BAB II

LANDASAN TEORI

II.1. Sistem Pendukung Keputusan / *Decision Support System* (DSS)

Konsep Sistem Pendukung Keputusan (SPK) pertama kali diungkapkan pada awal tahun 1970-an oleh Michael S.Scott Morton dengan istilah Manajemen *Decision Systems*. Menurut Raymond McLeod, Jr, Sistem Pendukung Keputusan adalah sistem penghasil informasi spesifik yang ditujukan untuk memecahkan masalah tertentu yang harus dipecahkan oleh manajer pada berbagai tingkatan. Man dan Watson mendefinisikan SPK sebagai suatu sistim interaktif, yang membantu pengambil keputusan melalui penggunaan data dan model-model keputusan untuk memecahkan masalah-masalah yang sifatnya semi terstruktur dan tidak terstruktur (Anastasya Latubessy, dkk ; Vol. 3 ; No. 1 ; 2013 : 80).

Dari defenisi di atas, terlihat bahwa SPK adalah suatu sistim berbasis komputer yang dirancang untuk meningkatkan efektifitas pengambil keputusan dalam memecahkan masalah yang sifatnya semi terstruktur atau tidak terstruktur. Kata berbasis komputer merupakan kata kunci, karena hampir tidak mungkin membangun SPK tanpa memanfaatkan komputer sebagai alat bantu, terutama untuk menyimpan data serta mengelola model. Kehadiran komputer sebagai perangkat teknologi yang mengotomatisasi sistem yang sedang berjalan menjadi daya tarik tersendiri di berbagai bidang. Sehingga semakin marak pengembangan teknologi diterapkan di berbagai bidang, seperti halnya pada bidang telekomunikasi (Anastasya Latubessy, dkk ; Vol. 3 ; No. 1 ; 2013 : 80).

Dengan pengertian diatas dapat dijelaskan bahwa sistem pendukung keputusan bukan merupakan alat pengambilan keputusan, melainkan merupakan sistem yang membantu pengambil keputusan dengan melengkapi mereka dengan informasi dari data yang telah diolah dengan relevan dan diperlukan untuk membuat keputusan tentang suatu masalah dengan lebih cepat dan akurat. Sehingga sistem ini tidak dimaksudkan untuk menggantikan pengambilan keputusan dalam proses pembuatan keputusan (Rika Yunitarini ; Vol. 1 ; No. 1 ; 2013 : 46).

Sistem Pendukung Keputusan adalah suatu sistem informasi berbasis komputer yang melakukan pendekatan untuk menghasilkan berbagai alternatif keputusan untuk membantu pihak tertentu dalam menangani permasalahan dengan menggunakan data dan model. Pengambilan keputusan merupakan hasil suatu proses pemilihan dari berbagai alternatif tindakan yang mungkin dipilih dengan mekanisme tertentu, dengan tujuan untuk menghasilkan keputusan yang terbaik (Nency Nurjannah ; Vol. 10 ; No. 2 ; 2015 : 20).

II.2.1. Karakteristik Sistem Pendukung Keputusan

Konsep Sistem Pendukung Keputusan pertama kali diperkenalkan pada tahun 1970-an oleh Michael S.Scott Morton dengan istilah *Management Decision Model* (Sprague, 1982). Konsep sistem pendukung keputusan ditandai dengan sistem interaktif berbasis komputer yang membantu membentuk keputusan, memanfaatkan data dan model untuk menyelasaiakn masalah-masalah yang tidak tersetruktur. Pada dasarnya sistem pendukung keputusan dirancang untuk

mendukung seluruh tahap pengambilan keputusan mulai dari mengidentifikasi masalah, memilih data yang relevan, menentukan pendekatan yang digunakan dalam pengambilan keputusan, sampai mengevaluasi pemilihan interaktif. Peranan sistem pendukung keputusan dalam konteks keseluruhan sistem informasi ditujukan untuk memperbaiki kinerja melalui aplikasi teknologi informasi. Terdapat sepuluh karakteristik dasar sistem pendukung keputusan yang efektif. (Maulida Indapuri ; Vol. 6 ; No. 2 ; 2014 : 86).

1. Mendukung proses pengambilan keptusan, menitikberatkan pada *management by perceptio.*
2. Adanya *interface* manusia atau mesin di mana manusia *(user)* tetap mengontrol proses pengambilan keputusan.
3. Menggunakan model-model matematis dan statistik yang sesuai.
4. Memiliki kapabilitas dialog untuk memperoleh informasi sesuai dengan kebutuhan model interaktif.
5. Output ditunjukkan untuk personil organisasi dalam semua tingkatan.
6. Memiliki subsistem-subsitem yang terintegrasi sedemikian rupa sehingga dapat berfungsi sebagai kesatuan sistem.
7. Membutuhkan struktur data komprehensif yang dapat melayani kebutuhan informasi keseluruhan tingkatan manajemen.
8. Pendekatan *easy to use.* Ciri suatu sistem pendukung keputusan yang efektif adalah kemudahan untuk digunakan dan memungkinkan keseluruhan keleluasaan pemakai untuk memilih atau

mengembangkan pendekatan-pendekatan baru dalam membahas masalah yang dihadapi.

9. Kemampuan sistem beradaptasi secara tepat, di mana pengambil keputusan dapat menghadapi menangani dengan cara mengadaptasi sistem terhadap kondisi-kondisi dan perubahan yang terjadi.
10. Mendukung pengambilan keputusan untuk membahas masalah-masalah terstruktur, semiterstruktur dan tidak terstruktur.

II.2.1. Komponen Sistem Pendukung Keputusan

Untuk dapat menerapkan SPK, ada 4 komponen subsistem yang harus disediakan yaitu (Syukron Hidayat, dkk ; Vol. 4 ; No. 2 ; 2015 : 44) :

1. Subsistem manajemen data

   Subsistem ini menyediakan data bagi sistem, termasuk didalamnya basis data. Berisi data yang relevan untuk situasi dan diatur oleh perangkat lunak yang disebut *Database Management System* (DBMS).

2. Subsistem manajemen model

   Subsistem ini berfungsi sebagai pengelola berbagai model, mulai dari model keuangan, statistik, matematik, atau model kuantitatif lainnya yang memiliki kemampuan analisis dan manajemen perangkat lunak yang sesuai. Perangkat lunak ini sering disebut *Model Base Management System* (MBMS).

3. Subsistem manajemen pengetahuan

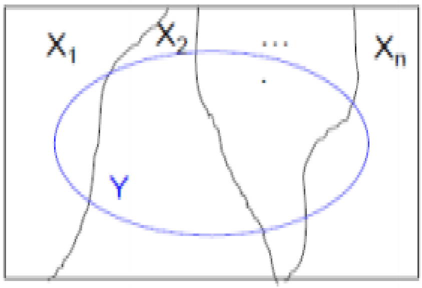
Subsistem ini mendukung berbagai subsistem lainnya, atau dapat dikatakan berperan sebagai komponen yang independen. Subsistem ini menyediakan intelegensi untuk menambah pertimbangan pengambil keputusan.

4. Subsistem manajemen antar muka pengguna

   Subsistem ini berupa tampilan yang disediakan yang mampu mengintegrasikan sistem terpasang dengan pengguna secara interaktif. Melalui subsistem ini pengguna dapat berkomunikasi dengan sistem pendukung keputusan serta memerintah sistem pendukung keputusan.

II.3. Metode *Bayes*

Metoda *Bayes* merupakan teknik yang digunakan untuk melakukan analisis dalam pengambilan keputusan terbaik dari sejumlah alternatif. Metoda *Bayes* dapat dirumuskan seperti yang ditunjukkan pada Gambar 1 (Anastasya Latubessy, dkk ; Vol. 3 ; No. 1 ; 2013 : 81):



Gambar 1. Model Bayes

$$P(X_k)|Y) = \frac{P\ (Y\ |X_k\ )}{\sum_j P\ (Y|X_j)} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots . (1)$$

Keadaan Posterior (Probabilitas X_k di dalam Y) dapat dihitung dari keadaan prior (Probabilitas Y di dalam X_k dibagi dengan jumlah dari semua probabilitas Y di dalam semua X_j).

Persamaan bayes yang digunakana untuk menghitung nilai setiap alternatif disederhanakan menjadi :

$$\text{Total Nilai}_{\ i} = \sum_{j=1}^{m} \text{Nilai}_{\ ij}\,(\text{Krit}_j) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

dimana:

Total nilai $_i$ = Total nilai akhir dari alternatif ke-i

NIlai $_{ij}$ = Nilai dari alternative ke-I pada kriteria ke-j

Krit $_j$ = Tingkat kepentingan (bobot) kriteria ke-j

i = 1,2,3,…n ; n = jumlah alternatif

j = 1,2,3,…n ; n = jumlah criteria

Berdasarkan pada persamaan Bayes tersebut maka dibuatlah pemodelan dengan fokus yang diangkat adalah pemilihinan SIM *Card.* Diambil tiga alternatif kartu operator seluler sebagai *sample* yaitu, IM3, Simpati dan As. Beberapa kriteria yang akan digunakan adalah tarif sms, tarif telepon, dan sinyal di luar Pulau Jawa. Metode penilaian yang digunakan adalah skala ordinal (skala 1 untuk penilaian sangat kurang, 2 untuk penilaian kurang, 3 untuk penilaian biasa, 4 untuk penilaian bagus, dan 5 untuk penilaian sangat bagus). Pembobotan criteria dilakukan untuk setiap alternatif sehingga diperoleh tabel matriks keputusan seperti yang ditunjukkan pada Tabel 1.

Tabel 1. Tabel Matriks Keputusan

Alternatif	KRITERIA			Nilai Alternatif	Peringkat
	Tarif SMS	Tarif Telpon	Sinyal Luar P.Jawa		
IM3	4	5	2	3.86	1
As	3	3	4	3.42	3
Simpati	2	4	4	3.52	2
Bobot Bayes	**0.3**	**0.4**	**0.33**		

Nilai (IM3) = 4(0.3) + 5(0.4) + 2(0.33) = 3.86

Nilai (As) = 3(0.3) + 3(0.4) + 4(0.33) = 3.42

Nilai(Simpati) = 2(0.3) + 4(0.4) + 4(0.33) = 3.52

Dengan menggunakan perumusan *Bayes*, diperoleh alternatif yang terurut adalah

1. IM3, analisis resiko : Sinyal diluar Pulau Jawa kurang baik, namun tarif SMS bagus, serta tarif telepon sangat bagus.
2. Simpati, analisis resiko : Sinyal diluar Pulau Jawa dan tarif telepon Bagus, namun tarif SMS kurang baik.
3. As, analisis resiko : Sinyal diluar Pulau Jawa Bagus, tarif SMS dan tarif telepon biasa.

Dengan analisa resiko masing-masing. Keputusannya tergantung preferensi dari tiap *decision maker*.Sebagai contoh, jika *decision maker* memiliki preferensi berikut: tarif SMS = 4, tarif telepon = 3, sinyal Luar P.Jawa = 3, maka dicari nilai preferensi untuk kemudian dibandingkan dengan nilai dari tiap alternatif. Selisih terkecil untuk tiap alternatif diasumsikan sebagai alternatif terbaik untuk preferensi diatas:

Nilai(Preferensi) = 4(0.3) + 3(0.4) + 3(0.33) = 3.39

Perbandingan:

a. Nilai (IM3) – Nilai(Preferensi) →3.86 - 3.39 = 0.47(3)

b. Nilai (As) – Nilai(Preferensi) →3.42 - 3.39 = 0.03(1)

c. Nilai (Simpati) – Nilai(Preferensi) →3.52 - 3.39 = 0.13(2)

Sehingga diperoleh selisih terkecil adalah untuk kartu As. Berdasarkan asumsi tersebut maka kartu As dapat dikatakan sebagai alternatif terbaik untuk preferensi ini (Anastasya Latubessy, dkk ; Vol. 3 ; No. 1 ; 2013 : 81-82).

II.4. *Microsoft Visual Studio 2010*

Microsoft Visual Studio 2010 merupakan sebuah perangkat lunak yang dapat digunakan untuk melakukan pengembangan aplikasi, yang di dalamnya terdapat beberapa kumpulan *tools* pemrograman seperti *Visual Basic .NET, Visual C++ .NET, Visual C# .NET,* dan *Visual J# .NET*. Namun yang paling umum digunakan yaitu *Visual Basic .NET*. *Visual Basic .NET* adalah *Visual Basic* yang direkayasa kembali untuk digunakan pada *platform .NET* sehingga aplikasi yang dibuat menggunakan *Visual Basic .NET* dapat berjalan pada sistem komputer apapun, dan dapat mengambil data dari *server* dengan tipe apapun asalkan terinstal *.NET Framework* (Priyanto Hidayatullah ; 2012 : 8).

II.5. *Microsoft SQL Server*

Microsoft SQL Server merupakan produk *Relational Database Management System* (RDBMS) yang dibuat oleh *Microsoft*. Pada tahun 2008

Microsoft mengeluarkan *SQL Server 2008 R2* yang merupakan versi yang banyak digunakan. *SQL* (*Structured Query Language*) adalah sebuah bahasa yang dipergunakan untuk mengakses data dalam basis data relasional. Bahasa ini secara *de facto* merupakan bahasa standar yang digunakan dalam manajemen basis data relasional. Saat ini hampir semua *server* basis data yang ada mendukung bahasa ini untuk melakukan manajemen datanya. *SQL* terdiri dari dua bahasa, yaitu *Data Definition Language* (DDL) dan *Data Manipulation Language* (DML). Implementasi DDL dan DML berbeda untuk tiap Sistem Manajemen Basis Data (SMBD), namun secara umum implementasi setiap bahasa ini memiliki bentuk standar yang ditetapkan oleh ANSI (Adelia, dkk ; 2011 : 115).

II.6. *Database*

Database sering didefinisikan sebagai kumpulan data yang terkait. Secara teknis, yang berada dalam sebuah database adalah sekumpulan tabel atau objek lain (indeks, view, dan lain-lain). Tujuan utama pembuatan database adalah untuk memudahkan dalam mengakses data (Eka Choliviana, dkk ; Vol. 5 ; No. 1 ; 2013 : 55).

Basis data dapat didefinisikan sebagai himpunan kelompok data yang saling berhubungan yang diorganisasikan sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah. Prinsip utamanya adalah pengaturan data. Tujuan utamanya kemudahan dan kecepatan dalam pengambilan kembali data (Priyanto Hidayatullah ; 2012 : 137).

II.7. Normalisasi

Proses normalisasi merupakan proses pengelompokan elemen data menjadi tabel-tabel yang menunjukkan entitas dan relasinya. Proses ini selalu diuji pada beberapa kondisi. Apakah ada kesulitan pada saat menambah *(insert),* menghapus *(delete),* mengubah *(update),* atau membaca *(retrieve)* pada suatu database. Bila ada kesulitan pada pengujian tersebut maka relasi dapat dipecah dalam beberapa tabel lagi (Tata Sutabri ; 2012 : 138).

Normalisasi sendiri merupakan cara pendekatan lain dalam membangun desain lojik basis data relasional yang tidak secara langsung berkaitan dengan model data, tetapi dengan menerapkan sejumlah aturan dan kriteria standar untuk menghasilkan struktur tabel yang normal (Elena Monica, dkk ; Vol. 14 ; No. 2 ; 2015 : 69-70).

Sebuah tabel dapat dikatagorikan baik (efisien) atau normal, jika telah memenuhi 3 (tiga) kriteria berikut:

1. Jika ada dekomposisi (penguraian) tabel, maka dekomposisinya harus dijamin aman (*Lossless-Join Decomposition*).
2. Terpeliharanya ketergantungan fungsional pada saat perubahan data (*Dependency Preservation*).
3. Tidak melanggar *Boyce-Code Normal Form* (BCNF).

   Jika kriteria ketiga (BCNF) tidak dapat terpenuhi, maka paling tidak harus diupayakan agar tabel tersebut tidak melanggar Bentuk Normal tahap ke ketiga (*3rd Normal Form/ 3NF*).

Bentuk-bentuk Normal (*Normal* Form) yang lain adalah:

1. Bentuk Normal tahap Pertama (*1st Normal Form/1NF*)

   Bentuk Normal tahap Pertama (1NF) terpenuhi jika sebuah tabel tidak memiliki Atribut Bernilai Banyak (*Multivalued Attribute*) atau lebih dari satu atribut dengan domain nilai yang sama.

2. Bentuk Normal tahap Kedua (*2nd Normal Form/2NF*) Bentuk Normal tahap Kedua (2NF) terpenuhi jika pada sebuah tabel, semua atribut yang tidak termasuk dalam *key primer* memiliki Ketergantungan Fungsional (KF) pada *key primer* secara utuh. Sebuah tabel dikatakan tidak memenuhi 2NF, jika ketergantungannya hanya bersifat parsial (hanya tergantung pada sebagian dari *key primer*).

3. Bentuk Normal tahap Keempat (*4th Normal Form*) dan Bentuk Normal tahap Kelima (5^{th} *Normal Form*).

Penerapan aturan Normalisasi sampai dengan tahap ketiga sesungguhnya sudah sangat memadai untuk menghasilkan tabel-tabel yang berkualitas baik. Namun demikian, dari sejumlah *literature* dapat pula dijumpai adanya pembahasan tentang Bentuk Normal tahap Keempat (4NF) dan Bentuk Normal tahap Kelima (5NF). Bentuk Normal tahap Keempat berkaitan dengan sifat Ketergantungan Banyak Nilai (*Multivalued Dependency*) pada suatu tabel yang merupakan pengembangan dari Ketergantungan Fungsional. Sedangkan Bentuk Normal tahap Kelima (merupakan nama lain dari *Project-Join Normal Form*/ PJNF) berkenaan dengan Ketergantungan Relasi antartabel (*Join Dependency*). Pembahasan kedua bentuk normal yang terakhir ini cukup kompleks, tetapi

manfaatnya sendiri tidak begitu besar (Elena Monica, dkk ; Vol. 14 ; No. 2 ; 2015 : 69-70).

II.8. *Unified Modeling Language* (UML)

Unified Modelling Language (UML) adalah suatu alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual. Juga merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang terkait dengan objek (Haviluddin ; Vol 6 ; No. 1 ; 2011 : 1).

Berdasarkan beberapa pendapat dikemukakan diatas dapat ditarik kesimpulan bahwa "*Unified Modeling Language* (UML) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk menvisualisasikan, menspesifikasikan, membangun dan pendokumentasian dari sebuah sistem pengembangan perangkat lunak berbasis OO (*Object Oriented*) (Aris, dkk ; 2015 : 3).

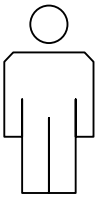
II.8.1. *Use Case* Diagram

Use case model adalah teknik pemodelan untuk mendapatkan *functional requirement* dari sebuah sistem, menggambarkan interaksi antara pengguna dan sistem, menjelaskan secara naratif bagaimana sistem akan digunakan, menggunakan skenario untuk menjelaskan setiap aktivitas yang mungkin terjadi.

Ada beberapa bagian didalam *use case model*.

Gambar II.4 *Use Case* Pada *Use Case Diagram*

(Sumber : Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 38)



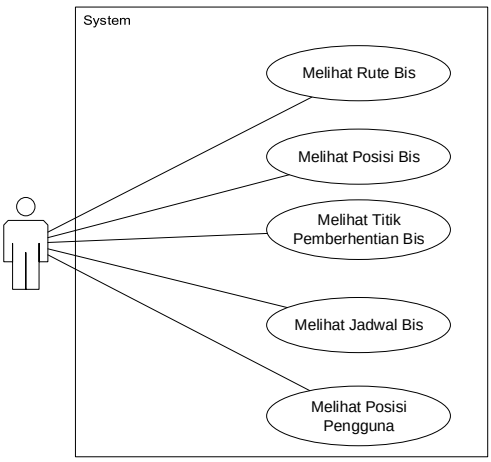
Actor

Gambar II.5. *Actor* Pada *Use Case Diagram*
(Sumber : Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 38)

a. Use Case, untuk mengetahui action atau prosedur apa yang ada didalam sistem.

b. Actor, siapa saja yang terlibat dalam action tersebut.

c. Relationship, bagaimana actions saling berelasi satu sama lain didalam sistem (Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 38).

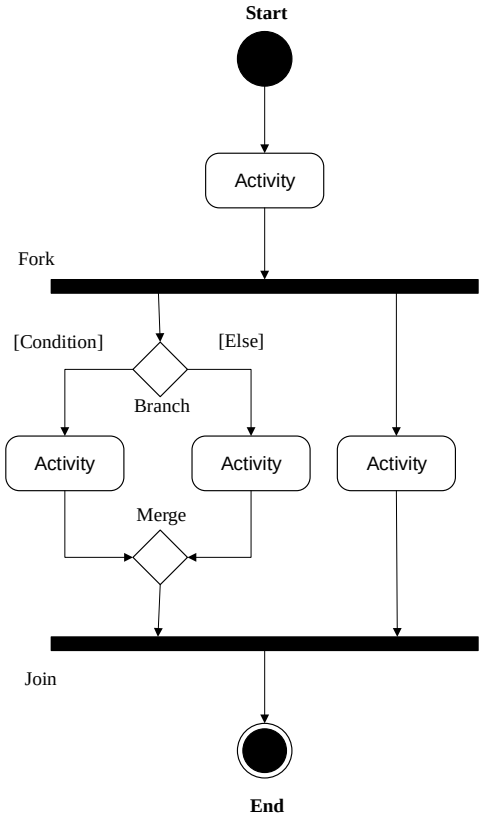


Gambar II.6. *Contoh Use Case Diagram*
(Sumber : Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 38)

II.8.2. *Activity* Diagram

Teknik untuk menjelaskan *business process*, *procedural logic*, dan *work flow*. Bisa dipakai untuk menjelaskan teks use case dalam notasi grafi dengan menggunakan notasi yang mirip *flow chart*, meskipun terdapat sedikit perbedaan notasi.

- *Nodes*, menandakan initial dan fial node, fial node boleh lebih dari 1.
- *Activity*, aktivitas sistem dapat berupa aktivitas fiik juga bagi user.
- *Flow/edge*, arah sebuah proses.
- *Fork*, awal sebuah proses paralel.
- *Join* akhir proses paralel.
- *Condition*, kondisi yang dituliskan dalam bentuk teks
- *Decision*, implementasi if dan then.
- *Merge*, penyatuan beberapa flw.
- *Partition*, siapa atau apa yang menjalankan aktivitas (Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 39).



Gambar II.9. Contoh *Activity Diagram*
(Sumber : Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 39)

II.8.3. *Class* Diagram

Class diagram merupakan diagram paling umum yang dijumpai dalam pemodelan berbasis UML. Didalam Class diagram terdapat class dan interface beserta atribut-atribut dan operasinya, relasi yang terjadi antar objek, constraint terhadap objek-objek yang saling berhubungan dan inheritance untuk organisasi class yang lebih baik. Class diagram juga terdapat static view dari elemen pembangun sistem. Pada intinya Class diagram mampu membantu proses

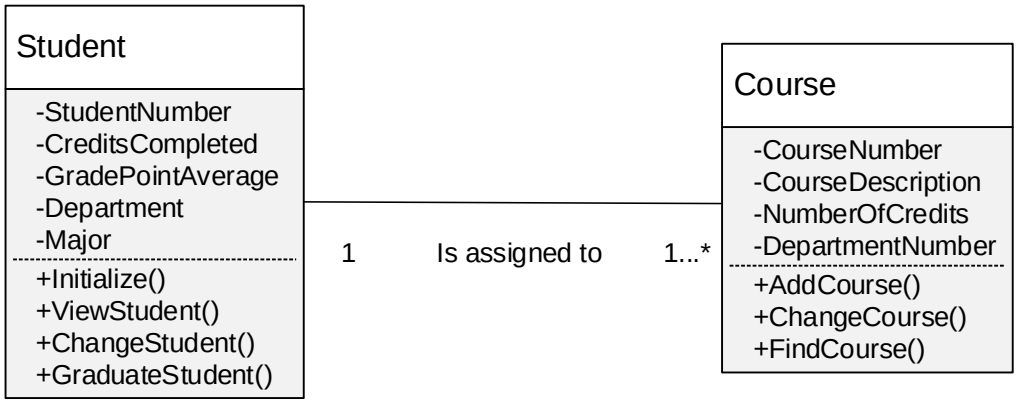
pembuatan sistem dengan memanfaatkan konsep forward ataupun reverse engineering (Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 38).

Berbagai simbol yang hadir didalam *class* diagram antara lain adalah (Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 38) :

1. *Class*, yang berfungsi untuk merepresentasikan tipe dari data yang dimilikinya. *Class* diagram dapat ditampilkan dengan menunjukkan atribut dan operasi yang dimilikinya atau hanya menunjukkan nama *class*-nya saja. Dapat juga kita tuliskan nama *class* dengan atributnya saja atau nama *class* dengan operasinya.
2. *Attribute*, merupakan data yang terdapat didalam *class* dan *instance*-nya dengan operator.
3. *Operation*, berfungsi untuk merepresentasikan fungsi-fungsi yang ditampilkan oleh *class* dan *instance*-nya dengan operator.
4. *Association*, digunakan untuk menunjukkan bagaimana dua *class* berhubungan satu sama lainnya. *Association* ditunjukkan dengan sebuah garis yang terletak diantara dua *class*. Didalam setiap *association* terdapat *multiplicity*, yaitu simbol yang mengindikasikan berapa banyak *instance* dari *class* pada ujung *association* yang satu dengan *instance class* di ujung *association* lainnya.
5. *Generalizations*, berfungsi untuk mengelompokkan *class* ke dalam hirarki *inheritance*.
6. *Aggregation*, merupakan bentuk khusus dari *association* yang merepresentasikan hubungan "*part-whole*". Bagian "*whole*" dari

hubungan ini sering disebut dengan *assembly* atau *aggregate*. *Class* yang satu dapat dikatakan merupakan bagian dari *class* yang lain yang ikut membentuk *class* tersebut.

7. *Composition*, merupakan jenis *aggregation* yang lebih kuat diantara dua *class* yang memiliki *association* dimana jika *whole* ditiadakan, maka *part*-nya juga ikut ditiadakan. Berbeda dengan *aggregation*, *part* akan tetap bisa berdiri sendiri meskipun bagian *whole*-nya ditiadakan.
8. Penggunaan operator (+) dalam *class* diagram diartikan dengan *public*, operator (-) diartikan *private*, dan operator (#) diartikan *protected*.



Gambar II.4. Contoh *Class* Diagram

(Sumber : Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 38)

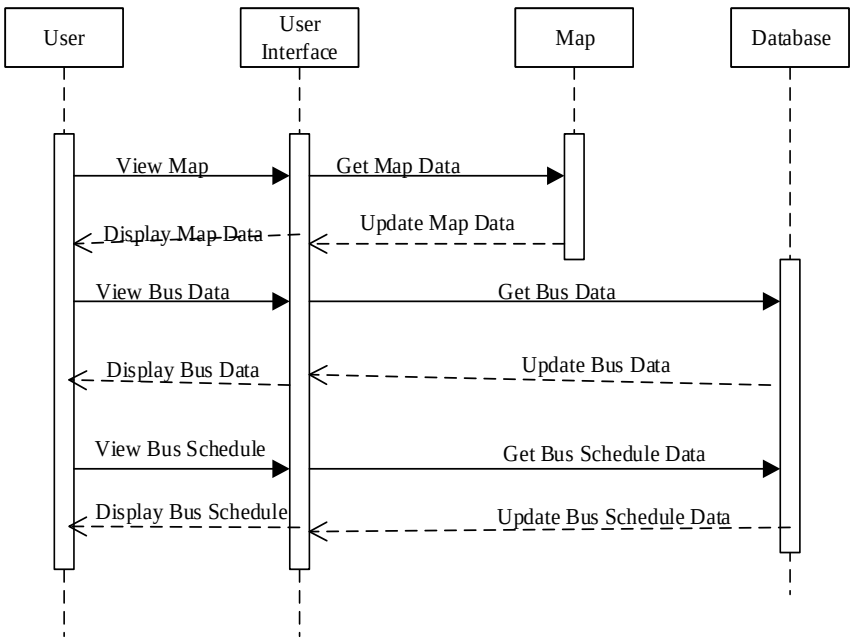
II.8.4. *Sequence* Diagram

Menjelaskan interaksi obyek-obyek yang saling berkolaborasi (berhubungan), mirip dengan *activity diagram* yaitu menggambarkan alur kejadian sebuah aktivitas tetapi lebih detil dalam menggambarkan aliran data

termasuk data atau behaviour yang dikirimkan atau diterima namun kurang mampu menjelaskan detil dari sebuah algoritma.

Dalam *sequence diagram* terdapat beberapa bagian.

1. *Participant*, yaitu objek yang terkait dengan sebuah urutan proses.
2. *Lifeline*, menggambarkan daur hidup sebuah objek.
3. *Activation*, suatu titik waktu dimana sebuah objek mulai berpartisipasi dalam sebuah sequence.
4. *Time*, elemen paling penting dalam sequence diagram yang konteksnya adalah urutan, bukan durasi.
5. *Return*, suatu hasil kembalian sebuah operasi. Operasi mengembalikan hasil tetapi boleh tidak ditulis jika tidak ada perbedaan dengan Getternya.



Gambar II.8. Contoh *Squence Diagram*
(Sumber : Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 38)

Dalam penggunaannya, *sequence diagram* tepat untuk memperlihatkan dengan jelas bagaimana urutan kejadian suatu proses karena didalamnya terlihat interaksi beberapa objek (Edgar Winata, dkk ; Vol. 4 ; No. 1 ; 2013 : 39).