BAB II

TINJAUAN PUSTAKA

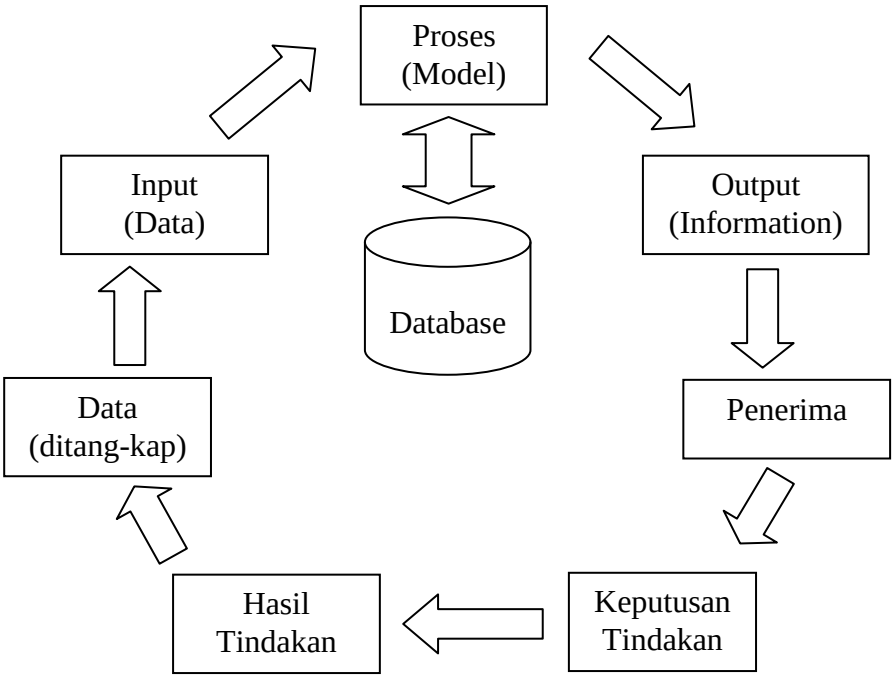
II.1. Pengertian Sistem

Dengan berbagai pendekatan, beragam pula istilah "sistem" didefinisikan. Sistem adalah suatu pengorganisasian yang saling berinteraksi, saling bergantung dan terintegrasi dalam kesatuan variabel atau komponen. Terdapat dua kelompok pendekatan sistem, yaitu menekankan pada prosedur dan komponen atau elemennya. Pendekatan sistem yang lebih menekankan pada prosedur mendefinisikan sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkelompok dan bekerjasama untuk melakukan kegiatan pencapaian sasaran tertentu. Makna dari prosedur sendiri, yaitu urutan yang tepat dari tahapan-tahapan instruksi. Sedangkan pendekatan yang menekankan pada komponen mendefinisikan sistem sebagai kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu. *"Serangkaian atau tatanan elemen-elemen yang diatur untuk mencapai tujuan yang ditentukan sebelumnya melalui pemrosesan informasi"* (Riyanto, dkk; 2009 : 21-22).

II.2. Data Dan Informasi

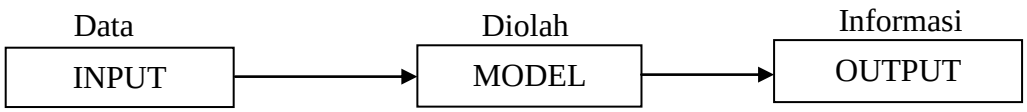
Data merupakan bentuk yang masih mentah yang belum dapat bercerita banyak, sehingga perlu diolah lebih lanjut. Data diolah melalui model tertentu menjadi informasi yang dapat dimanfaatkan oleh penerima dalam membuat keputusan dan melakukan tindakan, yang berarti melakukan suatu tindakan lain

yang akan membuat sejumlah data kembali. Data yang masih belum diolah akan disimpan dalam bentuk *database*. Data yang disimpan ini nantinya dapat diambil kembali untuk diolah kembali menjadi informasi. Data tersebut akan ditangkap sebagai input, diproses kembali lewat suatu model tertentu dan seterusnya membentuk suatu siklus. Siklus ini oleh John Burch disebut dengan siklus informasi (*information cycle*) (Riyanto, dkk. : 2009 : 24),



Gambar II.1. Siklus Informasi
(Sumber : Riyanto, dkk. : 2009 : 24)

Agar menjadi informasi yang berguna, data perlu diolah melalui sebuah siklus. Siklus ini disebut siklus pengolahan data (*data processing life cycle*) (Riyanto, dkk. : 2009 : 23).



Gambar II.2. Siklus Pengolahan Data
(Sumber : Riyanto, dkk. : 2009 : 23)

II.3. Sistem Informasi Geografis

Sistem Informasi Geografis (SIG) adalah sistem informasi khusus yang mengelola data yang memiliki informasi spasial (bereferensi keruangan). Atau dalam arti sempit, adalah sistem komputer yang memiliki kemampuan untuk membangun, menyimpan, mengelola dan menampilkan informasi bereferensi geografis, misalnya data yang diidentifikasi menurut lokasinya, dalam sebuah *database*. Beberapa definisi dari SIG adalah sebagai berikut :

1. Aronoff menyatakan bahwa SIG sebagai suatu sistem ber*basis*kan komputer yang digunakan untuk menyimpan dan memanipulasi informasi-informasi geografis. SIG dirancang untuk mengumpulkan, menyimpan, dan menganalisis objek-objek dan fenomena dimana lokasi geografi merupakan karakteristik yang penting atau kritis untuk dianalisis.
2. Subaryono menyatakan bahwa SIG sebagai suatu himpunan terpadu dari *hardware*, *software*, *data*, dan *liveware* (orang-orang yang bertanggung jawab dalam mendesain, mengimplementasikan, dan menggunakan SIG).
3. ESRI (*Environmental System Research Institute*) menyatakan bahwa SIG adalah kumpulan yang terorganisir dari perangkat keras komputer, perangkat lunak, data geografis dan personil yang dirancang secara efisien untuk memperoleh, menyimpan, meng*update*, memanipulasi, menganalisis, dan menampilkan semua bentuk informasi yang bereferensi geografi (Riyanto, dkk. : 2009 : 35-36).

Dari definisi tersebut dapat diambil kesimpulan bahwa SIG terdiri atas beberapa subsistem. Subsistem tersebut adalah sebagai berikut :

1. *Input*

   Pada tahap *input* (pemasukan data) yang dilakukan adalah mengumpulkan dan mempersiapkan data spasial dan data atribut dari berbagai sumber data. Data yang digunakan harus dikonversikan menjadi format *digital* yang sesuai. Proses konversi yang dilakukan dikenal dengan proses dijitalisasi (*digitizing*).

2. Manipulasi

   Manipulasi data merupakan proses *editing* terhadap data yang telah masuk, hal ini dilakukan untuk menyesuaikan tipe dan jenis data agar sesuai dengan sistem yang akan dibuat, seperti : penyamaan skala, pengubahan sistem, proyeksi, generalisasi dan sebagainya.

3. Manajemen data

   Tahap ini meliputi seluruh aktifitas yang berhubungan dengan pengolahan data (menyimpan, mengorganisasi, mengelola, dan menganalisis data) ke dalam sistem penyimpanan permanen, seperti : sistem *file server* atau *database server* sesuai kebutuhan sistem. Jika menggunakan sistem *file server*, data disimpan dalam bentuk *file-file* seperti : *.txt, *.dat, dan lain-lain. Sedangkan jika menggunakan sistem *database server*, biasanya memanfaatkan *software Database Management System (DBMS),* seperti : *MySQL, SQL Server, ORACLE,* dan *DBMS* sejenis lainnya.

4. *Query*

   Suatu metode pencarian informasi untuk menjawab pertanyaan yang diajukan oleh pengguna SIG. Pada SIG dengan sistem *file server*, *query* dapat dimanfaatkan dengan bantuan *compiler* atau *interpreter* yang digunakan dalam mengembangkan sistem, sedangkan untuk SIG dengan sistem *database server*, dapat memanfaatkan *SQL* (*structured query language*) yang terdapat pada *DBMS* yang digunakan. Penelusuran data menggunakan lebih dari satu *layer* dapat memberikan informasi untuk analisis data dan memperoleh data yang diinginkan.

5. Analisis

   Terdapat dua jenis fungsi analisis dalam SIG, yaitu fungsi analisis spasial dan analisis atribut. Fungsi analisis spasial adalah operasi yang dilakukan pada data spasial. Sedangkan, Fungsi analisis atribut adalah fungsi pengolahan data atribut, yaitu data yang tidak berhubungan dengan ruang.

6. Visualisasi (*Data Output*)

   Penyajian hasil berupa informasi baru atau *database* yang ada baik dalam bentuk *softcopy* maupun dalam bentuk *hardcopy* seperti dalam bentuk peta : peta (atribut peta dan atribut data), tabel, grafik, dan lain-lain (Riyanto, dkk : 2009 : 35-38).

Data *digital* geografis diorganisir menjadi dua bagian, yaitu Data Spasial dan Data Atribut. Definisi dari kedua bagian tersebut adalah sebagai berikut :

1. Data Spasial Raster

   Data spasial diperoleh dari peta *hard copy*, foto udara citra satelit, peta digital dan lainnya. Data spasial disini adalah data berupa gambar yang berhubungan dengan lokasi atau posisi, bentuk dan hubungan antar unsurnya. Pemasukan data vektor dilakukan dengan pendigitasian, sedangkan data raster dilakukan dengan scanning.

   Bentuk data spasial, terdiri dari :

   a. Titik, dengan format: sepanjang koordinat (x,y) yang tidak mempunyai dimensi panjang dan luas.
   b. Garis, dengan format: kumpulan pasangan koordinat yang mempunyai titik awal dan titik akhir, serta mempunyai dimensi panjang tetapi tidak mempunyai dimensi luas.
   c. Poligon/Area, dengan format : kumpulan pasanganpasangan koordinat yang mempunyai titik awal dan titik akhir, dimana titik awal dan titik akhir berhimpit atau sama serta mempunyai dimensi panjang dan luas.

2. Data Atribut

   Data atribut adalah suatu informasi dari suatu informasi dari suatu data grafis (titik, garis, ataupun area) yang disimpan dalam format data tabular.

   Data atribut terdiri dari :

   a. Formulir dan daftar, dengan format : kode *alfabetic*, kode *alfanumeric* dan angka.
   b. Laporan lengkap, dengan format : Kata, kalimat dan keterangan lain.

c. Keterangan gambar (*grafic chart*), dengan format : kata, angka, keterangan penunjuk, liputan area, keterangan simbol.(Runi Asmaranto : 2008 : 16)

II.4. ArcView

Kemampuan *Arcview* GIS pada berbagai serinya tidaklah diragukan lagi. *Arcview* GIS adalah software yang dikeluarkan oleh ESRI *(Environmental Systems Research Institute)*. Perangkat lunak ini memberikan fasilitas teknis yang berkaitan dengan pengolahan data spasial. Kemampuan grafis yang baik dan kemampuan teknis dalam pengolahan data spasial tersebut memberikan kekuatan secara nyata pada *Arcview* untuk melakukan analisis spasial. Kekuatan analisis inilah yang pada akhirnya menjadikan *Arcview* banyak diterapkan dalam berbagai pekerjaan, seperti analisis pemasaran, perencanaan wilayah dan tata ruang, sistem informasi persis, pengendalian dampak lingkungan, bahkan untuk keperluan militer. Mengapa *Arcview* dapat memiliki keluwesan yang sedemikian hebat? Hal itu disebabkan oleh adanya dukungan dari skrip *Avenue*. Melalui *avenue* ini dapat dibentuk suatu "kemampuan baru" pada *Arcview*. Tentu saja hal ini membuat Arcview menjadi sangat luwes untuk diterapkan pada berbagai permasalahan spasial. *Avenue* dapat digunakan untuk "merombak" wajah *Arcview* sesuai kebutuhan penggunaanya.

Dialog designer diperlukan untuk membentuk antarmuka penampil data atribut yang menjadi dasar pemilihan objek. Untuk menghubungkan menu dan tombol dengan berbagai aksi yang diinginkan maka perlu dibentuk skrip atau

program. Skrip atau program ini dibentuk menggunakan bahasa Avenue. Setiap aksi yang diperlukan diuraikan menjadi baris-baris perintah pada skrip Avenue dan selanjutnya dikaitkan ke masing-masing menu atau tombol yang bersangkutan (Eko Budiyanto ; 2010 : 177-178).

II.5. Pengertian UML

UML singkatan dari *Unified Modelling Language* yang berarti bahasa permodelan standar. UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk :

1. Merancang perangkat lunak
2. Sarana Komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasi sistem yang ada, proses-proses dan organisasinya (Prabowo Pudjo Widodo dan Herlawati : 2011 : 6-7).

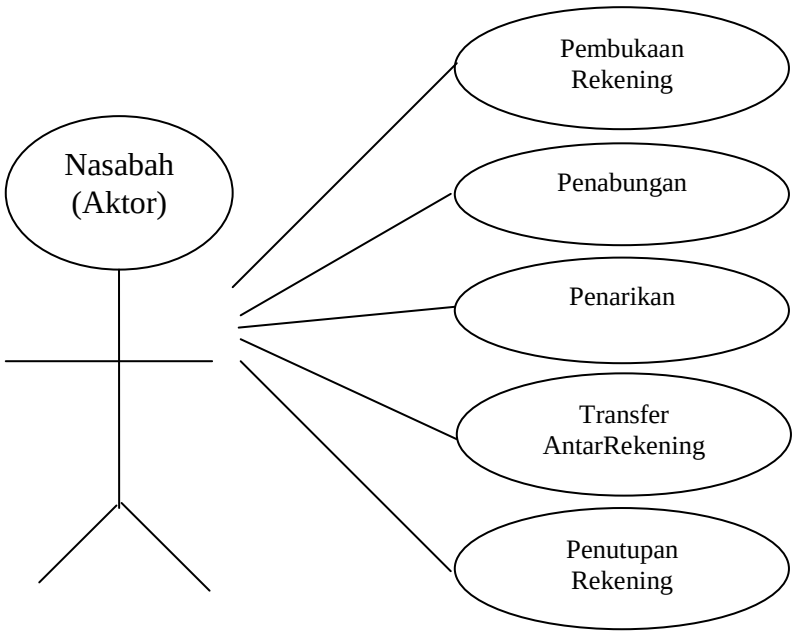
II.5.1. Use Case Diagram

Segala sesuatu yang secara akademis dikembangkan pada umumnya berawal dari suatu konsep. Demikian juga halnya dengan pengembangan sistem pada umumnya dikembangkan berdasarkan analisis kebutuhan. Analisis kebutuhan ini adalah tahap konseptualisasi, yaitu suatu tahap yang mengharuskan analis dan perancang sistem untuk berusaha tahu secara pasti mengenai hal yang menjadi kebutuhan dan harapan pengguna sehingga kelak aplikasi yang dibuat

memang akan digunakan oleh pengguna (*user*) serta akan memuaskan kebutuhan dan harapannya.

Dalam konteks UML, tahap konseptualisasi dilakukan dengan pembuatan *use case diagram* yang sesungguhnya merupakan deskripsi peringkat tinggi bagaimana perangkat lunak (aplikasi) akan digunakan oleh penggunanya. Selanjutnya, *use case diagram* tidak hanya sangat penting pada saat analisis, tetapi juga sangat penting dalam tahap perancangan (*design*), untuk mencari kelas-kelas yang terlibat alam aplikasi, dan untuk melakukan pengujian (*testing*).

Saat akan mengembangkan *use case diagram*, hal yang pertama kali harus dilakukan adalah mengenali *actor* untuk sistem yang sedang dikembangkan. Dalam hal ini, ada beberapa karakteristik untuk para *actor*, yaitu *actor* yang ada di luar sistem yang sedang dikembangkan dan *actor* yang berinteraksi dengan sistem yang sedang dikembangkan (Adi Nugroho ; 2009 : 7)



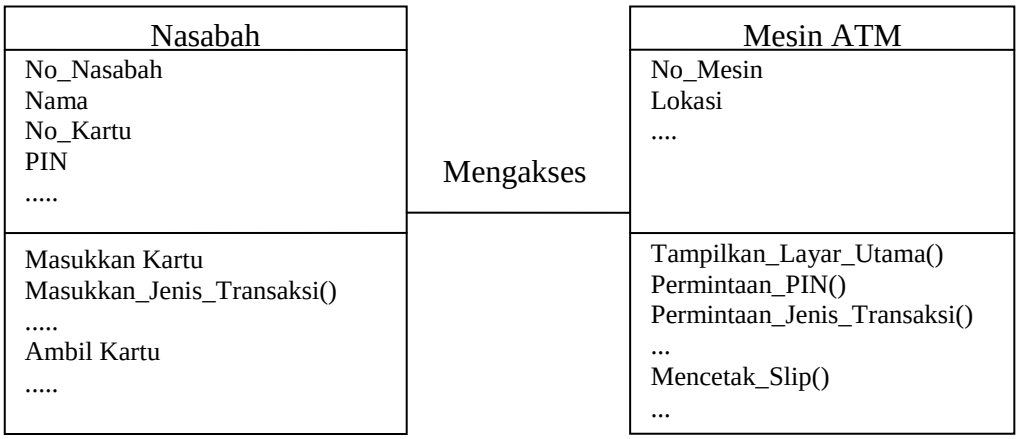
Gambar II.3. Contoh Use Case Diagram
(Sumber : Adi Nugroho ; 2009 : 8)

II.5.2. Class Diagram

Class didefenisikan sebagai kumpulan/himpunan objek yang memiliki kesamaan dalam atribut/properti, perilaku (operasi), serta cara berhubungan dengain objek lain (Adi Nugroho ; 2009 : 18).

Selain itu, kita juga mendefenisikan objek sebagai konsep, abstraksi dari sesuatu dengan batas nyata, sehingga kita dapat menggambarkan secara sistematis. Pemahaman objek memiliki dua fungsi, yaitu :

1. Memudahkan untuk mempelajari secara seksama hal-hal yang ada di dunia nyata.
2. Menyediakan suatu dasar yang kuat dalam implementasi ke dalam sistem terkomputerisasi (Adi Nugroho ; 2009 :17).



Gambar II.4. Contoh Class Diagram
(Sumber : Adi Nugroho ; 2009: 39)

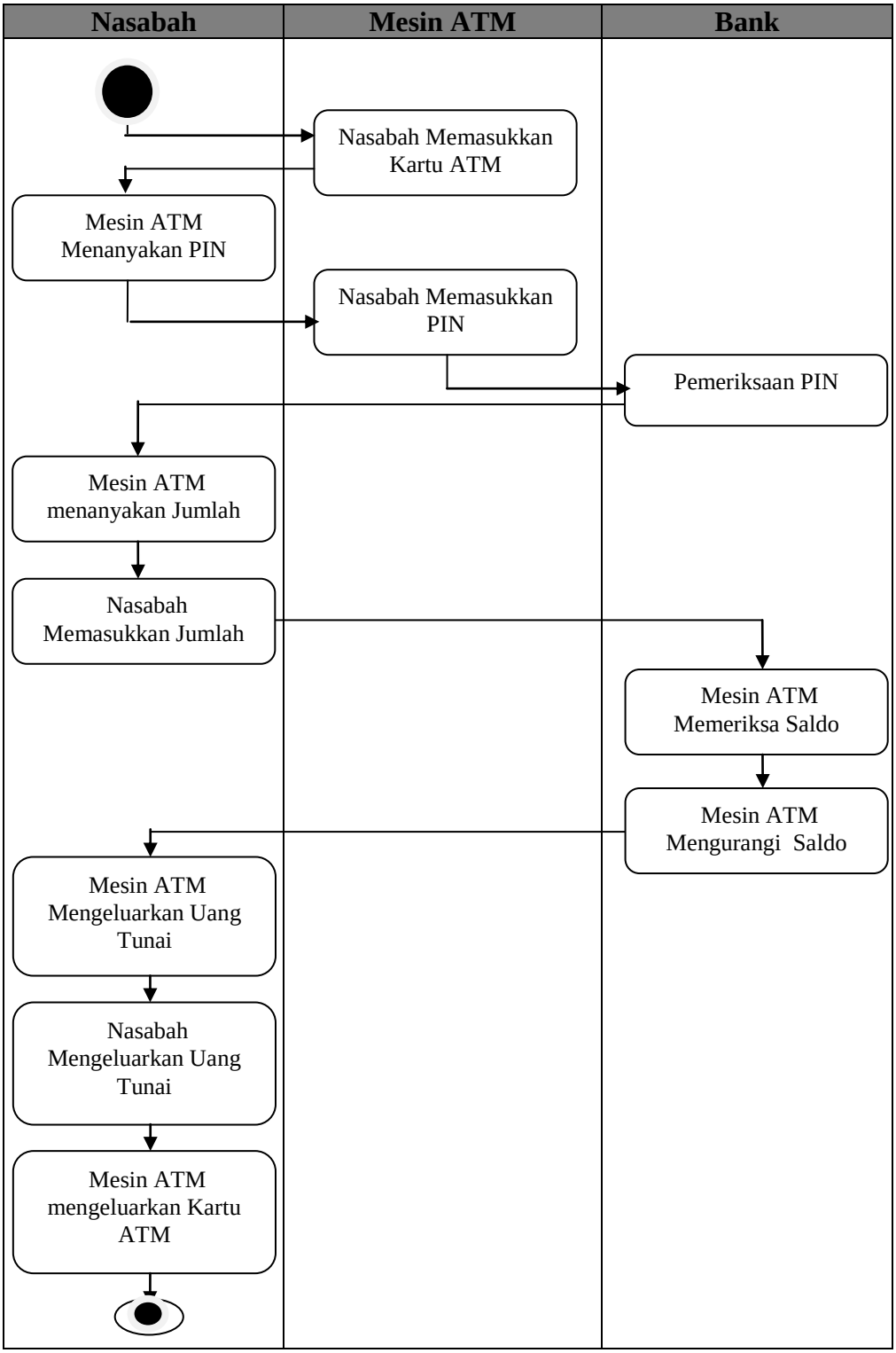
II.5.3. Activity Diagram

Diagram aktivitas atau *activity diagram* menggambarkan aliran fungsionalitas sistem. Pada tahap pemodelan bisnis, diagram aktivitas dapat digunakan untuk menujukkan aliran kerja bisnis *(business work flow).* Dapat juga

digunakan untuk menggambarkan aliran kejadian *(flow of event)* dalam *use case*.

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *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. Sama seperti *state* , standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis *horizontal* atau vertikal. *Activity diagram* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu (Adi Nugroho ; 2009 : 13).



Gambar II.5. Contoh Activity Diagram
(Sumber : Adi Nugroho ; 2009 : 11)

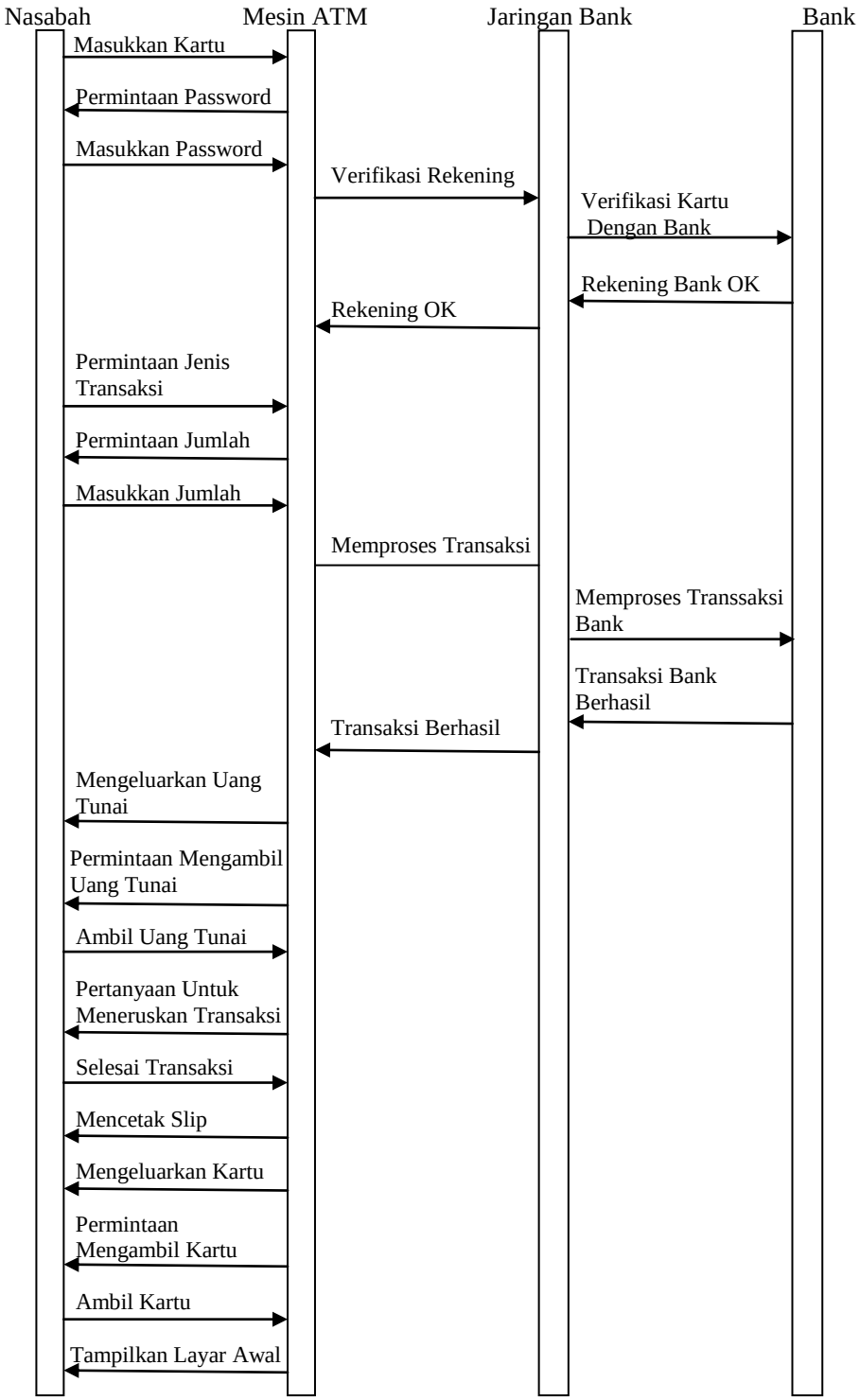
II.5.4. Sequence Diagram

Diagram sekuensial atau *sequence diagram* digunakan untuk menunjukkan aliran fungsionalitas dalam *use case*. Diagram sekuensial adalah diagram yang disusun berdasarkan urutan waktu. Kita membaca diagram sekuensial dari atas ke bawah. Setiap diagram sekuensial mempresentasikan suatu aliran dari beberapa aliran di dalam *use case*.

Jadi dengan kata lain sekuensial diagram menunjukkan aliran fungsionalitas berdasarkan urutan waktu serta kejadian yang nantinya akan menetukan metode/fungsi atribut masing-masing. Dimana fungsi-fungsi tersebut akan diterapkan pada suatu kelas/objek.

Perhatikan gambar II.6. dimana terlihat pengelompokkan *event-event* serta fungsi masing-masing atribut tersebut. Di dalam diagram terlihat jelas bagaimana aliran suatu proses kejadian dimana seorang nasabah yang akan melakukan transaksi dengan sebuah mesin ATM. Dari diagram tersebut kita mengetahui *event-event* yang terjadi, seperti : Nasabah memasukkan kartu ATM, Mesin ATM merespon dengan meminta *password* atau PIN, dan selanjutnya.

Kita dapat melihat setiap fungsi atribut dan *event-event* apa saja yang terjadi. Sehingga melalui diagram sekuensial ini kita dapat merancang suatu program aplikasi yang baik, sehingga dalam menghadapi sebuah kasus yang benar-benar kompleks diagram sekuensial ini sangat membantu.



Gambar II.6. Contoh Sequence Diagram
(Sumber : Adi Nugroho ; 2009 : 36)

II.6. Desain Database

Desain database merupakan pekerjaan yang penting dalam pembuatan atau pengembangan sistem, karena desain *database* akan mendapatkan susunan data atau *table* yang efektif dan efisien. Alat desain *database* yang popular ada dua, yaitu : ERD (*Entity Relationship Diagram*) dan Normalisasi. Jika memakai *Normalisasi* harus mendapatkan Data Dasar (Dokumen Dasar), sedangkan ERD tidak perlu. Dalam desain ERD terbagi dua tahapan yaitu: *Preliminary* Desain (Disain Awal) dan *Final Design* (Disain Akhir). Tetapi disain Akhir dari ERD juga berisi Normalisasi (Yuniar Supardi : 2008 : 9)

II.7. Basis Data

Kata "*basis data*" bisa digunakan untuk menguraikan segala sesuatu dari sekumpulan data tunggal, seperti daftar telepon. *Basis data* terdiri dari *file-file* fisik yang ditetapkan berdasarkan komputer saat menerapkan perangkat lunak basis data. Sedangkan menurut *Stephens* dan *Plew* (Janner Simarmata : 2007 : 1)

Basis Data adalah mekanisme yang digunakan untuk menyimpan informasi atau data. Informasi adalah sesuatu yang kita gunakan sehari-hari untuk berbagai alasan. Dengan basis data, pengguna dapat menyimpan data secara terorganisasi. Setelah data disimpan, informasi harus mudah diambil. Kriteria dapat digunakan untuk mengambil informasi. Cara data disimpan dalam basis data menentukan seberapa mudah mencari informasi berdasarkan banyak kriteria. Data pun harus mudah ditambahkan kedalam basis data, dimodifikasi, dan dihapus (Janner Simarmata dan Iman Paryudi ; 2007 : 1).

II.8. *Entity Relationship Diagram* (ERD)

Menurut Janer Simarmata dan Imam Prayudi (2007:59) Struktur yang mendasari suatu basisdata adalah model data yang merupakan kumpulan alat-alat konseptual untuk mendeskripsikan data, relasi data, data *semantik*, dan batasan konsistensi. Untuk mengilustrasikan konsep model data dapat disajikan dengan *entity relationalship* model. *Entity relationalship* mendeskripsikan rancangan basisdata pada tingkatan *logis*. *Entity relationship* (ER) data model didasarkan pada persepsi terhadap dunia nyata yang tersusun atas kumpulan objek-objek dasar yang disebut *entitas* dan hubungan antar obyek. Entitas adalah sesuatu atau objek dalam dunia nyata yang dapat dibedakan dari objek lain (Janer Simarmata dan Imam Prayudi ; 2007 : 59).

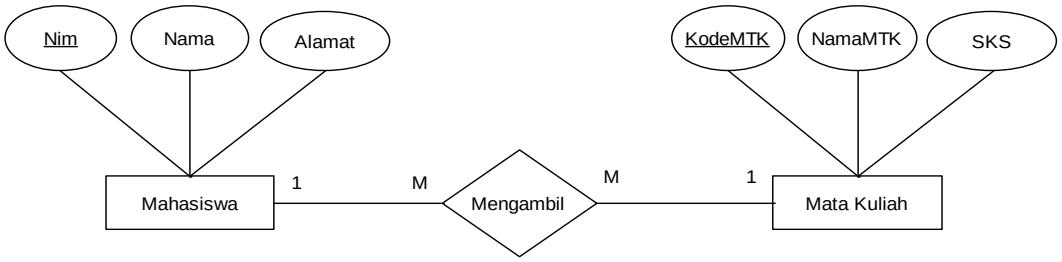
Struktur *logis* (skema *database* dapat ditunjukan secara *grafis* dengan ER yang dibentuk dari komponen-komponen dapat dilihat pada tabel II.1. berikut ini :

Tabel II.1. Komponen-komponen Entity Relationship

Simbol	Arti
Entitas	Persegi panjang mewakili kumpulan entitas.
Attribute	Elips mewakili attribute
Relasi	Belah ketupat mewakili relasi
———	Garis menghubungkan atribut dengan kumpulan entitas dengan relasi.

(Sumber : Janer Simarmata dan Imam Prayudi ; 2007 : 60)

Sebagai ilustrasi, bayangan anda mengambil bagian sistem basis data universitas yang terdiri dari mahasiswa dan mata kuliah. Gambar II.7. menunjukkan ER Diagram dari contoh. Diagram menunjukkan bahwa ada dua kumpulan entitas, yaitu mahasiswa dan mata kuliah, dan bahwa relasi mengambil mahasiswa dan mata kuliah.



Gambar II.7. Diagram ER
(Sumber : Janer Simarmata dan Imam Prayudi ; 2007 : 60)

II.9. Normalisasi

Proses normalisasi merupakan proses pengelompokan data ke dalam bentuk tabel atau relasi atau file untuk menyatakan entitas dan hubungan sehingga terwujud satu bentuk basis data yang mudah dimodifikasi (Janner Simarmata : 2007: 197).

Terdapat beberapa langkah Normalisasi diantaranya :

1. Bentuk Normal Pertama (1NF)

   Sebuah tabel relasional secara definisi selalu berada dalam bentuk normal pertama. Semua nilai pada kolom-kolomnya adalah *atomik*. Ini berarti kolom-kolom tidak mempunyai nilai berulang.

2. Bentuk Normal Kedua (2NF)

   Sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada 1NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada kunci utama. Ini berarti bahwa setiap kolom bukan kunci harus tergantung pada seluruh kolom yang membentuk kunci utama.

3. Bentuk Normal Ketiga (3NF)

   Secara definisi, sebuah tabel berada pada bentuk normal ketiga (3NF) jika tabel sudah berada pada 2NF dan setiap kolom yang bukan kunci tidak tergantung secara transitif pada kunci utamanya. Dengan kata lain, semua atribut bukan kunci tergantung secara fungsional hanya pada kunci utama.

4. Bentuk Normal Boyce-Code (BCNF)

   Setelah 3NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Bentuk Normal Boyce-Code (BCNF) adalah versi 3NF yang lebih teliti dan berhubungan dengan tabel relasional yang mempunyai banyak kunci kandidat, kunci kandidat gabungan, dan kunci kandidat yang saling tumpang tindih (Janner Simarmata dan Iman Paryudi : 2007 : 79-84).