *sequence number*, di mana *message* dari level tertinggi memiliki nomor 1. *Messages* dari level yang sama memiliki prefiks yang sama. ***Sumber : (http://erwan2213.blogspot.com/2009/06/uml-unified-modelling-language-rpl.html).***

8) *Component* Diagram

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) di antaranya.

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. ***Sumber : (http://erwan2213.blogspot.com/2009/06/uml-unified-modelling-language-rpl.html).***

9) *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. ***Sumber : (http://erwan2213.blogspot.com/2009/06/uml-unified-modelling-language-rpl.html).***

3.2.4. Pengujian *Software*(Perangkat Lunak)

Pengujian adalah suatu proses pengeksekusian program yang bertujuan untuk menemukan kesalahan. Pengujian sebaiknya menemukan kesalahan yang tidak disengaja dan pengujian dinyatakan sukses jika berhasil memperbaiki kesalahan tersebut. Selain itu, pengujian juga bertujuan untuk menunjukkan kesesuaian fungsi-fungsi perangkat lunak dengan spesifikasinya. ***Sumber : (Retno.H. 2003. Perancangan Pengujian Perangkat Lunak Berorientasi Obyek: Berbasis Status (state-based testing), Jurnal Universitas Paramadina, Vol. 2 No. 2, Januari: P175-190)***.

Pengujian dapat dikategorikan atas :

1) Pengujian terhadap proses pengembangan sistem dan dokumen – dokumen pendukung. Proses berarti sejumlah aktivitas yang didukung oleh dokumen yang mendeskripsikan aktivitas - aktivitas. ***Sumber : (Retno.H. 2003. Perancangan Pengujian Perangkat Lunak Berorientasi Obyek: Berbasis Status (state-based testing), Jurnal Universitas Paramadina, Vol. 2 No. 2, Januari: P175-190).***
2) Pengujian terhadap analisis dan model perancangan. Dalam sistem berorientasi objek, pengujian model analisis dan perancangan adalah hal yang sangat penting. ***Sumber : (Retno.H. 2003. Perancangan Pengujian Perangkat Lunak***

Berorientasi Obyek: Berbasis Status (state-based testing), Jurnal Universitas Paramadina, Vol. 2 No. 2, Januari: P175-190).

3) Pengujian secara statik dan dinamik untuk implementasi. Tujuannya adalah mencari kesalahan sedini mungkin dalam proses, tetapi kesalahan dalam kode untuk sistem yang besar dan kompleks tidak dapat dihindarkan. Pengujian statik merupakan inspeksi kode untuk menemukan kesalahan *logic*. Pengujian dinamik merupakan eksekusi dengan data uji untuk menemukan kesalahan dalam kode. ***Sumber : (Retno.H. 2003. Perancangan Pengujian Perangkat Lunak Berorientasi Obyek: Berbasis Status (state-based testing), Jurnal Universitas Paramadina, Vol. 2 No. 2, Januari: P175-190).***

Perangkat lunak berorientasi objek berbeda dari perangkat lunak *procedural* (konvensional) dalam hal analisis, perancangan, struktur dan teknik-teknik pengembangannya. Bahasa pemrograman berorientasi objek mempunyai ciri-ciri adanya pembungkusan (*encapsulation*), keanekaragaman (*polymorphism*), dan pewarisan (*inheritance*) yang membutuhkan dukungan pengujian tertentu. ***Sumber : (Retno.H. 2003. Perancangan Pengujian Perangkat Lunak Berorientasi Obyek: Berbasis Status (state-based testing), Jurnal Universitas Paramadina, Vol. 2 No. 2, Januari: P175-190).***

Fokus pengujian perangkat lunak berorientasi objek dimulai pada hasil analisisnya, dilanjutkan pada hasil perancangannya, dan diakhir pada hasil pemrogramannya **(Rochimah, 1997).** Model yang

dihasilkan pada analisis dan perancangan harus diperiksa terutama dalam hal :

a) *Semantic correctness*, yaitu kesesuaian model dengan domain permasalahan di dunia nyata. Jika model merefleksikan dunia nyata secara akurat, berarti model tersebut benar secara semantic.
b) *Consistency*, yaitu kesesuaian kelas dengan objek turunannya maupun kesesuaian asosiasi kelas dengan kelas lainnya.

3.2.4.1. Model Uji (*Test Model*)

Berbagai model pengujian perangkat lunak berorientasi objek diusulkan oleh para peneliti. Setiap model mempunyai konstruksi atau aturan yang menjadi dasar dalam langkah-langkah pengujian. ***Sumber : (Retno.H. 2003. Perancangan Pengujian Perangkat Lunak Berorientasi Obyek: Berbasis Status (state-based testing), Jurnal Universitas Paramadina, Vol. 2 No. 2, Januari: P175-190).***

Black Box Testing memfokuskan pada keperluan fungsional dari *software.* Karena itu ujicoba *blackbox* memungkinkan pengembang *software* untuk membuat himpunan kondisi input yang akan melatih seluruh syarat-syarat fungsional suatu program. Ujicoba *blackbox* bukan merupakan alternatif dari ujicoba *whitebox*, tetapi merupakan pendekatan yang melengkapi untuk menemukan kesalahan lainnya, selain menggunakan metode *whitebox*. ***Sumber : (Ayuliana. 2009. Testing dan Implementasi. Maret).***

Ujicoba *blackbox* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya :

1) Fungsi-fungsi yang salah atau hilang.
2) Kesalahan *interface.*
3) Kesalahan dalam struktur data atau akses database eksternal.
4) Kesalahan performa, inisialisasi dan terminasi. ***Sumber : (Ayuliana. 2009. Testing dan Implementasi. Maret).***