BAB II

TINJAUAN PUSTAKA

II.2. Sistem Informasi Geografis

Sistem informasi geografis adalah sistem berbasis komputer yang digunakan untuk menyimpan dan memanipulasi informasi-informasi geografis. Sistem informasi geografi diciptakan untuk mengumpulkan, menyimpan dan menganalisis obyek atau fenomena dimana lokasi geografis menjadi karakteristik atau kritik penting untuk analisi. Sistem informasi geografi adalah sistem berbasis komputer yang memiliki kemampuan dalam menangani data berefrensi dalam:

1. Masukkan data
2. Manajemen data
3. Manipulasi
4. Analisis

Pada awalnya data geografi hanya disajikan di atas peta dengan menggunakan simbol, garis, dan warna. Elemen-elemen geometri ini dideskripsikan di dalam legenda-nya misalnya, garis hitam tebal untuk jalan utama, garis hitam tipis untuk jalan sekunder dan jalan-jalan yang berikutnya. Selain itu, berbagai data juga di dapat di-overlay-kan berdasarkan sistem koordinat yang sama. Akibatnya, sebuah peta menjadi media yang efektif baik sebagai alat presentasi maupun sebagai bank tempat penyimpanan data geografis. Tetapi, media peta masih mengandung kelemahan atau keterbatasan. Informasi-informasi yang tersimpan, diproses dan dipresentasikan dengan suatu cara

tertentu, dan biasanya untuk tujuan tertentu pula. Tidak mudah untuk mengubah bentuk presentasi ini, sebuah peta selalu menyediakan gambar atau symbol unsure geografi dengan bentuk yang tetap atau statis meskipun diperlukan untuk kebutuhan yang berbeda. (S. Nofan Maulana Rachman : 2012 : 2)

Untuk mendukung suatu sistem informasi geografis, pada prinsipnya terdapat dua jenis data, yaitu :

1. Data Spasial

   Data yang berkaitan dengan aspek keruangan dan merupakan data yang menyajikan lokasi geografis atau gambaran nyata suatu tempat/wilayah dipermukaan bumi. Umumnya direpresentasikan berupa grafik, peta, ataupun gambar dengan format digital dan disimpan dalam bentuk koordinat x,y (vektor) atau dalam bentuk image (raster) yang memiliki nilai tertentu.

2. Data Non Spasial

   Disebut juga data atribut, yaitu data yang menerangkan keadaan atau informasi-informasi dari suatu objek (lokasi dan posisi) yang ditunjukkan oleh data spasial. Salah satu komponen utama dari sistem informasi geografis adalah perangkat lunak (*software*). Perangkat lunak berfungsi sebagai alat yang dapat membantu dalam memvisualisasikan, mengeksplorasi, menjawab *query* dan menganalisis data secara geografis. (Eko Budianto; 2009 : 10)

II.3. Metode *Hill Climbing Search*

Metode *Hill Climbing Search* adalah metode yang menentukan node - node yang telah diberi jarak antar node yang lain dengan membandingkan dengan node yang telah ada berdasarkan pemilihan jarak terdekat dari posisi sekarang.

Kelebihannya tidak perlu memilih node yang telah diuji untuk dibandingkan lagi sehingga tidak memakan waktu pemprosesannya.

Metode ini hampir sama dengan metode pembangkitan dan pengujian, hanya saja proses pengujian dilakukan dengan menggunakan fungsi heuristik. Pembangkitan keadaan berikutnya sangat tergantung pada *feedback* dari prosedur pengetesan. Tes yang berupa fungsi heuristik ini akan menunjukkan seberapa baiknya nilai terkaan yang diambil terhadap keadaan-keadaan lainnya yang mungkin. Ada dua macam metode *Hill Climbing Search*, yaitu *Simple Hill Climbing* dan *Steepest-ascent Hill Climbing*

1. Simple Hill Climbing Search

Algoritma untuk *Hill Climbing Search* adalah sebagai berikut :

1. Mulai dari keadaan awal, lakukan pengujian: jika merupakan tujuan, maka berhenti; dan jika tidak, lanjutkan dengan keadaan sekarang sebagai keadaan awal.

2. Kerjakan langkah-langkah berikut sampai solusinya ditemukan, atau sampai tidak ada node baru yang akan diaplikasikan pada keadaan sekarang : a. Cari node yang belum pernah digunakan; gunakan node ini untuk mendapatkan keadaan yang baru. b. Evaluasi keadaan baru tersebut.

i. Jika keadaan baru merupakan tujuan, keluar.

ii. Jika bukan tujuan, namun nilainya lebih baik daripada keadaan sekarang, maka jadikan keadaan baru tersebut menjadi keadaan sekarang

iii. Jika keadaan baru tidak lebih baik daripada keadaan sekarang, maka lanjutkan pencarian.

2. Steepest Ascent Hill Climbing Search

Steepest-Ascent Hill Climbing Search hampir sama dengan *Simple Hill Climbing Search* dan yang membedakan keduanya adalah pada gerakan pencarian yang tidak dimulai dari posisi paling kiri. Gerakan berikutnya dicari berdasarkan nilai heuristik terbaik. Dalam hal ini urutan penggunaan tidak menentukan penemuan solusi. Adapun algoritma untuk *SteepestAscent Hill Climbing Search* adalah :

1. Mulai dari keadaan awal, lakukan pengujian. Jika merupakan tujuan maka berhenti, dan jika tidak, lanjutkan dengan keadaan sekarang sebagai keadaan awal.
2. Ulangi hingga tujuan tercapai atau hingga pencarian tidak memberikan perubahan pada keadaan sekarang.

a. Tentukan SUCC sebagai nilai heuristik dari *successor-successor*.

b. Lakukan untuk tiap node yang digunakan oleh keadaan sekarang.

i. Gunakan node tersebut dan bentuk keadaan baru.

ii. Evaluasi keadaan baru tersebut jika merupakan tujuan keluar. Jika bukan, bandingkan nilai keuristiknya dengan SUCC. Jika lebih baik, jadikan nilai heuristik keadaan batu tersebut sebagai SUCC, tetapi jika tidak lebih baik, nilai SUCC tidak berubah.

iii. Jika SUCC lebih baik daripada nilai heuristik keadaan sekarang, ubah node SUCC menjadi keadaan sekarang.

Sebagai contoh implementasi berikut penentuan rute terdekat dengan menggunakan metode hillclimbing

1. Mulai dari keadaan awal, tentukan titik koordinat awal longitude = 98.666389 , latitude = 3.6376651 , simpan sebagai posisi awal saat ini
2. Lakukan perulangan hingga tujuan tercapai atau hingga pencarian tidak memberikan perubahan pada keadaan sekarang.
3. Tentukan SUCC sebagai nilai heuristik dari *successor-successor*.

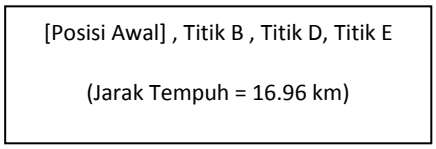
   Untuk kondisi saat ini maka SUCC dari tetangga terdekat adalah

   Titik B, Longitude = 98.6778 Latitude=3.7575

   Titik D, Longitude = 98.6671 Latitude=3.6647

   Titik E, Longitude = 98.67206 Latitude=3.61656

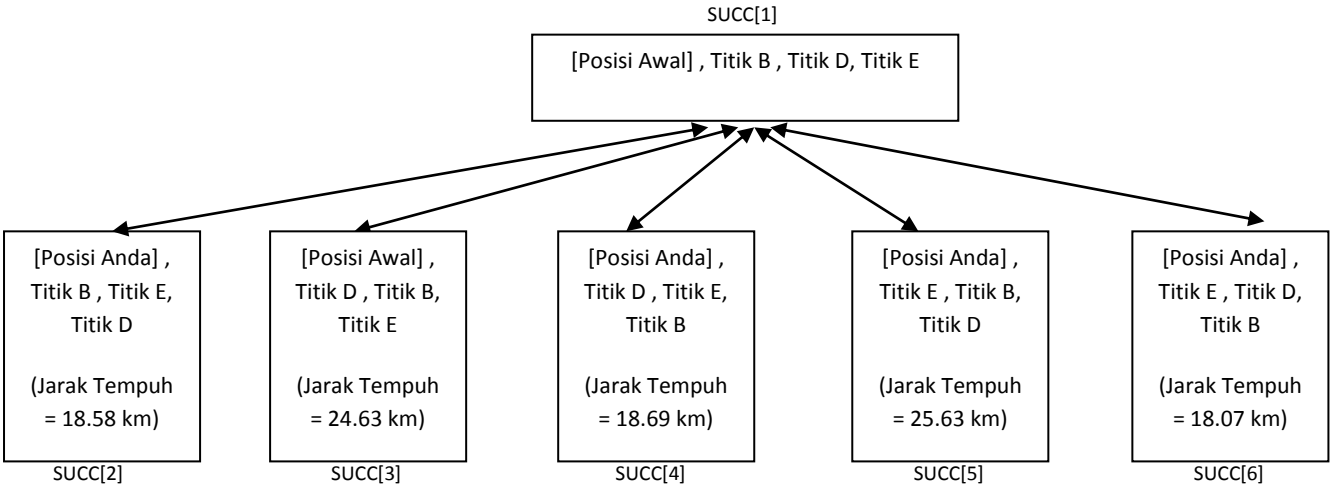
   Tentukan SUCC dan hitung jarak tempuh masing-masing node simpan dalam variabel jarak tempuh.



Gambar III.1. Algoritma *Steepest Ascent Hill Climbing Search*

4. Lakukan untuk tiap node yang digunakan oleh keadaan sekarang.
   a. Gunakan node tersebut dan bentuk keadaan baru.
   b. Evaluasi keadaan baru tersebut jika merupakan tujuan keluar. Jika bukan, bandingkan nilai keuristiknya dengan SUCC. Jika lebih baik, jadikan nilai heuristik keadaan batu tersebut sebagai SUCC, tetapi jika tidak lebih baik, nilai SUCC tidak berubah.

c. Jika SUCC lebih baik daripada nilai heuristik keadaan sekarang, ubah node SUCC menjadi keadaan sekarang.



Gambar III.2. Detail Algoritma *Steepest Ascent Hill Climbing Search*

Dari evaluasi setiap node SUCC diatas, SUCC[1] memiliki jarak tempuh sepanjang 16.96 km, SUCC[2] memiliki jarak tempuh sepanjang 18.56 km, SUCC[3] memiliki jarak tempuh sepanjang 24.63 km, SUCC[4] memiliki jarak tempuh sepanjang 18.69 km, SUCC[5] memiliki jarak tempuh sepanjang 25.63 km, SUCC[6] memiliki jarak tempuh sepanjang 18.07 km, maka dapat diambil kesimpulan rute terpendek dari node *successor-successor* adalah SUCC[1].

II.4. Android

Android adalah sistem operasi untuk perangkat selular yang berbasis Linux yang mencakup sistem operasi, *middleware* dan aplikasi. (Nazruddin Saffat : 2014 : 3).

Android menyediakan *platform* terbuka bagi para pengembang buat menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti

bergerak. Awalnya, Google Inc. membeli Android Inc.pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia. Pada saat perilisan perdana Android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Dilain pihak, Google merilis kode–kode Android di bawah lisensi *Apache*, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler. Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail Services* (GMS) dan kedua adalah yang benar–benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD).

II.4.1 Fitur dan Arsitektur Android

Fitur yang tersedia pada Android adalah:

1. Framework Aplikasi : memungkinkan penggunaan dan pemindahan dari komponen yang tersedia.
2. Dalvik Virtual Machine : virtual machine yang dioptimalkan untuk perangkat mobile.
3. Grafik : grafik 2D dan grafik 3D yang didasarkan pada library OpenGL.
4. SQLite : untuk menyimpan data.

5. Mendukung Media : audio, video, dan berbagai format gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
6. GSM, Bluetooth, Edge, 3G, WiFi, Camera, Global Positioning System (GPS), compass, dan accelerometer (tergantung hardware).
7. Lingkungan pengembangan yang kaya, termasuk emolator, peralatan debugging, dan plugin untuk Eclipse IDE. (Zara Zulfariana dan Ernastuti : 2012 : 1)

Sistem operasi Android dibangun berdasarkan kernel Linux, dan memiliki arsitekur sebagai berikut:

1. Applications

Lapisan ini adalah lapisan aplikasi, serangkaian aplikasi akan terdapat pada perangkat mobile. Aplikasi inti yang telah terdapat pada Android termasuk kalender, kontak, SMS (Short Message Service), dan lain sebagainya. Aplikasi-aplikasi ini ditulis dengan bahasa pemrograman Java.

2. Application Framework

Pengembang aplikasi memiliki akses penuh ke Android sama dengan aplikasi inti yang telah tersedia. Pengembang dapat mudah mengakses informasi lokasi, mengatur alarm, menambah pemberitahuan ke status bar dan lainnya sebagainya. Arsitektur aplikasi ini dirancang untuk menyederhanakan penggunaan kembali komponen, aplikasi apa pun yang dapat memublikasikan kemampuan dan aplikasi lainnya dapat menggunakan kemampuan mereka sesuai batasan keamanan. Dasar dari aplikasi adalah seperangkat layanan sistem, yaitu berbagai

view yang digunakan untuk membangun user interface, content provider yang memungkinkan aplikasi berbagi data, ResourceManager menyediakan akses bukan kode seperti grafik, string, dan layout, NotificationManager yang akan membuat aplikasi dapat menampilkan tanda pada status bar dan ActivityManager yang berguna mengatur daur hidup dari aplikasi.

3. Libraries

Satu set libraries dalam bahasa C/C++ yang digunakan oleh berbagai komponen pada sistem multimedia.

4. Android Runtime

Satu set libraries inti yang menyediakan sebagian besar fungsi yang tersedia di libraries inti dari bahasa pemrograman Java. Setiap aplikasi akan berjalan sebagai proses sendiri pada Dalvik Virtual Machine.

5. Linux Kernel

Android bergantung pada Linux versi 2.6 untuk layanan sistem initi seperti keamanan, manajemen memori, manajemen proses, network stack, dan model driver. Kernel juga bertindak sebagai lapisan antara hardware dan seluruh software. (Zara Zulfariana dan Ernastuti : 2012 : 1).

II.4.2 Jenis-Jenis Versi Android

1. Android versi 1.1

Andoid versi 1.1 di rilis pada 9 Maret 2009 oleh Google. Android versi ini dilengkapi disupport oleh Google Mail Service dengan pembaruan estetis pada aplikasi, jam alarm, voice search (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan email.

2. Android versi 1.5 Cup Cake

Android Cup Cake di rilis pada pertengahan Mei 2009, masih oleh Google Inc. Adroid ini dilengkapi software development kit dengan berbagai pembaharuan termasuk penambahan beberapa fitur antara lain yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke Youtube, upload gambar ke Picasa langsung dari telepon, serta mendapat dukungan Bluetooth A2DP.

3. Android versi 1.6 Donut

Android Donut di rilis pada September 2009 menampilkan proses pencarian yang lebih baik dibandingkan versi-versi sebelumnya. Selain itu Android Duut memiliki fitur-fitur tambahan seperti galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus; kamera, camcorder dan galeri yang dintegrasikan; Text-to-speech engine; kemampuan dial kontak; teknologi text to change speech. Android Donut juga dilengkapi baterai indikator, dan kontrol applet VPN.

4. Android versi 2.0/2.1 Eclair

Android Eclair dirilis pada 3 Desember 2009. Perubahan yang ada antara lain adalah pengoptimalan hardware, peningkatan Google Maps 3.1.2, perubahan

UI dengan browser baru dan dukungan HTML5, daftar kontak yang baru, dukungan flash untuk kamera 3,2 MP, digital Zoom, dan Bluetooth 2.1. Android Eclair merupakan Adroid pertama yang mulai dipakai oleh banyak smartphone, fitur utama Eclair yaitu perubahan total struktur dan tampilan user interface.

5. Android versi 2.2 Froyo (Frozen Yogurt)

Android Froyo dirilis pada 20 mei 2012. Adroid versi ini memiliki kecepatan kinerja dan aplikasi 2 sampai 5 kali dari versi-versi sebelumnya. Selain itu ada penambahan fitur-fitur baru seperti dukungan Adobe Flash 10.1, intergrasi V8 JavaScript engine yang dipakai Google Chrome yang mempercepat kemampuan rendering pada browser, pemasangan aplikasi dalam SD Card, kemampuan WiFi Hotspot portabel, dan kemampuan auto update dalam aplikasi Android Market.

6. Android versi 2.3 Gingerbread

Andoid Gingerbread di rilis pada 6 Desember 2010. Perubahan-perubahan umum yang didapat dari Android versi ini antara lain peningkatan kemampuan permainan (gaming), peningkatan fungsi copy paste, layar antar muka (User Interface) didesain ulang, dukungan format video VP8 dan WebM, efek audio baru (reverb, equalization, headphone virtualization, dan bass boost), dukungan kemampuan Near Field Communication (NFC), dan dukungan jumlah kamera yang lebih dari satu.

7. Android versi 3.0/3.1 Honeycomb

Android Honeycomb di rilis pada awal 2012. Merupakan versi Android yang dirancang khusus untuk device dengan layar besar seperti Tablet PC. Fitur baru yang ada pada Android Honeycomb antara lain yaitu dukungan terhadap prosessor multicore dan grafis dengan hardware acceleration. User Interface pada Honeycomb juga berbeda karena sudah didesain untuk tablet. Tablet pertama yang memakai Honeycomb adalah tablet Motorola Xoom yang dirilis bulan Februari 2011. Selain itu sebuah perangkat keras produksi Asus bernama Eee Pad Transformer juga menggunakan OS Android honeycomb dan diharapkan akan masuk ke pasaran Indonesia pada Mei 2011.

8. Android versi 4.0 ICS (Ice Cream Sandwich)

Android Ice Cream Sandwich diumumkan secara resmi pada 10 Mei 2011 di ajang Google I/O Developer Conference (San Francisco), pihak Google mengklaim Android Ice Cream Sandwich akan dapat digunakan baik di smartphone ataupun tablet. Android Ice Cream Sandwich membawa fitur Honeycomb untuk smartphone serta ada penambahan fitur baru seperti membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan kontrol, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari email secara offline, dan berbagi informasi dengan menggunakan NFC. Ponsel pertama yang menggunakan sistem operasi ini adalah Samsung Galaxy Nexus.

9. Android versi 4.1 Jelly Bean

Android Jelly Bean juga diluncurkan pada acara Google I/O 10 Mei 2011 yang lalu. Android versi ini membawa sejumlah keunggulan dan fitur baru, diantaranya peningkatkan input keyboard, desain baru fitur pencarian, UI yang

baru dan pencarian melalui Voice Search yang lebih cepat. Versi ini juga dilengkapi Google Now yang dapat memberikan informasi yang tepat pada waktu yang tepat pula. Salah satu kemampuannya adalah dapat mengetahui informasi cuaca, lalu-lintas, ataupun hasil pertandingan olahraga. Sistem operasi Android Jelly Bean 4.1 pertama kali digunakan dalam produk tablet Asus, yakni Google Nexus 7.

10.Android versi 4.2 Jelly Bean

Fitur photo sphere untuk panaroma, daydream sebagai screensaver, power control, lock screen widget, menjalankan banyak user (dalam tablet saja), widget terbaru. Android 4.2 Pertama kali dikenalkan melalui LG Google Nexus 4. (Alfa Satyaputra, M.Sc : 2014 : 5).

II.5. Google Map API

Google Maps adalah layanan pemetaan berbasis web service yang disediakan oleh Google dan bersifat gratis, yang memiliki kemampuan terhadap banyak layanan pemetaan berbasis web. Google Maps juga memiliki sifat server side, yaitu peta yang tersimpan pada serverGoogle dapat dimanfaatkan oleh pengguna. Google Maps API adalah suatu libraryyang berbentuk javascript yang berguna untuk memodifikasi peta yang ada di Google Maps sesuai kebutuhan. Untuk membangun aplikasi yang memanfaatkan Google Maps di desktop dan mobile devicemaka akan digunakan Google Maps Javascript API v3 yang

memiliki keunggulan lebih cepat dari versi sebelumnya.(*Alqod Elian*, dkk : 2012 : 1)

II.6. PHP

Menurut kamus komputer, PHP adalah bahasa pemrograman untuk dijalankan melalui halaman web, umumnya digunakan untuk mengolah informasi di internet. Sedangkan dalam pengertian lain, PHP adalah singkatan dari PHP *Hypertext Preprocessor* yaitu bahasa pemrograman web *server-side* yang bersifat *open source* atau gratis. PHP merupakan *script* yang menyatu dengan HTML dan berada pada server (*server side HTML embedded scripting*) (Rulianto Kurniawan; 2010: 2).

II.7. MySQL

MySQL adalah salah satu jenis database server yang sangat terkenal. MySQL termasuk jenis RDBMS (*Relational Database Management System*).MySQL ini mendukung bahasa pemrograman PHP. MySQL juga mempunyai *query* atau bahasa SQL (*Structured Query Language*) yang simpel dan menggunakan *escape character* yang sama dengan PHP (Rulianto Kurniawan; 2010: 16).

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

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 :

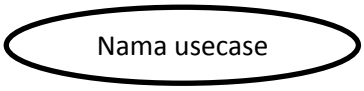
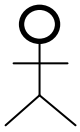
1. Menggambarkan batasan sistem dan fungsi – fungsi sistem secara umum, di buat dengan *use case* dan *actor*.
2. Menggambarkan kegiatan atau proses bisnis yang di laksanakan secara umum, di buat dengan *interaction diagrams*.
3. Menggambarkan representasi struktur *static* sebuah sistem dalam bentuk *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)

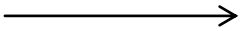
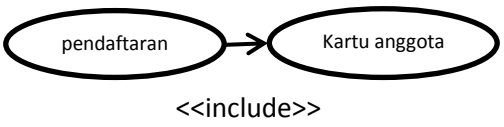
II.8.1. Use Case Diagram

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	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan dapat berdiri

 <<extend>>	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)

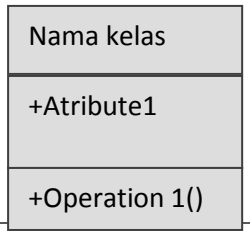
II.8.2. *Class* Diagram

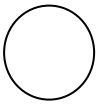
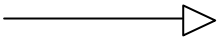
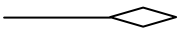
Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefenisian kelas – kelas yang akan di buat untuk membangun sistem. Kelas memiliki apa yang di sebut atribut dan metode atau operasi.

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*)

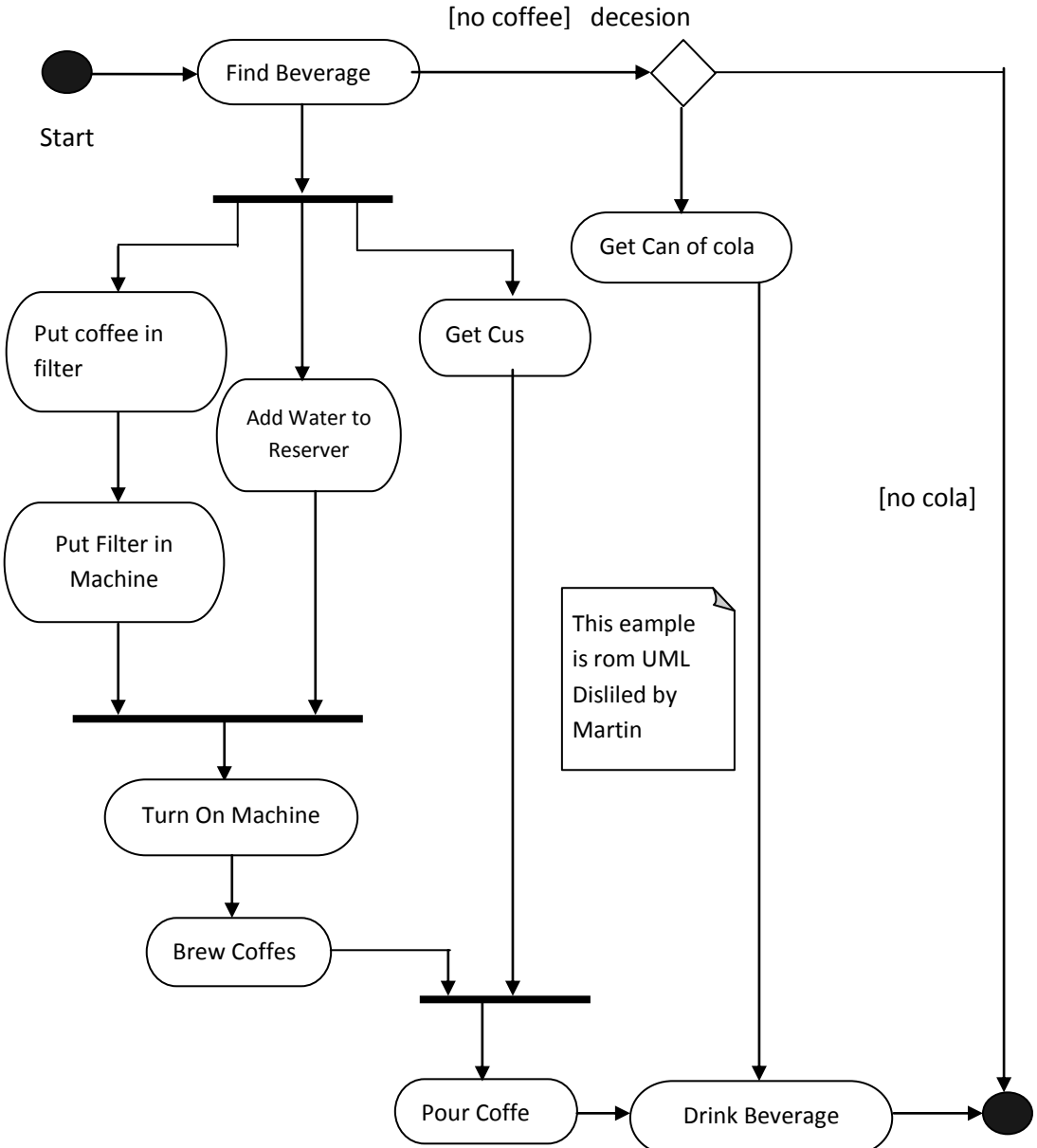
Sumber : (Yuni Sugiarti ; 2013)

II.8.3. *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)



Gamabar II.1 Activity Diagram

Sumber : (Yuni Sugiarti ; 2013)

II.8.4. *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.

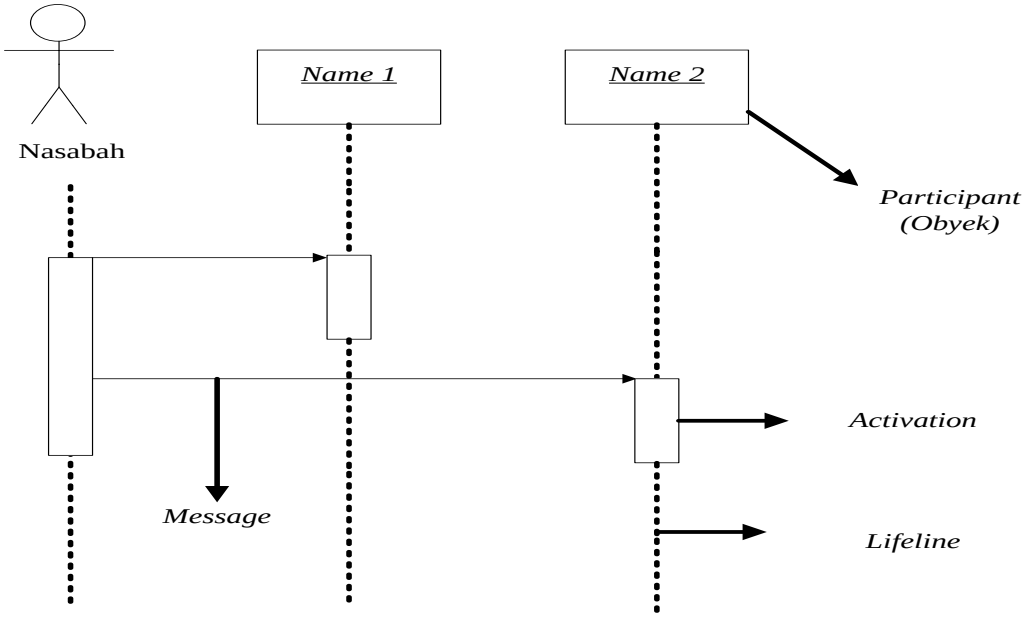
Diagram sequence memiliki ciri yang berbeda dengan diagram interaksi pada diagram kolaborasi sebagai berikut :

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*, jika kasus ini terjadi, maka garis hidupnya juga berakhir.

2. 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.9. 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.

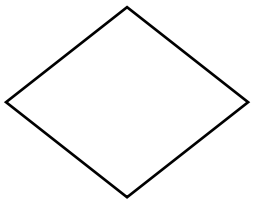
Gambar II.3 Simbol Entitas

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.

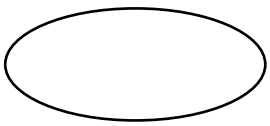


Gambar II.4 Simbol Relasi

Sumber : (Janner Simarmata,dkk; 2013)

3. Atribut (*Attribute*)

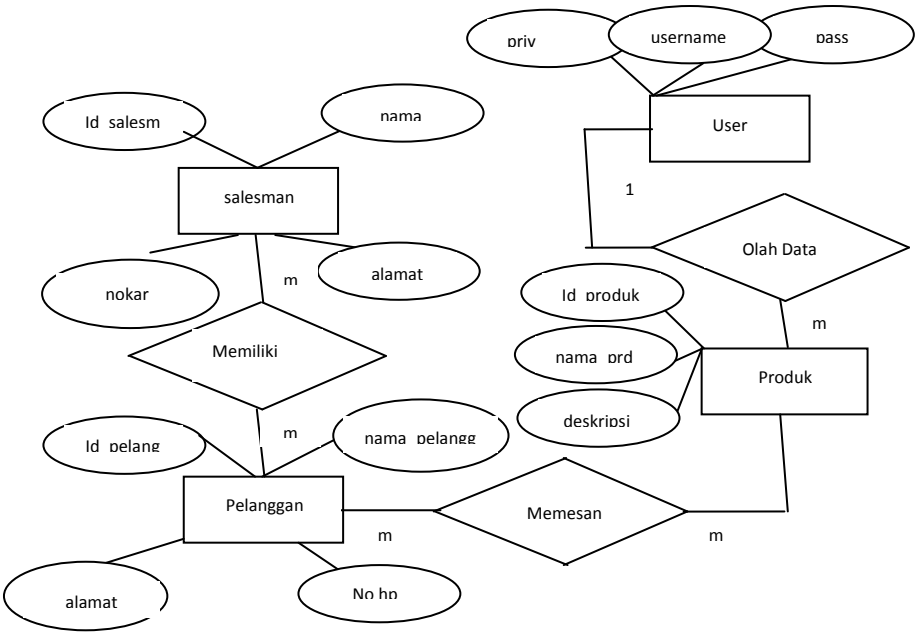
   Atribut adalah ciri umum semua semua atau sebagian besar instansi pada entitas tertentu. Sebutan lain atribut adalah *property*, elemen data, dan *field*. Misalnya, nama, alamat, nomor pegawai, dan gaji adalah atribut entitas pegawai. Sebuah atribut atau kombinasi atribut yang mengidentifikasikan satu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



II.10. 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.10.1. 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.11. 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)