BAB II

LANDASAN TEORI

2.1 Pengertian Sholat dan Do'a

2.1.1 Pengertian Sholat

Sholat menurut bahasa berarti do'a. Menurut istilah ahli fiqih berarti Perbuatan (gerak), dan perkataan yang dimulai dengan takbir dan diakhiri dengan salam dengan syarat-syarat tertentu. Sholat merupakan ibadah yang paling fundamental dalam Islam. Khususnya shalat fardhu lima kali sehari semalam yang tak boleh ditinggalkan. Sebagaimana Allah SWT berfirman :

رَبِّ ٱجۡعَلۡنِي مُقِيمَ ٱلصَّلَوٰةِ وَمِن ذُرِّيَّتِي رَبَّنَا وَتَقَبَّلۡ دُعَآءِ ٤٠

Artinya : *"Ya Tuhanku, jadikanlah aku dan anak cucuku sebagai orang-orang yang tetap mendirikan sholat, ya Tuhankami, perkenankanlah doaku."* (Q.S Ibrahim 14:40)

Menurut syariat Islam, praktik sholat harus sesuai dengan segala petunjuk tata cara Rasulullah SAW sebagai figur penyampai perintah Allah SWT. Nabi Muhammad SAW telah memberikan peringatan keras kepada orang yang suka meninggalkan sholat, diantaranya beliau bersabda: "*Perjanjian yang memisahkan kita dengan mereka adalah Shalat. Barang siapa yang meninggalkan sholat, berarti dia telah kafir"*, Hadist riwayat Imam Ahmad dan Tirmidzi.

Adapun rukun sholat, syarat sahnya sholat dan yang membatalkan / dilarang dalam sholat adalah sebagai berikut:

1. Rukun Sholat
    a. Niat

        Niat ialah menyengaja melakukan sesuatu bersamaan dengan dilakukannya bagian pertama dari sesuatu itu. Tempat niat didalam hati dibarengi takbiratul ihram.

    b. Berdiri bagi yang mampu

        Artinya bagi orang lanjut usia, bila dirasa tidak mampu berdiri, diperkenankan sholat dengan duduk, dan seterusnya melihat kondisi badannya (fisik).

    c. Mengucap Takbiratul Ihram

        Mengucap takbir sesaat setelah niat.

    d. Membaca Surat Al-Fatihah

    e. Ruku dengan thumaninah

        Thumaninah diartikan berhenti sejenak, pada waktu ruku dan i'tidal hendaklah yang sewajarnya, tidak terburu-buru atau dibuat selambat-lambatnya.

    f. I'tidal (Berdiri tegak sesudah ruku) dengan thumaninah

    g. Sujud 2 kali pada tiap-tiap rakaat dengan thumaninah

    h. Duduk diantara 2 sujud dengan thumaninah

    i. Tasyahud akhir

        Artinya duduk pada rakaat terakhir.

    j. Membaca tahiyat pada tasyahud akhir

k. Membaca sholawat atas Nabi pada tasyahud akhir

Maksudnya, membaca shalawat atas Nabi Muhammad SAW, sesudah selesai membaca tasyahud tersebut, sebelum salam.

l. Salam yang pertama

Salam yang pertama yaitu ucapan mushalli dengan menengok ke kanan

m. Tertib

Artinya dikerjakan sesuai dengan urutan, tidak boleh meloncat. Misalnya dari rukun yang ke 1 ke rukun yang ke 5.

2. Syarat sahnya sholat

Menurut Imam Asy Syafe'i ra., Syarat sahnya sholat ringkasnya ada 4 perkara, yaitu:

a. Bersuci (Thaharah) yaitu suci tubuh dari hadats kecil dan besar, suci tubuh dari najis, suci tempat dari najis, dan suci pakaian dari najis.
b. Mengetahui waktu sholat dengan ketentuan waktu sholatnya.
c. Menutup aurat.
d. Menghadap Ka'bah ketika sholat.

3. Hal-hal yang membatalkan sholat

a. Berbicara dengan sengaja
b. Bergerak dengan banyak
c. Meninggalkan salah satu rukun sholat
d. Berhadats
e. Terkena najis
f. Terbukanya aurat

g. Merobah niat

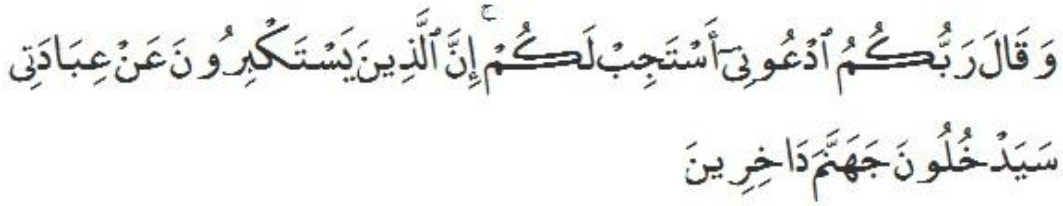
h. Membelakangi kiblat

i. Makan, minum, tertawa

j. Murtad (keluar dari agama islam).

2.1.2 Pengertian Do'a

Menurut bahasa "ad-du'aa" artinya memanggil, meminta tolong, atau memohon sesuatu. Sedangkan do'a menurut pengertian syariat adalah memohon sesuatu atau memohon perlindungan kepada Allah SWT dengan merendahkan diri dan tunduk kepada-Nya. Do'a merupakan bagian dari ibadah dan boleh dilakukan setiap waktu dan setiap tempat, karena Allah SWT selalu bersama hamba - hamba-Nya.



Artinya : *"Dan Tuhanmu berfirman: "Berdo'alah kepada-Ku, niscaya akan Kuperkenankan bagimu. Sesungguhnya orang-orang yang menyombongkan diri dari menyembah-Ku akan masuk neraka Jahannam dalam keadaan hina dina."* (QS. Al-Mu'min : 60)

Bagi orang mu'min yang ingin mendapatkan keberhasilan dalam kehidupan ada dua hal yang harus dilakukan, yaitu berusaha atau kerja keras dan berdo'a. Kedua cara tersebut harus ditempuh, karena di dalam kehidupan ini ada hal-hal yang tidak dapat dijangkau oleh pemikiran manusia. Oleh karena itu, di

dalam memecahkan masalah ini kehidupan kedua cara ini harus ditempuh secara bersama-sama. Do'a juga merupakan obat bagi orang yang sedang sakit.

2.2 Pengertian Aplikasi

Aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja, dan pemutar media.

2.3 Pengertian Aplikasi Mobile

Kata *mobile* mempunyai arti bergerak atau berpindah. Sehingga diperoleh pengertian bahwa aplikasi bergerak merupakan aplikasi yang dapat dijalankan walaupun pengguna berpindah atau karena pengguna berpindah.

Pemrograman aplikasi bergerak tidak banyak berbeda dengan pemrograman aplikasi bergerak tidak banyak berbeda dengan pemrograman konvensional pada *PC*. Aspek karakteristik dari perangkat bergerak sering mempengaruhi arsitektur dan implementasi dari aplikasi tersebut. Dalam pemrograman aplikasi bergerak berbagai aspek teknis perangkat lebih menonjol karena memiliki banyak keterbatasan dibandingkan komputer konvensional atau *PC*. (Budi Darytamo dkk, 2007:1).

Perangkat bergerak (*mobile*) memiliki banyak jenis dalam hal ukuran, desain dan layout, tetapi memiliki kesamaan karakteristik yang sangat berbeda dari sitem desktop.

1. Ukuran yang kecil

   Perangkat *mobile* memiliki ukuran yang kecil. Konsumen menginginkan perangkat yang terkecil untuk kenyamanan dan mobilitas mereka.

2. Memory yang kecil

   Perangkat *mobile* juga memiliki *memory* yang kecil, yaitu *primary* (RAM) dan *secondary (disk).* Pembatasan ini adalah salah satu faktor yang mempengaruhi penulisan program untuk berbagai jenis dari perangkat ini. Dengan pembatasan jumlah dari *memory*, pertimbangan-pertimbangan khusus harus diambil untuk memelihara pemakaian dari sumber daya yang mahal ini.

3. Daya proses yang terbatas

   Sistem *mobile* tidaklah setangguh rekan mereka yaitu desktop. Ukuran, teknologi dan biaya adalah beberapa faktor yang mempengaruhi status dari sumber daya ini. Seperti *harddisk* dan RAM, anda dapat menemukan mereka dalam ukuran yang pas dengan sebuah kemasan kecil.

4. Mengkonsumsi daya yang rendah

   Perangkat *mobile* menghabiskan sedikit daya dibandingkan dengan mesin desktop. Perangkat ini harus menghemat daya karena mereka

berjalan pada keadaan dimana daya yang disediakan dibatasi oleh baterai-baterai.

5. Kuat dan dapat diandalkan

   Karena perangkat *mobile* selalu dibawa kemana saja, mereka harus cukup kuat untuk menghadapim benturan-benturan, gerakan, dan sesekali tetesan-tetesan air.

6. Koneksi yang terbatas

   Perangkat *mobile* memiliki bandwith rendah, beberapa dari koneksi ini bahkan tidak tersambung. Kebanyakan dari mereka menggunakan koneksi *wireless*.

7. Masa hidup yang pendek

   Perangkat-perangkat konsumen ini menyala dalam hitungan detik kebanyakan dari mereka selalu menyala. Coba ambil kasus sebuah *handphone*, mereka *booting* dalam hitungan detik kebanyakan orang tidak mematikan *handphone* mereka bahkan ketika malam hari. PDA akan menyala jika anda menekan tombol power mereka.

2.4 Pengertian Informasi

Menurut Fatta (2007:9), informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau mendatang.

Menurut Rochim (2002:3), informasi menyandang arti manfaat, bila kita bisa memanfaatkanya, informasi mengandung makna usaha, untuk mendapatkannya, memahaminya, menggunakannya, menyebarkannya, menyimpannya dan memadukannya dengan informasi lain menjadi suatu bentuk informasi baru.

Menurut Kusrini dan Koniyo (2007:7), informasi adalah data yang sudah diolah menjadi sebuah bentuk yang berarti bagi pengguna, yang bermanfaat dalam pengembalian keputusan saat ini atau mendukung sumber informasi.

Menurut Wahyono (2004:3), informasi adalah data yang telah diolah menjadi bentuk yang berguna bagi penerimanya dan nyata, berupa nilai yang dapat dipahami di dalam keputusan sekarang maupun masa depan.

Berdasarkan definisi diatas dapat disimpulkan bahwa informasi adalah data yang telah diolah menjadi sesuatu yang bermanfaat bagi penggunanya itu sendiri.

2.4.1 Ciri-ciri Informasi

Ciri-ciri informasi menurut Azhar Susanto (2008 : 38) menyatakan bahwa:

1. Akurat

   Informasi harus mencerminkan keadaan yang sebenarnya.

2. Tepat waktu

   Informasi harus tersedia atau ada pada saat informasi tersebut diperlukan.

3. Relevan

   Informasi yang diberikan harus sesuai dengan yang dibutuhkan oleh suatu individu.

4. Lengkap

   Informasi harus diberikan secara lengkap.

2.4.2 Fungsi Informasi

Selain itu informasi mempunya berbagai fungsi, antara lain sebagai berikut:

1. Menambah pengetahuan.

   Adanya informasi yang akan menambah pengetahuan bagi penerimanya yang dapat digunakan sebagai baha pertimbangan yang mendukung proses pengambilan keputusan.

2. Mengurangi ketidakpastian.

   Adanya informasi yang akan mengurangi ketidakpastian karena apa yang akan terjadi dapat diketahui sebelumnya, sehingga menghindari keraguan pada saat pengambilan keputusan

3. Mengurangi resiko kegagalan.

   Adanya informasi akan mengurangi resiko kegagalan karena apa yang akan terjadi dapat diantisipasi dengan baik, sehingga kemungkinan terjadinya kegagalan akan dapat dikurangi dengan pengambilan keputusan yang tepat.

4. Mengurangi keanekaragaman/variasi yang tidak diperlukan.

   Adanya informasi akan mengurangi keanekaragaman yang tidak diperlukan, karena keputusan yang di ambil lebih terarah.

5. Memberi standar, aturan-aturan, ukuran dan keputusan keputusan yang menentukan pencapaian sasaran dan tujuan.

   Adanya informasi yang akan memberikan standar, aturan, ukuran, dan keputusan yang lebih terarah untuk mencapai sasara dan tujuan yang telah ditetapkan secara baik berdasarkan informasi yang diperoleh.

2.5 Metode Pendekatan Sistem

Metode pendekatan sistem yang digunakan oleh penulis adalah pendekatan dengan *Object Oriented* yang menggunakan OOAD *(Object Oriented Analisis and Design)*.

2.5.1 Pengertian OOAD

OOAD adalah metode analisis yang memeriksa *requirements* dari sudut pandang kelas kelas dan objek yang ditemui dalam ruang lingkup permasalahan yang mengarahkan arsitektur software yang didasarkan pada manipulasi objek-objek sistem atau subsistem. OOAD merupakan cara baru dalam memikirkan suatu masalah dengan menggunakan model yang dibuat menurut konsep sekitar dunia nyata. Dasar pembuatan adalah objek yang merupakan kombinasi antara struktur data dan perilaku dalam satu entitas.

2.5.2 Konsep Dasar OOAD

OOAD mencakup analisis dan desain sebuah sistem dengan pendekatan objek, yaitu analisis berorientasi objek (OOA) dan desain berorientasi objek (OOD). OOA adalah metode analisis yang memeriksa *requirement* (syarat/keperluan) yang harus dipenuhi sebuah sistem dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup tertentu. Sedangkan OOD adalah metode untuk mengarahkan arsitektur *software* yang didasarkan pada manipulasi objek-objek sistem atau subsistem.

2.5.2.1 Objek (Object)

1. Objek adalah benda secara fisik dan konseptual yang ada di sekitar kita. Sebuah objek memiliki keadaan sesaat yang disebut *state*.
2. *State* dari sebuah objek adalah kondisi dari objek atau himpunan keadaan yang menggambarkan objek tersebut. *State* dinyatakan dengan nilai dari atribut objeknya.
3. Atribut adalah nilai internal suatu objek yang mencerminkan karakteristik objek, kondisi sesaat, koneksi dengan objek lain dan identitas.
4. *Behaviour* (perilaku objek) mendefinisikan bagaimana sebuah objek bertindak dan memberi reaksi. *Behaviour* ditentukan oleh himpunan semua atau beberapa operasi yang dapat dilakukan oleh objek tersebut, yang dicerminkan oleh *interface, service,* dan *method* dari objek tersebut.

5. *Interface* adalah pintu untuk mengakses *service* dari objek.
6. *Service* adalah fungsi yang dapat dikerjakan oleh sebuah objek.
7. *Method* adalah mekanisme *internal* objek yang mencerminkan perilaku objek tersebut.

2.5.2.2 Kelas (Class)

Class adalah himpunan objek yang sejenis yaitu mempunyai sifat (atribut), perilaku umum (operasi), relasi umum dengan objek lain dan semantik umum. *Class* adalah abstraksi dari objek dalam dunia nyata. *Class* menetapkan spesifikasi perilaku dan atribut dari objek tersebut.

2.5.2.3 Kotak Hitam (*Black Box*)

Sebuah objek adalah kotak hitam. Konsep ini menjadi dasar implementasi objek. Dalam operasi *Object Oriented* hanya *developer* yang dapat memahami detail proses yang ada didalam kotak tersebut, sedangkan *user* tidak perlu mengetahui apa yang dilakukan yang penting mereka dapat menggunakan objek untuk memproses kebutuhan mereka. Kotak hitam berisi kode dan data.

1. *Encapsulation*, yaitu proses menyembunyikan detail implementasi sebuah objek. Untuk mengakses data objek tersebut adalah melalui *interface.* Untuk berkomunikasi dengan objek digunakan *message*.
2. *Message* adalah permintaan agar objek menerima untuk membawa metode yang ditunjukkan oleh perilaku dan mengembalikan *result* dari aksi tersebut kepada objek pengirim (*sender*).

2.5.2.4 Asosiasi dan Agregasi

Asosiasi adalah hubungan yang mempunyai makna antara sejumlah objek. Asosiasi digambarkan dengan sebuah garis penghubung diantara objeknya. Contohnya : Asosiasi karyawan dengan unit kerja. Setiap karyawan bekerja di satu unit kerja, sedangkan unit kerja dapat memiliki beberapa karyawan.

Agregasi adalah bentuk khusus sebuah asosiasi yang menggambarkan seluruh bagian pada suatu objek merupakan bagian dari objek yang lain. Contohnya : Kopling dan piston adalah bagian dari mesin, sedangkan mesin, roda, body merupakan bagian dari sebuah mobil.

2.5.3 Pengertian Objek

Objek adalah sesuatu yang dapat dilihat, disentuh atau dirasakan dan digunakan pengguna serta akan disimpan data dan perilakunya. Bisa berupa:

1. Orang, tempat, benda atau kejadian.
2. Pegawai, guru, dosen, mahasiswa, murid.
3. Kantor, gudang, bangunan, ruangan.
4. Kendaraan, produk, konputer, video.

2.5.4 Karakteristik Objek

1. Identitas berarti bahwa data diukur mempunyai nilai tertentu yang membedakan entitas disebut Objek.
2. Dua objek dapat berbeda walaupun bila semua nilai atributnya identik.

3. Objek dapat kongkrit, seperti halnya arsip dalam sistem, atau konseptual seperti kebijakan penjadualan dalam multiprocessing pada sistem operasi.
4. Setiap objek mempunyai sifat yang melekat pada identitasnya.

2.5.4.1 Kelas Objek

1. Kelas merupakan gambaran sekumpulan Objek yang terbagi dalam atribut, operasi, metode, hubungan, dan makna yang sama.
2. Suatu kegiatan mengumpulkan data (atribut) dan perilaku (operasi) yang mempunyai struktur data sama ke dalam satu grup.
3. Kelas Objek merupakan wadah bagi Objek. Dapat digunakan untuk menciptakan Objek.
4. Objek mewakili fakta/keterangan dari sebuah kelas.

2.5.5 Metodologi Berorientasi Objek

Metodologi adalah cara sistematis untuk mengerjakan *analisys* and *design*. Dengan metodologi, pihak yang membangun sistem *software* dapat merencanakan dan mengulangi pekerjaan dilain waktu. Metodologi juga menghilangkan perbedaan notasi untuk suatu hal yang sama karena setiap oarng akan berbicara dalam bahasa yang sama.

2.5.5.1 Karakteristik Metodologi Berorientasi Objek

1. *Encapsulation* (pengkapsulan) merupakan dasar untuk pembatasan ruang lingkup program terhadap data yang diproses, data dan prosedur atau fungsi dikemas bersama-sama dalam suatu objek, sehingga prosedur atau fungsi lain dari luar tidak dapat mengaksesnya, data terlindung dari prosedur atau objek lain, kecuali prosedur yang berada dalam objek itu sendiri.
2. *Inheritance* (pewarisan) adalah teknik yang menyatakan bahwa anak dari objek akan mewarisi data/atribut dan metode dari induknya langsung. Atribut dan metode dari objek induk diturunkan kepada anak objek, demikian seterusnya, *Inheritance* mempunyai arti bahwa atribut dan operasi akan dimiliki bersama di antara kelas yang mempunyai hubungan secara hirarki, suatu kelas dapat ditentukan secara umum, kemudian ditentukan spesifik menjadi subkelas. Setiap subkelas mempunyai hubungan atau mewarisi semua sifat yang dimiliki oleh kelas induknya, dan ditambah dengan sifat unik yang dimilikinya.
3. *Polymorphism* (polimorfisme) yaitu konsep yang menyatakan bahwa sesuatu yang sama dapat mempunyai bentuk dan perilaku berbeda, Polimorfisme mempunyai arti bahwa operasi yang sama mungkin mempunyai perbedaan dalam kelas yang berbeda, kemampuan objek-objek yang berbeda untuk melakukan metode yang sesuai

dalam merespon pesan yang sama, seleksi dari metode yang sesuai, bergantung pada kelas yang seharusnya menciptakan objek.

2.6 Teknik Pemodelan OOAD

Model Objek :

1. Model objek Menggambarkan struktur statis dari suatu objek dalam sistem dan relasinya.
2. Model objek berisi diagram objek. Diagram objek adalah *graph* dimana nodenya adalah kelas yang mempunyai relasi antar kelas.

Model Dinamik :

1. Model dinamik menggambarkan aspek dari sistem yang berubah setiap saat.
2. Model dinamik dipergunakan untuk menyatakan aspek kontrol dari sistem.
3. Model dinamik berisi *state diagram*. *State diagram* adalah *graph* dimana nodenya adalah *state* dan *arc* adalah tarnsisi antara *state* yang disebabkan oleh *event*.

Model Fungsional :

1. Model fungsional menggambarkan transformasi nilai data di dalam sistem.
2. Model fungsional berisi data suatu *graph* dimana nodenya menyatakan proses dan arcnya adalah aliran data.

2.7 Alat Bantu Analisis dan Perancangan

2.7.1 UML (Unified Modeling Languages)

Menurut Bambang Hariyanto. Ir.,MT. (2004:17) UML adalah bahasa grafis untuk mendokumentasikan, menspesifikasikan, dan membangun sistem perangkat lunak. UML berorientasi objek menerapkan banyak level abstraksi, tidak bergantung proses pengembangan, tidak bergantung bahasa dan teknologi.

Tujuan UML

1. Menyediakan bahasa pemodelan visual yang ekspresif dan siap pakai untuk mengembangkan pertukaran model-model yang berarti.
2. Menyediakan mekanisme perluasan dan spesialisasi untuk memperluas konsep-konsep inti.
3. Mendukung spesifikasi *independent* bahasa pemprograman dan proses pengembangan tertentu.
4. Menyediakan basis formal untuk pemahaman bahasa pemodelan.

2.7.1.1 Bagian-Bagian dari UML

1. Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya *login* ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya.

Seorang / sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.



Gambar 2.1. *Use Case Diagram*

(Munawar. 2005. *Pemodelan visual dengan UML*. Graha Ilmu. Yogyakarta.)

Diagram *Use Case* berguna dalam tiga hal :

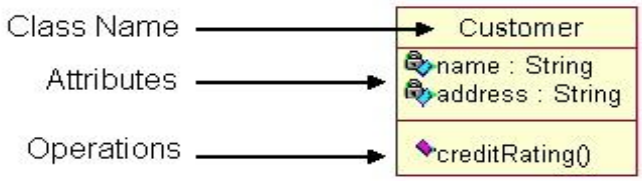
1. Menjelaskan fasilitas yang ada (*requirements) Use Case* baru selalu menghasilkan fasilitas baru ketika sistem di analisa, dan desain menjadi lebih jelas.
2. Komunikas dengan klien penggunaan notasi dan simbol dalam diagram *Use Case* membuat pengembang lebih mudah berkomunikasi dengan klien - kliennya.
3. Membuat tes dari kasus-kasus secara umum kumpulan dari kejadian-kejadian untuk *Use Case* bisa dilakukan tes kasus layak untuk kejadian - kejadian tersebut.

2. Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut / properti) suatu

sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda / fungsi).

Class diagram menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.



Gambar 2.2. Bagian-bagian dari *class*

(Munawar. 2005. *Pemodelan visual dengan UML*. Graha Ilmu. Yogyakarta.)

Class memiliki tiga area pokok :

1. Nama (dan *stereotype*)
2. Atribut
3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

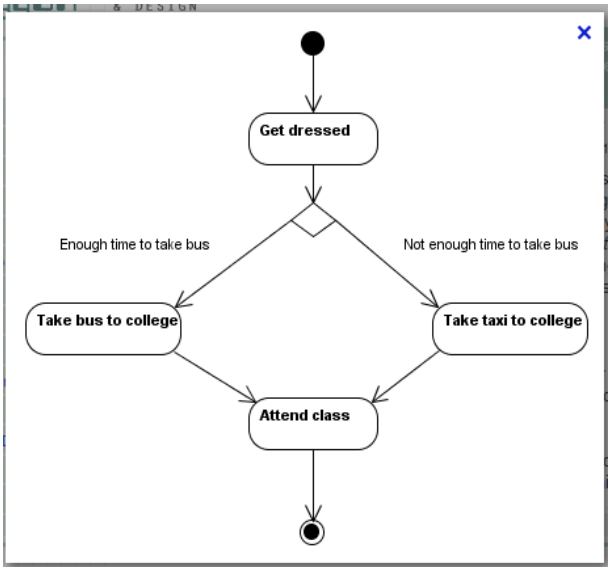
1. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.
2. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya.
3. *Public*, dapat dipanggil oleh siapa saja.

3. Statechart Diagram

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu state ke state lainnya) suatu objek pada sistem sebagai akibat dari stimuli yang diterima. Pada umumnya *statechart diagram* menggambarkan *class* tertentu (satu *class* dapat memiliki lebih dari satu *statechart diagram*).

4. Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.



Gambar 2.3. *Activity Diagram*

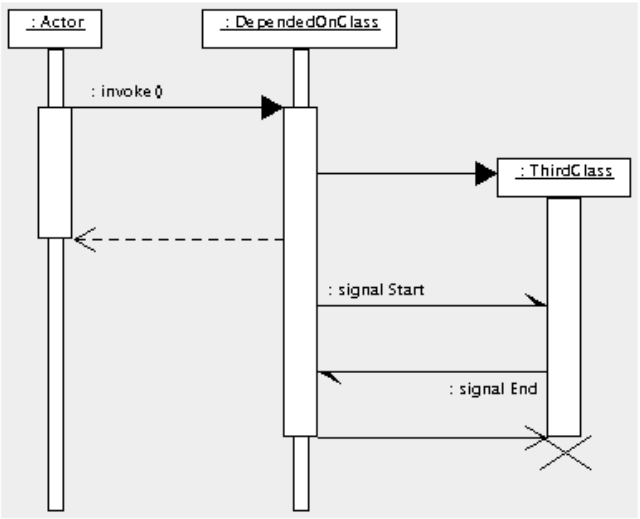
(Sumber : http://argouml-stats.tigris.org/documentation/ manual-0.32/ch25.html/

12 September 2012)

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.

5. Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).



Gambar 2.4. *Sequence Diagram*

(Sumber : http://argouml-stats.tigris.org/documentation/

manual-0.32/ch19.html/ 12 September 2012)

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah - langkah yang dilakukan sebagai respon dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara *internal* dan *output* apa yang dihasilkan.

1. Masing - masing objek, termasuk aktor, memiliki *lifeline* vertikal.
2. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi / metoda dari *class.*
3. *Activation bar* menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message*.
4. Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan *icon* khusus untuk objek *boundary*, *controller* dan *persistent entity.*

2.7.2 Tools yang Mendukung UML

2.7.2.1 Astah Community

Astah Community atau dikenal juga sebagai JUDE, adalah alat UML Modeling yang diciptakan oleh perusahaan Jepang Change Vision. Pengembangan JUDE pertama kali dilakukan oleh Kenji Hiranabe CEO dari Change Vision Inc pada tahun 1996. JUDE dijadikan *freeware* pada tahun 1999, dan lima tahun kemudian jude mulai dijual di pasar Jepang. *Software* ini berperan penting buat seorang perancang program/*software*. Dengan *software* ini kita bisa

membuat rancangan *software* seperti *use case,* DFD, perancangan UML, dll masih banyak lagi fitur yang ada didalamnya.

2.8 Java

2.8.1 Pengertian Java

Java adalah sebuah bahasa pemrograman pada komputer sama seperti pendahulunya c++. Bahasa pemrograman ini dikembangkan oleh Sun microsystems yang dikembangkan pada tahun 1995. Java pada awalnya dikembangkan pada lingkungan komputer dengan tujuan untuk menghasilkan suatu bahasa komputer sederhana tanpa harus terikat pada arsitektur tertentu. Perkemabangan Java semakin pesat sekarang ini dengan berbagai macam *platform* sistem operasi seperti Windows, Linux, maupun Unix, Mac OS, Sun Solaris.

Seperti diutarakan di atas, Java merupakan bahasa pemrograman yang dapat diimplementasikan pada berbagai lingkungan, seperti internet, telepon genggam maupun komputer. Bahasa pemrograman Java secara garis besar dikelompokan menjadi tiga edisi, yaitu, Java 2 Standar Edition (J2SE), Java 2 Enterprise Edition (J2EE) dan Java 2 Micro Edition (J2ME).

2.8.1.1 J2EE (Java 2 Enterprise Edition)

J2EE adalah kelompok dari beberapa API dari Java dan teknologi selain Java. J2EE dibuat untuk membuat aplikasi yang kompleks. J2EE

sering dianggap sebagai *middleware* atau teknologi yang berjalan di *server*, namun sebenarnya J2EE tidak hanya terbatas untuk itu.

2.8.1.2 J2SE (Java 2 Second Edition)

J2SE adalah inti dari bahasa pemrograman Java. JDK (Java Development Kit) adalah salah satu tool dari J2SE untuk mengkompilasi dan menjalankan program Java. Di dalamnya terdapat tool untuk mengkompilasi program Java dan JRE. Tool J2SE yang salah satunya adalah jdk1.5, jdk merupakan tool open source dari Sun.

2.8.1.3 J2ME (Java 2 Micro Edition)

2.8.1.3.1 Pengertian J2ME

Java Micro Edition atau yang disebut J2ME adalah lingkungan pengembangan yang didesain untuk meletakan perangkat lunak Java pada barang elektronik beserta perangkat pendukungnya. Pada J2ME, jika perangkat lunak berfungsi baik pada sebuah perangkat maka belum tentu juga berfungsi baik pada perangkat lainnya. J2ME membawa Java ke dunia informasi, komunikasi, dan perangkat komputasi selain perangkat komputer desktop yang biasanya lebih kecil dibandingkan perangkat komputer desktop. J2ME biasa digunakan pada telepon selular, *pager*, dan sejenisnya.

Teknologi J2ME juga memiliki beberapa keterbatasan, terutama jika diaplikasikan pada ponsel. J2ME sangat tergantung pada perangkat (*device*) yang digunakan, bisa dari segi merk ponsel, maupun kemampuan ponsel, dan

dukungan terhadap teknologi J2ME. Misalnya, jika sebuah ponsel tidak memiliki kamera maka jelas J2ME pada ponsel tersebut tidak bisa mengakses kamera. Keterbatasan lainnya adalah padaukuran aplikasi, karena memori pada ponsel sangat terbatas. Sebagian besar ponsel tidak mengijinkan aplikasi J2ME menulis pada file karena alasan keamanan.

2.8.1.3.2 Connected Limited Device Configuration (CLDC)

CLDC atau *Connected Limited Device Configuration* adalah perangkat dasar dari J2ME, spesifikasi dasar yang berupa *library* dan API yang diimplementasikan pada J2ME, seperti yang digunakan pada telepon selular, *pager*, dan PDA. Perangkat tersebut dibatasi dengan keterbatasan memori, sumber daya, dan kemampuan memproses. Spesifikasi CLDC [ada J2ME adalah spesifikasi minimal dari *package*, kelas, dan sebagian fungsi *Java Virtual Machine* yang dikurangi agar dapat diimplementasikan dengan keterbatasan sumber daya pada alat-alat tersebut, JVM yang digunakan disebut KVM (*Kilobyte Virtual Machine*).

2.8.1.3.3 Connected Device Configuration (CDC)

CDC adalah spesifikasi dari konfigurasi J2ME. CDC merupakan komunitas proses pada Java yang memiliki standarisasi. Implementasi CDC pada J2ME adalah *source code* yang menyediakan sambungan dengan macam-macam *platform.*

2.8.1.3.4 Mobile Information Device Profile (MIDP)

MIDP adalah spesifikasi untuk sebuah profil J2ME. MIDP memiliki lapisan di atas CLDC, API tambahan untuk daur hidup aplikasi, antarmuka, jaringan, dan penyimpanan persisten. MIDP berguna untuk memainkan *tone, tone squence* dan file WAV.

2.8.1.3.5 Kilobyte Virtual Machine (KVM)

KVM adalah paket JVM yang didesain untuk perangkat yang kecil. KVM mendukung sebagian dari fitur-fitur JVM, seperti misalkan KVM tidak mendukung operasi *floating-point* dan finalisasi objek. KVM diimplementasikan menggunakan C sehingga sangat mudah beradaptasi pada tipe *platform* yang berbeda.

2.8.1.3.6 C-Virtual Machine (CVM)

CVM adalah paket JVM optimal yang digunakan pada CDC. CVM mempunyai seluruh fitur dari *virtual machine* yang didesain untuk perangkat yang memerlukan fitur-fitur *Java 2 Virtual Machine*.

2.8.1.3.7 MIDlet

MIDlet adalah aplikasi yang ditulis untuk MIDP. Aplikasi MIDlet adalah bagian dari kelas ’javax.microedition.midlet.MIDlet’ yang didefinisikan pada MIDP. MIDlet berupa sebuah kelas abstrak yang merupakan subkelas dari

bentuk dasar aplikasi sehingga antarmuka antara aplikasi J2ME dan aplikasi manajemen pada perangkat dapat terbentuk.

2.8.1.3.8 JAD (Java Application Descriptor)

Digunakan untuk mendeskripsikan isi aplikasi untuk keperluan pemetaan. File JAD berisi deskripsi file JAR (*Java Archive*) dan pemetaan atribut MIDlet, sedangkan file JAR berisi kumpulan dan *resource.*

2.8.1.3.9 OTA (Over The Air)

OTA megnacu pada beberapa teknologi jaringan tanpa kabel. Dengan menggunakan OTA, *provider* MIDlet dapat menginstal MIDlet pada *web server* dan menyediakan *link* untuk men-download via WAP atau *internet microbrowser.*

2.8.2 Keuntungan Menggunakan Java

1. Sederhana

Kesederhanaan adalah kelebihan utama yang ditawarkan Java. Hal ini dimungkinkan karena *Java Scripting Language* (bahasa naskah) menghilangkan kemampuan kita untuk berinovasi dengan menganggap bahwa semua yang kita inginkan telah disatukan dalam satu objek yang telah ada dan seringkali perlu di-script-kan. Java memberi anda kemampuan untuk menuangkan semua ide anda dalam cara berorientasi-objek yang mudah dan jelas tanpa harus melihat proses bagian dalam sistem yang seringkali berbahaya.

2. Aman

Kebanyakan masalah keamanan terutama aplikasi jaringan *internet* yang dihasilkan dari sistem lama terjadi karena sistem tidak dirancang dengan konsep keamanan *internet*. Salah satu prinsip kunci perancangan Java adalah keselamatan dan keamanan. Java tidak pernah memiliki fasilitas dan keamanan yang tidak aman sampai perlu ditangani secara khusus untuk pengamanannya.

Oleh karena itu, program Java tidak dapat memanggil fungsi-fungsi global dan memperoleh akses ke berbagai sumber dalam sistem karena terdapat sejumlah pengawasan yang dilakukan oleh program Java yang tidak dapat dilakukan oleh sistem lain.

3. Berorientasi Objek

Java tidak diturunkan bahasa pemrograman manapun, juga sama sekali tidak kompatibel dengan semuanya. Hal ini dikarenakan adanya kebebasan dalam rancangan maka dipilih pendekatan yang jelas berguna, dan pragmatis. Modul objek Java adalah sederhanadan mudah dikembangkan namun sejalan dengan itu, bilangan dan tipe data sederhana lain dianggap sebagai non objek berkinerja tinggi.

Kebanyakan sistem berorientasi objek lain memilih hirarki objek yang kaku dan susah diatur atau memilih menggunakan model objek dinamik yang tidak memiliki kinerja tinggi dan kelengkapan. Java sekali lagi memiliki keseimbangan yang menyediakan mekanisme peng-class-an sederhana dengan model antarmuka dinamik yang intuitif hanya jika diperlukan. Memahami gaya

pemrograman berorientasi objek sangat penting dan membantu mempelajari bagaimana membuat program dengan Java.

4. Kokoh

Java membatasi anda dengan memberi beberapa kunci supaya anda dapat menemukan kesalahan lebih cepat saat mengembangkan program. Dengan demikian, anda tidak perlu mencemaskan banyaknya penyebab kesalahan pemrograman yang banyak terjadi pada bahasa lain.

Java langsung memeriksa program saat anda menuliskannya dan sekali lagi program akan dijalankan. Banyaknya kesalahan yang susah dilacak dan situasi jalannya program yang tidak dapat diulangi dapat dibuat secara sederhana dengan Java. Kemampuan untuk memastikan apa yang telah kita tulis akan berjalan semestinya dalam kondisi apapun adalah salah satu kemampuan penting Java.

5. Terdistribusi

Java memudahkan pembuatan aplikasi terdistribusi dengan sekumpulan kelas yang digunakan pada aplikasi-aplikasi jaringan. Dengan menggunakan kelas URL (*Uniform Resource Locator*) Java, suatu aplikasi dapat dengan mudah mengakses server-jauh (*remote server*).

6. Portable

Salah satu tujuan penting pembuatan Java adalah kode-kode Java haruslah *Portable*, sehingga ketika arsitektur baru (baik itu perangkat keras

ataupun sistem operasi atau keduanya) berkembang, lingkungan Java dapat diterapkan dan dipindahkan pada mereka. Pada Java, semua tipe data primitif (*integer, long, float, double* dan sebagainya) memiliki ukuran tertentu, tidak bergantung pada mesin atau sistem operasi dimana program Java dijalankan. Hal ini sangat berlawanan dengan bahasa seperti C atau C++ yang menyerahkan ukuran tipe data primitif pada kompiler dan mesin (serta sistem operasi). Java *portable* karena kompiler Java sendiri ditulis menggunakan Java.

7. Terinterpretasi

Java melengkapi keajaiban lintas *platform* yang luar biasa dengan kompilasi ke dalam representasi langsung yang disebut kode-byte Java (*Java byte code*) yang dapat diterjemahkan oleh sistem apapun yang memiliki program Java di dalamnya. Kebanyakan sistem tersebut menawarkan keajaiban lintas *platform* ini terseok pada satu hal: kinerja. Sistem lain yang netral terhadap *platform* juga adalah sistem terinterpretasi misalnya BASIC, Tcl dan Perl.

2.9 Perangkat Lunak yang Digunakan

Untuk memulai pemrograman *mobile,* memerlukan suatu *tool* pengembangan khusus. Salah satu *tool* yang paling populer untuk pemrograman berbasis J2ME adalah *Wireless Tool Kit* (WTK) yang secara resmi dikeluarkan oleh pihak Sun Microsystem agar para pengembang teknologi *mobile* dapat dengan mudah membangun aplikasi diatasnya.

2.9.1 Wireless Tool Kit

J2ME *Wireless Tool Kit* (WTK) adalah kakas yang menyediakan lingkungan *emulator*, dokumentasi beserta contoh-contoh aplikasi Java untuk perangkat kecil (*small device*). *Wireless Tool Kit* berbasiskan pada CLDC dan MIDP. *Wireless Tool Kit* adalah program yang meniru kerja ponsel yang mendukung MIDP atau yang biasa di sebut *emulator*. Oleh karena itu, belum tentu MIDlet yang berjalan di emulator juga berjalan pada ponsel yang sebenarnya, karena juga bergantung pada kemampuan dan kapasitas ponsel yang digunakan.

2.9.2 NetBeans

Netbeans merupakan sebuah aplikasi *Integrated Development Environment* (IDE) yang berbasiskan Java dari Sun Microsystems yang berjalan di atas *swing*. *Swing* merupakan sebuah teknologi Java untuk pengembangan aplikasi desktop yang dapat berjalan pada berbagai macam *platform* seperti windows, linux, Mac OS X dan Solaris. Sebuah IDE merupakan lingkup pemrograman yang di integrasikan ke dalam suatu aplikasi perangkat lunak yang menyediakan *Graphic User Interface* (GUI), suatu kode *editor* atau *text*, suatu *compiler* dan suatu *debugger*.

Platform NetBeans adalah *framework* yang dapat digunakan kembali *(reusable)* untuk menyederhanakan pengembangan aplikasi desktop. Ketika aplikasi berbasis *platform* NetBeans dijalankan, kelas Main dari *platform* dieksekusi. Modul-modul yang tersedia ditempatkan di sebuah *registry* di dalam

memori, dan tugas startup modul dijalankan. Secara umum, kode modul dimuatkan ke dalam memori hanya ketika ia diperlukan.

NetBeans IDE adalah IDE *open source* yang ditulis sepenuhnya dengan bahasa pemrograman Java menggunakan *platform* NetBeans. NetBeans IDE mendukung pengembangan semua tipe aplikasi Java (J2SE, web, EJB, dan aplikasi *mobile)*. Fitur lainnya adalah sistem proyek berbasis Ant, kontrol versi, dan *refactoring*.

NetBeans *Mobility Pack* adalah alat untuk mengembangkan aplikasi yang berjalan pada perangkat bergerak (*mobile*), umumnya telepon seluler, tetapi juga mencakup PDA, dan lain-lain. NetBeans *Mobility Pack* digunakan untuk menulis, menguji, dan debugging aplikasi untuk perangkat bergerak yang menggunakan teknologi berplatform *Java Micro Edition* (*platform* Java ME). Paket ini mengintegrasikan dukungan terhadap *Mobile Information Device Profile* (MIDP) 2.0, *Connected Limited Device Configuration* (CLDC) 1.1, dan *Connected Device Configuration* (CDC). *Emulator* dari pihak ketiga dapat diintegrasikan dengan mudah untuk lingkungan pengujian yang lebih kokoh. NetBeans *Mobility Pack* saat ini tersedia dalam dua klaster yang berbeda, yang satu memuat CDC dan yang lainnya CLDC.

2.10 Symbian

Symbian adalah sebuah sistem operasi dan *platform software* yang didesain untuk ponsel pintar dan saat ini diawasi oleh Nokia. *Platform* symbian adalah suksesor daripada symbian OS dan Nokia Seri 60 yang sebelumnya sudah pernah beredar di pasaran. Tidak seperti symbian OS, yang membutuhkan beberapa sistem antarmuka pengguna, symbian menyertakan sebuah komponen antarmuka pengguna berdasarkan S60 5th Edition. Versi terakhirnya yaitu symbian v3 sudah secara resmi dirilis pada kuartal 4 tahun 2010 lalu, pertama kali digunakan pada ponsel Nokia N8.

Symbian OS aslinya dikembangkan oleh Symbian Ltd.. Ini merupakan keturunan dari EPOC dari Psion dan berjalan secara eksklusif di prosesor ARM, walaupun telah ada x86 port yang belum dirilis resmi ke publik. Device berbasis symbian telah mencakup 29.2% pangsa pasar ponsel pintar dunia pada kuartal pertama tahun 2011. Beberapa perkiraan mengindikasikan bahwa jumlah kumulatif dari ponsel-ponsel yang dipaketkan dengan symbian OS ini hingga akhir kuartal 2 tahun 2010 lalu berjumlah 385 juta. Hingga 5 April 2011, Nokia merilis symbian di bawah lisensi baru dan dikonversikan ke model sumber kepemilikan yang dibagikan, berlawanan dari proyek kode sumber terbuka atau *open source*.

Symbian OS mampu melakukan operasi secara *multithreading*, *multitasking* dan pengamanan terhadap memori dan semua pemrograman pada

symbian dilakukan secara *event-based*, artinya *hardware* CPU menjadi tidak aktif ketika tidak ada inputan berupa aktivitas tertentu.

Symbian banyak ditemukan dalam telepon seluler keluaran Nokia. Selain itu ia juga digunakan dalam beberapa modelt elepon Nokia, Samsung, Fujitsudan DoCoMo.