BAB III

OBJEK DAN METODE PENELITIAN

3.1 Objek Penelitian

Dalam kesempatan ini, penulis mengadakan penelitian analisis sistem pada perusahaan PT. Kereta Api (Persero) Daop 2 Bandung di bagian reservasi tiket. Pada Bab ini akan dibahas sejarah secara singkat, misi dan visi, struktur organisasi, dan uraian tugas.

3.1.1 Sejarah Perusahaan

Tanpa peran raja-raja di jawa,mungkin sebutan Argogede,Argowilis dan Parahyangan serta KA-KA lain tidak akan terwujud.Maka keberadaan KeretaApi (KA) di Indonesia tidak lepas dari peran raja-raja di jawa.Dengan surat raja (diperkirakan Raja Mataram) NO.270 tanggal 28 Mei 1842,mengusulkan kepada pemerintah Hindia Belanda agar membangun jaringan KA (Kereta Api).

Setelah melakukan proses panjang,akhirnya pada tahun 1871 Gubernur Hindia Belanda Bose menyusun rancangan undang-undang pembuatan jalan Kereta api pemerintahan di Jawa,yang disahkan pada tanggal 6 April 1875,hal ini menandai lahirnya KA di bumi Nusantara secara *De-jure.*

Sedangkan secara *De-facto* jalan KA di Indonesia Spoorweg Mij (MSM) yang membangun jaringan Ka dari Kemijen-Temanggung Jawa Tengah sepanjang 25 kilometer.Pembangunan jaringan Ka ditandai pencangkulan pertama pembuatan badan jalan rel oleh Gubernur Jendral Belanda Mr.LA Baron Sloet Van De Beele tanggal 17 Juni 1864.Empat tahun kemudian ,17 Juni 1868 KA lintas Kemijen-Tamanggung dapat dioperasikan yang kemudian secara berturut-turut dilanjutkan dengan pembangunan jaringan KA diseluruh Jawa,Sumatra Utara,Aceh,Sumbar,dan Sumsel bahkan di Sulawesi,sebelum dipindahkan ke Burma saat penjajahan Jepang yang dikenal dengan Rhomusha.

Sejalan dengan perjalanan sejarah,berkat rahmat Allah SWT dan perjuangan rakyat Indonesia,proklamasi kemerdekaan RI tanggal 17 agustus dapat dikumandangkan yang diikuti pengambilalihan (nasionalisasi) secara resmi perusahaan-perusahaan penjajah oleh bangsa Indonesia.Nasionalisme perusahaan KA terjadi pada tanggal 28 September 1945 ,yang untuk selanjutnya dijadikan sebagai “Hari Kereta Api”.

Dengan nasionalisasi tersebut,maka berdirilah perusahaan KA pemerintah di Indonesia dengan bentuk perusahaan Djawatan KA Indonesia (DKARI) yang dipimpin oleh Direktur Jendral kepala jawatan Kereta Api (DDKA),Ir Muhamad Efendi Saleh.

Setelah berjalan hampir 8 tahun kepemimpinan tersebut digantikan oleh Ir R.Abu Prayitno bentuk perusahan,berubah menjadi Djwatan Kereta Api (DKA),hingga akhirnya pada tahun 1963 melalui peraturan pemerintah nomor 23/1963 perusahaan berubah status menjadi Perusahaan Negara Kereta Api

(PNKA). Karena status politik pada tahun 1965 hingga 1967 cukup genting dengan pecahnya peberontakan PKI,maka kepemimpinan di pegang oleh ABRI, dengan Brigjen Sentot Iskandardinata sebagai pemimpinannya,yang pada tahun 1968 digantikan oleh Ir.R.Soemali.

Tahun 1971,dengan PP NO.61/1971,status perusahaan kembali mengalami, PNKA diganti dengan PJKA(Perusahan Jawatan Kereta Api).Selanjutnya secara berturut-turut pimpinan PJKA dijabat oleh Ir.Partiaso.(1978- 1981).Drs.Anwar.Suprijadi,Msc(dengan sebutan Kaperjanka)

Pada tanggal 2 Januari 1991,dengan ditandai pemabacaan ikrar karyawan karyawati perumka.Status perusaan berubah kembali menjadi Perusaan Umum Kereta Api (PERUMKA)dengan PP 57/1990 tanggal 1 Oktober 1990,dengan direktur utama perumka dipimpin oleh,Ir.Soemino Eko Saputro (1995-1998).Pada tanggal 31 juli 1998 dirut perumka digantikan oleh Drs.Edie Haryoto.

Pada masa kepemimpinan Drs.Edie Hartoyo ini status perusahaan kembali mengalami perubahan..Bentuk perusahaan yang semula perusahaan umum kereta api(PERUMKA).Melalui PP no.19 tahun 1998,tanggal 3 Pebruari 1998,berubah status menjadi Perusahaan Perseroan ,PT.Kereta Api (PERSERO).

Setelah itu,melalui keputusan Presiden NO.38 tahun 1999 pengecualian perusahaan perseroan tertentu yang dapat dikecualikan dari pengalihan tugas dan kewenangan mentri Keuangan selaku pemegang saham atau rapat umum pemegang saham (RUPS) kepada Mentri Negara Pendaya gunaan Badan Usaha Milik Negara (PBUMN)ditetapkan bahwa PT.Kereta Api tetap dibawah naungan Departemen Perhubungan Kepres yang ditandatangani Presiden Baharudin Yusup

Habibie,tersebut ditandatangani pada tanggal 17 Mei 1999 menandai nilai diberlakukannya,status PT, namun demikian secara efektif baru diberlakukan pada tanggal 1 Juni 1999 melalui upacara,sebagai tanda persmian pengalihan status dari PERUMKA menjadi PT.Kereta Api (Persero) di kantor pusat Bandung.

3.1.2 Visi dan Misi

Visi adalah suatu obsesi yang melampau realitas yang ada sekarang, sesuatu yang akan kita ciptakan sebelumnya belum pernah ada atau suatu keadaan yang akan kita wujudkan di masa yang akan datang.

Guna mencapai visi yang kan diwujudkan tersebut maka dibutuhkan misi karena misi adalah suatu acara bagaimana perusahaan mencapai visi atau tujuan suatu organisasi atau perusahaan dan misi tersebut dapat diwujudkan dengan strategi, cara-cara dan pedoman berfikir sebagai langkah untuk menuju kondisi di masa depan.

Berdasarkan pemahaman tentang visi dan misi maka dapat dirumuskan bahwa visi dan misi PT. Kereta Api Indonesia (Persero) sebagai berikut:

a. Visi

Yaitu dengan mewujudkan kereta api sebagai pilihan utama jasa tansportasi yang mengutamakan keselamatan, pelayanan dan pengembangan Sumber Daya Manusia (SDM) yang professional.

b. Misi

Untuk lebih memperjelas arah misi PT. Kereta Api Indonesia (Persero), maka dijabarkan sebagai berikut:

1. Menjaga kualitas produk.
2. Untuk lebih meningkatkan pelayanan dan keselamatan.
3. Pelayanan jasa transportasi yang semakin efisiensi.

3. Tugas Pokok, Kedudukan dan Fungsi PT. Kereta Api Indonesia (Persero) Bandung.

3.1.2.1 Tugas Pokok PT. Kereta Api (Persero)

PT. Kereta Api (Persero) mempunyai tugas pokok menyelenggarakan Pengusahaan pelayanan jasa angkutan kereta api dalam rangka memperlancar atas perpindahan orang dan atau barang secara misal untuk menunjang pembangunan nasional.

3.1.2.2 Kedudukan PT. Kereta Api (Persero)

Kedudukan PT. Kereta Api adalah sebagai berikut :

1. PT. Kereta Api Indonesi (Persero) adalah BUMN dalam Lingkungan Departemen Perhubungan, dipimpin oleh suatu Direksi dan bertanggung jawab kepada Menteri Perhubungan.
2. Menteri Perhubungan melakukan pembnaan terhadap PT. Kereta Api Indonesia (Persero)dibantu oleh Direktur Jenderal Perhubangan Darat sesuai dengan kewenangan yang diberikan oleh Menteri.
3. Dewan pengawas pada PT. Kereta Api Indonesi (Persero) bertanggung jawab kepada Menteri melakukan pengawasan terhadap pengelolaan

PT. Kereta Api Indonesia (Persero) temasuk pelaksanaan rencana kerja dan anggaran PT. Kereta Api Indonesia (Persero)Bandung.

3.1.2.3 Fungsi PT. Kereta Api (persero)

PT. Kereta Api Indonesia (Persero) mempunyai fungsi sebagai berikut, yaitu:

1. Penyediaan, pengoprasian, pendayagunaan, pemeliharaan, perbaikan, pengendalian, dan pengembangan sarana angkutan atas jalan rel maupun sarana angkutan bukan diatas rel, yang merupakan kelanjutan angkutan kereta api.
2. Penyediaan, pemeliharaan, perbaikan, dan pendayagunaan prasarana jalan rel, jembatan, terowongan, instalansi sinyal, dan telekomunikasi, instalansi listrik umum dan listrik atas, seta penyediaan dan pendayagunaan bangunan stasiun, bangunan dipo, serta bangunan gedung lainnya yang diperuntukkan menunjang kereta api.
3. Pengoperasian, pendayagunaan, pengendalian, penguasaan stasiun, stasiun untuk pembarangkatan, penurunan dan atau angkutan penumpang maupun barang, serta jasa angkutan lain yang terkait.
4. Pengelolaan, pendayagunaan keuangan perusahaan.
5. Pengelolaan, pendayagunaan dan pengembangan SDM.
6. Pengawasan Internal.

3.1.3 Struktur Organisasi

Dalam suatu organisasi diperlukan suatu struktur organisasi yang mengatur tugas, wewenang dan tanggung jawab dari setiap bagian secara tegas dari struktur organisasi tersebut sehingga dapat dilihat hubungan antara bagian-bagian yang ada dalam perusahaan yang bersangkutan.

Struktur organisasi yang diterapkan PT. Kereta Api (Persero) adalah bentuk garis, struktur organisasi tingkat pusat berdasarkan PP No.19/1998 dan akte Notaris No.02/1999.

Berikut ini merupakan bagian yang terdapat dalam struktur organisasi kantor pusat, yaitu:

1. Direksi, dipimpin oleh seorang direktur utama.
2. Direktorat, terdiri dari :
    a. Direktorat Keuangan
    b. Direktorat Teknik
    c. Direktorat Operasi dan Pemasaran
    d. Direktorat Personalia dan Umum
    e. Direktorat Pengembangan Usaha
3. Unit yang bertanggung jawab langsung pada direksi, yaitu :
    a. Pusat Perencanaan dan Pengembangan

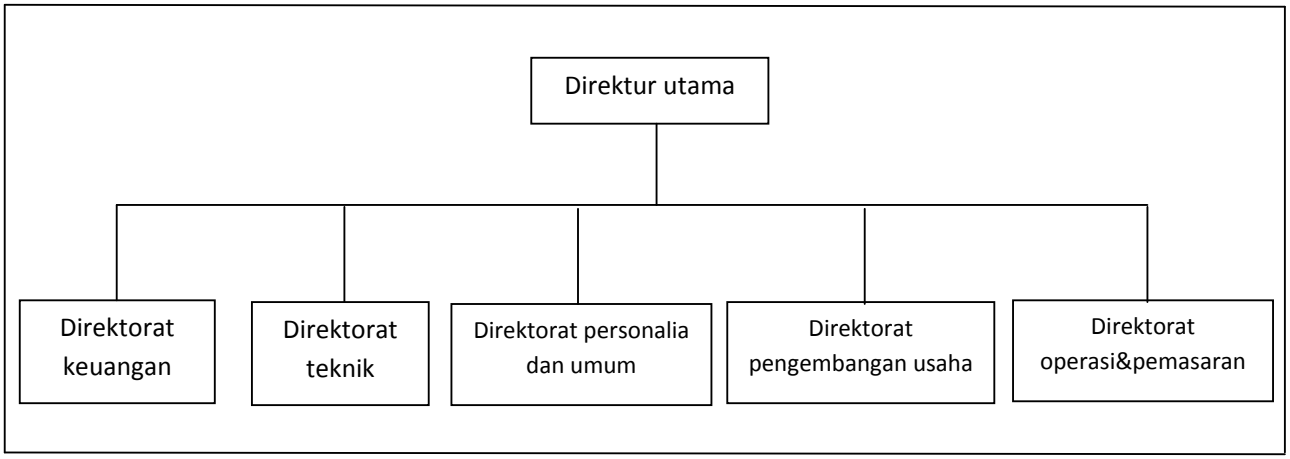
b. Pusat Pendidikan dan Pelatihan

c. Satuan Pengawasan Intern (SPI)

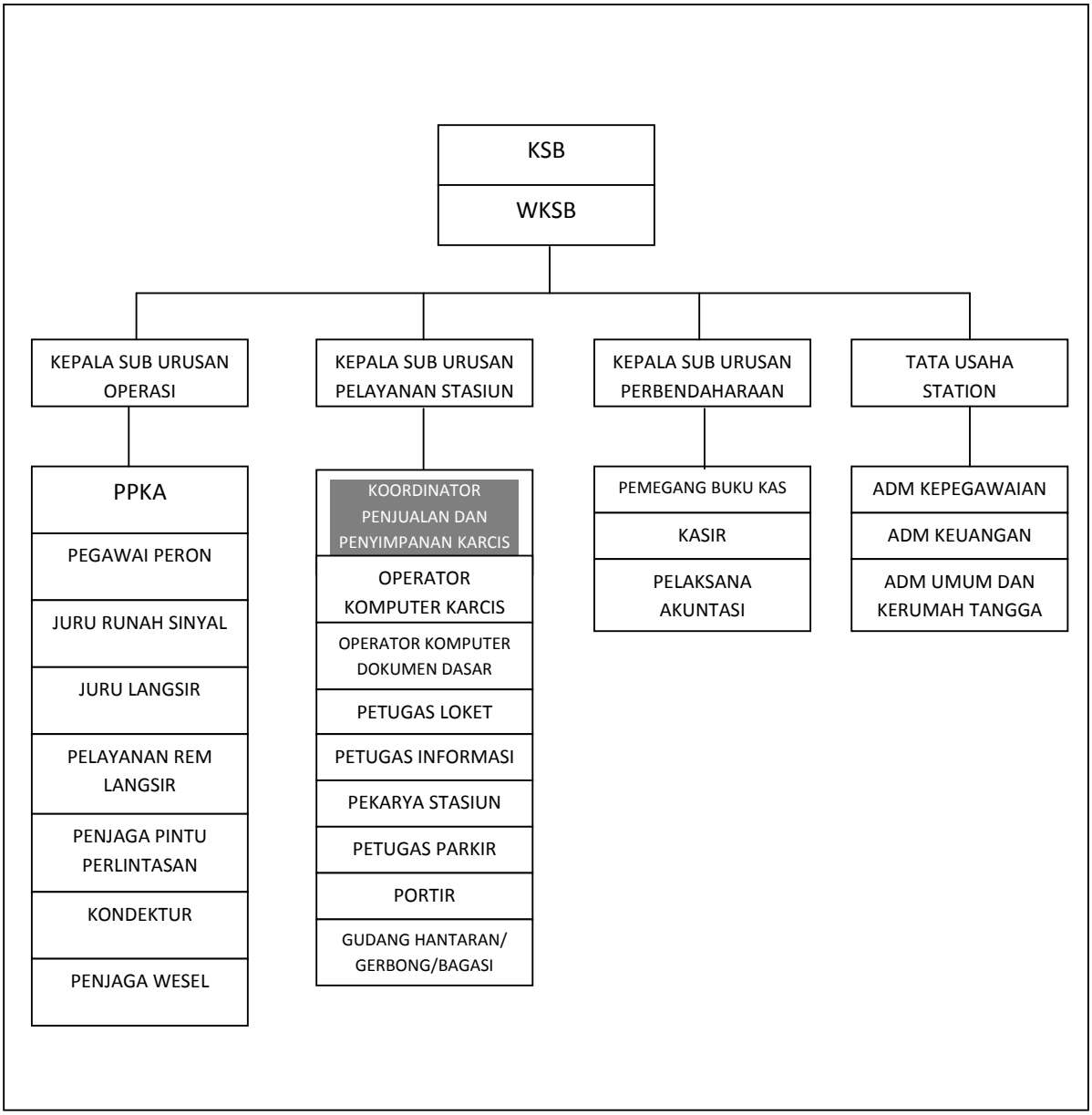
d. Pusat Logistik

Sedangkan bagian yang berhubungan dengan masalah penggajian dan pengupahan dalam hal ini adalah Direktorat Keuangan. Direktorat Keuangan dipimpin oleh seorang Direktur Keuangan yang bertanggung jawab kepada Direktur Utama.

Direktorat Keuangan mempunyai tugas melaksanakan sebagian tugas Direktur Utama dalam merumuskan kebijaksanaan serta mengelola keuangan perusahaan yang meliputi penyusunan Rencana Kerja Anggaran Perusahaan (RKAP) Tahunan, penyediaan dana baik untuk modal kerja dan investasi, perhitungan PSO, IMO, dan TAC, pelaksanaan administrasi keuangan/perbendaharaan, pelaksanaan akuntansi manajemen, akuntansi keuangan, serta membantu pengurusan barang inventaris kekayaan milik Negara.



Gambar 3.1 Struktur Organisasi Kantor Pusat



Gambar 3.2 Struktur Organisasi Stasiun Kereta Api

3.1.4 Job Description

Dalam suatu perusahaan tentunya tidak terlepas dari struktur organisasi yang berfungsi untuk mempermudah dalam pengaturan kerja sebuah perusahaan.

Dalam struktur organisasi yang ada pada Pt. Kereta Api (Persero) Daop 2 Bandung, yaitu ;

1. Kepala Stasiun Besar, mempunyai tugas mengkoordinasikan serta mengendalikan pelaksanaan angkutan penumpang dan barang serta pengamanan kegiatan angkutan kereta api di stasiun.

2. Wakil Kepala Stasiun Besar, mempunyai tugas mewakili atau membantu kepala stasiun besar dalam melaksanakan tugas mengkoordinasikan serta mengendalikan pelaksanaan angkutan penumpang dan barang serta pengamanan kegiatan angkutan kereta api di stasiun, bertanggung jawab kepada kepala stasiun.

3. Kepala Sub Urusan Operasi, mempunyai tugas melaksanakan tugas pengkoordinasian, pengawasan serta pengendalian pelaksanaan angkutan penumpang dan barang dengan menggunakan kereta api di wilayah operasinya dalam melaksanakan tugasnya bertanggung jawab kepada Kepala Stasiun Besan.

   Dalam melaksanakan tugasnya dibantu oleh petugas operasional sebagai berikut :

   a. Pemimpin Perjalanan Kereta Api, mempunyai tugas pengaturan penyiapan pemberangkatan perjalanan kereta api berdasarkan peraturan (*reglement* yang terkait) yang berlaku serta dijamin

keamanannya, dalam wilayah operasinya agar berjalan sesuai dengan jadwal.

b. Pengawas Peron, mempunyai tugas melaksanakan pekerjaan yang bersangkutan dengan urusan perjalanan kereta pi dan urusan langsir di stasiun.

c. Petugas Materiil, mempunyai tugas meninventarisir adanya kereta dan gerbong serta mengatur pemakaiannya serta koordinasi dengan pengawas urusan kereta dan gerbong.

d. Juru Runah Sinyal, mempunyai tugas mengopereasikan sinyal masuk dan keluar, dalam hal melayani kereta api.

e. Juru Langsir , mempunyai tugas menyusun rangkaian kereta api yang akan berangkat atau memisah-misahkan rangkaian kereta api yang datang, memindahkan kereta, gerbong, dan lain-lain jenis materiil dengan memperhatikan kesempurnaan saluran rem antara lokomotif dengan kereta atau gerbong.

f. Pelayan Rangkaian Langsir, mempunyai tugas membantu pelaksanaan tugas juru langsir.

g. Penjaga Pintu Perlintasan, mempunyai tugas pengamanan perjalanan kereta api, membuka dan menutup pintu perlintasan kereta api dalam hal pengamanan perjalanan kereta api.

h. Kondektur, mempunyai tugas penguasaan sepenuhnya atas kereta api di luar lingkungan stasiun, pemberian petunjuk pelaksanaan tugas awak kereta api dan mengatur keselamatan penumpang dan barang selama dalam perjalanan.

i. Penjaga Wesel, mempunyai tugas menjaga wesel untuk tetap berfungsi.

4. Kepala Sub Urusan Pelayanan, mempunyai tugas untuk memberikan palayanan yang sebaik-baiknya kepada para konsumen ataupun calon konsumen baik sebelum naik kereta api maupun sesudah turun dari kereta api dan bertanggung jawab kepada Kepala Stasiun Besar.

   Dalam melaksanakan tugasnya dibantu oleh petugas pelayanan stasiun sebagai berikut :

   a. Koordinator Penjualan Karcis, mempunyai tugas untuk mengkoordinasikan petugas penjualan karcis serta bertanggung jawab baik fisik karcis maupun keuangannya yang disetorkan kepada bendahara stasiun.

   b. Petugas Loket, selain melayani penjualan dan pemesanan tiket juga mempunyai wewenang untuk menginput data penumpang ke komputer. Bila ada pembatalan dari penumpang maka petugas berwenang untuk menghapus data sesuai dengan tiket yang bersangkutan.

c. Operator Komputer Dokumen Dasar, bagian ini bertugas membuat rekap hasil penjualan dan jumlah penumpang yang berangkat serta pembatalan pemesanan tiket.

d. Petugas Informasi, yang bertugas untuk memberikan informasi baik kepada penumpang yang akan berangkat maupun penumpang yang tiba stasiun.

e. Pekarya Stasiun, bertugas untuk melaksanakan kegiatan kebersihan dan keindahan di stasiun.

f. Portir, bertugas melaksanakan pengawasan dan pengamanan dan ketertiban pintu masuk ke stasiun.

g. Pengelola Barang Hantaran atau Gudang, mempunyai tugas menerima dan mengirim barang dinas atapun barang hantaran.

h. Mandor, yang bertugas untuk mengawasi kegiatan pelaksanaan kegiatan kebersihan dan keindahan di stasiun.

5. Kepala Sub Urusan Perbendaharaan Stasiun, mempunyai tugas penguasaan semua keuangan stasiun, akuntansi pendapatan, biaya dan pelaporan mengenai pendapatan stasiun dengan alat komputer dan bertanggung jawab kepada Kepala Stasiun Besan.

   Dalam melaksanakan tugasnya dibantu oleh petugas sebagai berikut :

a. Pemegang Buku Kas, mempunyai tugas mencatat pengeluaran dan pemasukan uang di lingkungan stasiun yang bersangkutan.

b. Kasir, mempunyai tugas menerima dan mengeluarkan uang secara Intern dan Ekstern di lingkungan stasiun terkait.

c. Pelaksana Akuntansi Perbendaharaan, mempunyai tugas melaksanakan kegiatan akuntansi pembendahraan dan berkoordinaasi dengan urusan akuntansi daerah operasi.

6. Kepala Sub Bagian Tata Usaha, mempunyai tugas melaksanakan kegiatan tata usaha stasiun mencatat surat-surat yang masuk dan keluar, administrasi kepegawaian dan umum.

   Dalam melaksanakan tugasnya Kepala Sub Urusan Tata Usaha Stasiun dibantu oleh :

   a. Pelaksana Administrasi Kepegawaian, yang mempunyai tugas untuk melaksanakan kegiataan yang berhubungan dengan masalah kepegawaian stasiun.

   b. Pelaksana Administrasi Keuangan, bertugas untuk melaksanakan kegiatan yang berkaitan dengan hak-hak keuangan bagi pegwai di stasiun.

   c. Pelaksana Administrasi Umum dan Kerumah tanggaan, bertugas untuk melaksanakan kegiatan yang berkaitan dengan kerumah tanggaan stasiun.

3.2 Metode Pengumpulan data

Metode pengumpulan data yang digunakan dalam sistem informasi yang akan dibangun yaitu berupa pengumpulan data yang dibutuhkan untuk dalam sistem informasi yang akan dibuat baik sumber data primer juga sekunder dari Pt. Kereta Api (Persero).

3.2.1 Sumber Data Primer

Sumber data primer di dapat adalah dengan cara sebagai berikut :

1. *Interview,* hal ini dilakukan oleh penulis dalam rangka mencari keterangan-keterangan yang diperlukan. Dimana penulis melakukan interview ini dengan pihak yang ikut terlibat diantaranya dengan petugas loket dan koordinator penjualan dan penyimpanan karcis yang bertempat di bagian penjualan dan penyimpanan karcis yang dimana nanti informasinya akan digunakan sebagai bahan acuan membangun sistem informasi.

2. Studi pengamatan (*Observasi*), yaitu kegiatan memperoleh data dengan mengamati langsung objek yang diteliti oleh penulis. Pengamatan ini dilakukan penulis dengan melihat aktifitas transaksi pembelian dan pemesanan tiket pada bagian loket penjualan dan reservasi tiket.

3. Kuesioner, yaitu dengan pengumpulan data dangan memberikan kuesioner secara langsung dengan mencari data-data berdasarkan skala

prioritas yang berhubungan dengan masalah yang dibahas terutama dalam perancangan pembangunan sistem informasi.

3.2.2 Sumber Data Sekunder

Sumber data sekunder disini berupa bukti pembayaran yaitu tiket, formulir pemesanan dan brosur jadwal keberangkatan kereta pada DAOP 2 Bandung.

3.3 Metode Pendekatan atau Pengembangan Sistem

Dalam penelitian ini penulis melakukan metode pendekatan terhadap objek penelitian dan mencari solusi untuk penyelesaian masalah dari sistem yang berjalan dengan metode-metode yang penulis ketahui dari sumber bacaan yang dibaca dan pernah dipelajari.

3.3.1 Metode Pendekatan Sistem

Metode pendekatan sistem yang penulis gunakan adalah metode pendekatan analisis dan pemograman notasi yang berorientasi objek.

"*Object Management Group, Inc. (OMG)* adalah sebuah grup internasional yang dibentuk pada tahun 1989, didukung lebih dari 800 angggota, terdiri dari perusahaan sistem informasi, *software developer* dan para *user* sistem komputer. OMG mempromosikan teori dan praktek-praktek *object oriented technology* dalam rekayasa *software*. Organisasi ini salah satunya bertugas

membuat spesifikasi “manajemen objek” untuk menetapkan kerangka bersama dalam rekayasa *software*. Spesifikasi tersebut dibuat dengan tujuan utama untuk mendapat *reusability*, *portabillity*, dan *interoperabillity software* yang berdasarkan *object oriented* (OO), dalam lingkungan yang heterogen, dapat dioperasikan dalam semua *platform hardware* dan sistem operasi.

Sasaran OMG adalah membantu perkembangan *object oriented technology* dan mengalahkan dengan mendirikan *Object Manajement Architecture (OMA).* OMA menentukan infastruktur konseptual yang didasarkan pada seluruh spesifikasi yang dikeluarkan OMG.

OMG kemudian mengeluarkan UML, dimana dengan adanya UML ini diharapkan dapat mengurangi kekacauan dalam bahasa pemodelan yang selama ini terjadi dalam lingkungan industri. UML diharapkan juga dapat menjawab masalah penotasian dan mekanisme tukar menukar model yang terjadi selama ini.

OMG telah menetapkan semantik (makna istilah) semua notasi UML dalam model struktur dan model *behavioral.* Model *structural* menekankan struktur objek dalam sebuah sistem, menyangkut kelas–kelas, *interface* dan hubungan antar komponen. Sedangkan model *behavioral* menekankan prilaku objek dalam sebuah sistem, termasuk metode, interaksi, kolaborasi, dan *state history*.” [http://www.ilmukomputer.com]

Dengan dibuatnya berbagai jenis diagram pada UML maka :

1. Setiap sistem yang kompleks selalu paling baik jika di dekati melalui himpunan berbagai sudut pandang yang kecil yang satu sama lain hampir saling bebas (*independent*). Sudut pandang tunggal senantiasa tidak mencukupi untuk melihat sistem yang besar dan kompleks.
2. Diagram yang berbeda-beda tersebut dapat menyatakan tingkatan yang berbeda-beda dalam proses dalam proses rekayasa.
3. Diagram-diagram tersebut dibuat agar model yang dibuat semakin mendekati realitas.

3.3.2 Metode Pengembangan Sistem

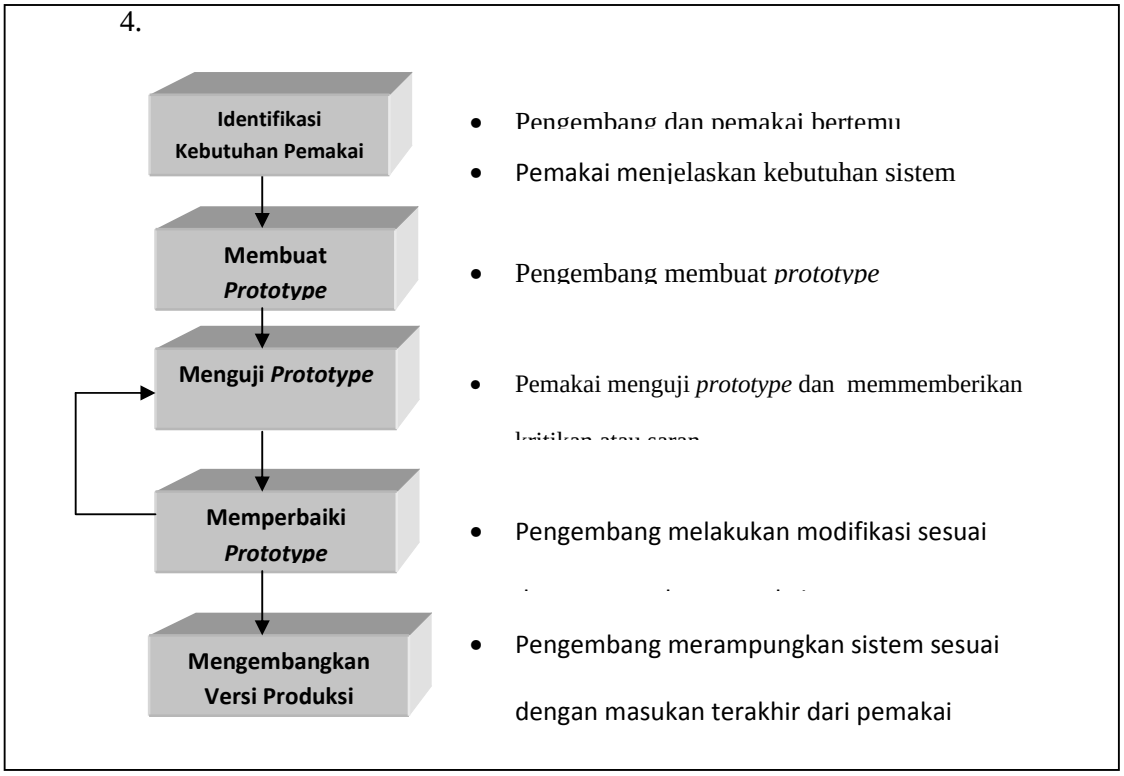
Paradigma perancangan sistem yang digunakan dalam perancangan perangkat lunak untuk tugas akhir ini adalah Metode *Prototype*. Metode *prototype* merupakan suatu metode yang dalam pengembangan sistemnya menggunakan pendekatan untuk membuat suatu program dengan cepat serta bertahap, sehingga kapanpun dapat segera dievaluasi oleh pemakai (*user*).

Adapun alasan penggunaan metode pengembangan sistem dengan *prototype*, yaitu :

1. Memudahkan dalam perancangan sistem, karena mendapatkan kebutuhan dan aturan yang jelas sesuai apa yang diinginkan oleh pemakai (*user*).
2. Perancangan sistem bisa dilakukan secara cepat dan memungkinkan untuk merubah kembali perangkat lunak agar sesuai dengan kebutuhan pemakai

(*user*). Apabila perancangan sistem dan perangkat lunak *prototype* sesuai dengan yang diinginkan oleh pemakai (*user*) maka *prototype* dihilangkan lalu dibuatkan perangkat lunak yang sebenarnya.

3. Perubahan akan ketidaksesuaian dan presentasi dalam model *prototype* ini dapat dilakukan berkali-kali sampai dicapai kesepakatan bentuk sistem informasi yang dibutuhkan. Karena metode *prototype* dirancang agar dapat menerima perubahan-perubahan dalam rangka penyempurnaan, sehingga pada akhirnya dapat menghasilkan suatu sistem informasi yang dapat diterima oleh pemakai (*user*) dan setiap perubahan yang ada, dianggap sebagian proses pengembangan sistem itu sendiri.
4.



Gambar 3.3 Mekanisme Pengembangan Model Sistem *Prototype.*

(Sumber : Abdul Kadir 2003)

Berikut adalah langkah-langkah dalam merancang sebuah sistem yang menggunakan mekanisme pengembangan sistem *prototype* yang sesuai dengan gambar 3.3 diatas, diantaranya sebagai berikut:

1. Tahap pertama mengidentifikasi kebutuhan pemakai (*user*), sehingga perancangan sistem yang akan dibangun sesuai dengan harapan pemakai (*user*). Untuk dapat mengidentifikasi kebutuhan pemakai (*user*), diperlukan analisis sistem dengan cara melakukan pengumpulan data dari penjelasan pemakai (*user*) melalui wawancara (*interview*), observasi lapangan/objek (*field research*), dan studi literatur dengan perbandingan beberapa hasil dokumentasi terhadap kebutuhan yang diinginkan pemakai (*user*), baik model tampilan (*interface*), teknik *procedural* maupun dalam aplikasi teknologi yang akan digunakan.
2. Tahap kedua yaitu membuat *prototype*, penulis sekaligus pengembang sistem akan membuat *prototype* sistem berdasarkan data kebutuhan (*requirement*) untuk memperlihatkan kepada pemakai (*user*) model dari sistem yang akan dirancang.
3. Tahap ketiga yaitu pengujian terhadap *prototype*, pada tahap ini akan dilakukan penguji cobaan sistem yang telah dirancang untuk memastikan bahwa sistem yang berupa *prototype* tersebut dapat digunakan dengan baik dan benar sesuai kebutuhan pemakai (*user*). Setelah diuji langsung oleh pemakai (*user*), maka pemakai (*user*) dapat langsung memberikan kritik dan saran untuk perbaikan sistem selanjutnya.

4. Tahap keempat yaitu memperbaiki *prototype*, erdasarkan hasil kritik dan saran dari pemakai (*user*), maka telah dapat ditentukan apakah *prototype* sistem tersebut dapat diterima oleh pemakai (*user*)? atau harus dilakukan perbaikan atau bahkan dirubah seluruhnya dan membuat dari awal lagi, sehingga pengembangan sistem berjalan dan setelah perbaikan sistem *prototype* tersebut itu selesai, maka langakah selanjutnya yaitu kembali melakukan pengujian *prototype* kembali hingga system sesuai dengan kebutuhan dari pemakai (*user*).
5. Tahap kelima yaitu tahap terakhir dalam mengembangkan versi produksi, pada tahapan ini seluruh hasil masukan kebutuhan pengembangan dari pemakai (*user*) setelah pengujian terakhir dirampungkan dan dideskripsikan bagaimana cara penggunaan sistem yang dirancang tersebut kepada pemakai (*user*), lalu disetujui apakah sistem tersebut telah memenuhi kebutuhan pemakai (*user*) atau tidak.

4.3.3 Alat Bantu Analisis dan Perancangan

Pembangunan sistem informasi dengan berorientasi objek merupakan suatu perancangan yang berbeda dengan yang berorientasi data. Namun secara konteks perancangan ini digunakan untuk membangun sistem informasi sesuai kebutuhan dari pemakai (*user*) sistem informasi.

Adapun cakupan dari UML itu sendiri, yaitu :

1. Pertama, UML menggabungkan konsep Booch, OMT, dan OOSE, sehingga UML merupakan suatu bahasa pemodelan tunggal yang umum dan digunakan secara luas oleh para *user* ketiga model tersebut dan bahkan para *user* metode lainnya.
2. Kedua, UML menekankan pada apa yang dapat dikerjakan dengan metode–metode tersebut.
3. Ketiga, UML berfokus pada suatu bahasa pemodelan standar, bukan pada proses standar. Mekipun UML harus diaplikasikan dalam konteks sebuah proses, dari pengalaman bahwa organisasi dan masalah yang berbeda juga.

Sedangkan UML tidak mencakup hal-hal sebagai berikut:

1. Bahasa pemograman

   UML adalah bahasa pemodelan visual, bukan dimaksudkan untuk menjadi suatu bahasa pemograman visual, tetapi UML memberikan arah untuk bergerak kearah kode.

2. *Tool* (*software* aplikasi)

   Membuat standar sebuah bahasa diperlukan oleh *tool–tool* dan proses. UML mendefinisikan semantikdan notasi, bukan sebuah *tool*. Contoh *tool* yang menggunakan UML sebagai bahasanya adalah *Rational Rose* dan *Enterprise Architect.*

3. Proses rekayasa

UML digunakan sebagai bahasa dalam proyek dengan proses yang berbeda–beda. UML bebas dari proses dan mendefinisikan sebuah proses standar bukan tujuan UML atau RFP dari OMG.

UML sendiri telah menyediakan alat-alat bantu berupa diagram visual yang menggambarkan berbagai aspek yang ada di dalam sistem. Adapun alat-alat bantu diagram yang disediakan di dalam UML diantaranya, yaitu :

3.3.3.1 *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 duplikasi fungsionalitas 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.

3.3.3.2 *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, dimana 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. Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih.

Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan

aktivitas. 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 vertical. *Activity diagram* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

3.3.3.3 *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. 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.

3.3.3.4 *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.

3.3.3.5 *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.

Class memiliki area pokok yaitu nama (dan stereotype), atribut, dan metoda. Atribut dan metoda dapat memiliki salah satu sifat berikut :

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

Class dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class.* Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time*. Sesuai dengan perkembangan *class* model, *class* dapat dikelompokkan menjadi *package*. Kita juga dapat membuat diagram yang terdiri atas *package*.

Hubungan Antar *Class* adalah sebagai berikut :

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.

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.

3.3.3.5.1 *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.

3.3.3.5.2 *Depeloyment* 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.

3.3.3.5.3 *Software* Pemodelan UML

Menurut A. Suhendar, S. Si. dan Hariman Gunandi, S. Si., MT dalam bukunya yang berjudul ”*Visual Modeling Menggunakan UML dan Rational Rose*”

mengatakan, *Rational rose* adalah *software* yang memiliki perangkat-perangkat pemodelan secara visual untuk membangun suatu solusi dalam rekayasa *software* dan pemodelan bisnis. Selain itu *Rational Rose* memiliki kemampuan membuat solusi *Client/Server*, yang kemudian dapat diterapkan dan didistribusikan dalam lingkungan perusahaan atau instansi.

Rational Rose dikeluarkan oleh perusahaan *software* bernama *Rational Software*, perusahaan yang mencetuskan ide bagi perusahaan-perusahaan yang memakai standar UML sebagai bahasa pemodelan di perusahaannya.

3.4 Faktor Pengujian Perangkat Lunak/*Software*

Definisi awal dari pengujian secara umum adalah proses untuk mengecek apakah suatu perangkat lunak yang dihasilkan sudah dapat dijalankan sesuai standar tertentu. Dan standar yang dijadikan acuan dapat berupa standar dari instansi ertentu ataupun disesuaikan dengan keperluan costumer dan pemakai (*user*).

Pengujian perangkat lunak/*software* adalah elemen kritis dari jaminan kualitas dan representasi dari kajian pokok spesifikasi, desain dan pengkodean dari perangkat lunak itu sendiri.

Adapun definisi pengujian menurut *Institute of Electrical and Electornics Engineering/American National Standards Institute* (IEEE/ANSI), yaitu:

1. *The process of operating a system or component under specified condition, observing or recording the result, and making an evaluation of some aspect of system/component*. [IEEE/ANSI, 1990 std 610.12-1990]
2. *The process of analyzing software item to detect the difference existing and required condition (that is bugs) and to evaluate the feature of the software items.* [IEEE/ANSI, 1983 std 829-1983]

Menurut [Cit08] terdapat 2 hal utama yang dilakukan dalam pengujian, yaitu:

1. Verifikasi, adalah suatu proses mengevaluasi suatu sistem atau komponen untuk menentukan apakah suatu produk yang diselesaikan setelah fase pengembangan memenuhi kondisi seperti yang telah ditetapkan pada awal pengembangan perangkat lunak.
2. Validasi, adalah suatu proses mengevaluasi suatu sistem atau komponen pada akhir atau selama masa pengembangan untuk menentukan apakah produk yang dihasilkan telah memenuhi kebutuhan-kebutuhan tertentu yang diminta.

Metode pengujian perangkat lunak/*software* yang penulis gunakan adalah sebagai berikut :

3.4.1 *White Box Testing*

Menurut [Cit08], pengujian *White Box* (*Glass Box*) adalah pengujian yang didasarkan pada pengecekan terhadap detil perancangan, menggunakan struktur

control dari desain program secara procedural untuk membagi pengujian ke dalam beberapa kasus pengujian.

Penggunaan metode pengujian *White Box* dilakukan untuk :

1. Memberikan jaminan bahwa semua jalur independen suatu modul digunakan minimal satu kali.
2. Menggunakan semua keputusan logis untuk semua kondisi *true* dan *false.*
3. Mengeksekusi semua perulangan pada batasan nilai dan operasional pada setiap kondisi.
4. Menggunakan struktur data internal untuk menjamin validitas jalur keputusan.

3.4.2 *Black Box Testing*

Menurut [Cit08], pengujian *black box* adalah pengujian aspek fundamental sistem tanpa memperhatikan struktur logika internal perangkat lunak. Metode ini digunakan untuk mengetahui apakah perangkat lunak berfungsi dengan benar. Pengujian *black box* merupakan metode perancangan data uji yang didasarkan pada spesifikasi perangkat lunak. Data uji dibangkitkan, dieksekusi pada perangkat lunak dan kemudian keluaran dari perangkat lunak dicek apakah telah sesuai dengan yang diharapkan. Langkah-langkah dalam pengujian *Black Box* adalah :

1. *Graf Based Testing*

   Langkah pertama pada pengujian *black box* adalah memahami objek yang terdapat dalam model perangkat lunak dan menentukan hubungan yang

dimiliki antara objek-objek tersebut. Pengujian berbasiskan model graf dilakukan terhadap perilaku sistem. Pengujian ini dimulai dari mendefinisikan semua simpul dan bobot simpul, dimana objek dan atribut diidentifikasikan, serta memberikan indikasi titik mulai dan berhenti.

2. *Equivalence Partitioning* (Partisi Ekuivalensi)

   Partisi ekuivalensi adalah metode yang membagi domain input dari suatu program ke dalam kelas data, menentukan kasus pengujian dengan mengungkapkan kelas-kelas kesalahan, sehingga akan mengurangi jumlah keseluruhan kasus pengujian.

3. *Boundary Value Analysis* (Analisis Nilai Batas)

   Analisis nilai batas adalah teknik desain proses yang melengkapi partisi ekuivalensi, dengan berfokus pada domain output.

4. *Comparison Testing*

   Pengujian perbandingan adalah metode pembangkitan data uji yang dilakukan pada perangkat lunak yang dibuat redundan. Perangkat lunak yang redundan mempunyai dua tim pengembang yang masing-masing mengembangkan perangkat lunak sendiri-sendiri untuk spesifikasi yang sama.

3.4.3 Faktor Uji

Beberapa Faktor yang diujikan dalam pengujian perangkat lunak/*Software* diantaranya sebagai berikut :

1. *Reliability*

Requirement	:	Menentukan toleransi
Desain	:	Desain kontrol dan integritas data
Program	:	Implementasi kontrol integritas data
Pengujian	:	Pengujian regresi, manual dan fungsional
Instalasi	:	Verifikasi ketepatan dan kelengkapan instalasi
Perawatan	:	*Update* ketepatan kebutuhan

2. *Authorization*

Requirement	:	Identifikasi aturan otorisasi
Desain	:	Desain aturan otorisasi
Program	:	Implementasi atuuran otorisasi
Pengujian	:	Pengujian kesesuaian
Instalasi	:	Mencegah perubahan data selama instalasi
Perawatan	:	Menjaga aturan otorisasi

3. *File Integrity*

Requirement	:	Identifikasi kebutuhan integritas file
Desain	:	Desain kontrol dan integritas file
Program	:	Implementasi kontrol integritas file
Pengujian	:	Pengujian fungsional
Instalasi	:	Verifikasi integritas dari produksi file
Perawatan	:	Menjaga integritas file

4. *Audit Trail*

Requirement	:	Identifikasi kebutuhan rekontruksi
Desain	:	Desain *audit trail*
Program	:	Implementasi *audit trail*
Pengujian	:	Pengujian fungsional
Instalasi	:	Menyimpan *audit trail* instalasi
Perawatan	:	*Update audit trail*

5. *Continuity of Processing*

Requirement	:	Identifikasi akibat dari kegagalan
Desain	:	Desain *contingency plan*

Program	:	Menyusun *contingency plan* dan prosedurenya
Pengujian	:	Pengujian pemulihan
Instalasi	:	Memastikan integritas dari pengujian sebelumnya
Perawatan	:	*Update contingency plan*

6. *Service Level*

Requirement	:	Identifikasi tingkat layanan yang diinginkan
Desain	:	Desain metode untuk mencapai tingkat layanan
Program	:	Desain sistem untuk mencapai tingkat layanan
Pengujian	:	Pengujian beban lebih
Instalasi	:	Implementasi rencana pecegahan kegagalan instalasi
Perawatan	:	Menjaga tingkat layanan

7. *Access Control*

Requirement	:	Identifikasi hak akses
Desain	:	Desain prosedur akses

Program : Implementasi prosedur keamanan

Pengujian : Pengujian kesesuaian

Instalasi : Kontrol akses selama instalasi

Perawatan : Menjaga keamanan

8. *Methodology*

Requirement : Penyesuaian kebutuhan dengan metodelogi

Desain : Penyesuaian desain dengan metodelogi

Program : Penyesuaian program dengan metodelogi

Pengujian : Penyesuaian pengujian dengan metodelogi

Instalasi : Penyesuaian integrasi dengan metodelogi

Perawatan : Penyesuaian perawatan dengan metodelogi

9. *Correctness*

Requirement : Identifikasi spesifikasi fungsional

Desain : Penyesuaian desain dangan *requirement*

Program : Penyesuaian program dangan desain

Pengujian : Pengujian fungsional

Instalasi : Ketepatan peempatan program dan data pada

produksi

Perawatan : *Update* kebutuhan

10. *Ease of Use*

Requirement : Identifikasi spesifikasi kegunaan

Desain : Desain penggunaan fasilitas

Program : Penyesuaian program dangan desain

Pengujian : Pengujian dukungan panduan

Instalasi : Penyebaran kegunaan instruksi

Perawatan : Menjaga kemudahan penggunaan

11. *Maintainable*

Requirement : Identifikasi spesifikasi perawatan

Desain : Desain dapat dirawat

Program : Program dapat dirawat

Pengujian : Inspeksi

Instalasi : Kelengkapan dokumentasi

Perawatan : Menjaga keremawatan

12. *Portable*

Requirement	:	Identifikasi kebutuhan portabilitas
Desain	:	Desain portabilitas
Program	:	Penyesuaian program dangan desain
Pengujian	:	*Disaster testing*
Instalasi	:	Kelengkapan dokumentasi
Perawatan	:	Menjaga portabilitas

13. *Coupling*

Requirement	:	Identifikasi antarmuka sistem
Desain	:	Kelengkapan desain antarmuka
Program	:	Penyesuaian program dangan desain
Pengujian	:	Pengujian fungsional dan regresi
Instalasi	:	Koordinasi antarmuka
Perawatan	:	Memastikan antarmuka yang benar

14. *Performance*

Requirement	:	Identifikasi kriteria peforma

Desain	:	Kriteria pencapaian desain
Program	:	Kriteria pencapaian program
Pengujian	:	Pengujian kesesuaian
Instalasi	:	Mengawasi perfoma instalasi
Perawatan	:	Menjaga tingkat performa

15. *Ease of Operation*

Requirement	:	Identifikasi kebutuhan operasional
Desain	:	Mengkomunikasikan kebutuhan pada operasi
Program	:	Mengembangkan prosedur operasi
Pengujian	:	Pengujian operasi
Instalasi	:	Implementasi prosedur operasi
Perawatan	:	*Update* prosedur operasi