BAB II
LANDASAN TEORI

Pada bab ini akan diuraikan tentang teori-teori yang melandasi penulisan Laporan Penelitian ini.

2.1 Konsep Dasar Sistem

Pengertian dan definisi sistem pada berbagai bidang berbeda-beda, tetapi meskipun istilah sistem yang digunakan bervariasi,semua sistem pada bidang-bidang tersebut mempunyai beberapa persyaratan umum, yaitu sistem harus mempunyai elemen, lingkungan, interaksi antar elemen, interaksi antara elemen dengan lingkungannya, dan yang terpenting adalah sistem harus mempunyai tujuan yang akan dicapai.

2.1.1 Definisi Sistem

Pada umumnya setiap organisasi selalu mempunyai suatu sistem informasi untuk mengumpulkan, menyimpan, melihat dan menyalurkan informasi. Sistem informasi dapat terbentuk karena didorong oleh kebutuhan akan informasi yang terus meningkat yang dibutuhkan oleh pengambil keputusan. Berikut beberapa definisi sistem diantaranya :

a. Sistem adalah sekelompok bagian yang disusun dan diatur dengan baik yang bekerjasama untuk melakukan suatu maksud.[12]
b. Sistem adalah suatu grup dari elemen-elemen baik yang berbentuk fisik maupun bukan fisik yang menunjukan suatu kumpulan yang saling berhubungan dan berinteraksi bersama-sama menuju satu atau lebih tujuan, sasaran, atau akhir dari system.[13]

Dari beberapa definisi sistem diatas dapat disimpulkan bahwa sistem dikelompokan menjadi dua bagian yang menekankan pada prosedurnya dan ada yang menekankan pada elemennya. Kedua kelompok ini adalah benar dan tidak bertentangan, yang berbeda adalah cara pendekatannya.

2.1.2 Karakteristik Sistem[11]

Suatu sistem mempunyai karakteristik dan memiliki sifat-sifat tertentu dimana ada komponen sistem, batasan, lingkungan luar, penghubung, masukan sistem, keluaran sistem, pengolahan sistem, serta sasaran sistem. Yang dijelaskan sebagai berikut:

a. Komponen Sistem (*Components*)

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

b. Batas Sistem (*Boundary*)

   Batas sistem merupakan suatu daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya, yang menunjukkan ruang lingkup (*scope*) sistem tersebut.

c. Lingkungan Luar Sistem (*Environment*)

   Lingkungan luar dari suatu sistem adalah segala sesuatu di luar batas sistem yang mempengaruhi operasi sistem, dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut. Lingkungan luar sistem yang menguntungkan yaitu energi dari sistem dan dengan demikian harus tetap dijaga dan dipelihara. Sedangkan lingkungan luar sistem yang merugikan harus ditahan dan dikendalikan, kalau tidak akan menggangu kelangsungan hidup sistem.

d. Penghubung Sistem (*Interface*)

   Penghubung merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya. Dengan penghubung satu subsistem dapat berinteraksi dengan subsistem lainnya membentuk satu kesatuan. Keluaran dari satu subsistem memungkinkan menjadi masukan untuk subsitem lainnya melalui media penghubung.

e. Masukan Sistem (*Input*)

Masukan adalah energi yang dimasukan ke dalam sistem dan menentukan keluaran sistem. Masukan dapat berupa masukan perawatan (*maintenance input*) dan masukan sinyal (*signal input*). *Maintenance input* adalah energi yang dimasukan supaya system tersebut dapat beroperasi. *Signal input* adalah energi yang diproses untuk mendapatkan keluaran.

f. Keluaran Sistem (*Output*)

Keluaran adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan. Keluaran dapat merupakan masukan untuk subsistem lain.

g. Pengolah Sistem (*Process*)

Pengolah merupakan bagian yang merubah masukan menjadi keluaran.

h. Sasaran Sistem (*Objectives*) atau Tujuan (*Goal*)

Jika suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Sasaran dari sistem sangat menentukan masukan yang akan dibutuhkan oleh sistem dan keluaran yang akan dihasilkan sistem. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya.

2.1.3 Klasifikasi Sistem[11]

Sistem dapat diklasifikasikan dari beberapa sudut pandang. Mulai dari Sistem abstrak, alamiah, sistem tertentu, system tertutup, yang dijelaskan sebagai berikut:

a. Sistem Abstrak (*Abstract System*) dan Sistem Fisik (*Physical System*)

Sistem abstrak adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik. Sistem fisik merupakan sistem yang ada secara fisik.

b. Sistem Alamiah (*Natural System*) dan Sistem Buatan Manusia (*Human Made System*) Sistem alamiah adalah sistem yang terjadi melalui proses alam, tidak dibuat manusia. Sistem buatan manusia adalah sistem yang dirancang dan dibuat oleh manusia.

c. Sistem Tertentu (*Deterministic System*) dan Sistem Tak Tentu (*Probabilistic System*)

Sistem tertentu beroperasi dengan tingkah laku yang sudah dapat diprediksi. Sistem tak tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilitas.

d. Sistem Tertutup (*Closed System*) dan Sistem Terbuka (*Open System*)

Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkungan luarnya. Sistem ini bekerja secara otomatis tanpa adanya turut campur dari pihak luarnya. Sistem terbuka adalah sistem yang berhubungan dan terpengaruh dengan lingkungan luarnya. Sistem ini menerima masukan dan menghasilkan keluaran untuk lingkungan luar atau subsistem yang lainnya.

2.2 Pengertian Persediaan (*Inventory*)

Persediaan dilakukan oleh suatu perusahaan untuk mengantisipasi permintaan dari konsumen dan untuk kelancaran proses produksi. Berikut beberapa pengertian dari persediaan (*inventory*) :

1. Menurut Richardus Eko Indrajit dan Richardus Djokopranoto (2003, hal. 3), barang-barang yang biasanya dapat dijumpai di gudang tertutup, lapangan, gudang terbuka, atau tempat-tempat penyimpanan lain, baik berupa bahan baku, barang setengah jadi, barang jadi, barang-barang untuk keperluan operasi, atau barang-barang unutk keperluan suatu proyek.[10]
2. Menurut David W. Buker, Inc and associates (1990, *unit 7 page 1*) "*Inventory is the material use in production that flows through the manufacturing process*".[6]
3. Persediaan adalah barang-barang yang ada yang dapat dipakai, diambil, dijual, dan sebagainya.[12]

Istilah *inventory* dapat digunakan untuk mengartikan hal-hal sebagai berikut :

1. *Stock on hand* pada waktu tertentu (asset yang berwujud yang dapat dilihat, diukur dan dihitung).
2. Daftar *item* dari barang pada *property*.
3. (sebagai kata kerja) tindakan mengukur berat dan menghitung *item on hand*.

4. (untuk finansial dan akuntansi) nilai dari *stock* barang yang dimiliki oleh organisasi pada waktu tertentu.

2.2.1 Jenis-jenis Persediaan[2]

Jenis-jenis persediaan dalam suatu perusahaan menurut fungsinya dapat dibedakan atas :

1. *Bath Stock/Lot Size Inventory* adalah persediaan yang diadakan karena kita membeli atau membuat bahan-bahan atau barang-barang dalam jumlah yang lebih besar daripada jumlah yang dibutuhkan pada saat itu. Keuntungannya adalah:
    a. Potongan harga pada harga pembelian.
    b. Efisiensi produksi.
    c. Penghematan biaya angkutan.
2. *Fluctuation Stock* adalah persediaan yang diadakan untuk menghadapi fluktuasi permintaan konsumen yang tidak dapat diramalkan.
3. *Anticipation Stock* adalah persediaan yang diadakan untuk menghadapi fluktuasi permintaan yang dapat diramalkan, berdasarkan pola musiman yang terdapat dalam satu tahun dan untuk menghadapi penggunaan, penjualan, atau permintaan yang meningkat.

Setiap jenis persediaan memiliki karakteristik tersendiri dan cara pengelolan yang berbeda, sehingga dapat dilihat dari jenis dan posisi barang. Persediaan menurut jenis dan posisi barang dapat dibedakan menjadi beberapa jenis:

1. Persediaan bahan mentah (*raw material*) yaitu persediaan barang-barang berwujud, seperti besi, kayu, serta komponen-komponen lain yang dugunakan dalam proses produksi.
2. Persediaan bagian produk atau komponen-komponen rakitan (*purchased parts/components*), yaitu persediaan barang-barang yang terdiri dari komponen-komponen yang diperoleh dari perusahan lain yang secara langsung dapat dirakit menjadi suatu produk.

3. Persediaan bahan pembantu atau penolong (*supplies*), yaitu persediaan barang-barang yang diperlukan dalam proses produksi, tetapi bukan merupakan bagian atau komponen barang jadi.
4. Persediaan barang dalam proses (*work in process*), yaitu persediaan barang-barang yang merupakan keluaran dari tiap-tiap bagian dalam proses produksi atau yang telah diolah menjadi suatu bentuk, tetapi masih perlu diproses lebih lanjut menjadi barang jadi.
5. Persediaan barang jadi (*finished goods*), yaitu persediaan barang-barang yang telah selesai diproses atau diolah dalam pabrik dan siap dijual atau dikirim kepada pelanggan.

2.2.2 Pengelolaan Persediaan

Menurut R. E. Indrajit dan R. Djokopranoto (2003, hal. 11), prinsip pengelolaan persediaan yaitu penentuan jumlah dan jenis barang yang disimpan dalam persediaan haruslah sedemikian rupa sehingga produksi dan operasi perusahaan tidak terganggu, tetapi dilain pihak harus bisa menjaga agar biaya investasi yang ditimbulkan dari penyediaan barang tersebut seminimal mungkin.[5]

2.3 FIFO *(First in First Out)*[3]

FIFO adalah akronim untuk *First In First Out* (Pertama Masuk, Pertama Keluar), sebuah abstraksi yang berhubungan dengan cara mengatur dan memanipulasi data relatif terhadap waktu dan prioritas. Ungkapan ini menggambarkan prinsip teknik pengolahan antrian atau melayani permintaan yang saling bertentangan dengan proses pemesanan berdasarkan perilaku *first-come, first-served* (FCFS): di mana orang-orang meninggalkan antrian dalam urutan mereka tiba, atau menunggu giliran satu di sebuah sinyal kontrol lalu lintas.

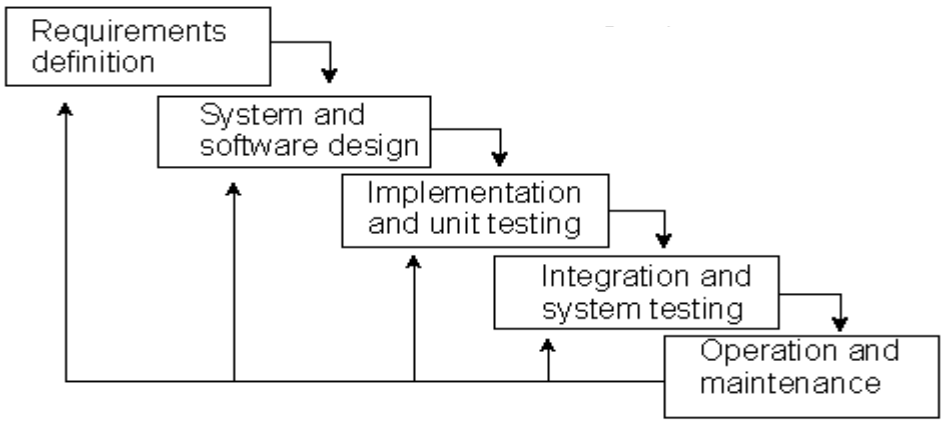
FCFS juga merupakan jargon istilah untuk sistem operasi penjadwalan algoritma FIFO, yang memberikan setiap proses CPU waktu sesuai dengan urutan mereka datang. Dalam arti yang lebih luas, abstraksi LIFO, atau *Last-In-First-Out* adalah kebalikan dari abstraksi organisasi FIFO. Bedanya mungkin adalah yang

paling jelas dengan mempertimbangkan sinonim yang kurang umum digunakan dari LIFO, FILO (berarti *First-In-Last-Out*). Pada intinya, keduanya adalah kasus khusus dari daftar yang lebih umum (yang dapat diakses di mana saja). Perbedaannya adalah tidak ada dalam daftar (data), tetapi dalam aturan untuk mengakses konten. Satu sub-tipe menambah satu ujung, dan melepaskan dari yang lain, sebaliknya mengambil dan menempatkan sesuatu hanya pada salah satu ujungnya.

Variasi bahasa populer pada pendekatan *ad-hoc* untuk menghapus item dari antrian telah diciptakan dengan nama OFFO, yang merupakan singkatan On-Fire-First-Out. Antrian prioritas adalah variasi pada antrian yang tidak memenuhi syarat untuk nama FIFO, karena tidak secara akurat menggambarkan perilaku struktur data. Teori antrian mencakup konsep yang lebih umum dari antrian, serta interaksi antara ketat-antrian FIFO.

2.4 Pembangunan Sistem dengan *Waterfall* [13]

Dalam pembangunan sistem ini digunakan model pengembangan *waterfall*. Inti dari metode *waterfall* adalah pengerjaan dari suatu sistem dilakukan secara berurutan atau secara linear. Jadi jika langkah satu belum dikerjakan maka tidak akan bisa melakukan pengerjaan langkah 2, 3 dan seterusnya. Secara otomatis tahapan ke-3 akan bisa dilakukan jika tahap ke-1 dan ke-2 sudah dilakukan.



Gambar 2.1 *Waterfall*

Secara garis besar metode *waterfall* mempunyai langkah-langkah sebagai berikut :

1. ***Requirements Definition***

   Langkah ini merupakan analisa terhadap kebutuhan sistem. Pengumpulan data dalam tahap ini bisa melakukan sebuah penelitian, wawancara atau *study literatur*. Seorang sistem analis akan menggali informasi sebanyak-banyaknya dari *user* sehingga akan tercipta sebuah sistem komputer yang bisa melakukan tugas-tugas yang diinginkan oleh *user* tersebut. Tahapan ini akan menghasilkan dokumen *user requirement* atau bisa dikatakan sebagai data yang berhubungan dengan keinginan *user* dalam pembuatan sistem. Dokumen ini lah yang akan menjadi acuan sistem analis untuk menterjemahkan ke dalam bahasa pemprogram.

2. ***System and Software Design***

   Proses desain akan menerjemahkan syarat kebutuhan ke sebuah perancangan perangkat lunak yang dapat diperkirakan sebelum dibuat coding. Proses ini berfokus pada : struktur data, arsitektur perangkat lunak, representasi *interface*, dan *detail* (algoritma) prosedural. Tahapan ini akan menghasilkan dokumen yang disebut *software requirement*. Dokumen inilah yang akan digunakan *proggrammer* untuk melakukan aktivitas pembuatan sistemnya.

3. ***Implementaion and Unit Testing***

   Coding merupakan penerjemahan *design* dalam bahasa yang bisa dikenali oleh komputer. Dilakukan oleh *programmer* yang akan meterjemahkan transaksi yang diminta oleh *user.* Tahapan ini lah yang merupakan tahapan secara nyata dalam mengerjakan suatu sistem. Dalam artian penggunaan komputer akan dimaksimalkan dalam tahapan ini. Setelah pengkodean selesai maka akan dilakukan *testing* terhadap sistem yang telah dibuat tadi. Tujuan *testing* adalah menemukan kesalahan-kesalahan terhadap sistem tersebut dan kemudian bisa diperbaiki.

4. ***Integration and System Testing***

   Tahapan ini bisa dikatakan *final* dalam pembuatan sebuah sistem. Setelah melakukan analisa, design dan pengkodean maka sistem yang sudah jadi akan digunakan oleh *user*.

5. ***Operation and Maintenance***

   Perangkat lunak yang sudah disampaikan kepada pelanggan pasti akan mengalami perubahan. Perubahan tersebut bisa karena mengalami kesalahan karena perangkat lunak harus menyesuaikan dengan lingkungan (periperal atau sistem operasi baru) baru, atau karena pelanggan membutuhkan perkembangan fungsional.

2.5 *Unified Modeling Language* (UML)[14]

UML adalah bahasa grafis untuk mendokumentasi, menspesifikasi, dan membangun sistem perangkat lunak. UML berorientasi objek, menerapkan banyak level abstraksi, tidak tergantung pada proses pengembangan, tidak bergantung bahasa dan teknologi, pemaduan beberapa notasi di beragam metodologi, usaha bersama dari banyak pihak, didukung oleh kakas-kakas yang diintegrasikan lewat XML (XMI). Standar UML dikelola oleh OMG (*Object Management Group*).

UML adalah bahasa pemodelan untuk menspesifikasikan, memvisualisasikan, membangun dan mendokumentasikan artifak-artifak dari sistem :

1. Di dalam *system intensive process*, metode diterapkan sebagai proses untuk menurunkan atau mengevolusikan sistem.
2. Sebagai bahasa, UML digunakan untuk komunikasi yaitu alat untuk menangkap pengetahuan (semantiks) mengenai satu subyek dan mengekspresikan pengetahuan (sintaks) yang memperdulikan subyek untuk maksud berkomunikasi. Subyek adalah sistem yang dibahas.
3. Sebagai bahasa pemodelan, UML fokus pada pemahaman subyek melalui formulasi model dari subyek (dan konteks yang terhubung). Model memuat

pengetahuan pada subyek, dan aplikasi dari pengetahuan ini berkaitan dengan intelejensia.

4. Berkaitan dengan unifikasi. UML memadukan praktek rekayasa terbaik sistem informasi dan industri, meliputi beragam tipe sistem (perangkat lunak dan non perangkat lunak), domain (bisnis, perangkat lunak), dan proses siklus hidup.
5. Begitu diterapkan untuk menspesifikasi sistem, UML dapat digunakan untuk mengkomunikasi "apa" yang diperlukan dari sistem dan "bagaimana" sistem dapat direalisasikan.
6. Begitu diterapkan untuk memvisualisasikan sistem, UML dapat digunakan untuk menjelaskan sistem secara visual sebelum direalisasikan.
7. Begitu diterapkan untuk membangun sistem, UML dapat digunakan untuk memandu realisasi sistem serupa dengan "*blueprint*".
8. Begitu diterapkan untuk mendokumentasikan sistem, UML dapat digunakan untuk menangkap pengetahuan mengenai sistem pada seluruh siklus hidup.

2.5.1 Tujuan UML

Tujuan utama perancangan UML adalah :

1. Memberikan model yang siap pakai, bahasa pemodelan visual yang ekspresif untuk mengembangkan dan saling menukar model dengan mudah dan dimengerti secara umum.
2. Memberikan bahasa pemodelan yang bebas dari berbagai bahasa pemrograman dan proses rekayasa.
3. Menyatukan praktek-praktek terbaik yang terdapat dalam pemodelan.

Konsep-konsep yang diterapkan di UML adalah satu model berisi informasi mengenai sistem (atau domain), model-model berisi elemen-elemen model seperti kelas-kelas, simpul-simpul, paket-paket, dan sebagainya. Satu diagram menunjukkan satu pandangan tertentu dari model.

2.5.2 Kegunaan Diagram

Kegunaan diagram pada pemodelan adalah untuk formalisasi ekspresi model objek secara koheren, presisi dan mudah dirumuskan. Pemodelan berorientasi objek memerlukan kakas untuk mengekspresikan model. UML (*Unified Modeling Language*) menyediakan sejumlah diagram untuk mengekspresikan pemodelan berorientasi objek yang dilakukan.

UML adalah bahasa untuk menspesifikasikan, memvisualisasikan, dan mendokumentasi artifak-artifak sistem perangkat lunak. UML merupakan sistem notasi (termasuk semantiks untuk notasi itu) yang membantu pemodelan sistem menggunakan konsep berorientasi objek.

2.5.3 Diagram dan Teknik Pemodelan

Diagram mengemukakan banyak hal, penggunaan notasi yang terdefinisi baik dan ekspresif adalah penting pada proses pengembangan perangkat lunak, yaitu :

1. Notasi standar memungkinkan pengembang mendeskripsikan skenario atau rumusan arsitektur dan kmudian mengkomunikasikan secara tidak ambigu.
2. Notasi yang bagus membebaskan otak untuk berkonsentrasi pada masalah-masalah yang lebih lanjut.
3. Notasi yang baik memungkinkan mengeliminasi keperluan pemeriksaan konsistensi dan kebenaran keputusan-keputusan dengan menggunakan tool terotomatisasi.

a. Diagram Struktur

Diagram ini untuk memvisualisasi, menspesifikasikan, membangun dan mendokumentasikan aspek statik dari sistem. Diagram struktur di UML terdiri dari :

1. Diagram kelas (*Class diagram*)
2. Diagram objek (*Object diagram*)
3. Diagram komponen (*Component diagram*)
4. Diagram *deployment* (*Deployment diagram*)

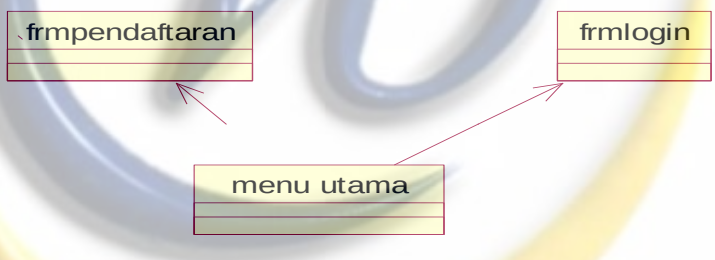
b. Diagram perilaku

Diagram ini untuk memvisualisasi, menspesifikasikan, membangun dan mendokumentasikan aspek dinamis dari sistem. Diagram perilaku di UML terdiri dari :

1. Diagram *use-case* (*Use case diagram*)
2. Diagram sekuen (*Sequence diagram*)
3. Diagram kolaborasi (*Collaboration diagram*)
4. Diagram *statechart* (*Statechart diagram*)
5. Diagram aktivitas (*Activity diagram*)

c. Diagram kelas (*Class diagram*)

Diagram ini menunjukan sekumpulan kelas, *interface* dan kolaborasi dan keterhubungannya. Diagram kelas ditujukan untuk pandangan statistik terhadap sistem. Dapat dilihat pada gambar 2.2 di bawah ini.



Gambar 2.2 *Class Diagram*

d. Diagram objek (*Object Diagram*)

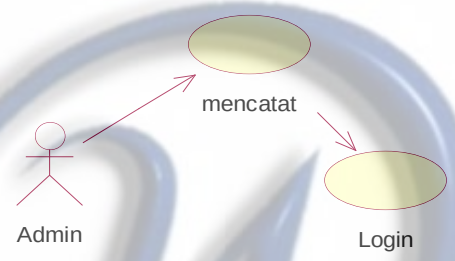
Diagram ini menunjukan sekumpulan objek dan keterhubungannya. Diagram ini menunjukan potongan statik dari instan-instan yang ada di diagram kelas. Diagram ini untuk memperlihatkan satu prototipe atau kasus tertentu yang mungkin terjadi.

Diagram objek menyediakan notasi grafis formal guna memodelkan objek, kelas, dan saling keterhubungan. Diagram objek berguna untuk *abstract modeling* dan perancangan program-program sesungguhnya.

Pada pendekatan ini, bentukan dasar dari sistem perangkat lunak adalah objek atau kelas. Kelas adalah deskripsi dari objek-objek yang *common*. Setiap objek mempunyai indentitas, *state* dan perilaku.

e. Diagram *use-case* (*Use Case Diagram*)

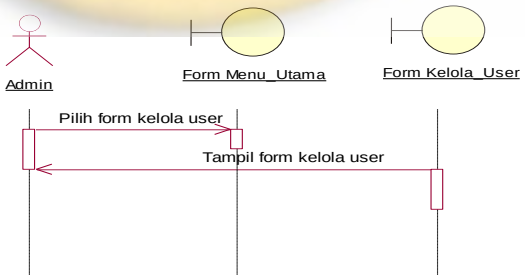
Diagram ini menunjukan sekumpulan kasus fungsional dan aktor (jenis kelas khusus) dan keterhubungannya. Dapat dilihat pada gambar 2.3 di bawah ini.



Gambar 2.3 *Use Case Diagram*

f. Diagram sekuen (*Sequence Diagram*)

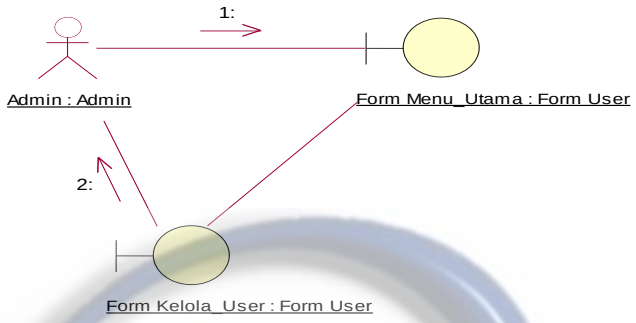
Diagram ini menunjukan interaksi yang terjadi antar objek. Diagram ini merupakan pandangan dinamis terhadap sistem. Diagram ini menekankan pada basis keberurutan waktu dari pesan-pesan yang terjadi. Dapat dilihat pada gambar 2.4 di bawah ini.



Gambar 2.4 *Sequence Diagram*

g. Diagram kolaborasi (*Colaboration Diagram*)

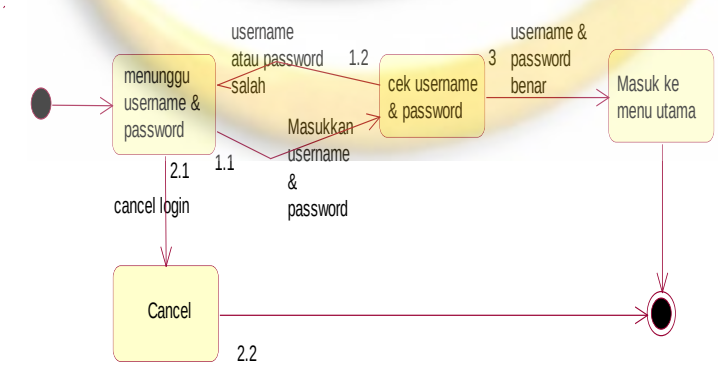
Diagram ini juga merupakan diagram interaksi. Diagram ini menekankan pada organisasi srtuktur dari objek-objek yang mengirim dan menerima pesan. Dapat dilihat pada gambar 2.5 di bawah ini.



Gambar 2.5 *Collaboration Diagram*

h. Diagram statechart (*Statechart Diagram*)

Diagram ini adalah *state-machine diagram,*berisi *state,*transisi, kejadian dan aktivitas. *Statechart* merupakan pandangan dinamis dari sistem. Diagram ini penting dalam memodelkan perilaku antarmuka, kelas, kolaborasi dan menekankan pada urutan kejadian. Penting untuk sistem reaktif yang dipicu kejadian di dunia nyata. Dapat dilihat pada gambar 2.6 di bawah ini.

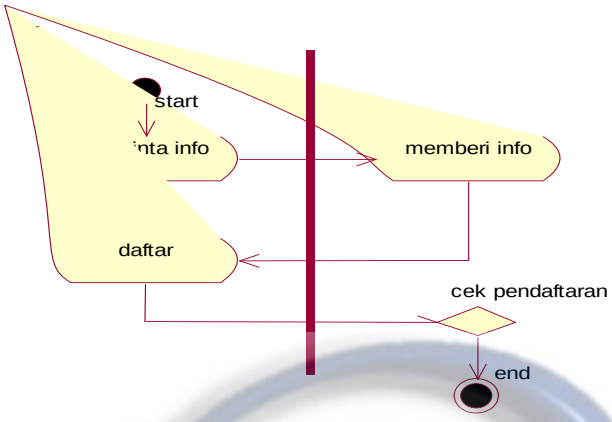


Gambar 2.6 *Statechart Diagram*

i. Diagram aktivitas (*Activity Diagram*)

Diagram ini untuk menunjukan aliran aktivitas di sistem. Diagram ini adalah pandangan dinamis terhadap sistem. Diagram ini penting untuk

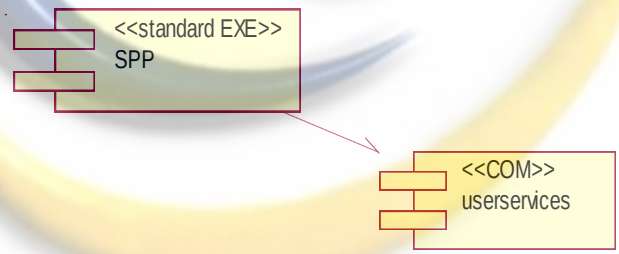
memodelkan fungsi sistem dan menekankan pada aliran kendali diantara objek-objek. Dapat dilihat pada gambar 2.7 di bawah ini.



Gambar 2.7 *Activity Diagram*

j. Diagram komponen (*Component Diagram*)

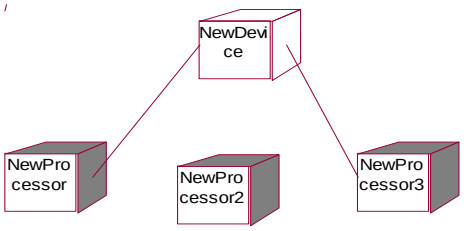
Diagram ini menunjukan organisasi dan kebergantungan diantara sekumpulan komponen. Diagram ini merupakan pandangan statik terhadap implementasi sistem. Dapat dilihat pada gambar 2.8 di bawah ini.



Gambar 2.8 *Component Diagram*

k. Diagram *deployment* (*Deployment Diagram*)

Diagram ini menunjukan konfigurasi pemrosesan saat jalan dan komponen-komponen yang terdapat didalamnya. Diagram ini merupakan pandangan statik dari arsitektur. Dapat dilihat pada gambar 2.9 di bawah ini.



Gambar 2.9 *Deployment Diagram*

Pilihan model dan diagram yang digunakan dipengaruhi oleh bagaimana persoalan ditangani dan bagaimana solusi dibentuk. Abstraksi, fokus pada yang relevan sambil mengabaikan rincian-rincian yang tidak relevan merupakan kuncinya. Karena itu, setiap sistem kompleks perlu didekati melalui sekumpulan pandangan model yang kompleks. Setiap model diekspresikan pada level-level berbeda. Model-model yang terbaik dapat dikoneksikan ke kenyataan.

2.6 Pemrograman dan *Database*

Dalam pembangunan "**Implementasi Metode FIFO (*First in First out*) pada *Inventory System* Berbasis *Web Application***" ini akan dijelaskan secara jelas mengenai bahasa pemrograman PHP dan database MySQL.

2.6.1 Bahasa Pemrograman PHP (*Hypertext Preprocessor*)[4]

PHP adalah sebuah bahasa pemograman yang berjalan dalam sebuah web-server (*server side*). PHP diciptakan oleh programmer unix dan Perl yang bernama Rasmus Lerdoft pada bulan Agustus-September 1994. Pada awalnya, Rasmus mencoba menciptakan sebuah script dalam website pribadinya dengan tujuan untuk memonitor siapa saja yang pernah mengunjungi website-nya.

Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (Situs *personal*). Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI pada sekitar tahun 1995, dan diperkenalkan kepada beberapa programmer pemula dengan alasan bahasa yang digunakan oleh PHP cukup sederhana dan mudah dipahami. Selanjutnya Rasmus menulis ulang PHP dengan bahsa C untuk meningkatkan kecepatan aksesnya.

Mulai bulan September sampai Oktober 1995, kode PHP ditulis ulang dan digabungkan menjadi PHP/F1. Baru di akhir tahun 1995 dirilis bagi umum secara gratis. Mengapa Rasmus membagikan ke publik secara gratis ? Rasmus berangapan apabila kode PHP ini berguna bagi dirinya, tentu juga akan bermanfaat untuk oranglain. Toh pada akhirnya akan kembali bermanfaat bagi dirinya sendiri.

Pada November 1997, dirilis PHP/FI 2.0. Pada rilis ini, interpreter PHP sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan.

Pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang interpreter PHP menjadi lebih bersih, lebih baik, dan lebih cepat. Kemudian pada Juni 1998, perusahaan tersebut merilis interpreter baru untuk PHP dan meresmikan rilis tersebut sebagai PHP 3.0 dan singkatan PHP dirubah menjadi akronim berulang dan singkatannya menjadi PHP: *Hypertext Preprocessing*.

Pada pertengahan tahun 1999, Zend merilis interpreter PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi dibalik ke kompleksan aplikasi web tersebut.

Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari interpreter PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek.

Beberapa keistimewaan atau kelebihan yang dimiliki PHP dari bahasa pemrograman web, antara lain:

1. Bahasa pemrograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaanya.
2. *Web Server* yang mendukung PHP dapat ditemukan dimana - mana dari mulai apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.

3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis - milis dan *developer* yang siap membantu dalam pengembangan.
4. Dalam sisi pemahamanan, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
5. PHP adalah bahasa *open source* yang dapat digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah *system*.

2.6.2 MySQL[5]

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (*database management system*) atau DBMS yang *multithread, multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratisdibawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL.

Tidak sama dengan proyek-proyek seperti Apache, dimana perangkat lunak dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia MySQL AB, dimana memegang hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan MySQL AB adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius.

Beberapa keistimewaan atau kelebihan yang dimiliki database MySQL, antara lain :

1. Portabilitas. MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X Server, Solaris, Amiga, dan masih banyak lagi.
2. Perangkat lunak sumber terbuka. MySQL didistribusikan sebagai perangkat lunak sumber terbuka, dibawah lisensi GPL sehingga dapat digunakan secara gratis.

3. *Multi-user*. MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
4. *Performance tuning*, MySQL memiliki kecepatan yang menakjubkan dalam menangani query sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.
5. Ragam tipe data. MySQL memiliki ragam tipe data yang sangat kaya, seperti *signed*/*unsigned integer*, *float*, *double*, *char*, *text*, *date*, *timestamp*, dan lain-lain.
6. Perintah dan Fungsi. MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah Select dan Where dalam perintah (*query*).
7. Keamanan. MySQL memiliki beberapa lapisan keamanan seperti level subnetmask, nama host, dan izin akses *user* dengan sistem perizinan yang mendetail serta sandi terenkripsi.
8. Skalabilitas dan Pembatasan. MySQL mampu menangani basis data dalam skala besar, dengan jumlah rekaman (*records*) lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.
9. Konektivitas. MySQL dapat melakukan koneksi dengan klien menggunakan beberapa protokol seperti TCP/IP, Unix soket (UNIX), atau Named Pipes (NT).
10. Lokalisasi. MySQL dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meski pun demikian, bahasa Indonesia belum termasuk di dalamnya.
11. Antar muka. MySQL memiliki antar muka (*interface*) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (*Application Programming Interface*).
12. Klien dan Peralatan. MySQL dilengkapi dengan berbagai peralatan (*tool*)yang dapat digunakan untuk administrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk *online*.

13. Struktur tabel. MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan basis data lainnya semacam PostgreSQL ataupun Oracle.

2.6.3 *Web Application* [6]

Dalam rekayasa perangkat lunak, suatu aplikasi web (bahasa Inggris: *web application* atau sering disingkat *webapp*) adalah suatu aplikasi yang diakses menggunakan *browser* melalui suatu jaringan seperti Internet atau intranet.Ia juga merupakan suatu aplikasi perangkat lunak komputer yang dikodekan dalam bahasa yang mendukng *browser* (seperti HTML, JavaScript, AJAX, Java, dan lain-lain) dan bergantung pada penjelajah tersebut untuk menampilkan aplikasi.

Aplikasi web menjadi populer karena kemudahan tersedianya aplikasi klien untuk mengaksesnya, penjelajah web, yang kadang disebut sebagai suatu *thin client* (klien tipis). Kemampuan untuk memperbarui dan memelihara aplikasi web tanpa harus mendistribusikan dan menginstalasi perangkat lunak pada kemungkinan ribuan komputer klien merupakan alasan kunci popularitasnya.