BAB II

TINJAUAN PUSTAKA

II.1 Pengertian Sistem

Kata sistem berasal dari bahasa Yunani yaitu ”*Systema*” yang berarti kesatuan. Sistem adalah sekumpulan komponen yang saling berhubungan yang harus bekerja bersama-sama untuk menghasilkan suatu kesatuan metode, prosedur teknik yang digabungkan dan diatur sedemikian rupa sehingga menjadi satu kesatuan yang berfungsi untuk mencapai tujuan (Jogianto HM ; 2005 : 4).

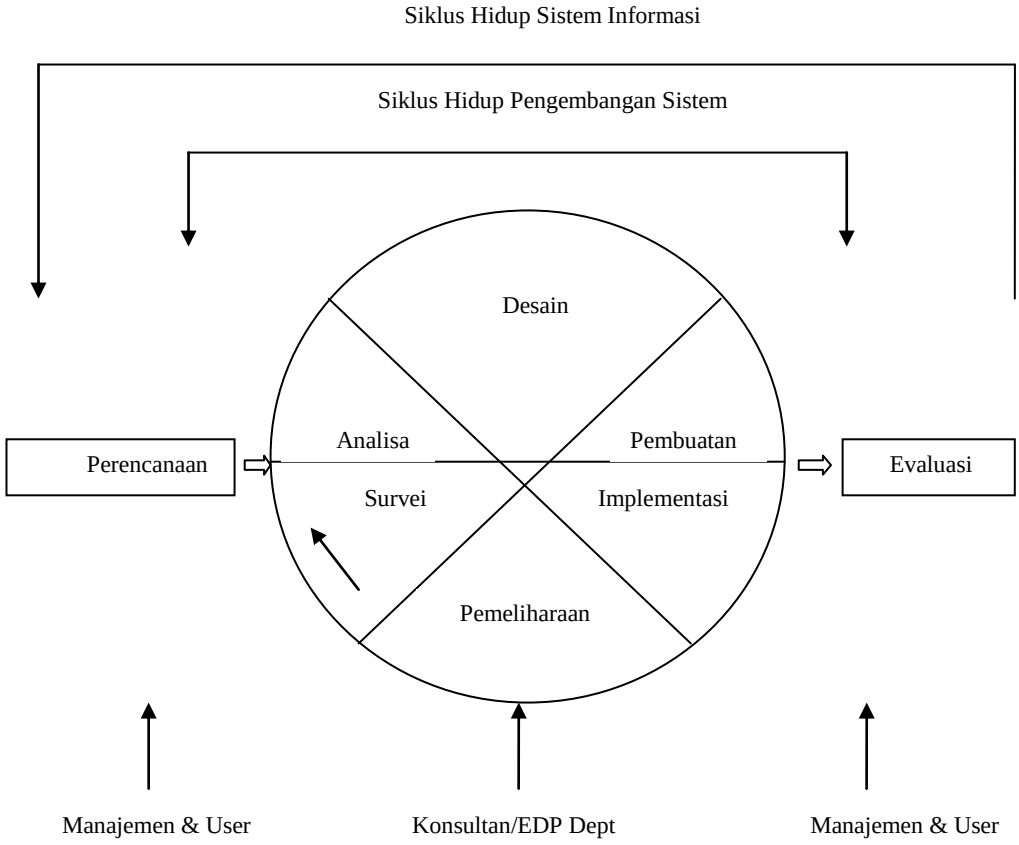
Sistem adalah jaringan dari pada elemen-elemen yang saling berhubungan membentuk suatu kesatuan untuk melaksanakan suatu tujuan pokok dari sistem tersebut (Jogiyanto HM ; 2005 : 4).

Secara sederhana sistem dapat diartikan sebagai suatu kumpulan dan himpunan dari unsur, komponen atau variabel-variabel yang terorganisasi, saling berinteraksi, saling tergantung satu sama lain dan terpadu. Teori sistem secara umum pertama kali diuraikan oleh Kenneth Boulding, terutama menekankan pentingnya perhatian terhadap setiap bagian yang membentuk sebuah sistem (Tata Sutabri,MM ; 2004 : 4).

II.1.1 Siklus Hidup Pengembangan Sistem

Siklus hidup sistem informasi dimulai dari fase perencanaan, fase pengembangan (investigasi, analisis, disain, implementasi), dan dievaluasi secara terus-menerus untuk menetapkan apakah sistem informasi tersebut masih layak diaplikasikan. Jika tidak maka sistem informasi tersebut akan diganti dengan yang

baru dan dimulai dengan perencanaan kembali, seperti tampak pada Gambar 2.1 dibawah ini.



Gambar II.1 Siklus Hidup Pengembangan Sistem

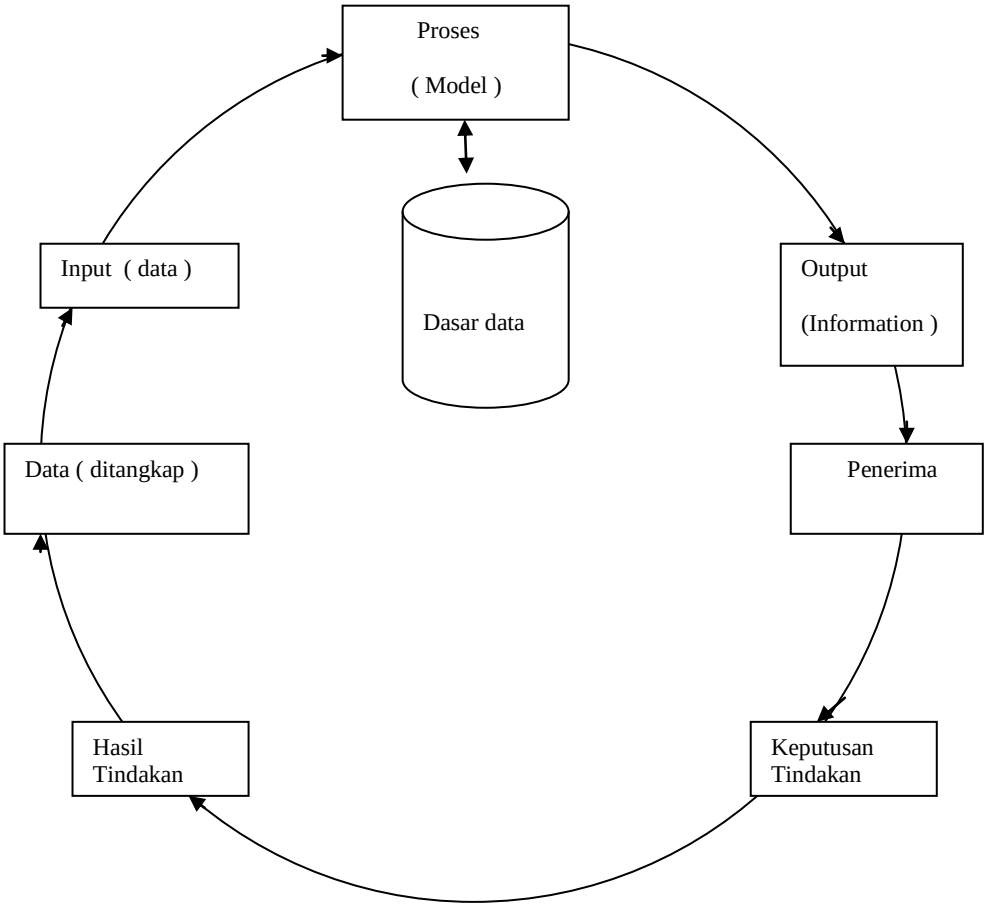
(Sumber : Tata Sutabri ; 2004 :3)

II.1.2 Pengertian Informasi Dan Data

Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi penerimanya. Sumber dari informasi adalah data. Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan nyata. Kejadian-kejadian *(event)* adalah sesuatu yang terjadi pada saat yang tertentu. Data merupakan bentuk yang masih mentah yang belum dapat bercerita banyak,

sehingga perlu diolah lebih lanjut. Data diolah melalui suatu model untuk dihasilkan informasi (Jogiyanto HM ; 2005 : 8).

Data yang diolah untuk menghasilkan informasi menggunakan suatu model proses tertentu. Data yang diolah menjadi suatu model menjadi informasi, penerima kemudian menerima informasi tersebut, membuat suatu keputusan dan melakukan suatu tindakan, yang berarti menghasilkan suatu tindakan yang lain yang akan membuat sejumlah data kembali. Data tersebut akan ditangkap sebagai input, diproses kembali lewat suatu model dan seterusnya membentuk suatu siklus. Siklus in oleh John Burch disebut dengan siklus informasi (*information cycle)* . Siklus ini disebut juga dengan siklus pengolahan data *(data processing cycle).* Pada gambar II.2

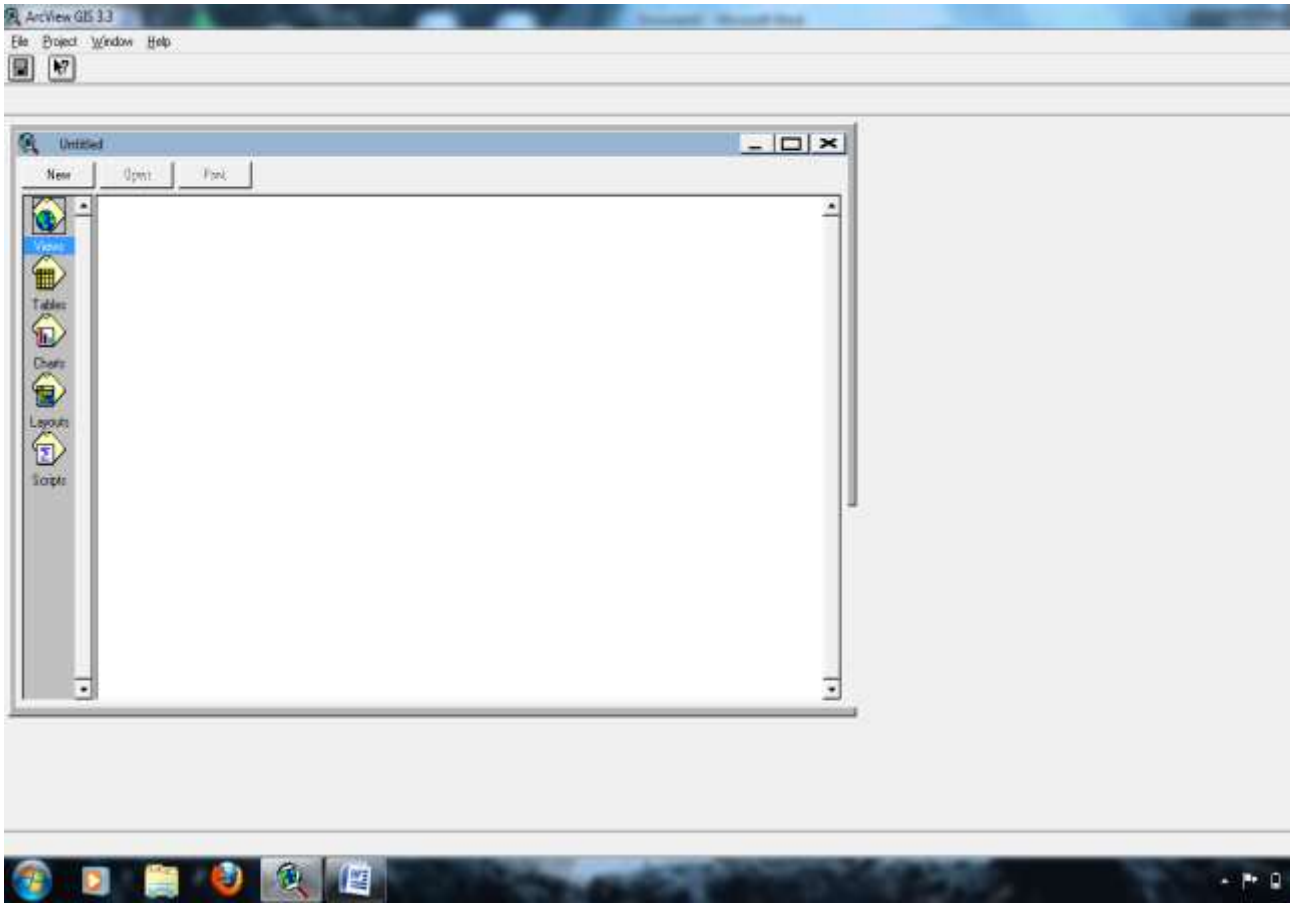


Gambar II.2 Siklus Informasi

(Sumber Jogiyanto HM ; 2005 :9)

II.2 Sistem Informasi Geografis (*Geographical Information System)*

Sistem Informasi Geografis (SIG) adalah sebuah rangkaian sistem yang memanfaatkan teknologi digital untuk melakukan analisa spasial. Sistem ini memanfaatkan perangkat lunak dari komputer untuk melakukan pengolahan data (Eko Budiyanto ; 2004 : 2). Sistem informasi geografis digunakan untuk menangani data spasial atau data tentang keruangan. Sistem seperti ini banyak digunakan antara lain untuk pemetaan tanah dan agrikultur, arkeologi, jaringan listrik, dan geologi. Sistem seperti ini sudah lama diterapkan (Abdul khadir;2003).



Gambar II.4.: Contoh Software *ARCVIEW* GIS

Sumber : Eddy Prahasta

Software ini bekerja untuk mengambarkan sebuah peta yang akan mau kita gambar atau yang akan mau kita rancang. Dan dimana Sistem Infornasi Geografis mempelajari tentang pemetaan dan di mana software yang dipakai oleh penulis adalah *ARCVIEW*. Dan software tersebut dapat digunakan untuk dalam sebeuah pemetaan sebuah wilayah. Dan di dalam gambar tersebut akan dijelaskan sebagai berikut:

1. View : Berfungsi untuk menampilkan tampilan awal.
2. Table : Berfungsi untuk menggambarkan suatu peta yang akan mau kita gambar.
3. Script : Berfungsi untuk membuat suatu program dalam suatu pemataan.

II.3 PHP

PHP adalah salah satu bahasa pemograman yang berjalan dalam sebuah web server dan berfungsi sebagai pengolah data pada sebuah server. Dengan menggunakan PHP, sebuah website akan lebih interaktif dan dinamis (MADCOMS ; 2004; 2009 : 1).

PHP (*PHP Hypertext Prepocessor*) adalah bahasa scripting server-side bagi pemrograman web. Secara sederhana, PHP merupakan tool bagi pengembangan web dinamis. Php sangat populer karena memiliki fungsi lengkap, cepat, mudah dipelajari, dan bersifat gratis. Skrip PHP cukup disisipkan pada kode HTML agar dapat bekerja. PHP dapat berjalan di berbagai web server dan sistem operasi yang berbeda (Angga Wibowo ; 2007 : 2).

PHP Hypertext Prepocessor merupakan singkatan dari PHP yang merupakan bahasa yang berbentuk skrip yang ditempatkan dalam server dan diproses di server. Hasilnyalah yang dikirimkan ke klien tempat pemakai menggunakan *browser* (Abdul Kadir ; 2008 : 2).

Model kerja HTML diawali dengan permintaan suatu halaman web oleh browser. Berdasarkan URL (Uniform Resource Locator) atau dikenal dengan sebutan alamat Internet, browser mendapatkan alamat web server, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala infromasi yang dibutuhkan oleh web server.

Selanjutnya web server akan mencarikan file yang meminta dan memberikan isinya ke web browser (atau yang biasa disebut browser saja). Browser yang mendapatkan isinya segera melakukan proses penerjemahan kode HTML dan menampilkan ke layar pemakai (Abdul Kadir ; 2008 : 4).

PHP memiliki banyak kelebihan yang tidak dimiliki oleh bahasa script sejenis. PHP difokuskan pada pembuatan sript server-side, yang bisa melakukan apa saja yang dilakukan oleh CGI, seperti mengumpulkan data dari form, menghasilkan isi halaman web dinamis, dan kemampuan mengirim serta menerima cookies, bahkan lebih pada kemampuan CGI (Kasiman Perangin-Angin ; 2006 : 3).

II.4 MySQL

MySQL adalah salah satu jenis database server yang sangat terkenal. Kepopulerannya disebabkan MySQL menggunakan SQL sebagai bahasa dasar

untuk mengakses databasenya. Selain itu, ia bersifat Open Source (Anda tidak perlu membayar untuk menggunakannya) pada pelbagai platform (kecuali untuk jenis Enterprice, yang bersifat komersial). Perangkat Lunak MySQL sendiri bisa di download dari http://mysql.com. (Abdul Kadir ; 2008 : 348).

MySQL termasuk jenis RDBMS (Relational Database Management System). Itulah sebabnya, istilah seperti tabel, baris, dan kolom digunakan pada MySQL. Pada MySQL, sebuah database mengandung satu atau sejumlah tabel. (Abdul Kadir ; 2008 : 348).

MySQL (*My Structure Query Language*) atau yang biasa dibaca "mai-se-kuel" adalah sebuah program pembuat database yang bersifat open source dan berjalan di semua platform baik Windows maupun Linux. Selain itu, MySQL juga merupakan program pengakses database yang bersifat jaringan sehingga dapat digunakan untuk aplikasi Multi User (Banyak Pengguna).

Sebagai sebuah program penghasil database, MySQL tidak dapat berjalan sendiri tanpa adanya sebuah aplikasi lain (*interface*). Oleh karena itu harus ada software pendukung antara lain PHP (*Paper Hipertext Preposesor*), Visual Delphi, Visual Basic, Cold Fusion, dan lain-lain.

MySQL memiliki layer utama seperti layer DOS yaitu memiliki prompt utama yang disebut mysql, tetapi sekarang ada suatu program dump yang dibuat seperti web berjalan di bawah server database yang disebut PhpMyAdmin (www.**artic**lecenter.org/**pengertian-mysql**-my-structure-query-language/).

II.5. Unified Modeling Languange (UML)

Notasi UML dibuat sebagai kolaborasi dari Grady Booch, DR.James Rumbaugh, Ivar Jacobson, Rebecca Wirfs-Brock, Peter Yourdon, dan lainnya. Jacobson telah menulis tentang bagaimana mendapatkan persyaratan-persyaratan sistem dalam paket-paket transaksi yang disebut use case. Ia juga mengembangakan sebuah metode untuk perancangan sistem yang disebut *Object Oriented Software Enginnering* (OOSE) yang berfokus pada analisis. Booch, Rumbaugh dan Jacobson biasa disebut dengan tiga sekawan (tree amigos). Ketiganya bekerja di Rational Software Corporation dan fokus pada standarisasi dan perbaikan ulang UML. Simbil-simbol UML mirip dengan Booch, notasi OMT, dan juga ada kemiripan dengan notasi lainnya. Penggabungan beberapa metode menjadi UML dimulai tahun 1993. Setiap orang dari tiga sekawan di rational mulai menggabungkan idenya dengan metode-metode lain yang saat itu ada. Akhir tahun 1995 unified method versi 0.8 diperkenalkan. Unified Method diperbaiki dan diubah menjadi UML pada tahun 1996, UML 1.0 disahkan dan diberikan pada object technology group (OTG) pada tahun 1997, dan pada tahun itu juga beberapa perusahaan pengembang utama perangkat lunak mulai mengadopsinya. Pada tahun yang sama OMG merilis UML 1.1 sebagai standar industri (Sholiq, 2010; 18).

II.5.1 Bagian – Bagian UML

a.Use Case Diagram

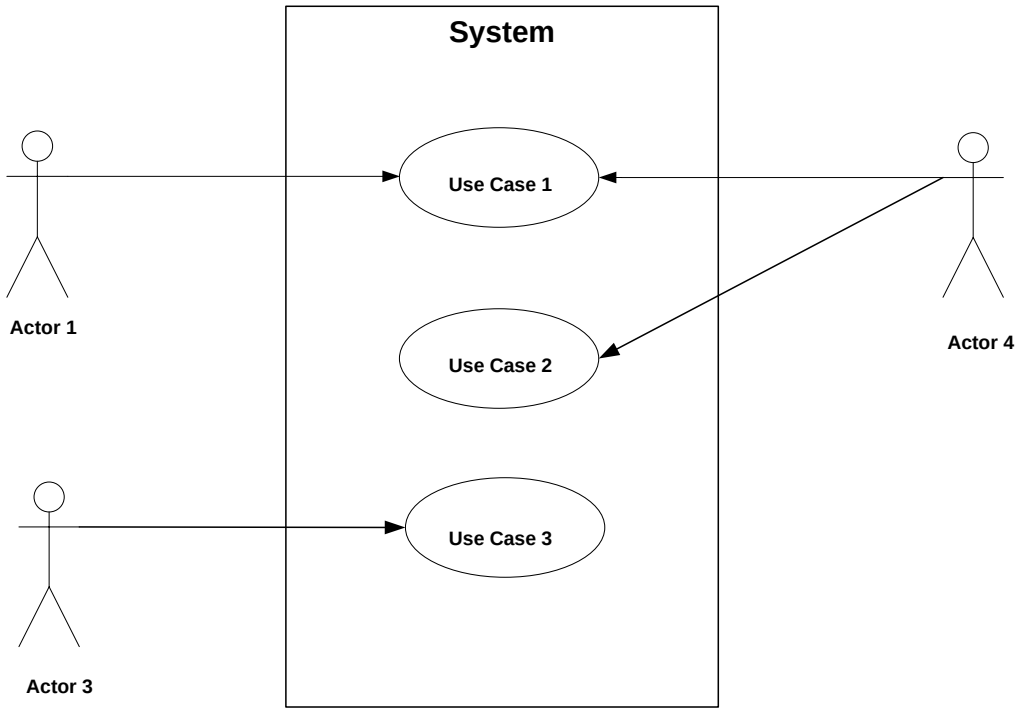
Use case diagram menunjukkan beberapa use case dalam sistem, beberapa aktor dalam sistem,dan relasi antar mereka. Use case sendiri adalah potongan - fungsionalitas tingkat tinggi yang yang disediakan oleh sistem. Aktor adalah seseorang atau sesuatu yang berinteraksi terhadap sistem yang akan dibangun. Dalam sebuah sistem dimungkinkan mempunyai diagram use case lebih dari satu yang merupakan kelompok-kelompok diagram use case yang diorganisasikan dengan tujuan tertentu. Kelompok-kelompok diagram use case dapat dilakukan dengan membuat paket-paket, misalkan paket-paket yang disusun berdasarkan proses bisnisnya, satu proses bisnis yang digambarkan dengan use case bisnis mungkin didukung oleh beberapa use case sebagai prosedur otomatisasi. Paket-paket juga dapat disusun berdasarkan fungsi-fungsi bisnis didalam organisasi, misalnya : Bagian Pemasaran, Bagian Produksi, Bagian Keuangan, Bagian Sumber Daya Manusia, dan lain-lain (Sholiq, 2010; 99).

Manfaat pemodelan use case :

1. Menyediakan tool untuk meng-capture persyaratan fungsional.
2. Membantu menyusun ulang lingkup system menjadi bagian-bagian yang lebih dapat dikelola.
3. Menyediakan alat komunikasi dengan para pengguna dan stakeholder yang berhubungan dengan fungsionalitas sistem. Use case menyajikan bahasa umum yang dapat dipahami oleh berbagai macam stakeholder

4. Memberikan cara bagaimana mengidentifikasi, menetapkan, melacak, dan mengelola pengembangan sistem, terutama pengembangan *incremental* dan *iteratif*.
5. Menyajikan panduan untuk mengistemasi lingkup, usaha, dan jadwal proyek,
6. Menyajikan garis pokok pengujian, khususnya menentukan rencana tes dan test case.
7. Menyajikan garis pokok bagi help sistem dan manual pengguna, dan juga dokumentasi pengembangan sistem.
8. Menyajikan tool untuk melacak persyaratan.
9. Menyajikan titik mulai/awal untuk identifikasi objek data atau entitas.
10. Menyajikan spesifikasi fungsional untuk mendesain antarmuka pengguna dan sistem.
11. Menyajikan alat untuk menentukan persyaratan akses database dalam hal menambah, mengubah, menghapus, dan membaca.
12. Menyajikan kerangka kerja untuk mengarahkan proyek pengembangan sistem (Jeffery L.Whitten dkk ;)

Adapun contoh gambar diagram Use Case Diagram dapat dilihat pada gambar : II.5 Sebagai Berikut:



Gambar II.5 : Contoh Diagram Use case (*Use Case Diagram*)

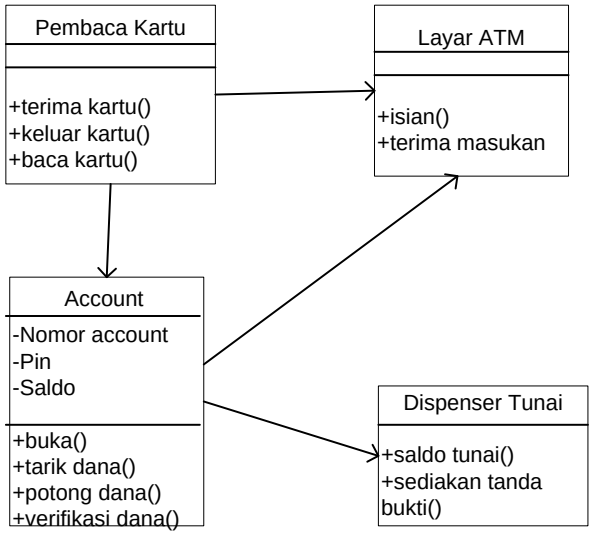
Sumber : Jeffery L.Whitten dkk (2004 ;258)

II.5.2 Class Diagram

Diagram kelas digunakan untuk menampilkan kelas-kelas atau paket-paket dalam sitem dan relasi antar mereka. Biasanya, dibuat beberapa diagram kelas untuk satu sistem. Sebuah kelas mengandung informasi (atribut) dan tingkah laku (behavior) yang berkaitan dengan informasi tersebut. Satu diagram kelas menampilkan subset dari kelas-kelas dan relasinya. Diagram kelas lainnya, mungkin menampilkan kelas-kelas termasuk atribut dan operasi dari kelas-kelas pembentuk diagram. Sedangkan diagram kelas yang lainnya lagi, mungkin menampilkan paket-paket kelas dan relasi antar paket-paket. Diagram kelas adalah alat perancangan terbaik untuk tim pengembang-pengembang perangkat

lunak. Diagram kelas membantu tim pengembang mendapatkan pola kelas-kelas dalam sistem, struktur sistem sebelum menuliskan kode program, dan membantu untuk memastikan bahwa sistem adalah rancangan terbaik dari beberapa alternatif rancangan (Sholiq, 2010; 27; 149-150).

Adapun contoh gambar untuk Diagram Kelas adalah sebagai berikut :



Gambar II.6 : Contoh Diagram Kelas (*Class Diagram*)

Sumber : Jeffery L.Whitten dkk (2004 ;28)

II.5.3 Activity Diagram

Diagram Aktivitas (*Activity Diagram*) menggambarkan aliran fungsionalitas sistem. Ada dua kegunaan diagram aktivitas dalam pemodelan dengan UML. Dua kegunaan tersebut yaitu:

1. Pada tahap pemodelan bisnis, diagram aktivitas dapat digunakan untuk menunjukkan alur kerja bisnis (*business workflow*)

2. Pada tahap pemodelan sistem, diagram aktivitas dapat digunakan untuk menjelaskan aktivitas yang terjadi didalam sebuah *use case*.

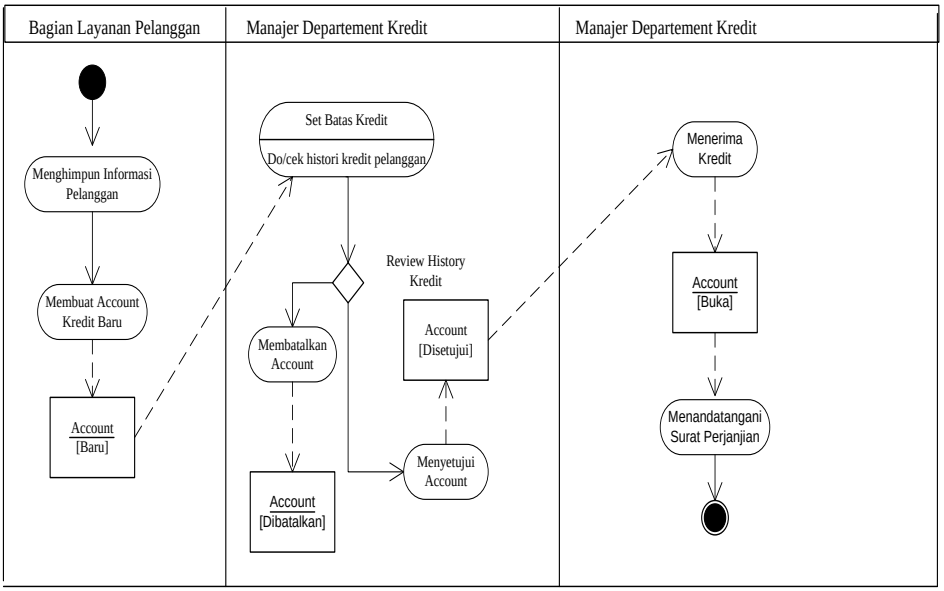
Diagram aktivitas menunjukkan dari mana *workflow* dimulai, dimana *workflow* berakhir, aktivitas apa saja yang terjadi didalam *workflow*, dan apa saja yang dilakukan saat sebuah aktivitas terjadi. Aktivitas adalah tugas yang dilakukan selama dalam *workflow*.

Elemen-elemen utama yang digunakan dalam diagram aktivitas:

1. Swimlanes, menunjukkan siapa yang bertanggung jawab melakukan aktivitas dalam suatu diagram.
2. Aktivitas, adalah kegiatan dalam alur kerja (workflow). Merupakan inti dari dioagram ini. Aktivitas dinyatakan dengan simbol oval.
3. Entitas bisnis, adalah entitas-entitas yang digunakan dalam alur kerja. Entitas bisnis digambarkan dengan simbol persegi panjang atau kotak.
4. Transisi, menunjukkan bagaimana alur kerja itu berjalan dari satu aktivitas ke aktivitas lainnya. Transisi dinyatakan dengan simbol anak panah yang mentrasisikan dari satu aktivitas ke aktivitas lainnya.
5. Titik keputusan, menunjukkan sebuah keputusan perlu dibuat dalam alur kerja. Titik keputusan dinyatakan dengan simbol wajib.
6. Sinkronisasi, menunjukkan dua atau lebih langkah dalam alur kerja berjalan secara serentak. Dua atau lebih aktivitas yang dijalankan secara serentak menggunakan simbol sinkronisasi.

7. Keadaaan awal (*start state*), menunjukkan alur kerja dimulai. Pada satu diagram aktivitas hanha mempunyai satu *start state.* Simbol *start state* menggunakan linkaran kecil dengan bagian dalam bulat terisi warna.
8. Keadaan akhir (*end state*), menunjukkan alur kerja berakhir. Dalam satu diagram aktivitas isa mempunyai beberapa *end state*. Simbol end state dinyatakan dengan notasi lingkarann kecil seperti mata sapi (Sholiq, 2010;22-23;65-66)

Adapun contoh gambar untuk Diagram Aktivitas adalah sebagai berikut dibawah ini :



Gambar II.7 : Contoh Diagram Aktivitas (*Activity Diagram*)

Sumber : Sholiq (2010; 23)

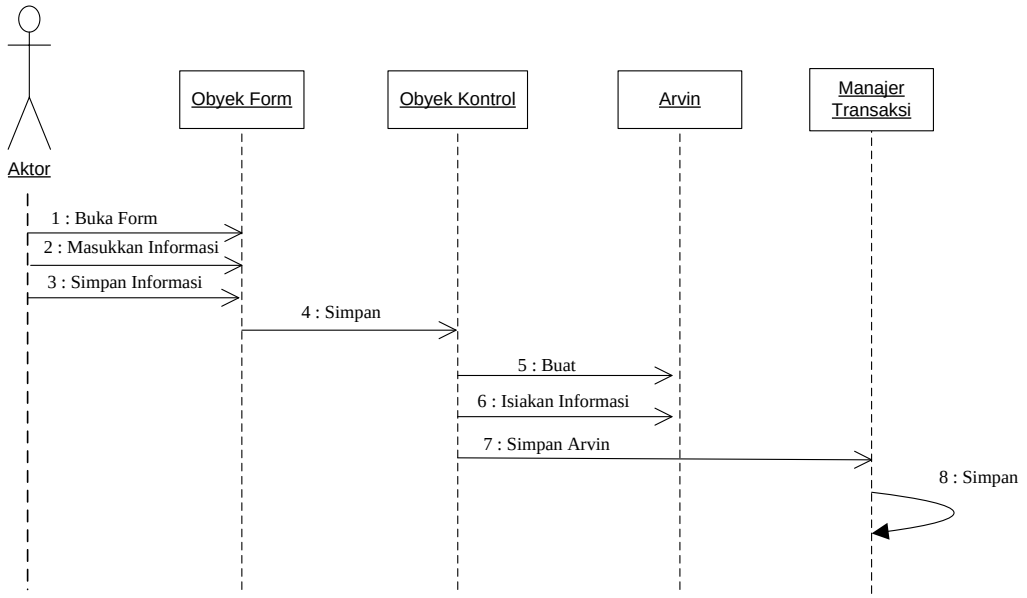
II.5.4 Sequence Diagram (Diagram Sekuensial)

Diagram sekuensial adalah diagram interaksi yang disusun berdasarkan urutan waktu. Diagram sekuensial bermanfaat jika sesorang ingin melakukan review aliran logika dalam sebuah skenario. Diagram sekuensial dibaca dari atas ke bawah. Setiap use case memiliki sejumlah flow (utama dan alternatif). Setiap diagram sekuensial mempresentasikan satu flow dari beberapa flow di dalam use case.

Langkah-langkah yang dilakukan untuk membuat diagram sekuensial adalah :

1. Menentukan obyek-obyek yang terlibat dalam diagram
2. Menentukan aktor
3. Menambahkan beberapa pesan (Sholiq,2010;124 ;127-128)

Adapun contoh gambar untuk Diagram Sekuensial adalah sebagai berikut dibawah ini :



Gambar II.8 : Contoh Diagram Sekuensial (*Sequence Diagram*)

Sumber : Sholiq (2010; 136)

II.6. Entity Relationship Diagram (ERD)

Model sistem memainkan peran penting dalam pengembangan sistem. Pemodelan data merupakan teknik untuk mendefenisikan persyaratan bisnis untuk sebuah database, yaitu mengatur dan mendokumentasikan data sistem. Ada beberapa catatan mengenai pemodelan data. Model yang aktual disebut entity relationship diagram (ERD). Model ini menjelaskan data dalam konteks entitas dan hubungan yang digambarkan oleh data tersebut yang menggunakan beberapa notasi untuk mneggambarkan data dalam konteks entitas dan hubungan yang dideskripsikan oleh data tersebut (Jeffery L.Whitten, Lonnie D.Bentley, Kevin C.Dittman, 2004 ; 280-281).

II.7. Normalisasi

Model data mengkomunikasikan secara efektif persyaratan database, namun model tersebut tidak selalu merupakan desain database yang bagus. Model tersebut dapat megandung karakteristik struktural yang mengurangi fleksibilitas dan pengembangan atau menghasilkan redundansi yang tidak perlu. Model data yang bagus adalah sederhana, nonredundan, fleksibel dan mudah menyesuaikan dengan kebutuhan di masa datang. Teknik yang digunakan untuk meningkatkan model data untuk menyiapkan desain database disebut analisis data. Analisis data adalah proses yang mempersiapkan model data untuk implementasi database yang sederhana, redundan, fleksibel dan mudah beradaptasi. Teknik yang spesifik disebut normalisasi. Normalisasi adalah teknik analisis data yang mengatur atribut data dalam kelompok untuk membentuk entitas yang nonrendundan, stabil, fleksibel, dan mudah beradaptasi.. Normalisasi merupakan teknik tiga langkah yang menempatkan model data menjadi first normal form (1NF), second normal form (2NF), dan third normal form (3NF).

1. First Normal Form (1NF)

Pada *first normal form* (1NF) tidak ada atribut yang dapat memiliki lebih dari satu nilai untuk contoh entitas tunggal. Atribut yang dapat memiliki banyak nilai sebenarnya mendeskipsikan entittas terpisah, mungkin sebuah entitas dan hubungan.

2. First Normal Form (2NF)

Pada *second normal form* (2NF), entitas harus sudah berada dalam 1NF dan jika nilai semua atribut nonprimary-key tergantung pada primary key lengkap,

bukan hanya sebagian. Atribut nonkey yang hanya tergantung pada sebagian primary key seharusnya dipindahkan ke entitas lain dimana partial key tersebut sebenarnya merupakan full key. Mungkin pada model tersebut harus dibuat entitas dan hubungan baru.

3. Third Normal Form (3NF)

Pada *third normal form* (3NF), entitas harus berada dalam 2NF dan jika nilai atribut non-primary key nya tidak tergantung pada atribut nonprimary key lainnya.Atribut nonkey yang tergantung pada atribut nonkey lainnya harus dipindahkan atau dihapus. Sekali lagi, entitas dan hubungan baru mungkin harus ditambahkan ke model data (Jeffery L.Whitten, Lonnie D.Bentley, Kevin C.Dittman, 2004 ; 518-519; 306-307).