BAB II

LANDASAN TEORI

2.1 Konsep Dasar Program Aplikasi

Program adalah ekspresi pernyataan atau kombinasi yang disusun dan dirangkai menjadi satu kesatuan prosedur yang berupa urutan langkah untuk menyelesaikan masalah yang diimplementasikan dengan menggunakan bahasa pemrograman, sehingga dapat dieksekusi oleh program.

Aplikasi merupakan penerapan, penyimpanan sesuatu hal, data, permasalahan pekerja ke dalam suatu sarana atau media yang digunakan untuk menerapkan atau mengimplementasikan suatu hal atau permasalahan tersebut sehingga berubah menjadi suatu bentuk yang baru tanpa menghilangkan nilai-nilai dasar dari hal, data, permasalahan atau pekerjaan. Jadi, dalam hal ini hanya berbentuk tampilan data yang berubah, sedangkan isi yang termuat dalam data tersebut tidak mengalami perubahan.

Program Aplikasi adalah sederetan kode yang digunakan untuk mengatur komputer agar dapat melakukan pekerjaan sesuai dengan keinginan *programmer* atau *user*.

2.2 Pengertian Internet

Internet merupakan jaringan global yang terdiri dari berbagai komputer yang saling berhubungan dan bekerjasama dengan cara berbagai informasi dan data. Media penghubung tersebut bisa melalui kabel, kanal satelit maupun frekuensi radio.

Setiap komputer yang terhubung dengan jaringan tersebut, diberikan sebuah nomor yang unik, dan berkomunikasi satu sama lainnya dengan bahasa komunikasi yang sama. Bahasa komunikasi yang sama ini disebut protokol. Protokol yang digunakan di internet adalah TCP/IP (*Transmission Control Protocol* / *Internet Protocol*).

2.3 Sejarah Internet

Berasal dari ARPANet, suatu proyek yang dimulai dari Departemen Pertahanan Amerika Serikat (*US Departement of Defense –DOD*) pada tahun 1969, suatu percobaan dalam *reliable networking* (jaringan secara terpercaya) untuk menghubungkan antara DOD dengan kontraktor penelitian militer, termasuk sejumlah besar universitas yang melakukan penelitian dengan dana militer.[5]

ARPA merupakan singkatan dari *Advanced Research Projects Agency*, cabang dari *Defence* (Pertahanan) yang mempunyai kewajiban membagi-bagikan uang *grant* sehingga menjadi DARPA (*Defence-ARPA*). ARPANet mulai dengan 3 komputer kecil yang dikoneksi di California digabungkan dengan satu di Utah, tetapi secara cepat berkembang di seluruh kontinen. Internet ini mulai tumbuh pesat pada dekade 1990.

2.4 Kegunaan Internet

Kegunaan internet yang utama antara lain :

1. Fungsi komunikasi

   Internet adalah alat komunikasi, kegunaan yang sangat penting dari internet adalah pertukaran pesan dengan menggunakan *electronic mail* (*e-mail*).

2. Fungsi *Resource Sharing*

   Dengan internet, kita dapat mencari *software*, essay, data dan program dari ribuan titik distribusi di seluruh dunia.

3. Fungsi *Resource Discovery*

   Navigasi untuk mencari file tertentu, dokumen, *host* atau orang diantara jutaan host.

4. Fungsi Komunitas

   Masyarakat pengguna internet.

2.5 Perkembangan Internet

Jumlah pengguna internet yang besar dan semakin berkembang, telah mewujudkan budaya internet. Internet juga mempunyai pengaruh yang besar atas ilmu, dan pandangan dunia. Dengan hanya berpandukan mesin pencari seperti *Google*, pengguna di seluruh dunia mempunyai akses yang mudah atas bermacam-macam informasi. Dibanding dengan buku dan perpustakaan, internet melambangkan penyebaran (*decentralization*) informasi dan data secara ekstrim.

Perkembangan internet juga telah mempengaruhi perkembangan ekonomi. Berbagai transaksi jual beli yang sebelumnya hanya bisa dilakukan dengan cara tatap muka (dan sebagian sangat kecil melalui pos atau telepon), kini sangat mudah dan sering dilakukan melalui internet. Transaksi melalui internet ini dikenal dengan nama *e-commerce*.

Terkait dengan pendidikan, internet juga memicu meningkatnya kualitas pendidikan karena banyaknya informasi yang bisa didapat berkaitan dengan pendidikan tersebut. Sarana-sarana untuk bertukar pikiran pun telah banyak

tersedia seperti forum,milis,*e-learning*,kuliah online, sampai yang terbaru saat ini yaitu pembelajaran kolaboratif (*collaborative learning*). *Collaborative learning* ini merupakan salah satu implementasi dari teknologi internet yang sedang berkembang saat ini yaitu *Web* 2.0.[4]

2.6 Pengertian *Web* 2.0

Web 2.0 merupakan perkembangan dari teknologi *web* saat ini. Teknologi ini berupa layanan di *web* yang lebih menekankan pada *social network* atau jalinan sosial antara penggunanya dan memungkinkan pengguna dapat saling berkolaborasi dan berbagi informasi secara *online*. *Web* 2.0 juga merupakan istilah yang digunakan untuk menunjukkan suatu aplikasi web yang mempunyai interaktivitas dengan penggunanya sama seperti halnya dengan aplikasi *desktop*.

2.7 Pengertian *Social Networking*

Menurut Wikipedia, *social networking* atau jejaring sosial adalah suatu struktur sosial yang terdiri dari elemen-elemen individual atau organisasi. Jejaring ini menunjukan jalan dimana mereka berhubungan karena kesamaan sosialitas, mulai dari mereka yang dikenal sehari-hari sampai dengan keluarga. Istilah ini diperkenalkan oleh profesor J.A. Barnes di tahun 1954.

Jejaring sosial adalah suatu struktur sosial yang dibentuk dari simpul-simpul (yang umumnya adalah individu atau organisasi) yang diikat dengan satu atau lebih tipe relasi spesifik seperti nilai, visi, ide, teman, keturunan, dll.

2.8 Sejarah Jejaring Sosial

Sejak komputer dapat dihubungkan satu dengan lainnya dengan adanya internet banyak upaya awal untuk mendukung jejaring sosial melalui komunikasi

antar komputer. Situs jejaring sosial diawali oleh Classmates.com pada tahun 1995 yang berfokus pada hubungan antar mantan teman sekolah dan SixDegrees.com pada tahun 1997 yang membuat ikatan tidak langsung. Dua model berbeda dari jejaring sosial yang lahir sekitar pada tahun 1999 adalah berbasiskan kepercayaan yang dikembangkan oleh Epinions.com, dan jejaring sosial yang berbasiskan pertemanan seperti yang dikembangkan oleh Uskup Jonathan yang kemudian dipakai pada beberapa situs UK regional di antara 1999 dan 2001.[4]

Inovasi meliputi tidak hanya memperlihatkan siapa berteman dengan siapa, tetapi memberikan pengguna kontrol yang lebih akan isi dan hubungan. Pada tahun 2005, suatu layanan jejaring sosial *MySpace*, dilaporkan lebih banyak diakses dibandingkan *Google* dengan *Facebook*, pesaing yang tumbuh dengan cepat.

Jejaring sosial mulai menjadi bagian dari strategi internet bisnis sekitar tahun 2005 ketika Yahoo meluncurkan Yahoo! 360°. Pada bulan juli 2005 *News Corporation* membeli *MySpace*, diikuti oleh ITV (UK) membeli *Friends Reunited* pada Desember 2005. Diperkirakan ada lebih dari 200 situs jejaring sosial menggunakan model jejaring sosial ini.

2.9 Layanan Jejaring Sosial

Banyak layanan jejaring sosial berbasiskan *web* yang menyediakan kumpulan cara yang beragam bagi pengguna untuk dapat berinteraksi seperti *chat*, *messaging*, *email*, video, *chat* suara, *share file*, *blog*, diskusi grup, dan lain-lain.

Umumnya jejaring sosial memberikan layanan untuk membuat biodata dirinya. Pengguna dapat meng-*upload* foto dirinya dan dapat menjadi teman dengan pengguna lainnya. Beberapa jejaring sosial memiliki fitur tambahan seperti pembuatan grup untuk dapat saling *sharing* didalamnya.

2.10 *Collaborative Learning*

Collaboration didefinisikan sebagai kerjasama antar peserta dalam rangka mencapai tujuan bersama.

Collaborative learning merupakan salah satu metode dalam pembelajaran berbasis pada siswa(*student-centered learning*) yang dapat didefinisikan sebagai proses belajar kelompok yang setiap anggotanya aktif menyumbangkan informasi, pengalaman, ide, sikap, pendapat, kemampuan, dan keterampilan yang dimiliki untuk bersama-sama saling meningkatkan pemahaman.

2.11 *Object Oriented Analysis and Design* (OOAD)

Object oriented analysis adalah metode analisis yang memeriksa keperluan (*requirements*) dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup permasalahan (ASU[1]).

Object oriented design adalah metode untuk mengarahkan arsitektur *software* yang didasarkan pada manipulasi objek-objek sistem atau subsistem.

2.12 Objek

Objek (*object*) adalah benda, secara fisik ataupun konseptual. *Hardware*, *software*, dokumen dan manusia adalah beberapa contoh dari objek. Sebuah objek memiliki keadaan sesaat (*state*) dan prilaku (*behavior*) (ASU[1]).

State adalah kondisi objek yang menggambarkan objek tersebut, sedangkan *behaviour* adalah suatu definisi tindakan dan reaksi suatu objek.

2.13 Kelas

Kelas (*class*) adalah definisi umum untuk himpunan objek sejenis. Kelas menetapkan spesifikasi prilaku (*behavior*) dan atribut objek-objek tersebut. *Class* adalah abstraksi dari entitas dalam dunia nyata. Objek adalah contoh (*instance*) dari sebuah kelas (ASU[1]).

2.14 UML (*Unified Modeling Language*)

UML merupakan model yang dapat digunakan secara luas dalam pemodelan bisnis, pemodelan perangkat lunak dari semua fase pembentukan dan semua tipe sistem, dan pemodelan secara umum dari berbagai pembentukan / konstruksi yang memiliki dua perilaku yaitu baik statis maupun dinamis (MUS[3]).[6]

2.15 Sejarah UML

UML dimulai secara resmi pada oktober 1994, ketika Rumbaugh bergabung dengan Booch pada Relational Software Corporation. Proyek ini memfokuskan pada penyatuan metode Booch dan OMT. UML versi 0.8 merupakan metode penyatuan yang dirilis pada bulan Oktober 1995. Dalam waktu yang sama, Jacobson bergabung dengan Relational dan cakupan dari UML semakin luas sampai diluar perusahaan OOSE. Dokumentasi UML versi 0.9 akhirnya dirilis pada bulan Juni 1996. Meskipun pada tahun 1996 ini melihat dan menerima feedback dari komunitas Software Engineering . Dalam waktu tersebut, menjadi lebih jelas bahwa beberapa organisasi perangkat lunak melihat UML sebagai

strategi dari bisnisnya. Kemudian dibangunlah UML Consortium dengan beberapa organisasi yang akan menyumbangkan sumber dayanya untuk bekerja, mengembangkan, dan melengkapi UML (SOF[10]).

Di sini beberapa partner yang berkontribusi pada UML 1.0, diantaranya Digital Equipment Corporation, Hewlett-Packard, I-Logix, Intellicorp, IBM, ICON Computing, MCI Systemhouse, Microsoft, Oracle, Relational, Texas Instruments dan Unisys. Dari kolaborasi ini dihasilkan UML 1.0 yang merupakan bahasa pemodelan yang ditetapkan secara baik, kuat, dan cocok untuk lingkungan masalah yang luas. UML 1.0 ditawarkan menjadi standarisasi dari Object Management Group (OMG). Dan pada Januari 1997 dijadikan sebagai standar bahasa pemodelan.

Antara Januari–Juli 1997 gabungan group tersebut memperluas kontribusinya sebagai hasil respon dari OMG dengan memasukkan Adersen Consulting, Ericsson, ObjectTimeLimeted, Platinum Technology, Ptech, Reich Technologies, Softeam, Sterling Software dan Taskon. Revisi dari versi UML (versi 1.1) ditawarkan kepada OMG sebagai standarisasi pada bulan Juli 1997. Dan pada bulan September 1997, versi ini dierima oleh OMG Analysis dan Design Task Force (ADTF) dan OMG ArchitectureBoard. Dan Akhirnya pada Juli 1997 UML versi 1.1 menjadi standarisasi.

Pemeliharaan UML terus dipegang oleh OMG Revision Task Force (RTF) yang dipimpin oleh Cris Kobryn. RTP merilis editorial dari UML 1.2 pada Juni 1998. Dan pada tahun 1998 RTF juga merilis UML 1.3 disertai dengan user guide dan memberikan technical cleanup.

2.16 Tujuan Penggunaan UML

Memodelkan suatu sistem (bukan hanya perangkat lunak) yang menggunakan konsep berorientasi *object*. Dan menciptakan suatu bahasa pemodelan yang dapat digunakan baik oleh manusia maupun mesin (SOF[10]).

2.17 Gambaran dari UML

UML dapat dimanfaatkan untuk berbagai macam kebutuhan, diantaranya untuk melakukan pemodelan, visualisasi, spesifikasi, konstruksi dan dokumentasi.

2.18 UML sebagai Bahasa Pemodelan

UML merupakan bahasa pemodelan yang memiliki pembendaharaan kata dan cara untuk mempresentasikan secara fokus pada konseptual dan fisik dari suatu sistem. Contoh untuk sistem *software* yang *intensive* membutuhkan bahasa yang menunjukkan pandangan yang berbeda dari arsitektur sistem, ini sama seperti menyusun/mengembangkan *software development life cycle*. Dengan UML akan memberitahukan kita bagaimana untuk membuat dan membaca bentuk model yang baik, tetapi UML tidak dapat memberitahukan model apa yang akan dibangun dan kapan akan membangun model tersebut. Ini merupakan aturan dalam *software development process*.

2.19 UML sebagai bahasa Visualizing

UML tidak hanya merupakan rangkaian simbol grafikal, cukup dengan tiap simbol pada notasi UML merupakan penetapan semantik yang baik. Dengan cara ini, satu pengembang dapat menulis model UML dan pengembang lain atau perangkat yang sama lainnya dapat mengartikan bahwa model tersebut tidak ambigu. Hal ini akan mengurangi error yang terjadi karena perbedaan bahasa

dalam komunikasi model konseptual dengan model lainnya. UML menggambarkan model yang dapat dimengerti dan dipresentasikan ke dalam model tekstual bahasa pemograman. Contohnya kita dapat menduga suatu model dari sistem yang berbasis web tetapi tidak secara langsung dipegang dengan mempelajari *code* dari sistem.

Dengan model UML maka kita dapat memodelkan suatu sistem web tersebut dan direpresentasikan ke bahasa pemrograman. UML merupakan suatu model eksplisit yang menggambarkan komunikasi informasi pada sistem. Sehingga kita tidak kehilangan informasi *code* implementasi yang hilang dikarenakan *developer* memotong *coding* dari implementasi.

2.20 UML sebagai bahasa Specifying

Maksudnya membangun model yang sesuai, tidak ambigu dan lengkap. Pada faktanya UML menunjukan semua spesifikasi keputusan analisis, desain dan implementasi yang penting yang harus dibuat pada saat pengembangan dan penyebaran dari sistem *software* intensif.

2.21 UML sebagai bahasa Constructing

UML bukan bahasa pemograman visual, tetapi model UML dapat dikoneksikan secara langsung pada bahasa pemograman visual. Maksudnya membangun model yang dapat dimapping ke bahasa pemograman seperti java, C++, VB atau tabel pada *database relational* atau penyimpanan tetap pada *database* berorientasi *object*.

2.22 UML sebagai bahasa Documenting

Maksudnya UML menunjukan dokumentasi dari arsitektur sistem dan detail dari semuanya. UML hanya memberikan bahasa untuk memperlihatkan permintaan dan untuk tes. UML menyediakan bahasa untuk memodelkan aktifitas dari perencanaan project dan manajemen pelepasan (*release management*).

2.23 Bagian-Bagian dari UML

Bagian-bagian utama dari UML adalah view, diagram, model element, dan *general mechanism.*

2.24 View

View digunakan untuk melihat sistem yang dimodelkan dari beberapa aspek yang berbeda. View bukan melihat grafik, tapi merupakan suatu abstraksi yang berisi sejumlah diagram. Beberapa jenis view dalam UML antara lain: use case view, logical view, component view, concurrency view, dan deployment view.

2.25 Use case view

Mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai yang diinginkan *external actors*. *Actor* yang berinteraksi dengan sistem dapat berupa user atau sistem lainnya.

View ini digambarkan dalam *use case diagrams* dan kadang-kadang dengan *activity diagrams*. View ini digunakan terutama untuk pelanggan, perancang (*designer*), pengembang (*developer*), dan penguji sistem (*tester*).

2.26 Logical view

Mendeskripsikan bagaimana fungsionalitas dari sistem, struktur statis (*class*, *object*, dan *relationship*) dan kolaborasi dinamis yang terjadi ketika *object* mengirim pesan ke *object* lain dalam suatu fungsi tertentu. View ini digambarkan dalam *class diagrams* untuk struktur statis dan dalam *state*, *sequence*, *collaboration*, dan *activity diagram* untuk model dinamisnya. View ini digunakan untuk perancang (*designer*) dan pengembang (*developer*).

2.27 Component view

Mendeskripsikan implementasi dan ketergantungan modul. Komponen yang merupakan tipe lainnya dari *code module* diperlihatkan dengan struktur dan ketergantungannya juga alokasi sumber daya komponen dan informasi administrative lainnya. View ini digambarkan dalam component view dan digunakan untuk pengembang (*developer*).

2.28 Concurrency view

Membagi sistem ke dalam proses dan prosesor. View ini digambarkan dalam diagram dinamis (*state*, *sequence*, *collaboration*, dan *activity diagrams*) dan diagram implementasi (*component* dan *deployment diagrams*) serta digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

2.29 Deployment view

Mendeskripsikan fisik dari sistem seperti komputer dan perangkat (*nodes*) dan bagaimana hubungannya dengan lainnya. View ini digambarkan dalam *deployment diagrams* dan digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

2.30 Diagram

Diagram berbentuk grafik yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah diagram merupakan bagian dari suatu view tertentu dan ketika digambarkan biasanya dialokasikan untuk view tertentu. Adapun jenis diagram antara lain :

2.31 Use Case Diagram

Menggambarkan sejumlah external actors dan hubungannya ke use case yang diberikan oleh sistem. Use case adalah deskripsi fungsi yang disediakan oleh sistem dalam bentuk teks sebagai dokumentasi dari use case symbol namun dapat juga dilakukan dalam *activity diagrams*. Use case digambarkan hanya yang dilihat dari luar oleh *actor* (keadaan lingkungan sistem yang dilihat user) dan bukan bagaimana fungsi yang ada di dalam sistem.

2.32 Class Diagram

Menggambarkan struktur statis *class* di dalam sistem. *Class* merepresentasikan sesuatu yang ditangani oleh sistem.

Class dapat berhubungan dengan yang lain melalui berbagai cara: *associated* (terhubung satu sama lain), *dependent* (satu class tergantung/menggunakan class yang lain), *specialed* (satu class merupakan spesialisasi dari class lainnya), atau *package* (grup bersama sebagai satu unit). Sebuah sistem biasanya mempunyai beberapa *class diagram*.

2.33 State Diagram

Menggambarkan semua *state* (kondisi) yang dimiliki oleh suatu *object* dari suatu *class* dan keadaan yang menyebabkan *state* berubah. Kejadian dapat berupa

object lain yang mengirim pesan. *State class* tidak digambarkan untuk semua *class*, hanya yang mempunyai sejumlah *state* yang terdefinisi dengan baik dan kondisi *class* berubah oleh *state* yang berbeda.

2.34 Sequence Diagram

Menggambarkan kolaborasi dinamis antara sejumlah *object*. Kegunaanya untuk menunjukkan rangkaian pesan yang dikirim antara *object* juga interaksi antara *object,* sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem.

2.35 Collaboration Diagram

Menggambarkan kolaborasi dinamis seperti *sequence diagrams*. Dalam menunjukkan pertukaran pesan, *collaboration diagrams* menggambarkan *object* dan hubungannya (mengacu ke konteks). Jika penekannya pada waktu atau urutan gunakan *sequence diagrams*, tapi jika penekanannya pada konteks gunakan *collaboration diagram*.

2.36 Activity Diagram

Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi.

2.37 Component Diagram

Menggambarkan struktur fisik kode dari komponent. Komponent dapat berupa *source code*, komponent *biner*, atau *executable component.* Sebuah komponent berisi informasi tentang *logic class* atau *class* yang diimplementasikan sehingga membuat pemetaan dari *logical view* ke *component view*.

2.38 Deployment Diagram

Menggambarkan arsitektur fisik dari perangkat keras dan perangkat lunak sistem, menunjukkan hubungan komputer dengan perangkat (*nodes*) satu sama lain dan jenis hubungannya. Di dalam *nodes*, *executeable component* dan *object* yang dialokasikan untuk memperlihatkan unit perangkat lunak yang dieksekusi oleh node tertentu dan ketergantungan komponen.

2.39 Area Penggunaan UML

UML digunakan paling efektif pada domain seperti :

1. Sistem Informasi Perusahaan
2. Sistem Perbankan dan Perekonomian
3. Bidang Telekomunikasi
4. Bidang Transportasi
5. Bidang Penerbangan
6. Bidang Perdagangan
7. Bidang Pelayanan Elekronik
8. Bidang Pengetahuan
9. Bidang Pelayanan Berbasis Web Terdistribusi

Namun UML tidak terbatas untuk pemodelan *software*. Pada faktanya UML banyak untuk memodelkan sistem non *software* seperti:

1. Aliran kerja pada sistem perundangan.
2. Struktur dan kelakuan dari Sistem Kepedulian Kesehatan Pasien
3. Desain *hardware* dll.

2.40 Rational Rose

Rational Rose adalah perangkat lunak yang memiliki perangkat-perangkat pemodelan secara visual untuk membangun solusi dalam rekayasa perangkat lunak dan pemodelan bisnis.[6]

Rational Rose juga dapat didefinsikan sebagai tools pemodelan visual untuk pengembangan sistem berbasis objek yang handal untuk digunakan sebagai bantuan bagi para pengembang dalam melakukan analisis dan perancangan sistem. *Rational rose* mendukung permodelan bisnis yang membantu para pengembang memahami sistem secara komprehensif. Ia juga membantu analisis sistem 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.[6]

Dalam *Rational rose*, pemodelan adalah cara melihat sistem dari berbagai sudut pandang. Ia mencakup semua diagram yang dikenal dalam UML, actor-aktor yang terlibat dalam sistem, use-case, objek-objek, kelas-kelas, komponen-komponen, serta simpul-simpul penyebaran. Model juga mendeskripsikan rincian yang diperlukan sistem dan bagaimana ia akan bekerja, sehingga para pengembang dapat menggunakan model itu sebagai blue print untuk sistem yang akan dikembangkan.[6]

2.41 Fitur Rational Rose

Bahasa yang digunakan adalah bahasa pemodelan standar yaitu UML.

1. Rational Rose mendukung round-trip engineering sehingga kita dapat men-*generate* model kedalam kode (Java, C++, Visual Basic, dan sebagainya) dan melakukan reverse engineering untuk menampilkan arsitektur sistem dari kode yang ada.
2. Model dan kode senantiasa singkron.
3. Mudah dalam memperbaiki perangkat lunak, karena kita dapat menggambarkan kembali arsitektur perangkat lunak tersebut dalam UML.
4. Tersedia Rational Rose untuk berbagai platform.
5. Memiliki Rose Web Publisher sehingga suatu tim dapat mengkomunikasikan model dan spesifikasinya dalam web browser.
6. Mendukung rekayasa perangkat lunak untuk sistem client/server.

2.42 Apache

Apache merupakan web *server* yang paling sering digunakan sebagai *server* internet dibandingkan *web server* lainnya. *Web server* merupakan *server* internet yang mampu melayani koneksi transfer di dalam protokol *HTTP* (*Hypertext Transfer Protocol*) saat ini *web server* merupakan inti dari *server-server* internet selain *e-mail server*, *ftp server*, dan *news server*.

Web server sendiri dirancang untuk dapat melayani berbagai jenis data, diantaranya *text, hypertext*, gambar, suara dan bentuk file *HTML*. Dari file *HTML* kemudian dapat dikaitkan ke file *HTML* lainnya yang hendak dipublikasikan lewat internet. (ADA[7])

2.43 PHP

Word Wide Web (WWW) telah berubah dengan cepat dengan berbagai cara Bahasa standar yang dipakai dalam WWW adalah bahasa HTML (*Hyper Text Language*). Dengan adanya teknologi ini, kita dapat melihat dokumen yang kadang-kadang berada dibagian lain di dunia ini dengan hanya sebuah program sederhana dan kita dapat banyak mendapatkan informasi yang dengan cepat dapat kita terima sesuai dengan yang kita perlukan.

Sekarang kita mengharapkan lebih dari internet. Kita mengharapkan sebuah situs web yang berisi dengan informasi yang banyak dan dalam tampilan yang menarik serta kita dapat mencari dokumen yang diperlukan dengan mudah. Untuk mencapai hala tersebut kita memerlukan suatu web yang dinamis. Karena jikan hanya mengandalkan situs web yang statis maka akan memerlukan pemeliharaan yang susah, sebagai contoh sebuah perusahaan ingin membuat suatu situs web yang berisi promosi produk sebanyak 1000 jenis produk maka dalam situs tersebut minimal kita harus membuat 1000 web statis yang berbeda (itu pasti merepotkan). Dengan memanfaatkan teknologi situs web dinamis maka dalam situs tersebut hanya memerlukan halaman web yang lebih sedikit karena data produk disimpan dalam database.

PHP merupakan bahasa *script* yang digunakan untuk membuat halaman web yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh *client*. Mekanisme ini menyebabkan informasi yang diterima *client* selalu baru. Semua *script* php dari pada *spesifikasi client*. Namun tetap diperhatikan bahwa halaman web yang dihasilkan tentunya harus dapat

dibuka oleh *browser* pada *client.* Dalam hal ini versi dari html yang digunakan harus didukung oleh *browser client*.

PHP termasuk dalam *Open Source Product*. Jadi dapat merubah *source code* dan mendistribusikannya secara bebas. PHP juga diedarkan secara gratis.

2.44 MySQL

MySQL merupakan sebuah *software* yang berguna sebagai suatu *Database Server* yang cukup terkenal. Kepopulerannya seiring dengan pengguanan *script* PHP untuk web *programming*. *Database server* itu sendiri merupakan suatu *software* yang bertugas untuk melayani permintaan (*request*) *query* dari *client.* MySQL sebagai suatu *database server* mempunyai beberapa kemampuan, salah satunya harus menyediakan suatu sistem manajemen *database* yang dapat mengatur bagaimana menyimpan, menambah, mengakses data dan transaksi-transaksi *database* lainnya. MySQL cepat sekali berkembang, karena MySQL merupakan suatu *software* yang *Open Source*.

2.45 Macromedia Dreamweaver

Macromedia Dreamweaver adalah sebuah HTML editor professional untuk mendesain secara visual dan mengelola situs Web maupun halaman Web. *Macromedia Dreamweaver* merupakan *software* utama yang digunakan oleh *Web Designer* maupun *Web Progremer* guna mengembangkan situs Web. Ruang kerja, fasilitas dan kemampuan *Macromedia Dreamweaver* mampu meningkatkan produktivitas dan efektivitas dalam desain maupun membangun situs Web.