BAB II
LANDASAN TEORI

II.1. Perancangan Sistem

Menurut McLeod (2001, p192) yang diterjemahkan oleh Teguh adalah "Perancangan sistem adalah penentuan proses data yang diperlukan oleh sistem baru. Jika sistem itu berbasis komputer, rancangan dapat menyertakan spesifikasi jenis peralatan yang digunakan". Menurut McLeod yang diterjemahkan oleh Teguh (2001, p192), tahap-tahap perancangan sistem informasi adalah sebagai berikut :

1. Menyiapkan rancangan sistem yang terinci
2. Mengidentifikasi berbagai alternatif konfigurasi sistem
3. Mengevaluasi berbagai alternatif konfigurasi sistem
4. Memilih konfigurasi terbaik
5. Menyiapkan usulan penerapan
6. Menyetujui atau menolak penerapan sistem

Menurut Whitten, Bentley dan Dittman (2004, p39): "*System design is the specification or construction of a technical, computer based solution for the business requirements identified in a system analysis*". Yang diterjemahkan sebagai berikut: "Perancangan sistem adalah spesifikasi atau perwujudan dari solusi teknis berbasiskan komputer untuk kebutuhan bisnis yang diidentifikasikan di sistem analisis".

Menurut Romney dan Steinbart (2006, p792): "*System design is the process of preparing detail specifications for development of a new information*

system". Yang diterjemahkan sebagai berikut: "Perancangan sistem adalah suatu proses detail spesifikasi untuk mengembangkan sebuah sistem informasi yang baru". Berdasarkan pendapat-pendapat diatas, dapat disimpulkan bahwa perancangan sistem adalah proses mengimplementasikan hasil-hasil dari analisis sistem ke dalam suatu rancangan sistem yang baru. (Henny Hendarti, 2009)

II.2. Sistem Informasi Geografis

Sistem informasi geografis adalah suatu sistem berbasis komputer untuk menangkap, menyimpan, mengecek, mengintegrasikan, memanipulasi, dan mendisplay data dengan peta digital. (I Wayan Eka Swastikayana, 2011)

II.2.1. Konsep Dasar Sistem Informasi Geografis

Sistem Informasi Geografis (SIG) atau juga dikenal sebagai *Geographic Information System* (GIS) pertama pada tahun 1960 yang bertujuan untuk menyelesaikan permasalahan geografis. 40 tahun kemudian GIS berkembang tidak hanya bertujuan untuk menyelesaikan permasalahan geografi saja tetapi sudah merambah ke berbagai bidang seperti analisis penyakit epidemik (demam berdarah) dan analisis kejahatan (kerusuhan) termasuk analisis kepariwisataan.

Kemampuan dasar dari SIG adalah mengintegrasikan berbagai operasi basis data seperti query, menganalisisnya serta menampilkannya dalam bentuk pemetaan berdasarkan letak geografisnya. Inilah yang membedakan SIG dengan sistem informasi lain. (I Wayan Eka Swastikayana, 2011)

II.2.2. Definisi Sistem Informasi Geografis

Istilah *geography* digunakan karena SIG dibangun berdasarkan pada geografi atau spasial. Objek ini mengarah pada spesifikasi lkasi dalam suatu space. *Geographic Information System* (GIS) merupakan sistem komputer yang berbasis pada sistem informasi yang digunakan untuk memberikan bentuk digital dan analisis terhadap permukaan geografi bumi.

Geografi adalah informasi mengenal permukaan bumi dan semua obyek yang berada diatasnya, sedangkan sistem informasi geografis (SIG) atau dalam bahasa inggris disebut *Geographic Information System* (GIS) adalah sistem informasi khusus yang mengelola data yang memiliki informasi spasial (bereferensi keruangan). Sistem informasi geografis adalah bentuk sistem informasi yang menyajikan informasi dalam bentuk grafis dengan menggunakan peta sebagai antar muka. SIG tersusun atas konsep beberapa lapisan (*layer*) dan relasi. (I Wayan Eka Swastikayana, 2011)

II.2.3. Cara kerja Sistem Informasi Geografis

SIG dapat menyajikan *real world* (dunia nyata) pada monitor sebagaimana lembaran peta dapat merepresentasikan dunia nyata diatas kertas. Tetapi, SIG memiliki kekuatan lebih dan fleksibilitas dari pada lembaran pada kertas. Peta merupakan representasi grafis dari dunia nyata, obyek-obyek yang dipresentasikan di atas peta disebut unsur peta atau map features (contohnya adalah sungai, taman, kebun, jalan dan lain-lain). Karena peta mengorganisasikan unsur-unsur berdasrkan lokasi-lokasinya. SIG menyimpan semua informasi

deksriptif unsur-unsurnya sebagai atribut-atribut didalam basis data. Kemudian, SIG membentuk dan menyimpannya didalam tabel-tabel (relasional) dengan demikian, atribut-atribut ini dapat diakses melalui lokasi-lokasi unsur-unsur peta dan sebaliknya, unsur-unsur peta juga dapat diakses melaluiatribut-atributnya. (I Wayan Eka Swastikayana, 2011)

II.3. Hotel

Hotel berasal dari kata *hostel*, konon diambil dari bahasa Perancis kuno. Bangunan publik ini sudah disebut-sebut sejak akhir abad ke-17. Maknanya kira-kira, "tempat penampungan buat pendatang" atau bisa juga "bangunan penyedia pondokan dan makanan untuk umum". Jadi, pada mulanya hotel memang diciptakan untuk meladeni masyarakat. Di Indonesia, kata hotel selalu dikonotasikan sebagai bangunan penginapan yang cukup mahal. Umumnya di Indonesia dikenal hotel berbintang, hotel melati yang tarifnya cukup terjangkau namun hanya menyediakan tempat menginap dan sarapan pagi, serta guest house baik yang dikelola sebagai usaha swasta (seperti halnya hotel melati) ataupun mess yang dikelola oleh perusahaan-perusahaan sebagai tempat menginap bagi para tamu yang ada kaitannya dengan kegiatan atau urusan perusahaan.

Pengertian Hotel menurut L.Foster dalam bukunya "*An Introduction to Travel & Tourism*" mengungkapkan bahwa dalam arti luas, hotel mungkin merujuk pada segala jenis penginapan. Sedangkan dalam arti sempit, hotel adalah sebuah bangunan yang dibangun khusus untuk menyediakan penginapan bagi para pejalan dengan pelayanan makanan dan minuman.

Hotel adalah bangunan yang menyediakan kamar-kamar untuk menginap para tamu dilengkapi makanan, minuman serta fasilitas-fasilitas lainnya yang diperlukan dan dikelola secara profesional untuk mendapatkan keuntungan.

Hotel adalah suatu bentuk akomodasi yang dikelola secara komersial disediakan bagi setiap orang untuk memperoleh pelayanan dan penginapan berikut makan dan minum. (SK.Menteri Perhubungan No. Pon.10/ Pw.301/Phb.77).

Hotel merupakan jenis akomodasi yang dikelola secara komersial, disediakan bagi seseorang atau sekelompok orang, menyediakan pelayanan penginapan, makanan dan minuman serta layanan lain sesuai perkembangan kebutuhan dan teknologi. (Lia Kusumawardani, 2014)

II.4. *Google Maps API*

Google Maps API adalah suatu library yang berbentuk *JavaScript*. Cara membuat *Google Maps* untuk ditampilkan pada suatu *web* atau *blog* sangat mudah hanya dengan membutuhkan pengetahuan mengenai HTML serta *JavaScript*, serta koneksi Internet yang sangat stabil. Dengan menggunakan *Google Maps API*, kita dapat menghemat waktu dan biaya untuk membangun aplikasi peta digital yang handal, sehingga kita dapat fokus hanya pada data-data yang akan ditampilkan. Dengan kata lain, kita hanya membuat suatu data sedangkan peta yang akan ditampilkan adalah milik *Google* sehingga kita tidak dipusingkan dengan membuat peta suatu lokasi, bahkan dunia. (I Wayan Eka Swastikayana, 2011)

Dalam pembutan program *Google Map API* menggunakan urutan sebagai berikut:

1. Memasukkan *Maps API JavaScript* ke dalam HTML.
2. Membuat *element div* dengan nama *map_canvas* untuk menampilkan peta.
3. Membuat beberapa objek literal untuk menyimpan properti-properti pada peta.
4. Menuliskan fungsi *JavaScript* untuk membuat objek peta.
5. Meng-inisiasi peta dalam *tag body* HTML dengan *event onload*.

II.5. Pengertian Android

Android adalah sistem operasi yang digunakan di *smartphone* dan juga *tablet* PC. Fungsinya sama seperti sistem operasi *Symbian* di Nokia, iOS di *Apple* dan *BlackBerry* OS. Android tidak terikat ke satu merek *handphone* saja, beberapa vendor terkenal yang sudah memakai Android antara lain Samsung, Sony Ericsson, HTC, Nexus, Motorolla, dan lain-lain.

II.5.1. Sejarah Android

Android adalah sistem operasi untuk telepon seluler yang berbasiskan Linux. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri sehingga dapat digunakan oleh bermacam peranti penggerak. Awalnya Google Inc. membeli Android Inc. pendatang baru yang membuat *software* (perangkat lunak) untuk telepon genggam. Kemudian untuk mengembangkan Android di bentuklah *Open Handset Alliance* yang merupakan gabungan dari 34 perusahaan peranti keras, peranti lunak dan

telekomunikasi termasuk Google, HTC, Intel, Motorola, Qualcomm, TMobile, dan NVidia.

Pada saat perilisan perdana Android pada tanggal 5 november 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode-kode Android dibawah lisensi Apache, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler. Terdapat dua jenis distributor sistem operasi Android. Pertama yang dapat dukungan penuh dari Google atau *Google Mail Service* (GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung dari Google atau dikenal sebagai *Open Handset Distribution* (DHD) (Firdan Ardiansyah; 2011).

II.5.2. Kelebihan Android

Setiap sistem operasi memiliki kelebihan dalam kinerja dan fitur-fiturnya, berikut adalah kelebihan sistem operasi android :

1. *User interface* menarik.
2. *Open Source*. Sumber bebas dan terbuka.
3. Multitasking. Bisa menjalankan berbagai aplikasi dalam satu waktu.
4. Kemudahan dalam notifikasi. Setiap ada SMS, Email, atau bahkan artikel terbaru dari RSS *Reader*, akan selalu ada notifikasi di *Home Screen* Ponsel Android.
5. Akses Mudah terhadap Ribuan Aplikasi Android lewat Google Android *App Market*.
6. Pilihan Ponsel yang beranekaragam. Android tersedia di ponsel dari berbagai

produsen, mulai dari Sony Ericsson, Motorola, HTC sampai Samsung.

7. Bisa menginstal ROM yang dimodifikasi.
8. Widget.
9. Terintegrasi dengan google.
10. Aman dari virus. Karena menggunakan kernel dari linux.

II.6. Eclipse

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua *platform* (*platformindependent*) (Ardzi Firman Ihtiyar Rohanianto, 2014). Berikut ini adalah sifat dari Eclipse:

1. *Multi-platform*: Target sistem operasi Eclipse adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
2. *Multi-language*: Eclipse dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lainnya, seperti C/C++, Cobol, Python, Perl, PHP, dan lain sebagainya.
3. *Multi-role*: Selain sebagai IDE untuk pengembangan aplikasi, Eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, *test* perangkat lunak, pengembangan *web*, dan lain sebagainya.

Eclipse pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari Eclipse yang membuatnya

populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan *plug-in* (Ardzi Firman Ihtiyar Rohanianto, 2014)

II.7. Pemodelan 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.

Sejarah UML sendiri terbagi dalam dua *fase* sebelum dan sesudah munculnya UML. Dalam *fase* sebelum, UML sebenarnya sudah mulai diperkenalkan sejak tahun 1990an namun notasi yang dikembangkan oleh para ahli analisis dan desain berbeda-beda, sehingga dapat dikatakan belum memiliki *standarisasi.*

Fase kedua dilandasi dengan pemikiran untuk mempersatukan metode tersebut dan dimotori oleh *Object Management Group* (OMG) maka pengembangan UML dimulai pada akhir tahun 1994 ketika Grady Booch dengan metode OOD (*Object-Oriented Design*), Jim Rumbaugh dengan metode OMT (*Object Modelling Technique*) mereka ini bekerja pada *Rasional Software Corporation* dan Ivar Jacobson dengan metode OOSE (*Object-Oriented Software Engineering*) yang bekerja di perusahaan *Objectory Rasional* (Haviluddin; 2011).

UML menyediakan 10 macam diagram untuk memodelkan aplikasi berorientasi objek, yaitu:

1. *Use Case* Diagram untuk memodelkan proses bisnis.

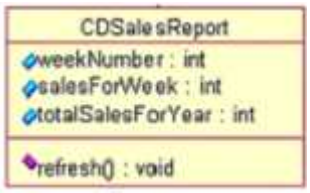
2. *Conceptual* Diagram untuk memodelkan konsep-konsep yang ada di dalam aplikasi.
3. *Sequence* Diagram untuk memodelkan pengiriman pesan *(message*) antar objek.
4. *Collaboration* Diagram untuk memodelkan interaksi antar objek.
5. *State* Diagram untuk memodelkan perilaku *objects* di dalam sistem.
6. *Activity* Diagram untuk memodelkan perilaku *Use Cases* dan objek di dalam *system.*
7. *Class* Diagram untuk memodelkan struktur kelas.
8. *Object* Diagram untuk memodelkan struktur objek.
9. *Component* Diagram untuk memodelkan komponen *object.*
10. *Deployment* Diagram untuk memodelkan distribusi aplikasi.

Untuk menggambarkan analisa dan *desain* diagram, UML memiliki seperangkat notasi yang akan digunakan ke dalam tiga kategori diatas yaitu struktur diagram, *behaviour* diagram dan *interaction* diagram. Berikut beberapa notasi dalam UML diantaranya :

1. *Actor*; menentukan peran yang dimainkan oleh *user* atau sistem lain yang berinteraksi dengan subjek. *Actor* adalah segala sesuatu yang berinteraksi langsung dengan sistem aplikasi komputer, seperti orang, benda atau lainnya. Tugas *actor* adalah memberikan informasi kepada sistem dan dapat memerintahkan sistem untuk melakukan sesuatu tugas.

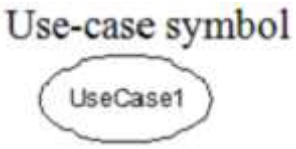
Gambar 2.2 Actor
Sumber : Haviluddin, 2011

2. *Class* diagram; Notasi utama dan yang paling mendasar pada diagram UML adalah notasi untuk mempresentasikan suatu *class* beserta dengan atribut dan operasinya. *Class* adalah pembentuk utama dari sistem berorientasi objek.



Gambar 2.3 Class Diagram
Sumber : Haviluddin, 2011

3. *Use Case* dan *use case specification*; *Use case* adalah deskripsi fungsi dari sebuah sistem perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut skenario. *Use case* merupakan awal yang sangat baik untuk setiap *fase* pengembangan berbasis objek, *design, testing*, dan dokumentasi yang menggambarkan kebutuhan sistem dari sudut pandang di luar sistem. Perlu diingat bahwa *use case* hanya menetapkan apa yang seharusnya dikerjakan oleh sistem.



Gambar 2.4 Use Case
Sumber : Haviluddin,2011

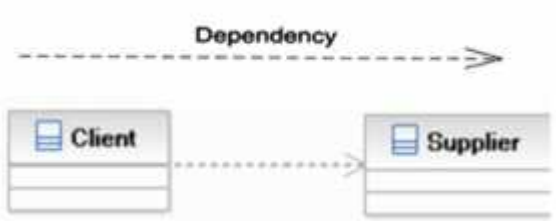
4. *Realization; Realization* menunjukkan hubungan bahwa elemen yang ada di bagian tanpa panah akan merealisasikan apa yang dinyatakan oleh elemen yang ada di bagian dengan panah.

Gambar 2.5 Realization
Sumber : Haviluddin, 2011

5. *Interaction; Interaction* digunakan untuk menunjukkan baik aliran pesan atau informasi antar *obyek* maupun hubungan antar *obyek.*

Gambar 2.6 Interaction
Sumber : Haviluddin, 2011

6. *Dependency; Dependency* merupakan relasi yang menunjukan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain. Terdapat 2 *stereotype* dari *dependency*, yaitu *include* dan *extend*. *Include* menunjukkan bahwa suatu bagian dari elemen (yang ada digaris tanpa panah) memicu eksekusi bagian dari elemen lain (yang ada di garis dengan panah). *Extend* menunjukkan bahwa suatu bagian dari elemen di garis tanpa panah bisa disisipkan ke dalam elemen yang ada di garis dengan panah.

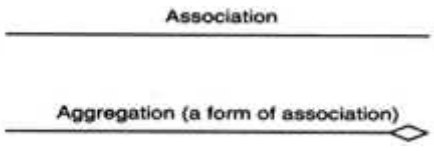


Gambar 2.7 Dependency
Sumber : Haviluddin, 2011

7. *Note; Note* digunakan untuk memberikan keterangan atau komentar tambahan dari suatu elemen sehingga bisa langsung terlampir dalam model. Note ini bisa disertakan ke semua elemen notasi yang lain.

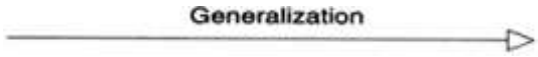
Gambar 2.8 Note
Sumber : Haviluddin, 2011

8. *Association; Association* menggambarkan navigasi antar *class (navigation*), berapa banyak obyek lain yang bisa berhubungan dengan satu obyek (*multiplicity* antar *class*) dan apakah suatu *class* menjadi bagian dari *class* lainnya (*aggregation*).



Gambar 2.9 Association
Sumber : Haviluddin, 2011

9. *Generalization; Generalization* menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik.

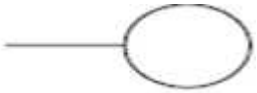


Gambar 2.10 Generalization
Sumber : Haviluddin,, 2011

10. Package; package adalah mekanisme pengelompokkan yang dgunakan untuk menandakan pengelompokkan elemen-elemen model.

Gambar 2.11 Package
Sumber : Haviluddin, 2011

11. *Interface; Interface* merupakan kumpulan operasi berupa implementasi dari suatu class. Atau dengan kata lain implementasi operasi dalam *interface* dijabarkan oleh operasi di dalam *class.*

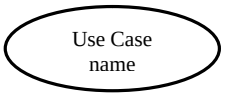
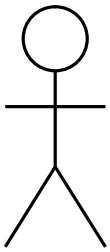


Gambar 2.12 Interface
Sumber : Haviluddin, 2011

Berikut adalah simbol dan komponen-komponen diagram pada UML :

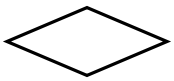
1. *Use Case* Diagram

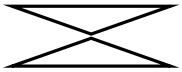
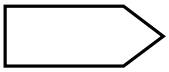
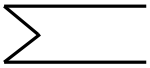
Tabel II.1 Komponen *Use Case* Diagram

Nama Komponen	Keterangan	Simbol
Use Case	*Use case* digambarkan sebagai lingkaran elips dengan nama *use case* dituliskan didalam elips tersebut.	
Actor	*Actor* adalah pengguna sistem. *Actor* tidak terbatas hanya manusia saja, jika sebuah sistem berkomunikasi dengan aplikasi lain dan membutuhkan *input* atau memberikan *output*, maka aplikasi tersebut juga bisa dianggap sebagai *actor*.	
Association	Asosiasi digunakan untuk menghubungkan *actor* dengan *use case*. Asosiasi digambarkan dengan sebuah garis yang menghubungkan antara *Actor* dengan *Use Case*.	

2. *Activity* Diagram

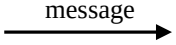
Tabel II.2 Komponen *Activity* Diagram

Simbol	Keterangan
	Titik awal
	Titik akhir
	*Activity*
	Pilihan untuk mengambil keputusan
	Fork; Digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan paralel menjadi satu.

	*Rake*; Menunjukkan adanya dekomposisi
	Tanda waktu
	Tanda pengiriman
	Tanda penerimaan
	Aliran akhir (*Flow Final*)

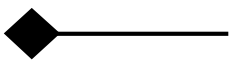
3. *Sequence* Diagram

Tabel II.3 Komponen *Sequence* Diagram

Nama Komponen	Keterangan	Simbol
Lifeline	*Lifeline* mengindikasikan keberadaan sebuah *object* dalam basis waktu. Notasi untuk *Lifeline* adalah garis putus-putus *vertikal* yang ditarik dari sebuah *object* .	
Activation	*Activation* dinotasikan sebagai sebuah kotak segi empat yang digambar pada sebuah *lifeline*. mengindikasikan sebuah obyek yang melakukan sebuah aksi.	
Message	*Message*, digambarkan dengan anak panah horizontal antara *Activation* *Message* mengindikasikan komunikasi antara *object -object* .	

4. Class Diagram

Tabel II.4 Komponen *Class* Diagram

Nama Komponen	Keterangan	Simbol
Class	*Class* adalah blok - blok pembangun pada pemrograman berorientasi obyek. Sebuah class digambarkan sebagai sebuah kotak yang terbagi atas 3 bagian. Bagian atas adalah bagian nama dari class. Bagian tengah mendefinisikan *property*/*atribut class*. Bagian akhir mendefinisikan *method-method* dari sebuah *class*.	Nama *Class* +atribut +atribut +atribut +*method* +*method*
Association	Sebuah asosiasi merupakan sebuah *relationship* paling umum antara 2 *class* dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 *class*. Garis ini bisa melambangkan *tipe-tipe relationship* dan juga dapat menampilkan hukum-hukum multiplisitas pada sebuah *relationship*. (Contoh: *One-to-one*, *one-to-many*, *many-to-many*).	1..n owned by 1
Composition	Jika sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain, maka *class* tersebut memiliki relasi *Composition* terhadap *class* tempat dia bergantung tersebut. Sebuah *relationship composition* digambarkan sebagai garis dengan ujung berbentuk jajaran genjang berisi/*solid*.	
Dependency	Kadangkala sebuah *class* menggunakan *class* yang lain. Hal ini disebut *dependency*. Umumnya penggunaan *dependency* digunakan untuk menunjukkan operasi pada suatu	

	class yang menggunakan *class* yang lain. Sebuah *dependency* dilambangkan sebagai sebuah panah bertitik-titik.	
Aggregation	*Aggregation* mengindikasikan keseluruhan bagian *relationship* dan biasanya disebut sebagai relasi.