BAB II
LANDASAN TEORI

2.1 Konsep Sistem Informasi

2.1.1 Konsep Dasar Sistem

Sistem kebanyakan dapat didefinisikan secara sederhana sebagai sekelompok elemen yang saling berhubungan atau berinteraksi hingga membentuk satu kesatuan. Akan tetapi, konsep umum sistem berikut ini memberikan konsep dasar yang lebih tepat untuk bidang Sistem Informasi.

Sistem adalah sekelompok komponen yang saling berhubungan, bekerja bersama untuk mencapai tujuan bersama dengan menerima input serta menghasilkan output dalam proses transformasi yang teratur. (O'brien 2006:29)

Suatu sistem mempunyai karakteristik atau sifat-sifat tertentu, antara lain sebagai berikut :

a. Komponen Sistem (*Component*)

   Suatu sistem terdiri dari sejumlah komponen atau elemen yang saling berinteraksi, artinya komponen atau elemen yang saling bekerja sama dalam bentuk satu kesatuan. Komponen atau elemen sistem dapat berupa subsistem atau bagian dari sistem. Setiap subsistem mempunyai sifat-sifat dari sistem. Untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan.

b. Batas Sistem (*Boundary*)

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

c. Lingkungan luar (*environtments*)

   Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi dari sistem.

d. Penghubung (*interface*)

Penghubung sistem merupakan media penghubung antara satu subsistem dengan subsistem yang lain untuk dapat berinteraksi membentuk suatu kesatuan.

e. Masukan (*input*)

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

f. Pengolahan (*process*)

Suatu sistem dapat mempunyai suatu bagian pengolahan yang akan merubah masukan menjadi keluaran.

g. Keluaran (*output*)

Keluaran sistem adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

h. Sasaran (*objective*)

Suatu sistem harus mempunyai sasaran, karena sasaran sangat menentukan sekali masukan yang dibutuhkan oleh sistem dan keluaran yang akan dihasilkan sistem. Suatu sistem dikatakan berhasil apabila mengenai sasaran atau tujuan.

2.1.2 Konsep Dasar Informasi

Informasi adalah hasil dari pengolahan data dalam bentuk yang lebih berguna dan berarti bagi penerimanya yang menggambarkan suatu kejadian-kejadian yang nyata yang digunakan untuk pengambilan keputusan.

Sumber dari informasi adalah data. Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan nyata. Kejadian-kejadian (*event*) adalah suatu yang terjadi pada saat tertentu. Kesatuan nyata (*fact*) adalah berupa suatu obyek nyata seperti tempat, benda atau orang yang benar-benar ada dan terjadi.

Menurut Jogiyanto (2003:36) :

"Informasi (information) adalah data yang diolah menjadi bentuk yang berguna bagi para pemakainya. Data yang diolah saja tidak cukup dapat dikatakan sebagai suatu informasi. Untuk menjadi suatu informasi, maka data yang diolah tersebut harus berguna bagi pemakainya."

Untuk dapat berguna, maka informasi harus didukung oleh tiga pilar, yaitu sebagai berikut :

a. Akurat (*accurate*)

Akurat berarti informasi harus bebas dari kesalahan-kesalahan dan tidak menyesatkan. Akurat juga berarti informasi harus jelas mencerminkan maksudnya. Informasi harus akurat karena dari sumber informasi sampai ke penerima informasi kemungkinan banyak terjadi gangguan (*noise*) yang dapat merubah atau merusak informasi tersebut.

b. Tepat waktu (*Timeliness*)

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 didalam pengambilan keputusan.

c. Relevan (*Relevance*)

Relevan berarti informasi tersebut mempunyai manfaat untuk pemakianya. Relevansi informasi untuk tiap-tiap orang berbeda. Nilai informasi bagi seorang pemakai ditentukan oleh keandalan (*reliabilitas*). Keluaran yang tidak didukung oleh ketiga pilar ini tidak dapat dikatakan sebagai informasi yang berguna, tetapi merupakan sampah (*garbage*).

2.2 Analisa dan Perancangan Sistem Berorientasi Obyek dengan UML

Analisa sistem dapat dinyatakan sebagai pemisahan suatu hal dalam bagian bagian tertentu. Bagian-bagian tersebut kemudian dipelajari dan dievakuasi untuk mengetahui apakah terdapat cara-cara yang lebih baik untuk memenuhi kebutuhan manajemen.

"Analisa sistem adalah proses menentukan kebutuhan sistem – apa yang harus dilakukan sistem untuk memenuhi kebutuhan klien, bukanlah bagaimana sistem tersebut diimplementasikan." (Ariesto Hadi Sutopo, 2002:242):

Konsep dasar berorientasi obyek mencapai kematangannya pada saat masalah analisis dan desain menjadi lebih diperhatikan dari pada masalah coding. Secara spesifik, pengertian "berorientasi obyek" (Ariesto Hadi Sutopo, 2002:3) berarti bahwa "kita mengorganisasi perangkat lunak sebagai kumpulan dari objek tertentu yang memiliki struktur data dan perilakunya".

2.2.1 Unified Modelling Language (UML)

Unified Modelling Language (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi danjaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi obyek. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya : Grady Booch OOD (*Object-Oriented Design*), Jim Rumbaugh OMT (*Object Modelling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*).

Cakupan UML diantaranya : Pertama, UML menggabungkan konsep BOOCH, OMT, dan OOSE, sehingga UML merupakan suatu bahasa permodelan tunggal yang umum dan digunakan secara luas oleh para user ketiga metode tersebut dan bahkan para user metode lainnya. Kedua, UML menekankan pada apa yang dapat dikerjakan dengan metode-metode tersebut. Ketiga, UML berfokus pada suatu bahasa permodelan standar, bukan pada proses standar.

2.2.2 Analisa Sistem Berorientasi Obyek

a. Activity Diagram

Diagram memodelkan alur kerja (*work floaw*) sebuah proses bisnis dan urutan aktivitas pada suatu proses. Diagram ini sangat mirip dengan *flow chart* karena kita dapat memodelkan prosedur logika, proses bisnis

dan alur kerja. Perbedaan utamanya adalah *flow chart* dibuat untuk menggambarkan alur kerja dari sebuah sistem, sedangkan *activity diagram* dibuat untuk menggambarkan aktivitas dari aktor.

Activity diagram adalah teknik untuk mendiskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity diagram* mempunyai persan seperti halnya *flow chart*, akan tetapi perbedanya dengan *flow chart* adalah *activity diagram* bisa mendukung perilaku paralel sedangkan *flow chart* tidak bisa.

Simbol – simbol yang digunakan pada saat pembuatan *activity diagram* adalah sebagai berikut :

a. *Start Point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.(Munawar 2005:109)

Start Point

b. *End Point*, akhir aktifitas.(Munawar 2005:109)

End Point

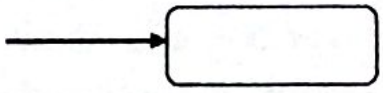
c. *Activity*, menggambarkan suatu proses / kegiatan bisnis.(Munawar 2005:109)

Activity

Jenis – jenis *Activities*, yaitu :

a. *Black Hole Activities*

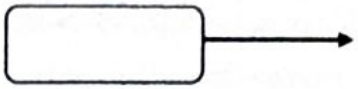
Ada masukan dan tidak ada keluaran, biasanya digunakan jika dikehendaki ada satu atau lebih transisi.



Simbol *Black Hole Activities*

b. *Miracle Activities*

Tidak ada masukan dan ada keluaran, biasanya dipakai pada waktu start point dan dikehendaki ada 1 atau lebih transisi.



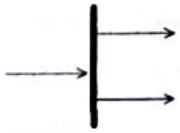
Simbol *Miracle Activities*

c. *Paralel Activities*

Suatu *activity* yang berjalan secara bersamaan terdiri dari :

1. *Fork* (Percabangan)

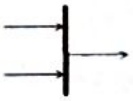
   Fork digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu.(Munawar 2005:110)



Simbol *Fork* (Percabangan)

2. *Join* (Penggabungan))

   Join (penggabungan) / *Rake* ,menunjukkan adanya dekomposisi.(Munawar 2005:110) .Yaitu mempunyai 2 atau lebih transisi masuk dan hanya 1 transisi keluar, dan *fork* harus berhubungan dengan *join.*



Simbol *Join* (Penggabungan)

3. *Decision Point*

   Decision digambarkan dengan lambing wajik atau belah ketupat. Mempunyai transisi (sebuah garis dari atau ke *decision point).* Setiap transisi yang ada harus mempunyai *guard* (kunci). Tidak ada keterangan (pernyataan) pada tengah belah ketupat seperti pada *flowchart.*

Simbol *Decision Point*

4. *Guard* (Kunci)

   Guard (kunci) adalah kondisi benar sewaktu melewati sebuah transisi. Digambarkan dengan diletakkan diantara tanda [ ]. Tanda (*otherwise*) guard untuk menangkap suatu kondisi yang belum terdeteksi. Setiap transisi dari atau ke decision point harus mempunyai *guard* yang harus konsisten dan lengkap serta tidak *overlap*.

5. *Swimlane*

   Swimlane merupakan sebuah cara untuk mengelompokkan activity berdasarkan *actor*. *Actor* bisa ditulis nama *actor* ataupun sekaligus dalam lambang actor. *Swimlane* digambarkan secara *vertical*, walaupun kadang-kadang digambarkan secara *horizontal*.

6. *Swimarea*

   Ketika sebuah *activity diagram* mempunyai banyak *swimlane*, perlu dipikirkan dengan pendekatan *swimarea*. *Swimarea* mengelompokkan *activity* berdasarkan kegiatan didalam *use case*.

b. Analisa Dokumen Keluaran

Analisa keluaran adalah analisa mengenai dokumen – dokumen keluaran yang dihasilkan dari sebuah sistem.

c. Analisa Dokumen Masukan

Analisa masukkan adalah bagian dari pengumpulan informasi tentang system yang sedang berjalan. Tujuan analisa masukkan adalah memahami prosedur berjalan.

d. *Use Case Diagram*

Use Case Diagram menggambarkan sebuah fungsionalitas yang diharapkan dari sebuah sistem dan bagaimana sistem berinteraksi dengan dunia luar. Yang ditekankan dalam *use case diagram* adalah "apa" yang diperbuat sistem, dan bukan "bagaimana" sistem itu melakukannya. Sebuah *use case* merepresentasikan sebuah interaksi antara actor dengan sistem. *Use Case Diagram* juga menjelaskan manfaat sistem jika dilihat menurut pandangan orang yang berada diluar sistem (*actor*). *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-create sebuah daftar belanja, dan sebagainya.

Secara umum use case diagram terdiri dari :

a. Aktor (*Actor*)

Actor adalah sebuah peran yang bisa dimainkan oleh pengguna dalam interaksinya dengan sistem. Untuk mengidentifikasikan actor harus ditentukan pembagian kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. *Actor* dilukiskan dengan peran yang mereka mainkan dalam *use case*, seperti staff penjualan, pelanggan, dll.

Simbol *Actor*

b. Use case

Use case menggambarkan perilaku, termasuk didalamnya interaksi antara actor dengan sistem. *Use case* dibuat berdasarkan keperluan *actor*, merupakan "apa" yang dikerjakan sistem bukan "bagaimana" sistem mengerjakannya. Setiap *use case* harus diberi nama yang menyatakan apa hal yang dicapai dari hasil interaksinya dengan *actor*. Nama *use case* boleh terdiri dari beberapa kata dan tidak boleh ada dua *use case* yang memiliki nama yang sama.

Simbol *Use Case*

c. Asosiasi (*Asociation*)

Asosiasi menggambarkan aliran data / informasi. Asosiasi / relasi juga digunakan untuk menggambarkan bagaimana *actor* terlibat dalam *use case*. Relasi (*relationship*) digambarkan sebagai bentuk garis antara dua simbol dalam *use case diagram*.

Simbol *Asosiation*

Ada empat jenis relasi / asosiasi yang dapat timbul pada *use case diagram*, yaitu :

1. Asosiasi antara Actor dan Use Case

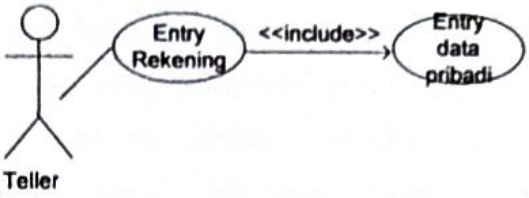
   Ujung panah pada *association* antara *actor* dan *use case* mengindikasikan siapa / apa yang meminta interaksi dan bukannya mengindikasikan aliran data. Sebaiknya gunakan garis tanpa panah untuk *association* antara *actor* dan *use case*. *Association* antar *actor* dan *use case* yang menggunakan panah terbuka untuk mengindikasikan bila *actor* berinteraksi secara pasif dengan sistem.

Simbol *Asosiation* antara *Actor* dan *Use Case*

2. Asosiasi antara *Use Case*

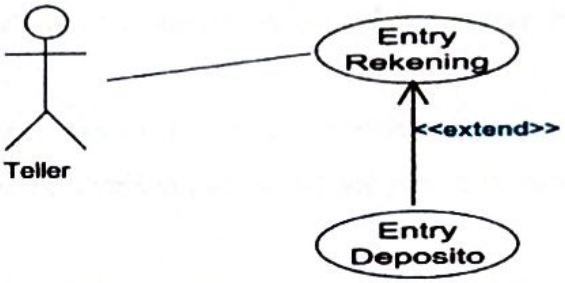
   Relasi antara use case dengan *use case* :

   a. *Include*, menggambarkan suatu *use case* termasuk di dalam *use case* lain (diharuskan). Contohnya adalah pemanggilan sebuah fungsi program. Digambarkan dengan garis lurus berpanah dengan tulisan <<include>>.



Contoh *Include*

b. *Extend*, digunakan ketika hendak menggambarkan variasi pada kondisi perilaku normal dan menggunakan lebih banyak *control form* dan mendeklarasikan ekstension pada *use case* utama atau dengan kata lain adalah perluasan dari *use case* lain jika syarat atau kondisi terpenuhi. Digambarkan dengan garis lurus berpanah dengan tulisan <<extend>>.



Contoh *Extend*

c. *Generalization / Inheritance* antar *Use Case Generalization* dipakai ketika ada sebuah perlakuan khusus (*single condition*) dan merupakan pola hubungan *base-parent use case*. Digambarkan dengan *generalization / inheritance* antar use case secara vertikal dengan *inheriting use case* dibawah *base / parent use case*
d. *Generalization / Inheritance* antar *Actors* Digambarkan *generalization* antar *actors* secara vertikal dengan *inheriting actor* dibawah *base / parent use case*.

c. *Deskripsi Use Case Diagram*

Bagian terbesar dari *use case* merupakan deskripsi naratif dari urutan utama *use case* yang merupakan urutan yang paling umum dari interaksi antara aktor dan sistem. Deskripsi tersebut dalam bentuk input dari aktor, diikuti oleh respon pada sistem. Sistem ditandai dengan sebuah kotak hitam (*black box*) yang berkaitan dengan apa yang sistem lakukan dalam merespon input aktor, bukan bagaimana internal melakukannya.

2.2.3 Perancangan Sistem Berorientasi Obyek

Perancangan berorientasi obyek merupakan tahap lanjutan setelah analisa berorientasi obyek, perancangan berorientasi obyek adalah suatu pendekatan yang digunakan untuk menspesifikasi kebutuhan – kebutuhan sistem dengan mengkolaborasikan obyek–obyek, atribut–atribut, dan *method–method* yang ada.(Whitten 2004:686).

Tujuan perancangan sistem itu untuk memahami kebutuhan kepada pemakai sistem (*user*) dan memberikan gambaran yang jelas serta rancang bangun yang lengkap.

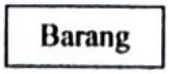
Tahap-tahap yang dilakukan pada perancangan berorientasi obyek adalah:

a. *Entity Relationship Diagram* (ERD)

ERD adalah sebuah model data yang menggunakan beberapa notasi untuk menggambarkan data dalam hal entitas dan relasi yang digambarkan oleh data tersebut.(Whitten 2004:295). Elemen-elemen ERD yaitu sebagai berikut :

a. *Entity* (Entitas)

Sesuatu (obyek) yang ada didalam sistem. Entitas merupakan kata benda yang dikelompokkan menjadi empat jenis nama, yaitu : orang, benda, lokasi dan kejadian. Entitas disimbolkan dengan persegi pajang.

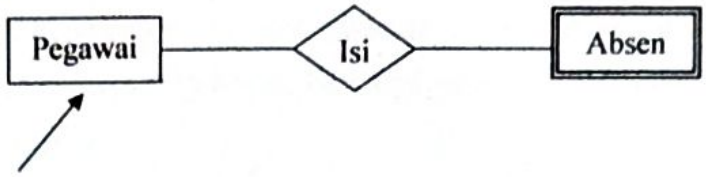


Simbol Entitas

Jenis-jenis Entitas

a. *Strong Entity Set*

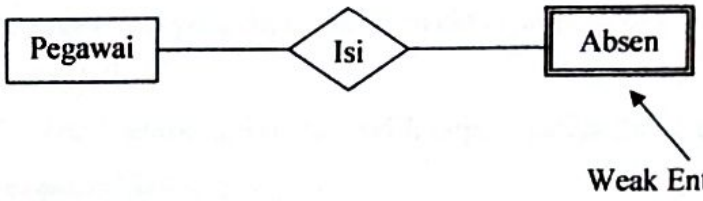
Strong entity set yaitu *entity set* satu atau banyak atribut nya digunakan oleh entitas lain.



Strong Entity

b. *Weak Entity Set*

Weak entity set yaitu *entity set* yang tidak memiliki atribut yang dapat dijadikan kunci, sehingga membutuhkan atribut dari entitas lain. Dengan kata lain entitas yang bergantung pada entitas lain (*strong entity*).



b. *Relationship* (Hubungan atau relasi)

Sebuah asosiasi bisnis alami antara satu atau lebih entitas. Sebuah relasi bisa menunjukkan sebuah peristiwa yang menghubungkan sebuah entitas ke entitas yang lain.(Whitten 2004:298) Simbol *Relationship* pada ERD digambarkan dengan *diamond* atau *decision*. Jika satu entitas dihubungkan dengan *Relationship*, maka digambarkan dengan garis lurus. Kumpulan dari *Irelationship* yang sejenis disebut *Relationship Set*.

Simbol Relasi

c. *Atribute* (Atribut)

Suatu deskripsi karakteristik dari entitas.(Whitten 2004:296). Atribut juga merupakan karakteristik dari *relationship*, maksudnya sesuatu yang menjelaskan apa sebenarnya yang dimaksud dengan entitas maupun *relationship*. Atribut disimbolkan dengan sebuah elips. Dari setiap atribut entitas terdapat satu atribut yang dijadikan sebuah *key* (kunci). Beberapa jenis *key*, yaitu :

1. *Primary Key*

   Field yang mengidentifikasikan sebuah *record* dalam *file* dan bersifat unik.

2. *Secondary Key*

   Field yang mengidentifikasikan sebuah *record* dalam *file* dan tidak bersifat unik.

3. *Candidate Key*

   Beberapa *field* yang dapat dijadikan calon *primary key*.

4. *Alternate Key*

   Field dari *candidate key* yang tidak terpilih jadi *primary key*.

5. *Composite Key*

   Beberapa *field* yang digabungkan untuk membentuk *primary key*.

6. *Foreign Key*

   Field yang bukan *key*, tetapi merupakan *key* pada *file* lain.

d. *Cardinality* (Kardinaliti)

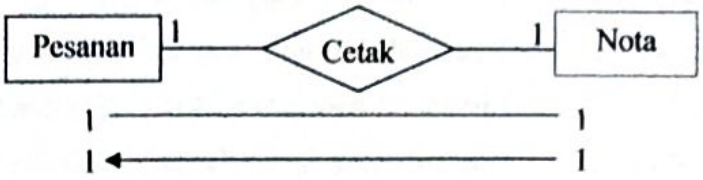
Jumlah kejadian minimum dan maksimum dari satu entitas yang dihubungkan dengan kejadian yang tunggal dari entitas lain.(Whitten 2004:299)

Ada 3 (tiga) kemungkinan hubungan yang ada yaitu :

1. *One To One* (1 : 1)

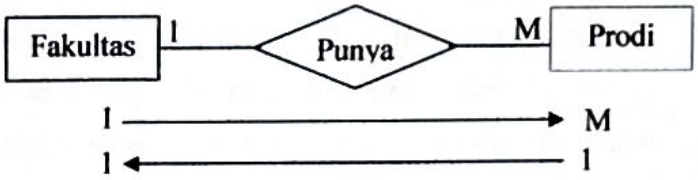
   Jumlah kejadian adalah satu ke satu antara entitas yang saling berhubungan.(Whitten 2004:299) Artinya tingkat hubungan dimana satu kejadian pada entitas yang pertama hanya

mempunyai satu hubungan dengan satu kejadian pada entitas kedua, demikian juga sebaliknya.



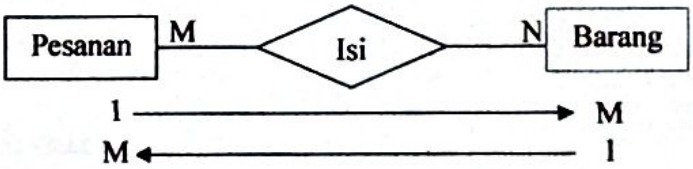
2. *One To Many* (1 : M)

Jumlah kejadian adalah satu ke banyak dari satu entitas ke entitas lain yang berhubungan.(Whitten 2004:299) Artinya tingkat hubungan dimana satu kejadian pada entitas yang pertama mempunyai banyak hubungan dengan kejadian pada entitas kedua, demikian juga sebaliknya.



3. *Many To Many* (M : N)

Jumlah kejadian adalah banyak ke banyak dari satu entitas ke entitas lain yang berhubungan.(Whitten 2004:299) Artinya tingkat hubungan dimana tiap kejadian pada sebuah entitas akan mempunyai banyak hubungan dengan kejadian pada entitas lainnya.



b. *Logical Record Structure* (LRS)

Sebuah model sistem yang digambarkan dengan sebuah Diagram-ER akan mengikuti pola/aturan pemodelan tertentu. Dalam kaitannya dengan konversi ke LRS, maka perubahan yang terjadi adalah mengikuti aturan-aturan berikut ini:

a. Setiap entitas akan diubah ke bentuk kotak.
b. Sebuah atribut relasi disatukan dalam sebuah kotak bersama entitas jika hubungan yang terjadi pada diagram-ER 1:M (relasi bersatu dengan *cardinality* M) atau tingkat hubungan 1:1 (relasi bersatu dengan *cardinality* yang paling membutuhkan referensi), sebuah relasi dipisah dalam sebuah kotak tersendiri (menjadi entitas baru) jika tingkat hubungannya M:M (*many to many*) dan memiliki *foreign key* sebagai *primary key* yang diambil dari kedua entitas yang sebelumnya saling berhubungan.

c. Tabel/Relasi

Tabel adalah koleksi objek yang terdiri dari sekumpulan elemen yang diorganisasi secara kontinyu, artinya memori yang dialokasi antara satu elmen dengan elmen yang lainnya mempunyai *address* yang berurutan. Pada tabel, pengertian perlu dipahami adalah:

a. Keseluruhan tabel (sebagai koleksi) adalah kontainer yang menampung seluruh elemen.
b. Indek tabel, yang menujukan *address* dari sebuah elemen.
c. *Elment* tabel, yang dapat dipacu melalui indeknya, bertipe tertentu yang sudah terdefinisi
d. Seluruh elemen tabel bertipe:"sama". Dengan catatan: beberapa bahasa pemograman memungkinkan pendefinisian tabel dengan elmen generik, tapi pada saat diinstansiasi, harus diinstanisasi dengan tipe sama.

d. Spesifikasi Basis Data

Basis data merupakan kumpulan dari data yang saling berhubungan satu dengan yang lain dan tersimpan diluar komputer serta digunakan perangkat lunak (*software*) tertentu untuk memanipulasinya.

Sedangkan sistem berbasis data adalah suatu sistem penyusunan dan pengelolaan *record-record* dengan menggunakan komputer dengan tujuan

untuk menyimpan atau merekam serta melihat data operasional lengkap pada sebuah organisasi, sehingga mampu menyediakan informasi yang diperlukan untuk kepentingan proses pengambilan keputusan.

e. Rancangan Dokumen Keluaran

Rancangan keluaran merupakan informasi yang akan dihasilkan dari keluaran sistem yang dirancang.

f. Rancangan Dokumen Masukan

Rancangan masukan merupakan data yang dibutuhkan untuk menjadi masukan sistem yang dirancang.

g. Rancangan Layar Program

Rancangan tampilan merupakan bentuk tampilan sistem layar komputer sebagai antar muka dengan pemakai yang akan dihasilkan dari sistem yang dirancang.

h. *Sequence Diagram*

Sequence diagram adalah suatu diagram UML yang memodelkan logika dari suatu *use case* dengan menggambarkan interaksi berupa pengiriman pesan *(message)* antar obyek dalam urutan waktu.(Whitten 2004:702)

Beberapa simbol yang umum digunakan pada sequence diagram, yaitu:

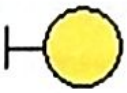
a. *Actor*, menggambarkan orang yang sedang berintraksi dengan sistem

Simbol *Actor*

b. *Entity Object*, suatu obyek yang berisi informasi kegiatan yang terkait yang tetap dan disimpan ke dalam suatu *database*.(Whitten 2004:686)

Simbol *Entity Object*

c. *Interface/Boundary Object*, sebuah obyek yang menjadi penghubung antara user dengan sistem. Contohnya *window*, *dialogue box* atau *screen* (tampilan layar).(Whitten 2004:686)



Simbol *Boundary Object*

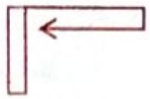
d. *Control Object*, suatu obyek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas. contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai obyek. *Control object* mengkoordinir pesan (*message*) antara *boundary* dengan entitas.(Whitten 2004:686)

Simbol *Control*

e. *Simple Message*, simbol pengiriman pesan dari sebuah obyek ke obyek lain.(Whitten 2004:704)

Simbol *Message*

f. *Recursive*, sebuah obyek yang mempunyai sebuah *operation* kepada dirinya sendiri.(Munawar 2005:89)



Simbol *Recursive*

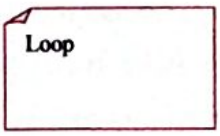
g. *Activation*, mewakili sebuah eksekusi operasi dari obyek, panjang kotak ini berbanding lurus dengan durasi aktivasi sebuah operasi.(Munawar 2005:87;89)

Simbol *Activation*

h. *Lifeline*, garis titik-titik yang terhubung dengan obyek, sepanjang *lifeline* terdapat *activation*. (Munawar 2005:87;89)

Simbol *Lifeline*

i. *Loop*, menggambarkan suatu kegiatan yang dilakukan secara berulang-ulang.



Simbol *Loop*

i. ***Class Diagram (Entity Class)***

Clas diagram sangat membantu dalam visualisasi struktur kelas dari suatu sistem. Hal ini disebabkan karena class adalah diskrifsi kelompok obyek-obyek dengan properti , perilaku (operasi) dan relasi yang sama. Disamping itu *class diagram* bisa memberikan pandangan global atas sebuah sistem. Hal tersebut tercermin dari class-class yang ada yang

relasinya satu dengan yang lainnya. Itulah sebabnya *class diagram* menjadi diagram paling populer di UML.

Diagram kelas memperlihatkan aturan dan tanggung jawab entitas yang menentukan perilaku sistem. Diagram ini berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur yang dibuat. Diagram ini merupakan fondasi untuk component diagram dan *deployment diagram*. Dalam notasi UML *class* digambarkan dengan kotak. Nama class menggunakan huruf besar diawal kalimatnya dan diletakkan diatas kotak.

a. Asosiasi (*Asociation*)

Association/asosiasi adalah kelas-kelas yang terhubungkan satu sama lain secara konseptual. Setiap association mempunyai dua *association* end. Sebuah association end juga memiliki " *multiplicity* ". *Multiplicity* menunjukkan beberapa banyak obyek yang berpartisipasi dalam suatu relasi. Secara umum, *multiplicity* menunjukkan batasan terendah dan tertinggi untuk obyek-obyek yang berpartisipasi. *Multiplicity* yang paling umum digunakan adalah 1, *, dan 0..1.

Langkah-langkah transformasi dari conceptual data model ke tabel relasi adalah sebagai berikut :

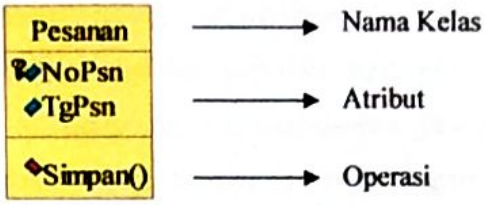
1. Jika hubungan yang terjadi antar *class* adalah 1 to 1 (*one to one*) maka atribut dari *relationship set* diambil dan dimasukkan ke entitas yang lebih membutuhkan.
2. Jika hubungan yang terjadi antar *class* adalah 1 to 0..1 (*one to zero one*) maka atribut dari *relationship set* digabung ke entitas yang memiliki *multiplicity* 0..1.
3. Jika hubungan yang terjadi antar *class* adalah 1 to * (*one to many*) maka atribut dari *relationship set* digabung dengan set entitas yang memiliki *multiplicity* banyak (*many*).

b. Atribut (*Atribute*)

Attribute adalah properti dari sebuah *class*. *Attribute* ini melukiskan batas nilai yang mungkin ada pada obyek dari *class*.

c. Operasi

Operasi adalah sesuatu yang bisa dilakukan oleh sebuah *class* atau yang ada (atau *class* lain) dapat dilakukan untuk sebuah *class*.



Class Diagram

2.3 Konsep Manajemen Proyek

2.3.1 Definisi Proyek

Proyek adalah upaya temporer untuk menghasilkan produk, jasa, atau hasil yang tertentu/unik.

Proyek bersifat temporer → artinya waktu berlangsungnya dibatasi, ada awal dan ada akhir untuk pekerjaan yang dilakukan dan tim yang dibentuk.

Proyek menghasilkan yang unik → berarti hasil dari proyek merupakan suatu entitas baru yang memiliki karakteristik yang berbesda dengan hasil yang sudah ada.

2.3.2 Definisi Manajemen Proyek

Manajemen proyek adalah salah satu cara yang ditawarkan untuk maksud pengelolaan suatu proyek, yaitu suatu metode pengelolaan yang dikembangkan secara ilmiah dan intensif sejak pertengahan abad ke-20 untuk menghadapi kegiatan khusus yang berbentuk proyek. **(Iman Soeharto, 1999)**

Definisi Manajemen Proyek menurut PMBOK (Project Management Body of Knowledge) adalah aplikasi dari pengetahuan, keahlian, alat-alat, dan teknik untuk melaksanakan aktivitas sesuai dengan kebutuhan proyek.

"Project managements is the application of knowledge, skills, tools, and techniques to project activities to meet project requirements."

2.4 Stakeholder

Stakeholder merupakan individu, sekelompok manusia, komunitas atau masyarakat baik secara keseluruhan maupun secara parsial yang memiliki hubungan serta kepentingan terhadap perusahaan. Individu, kelompok, maupun komunitas dan masyarakat dapat dikatakan sebagai *stakeholder* jika memiliki karakteristik yaitu mempunyai kekuasaan, legitimasi, dan kepentingan terhadap perusahaan. Di dalam pengembangan Sistem Informasi stakeholder dapat dibedakan menjadi :

2.4.1 Manager Sistem Informasi

Manager dalam departemen Sistem informasi memiliki peranan secara langsung dalam proses pengembangan sistem jika organisasi yang ditanganinya berskala kecil.

Manager SI berperan dalam mengalokasikan dan mengawasi proyek pengembangan sistem daripada terlibat langsung dalam proses pengembangan sistem.

2.4.2 Analyst Sistem

Sistem analis merupakan individu kunci dalam proses pengembangan sistem. Sistem analis mempelajari masalah dan kebutuhan dari organisasi untuk menentukan bagaimana orang, data, proses, komunikasi dan teknologi informasi dapat meningkatkan pencapaian bisnis. Seorang sistem analis juga merupakan orang yang paling bertanggung jawab pada proses analisa dan perancangan sistem informasi.

Menurut Yogiyanto (1995) analis sistem (analis informasi) adalah orang yang menganalis sistem (mempelajari masalah-masalahan yang timbul dan

menentukan kebutuhan pemakai sistem) untuk mengidentifikasikan pemecahan permasalahan tersebut.

Menurut Kristanto (2003) analis sistem adalah orang yang mempunyai kemampuan untuk menganalisis sebuah sistem, memilih alternatif pemecahan masalah dan menyelesaikan masalah tersebut dengan menggunakan komputer.

Analis sistem secara sistematis menilai bagaimana fungsi bisnis dengan cara mengamati proses input dan pengolahan data serta proses output informasi untuk membantu peningkatan proses organisasional. Dengan demikian, analis sistem mempunyai tiga peranan penting, yaitu :

a. Sebagai konsultan
b. Sebagai ahli pendukung
c. Sebagai agen perubahan

2.4.3 Programmer

Programmer adalah orang yang menulis kode program untuk suatu aplikasi tertentu berdasarkan rancang bangun yang telah dibuat oleh analis sistem. Programmer lebih memahami tentang teknologi komputer tetapi kurang memahami tentang aspekaspek bisnis dan tentang kebutuhan-kebutuhan yang diperlukan oleh pemakai sistem.

2.4.4 Desainer Sistem

Adalah spesialis teknis yang menterjemahkan persyaratan bisnis pengguna sistem dan pembatas solusi teknis. Dia mendesain data base, input, output, tampilan, jaringan dan perangkat lunak komputer yang akan memenuhi persyaratan pengguna sistem.

2.4.5 Pengguna Sistem

Adalah "pelanggan" yang akan menggunakan atau terpengaruh sistem informasi pada basis reguler, meng-capture, memvalidasikan, memasukkan, menanggapi, menyimpan, dan bertukar data dan informasi.

Kelas-kelas pengguna sistem dibagi dalam 2 bagian besar yaitu pengguna sistem internal dan pengguna sistem eksternal. Pengguna sistem internal adalah karyawan-karyawan bisnis yang kebanyakan sistem informasi dibangun untuk mereka.

Pengguna sistem eksternal adalah mayoritas pengguna sistem informasi modern. Pengguna eksternal sering disebut remote user jauh (pengguna yang secara fisik tidak berada di tempat tapi masih membutuhkan akses ke sistem informasi) dan mobile user (pengguna yang lokasinya selalu berubah tapi membutuhkan akses ke sistem informasi dari manapun).

2.4.6 Business Manager

Kelompok lain dalam pengembangan sistem adalah manajer bisnis misalnya kepala bagian atau kepala departemen atau eksekutif perusahaan. Manajer-manajer ini penting karena mereka memiliki kekuatan pendanaan pengembangan sistemdan mengalokasikan sumber daya yang diperlukan untuk keberhasilan proyek.

2.5 Project Execution Plan (PEP)

Sebuah rencana eksekusi suatu proyek sangat erat kaitannya dengan estimasi biaya,dimana keduanya saling bergantung dan tidak akan terpenuhi keduanya secara total jika satudiantara keduanya tidak terselesaikan. Informasi yang ada dalam rencana eksekusi proyek iniantara lain :

a. Preeliminary Activites Schedule : merupakan informasi yang harus ada pertama dalamsebuah rencana eksekusi, meliputi : process design, AFE estimate and ,pproval danseleksi kontraktor pelaksana.
b. Procurement Schedule: hal ini berkaitan dengan pembelian dan pengiriman alatterutama yang membutuhkan waktu lama untuk pengiriman alat.
c. Engineering Schedule :semua jadwal perekayasaan (engineering) sudah dibuat sesuaid engan prinsip desain yang ada.
d. Subcontracting strategy and schedule : jenis kontrak yang ada sudah ditentukan (lump sum / reimbursable, unit price, competitive/negotiated)

c. Loaded construction schedule : seluruh kegiatan konstruksi sudah dirinci dengan durasiwaktu pengerjaan masing - masing tahap.

2.6 Devireables

Dalam manajemen proyek, hasil kerja (bahasa Inggris: *deliverable*) adalah objek berwujud atau tak berwujud yang merupakan hasil pelaksaan proyek, sebagai bagian dari suatu kewajiban atau obligasi. Istilah yang biasa dikaitkan secara spesifik dengan objektif ini, dapat berupa suatu kata benda: suatu barang, produk, atau artefak yang harus dibuat dan diberikan sebagai bagian kewajiban, atau suatu kata keterangan: menjelaskan sesuatu yang harus diberikan sebagai bagian dari kewajiban.

2.7 Pengertian Penjadwalan Proyek

Penjadwalan proyek merupakan salah satu elemen hasil perencanaan. Yang dapat memberikan informasi tentang jadwal rencana dan kemajuan proyek dalam hal kinerja sumber daya berupa biaya, tenaga kerja, peralatan dan material serta rencana durasi proyek dan progres waktu untuk menyelesaikan proyek. Dalam proses penjadwalan, penyusunan kegiatan dan hubungan antar kegiatan dibuat lebih terperinci dan sangat detail. Hal ini dimaksudkan untuk membantu pelaksanaan evaluasi proyek. Penjadwalan atau scheduling adalah pengalokasian waktu yang tersedia melaksanakan masing – masing pekerjaan dalam rangka menyelesaikan suatu proyek hingga tercapai hasil optimal dengan mempertimbangkan keterbatasan yang ada.

Selama proses pengendalian proyek, penjadwalan mengikuti per-kembangan proyek dengan berbagai permasalahannya. Proses monitoring serta updating selalu dilakukan untuk mendapatkan penjadwalan yang paling realistis agar alokasi sumber daya dan penetapan durasinya sesuai dengan sasaran dan tujuan proyek.

2.7.1 WBS (Work Breakdown Structure)

WBS adalah suatu metode pengorganisaian proyek menjadi struktur pelaporan hierarakis. WBS digunakan untuk melakukan Breakdown atau memecahkan tiap proses pekerjaan menjadi lebih detail.hal ini dimaksudkan agar proses perencanaan proyek memiliki tingkat yang lebih baik.

WBS disusun bedasarkan dasar pembelajaran seluuuh dokumen proyek yang meliputi kontrak, gambar-gambar, dan spesifikasi. Proyek kemudian diuraikan menjadi bagian-bagian dengan mengikuti pola struktur dan hirarki tertentu menjadi item-item pekerjaan yang cukup terperinci, yang disebut sebagai Wok Breakdown Structure.

2.7.2 Gantt Chart

Henry Laurence Gantt (1861-23 November 1919 di Calvert Country, Amerika) adalah seorang konsultan manajemen berlatarbelakang insinyur mekanik yang menciptakan peta Gantt (*Gantt Chart*) terkenal.

Gantt Chart merupakan gambaran dari macam-macam bagan yang mempunyai fungsi untuk:

a. Menentukan durasi pekerjaan terhadap perkembangan waktu.
b. Perencanaan dan penjadwalan proyek pekerjaan.
c. Pemantauan kemajuan proyek pekerjaan.

Gantt chart adalah bagan balok yang disusun dengan maksud mengidentifikasi unsur waktu dan urutan dalam merencanakan suatu kegiatan yang terdiri dari waktu mulai, waktu penyelesaian, dan pada saat pelaporan.

2.7.3 Milestone

Milestone adalah suatu bagian item pekerjaan yang dibuat seolah-olah menjadi temporary finish atau selesai sementara atas sekelompok atau serangkaian pekerjaan-pekerjaan yang menjadi bagian dari schedule besar. Item pekerjaan yang dijadikan milestone haruslah item pekerjaan yang dianggap menjadi bagian penting sebelum melanjutkan pekerjaan berikutnya atau berpengaruh atas kelangsungan pekerjaan berikutnya.

2.8 RAB

RAB (Rencana Anggaran Biaya) adalah penghitungan banyaknya biaya yang diperlukan untuk bahan dan upah, serta biaya-biaya lain yang berhubungan dengan pelaksanaan bangunan atau proyek, baik secara kasar/taksiran maupun secara teliti. Dalam penghitungan RAB suatu proyek, sering kali membutuhkan sebuah aplikasi program komputer agar perhitungan RAB cepat dan akurat.

2.9 Responsibility Assignment Matrix (RAM)

adalah sebuah matriks yang memetakan pekerjaan proyek, seperti yang dijelaskan dalam WBS, kepada orang-orang yang bertanggung jawab untuk melaksanakan pekerjaan.

Matriks ini terutama bermanfaat dalam menjelaskan peran dan tanggung jawab antarbagian di dalam suatu proyek atau proses. RACI merupakan akronim dari empat peran yang paling sering dicantumkan dalam matriks ini, yaitu *responsible*, *accountable*, *consulted*, dan *informed*.

Berikut keterangan tentang tiap peran ini:

a. Pelaksana (*responsible*): Orang yang melakukan pekerjaan.
b. Penanggung jawab (*accountable* atau *approver*): Orang yang bertanggung jawab terhadap penyelesaian pekerjaan atau menyetujui hasil suatu pekerjaan.
c. Penasihat atau pengarah (*consulted*): Orang yang dimintai pendapat tentang suatu pekerjaan.
d. Terinformasi (*informed*): Orang yang selalu mendapatkan informasi tentang kemajuan pekerjaan.

2.10 Analisa Resiko

Yaitu Proses mengidentifikasi, menganalisis, dan merencanakan risiko-risiko yang baru muncul, melacak risiko teridentifikasi, menganalisis ulang risiko sekarang, memonitor kondisi pemicu rencana kontingensi, memonitor sisa risiko, dan mereview pelaksanaan respon risiko saat mengevaluasi keefektivannya.

Dengan kata lain tujuannya adalah untuk memastikan bila: asumsi proyek masih valid, risiko (sebagaimana telah dinilai) berubah dari sebelumnya, kebijakan dan prosedur manajemen risiko diikuti, cadangan biaya dan jadwal kontingensi dimodifikasi sesuai risiko proyek.

Resiko juga dapat dibagi dua sisi yakni internal resiko dan eksternal resiko.

a. Internal Resiko

Dalam internal resiko ini berhubungan dengan perencanaan dari proyek tersebut. Jika kita tidak memikirkan rencana proyek tersebut dan merencanakan dengan baik . tidak jarang kita menemui hambatan dalam menjalankan proyek yang kita jalani. walaupun dalam perencanaan proyek sudah kita pikirkan semua namun tetap saja yang namanya resiko tetap ada di setiap perencanaan, sebab kita tidak bisa menebak-nebak apa yang terjadi didepan kita, jadi dengan merencanakan proyek kita sudah siap dengan resiko yang ada dan siap menghadapi resiko terutama internal resiko.

b. Eksternal Resiko

Dalam eksternal resiko ini berhubungan dalam pelaksanaan proyek. Yakni dalam pelaksaan proyek ini kita juga tidak bisa menebak-nebak apa yang akan terjadi dalam pelaksanaan proyek. Walaupun sebelumnya kita merencanakan proyek tersebut tidak menutup kemungkinan resiko dalam pelaksanaan proyek tetap ada. misalnya pekerjaan yang deadline tidak bisa terealisasi karena misalnya pekerja yang kurang atau proyeknya gagal karena keterbatasan dana itu dapat terjadi di eksternal resiko.

2.11 Metting Plan

Ketika ruang lingkup proyek telah ditetapkan dan tim proyek terbentuk, maka aktivitas proyek mulai memasuki tahap perencanaan. Pada tahap ini, dokumen perencanaan akan disusun secara terperinci sebagai panduan bagi tim proyek selama kegiatan proyek berlangsung. Adapun aktivitas yang akan dilakukan pada tahap ini adalah membuat dokumentasi project plan, resource

plan, financial plan, risk plan, acceptance plan, communication plan, procurement plan, contract supplier dan perform phare review.

2.12 Teori Pendukung Sistem Informasi Kepegawaian

2.12.1 Konsep Dasar Kepegawaian

Sebagaimana kita ketahui, landasan hukum pengaturan kepegawaian Pegawai Negeri Sipil adalah Undang-Undang Nomor 8 Tahun 1974 sebagaimana telah diubah dengan Undang-Undang Nomor 43 Tahun 1999 tentang Pokok-pokok Kepegawaian. Undang-undang Nomor 43 Tahun 1999 hanya bersifat perubahan beberapa ketentuan Undang-undang Nomor 8 Tahun Undang-undang Nomor 8 Tahun 1974 masih tetap berlaku sepanjang belum dicabut/dirubah keseluruhannya. diturunkan dalam Peraturan Pemerintah dan Surat Edaran Badan Kepegawaian Negara, kemudian dapat diturunkan lagi dalam peraturan menteri terkait. Pengaturan secara teknis Undang-undang tersebut

2.12.2 Pengertian Administrasi Kepegawaian

Administrasi kepegawaian berkaitan dengan penggunaan sumber daya manusia dalam suatu organisasi. Berkaitan dengan hal tersebut, dalam kegiatan belajar ini telah dikemukakan beberapa pendapat para ahli tentang pengertian, ruang lingkup, dan fungsi/aktivitas kepegawaian.

2.12.3 Sistem Administrasi Kepegawaian

Sistem administrasi kepegawaian adalah bagian dari administrasi negara yang kebijaksanaannya ditentukan dari tujuan yang ingin dicapai. Pola kebijaksanaannya tergantung pada bentuk negara yang dianut suatu negara, apakah federal ataukah kesatuan.

Kebijaksanaan dasar sistem administrasi kepegawaian di negara kita mengacu pada Undang-Undang Nomor 43 Tahun 1999 tentang perubahan atas Undang-Undang Nomor 8 Tahun 1974 tentang Pokok-Pokok Kepegawaian. Dalam undang-undang tersebut dinyatakan bahwa dalam rangka usaha mencapai tujuan nasional untuk mewujudkan masyarakat madani yang taat hukum,

berperadaban modern, demokratis, adil, dan bermoral tinggi, diperlukan pegawai negeri yang merupakan unsur aparatur negara yang bertugas sebagai abdi masyarakat yang menyelenggarakan pelayanan secara adil dan merata, menjaga persatuan dan kesatuan bangsa dengan penuh kesetiaan kepada Pancasila dan Undang-Undang Dasar 1945.

2.12.4 Fungsi Teknis Administrasi Kepegawaian

Administrasi kepegawaian pada hakikatnya melakukan dua fungsi yaitu fungsi manajerial, dan fungsi operatif (teknis). Fungsi manajerial berkaitan dengan pekerjaan pikiran atau menggunakan pikiran (mental) meliputi perencanaan, pengorganisasian, pengarahan, dan pengendalian pegawai. Sedangkan fungsi operatif (teknis), berkaitan dengan kegiatan-kegiatan yang dilakukan dengan fisik, meliputi pengadaan, pengembangan, kompensasi, integrasi, pemeliharaan, dan pemensiunan pegawai.