BAB II
LANDASAN TEORI

II.1 Pengertian Aplikasi

Aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Contoh utama aplikasi adalah pengolah kata, lembar kerja, memanipulasi foto, merancang rumah dan pemutar media. Beberapa aplikasi yang digabung bersama menjadi suatu paket disebut sebagai suatu paket atau *suite* aplikasi (*application suite*).

II.2 Pembelajaran

Istilah belajar dan pembelajaran merupakan suatu istilah yang memiliki keterkaitan yang sangat erat dan tidak dapat dipisahkan satu sama lain dalam proses pendidikan. Pembelajaran seharusnya merupakan kegiatan yang dilakukan untuk menciptakan suasana atau memberikan pelayanan agar siswa belajar. Untuk itu, harus dipahami bagaimana siswa memperoleh pengetahuan dari kegiatan belajarnya. Jika guru dapat memahami proses pemerolehan pengetahuan, maka guru akan dapat menentukan strategi pembelajaran yang tepat bagi siswanya.

Menurut Sudjana (2000) dalam Sugihartono, dkk (2007: 80) pembelajaran merupakan setiap upaya yang dilakukan dengan sengaja oleh pendidik yang dapat menyebabkan peserta didik melakukan kegiatan belajar. Sedangkan Nasution (2005) dalam Sugihartono, dkk (2007: 80) mendefinisikan pembelajaran sebagai suatu aktifitas mengorganisasi atau mengatur lingkungan sebaik-baiknya dan menghubungkannya dengan anak didik sehingga terjadi proses belajar.

Lingkungan dalam pengertian ini tidak hanya ruang belajar, tetapi juga meliputi guru, alat

peraga, perpustakaan, laboratorium, dan sebagainya yang relevan dengan kegiatan belajar siswa. Pembelajaran sebagai proses belajar yang dibangun oleh guru untuk mengembangkan kreatifitas berfikir yang dapat meningkatkan kemampuan berfikir siswa, serta dapat meningkatkan kemampuan mengkonstruksi pengetahuan baru sebagai upaya meningkatkan penguasaan yang baik terhadap materi pelajaran.

Dari berbagai pengertian pembelajaran di atas dapat disimpulkan bahwa pembelajaran merupakan suatu upaya yang dilakukan dengan sengaja oleh pendidik untuk mentransfer ilmu pengetahuan, mengorganisasi dan menciptakan sistem lingkungan dengan berbagai metode sehingga siswa dapat melakukan kegiatan belajar secara efektif dan efisien sehingga akan mendapatkan hasil yang seoptimal mungkin.

II.3 Ujian Akhir

Ujian akhir semester atau UAS merupakan kegiatan rutin yang dilaksanakan dalam sebuah institusi pendidikan. UAS dilaksanakan pada akhir semester sesuai dengan kalender akademik institusi bersangkutan. Sebelum UAS dilaksanakan, terdapat proses penjadwalan UAS yang bertujuan untuk mengatur jadwal UAS agar sesuai dengan waktu dan ruangan yang tersedia. Penjadwalan UAS ada yang mengikuti jadwal perkuliahannya, ada pula yang berbeda dengan jadwal perkuliahannya.

II.4 *Adobe Flash CS6*

Adobe Flash merupakan salah satu *software* yang banyak dinikmati oleh kebanyakan orang karena keandalannya mampu mengerjakan segala hal yang berkaitan dengan multimedia (Pranowo, 2011 : 1). Adobe Flash merupakan *software* yang sangat terkenal yang dirilis oleh perusahaan *software* ternama dari Amerika Serikat, yaitu *Adobe System Incorporated*. Selain

Flash, *Adobe* juga mengeluarkan software-sofware lainnya yang tidak kalah hebatnya, semisal *Adobe Photoshop, Premiere, Dreamweaver, Illustrator, After Effect, InDesign* dan lainnya.

Adobe Flash CS6 merupakan versi terbaru dari versi sebelumnya, *Adobe Flash CS5*. Program ini memiliki banyak fungsi, seperti pembuatan animasi objek, membuat presentasi, animasi iklan, game, pendukung animasi halaman web, hingga dapat digunakan untuk pembuatan film animasi. Meski secara keseluruhan *Adobe Flash CS6* memilki tampilan dan proses kerja yang sama dengan versi sebelumnya, namun pada versi baru ini memiliki beberapa fitur baru.

II.5 Multimedia

Multimedia adalah media yang menggabungkan dua unsur atau lebih media yang terdiri dari teks, grafis, gambar, foto, audio, video dan animasi secara terintegrasi. Multimedia terbagi menjadi dua kategori, yaitu: multimedia linier dan multimedia interaktif. Multimedia linier adalah suatu multimedia yang tidak dilengkapi dengan alat pengontrol apapun yang dapat dioperasikan oleh pengguna.

Multimedia ini berjalan sekuensial (berurutan), contohnya: TV dan film. Multimedia interaktif adalah suatu multimedia yang dilengkapi dengan alat pengontrol yang dapat dioperasikan oleh pengguna, sehingga pengguna dapat memilih apa yang dikehendaki untuk proses selanjutnya. Contoh multimedia interaktif adalah multimedia pembelajaran interaktif, aplikasi *game*, dan lain-lain.

II.6 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:

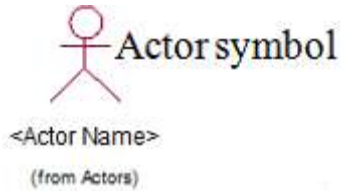
1. *Use Case* Diagram untuk memodelkan proses bisnis.
2. *Conceptual* Diagram untuk memodelkan konsep-konsep yang ada di dalam aplikasi.
3. *Sequence* Diagram untuk memodelkan pengiriman pesan *(message*) antar objek.
4. *Collaboration* Diagram untuk memodelkan interaksi antar objek.
5. *State* Diagram 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.

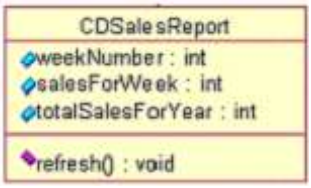
Untuk menggambarkan analisa dan *desain* diagram, UML memiliki seperangkat notasi yang akan digunakan ke dalam tiga kategori diatas yaitu struktur diagram, *behaviour* diagram dan *interaction* diagram. Berikut beberapa notasi dalam UML diantaranya :

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 II.1. Actor
(Sumber : Haviluddin, 2011)

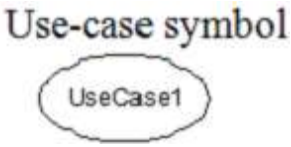
2. *Class* diagram; Notasi utama dan yang paling mendasar pada diagram UML adalah notasi untuk mempresentasikan suatu *class* beserta dengan atribut dan operasinya. *Class* adalah pembentuk utama dari sistem berorientasi objek.



Gambar II.2. Class Diagram
(Sumber : Haviluddin, 2011)

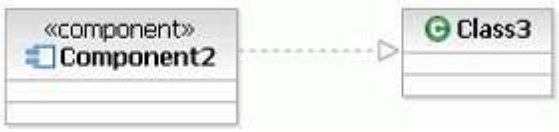
3. *Use Case* dan *use case specification*; *Use case* adalah deskripsi fungsi dari sebuah sistem perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana

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 II.3. 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.



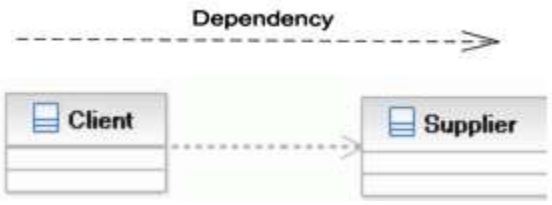
Gambar II.4. Realization
(Sumber : Haviluddin, 2011)

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

Gambar II.5. 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.

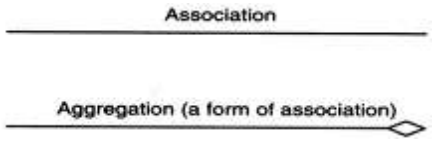


Gambar II.6. Dependency
(Sumber : Haviluddin, 2011)

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 II.7. 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*).

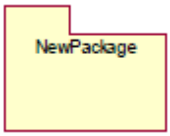


Gambar II.8. Association
(Sumber : Haviluddin, 2011)

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

Gambar II.9. Generalization
(Sumber : Haviluddin,, 2011)

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



Gambar II.10. Package
(Sumber : Haviluddin, 2011)

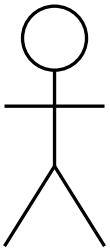
11. *Interface; Interface* merupakan kumpulan operasi berupa implementasi dari suatu class. Atau dengan kata lain implementasi operasi dalam *interface* dijabarkan oleh operasi di dalam *class.*

Gambar 2.11. Interface
(Sumber : Haviluddin, 2011)

Berikut adalah simbol dan komponen-komponen diagram pada UML :

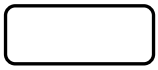
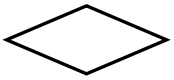
1. *Use Case* Diagram

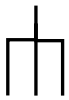
Tabel II.1. Komponen *Use Case* Diagram

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

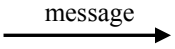
Tabel II.2. Komponen *Activity* Diagram

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*)

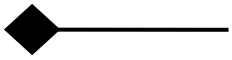
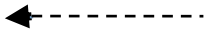
3. *Sequence* Diagram

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 akan 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.