BAB II

LANDASAN TEORI

Pada bab ini akan diuraikan mengenai teori dalam pemecahan masalah yang berkaitan dengan pembuatan aplikasi ini .

2.1 Sejarah singkat Jawa Barat

Jawa Barat adalah sebuah provinsi di Indonesia. Ibu kotanya berada di Kota Bandung. Perkembangan Sejarah menunjukkan bahwa Provinsi Jawa Barat merupakan Provinsi yang pertama dibentuk di wilayah Indonesia. Provinsi Jawa Barat dibentuk berdasarkan UU No.11 Tahun 1950, tentang Pembentukan Provinsi Jawa Barat. Jawa Barat merupakan provinsi dengan jumlah penduduk terbanyak di Indonesia. Bagian barat laut provinsi Jawa Barat berbatasan langsung dengan Daerah Khusus Ibukota Jakarta, ibu Kota negara Indonesia. Pada tahun 2000, Provinsi Jawa Barat dimekarkan dengan berdirinya Provinsi Banten, yang berada di bagian barat

Provinsi Jawa Barat secara geografis terletak di antara 5°50' - 7°50' LS dan 104°48' - 104°48 BT. Luas wilayah Provinsi Jawa Barat Barat pada tahun 2008 adalah 34.816,96 Km2, terdiri atas 16 Kabupaten dan 9 Kota. Secara administrasi batas-batas Provinsi Jawa Barat adalah sebagai berikut:

1. Utara : Laut Jawa
2. Timur : Jawa Tengah
3. Selatan : Samudra Hindia

4. Barat : DKI Jakarta dan Provinsi Banten

Sebagian besar wilayah kabupaten/kota di Jawa Barat berbatasan dengan laut, sehingga Wilayah Jawa Barat memiliki garis pantai cukup panjang, yaitu 755,83Km.

Jawa Barat memiliki iklim tropis, selama ini suhu terendah tercatat 9o C yaitu di Puncak Gunung Pangrango dan suhu tertinggi tercatat 34oC di daerah pantai utara. Tetapi pada bulan Oktober 2008 yang baru saja berlalu, suhu di Jawa Barat sempat mencapai 35 oCelcius selama 3 – 4 pekan lamanya yang hampir merata dialami oleh seluruh daerah di Jawa Barat. Curah hujan rata-rata tahunan di Jawa Barat mencapai 2.000 mm/tahun, namun di beberapa daerah pegunungan bisa mencapai 3.000 - 5.000 mm/tahun.

Proses geologi yang terjadi jutaan tahun lalu menyebabkan Provinsi Jawa Barat – dengan luas 3,7 juta hektar- terbagi menjadi sekitar 60 % daerah bergunung dengan ketinggian antara 500–3.079 meter dpl dan 40 % daerah dataran yang memiliki variasi tinggi antara 0–500 meter dari permukaan laut . Wilayah pegunungan umumnya menempati bagian tengah dan selatan Jawa Barat. Pada bagian tengah dapat ditemukan gunung-gunung berapi aktif seperti Gunung. Salak (2.211 m), Gede-Pangrango (3.019 m) , Ciremai (3.078 m) dan Tangkuban Perahu (2.076) berpadu dengan deretan pegunungan yang sudah tidak aktif seperti Gunung Halimun (1.744 m), Gn. Ciparabakti (1.525 m) dan Gn. Cakrabuana (1.721 m). Demikian pula halnya di wilayah selatan, gunung-gunung berapi masih umum dijumpai seperti Gunung Galunggung (2.168 m), Papandayan (2.622 m),

dan Guntur (2.249 m); bersama deretan pegunungan yang sudah tidak aktif seperti pegunungan selatan Jawa. Keadaan sebaliknya dijumpai di wilayah utara Jawa Barat yang merupakan daerah dataran sedang hingga rendah dengan didominasi oleh dataran aluvial. Daerah daratan Jawa Barat dapat dikelompokkan menjadi beberapa karakter sebagai berikut:

1. Daerah pegunungan curam di bagian selatan dengan ketinggian > 1.500 m dpl,
2. Daerah lereng bukit landai di bagian tengah dengan ketinggian 100-1.500 m dpl.
3. Daerah dataran rendah yang luas di bagian utara dengan ketinggian 0-10 m dpl.

Secara geologis daratan Jawa Barat merupakan bagian dari busur kepulauan gunung api (aktif dan tidak aktif) yang membentang dari ujung utara Pulau Sumatera hingga ujung utara Pulau Sulawesi.

Jumlah penduduk Jawa Barat pada tahun 2007 adalah sebesar 41.483.729 jiwa. Pertumbuhan penduduk Jawa Barat termasuk tinggi dibandingkan dengan provinsi lain di Indonesia. Dalam kurun waktu 2003 – 2007 telah mengalami pertumbuhan rata-rata sebesar 2, 13% / tahun.

Tingkat pertumbuhan yang cukup tinggi terjadi di Kab. Bogor dan Kab. Tasikmalaya. Sebetulnya pertumbuhan penduduk di daerah sekitar Kota Bandung (tadinya termasuk dalam wilayah Kabupaten Bandung) termasuk dalam kategoori

tinggi, namun karena adanya pemecahan wilayah Kabupaten Bandung menjadi Kabupaten Bandung dan Kabupaten Bandung Barat, maka secara numerik tingkat pertumbuhan penduduk di Kabupaten Bandung menjadi kecil.

2.1.1 Pembagian Administratif Jawa Barat

Wilayah Propinsi Jawa Barat meliputi areal seluas 43.177 km² terdiri dari 25 Kabupaten/ Kota,yaitu 16 Kabupaten dan 9 Kota, yang masing-masing mempunyai ciri-ciri khususbaik dari segi sumberdaya alam maupun sumberdaya manusia.

1. Jumlah Kabupaten = 17 Kabupaten
2. Jumlah Kota = 9 Kota
3. Jumlah Kecamatan = 618 Kecamatan
4. Jumlah Kelurahan = 1.859 Kelurahan
5. Jumlah Desa = 4.004 Desa

Sumber Data: Podes 2008 BPS – Jawa Barat

Berdasarkan Fisiografi Jawa Barat dapat distratifikasikan kedalam 4 strata wilayahPembangunan Pertanian yaitu:

1. Wilayah dataran rendah sebelah utara, yaitu wilayah yang terutama berpokok kepadausahatani sawah dengan hasil utama padi meliputi kabupaten Karawang, Indramayu, Subang, Bekasi dan Cirebon .
2. Wilayah dataran tinggi bagian tengah, dimana keadaan usaha padi sawah dan darat hampir berimbang keadaannya, dengan hasil yang beraneka

ragam seperti padi, palawija,sayur-sayuran dan tanaman hias meliputi kabupaten Bandung, Sumedang, Purwakarta, Garut, Kuningan , Bogor , Cianjur .

3. Wilayah Banten bagian tengah dan selatan dimana pertanian pada lahan kering (darat) lebih dominan, sawah lebih sedikit dan pertanian pada lahan kering terutamakomoditas buah-buahan meliputi Serang, Pandeglang dan Lebak.
4. Wilayah Jawa Barat selatan, dimana didaerahnya bergunung-gunung dengan sifat tanahnya yang tidak mungkin untuk dikembangkan dengan usaha tani yang pengolahan tanah nya dilakukan secara intensif meliputi Kabupaten Sukabumi, Ciamis, Tasikmalaya.

2.1.2 Daerah Administratif Provinsi Jawa Barat

Jawa Barat terdiri atas 17 kabupaten dan 9 kota. Kota-kota hasil pemekaran sejak tahun 1996 adalah :

1. Kota Bekasi, dimekarkan dari Kabupaten Bekasi pada tahun 1996
2. Kota Depok, dimekarkan dari Kabupaten Bogor pada tahun 1999
3. Kota Cimahi, dimekarkan dari Kabupaten Bandung pada tahun 2001
4. Kota Tasikmalaya, dimekarkan dari Kabupaten Tasikmalaya pada tahun 2001
5. Kota Banjar, dimekarkan dari Kabupaten Ciamis pada tahun 2002

6. Kabupaten Bandung Barat, dimekarkan dari Kabupaten Bandung tahun 2007
7. Kabupaten Pangandaran, dimekarkan dari Kabupaten Ciamis tahun 2012

No.	Kabupaten/Kota	Ibu kota
1	Kabupaten Bandung	Soreang
2	Kabupaten Bandung Barat	Ngamprah
3	Kabupaten Bekasi	Cikarang
4	Kabupaten Bogor	Cibinong
5	Kabupaten Ciamis	Ciamis
6	Kabupaten Cianjur	Cianjur
7	Kabupaten Cirebon	Sumber
8	Kabupaten Garut	Tarogong Kidul
9	Kabupaten Indramayu	Indramayu
10	Kabupaten Karawang	Karawang
11	Kabupaten Kuningan	Kuningan

12	Kabupaten Majalengka	Majalengka
13	Kabupaten Pangandaran	Parigi
14	Kabupaten Purwakarta	Purwakarta
15	Kabupaten Subang	Subang
16	Kabupaten Sukabumi	Pelabuhanratu
17	Kabupaten Sumedang	Sumedang
18	Kabupaten Tasikmalaya	Singaparna
19	Kota Bandung	-
20	Kota Banjar	-
21	Kota Bekasi	-
22	Kota Bogor	-
23	Kota Cimahi	-
24	Kota Cirebon	-
25	Kota Depok	-
26	Kota Sukabumi	-

27	Kota Tasikmalaya	-

Tabel 2.1 Daftar Kab / Kota Provinsi Jawa Barat

2.3 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.

2.4 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 computer 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 mereka 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.

2.5 Pengertian Informasi

Menurut: Fatta (20079), 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.

2.5.1 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.

2.5.2 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 bahan 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.
6. 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.

2.6 Metode Pendekatan Sistem

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

2.6.1. 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 system 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

2.6.2. 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 memerika 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.

2.6.2.1 Objek (*Object)*

Objek adalah benda secara fisik dan konseptual yang ada di sekitar kita. Sebuah objek memiliki keadaan sesaat yang disebut state.

1. State dari sebuah objek adalah kondisi dari objek atau himpunan keadaan yang menggambarkan objek tersebut. State dinyatakan dengan nilai dari atribut objeknya.
2. Atribut adalah nilai internal suatu objek yang mencerminkan karakteristik objek, kondisi sesaat, koneksi dengan objek lain dan identitas.
3. 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.

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

2.6.2.2 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.

2.6.2.3 Kotak Hitam (*Black Box)*

Sebuah objek adalah kotak hitam. Konsep ini menjadi dasar implementasi objek. Dalam operasi OO 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)

2.6.2.4 Asosiasi dan Agregasi

1. 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.
2. 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.

2.6.3 Pengertian Object

Object 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, pelanggan, guru, dosen, mahasiswa, murid.
3. Gudang, kantor, bangunan, ruangan.
4. Kendaraan, produk, konputer, video.

2.6.4 Karakteristik Object

1. Identitas berarti bahwa data diukur mempunyai nilai tertentu yang membedakan entitas disebut Objek.
2. Objek dapat kongkrit, seperti halnya arsip dalam sistem, atau konseptual seperti kebijakan penjadualan dalam multiprocessing pada sistem operasi.
3. Setiap objek mempunyai sifat yang melekat pada identitasnya.
4. Dua objek dapat berbeda walaupun bila semua nilai atributnya identik.

2.6.4.1 Kelas Object

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.

2.6.5 Metodologi Berorientasi Objek

Metodologi adalah cara systematis untuk mengerjakan analisys and design. Dengan metodologi, pihak yang membangun system 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.

2.6.5.1 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.

2.7 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 menggambrakan transformasi nilai data di dalam sistem.
2. Model fungsional berisi data flow diagram. DFD adalah suatu *graph* dimana *node*nya menyatakan proses dan *arc*nya adalah aliran data.

2.8 Alat Bantu Analisis dan Perancangan

2.8.1 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.

2.8.1.1 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.

Use case diagram dapat sangat membantu bila kita sedang menyusun requirement sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang test case untuk semua feature yang ada pada sistem.

1. Sebuah use case dapat meng-include fungsionalitas use case lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa use case yang di- include akan dipanggil setiap kali use case yang meng-include dieksekusi secara normal.
2. Sebuah use case dapat di-include oleh lebih dari satu use case lain, sehingga duplikasi fungsionalitas dapat dihindari dengan Cara menarik keluar fungsionalitas yang common.
3. Sebuah use case juga dapat meng-extend use case lain dengan behavior- nya sendiri.
4. Sementara hubungan generalisasi antar use case menunjukkan bahwa use case yang satu merupakan spesialisasi dari yang lain



Gambar 2.1. *Use Case Diagram*

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

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. Class memiliki tiga area pokok :

1. Nama (dan stereotype)

2. Atribut

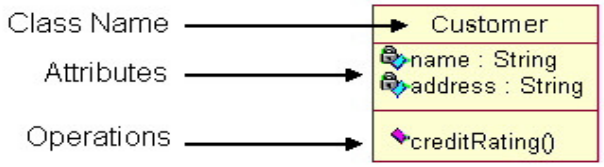
3. Metoda

Untuk 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



Gambar 2.2. Bagian-bagian dari *class*

4. Activity Diagram

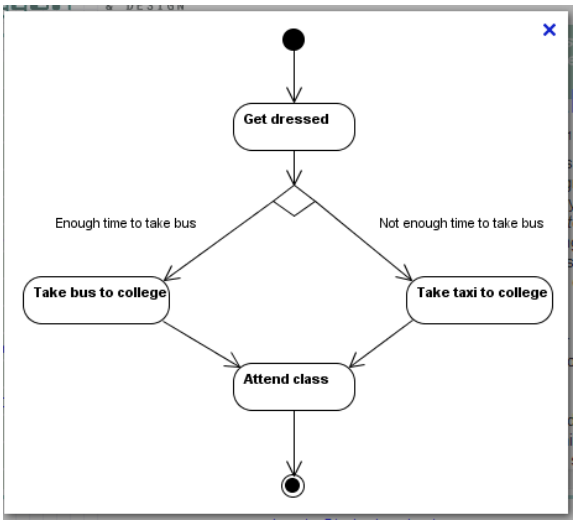
Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

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.

Sama seperti state, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. Decision digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (fork dan join) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal.

Activity diagram dapat dibagi menjadi beberapa object swimlane untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.



Gambar 2.3. *Activity Diagram*

(Sumber:http://www.albertoaden.wordpress.com/ msf-fase-planning-planning-phase/ 04 juli 2012)

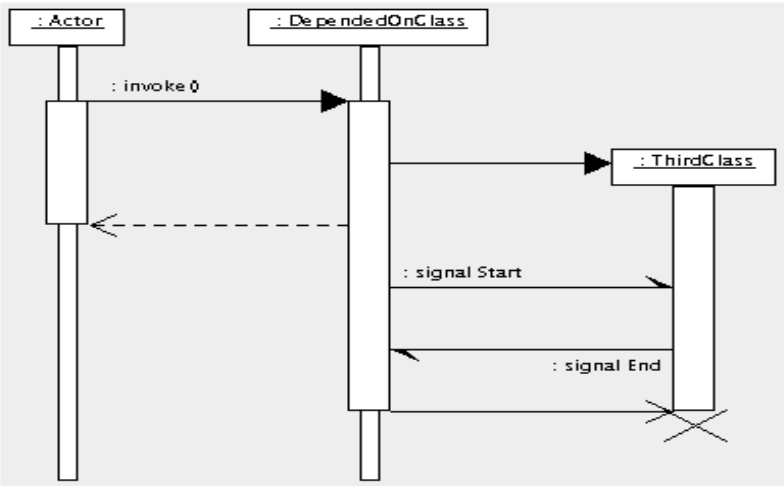
5. Sequence Diagram

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).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons 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 massage.
4. Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan icon khusus untuk objek boundary, controller dan persistent entity.



Gambar 2.4. *Sequence Diagram*

(Sumber: http://argouml-stats.tigris.org/documentation/manual-0.32/ch19.html/ 04 juli 2012)

2.8.2 Tools Yang Mendukung UML

2.8.2.1 Rational Rose

Dalam penelitian ini, *Rational Rose* berfungsi sebagai *tool* untuk pemodelan sistem yang menggambarkan proses-proses yang ada pada sistem ini. *Rational Rose* adalah kakas (*tools)* pemodelan visual

untuk pengembangan system berbasis objek yang sangat handal untuk digunakan sebagai bantuan bagi para pengembang dalam melakukan analisis dan perancangan sistem.

Rational Rose digunakan untuk melakukan pemodelan sistem sebelum pengembang menulis kode-kode dalam bahasa pemrograman tertentu. Ia juga membantu analisis system dengan cara pengembang membuat diagram *use case* untuk melihat fungsionalitas sistem secara keseluruhan sesuai dengan harapan dan keinginan pengguna. Kemudian, ia juga menuntut pengembang untuk mengembangkan *Interaction* diagram untuk melihat bagaimana objek-objek saling bekerjasama dalam menyediakan fungsionalitas yang diperlukan (Adi Nugroho, 2005:20).

2.9 Java

2.9.1 Tentang Java

Java menurut definisi dari sun adalah nama untuk sekumpulan tekhnologi untuk membuat dan menjalankan perangkat lunak pada komputer standalone ataupun pada lingkungan jaringan. Java2 adalah generasi kedua dari java platform (generasi awalnya adalah Java Development Kit). Java berdiri di atas sebuah mesin interpreter yang diberi nama Java Virtual Machine (JVM). JVM inilah yang akan membaca bytecode dalam file .class dari suatu program sebagai representasi langsung program yang berisi bahasa mesin. Oleh karena itu , bahasa Java disebut sebagai bahasa pemrograman

yang portable karena dapat dijalankan pada berbagai sistem operasi, asalkan pada sistem operasi tersebut terdapat JVM.

Platform Java terdiri dari sekumpulan library, JVM, kelas-kelas loaderyang dipaket dalam sebuah lingkungan rutin Java, dan sebuah kompiler, debuger dan kakas lain yang dipaket dalam Java Development Kit (JDK). Java2 adalah generasi yang sekarang sedang berkembang dari platform Java. Agar sebuah program Java dapat dijalankan, maka file dengan ekstensi .java harus dikompilasi menjadi file bytecode. Untuk menjalankan bytecode tersebut dibutuhkan JRE (Java Runtime Environment) yang memungkinkan pemakai untuk menjalankan program Java, hanya menjalankan, tidak untuk membuat kode baru lagi.

Platform Java memiliki tiga buah edisi yang berbeda, yaitu J2EE (Java Enterprise Edition), J2ME (Java2 Micro Edition), dan J2SE (Java2 Second Edition).

2.9.1.1 J2EE (Java Enterprise Edition)

J2EE adalah kelompok dari beberapa API dari Java dan tekhnologi 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.

2.9.1.2 J2SE (Java2 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

2.9.1.3 J2ME (Java2 Micro Edition)

2.9.1.3.1 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, personal digital assistants (PDA's) 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.

2.9.1.3.2 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).

2.9.1.3.3 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

2.9.1.3.4 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.

2.9.1.3.5 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.

2.9.1.3.6 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.

2.9.1.3.7 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.

2.9.1.3.8 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.

2.9.1.3.9 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.

2.9.2 Keuntungan Memakan Java

1. Sederhana

Inti dari ke-"sederhana"-an Java terutama terletak pada kemiripannya dengan C dan C++. Karena programmer-programmer masa kini khususnya yang tertarik menggunakan Java telah memiliki pengalaman setidaknya dengan C dan mungkin dengan C++, Java tentu saja terlihat sederhana dan akrab bagi programmer- programmer ini.

Java menyederhanakan bahasa C++ dengan menambahkan fitur-fitur pendukung yang belum terdapat dalam C++ dan membuang beberapa fitur yang membuat C++ menjadi bahasa yang rumit dan sulit untuk dikuasai. Java sederhana karena hanya ia memiliki tiga tipe angka data primitif, tipe Boolean dan array. Selebihnya, semua dalam Java adalah sebuah kelas. Salah satu fitur penting yang ditawarkan Java (yang belum ada pada C++) adalah pengumpulan sampah (garbage collection). Dengan mekanisme ini, user tidak perlu membebaskan memori yang dialokasikan, karena semua dilakukan oleh Mesin Virtual Java. Java juga mendukung penulisan program multi jalinan, yaitu suatu program yang dapat melakukan lebih dari satu pekerjaan dalam waktu yang bersamaan.

2. Berorientasi Obyek

Dalam pendekatannya pada orientasi-obyek, Java lebih merujuk pada SmallTalk daripada C++. Selain tipe data primitive-nya, semua yang ada pada Java adalah kelas. Sebaliknya dalam C++ semuanya serba campur-aduk, dimana programmer dapat secara bebas (benar-benar bebas)dalam mencampur kode-kode orientasi obyek (kelas) (kelas) dengan kode prosedural (fungsi). Dalam Java semua ini tidak diperbolehkan. Tidak ada fungsi global dalam Java, semua fungsi harus dipanggil melalui sebuah obyek.

Dukungan Java terhadap orientasi-obyek tidak termasuk penurunan (inheritansi) ganda. Para perancang Java merasa kerumitan yang dimunculkan inheritansi ganda tidak sebanding dengan keuntungan yang dihasilkan. Sebagai

ganti dari inheritansi ganda, Java kemudian menyediakan antarmuka (interface).

3. 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).

4. Terinterpretasi

Karena Java adalah bahasa interpretasi, sekali Interpreter Java terpasang, terinstal pada suatu mesin tertentu, mesin tersebut dapat secara langsung menjalankan aplikasi-aplikasi Java (tidak peduli platform atau sistem operasi apa yang terpasang pada mesin tersebut). Ketika menggunakan bahasa interpreter, programmer juga terbebas dari kekhawatiran berkaitan dengan ketergantungan antar modul. Keuntungan lain adalah waktu yang digunakan untuk lingkaran edit-compile-link- test dapat dipotong. Tanpa adanya langkah kompile dan link, maka bekerja dalam ingkungan interpreter lebih sederhana dan lebih hemat waktu karena hanya melalui lingkaran edit-test.

5. Kokoh

Membuat suatu program yang terdistribusi, mendukung multi-jalinan yang dapat dijalankan pada berbagai sistem operasi dan berbagai prosesor bukanlah suatu pekerjaan yang mudah. Berdasarkan pemikiran ini, maka Java

diciptakan sebagai bahasa yang sangat ketat dalam penulisan (strongly typed language). Dalam Java management memori telah disederhanakan dengan dua cara. Pertama Java tidak mendukung manipulasi pointer atau aritmatik secara langsung, sehingga mustahil bagi program Java untuk menumpuk (meng-overwrite) memori atau mengkorupsi data. Kedua Java menggunakan mekanisme pengumpulan sampah saat program berjalan (runtime) daripada pembebasan memori secara eksplisit.

6. Aman

Karena Java tidak menggunakan pointer yang secara langsung merujuk pada lokasi memori seperti yang terdapat pada C atau C++, Java memiliki kendali penuh terhadap semua kode yang ada pada lingkungan Java. Sebelumnya telah diantisipasi bahwa aplikasi-aplikasi Java akan dijalankan di internet dan akan secara dinamis dapat dijalankan bersama atau mengeksekusi program lain di tempat yang berlainan melalui internet, maka para pengembang Java berpendapat tentang perlu adanya kompiler Java yang akan menghasilkan kode bite Java yang akan melewati standar keamanan runtime Java. Gagasan ini memunculkan ide tentang pemeriksa kode byte yang akan memeriksa semua kode yang masuk dan memastikan bahwa semua kode tersebut telah mematuhi serangkaian aturan yang telah ditentukan dan aman untuk dijalankan.

7. 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.

2.10 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.

2.10.1 Wireless Tool Kit (WTK)

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.

2.10.2 NetBeans

Pengembangan NetBeans diawali dari Xelfi, sebuah proyek mahasiswa tahun 1997 di bawah bimbingan Fakultas Matematika dan Fisika Universitas Charles, Praha. Sebuah perusahaan kemudian dibentuk untuk proyek tersebut dan menghasilkan versi komersial NetBeans IDE hingga kemudian dibeli oleh Sun Microsystem pada tahun 1999. Sun kemudian menjadikan NetBeans open source pada bulan Juni tahun 2000. Sejak itu komunitas NetBeans terus berkembang.

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.

2.11 Symbian

Symbian OS adalah sistem operasi tak bebas yang dikembangkan oleh Symbian Ltd. yang dirancang untuk digunakan peralatan bergerak (mobile).

Sebelum Nokia mengumumkan pembelian seluruh sisa saham Symbian Ltd. yang tidak dimilikinya pada 24 Juni 2008,[1] Symbian dimiliki Nokia(47,9%), Ericsson (15,6%), Panasonic (10,5%),Samsung (4,5%),Siemn s BenQ (8,4%), Sony Ericsson (13,1%). Versi Symbian yang terbaru adalah

Symbian ^3 dengan PR 02 yang disebut symbian belle. Sedangkan ponsel yang paling banyak beredar saat ini menggunakan Symbian OS v6.1s, v7.0s, RV 47 75, v8.OS,v9.1s dan s60 v5. Nokia Nseries rata-rata menggunakan Symbian OS v9.1s, kecuali Nokia N95 yang menggunakan Symbian OS v9.2s. Ponsel symbian s60 v5 dan symbian ^3 rata rata menggunakan layar sentuh kecuali nokia e6-00 dan nokia e7.ponsel symbian ^3 yang pertama meluncur aadalah nokia n8 dengan kamera 12 megapiksel.symbian ^3 mendapat pembaruan pertama yang disebut symbian anna , digunakan di ponsel nokia x7-00 dan e6-00. Sedangkan pembaruan terbaru disebut symbian belle, semua ponsel symbian ^3 bisa diupgrade ke symbian belle.

Sekarang keseluruhan saham di symbian limited telah dimiliki oleh nokia dan posisi sistem operasi symbian sebagai penguasa pasar sistem operasi bergerak telah direbut oleh sistem operasi buatan google yaitu sistem operasi android yang meluncur pada akhir tahun 2009 dengan ponsel pertamanya htc dreaam g1.vendor ponsel seperti samsung, lg, sony ericsson sekarang tidak laagi menggunakan sistem operasi symbian dan menyatakan berhenti mendukung sistem operasi ini.

2.11.1 Arsitektur system Operasi Syimbia

Secara umum arsitektur Symbian OS sendiri dapat gambarkan menjadi empat lapisan berdasarkan penggunaan API yang tersedia, yaitu:

1. Lapisan pendukung aplikasi (*Application Utility Layer*) :

Lapisan ini terdiri dari berbagai pendukung yang berorientasi pada aplikasi. Hal ini memungkinkan aplikasi lain (diluar sistem operasi) untuk berintegrasi dengan aplikasi dasar yang tersedia padasistem operasi. Bentuk layanan lain termasuk proses pertukaran data dan manajemen data.

2. Lapisan layanan dan framework antarmuka grafis (*GUI Framework*) :

   Lapisan ini merupakan framework API yang tersedia untuk memberi dukungan terhadap penanganan input user secara grafis maupun suara yang dapat digunakan oleh aplikasi lain.

3. Lapisan komunikasi

   Lapisan ini berfungsi sebagai sistem operasi yang fokus diimplementasi pada peralatan komunikasi mobile. Symbian OS memiliki kumpulan API yang fokus pada lapisan komunikasi , bagian teratas pada lapisan ini terdapat dukungan pencarian dan pengiriman pesan teks. Berikutnya adalah antarmuka yang memberi dukungan komunikasi seperti Bluetooth dan infrared (IrDA) sertaUSB. Yang terakhir pada lapisan ini adalah protokol komunikasi berupa TCP/IP, HTTP, WAP dan layanan telepon

4. Lapisan Dasar API dasar

   Lapisan ini merupakan kumpulan API yang mendukung pengasksesan data memori, tanggal dan waktu, serta sistem dasar lainnya

2.11.2 Klasifikasi Sistem Operasi

Klasifikasi ini berdasar fungsionalitas dan hak akses dari API tertentu. Tujuan dari pendefinisian sistem ini selain untuk membedakan API mana saja yang bisa diakses oleh aplikasi yang dibuat oleh pihak pengembang aplikasi, juga tetap memelihara integrasi dari layanan yang disediakan bagi pihak pengembang aplikasi dengan API yang umum digunakan. Hal ini juga dilakukan untuk memaksimumkan interoperabilitas antara berbagai produk yang menggunakan Symbian OS. Terdapat empat kategori dalam klasifikasi API yang tersedia, yaitu :

2.11.2.1 (API) Symbian Umum

Komponen ini merupakan komponen (API) inti dari Symbian OS. Setiap pengembang aplikasi dapat berasumsi bahwa komponen ini terdapat pada setiap versi Symbian OS sehingga dapat digunakan pada setiap perangkat telepon bergerak yang menggunakan Symbian OS sebagai sistem operasinya. Dengan kata lain setiap kode program yang hanya menggunakan API pada kategori ini dapat dikompail

dan dijalankan tanpa kesalahan pada setiap telepon yang menggunakan Symbian OS. Dengan adanya lisensi kerjasama, pengembang aplikasi dapat menambahkan dengan syarat tidak mengganti ataupun mengubah fungsi API standar yang dikategorikan pada bagian ini.

2.11.2.2 (API) Symbian Umum Tergantikan

Komponen yang memerlukan kostumisasi dari komponen Symbian Umum yang diperlukan untuk bekerja dengan ROM dari sistem dimana ia diinstal. Komponen ini merupakan komponen yang bekerja pada *low-level* dari hardware tertentu. Untuk mendapatkan komponen ini pihak pengembang aplikasi memerlukan lisensi dengan pihak Symbian karena versi komponen ini disediakan oleh pihak Symbian. Namun pada dasarnya komponen ini merupakan komponen standar (umum) yang tersedia pada semua versi Symbian OS.

2.11.2.3 (API) Symbian Opsional

Komponen-komponen ini sifatnya opsional (tidak selalu ada) pada semua versi Symbian OS. Namun jika tersedia, maka pengembang aplikasi mendapat jaminan

bahwa aplikasinya dapat menggunakan API pada kategori ini pada versi Symbian OS yang sama.

2.11.2.4 (API) Symbian Opsional Tergantikan

Bentuk kategori ini mirip dengan kategori Symbian Opsional adalah kumpulan API yang tidak terikat dengan API umum yang ada pada versi Symbian OS dan dapat ditambahkan oleh pihak pengembang dengan suatu lisensi dari pihak Symbian.