BAB II

TINJAUAN PUSTAKA

II.1. Aplikasi

Aplikasi berasal dari kata *application* yang artinya penerapan, lamaran, penggunaan. Secara istilah aplikasi adalah program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju. Perangkat lunak aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan tugas yang diinginkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja, dan pemutar media (Fricles Ariwisanto Sianturi; 2013: 43).

II.2. Kriptografi

Kriptografi (*cryptography*) berasal dari Bahasa Yunani: *"cryptós"* artinya *"secret"* (rahasia), sedangkan *"gráphein"* artinya *"writing"* (tulisan). Jadi, kriptografi berarti *"secret writing"* (tulisan rahasia). Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data, serta otentikasi(Mariana; 2011: 2).

Kriptografi merupakan seni dan ilmu menyembunyikan informasi dari penerima yang tidak berhak. Kata kriptografi berasal dari kata Yunani *kryptos* (tersembunyi) dan *graphein*(menulis). Dalam teknologi informasi telah dan

sedang dikembangkan cara-cara untuk menangkal berbagai serangan, seperti penyadap dan pengubahan data yang sedang dikirimkan.Transformasi ini memberikan solusi pada dua macam masalah keamanan data, yaitu masalah privasi *(privacy*) dan keotentikan (*authenticatioan*). Kriptografi tidak berarti hanya memberikan keamanan informasi saja, namun lebih ke arah teknik-tekniknya.

a. Enkripsi

Proses enkripsi adalah proses penyandian pesan terbuka (plaintext) menjadi pesan rahasia (ciphertext). Ciphertext inilah yang nantinya akan dikirimkan melalui saluran komunikasi terbuka. Pada saat ciphertext diterima oleh penerima pesan, maka pesan rahasia tersebut diubah lagi menjadi pesan terbuka melalui proses dekripsi sehingga pesan tadi dapat dibaca kembali oleh penerima pesan.

b. Dekripsi

Dekripsi merupakan proses kebalikan dari proses enkripsi, merubah ciphertext kembali ke dalam bentuk plaintext.Untuk menghilangkan penyandian yang diberikan pada saat proses enkripsi,membutuhkan penggunaan sejumlah informasi rahasia,yang disebut sebagai kunci (Fricles Ariwisanto Sianturi; 2013:43).

II.3. E-mail

Email (Electronic Mail) adalah pertukaran pesan digital dari seseorang dengan satu atau beberapa orang. Email dikirimkan melalui media Internet atau jaringan komputer lokal. Sistem email terdiri dari dua subsistem yaitu Mail User Agent (MUA) dan Mail Transfer Agent (MTA). Berikut adalah fungsi dasar dari sistem email: 1. Composition, yaitu membuat pesan. Sistem menyediakan cara atau antarmuka untuk mengisi pengalamatan dan beberapa header fields untuk dilekatkan pada pesan, 2. Transfer, yaitu memindahkan pesan dari alamat asal ke alamat tujuan. Proses ini membutuhkan koneksi dengan tujuan untuk bertukar data. Proses transfer menggunakan protokol tertentu (IMAP atau POP3) dan keduanya harus menggunakan protokol yang sama, 3. Reporting, yaitu proses memberikan informasi tentang status pesan seperti apakah pesan terkirim, apakah pesan ditolak, dan sebagainya, 4. Displaying, yaitu proses menampilkan pesan agar dapat dibaca oleh pengguna, 5. Disposition, yaitu proses yang terjadi setelah pesan diterima oleh pengguna seperti penghapusan pesan, penyimpanan pesan, meneruskan pesan, dan sebagainya (Pandu Wicaksono;2011;2).

II.4. Visual Basic .Net

Visual basic 2010 merupakan salah satu bagian dari produk pemrograman terbaru yang dikeluarkan oleh Microsoft, yaitu microsoft Visual Studio 2010. Sebagai produk lingkungan pengembangan terintegrasi atau IDE andalan yang di keluarkan oleh microsoft, visual studio 2010 menambahkan perbaikan-perbaikan

fitur dan fitur baru yang lebih lengkap visual studio pendahuluanya, yaitu mirosoft visual studio 2008. (Wahana Komputer;2010;2).

Sedangkan menurut Aswan (2012 : 1) Visual basic 2010 adalah salah satu bagian dari microsoft visual studio 2010. Sebuat alat yang digunakan oleh pengembang windows dari berbagai level untuk mengembangkan dan membangun aplikasi yang bergerak diatas sistem .NET Framework, dengan menggunakan bahasa BASIC. Visual Basic menyediakan cara cepat dan mudah untuk membuat aplikasi.

Setiap generasi baru dari perangkat lunak bahasa pemrograman datang karena adanya keterbatasan dari generasi sebelumnya. Teknologi device, hardware, network dan internet baru yang muncul menyebabkan bahasa pemrograman yang ada tidak lagi menjadi alat yang ideal untuk mengembangkan perangkat lunak yang dapat bekerja dengan teknologi baru tersebut (WAH[12]). Sekarang untuk pertama kalinya, platform pengembang perangkat lunak yang lengkap, Microsoft .NET telah didesain dari dasar dengan internet sebagai fokus utamanya (walaupun tidak secara eksklusif hanya untuk pengembang internet saja). Banyak inovasi baru yang berada dalam platform ini akan mengatasi keterbatasan dari tool-tool dan teknologi lama. Visual Basic .NET adalah pengembangan dari Visual basic sebelumnya. Kelebihan VB .NET 2010 terletak pada tampilannya yang lebih canggih dibandingkan dengan edisi Visual Basic sebelumnya. Selain memiliki kelebihan, VB .NET 2005 memiliki kekurangan. Kekurangan VB .NET 2005 yang terlihat jelas adalah beratnya aplikasi ini apabila dijalankan pada komputer yang memiliki spesifikasi sederhana.

II.5. AES (*Advanced Encryption Standard*)

AES dipublikasikan oleh NIST (*National Institute of Standard and Technology*) pada tahun 2001 yang digunakan untuk menggantikan algoritma DES yang semakin lama semakin mudah untuk membobol kuncinya. AES diperoleh dari hasil kompetisi yang diadakan NIST pada tahun 1997. Pada tahap pertama, 15 peserta dari 21 peserta lolos ke tahap berikutnya berdasarkan penilaian tingkat keamanan, harga, algoritma, dan karakteristik implementasi. Sepuluh dari 15 peserta tersebut gugur pada tahap berikutnya karena dianggap kurang aman dan kurang efektif. Pada Agustus 1999 dipilih lima kandidat untuk seleksi akhir, yaitu Mars (IBM, Amerika Serikat), RSA (RSA corp., Amerika Serikat), Rijndael (Belgia), Serpent (Israel, Norwegia, dan Inggris), dan Twofish (Counterpane, Amerika Serikat). Pada tahap ini, NIST memberikan penilaian terhadap *general security*, implementasi *software*, ruang lingkup, implementasi *hardware*, implementasi atas serangan, enkripsi dan dekripsi, kemampuan kunci, kemampuan lain dan fleksibilitas, dan kepotensialan untuk tingkat intruksi paralel. Akhirnya, pada tanggal 2 Oktober 2000 terpilihlah algoritma *Rijndael*, yang dibuat oleh Dr. Vincent Rijment dan Dr. Joan Daemen, sebagai pemenang.

Input dan output dari algoritma AES terdiri dari urutan data sebesar 128 bit. Urutan data yang sudah terbentuk dalam satu kelompok 128 bit tersebut disebut juga sebagai blok data atau *plaintext* yang nantinya akan dienkripsi menjadi *ciphertext*. Algoritma AES merupakan algoritma simetris yaitu mengunakan kunci yang sama untuk proses enkripsi dan dekripsi. Algoritma AES memiliki tiga pilihan kunci yaitu tipe: AES-128, AES-192 dan AES-256. Masing-

masing tipe menggunakan kunci internal yang berbeda yaitu *round key* untuk setiap proses putaran. Proses putaran enkripsi AES-128 dikerjakan sebanyak 10 kali (a=10), yaitu sebagai berikut: 1. Addroundkey 2. Putaran sebanyak a-1 kali, proses yang dilakukan pada setiap putaran adalah: *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey*. 3. Final round, adalah proses untuk putaran terakhir yang meliputi *SubBytes*, *ShiftRows*, dan *AddRoundKey*. Sedangkan pada proses dekripsi AES-128, proses putaran juga dikerjakan sebanyak 10 kali(a=10).

Proses enkripsi algoritma AES terdiri dari 4 jenis transformasi bytes, yaitu *SubBytes*, *ShiftRows*, *Mixcolumns*, dan *AddRoundKey* (Fricles Ariwisanto Sianturi; 2013: 43).

II.6. UML *(Unified Modeling Language)*

UML(*Unified Modeling Language)* yang merupakan metodologi kolaborasi antara metoda booch, OMT (*Object Modeling Technique),* serta OOSE (*Oriented Sofware Engineering)* dan beberapa metoda lainnya, merupakan metodologi yang paling sering digunakan saat ini untuk mengadaptasi maraknya penggunaan bahasa "pemrograman berorientasi objek" (OOP). (Adi Nugroho;2009;4)

UML (*Unified Modeling Language*) adalah sebuah "bahasa" yang telah menjadi standar dalam industry untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Seperti bahasa-bahasa lainnya, UML mendefenisikan notasi dan *sintax/semantic*. Notasi UML merupakan sekumpulan

bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *sintax* mendefinisikan bagaimana bentuk – bentuk tersebut dapat dikombinasikan. *Unified Modeling Language* biasa digunakan untuk :

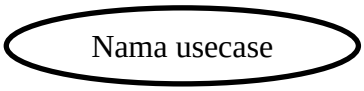
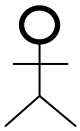
1. Menggambarkan batasan sistem dan fungsi – fungsi sistem secara umum, di buat dengan *use case* dan *actor*.
2. Menggambarkan kegiatan atau proses bisnis yang di laksanakan secara umum, di buat dengan *interaction diagrams*.
3. Menggambarkan representasi struktur *static* sebuah sistem dalam bentuk *class diagram*s.
4. Membuat model behavior "yang menggambarkan kebiasaan atau sifat sebuah sistem" dengan *state transition diagrams*.
5. Menyatakan arsitektur implementasi fisik menggunakan *component and development diagrams*.
6. Menyampaikan atau memperluas *fungsionality* dengan *stereotypes*. (Yuni Sugiarti; 2013 :36)

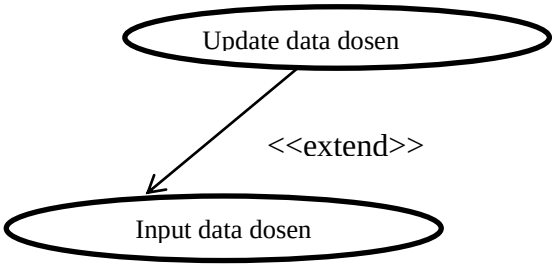
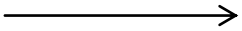
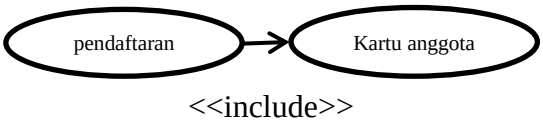
II.6.1. Use Case Diagram

Use case diagrams merupakan pemodelan untuk menggambarkan kelakuan (*behavior*) sistem yang akan dibuat. Diagram *use case* mendeskripsikan sebuah interaksi antara satu atau lebih *actor* dengan sistem yang akan dibuat. Dengan pengertian yang cepat, diagram *use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem dan siapa saja yang berhak

menggunakan fungsi–fungsi tersebut. Terdapat beberapa simbol dalam menggambarkan diagram *use case*, yaitu *use case, actor* dan relasi. Berikut adalah sismbol – simbol yang ada pada diagram *use case*. (Yuni Sugiarti; 2013: 42)

Tabel II.1 Simbol – simbol pada Use Case Diagram

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau *actor*; biasanya ditanyakan dengan menggunakan kata kerja di awal frase nama *use case*.
Aktor  nama aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari *acto*r adalah gambar orang, tapi *actor* belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda diawal frase nama *actor*.
Asosiasi/ *association* 	Komunikasi antara actor daan use case yang berpartisipasi pada use case atau use case memiliki interaksi dengan kator.

Extend  <<extend>>	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan itu; mirip dengan prinsip inheritance pada pemrograman berorientasi objek; biasanya use case tambahan memiliki nama depan yang sama dengan use case yang ditambahkan, arah panah menunjuk pada use case yang dituju. Contoh : 
Include <<include>> 	Relasi use case tambahan kesebuah use case dimana use case yang yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini. Ada dua sudut pandang yang cukup besar mengenai include di use case, *include* berarti use case yang ditambahkan akan selalu dipanggil saat use case tambahan dijalankan, contoh : 

Sumber: (Yuni Sugiarti; 2013)

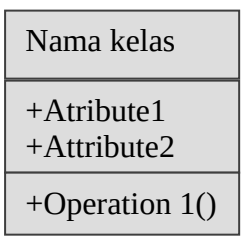
II.6.2. *Class* Diagram

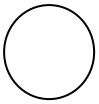
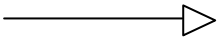
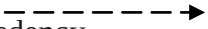
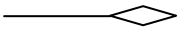
Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefenisian kelas – kelas yang akan di buat untuk membangun sistem. Kelas memiliki apa yang di sebut atribut dan metode atau operasi.

1. Atribut merupakan variabel- variabel yang di miliki oleh suatu kelas.
2. Atribut mendeskripsikan properti dengan sebaris teks di dalam kotak kelas tersebut.
3. Operasi atau metode adalah fungsi – fungsi yang di miliki oleh suatu kelas.

Diagram kelas mendeskripsikan jenis – jenis objek dalam sistem dan berbagai hubungan statis yang terdapat di antara mereka. Diagram kelas juga menunjukkan properti dan operasi sebuah kelas dan batasan – batasan yang terdapat dalam hubungan – hubungan objek tersebut. (Yuni Sugiarti; 2013: 57)

Tabel II.2 Simbol – simbol Class Diagram

Simbol	Deskripsi
Package 	Package merupakan sebuah bungkusan dari satu atau lebih kelas
Operasi 	Kelas pada struktur sistem

Antarmuka / interface 	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi 1 1..*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity.*
Asosiasi berarah/directed asosiasi	Relasi antar kelas dengan makna kelas yang satu di gunakan oleh kelas yang lain, asosiasi biasanya juga di sertai dengan *multiplicity*.
Generalisasi 	Relasi antar kelas dengan makna generalisasi – spesialisasi (umum khusus).
Kebergantungan / defedency 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi 	*Relasi antar kelas dengan makna semua bagian (whole-part)*

Sumber : (Yuni Sugiarti ; 2013)

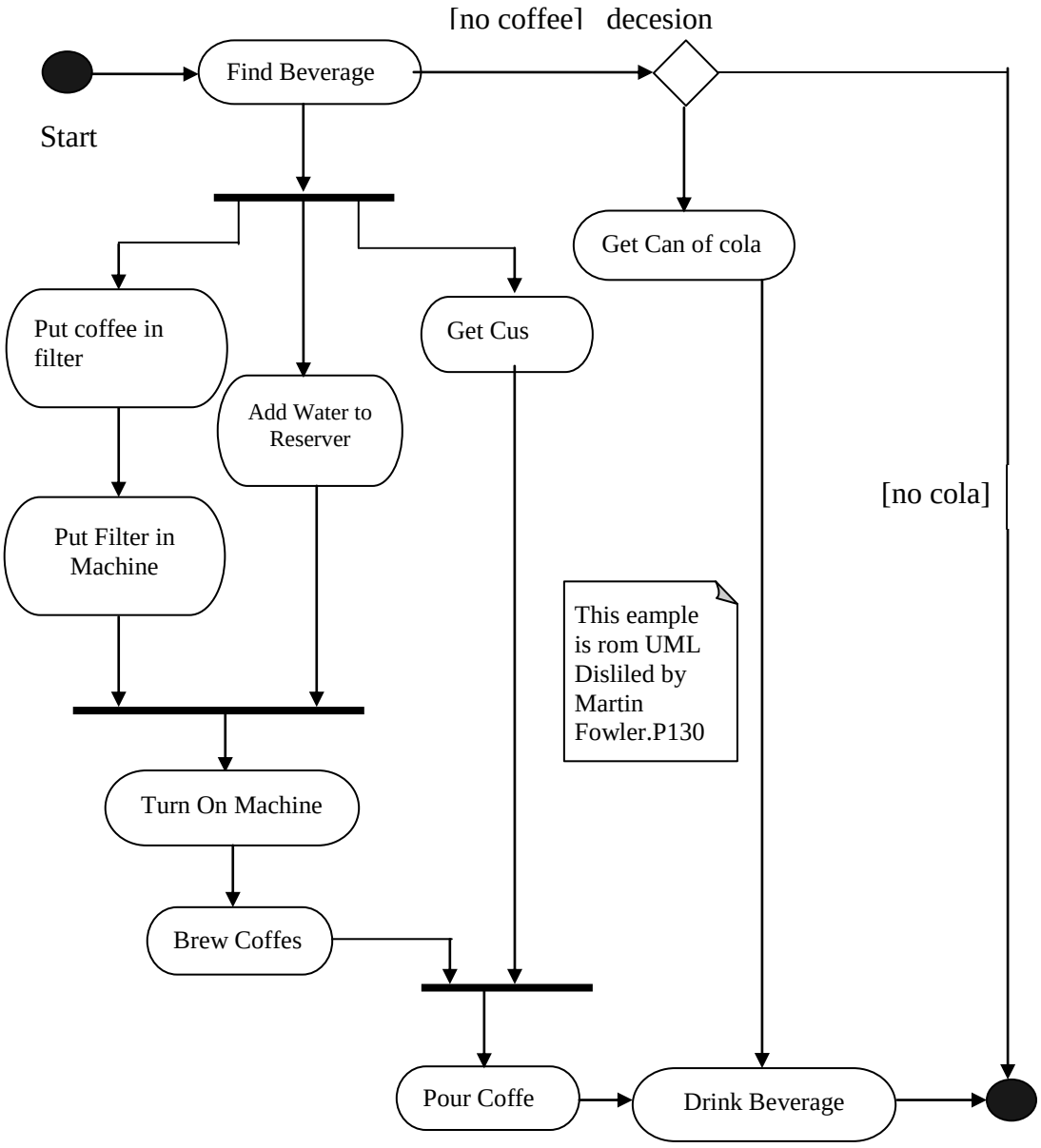
II.6.3. *Activity* Diagram

Diagram aktivitas atau *activity* diagram menggambarkan *workflow* (aliran kerja) atau aktifitas dari sebuah sistem atau proses bisnis.

Activity diagram merupakan *state* diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya state sebelumnya (*internal processing*). Oleh karena itu *activity* diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem)

secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu use case atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. (Yuni Sugiarti; 2013: 75)



Gambar II.1 Activity Diagram

Sumber : (Yuni Sugiarti ; 2013)

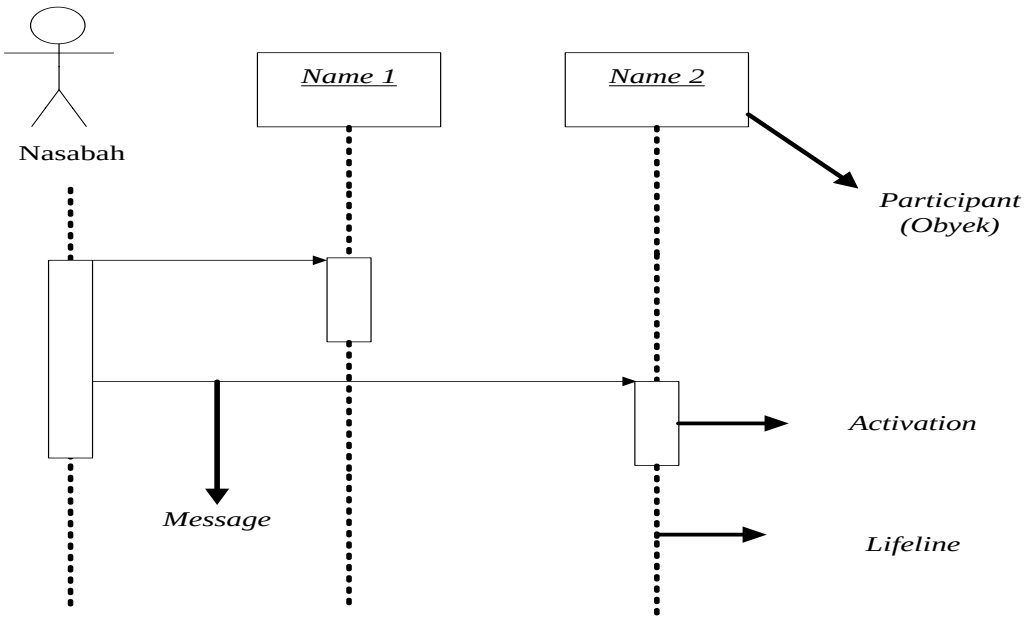
II.6.4. *Sequence* Diagram

Diagram sekuence menggambarkan kelakuan/ pelaku objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sequence maka harus diketahui objek – objek yang terlibat dalam sebuah use case beserta metode – metode yang dimiliki kelas yang diinstasiasi menjadi objek itu.

Diagram sequence memiliki ciri yang berbeda dengan diagram interaksi pada diagram kolaborasi sebagai berikut :

1. Pada diagram sequence terdapat garis hidup objek. Garis hidup objek adalah garis vertical yang mencerminkan eksistensi sebuah objek sepanjang periode waktu. Sebagian besar objek – objek yang tercakup dalam diagram interaksi akan eksis sepanjang durasi tertentu dari interaksi, sehingga objek – objek itu diletakkan dibagian atas diagram dengan garis hidup tergambar dari atas hingga bagian bawah diagram. Suatu objek lain dapat saja diciptakan, dalam hal ini garis hidup dimulai saat pesan *destroy*,
2. jika kasus ini terjadi, maka garis hidupnya juga berakhir.
3. Terdapat focus kendali (*Focus Of Control*), berupa empat persegi panjang ramping dan tinggi yang menampilkan aksi suatu objek secara langsung atau sepanjang sub ordinat. Puncak dari empat persegi panjang adalah permulaan aksi, bagian dasar adalah akhir dari suatu aksi. Pada diagram ini mungkin juga memperhatikan penyaringan (*nesting*) dan *focus* kendali yang disebabkan oleh proses rekursif dengan menumpuk *focus* kendali yang lain pada induknya. (Yuni Sugiarti; 2013: 70)

Berikut simbol – simbol yang ada pada sequence diagram.



Gamabar II.2 Simbol Squence

Sumber : (Yuni Sugiarti ; 2013)

II.7. ERD *(Entity Relationship Diagram)*

Entity relationship diagram adalah alat pemodelan data utama dan akan membantu mengorganisasi data dalam suatu proyek ke dalam entitas – entitas dan menentukan hubungan antar entitas. Proses memungkinkan analis menghasilkan struktur basis data yang baik sehingga data dapat disimpan dan diambil secara efisien. Elemen – elemen diagram hubungan entitas yaitu :

1. Entitas (*Entity*)

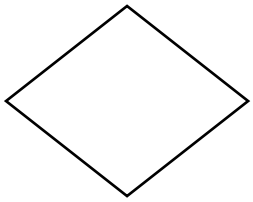
   Entitas dalah sesuatu yang nyata atau abstrak diman kita akan menyimpan data. Ada 4 kelas entitas, yaitu misalnya pegawai, pembayaran, kampus dan buku.

Gambar II.3 Simbol Entitas

Sumber : (Janner Simarmata,dkk; 2013)

2. Relasi (*Relationship*)

   Relasi adalah hubungan alamiah yang terjadi antara satu atau lebih entitas, misalny proses pembayaran pegawai. Kardinalitas menentukan kejadian suatu entitas untuk satu kejadian pada entitas yang berhubungan. Misalnya mahasiswa bisa mengambil banyak mata kuliah.

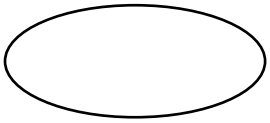


Gambar II.4 Simbol Relasi

Sumber : (Janner Simarmata,dkk; 2013)

3. Atribut (*Attribute*)

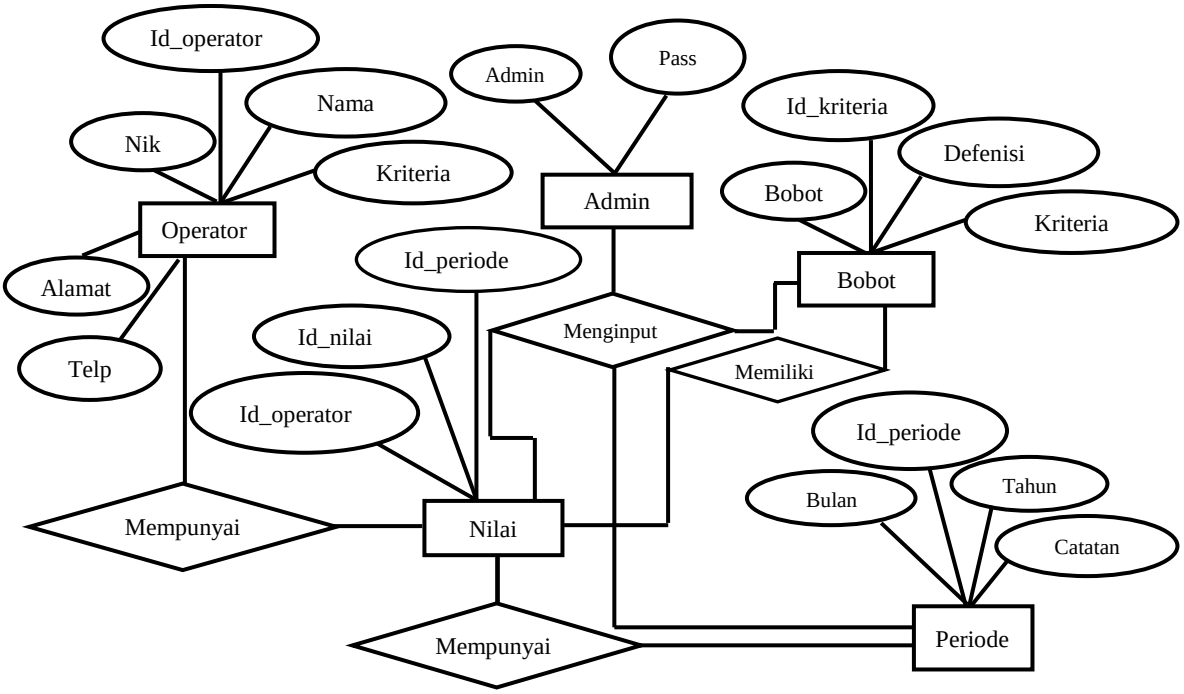
Atribut adalah ciri umum semua semua atau sebagian besar instansi pada entitas tertentu. Sebutan lain atribut adalah *property*, elemen data, dan *field*. Misalnya, nama, alamat, nomor pegawai, dan gaji adalah atribut entitas pegawai. Sebuah atribut atau kombinasi atribut yang mengidentifikasikan satu dan hanya satu instansi suatu entitas disebut kunci utama atau pengenal. Misalnya, nomor pegawai adalah kunci utama untuk pegawai.



Gambar II.5 Simbol Atribut

Sumber : (Janner Simarmata,dkk; 2013)

Berikut contoh ERD



Gambar II.6 Contoh ERD

Sumber : (Janner Simarmata,dkk; 2013)

II.8. Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basisdata relasional. Pada dasarnya, normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan menghilangkan kelompok berulang lalu menghilang data yang terduplikasi dari tabel relasional (www.utexas.edu). (Janner Simarmata & dkk; 2010 : 77)

Normalisasi adalah bagian perancangan basis data.Tanpa normalisasi, sistem basis data menjadi tidak akurat, lambat, tidak efisien, serta tidak memberikan data yang diharapkan. Pada waktu menormalisasikan basis data, ada empat tujuan yang harus dicapai, yaitu:

1. Mengatur data dalam kelompok – kelompok sehingga masing – masing kelompok hanya mengenai bagian kecil sistem.
2. Meminimalkan jumlah data berulang dalam basis data.
3. Membuat basis data yang datanya diakses dan dimanipulasi secara cepat dan efisien tanpa melupakan integritas data.
4. Mengatur data sedemikian rupa sehingga ketika memodifikasi data, anda hanya mengubah pada satu tempat.

II.8.1. Bentuk – Bentuk Normalisasi

1. Bentuk normal pertama (1NF)

   Tahap ini dilakukan penghilangan beberapa group elemen yang berulang agar menjadi satu harga tunggal yang berinteraksi di antara setiap baris

pada suatu tabel, dan setiap *atribut* harus mempunyai nilai data yang *atomi*c (bersifat *atomic value*).

2. Bentuk normal kedua (2NF)

   Normal kedua didasari atas konsep *full functioanl dependency* (ketergantungan fungsional sepenuhnya) yang dapat didefinisikan sebagai berikut.

   Jika A dan B adalah *atribut-atribut* dari suatu relasi, B dikatakan *full function dependency* (mimiliki ketergantungan fungsional sepenuhnya) terhadap A, jika B adalah tergantung fungsioal terhadap A, tetapi tidak secara tepat memiliki ketergantungan fungsional dari *subset* (himpunan bagian) dari A.

3. Bentuk normal ketiga (3NF)

   Jika kita hanya meng*update*satu baris saja, sementara baris yang lainnya tidak, maka data di dalam *database* tersebut akan *inkonsisten*/tidak teratur. Anomali *update* ini disebabkan oleh suatu ketergantungan transitif *(transitive dependency)*. Kita harus menghilangkan ketergantungan tersebut dengan melakukan normalisasi ketiga (3-NF).

4. Bentuk normal *boyce-code* (BCNF)

   Suatu relasi dalam basis data harus dirancang sedemikian rupa sehingga mereka memiliki ketergantungan sebagaian *(partial dependency)*, maupun ket-ergantungan transitif.

II.9. Kamus Data

Kamus data (*data dictionary*) dipergunakan untuk memperjelas aliran data yang digambarkan pada DFD. Kamus data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukkan (*input)* dan keluaran (*output*) dapat dipahami secara umum (memiliki standar cara penulisan). Kamus data biasanya berisi:

1. Nama - nama dari data
2. Digunakan pada – merupakan proses-proses yang terkait data
3. Deskripsi – merupakan deskripsi data
4. Informasi tambahan – seperti tipe data, nilai data, batas nilai data dan komponen yang membentuk data. (Rosa A.S & M Shalauddin; 2011 : 67)