BAB II
LANDASAN TEORI

2.1 Tinjauan Umum Perusahaan

Gambar 2.1 Perusahaan PDAM Tirta Indragiri

Awalnya prasarana air bersih di Kabupaten Indragiri Hilir dibangun pada tahun 1980 dengan paket BNA kapasitas 20 l/dt yang terletak di desa Pulau Palas kurang lebih 13 Km dari kota Tembilahan. Dengan sistem paket Pengolahan Lengkap, sungai Indragiri yang melintasi desa Pulau Palas dijadikan sebagai sumber air baku bagi Instalasi Pengolahan Air (IPA) yang dibangun guna melayani kebutuhan air bersih masyarakat di kota Tembilahan Ibukota Kabupaten Indragiri.

Pada tahun 1983 prasarana yang dibangun telah mulai dioperasikan untuk melayani kebutuhan air bersih masyarakat kota Tembilahan dengan membentuk Badan Pengelola Air Minum (BPAM) berdasarkan Surat Keputusan Menteri Pekerjaan Umum Republik Indonesia Nomor : 148/KPTS/CK/1983 Tanggal 20 Agustus 1983.

Pada tahun 1992, tepatnya Tanggal 28 Nopember 1992 Badan Pengelola Air Minum (BPAM) diserah terimakan pengelolaanya dari Menteri Pekerjaan Umum RI kepada Pemerintah Daerah Tingkat I Riau melalui Surat Keputusan Menteri

Pekerjaan Umum RI Nomor : 759/KPTS/1992, Tanggal 24 Nopember 1992, dan selanjutnya oleh Pemerintah Daerah Tingkat I Riau menyerahkan kepada Pemerintah Daerah Tingkat II Indragiri Hilir melalui Berita Acara Penyerahan Pihak Pertama (Pemda Tk. I Riau) Nomor : 174/BA/1992, dan Pihak Kedua (Pemda Tk. II Indragiri Hilir) Nomor : 3584/UM.1992/690.

Pemerintah Daerah Tingkat II Indragiri Hilir telah mengesahkan Peraturan Daerah Tentang Pendirian Perusahaan Daerah Air Minum Tirta Indragiri Nomor : 2 Tahun 1990, dan telah disahkan oleh Gubernur Kepala Daerah Tingkat I Riau, Nomor : KPTS.325/VI/91 Tanggal 15 Juni 1991, yang selanjutnya diundangkan dalam Lembaran Daerah Kabupaten Daerah Tingkat II Indragiri Hilir Nomor : 11 Tahun 1991 Tanggal 25 September 1991 seri D Nomor 8. Dan selanjutnya sejak tahun disyahkannya Perda Pendirian tersebut, pengelolaan sarana prasarana air minum sepenuhnya menjadi tanggung jawab Perusahaan Daerah Air Minum (PDAM) Tirta Indragiri sebagai suatu Badan Usaha Milik Pemerintah Kabupaten Indragiri Hilir yang dalam Peraturan Daerah tersebut didirikan dengan tujuan sebagai berikut :

1) Memproduksi dan mendistribusikan air yang memenuhi syarat kesehatan kepada masyarakat di kabupaten Indragiri Hilir.
2) Melaksanakan fungsi sebagai suatu perusahaan yang efisien sehingga mampu memperoleh keuntungan untuk mengembangkan pelayanan tanpa melupakan fungsi social kemasyarakatan.
3) Mampu menjadi salah satu alternatif sumber pendapatan asli daerah melalui kontribusi keuntungan yang diperoleh tanpa mengabaikan upaya pengembangan perusahaan dan tidak memberatkan masyarakat.

Pada awal beroperasinya, tahun 1992 PDAM Tirta Indragiri hanya mengelola aset-aset yang telah dibangun sejak tahun 1983 dari proyek BPAM yang dibangun oleh Departemen Pekerjaan Umum RI dengan kapasitas produksi 20 l/dt yang berada di desa Pulau Palas yang khusus untuk melayani masyarakat dikota Tembilahan. Dengan rentang panjang pipa transmisi 10.500 M, yang berdiameter 200 MM, jenis pipa ACP yang dibangun tahun 1984. Sedangkan untuk pipa distribusinya berdiameter 250 MM jenis pipa DCIP, 250 MM jenis

pipa PVC, 200 MM, 50 MM jenis pipa ACP, 150 MM, 100 MM, 75 MM dan 50 MM jenis pipa PVC dengan keseluruhan panjang pipa distribusi di kota Tembilahan 17.750 M. Dari kapasitas terpasang sebagaimana tersebut, jumlah pelanggan yang baru dapat dilayani hanya 376 SR (sambungan rumah) pada tahun 1992 dengan jam pelayanan 7 jam setiap harinya. Dengan kondisi yang masih sangat terbatas ini praktis PDAM hanya mampu melayani 5,3% dari jumlah penduduk.

Selanjutnya pada tahun 1993, melalui proyek pengembangan air bersih, dibangun 1 paket Instalasi Pengolahan Air (IPA) lengkap dengan kapasitas 20 l/dt di desa Pulau Palas, sehingga tahun 1994 PDAM Tirta Indragiri memiliki 2 unit IPA dengan kapasitas 40 l/dt. Pada tahun yang sama jaringan pipa transmisi ditambah sepanjang 10.500 m lagi dengan ukuran diameter 250 mm jenis pipa PVC dari desa Pulau Palas ke kota Tembilahan, yang dengan demikian terdapat 2 jaringan pipa transmisi yang *mensuplai reservoir* 450 M3 yang dibangun tahun 1979-1980 pada buster pump parit 7 Tembilahan. Jaringan pipa distribusi terus pula bertambah sampai akhir tahun 1993 sepanjang 30.000 M sehingga jumlah keseluruhan panjang pipa distribusi dari diameter 250 MM s/d 75 MM dibangun sejak tahun 1994 adalah 47.750 M yang berada di kota Tembilahan.

Sejak tahun 1994, dengan peningkatan kapasitas produksi dan jaringan yang ada, PDAM Tirta Indragiri terus menambah jumlah pelanggan kota Tembilahan. Pada tahun 1994 telah dapat melayani sebanyak 9,12% penduduk kota Tembilahan dengan jumlah sambungan rumah 1.009 SR. Namun sejalan dengan berjalannya waktu, jaringan pipa transmisi yang dibangun tahun 1984 jenis pipa SCP sudah mengalami banyak kebocoran, dan pada beberapa ruas pipa sepanjang desa Pulau Palas ke buster pump parit 7 sering mengalami gangguan akibat korosif. Keadaan ini membuat biaya perawatan cukup tinggi, dengan kemampuan yang juga terbatas, suplai air bersih ke buster pump parit 7 sering terganggu dan sering terjadi kemacetan. Kondisi inilah yang pada gilirannya berdampak pada menurunnya pelayanan kepada pelanggan. Hampir 40% pelanggan yang ada sudah tidak mau lagi atau tidak disiplin membayar.

Kondisi ini sampai akhir tahun 1998, meskipun kapasitas Instalasi Pengolahan Air Bersih (IPA) sudah cukup memadai yaitu 110 l/dt, dengan panjang pipa distribusi sudah mencapai 88.900 M yang berada di kota Tembilahan, dengan cakupan pelayanan mencapai 14,7% atau 3.095 Sambungan Rumah. Karena pipa transmisi masih terbatas yang lama jenis ACP (dibangun tahun 1984) sudah tidak mungkin lagi dipakai, pelayanan yang diberikan PDAM masih belum membaik, terjadi banyak tunggakan karena komplain pelanggan yang tidak mendapatkan suplai air bersih secara *kontiniu*.

Tahun 1999 s/d 2000 pipa transmisi dibangun dengan diameter 300 MM jenis pipa PVC sepanjang 10.500 M, sejak tahun 2001 s/d awal tahun 2005 pelayanan semakin membaik, sehingga awal tahun 2005 PDAM Tirta Indragiri untuk kota Tembilahan telah mampu menambah cakupan pelayanannya hingga 36,6% atau 5.900 Sambungan Rumah, dengan kontinuitas pelayanan 24 jam, tahun 2010 PDAM telah membangun jaringan pipa transmisi baru 16 inci jenis PE 100 yang akan diselesaikan tahun 2010 ini sepanjang 11.000 meter dari instalasi Pulau Palas sampai ke Boster pump yang berada di parit 7 Tembilahan hulu.

Sejalan dengan bertambahnya usia, sebagai Perusahaan Daerah, PDAM Tirta Indragiri terus berbenah dan mengembangkan kinerjanya, sampai awal tahun 2010 telah memiliki 1 kantor pusat yang berada di Tembilahan ibukota kabupaten dan 18 cabang pelayanan yang berada di ibukota kecamatan, serta 4 unit pelayanan desa yang berada dalam wilayah Kabupaten Indragiri Hilir dengan cakupan pelayanan pada awal tahun 2010 mencapai 13.804 Sambungan Rumah.

2.2 Logo PDAM Tirta Indragiri

Gambar 2.2 Logo PDAM Tirta Indragiri

2.3 Visi dan Misi Perusahan PDAM Tirta Indragiri Tembilahan

1. Visi Perusahaan

Menjadi salah satu PDAM terbaik untuk katogori PDAM Kabupaten se-Sumatera 2016

2. Misi Perusahaan

a. Memberikan layanan air minum kepada masyarakat secara berkesinambungan dengan mengutamakan kepuasan pelanggan
b. Meningkatkan kontribusi perusahaan kepada Pemerintah Daerah
c. Meningkatkan profesionalisme Sumber Daya Manusia
d. Turut melestarikan Sumber Daya Air

2.4 Konsep Dasar Sistem

2.4.1 Pengertian Sistem

Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran yang tertentu. Pendekataan sistem yang merupakan jaringan kerja dari prosedur lebih menekankan urut-urutan operasi di dalam sistem. Dengan demikian definisi ini akan mempunyai peranan yang penting di dalam pendekatan untuk mempelajari suatu sistem. Pendekatan sistem yang merupakan kumpulan dari elemen-elemen atau komponen-komponen atau subsistem-subsistem merupakan definisi yang lebih luas (Jogyianto, 2005, 1).

2.4.2 Karakteristik Sistem

Suatu sistem mempunyai karekteristik atau sifat-sifat yang tertentu, yaitu mempunyai komponen-komponen (*componen),* batas sistem (*bondary),* lingkungan luar sistem (*environments),* penghubung (*interface),* masukan (*input),* keluaran (*output),* pengolah (*process)* dan sasaran (*objectives)* atau tujuan (*goal).*

a. Komponen sistem

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu

subsistem atau bagian-bagian dari sistem. Setiap subsistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan.

b. Batasan Masalah

   Batasan sistem (*boundary)* merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem di pandang sebagai satu kesatuan. Batas suatu sistem menunjukan ruang lingkup (*scope)* dari sistem tersebut.

c. Lingkungan Luar Sistem

   Lingkungan luar (*environment)* dari suatu sistem adalah apapun di luar batas dari sistem yang mempengaruhi sistem operasi. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan energi dari sistem tersebut.

d. Penghubung Sistem

   Penghubung (*interface)* merupakan media penghubung antara satu subsistem dengan sistem sistem lainya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsitem lainnya.

e. Masukan Sistem

   Masukan (*input)* adalah energi yang di masukkan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintenance input)* dan masukan sinyal (*signal input)*. (*maintenance input)* adalah energi yang di masukan supaya sistem tersebut dapat beroperasi. *Signal input* adalah energi yang di proses untuk di dapatkan keluaran.

f. Keluaran Sistem

   Keluaran (*output*) adalah hasil dari energi yang diolah dan di klasifikasi menjadi keluarkan yang berguna dan sisa pembuangan. Keluaran dapat merupakan masukan untuk subsistem yang lain atau kepada supra sistem.

g. Pengelolah Sistem

Suatu sistem dapat mempunyai suatu bagian pengelolah yang akan merubah masukan menjadi keluaran. Suatu sistem produksi akan mengelilah masukan berupa bahan baku dan bahan-bahan yang lainnya menjadi keluaran berupa barang jadi.

h. Sasaran Sistem

Suatu sistem pasti mempunyai tujuan (*goal)* atau sasaran (*objective).* Sasaran dari sistem sangat menentukan sekali masukan yang di butuhkan sistem dan keluaran yang akan dihasilkan sistem. Suatu sistem di katakan berhasil bila mengenai sasaran atau tujuannya.

2.5 Konsep Dasar Informasi

2.5.1 Pengertian Informasi

Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya, sedangkan data merupakan sumber informasi yang menggambarkan suatu kejadian yang nyata. Informasi merupakan data yang telah diproses sedemikian rupa sehingga meningkatkan pengetahuan seseorang yang menggunakan data tersebut. Informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengembilan keputusan saat ini atau saat mendatang (Jogiyanto, 2005, 8).

2.5.2 Kualitas Informasi

Kualitas dari suatu informasi *(quality of information)* tergantung dari tiga hal, yaitu:

1. Akurat, berarti informasi harus bebas dari kesalahan-kesalahn dan tidak bias atau menyesatkan. Akurat juga berarti informasi harus jelas mencerminkan maksudnya.
2. Tepat pada waktunya, berarti informasi yang datang pada penerima tidak boleh terlambat. Informasi yang sudah usang tidak akan mempunyai nilai lagi. Karena informasi merupakan landasan di dalam pengambilan keputusan.

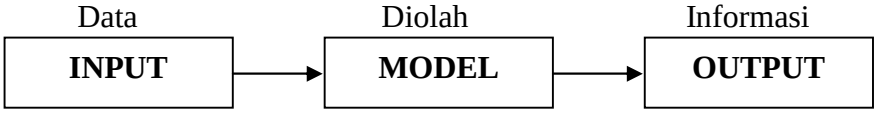
3. Relevan, berarti informsi tersebut mempunyai manfaat untk pemakainya. Relevansi informas untuk tiap-tiap orang satu dengan yang lainnya berbeda. Misalnya informasi mengenai sebab-musabab kerusakan mesin produksi kepada akuntan perusahaan adalah kurang relevan dan akan lebih relevan bila ditujukan kepada ahli teknik perusahaan. Sebaliknya informasi mengenai harga pokok produksi untuk ahli teknik merupakan informasi kurang relevan, tetapi relevan untuk akuntan (Jogiyanto, 2005,10).

2.6 Konsep Dasar Sistem Informasi

2.6.1 Definisi Sistem Informasi

Informasi dapat diperoleh dari sistem informasi (*information system*s) atau disebut juga dengan *processing systems* atau *information processing systems* atau *information - generating systems*. Sistem informasi didefenisikan oleh Robert A. Leitch dan K. Roscoe Davus yaitu suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan (Jogiyanto, 2005, 11).

2.6.2 Komponen Sistem Informasi



Gambar 2.3 Siklus Pengolahan Data

Dari gambar diatas, terlihat bahwa untuk melakukan siklus pengolahan data diperlukan tiga buah komponen, yaitu komponen input, komponen model dan komponen output. Data yang belum diolah perlu disimpan untuk pengolahan lebih lanjut, karena tidak semua data yang diperoleh langsung diolah (Jogiyanto ,2005).

John Burch dan Gary Gurdnitski mengemukakan bahwa sistem informasi terdiri dari komponen-komponen perangkat keras, perangkat lunak, *database*,

telekomunikasi, dan manusia. Sementara Burch dan Grudnistki (1986) berpendapat bahwa sistem informasi yang terdiri dari komponen-komponen diatas disebut dengan istilah blok bangunan (*building block*), yaitu blok masukan *(input*), blok model (*model block*), blok keluaran (*output block*), blok teknologi (*technologi block*), dan blok kendali (*control block*) (Jogiyanto, 2005, 12).

Keenam blok tersebut masing-masing saling berinteraksi satu sama lainnya membentuk satu kesatuan untuk mencapai sasarannya. Keenam block tersebut yaitu:

1. Blok masukan

   Input Mewakili data yang masuk ke dalam sistem informasi. Input termasuk metode dan media untuk memperoleh data yang akan dimasukkan, yang dapat berupa dokumen dasar.

2. Blok model

   Blok ini terdiri dari kombinasi prosedur, logika dan model matematik yang akan memanipulasi/ mentransformasi data masukan dan data yang tersimpan dalam basis data untuk menghasilkan keluaran yang diinginkan.

3. Blok keluaran

   Produk dari sistem informasi adalah keluaran berupa informasi yang berkualitas dan dokumentasi yang berguna untuk semua semua pemakai sistem.

4. Blok teknologi

   Teknologi merupakan kotak alat (*tool-box*) dalam sistem informasi. Teknologi digunakan untuk menerima input, menjalankan model, menyimpan dan mengakses data, menghailkan sekaligus mengirimkan keluaran dan membantu pengendalian dari sistem secara keseluruhan.

5. Blok basis data

   Merupakan kumpulan dari file data yang saling berhubungan satu dengan lainnya, tersimpan dalam komputer dan digunakan perngkat lunak untuk memanipulasinya.

6. Blok kendali

   Pengendalian perlu dirancang dan diterapkan untuk menyakinkan bahwa hal-hal yang dapat merusak sistem dapat dicegah atau bila terlanjur terjadi kesalahan dapat langsung diatasi.

2.7 Analisis Sistem

2.7.1 Definisi Analisis Sistem

Analisis sistem (*System analysis*) dapat didefenisikan sebagai penguraian dari suatu sistem informasi yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan-permasalahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat di usulkan perbaikan-perbaikannya (Jogiyanto, 2005, 129)

2.7.2 Langkah-langkah Analisis Sistem

Didalam tahap analisis sistem terdapat langkah-langkah dasar yang harus dilakukan oleh analisis sistem yaitu (Jogiyanto, 2005, 130):

1. *Identify*, yaitu mengidentifikasi masalah
2. *Understand*,yaitu memahami kerja sistem yang ada
3. *Analyze*, yaitu menganalisis sistem
4. *Report*, yaitu membuat laporan hasil analisis

2.7.3 Desain Sistem

Tujuan dari desain sistem secara umum adalah untuk memberikan gambaran secara umum kepada *use*r tentang sistem yang baru dan mengidentifikasikan komponen-komponen sistem informasi yang akan didesain secara rinci (Jogiyanto, 2005, 196). Desain sistem dapat didefenisikan oleh John Burch & Gary Grudnitski sebagai berikut :

"*system design can be defined as the drawing, planning, sketching or arranging of many saparate elements into aviable, reunifield awhole* (Desain sistem dapat didefenisikan sebagai penggambaran, perencanaan dan pembuatan

sketsa dan pengaturan dari beberapa elemen yang terpisah kedalam suatu kesatuan yang utuh dan berfungsi)".

Untuk mencapai tujuan ini, sasaran-sasaran yang harus dicapai adalah (Jogiyanto, 2005, 197):

a. Desain sistem harus berguna, mudah dipahami dan mudah digunakan
b. Desain sistem harus dapat mendukung tujuan utama perusahaan sesuai dengan yang telah didefenisikan pada tahap perencanaan sistem yang dilanjutkan pada tahap analisa sistem
c. Desain sistem harus efisien dan efektif untuk dapat mendukung pengolahan transaksi
d. Desain sistem harus dapat mempersiapkan rancang bangun yang terinci untuk masing-masing komponen

2.8 Pendekatan Berorientasi Objek

Pendekatan berorientasi objek merupakan paradigma pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek.

Pada Objek Oriented terdapat beberapa model pendekatan, yaitu *Object Oriented Programming* (OOP) dan *Object Oriented Analysis and Design* (OOAD). *Object Oriented Programming* (OOP) atau Pemrograman Berorientasi Objek adalah konsep pemrograman yang difokuskan pada penciptaan kelas yang merupakan abstraksi/ *blueprint/ prototype* dari suatu objek. Kelas ini harus mengandung sifat (data) dan tingkah laku *(method)* umum yang dimiliki oleh objek-objek yang kelak akan dibuat (*instansiasi*). Data dan *method* merupakan anggota dari suatu kelas. Sedangkan *Object Oriented Analysis and Design* (OOAD) adalah metode analisis yang memeriksa *requirements* dari sudut pandang kelas dan objek yang ditemui dalam ruang lingkup permasalahan yang mengarahkan arsitektur *software* yang didasarkan pada manipulasi objek-objek sistem atau subsistem. OOAD merupakan cara baru dalam memikirkan suatu masalah dengan menggunakan model yang dibuat menurut konsep sekitar dunia nyata. Dasar pembuatan adalah objek, yang merupakan kombinasi antara struktur

data dan perilaku dalam satu entitas. Dari beberapa model *Object Oriented* diatas, maka dalam peneletian ini, saya akan menggunakan model *Object Oriented Analysis and Design* (OOAD).

2.8.1 Konsep OOAD

OOAD mencakup analisis dan desain sebuah sistem dengan pendekatan objek, yaitu *Object Oriented Analysis* (OOA) dan *Object Oriented Design* (OOD). OOA adalah metode analisis yang memerika *requirement* (syarat/ keperluan) yang harus dipenuhi sebuah sistem) dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup perusahaan. Sedangkan OOD adalah metode untuk mengarahkan arsitektur *software* yang didasarkan pada manipulasi objek-objek sistem atau subsistem. Terdapat beberapa konsep dalam OOAD, yaitu :

1. Objek *(object)*
    a. *Object* adalah benda secara fisik dan konseptual yang ada di sekitar kita. Sebuah objek memiliki keadaan sesaat yang disebut *state*.
    b. *State* dari sebuah objek adalah kondisi dari objek atau himpunan keadaan yang menggambarkan objek tersebut. *State* dinyatakan dengan nilai dari atribut objeknya.
    c. *Atribut* adalah nilai internal suatu objek yang mencerminkan karakteristik objek, kondisi sesaat, koneksi dengan objek lain dan identitas.
    d. *Behaviour* (perilaku objek) mendefinisikan bagaimana sebuah objek bertindak dan memberi reaksi. Behaviour ditentukan oleh himpunan semua atau beberapa operasi yang dapat dilakukan oleh objek tersebut, yang dicerminkan oleh *interface*, *service*, dan *method* dari objek tersebut.
    e. *Interface* adalah pintu untuk mengakses *service* dari objek
    f. *Service* adalah fungsi yang dapat dikerjakan oleh sebuah objek
    g. *Method* adalah *mekanisme internal* objek yang mencerminkan perilaku objek tersebut.

2. Kelas *(class)*

   Class adalah himpunan objek yang sejenis yaitu mempunyai sifat (*atribut*), perilaku umum (operasi), relasi umum dengan objek lain dan semantik umum. *Class* adalah abstraksi dari objek dalam dunia nyata. *Class* menetapkan spesifikasi perilaku dan atribut dari objek tersebut.

3. Kotak Hitam *(black boxes)*

   Sebuah objek adalah kotak hitam. Konsep ini menjadi dasar implementasi objek. Dalam operasi *object oriented* hanya developer yang dapat memahami detail proses yang ada didalam kotak tersebut, sedangkan *user* tidak perlu mengetahui apa yang dilakukan yang penting mereka dapat menggunakan objek untuk memproses kebutuhan mereka. Kotak hitam berisi kode dan data.

   a. *Encapsulation*, yaitu proses menyembunyikan detail implementasi sebuah objek. Untuk mengakses data objek tersebut adalah melalui *interface*. Untuk berkomunikasi dengan objek digunakan *message.*
   b. *Message* adalah permintaan agar objek menerima untuk membawa metode yang ditunjukkan oleh perilaku dan mengembalikan result dari aksi tersebut kepada objek pengirim (*sender).*

4. *Asosiasi* dan *Agregasi*

   a. *Asosiasi* adalah hubungan yang mempunyai makna antara sejumlah objek. Asosiasi digambarkan dengan sebuah garis penghubung diantara objeknya. Contohnya : *Asosiasi* karyawan dengan unit kerja. Setiap karyawan bekerja di satu unit kerja, sedangkan unit kerja dapat memiliki beberapa karyawan.
   b. *Agregasi* adalah bentuk khusus sebuah *asosiasi* yang menggambarkan seluruh bagian pada suatu objek merupakan bagian dari objek yang lain. Contohnya : Kopling dan piston adalah bagian dari mesin, sedangkan mesin, roda, body merupakan bagian dari sebuah mobil.

2.8.2 Teknik Pemodelan dalam OOAD

1. Model Objek
   a. Model objek Menggambarkan struktur statis dari suatu objek dalam sistem dan relasinya
   b. Model objek berisi diagram objek. Diagram objek adalah *graph* dimana *node*nya adalah kelas yang mempunyai relasi antar kelas.
2. Model Dinamik
   a. Model dinamik menggambarkan aspek dari sistem yang berubah setiap saat.
   b. Model dinamik dipergunakan untuk menyatakan aspek kontrol dari sistem.
   c. Model dinamik berisi state diagram. State diagram adalah graph dimana *node*nya adalah *state* dan *arc* adalah tarnsisi antara *state* yang disebabkan oleh *event.*
3. Model Fungsional
   a. Model fungsional menggambarkan transformasi nilai data di dalam sistem.
   b. Model fungsional berisi data *flow* diagram. DFD adalah suatu *graph* dimana *node*nya menyatakan proses dan *arc*nya adalah aliran data.

2.8.3 *Object Oriented Analysis* (OOA)

Menurut Adi Nugroho (2005), OOA adalah tahapan perangkat lunak dengan menentukan spesifikasi sistem (sering orang menyebutnya sebagai SRS/ *System Requirement Spesification*) dan mengidentifikasi kelas-kelas serta hubungannya satu terhadap yang lain.

Analisa Berorientasi Objek atau *Object Oriented Analysis* (OOA) dimulai dengan menyatakan suatu masalah, analisa membuat model situasi dari dunia nyata, mengambarkan sifat yang penting. Dalam menganalisa suatu sistem, analisa harus bekerja dengan pihak yang membutuhkan sistem untuk memahami masalah tersebut dengan jelas. Model analisis adalah abstraksi yang ringkas dan tepat dari apa yang harus dilakukan oleh sistem, dan bagaimana melakukannya.

Objek dalam model harus merupakan konsep domain dari aplikasi, Untuk memahami spesifikasi sistem, kita perlu mengidentifikasi para pengguna atau yang sering disebut sebagai aktor-aktor. Siapa aktor-aktor yang akan menggunakan sistem dan bagaimana mereka menggunakan sistem.

Menurut Ivar Jacobson (dikutip dari buku *Object Oriented System Development* tulisan Ali Bahrawai, 1999) memperkenalkan konsep *use case* sebagai skenario untuk menjelaskan interaksi pengguna dengan sistem. Konsep ini bekerja dengan baik sehingga ia menjadi elemen utama pada pengembangan sistem. Skenario-skenario adalah cara yang paling baik untuk menguji siapa yang berbuat apa pada interaksi antar objek dan peran apa yang mereka mainkan. Apa yang objek-objek mainkan untuk mencapai sasaran tertentu dinamakan kolaborasi (*collaboration*).

2.8.4 *Object Oriented Design* (OOD)

Desain Berorientasi Objek atau *Object Oriented Design* (OOD) merupakan tahap lanjutan setela analisis berorientasi objek, dimana tujuan sistem diorganisasi ke dalam sistem-sistem berdasarkan struktur analisis dan arsitektur yang dibutuhkan. Desainer sistem *(System Desginer)* menentukan karateristik penampilan secara optimal, menentukan strategi memecahkan masalah dan menentukan pilihan alokasi sumber daya.

Desain model yang digunakan berdasarkan model analisis dengan dilengkapi rincian untuk implementasi. Fokus dari desain objek *(object design)* adalah perencanaan struktur data dan algoritma yang diperlukan untuk implementasi setiap kelas. Objek domain aplikasi dan objek domain komputer dijelaskan dengan menggunakan konsep dan notasi berorientasi objek yang sama.

Menurut Adi Nugroho (2005), OOD adalah merancang kelas-kelas yang teridentifikasi selama tahap analisis dan antarmuka *(user interface).* Selama tahap ini, kita mengidentifikasi dan (mungkin) menambahkan beberapa objek dan kelas yang mendukung implementasi dari spesifikasi kebutuhan. Proses-proses dalam OOD adalah:

1. Mendefiniskan kontek dan mode dari penggunaan sistem.
2. Mendesain arsitrektur sistem.
3. Indentifikasi Onjek sistem utama.
4. Mengembangkan model desain.
5. Menentukan interface objek.

2.8.5 Karateristik dari Objek

1. Objek
   a. Identitas berarti bahwa data diukur mempunyai nilai tertentu yang membedakan entitas disebut Objek.
   b. Objek dapat kongkrit, seperti halnya arsip dalam sistem, atau konseptual seperti kebijakan penjadualan dalam *multiprocessing* pada sistem operasi.
   c. Setiap objek mempunyai sifat yang melekat pada identitasnya.
   d. Dua objek dapat berbeda walaupun bila semua nilai atributnya identik
2. Kelas Objek
   a. Kelas merupakan gambaran sekumpulan objek yang terbagi dalam atribut, operasi, metode, hubungan, dan makna yang sama.
   b. Suatu kegiatan mengumpulkan data *(atribut)* dan perilaku (operasi) yang mempunyai struktur data sama ke dalam satu grup.
   c. Kelas objek merupakan wadah bagi objek. Dapat digunakan untuk menciptakan objek.
   d. Objek mewakili fakta/ keterangan dari sebuah kelas.
3. Istilah-istilah Objek
   a. Atribut yaitu data item yang menegaskan objek
   b. Operasi yaitu fungsi di dalam kelas yang dikombinasikan ke bentuk tingkah laku kelas
   c. Metode yaitu pelaksanaan prosedur (badan dari kode yang mengeksekusi respon terhadap permintaan objek lain di dalam sistem).

2.8.6 Karakteritik Metodologi Berorientasi Objek

Metodologi pengembangan sistem berorientasi objek mempunyai tiga karakteristik utama, yaitu:

1. *Encapsulatio* (Pengkapsulan)

   Encapsulation merupakan dasar untuk pembatasan ruang lingkup program terhadap data yang diproses. Data dan prosedur atau fungsi dikemas bersama-sama dalam suatu objek, sehingga prosedur atau fungsi lain dari luar tidak dapat mengaksesnya. Data terlindung dari prosedur atau objek lain, kecuali prosedur yang berada dalam objek itu sendiri.

2. *Inheritance* (Pewarisan*)*

   Inheritance (pewarisan) adalah teknik yang menyatakan bahwa anak dari objek akan mewarisi data/ *atribut* dan metode dari induknya langsung. *Atribut* dan metode dari objek induk diturunkan kepada anak objek, demikian seterusnya, karena telah mewarisi sifat induknya. *Inheritance* mempunyai arti bahwa *atribut* dan operasi yang dimiliki bersama di antara kelas yang mempunyai hubungan secara hirarki. Suatu kelas dapat ditentukan secara umum, kemudian ditentukan spesifik menjadi subkelas. Setiap subkelas mempunyai hubungan atau mewarisi semua sifat yang dimiliki oleh kelas induknya, dan ditambah dengan sifat unik yang dimilikinya, kelas objek dapat didefinisikan *atribut* dan *service* dari kelas objek lainnya. *Inheritance* menggambarkan generalisasi sebuah kelas. Sifat yang dimiliknya oleh kelas induknya tidak perlu diulang dalam setiap subkelas. Sebagai contoh, Sedan dan Sepeda motor adalah subkelas dari Kendaraan Bermotor. Kedua subkelas mewarisi sifat yang dimiliki oleh Kendaraan Bermotor, yaitu:

   a. Mempunyai mesin
   b. Dapat berjalan

   Kedua subkelas mempunyai sifat masing-masing yang berbeda, misalnya jumlah roda, dan kemampuan untuk berjalan mundur yang

tidak dimiliki oleh sepeda motor. Beberapa faktor dari subkelas yang bersifat umum dan dimasukkan kedalam kelas induknya serta mewariskan sifat tersebut pada kelas turunannya, sehingga mengurangi pengulangan yang terjadi dalam desain dan pemrograman. Hal ini merupakan keuntungan dari sistem berorientasi objek.

3. *Polymorphism* (Perbedaan Bentuk)

   Polimorfisme yaitu konsep yang menyatakan bahwa seuatu yang sama dapat mempunyai bentuk dan perilaku berbeda. *Polimorfisme* mempunyai arti bahwa operasi yang sama mungkin mempunyai perbedaan dalam kelas yang berbeda. Kemampuan objek-objek yang berbeda untuk melakukan metode yang pantas dalam merespon *message* yang sama. Seleksi dari metode yang sesuai bergantung pada kelas yang seharusnya menciptakan Objek.

2.9 *SMS Gateway*

Menurut Janner Simarmata (2010, 366), aplikasi untuk transmisi teks kecil melalui standar GSM (*Global System for Mobile Communication)* adalah SMS. Pada kenyataannya, setiap telepon seluler yang kompatibel dengan GSM bisa mengirimkan dan menerima pesan teks SMS. Antar muka efektif yang sederhana dalam batas-batas *mobil device* mengijinkan pengguna untuk membaca dan menulis pesan dengan mudah dan cepat. Aplikasi SMS sangat terintegrasi baik dengan *device*, seperti antar muka yang menyajikan kunci langsung untuk membaca dan menulis pesan. Peningkatan *usabilitas* lainnya mencakup masukan teks yang bersifat prediksi dan mempercepat masukan teks pada *keypad*. SMS *gateway* adalah suatu *platform* yang menyediakan mekanisme untuk menghantar dan menerima pesan dari peralatan *mobile* (HP, *phone*, dll). (Ben Forta, 2005, 326).

SMS *gateway* merupakan pintu gerbang bagi penyebaran informasi dengan menggunakan SMS. Anda dapat menyebarkan pesan kebanyak nomor secara otomatis dan cepat yang langsung terhubung dengan *database* nomor-nomor ponsel saja, tanpa harus mengetik ratusan nomor dan pesan dari ponsel karena

semua nomor akan diambil secara otomatis dari database tersebut. SMS *gateway* pada dasarnya hampir sama dengan mengirimkan SMS melalui *handphone*, dan perbedaannya adalah perangkat yang digunakan bukan *handphone* tetapi modem GSM. Dan modem inilah yang dikendalikan oleh komputer menggunakan aplikasi SMS. (Daud Edison Tarigan, 2011, 2).

2.10 *Unified Modelling Language (UML)*

2.10.1 Pengenalan UML

Pada perkembangan perangkat lunak, diperlukan bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang di berbagai negara dapat mengerti pemodelan perangkat lunak. Banyak orang yang telah membuat bahasa pemodelan pembangunan perangkat lunak sesuai dengan teknologi pemrograman yang berkembang pada saat itu, misalnya sempat berkembang dan digunakan oleh banyak pihak adalah *Data Flow Diagram* (DFD) untuk memodelkan perangkat lunak yang menggunkan pemrograman prosedural atau stuktural, kemudian juga ada *State Transition Diagram* (STD) yang digunakan untuk memeodelkan sistem *real time* (waktu nyata).

Pada perkembangan teknik pemrograman berorientasi objek, munculah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak (Rosa A.S,M.Salahuddin, 2011, 118).

2.10.2 Sejarah UML

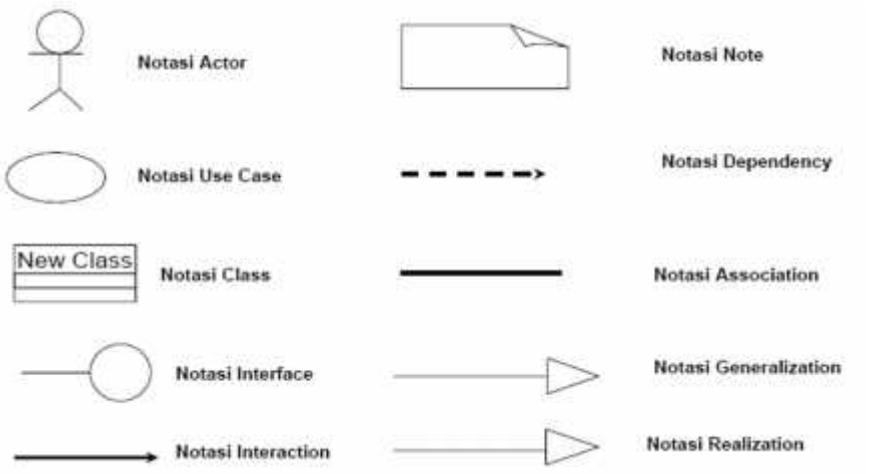
Bahasa pemrograman berorientasi objek yang pertama dikembangkan dikenal dengan Simula-67 yang dikembangkan pada tahum 1967. Bahasa pemrograman ini kurang berkembang dan dikembangkan lebih lanjut, namun dengan kemunculannya telah memberikan sumbangan yang besar pada depelover

pengembang bahasa pemrograman berorientasi objek selanjutnya (Rosa A.S,M.Salahuddin, 2011, 120).

2.10.3 Tujuan UML

Tujuan utama UML diantaranya yaitu untuk :

1. Menyediakan bahasa pemodelan visual yang expresip dan siap pakai untuk mengembangkan dan pertukaran model-model yang berarti
2. Menyediakan *mekanisme* perluasan dan spesialisasi untuk memperluas konsep-konsep inti
3. Mendukung spesifikasi independen bahasa pemrograman dan proses pengembangan tertentu
4. Menyediakan basis formal untuk pemahaman bahasa pemodelan
5. Mendukung konsep-konsep pengembangan level lebih tinggi seperti komponen, kolaborasi, *framework* dan *pattern*



Gambar 2.4 Notasi di Dalam UML

2.10.4 Diagram UML

2.10.4.1 Use Case Diagram

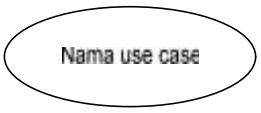
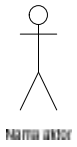
Use case adalah pemodelan untuk kelakukan sistem informasi yang akkan dibuat. Use case mendeskripsikan sebuah interaksi antara satu atau lebih

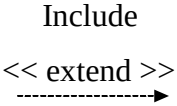
aktor dengan sistem informasi yang akan dibuat (Rosa A.S,M.Salahuddin, 2011, 130).

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.

Berikut adalah simbol-simbol yang ada pada diagram *use case:*

Tabel 2.1 Simbol *Use Case*

Simbol	**Deskripsi**
Use case 	Fungsionalisasi yang di sediakan sistem sebagai untit-unit yang saling bertukar pesan antar unit atau aktor, biasanya di nyatakan dengan menggunakan kata kerja di awal *frase name use case*.
Aktor/ actor 	Orang, proses atau sistem yang lain yang berinteraksi dengan sistem informasi yang akan di buat diluar sistem yang akan di buat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tetapi aktor belum tentu menggunakan orang, biasanya di nyatakan menggunakan kata benda di awal *frase* nama aktor
Asosiasi/ association ———————	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor
Ekstensi/ extend << extend >> ----------►	Relasi*use case* tambahan ke sebuah *use case* dimana use case yang di tambahkan dapat berdiri sendiri walaupun tanpa *use case* tambahan itu mirip dengan prinsip *inheritance* pada pemograman beroriantasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahakan

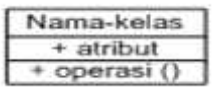
Simbol	Deskripsi
	misal arah panah mengarah pada *use case* yang di tambahakan
Generalisasi	Hubungan generalisasi dan spesialisasi antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
Include 	Relasi *use case* tambahakan ke sebuah *use case* dimana *use case* di tambah akan memerlukan *use case* ini menjalakan fungsinya atau syarat di jalankan *use case* ini

Gambar 2.5 *Use Case* Kegiatan Pasien Membuat Janji

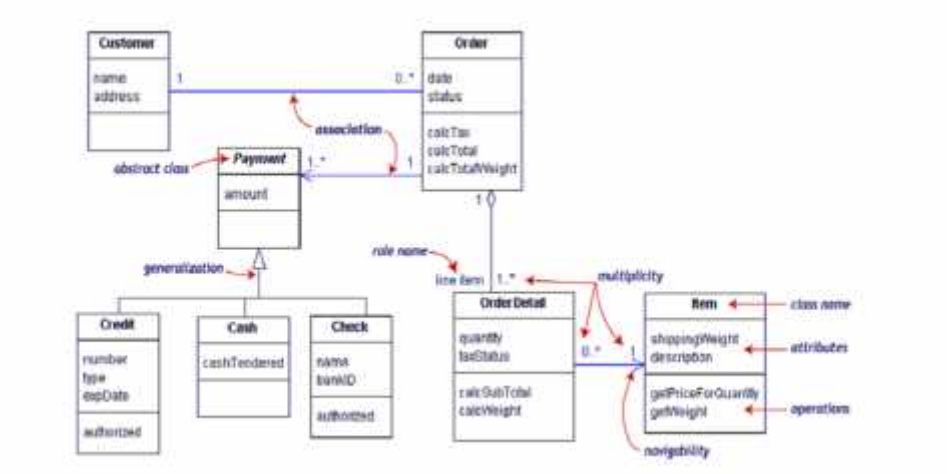
2.10.4.2 *Class Diagram*

Class Diagram menggambarkan struktur sistem dari segi pendefenisian kelas-kelas yang akan dibuat untuk membangun sistem (Rosa A.S,M.Salahuddin, 2011, 122).

Tabel 2.2 Simbol *Class Diagram*

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
Antarmuka/ *interface*  Nama_*interface*	Sama dengan konsep *interface* dalam pemograman berorientasi objek

Simbol	Deskripsi
Asosiasi/ *association* ———————	Relasi antar kelas dengan makna umum, asosiasi biasanya juga di sertai dengan *multiplicity*
Generalisasi	Relasi antar kelas dengan makna generalisasi-spesifikasi (umum-khusus)
Keberuntung/*dependency*	Relasi antar kelas dengan makna ketergantungan antar kelas
Agregasi/ *aggregation* ————◇	Relasi antar kelas dengan makna semua bagian



Gambar 2.6 *Class Diagram* Transaksi Pembelian Barang

2.10.4.3 *Activity Diagram*

Activity diagram menggambarkan berbagai aliran kerja atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah diagram aktivitas menggambaran aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan sistem (Rosa A.S,M.Salahuddin, 2011, 134).

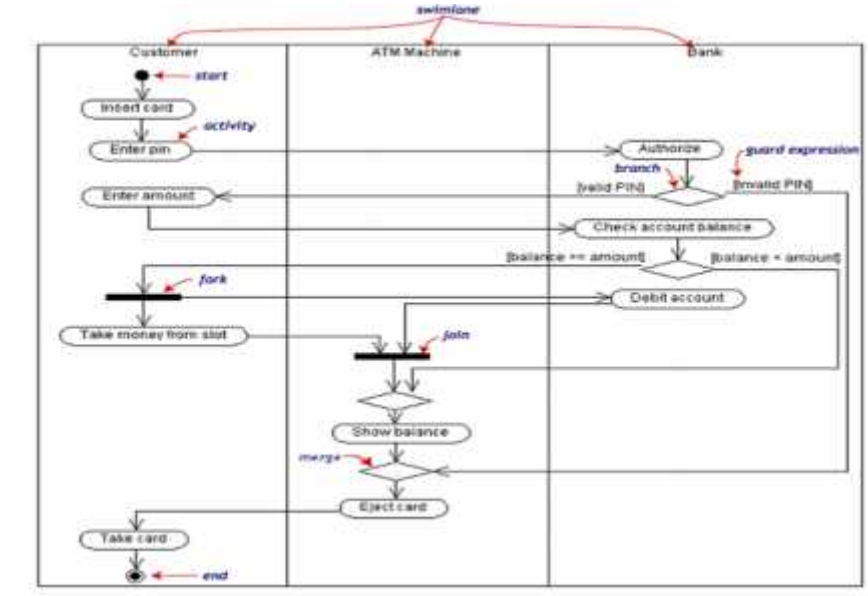
Diagram aktivitas juga banyak di gunakan untuk mendefenisikan hal-hal sebagi berikut :

a. Rancangan proses bisnis dimanasetiap urutan aktivitas yang di ambarkan merupakn proses bisnis sistem yang di defenisikan

b. Urutan atau penelompokan tampilan dari sistem/ *user interface* di mana setiap aktivitas di angap memiliki sebuah rancangan antarmuka tampilan
c. Rancangan pengujian dimana setiap aktivitas di angap memerlukan sebuah pengujian yang di perlukan didefenisikan kasus ujianya

Tabel 2.3 Simbol *Activity diagram*

Simbol	Deskripsi
Status awal	Status awal aktivitas sistem, sebuah diagram aktivitas memilih sebuah status awal.
Aktivitas	Aktivitas yang di lakukan sistem aktivitas biasanya di awali dengan kata kerja
Percabangan/ *decision*	Asosiasi percabangan dimana ada pilihan aktivitas lebih dari
Penggabungan/ *join*	Asosiasi penggabungan dimana lebih satu aktivitas di gabung menjadi satu
Status akhir	Status akhir yang di lakukan sistem sebuah diagram aktivitas memiliki sebuah status akhir

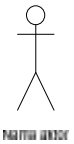
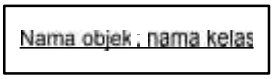
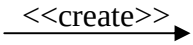


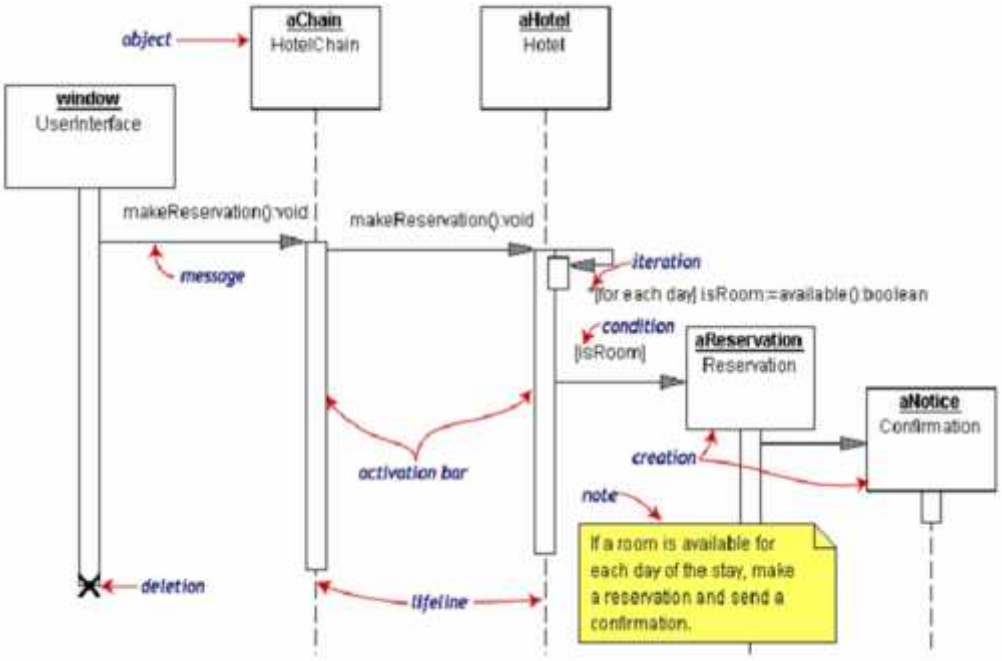
Gambar 2.7 *Aktivity Diagram* Pengambilan Uang di ATM

2.10.4.4 *Sequence Diagram*

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirim dan diterima antar objek (Rosa A.S,M.Salahuddin, 2011, 137)

Tabel 2.4 Simbol *Sequence diagram*

Simbol	Deskripsi
Aktor 	Orang, proses atau sistem yang lain yang berinteraksi dengan sistem informasi yang akan di buat diluar sistemyang akan di buat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tetapi aktor belum tentu menggunakan orang, biasanya di nyatakan menggunakan kata bemda di awal *frase* nama aktor
Garis hidup / *lifeline*	Menyatakan hidup suatu objek
Objek 	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
Pesan tipe create 	Objek yang lain, arah panah mengarah pada objek yang di buat



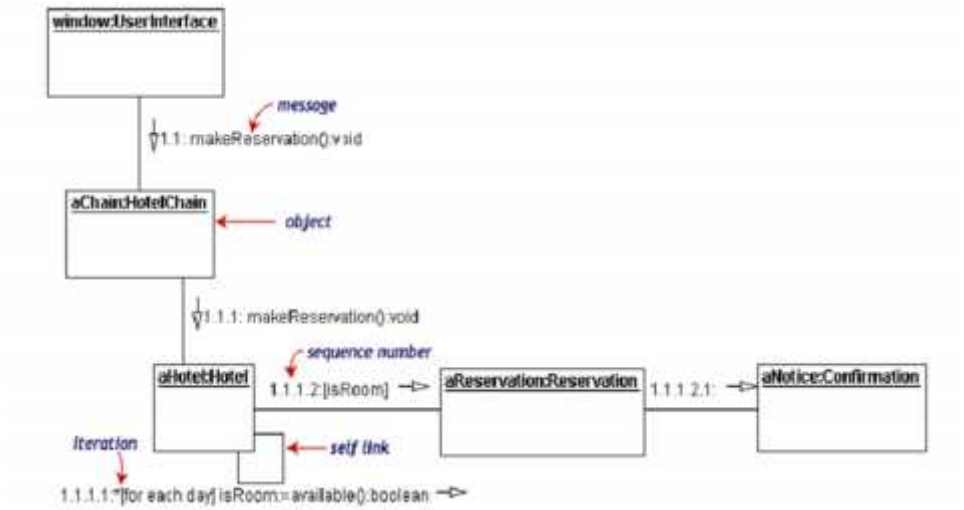
Gambar 2.8 *Sequence* Diagram Pemesanan Kamar Hotel

2.10.4.5 *Colaboration Diagram*

Colaboration adalah tipe diagram interaksi yang mendeskripsikan *layout* organisasi dari objek-objek yang menggirim dan menerima pesan (Bambang Hariyanto, 2004). Melihat pada interaksi dan hubungan terstruktur antar obyek. Tipe diagram ini menekankan pada hubungan *(relationship)* antar obyek, sedangkan *sequence* diagram menekankan pada urutan kejadian. *Collaboration* diagram digunakan sebagai alat untuk menggambarkan interaksi yang mengungkapkan keputusan mengenai perilaku sistem.

Tabel 2.5 Simbol *Colaboration Diagram*

Simbol	***Deskripsi***
Object : Object1	*Object* merupakan *instance* dari sebuah *class.* Digambarkan sebagai sebuah c*lass* (kotak) dengan nama obyek didalamnya yang diawali dengan sebuah titik koma.
Actor 	*Actor* juga dapat berkomunikasi dengan *object*, maka *actor* juga dapat disertakan ke dalam *collaboration* diagram. Simbol *Actor* sama dengan simbol pada *Actor Use Case* Diagram.



Gambar 2.9 Colloboration Diagram Pemesanan Kamar di Hotel

2.11 *Personal Home Page* (PHP)

PHP adalah bahasa standar yang di gunakan dalam dunia website. PHP adalah bahasa pemograman yang berbentuk *script* yang di letakkan di dalam web server. PHP di ciptakan dari ide Rasmus Lerdof yang membuat sebuah *script perl*. *Script* tersebut sebenarnya di maksudkan untuk di gunakan sebagai program untuk dirinya sendiri. Akan tetapi kemudian di kembangkan lagi sehingga menjadi sebuah bahasa yang disebut "*Personal Home Page*" (Bunafit Nugroho, 2004, 140).

PHP telah dicipta terutama untuk kegunaan web dan boleh menghubungkan *guery database* dan menggunakan *simple task* yang boleh di luruskan dengan 3 atau 4 baris kod saja. PHP adalah bahasa *programing* yang baru di bangun sekitar tahun 1994/ 1995. Malah penggunanya masih baru di malaysia dan sedang meningkat popular kegunaanya. PHP sebenarnya merupakan program yang berjalan pada *platform linux* sehingga program ini menjadi *free ware*. Selanjutnyan PHP mengalami perkembangan yakni di buat dalam versi *windows. Script* murni PHP dapat anda dapatkan pada alamat www.php.net. disana anda mendapatkan *script-script* PHP secara gratis mulai dari awal sampai dengan akhir.

Hampir seluruh aplikasi berbasis web dapat di buat dengan PHP namun fungsi PHP yang paling utama adalah untuk menghubungkan *database* dengan

web. Dengan PHP membuat aplikasi web yang terkoneksi ke *database* menjadi sangat mudah. Sistem *database* yang telah di dukung oleh PHP adalah :

a. Oracle
b. Sybase
c. Msql
d. MySQL
e. Solid
f. Generic ODBC
g. PostgresSQL

PHP juga mendukung komunikasi dengan lainnya melalui protokol IMAP, SNMP, NNTP, dan POP3 atau HTP.

2.12 *My SQL*

MySQL (*My Struktur Query Language)* atau atau dibaca "mai-se-kuel" adalah sebuah program pembuat *database* yang bersifat *open source* , artinya siapa saja boleh mengunakan dan tidak di cekal. Saat ini kita mendengar *open source* kita ingat dengan sistem operasi handal keturunan *unix*, yaitu *linux* (Bunafit Nugroho, 2004, 30).

MySQL sebenarnya produk yang berjalan pada *flatform linux*. Karena sifatnya yang *open source*, dia dapat di jalankan pada semua *platform* baik *windows* maupun *linux*. Selain itu, MySQL juga merupakan program pengakses untuk aplikasi *multi user* (banyak pengguna). Saat ini *database* MySQL telah di gunakan hampir oleh semua *programer database*, apalagi hampir semua *programer database*, apalagi dalam pemograman web.

Kelebihan lain MySQL adalah ia menggunakan bahasa *Query* standar yang di miliki oleh SQL (*Struktur Query Language)*. SQL adalah suatu bahasa permintaan yang berstruktur yang telah di standarkan untuk semua

program pengakses *database* seperti *Oracle, Posgres, SQL, SQL Server* dan lain- lain. Sebagai sebuah program penghasil *database*, MySQL tidak dapat berjalan sendiri tanpa adanya sebuah aplikasi lain (*interface)* . MySQL dapat di dukung oleh hampir semua program aplikasi baik yang *open source* seperti PHP

maupun yang tidak, yang ada pada *platform Windows* seperti *Visual Basic, Delphi*, dan lainnya.

Program-program yang menggunakan bahasa SQL antara lain :

a. MySQL
b. *Posgres SQL*
c. *Oracle*
d. *SQL Server* 97, 2000
e. *Interbase*

Dan program-program aplikasi pendukung MySQL antara lain:

a. PHP (Page Hipertext Preprosesor)
b. Visual Delphi
c. Visual Basic
d. Cold Funsion, dll

2.13 *Rational Rose*

Rational Rose adalah salah satu kakas (*tool)* pemodelan visual untuk pengembangan sistem berbasis objek yang sangat handal untuk digunakan sebagai bantuan bagi para pengembang dalam melakukan analisis dan perancangan sistem. *Rational Rose* digunakan untuk melakukan pemodelan sistem sebelum pengmbang menulis kode-kode dalam bahasa pemrograman tertentu. *Rational Rose* mendukung pemodelan bisnis, yang membantu para pengembang untuk memahami sistem secara komprehensif (Adi Nugroho, 2005, 20).

Dalam *UML* terdapat beberapa istilah yang sering digunakan yaitu sebagai berikut:

1. View

   Rational Rose memiliki empat *view* yaitu: *Use Case View*, *Logical View*, *Componen View* dan *Deployment View*.

2. Diagram

   Rational Rose memiliki delapan diagram yaitu: *Use case* diagram, *Sequence* diagram, *Collaboration* diagram, *Activity* diagram, *Class*

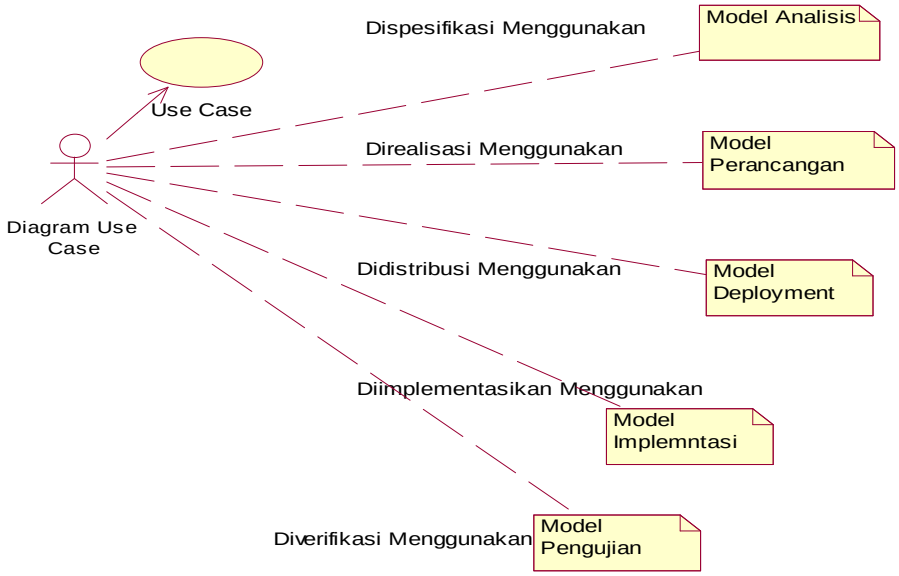
diagram, *State* diagram, *Component* diagram dan *Deployment* diagram.

3. Element Model

   Konsep-konsep yang digunakan dalam diagram merupakan elemen-elemen model yang menyatakan konsep berorientasi obyek secara umum, seperti *class*, *object* dan *message*, serta hubungan antar konsep termasuk *association*, *dependency* dan *generalization*.

2.14 *Unifield Software Development Process* (USDP)

USDP merupakan metode pengembangan/ rekayasa perangkat lunak yang berbasiskan komponen (*component based software engineering*), yang berarti sistem perangkat lunak yang kelak dihasilkan akan terdiri atas komponen-komponen perangkat lunak yang saling berhubungan melalui antarmuka yang terdefinisi dengan baik. USDP juga merupakan pengembangan sistem/ perangkat lunak yang dikendalikan *use case (use case driven software engineering)*, sehingga *use case* diagram merupakan kendali dalam seluruh tahapan pengembangan sistem/ perangkat lunak, mulai perencanaan analisis perancangan implementasi.



Gambar 2.10 Model *Use Case Driven Software Engineering* (USDP)