BAB II

LANDASAN TEORI

2.1 Jaringan Komputer LABKOM STIKOM Surabaya

LABKOM adalah sebuah fasilitas yang digunakan untuk keperluan praktek ataupun riset. LABKOM memiliki terminal untuk kegiatan praktikum 400 unit PC yang digunakan setiap hari. Terdapat beberapa server yang diakses oleh praktikan dari terminal tersebut. Pada kondisi tertentu jika diperlukan akses internet dari masing-masing terminal bisa dilakukan karena LABKOM terhubung dengan jaringan internet.

2.2 Konsep Monitoring Jaringan Dan Protokol

Untuk membangun sebuah sistem monitoring, ada beberapa *protocol* yang digunakan. Dan terlebih dahulu perlu diketuhui konsep dari *network monitoring.* Berikut adalah konsep dari *network monitoring* dan *protocol* yang digunakan.

2.2.1 Monitoring

Monitoring(pemantauan) merupakan sebuah proses penaksiran atau penilaian kualitas kinerja sistem dari waktu ke waktu. Pemantauan ini dilakukan secara berkelanjutan sejalan dengan kegiatan usaha yang mencakup kegiatan sehari hari (Tampubolon, 2005). Pengawasan adalah pengendalian yang dilakukan dengan melaksanakan pemeriksaan, penilaian kemampuan, meningkatkan dan menyempurnakan, baik manajemen maupun bidang operasionalnya (Rusyani, 1997). Penggunaan sistem monitoring bertujuan untuk dapat mengontrol,

mengawasi serta mengecek sejumlah aktivitas yang telah dilakukan (Tan, 2010). Dari beberapa pengertian di atas dapat disimpulkan bahwa monitoring adalah proses pengumpulan informasi secara berkelanjutan dengan tujuan untuk dapat mengawasi kegiatan yang telah dilakukan guna meningkatkan dan menyempurnakan tujuan yang akan dicapai.

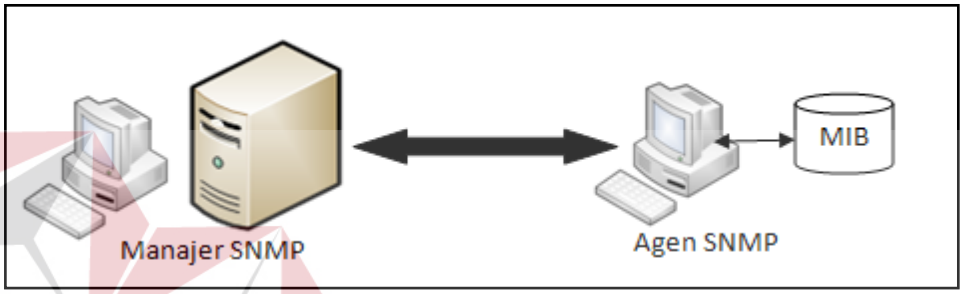
2.2.2 Monitoring Jaringan

Monitoring jaringan adalah salah satu fungsi dari *management* yang berguna untuk menganalisa apakah jaringan masih cukup layak untuk digunakan atau perlu tambahan kapasitas. Hasil monitoring juga dapat membantu jika admin ingin mendesain ulang jaringan yang telah ada. Banyak hal dalam jaringan yang bisa dimonitoring, salah satu diantaranya *load traffic* jaringan yang lewat pada sebuah router atau *interface* komputer. Monitoring dapat dilakukan dengan standar *simple network management protocol*(SNMP), selain *load traffic* jaringan, kondisi jaringan pun harus dimonitoring, misalnya status up atau down dari sebuah peralatan jaringan. Hal ini dapat dilakukan dengan utilitas ping (Fawaidus, 2012:1).

2.2.3 Simple Network Management Protocol

SNMP adalah sebuah protokol apikasi pada jaringan TCP/IP yang menangani manajemen jaringan. Protokol ini didesain sehingga pengguna dapat dengan mudah memantau kondisi jaringan komputer. Pemantauan kondisi jaringan dapat dilakukan dengan cara pengumpulan nilai-nilai informasi dari kondisi jaringan secara jarak jauh atau menggunkan satu pusat pengamatan.

SNMP menjadi protokol yang terus dikembangkan karena banyak perangkat jaringan yang mendukung dan tersedia layanan SNMP seperti *router*, *switch*, server, *workstation*, dan *printer*. Protokol SNMP pada jaringan TCP/IP menggunakan *transport* UDP oleh karena itu dalam penggunaannya tidak akan membebani trafik jaringan (Pradikta, Affandi, & Setijadi, 2013). Struktur SNMP dari manajer, agen, dan MIB dapat dilihat pada Gambar 2.1.



Gambar 2.1 Manajer, Agen, Dan MIB. Sumber (Pradikta, Affandi, & Setijadi, 2013)

Pada sistem pemantauan jaringan dengan menggunakan layanan SNMP, terdapat tiga komponen dasar antara lain (Pradikta, Affandi, & Setijadi, 2013) :

1. Manajer SNMP

Manajer adalah perangkat yang menjalankan dan dapat menangani tugas-tugas manajemen jaringan.

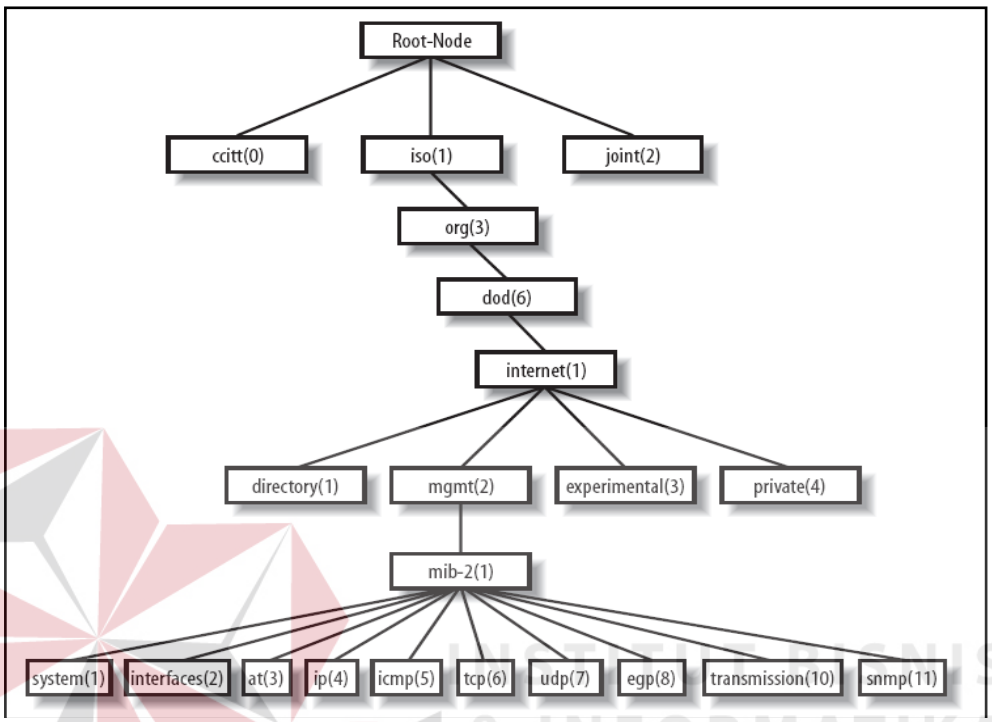
2. Agen SNMP

Agen SNMP adalah perangkat pada jaringan yang akan diamati dan dikelola. Setiap agen akan merespon dan menjawab permintaan manajer SNMP.

3. *Management Information Base* (MIB)

MIB pada SNMP dapat dikatakan sebagai tempat penyimpanan informasi yang dimiliki agen. MIB yang terdapat pada SNMP didefinisikan secara hirarki

dan setiap bagian mempunyai identifikasi objek (OID). Diagram pohon MIB dapat dilihat pada Gambar 2.2.



Gambar 2.2 Diagram Pohon MIB. Sumber (Pradikta, Affandi, & Setijadi, 2013)

Berikut adalah beberapa objek yang dapat digunakan untuk mengumpulkan informasi tentang sistem *network interface* (Mauro & Schmidt, 2005):

1. *ifDescr*

   Deskripsi yang diberikan kepada pengguna tentang *interface.*

2. *ifType*

   Jenis dari *interface* (*token ring, Ethernet,etc.*).

3. *ifOperStatus*

   Informasi apakah *interface* dalam kondisi *up, down,* atau dalam beberapa jenis mode testing.

4. *ifMtu*

   ukuran dari besar paket yang dapat dikirim melalui *interface.*

5. *ifSpeed*

   Maximum bandwith dari sebuah *interface.*

6. *ifPhysAddress*

   Alamat tingkat rendah(*hardware*) dari sebuah *interface.*

7. *ifInOctets*

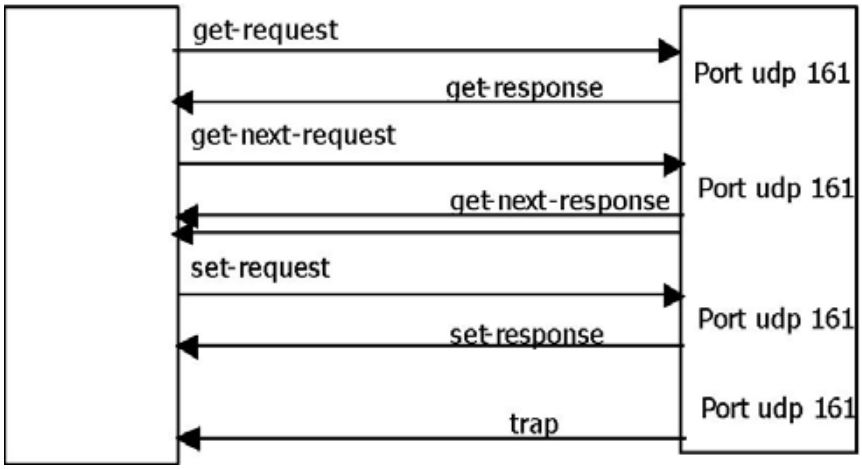
   Angka dari oktet yang diterima oleh *interface.*

8. *ifOutOctets*

   Angka dari oktet yang dikirim oleh *interface.*

2.2.3.1 Protocol Data Unit (PDU) SNMP

PDU merupakan unit data yang terdiri atas sebuah *header* dan beberapa data yang ditempelkan. Analogi dari PDU seperti sebuah benda yang mengandung variable-variabel. Variabel ini memiliki nama dan nilai. Protokol SNMP menggunakanan operasi yang relatif sederhana dan PDU dalam jumlah terbatas untuk menjalankan fungsinya. Lima PDU yang telah didefinisikan dalam *standart* adalah *Get Request, Get-Next Request, Get Response, Set Request, Set Response, Trap* (Sajati, 2013). Skema PDU dari SNMP dapat dilihat pada Gambar 2.3.



Gambar 2.3 Skema PDU Manajer SNMP Agen SNMP. Sumber (Sajati, 2013)

2.3 Network Monitoring Tools

Beberapa *tools* yang digunakan untuk melakukan *monitoring* jaringan adalah sebagai berikut:

1. SNMPWALK

SNMPWALK adalahs sebuah aplikasi yang menggunakan SNMP GETNEXT untuk mengambil data informasi. *Object identifier* (OID) bisa didapatkan dengan perintah SNMPWALK (Mauro & Schmidt, 2005).

2. TCPDump

Untuk mendukung pengumpulan informasi Log website yang di akses oleh *client* maka dibutuhkan sebuah sistem yang dapat melakukan *capture* log web transaksi. Untuk itu dibutuhkan aplikasi *pre-instaled* dari ubuntu yakni TCPDump untuk mendukung pembangungan aplikasi.

TCPDump adalah *command-line network traffic-monitoring tool* yang dapat mengumpulkan informasi *packets* di *network interface* dan memungkinkan

administrator untuk menganalisa hasilnya informasi yang dikumpulkan (Stanger & Lane, 2001).

Opsi-opsi untuk sistem aplikasi TCPDump dapat dilihat pada Tabel 2.1.

Tabel 2.1 TCPDump Option. Sumber (Stanger & Lane, 2001)

Option	Description
-a	Display data in ASCII
-b	Capture packets using the Data Link Layer specified. These include the following protocols(see RFC 1340 if you want to list these protocols as decimal values): IP IPv6 802.2 802.3 Arp Rarp Dec Lat Atalk Aarp X25 Ipx
-c	Quit TCPDump after specific number(count) of packets have been capture
-e	List the link-level header for each packet that is captured
-F	Instead of listing option and expressions, you can use a file as the input for your filter expression. If a file is used, any expressions you list on the command line will be ignored
-i	Specify the interface you want TCPDump to listen. If the interface is not specified, it searches for the lowest interface number, excluding the loopback, and prints the packets for that interface
-n	Do not list hostname, only list host address as number(such as IP Addresses). This avoids Domain Name Syste(DNS) lookups.
-nn	Do not list port number as a service names. The /etc/services file is used for the service names
-p	Do not enable promiscuous mode for the interface
-q	Quick output to reduce the amount of protocol information displayed by TCPDump on each line
-r	Read packet data from a saved file instead of capturing data on an interface
-t	Do not list the timestamp
-v	Print out verbose output from the capture. Includes even more data, such as time-to-live(TTL) and type of service data. The –vv option will list the additional data
-w	Write the TCPDump packet capture to a file instead of displaying it
-x	Display the packet information in hexadecimal format

2.4 Konsep Network Manajemen

Manajemen jaringan adalah sebuah konsep umum yang mempekerjakan penggunaan berbagai alat, teknik, dan sistem untuk membantu manusia dalam mengelola berbagai perangkat, sistem, atau jaringan (Mauro & Schmidt, 2005).

2.4.1 Bandwith

Istilah *bandwidth* dapat didefinisikan sebagai kapasitas atau daya tampung suatu kanal komunikasi untuk dapat dilewati trafik dalam satuan waktu tertentu. Pengalokasian *bandwidth* yang tepat dapat menjadi salah satu metode dalam memberikan jaminan kualitas suatu layanan jaringan (QoS = *Quality Of Services*). Sedangkan istilah trafik dapat didefinisikan sebagai banyaknya informasi yang melewati suatu kanal komunikasi (Riza, Eryzebuan, & Ahmad, 2011). Berikut adalah perhitungan *bandwith utilization* menggunakan SNMP:

1. *Half Duplex*

$$\frac{(\Delta ifInOctets + \Delta ifOutOctets) \text{ x } 8 \text{ x } 100}{(number\ of\ second \text{ in } \Delta) \text{ x } ifSpeed}$$

2. *Full Duplex*

$$\frac{\max(\Delta ifInOctets + \Delta ifOutOctets) \text{ x } 8 \text{ x } 100}{(number\ of\ second \text{ in } \Delta) \text{ x } ifSpeed}$$

3. *Input/Output Utilization*

$$\text{Input Utilization} = \frac{\Delta ifInOctets \text{ x } 8 \text{ x } 100}{(number\ of\ second \text{ in } \Delta) \text{ x } ifSpeed}$$

$$\text{Output Utilization} = \frac{\Delta ifOutOctets \text{ x } 8 \text{ x } 100}{(number\ of\ second \text{ in } \Delta) \text{ x } ifSpeed}$$

2.5 Network Management Tools

Ada beberapa *tools* yang digunakan untuk mengatur sistem jaringan yang dapat di integrasikan dengan sistem yang akan dibuat, *tools* manajemen jaringan tersebut adalah:

2.5.1 IPTables

Untuk melakukan pengaturan keluar masuk atau tidak diijinkannya user untuk mengakses sebuah website maupun jaringan tertentu maka IPTables sangat diperlukan. IPTables adalah aplikasi *pre-instaled* di ubuntu.

IPTables merupakan *firewall* bawaan Linux. IPTables mampu melakukan *filtering dari layer transport* sampai *layer physical*. Sebagai contoh rule dalam sebuah *firewall* akan menutup semua koneksi kecuali ke port 80 protokol TCP, atau sebuah *rule firewall* mendefiniskan bahwa yang dapat melakukan koneksi hanya paket data yang berasal dari MAC *address* 00-80-48-24-3b-e5. Variabel-variabel dalam IPTables *Firewall* meliputi (Hartono, 2006):

1. Protokol
2. Port asal
3. Port tujuan
4. IP asal/Jaringan asal
5. IP tujuan/Jaringan tujuan
6. *Chain* (aliran)
7. *Code bit (flag).*

Berikut akan diberikan gambaran secara singkat tentang teknik paket dari IPTables (Purdy, 2004).

1. *Packet filtering*

Packet filtering adalah tipe paling dasar dari *network packet processing*. *Packet filtering* melibatkan pemeriksaan paket diberbagai titik ketika mereka bergerak melalui kode kernel jaringan dan membuat keputusan tentang bagaimana paket harus ditangani.

2. *Accouting*

Accounting melibatkan menggunakan byte dan/atau *packet counter* terkait dengan kriteria yang sesuai paket untuk memantau volume trafik jaringan.

3. *Connection Tracking*

Connection tracking menyediakan informasi tambahan yang dapat mencocokan paket terkait cara-cara yang tidak memungkinkan.

4. *Packet Mangling*

Packet mangling melibatkan pembuat perubahan untuk *header packet*(seperti alamat jaringan dan *port number*) atau *payloads.*

5. *Network Address Translation*(NAT)

NAT adalah jenis paket mangling yang melibatkan penggantian sumber dan alamat tujuan dan nomor *port*.

6. *Port Forwarding*

Port forwarding adalah tipe dari DNAT dimana salah satu komputer (seperti *firewall*) berperan sebagai *proxy* untuk satu atau lebih dari jumlah komputer.

7. *Load Balancing*

Load balancing mendistribusikan koneksi pada kelompok server sehingga mendapatkan hasil yang lebih tinggi.

2.5.2 HTB-Tools

HTB-Tools *Bandwith Management Software* adalah *software* yang membantu menyederhanakan proses alokasi *bandwith* baik *upload* maupun *download traffic* dengan menggunakan Linux kernel (BALAN & POTORAC, 2009).

2.6 Web Application Development

Dari sistem yang akan dibangun, digunakan aplikasi berbasis web. Berikut adalah penjelasan dari *tools* yang digunakan untuk mengembangkan sebuah aplikasi berbasis web:

2.6.1 Personal Home Page

*Personal Home Page (*PHP) adalah skrip bersifat *server-side* yang ditambahkan ke dalam *HyperText Markup Language* (HTML). Skrip ini akan membuat suatu aplikasi dapat diintegrasikan ke dalam HTML sehingga suatu halaman web tidak lagi bersifat statis, namun menjadi bersifat dinamis. Sifat *server-side* berarti pengerjaan skrip akan dilakukan di server, baru kemudian hasilnya dikirim ke browser (Kurniawan, 2002).

Keunggulan dari sifatnya yang *server-side* tersebut antara lain :

1. Tidak diperlukan kompatibilitas browser atau harus menggunakan browser tertentu, karena serverlah yang akan mengerjakan skrip PHP. Hasil yang dikirimkan kembali ke browser umumnya bersifat teks atau gambar saja sehingga pasti dikenal oleh bowser apa pun.

2. Dapat memanfaatkan sumber-sumber aplikasi yang dimiliki oleh server, misalnya koneksi ke database.

3. Skrip tidak dapat "diintip" dengan menggunakan fasilitas *view* HTML *sourcode*.

Kelebihan PHP dapat melakukan semua aplikasi program *Common Gateway Interface*(CGI), seperti mengambil nilai form, menghasilkan halaman web yang dinamis, serta mengirim dan menerima *cookie*. PHP juga dapat berkomunikasi dengan layanan-layanan yang mengunakan protokol IMAP, SNMP, NNTP POP3, HTTP, dan lain-lain. Namun kelebihan yang paling signifikan adalah kemampuannya untuk dapat melakukan koneksi yang baik dengan berbagai macam database.

2.6.2 Web Server

Web server, untuk berkomunikasi dengan *client*-nya (web browser) mempunyai protokol sendiri, yaitu *hypertext tarnsfer protocol*(HTTP). Dengan protokol ini, komunikasi antar web server dengan *client*-nya dapat saling dimengerti dan lebih mudah (Azmi, 2012).

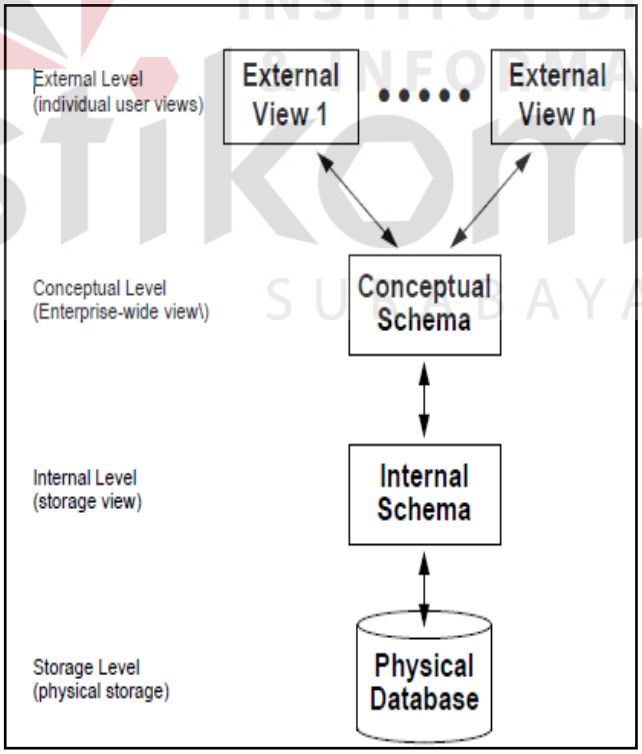
Web server bisa di deskripsikan dengan formula"*Web Server = Platform + Software + Information*". Web server haruslah komputer dengan koneksi ke internet, dengan sistem software untuk menjalankan komputer dan untuk dapat terhubung dengan sistem lain di internet (Yeager & McGrath, 1996).

2.6.2.1 Web Server Apache

Apache merupakan web server yang paling banyak dipergunakan di Internet. Program ini pertama kali didesain untuk sistem operasi lingkungan UNIX. Namun demikian, pada beberapa versi berikutnya Apache mengeluarkan programnya yang dapat dijalankan di Windows NT (Azmi, 2012).

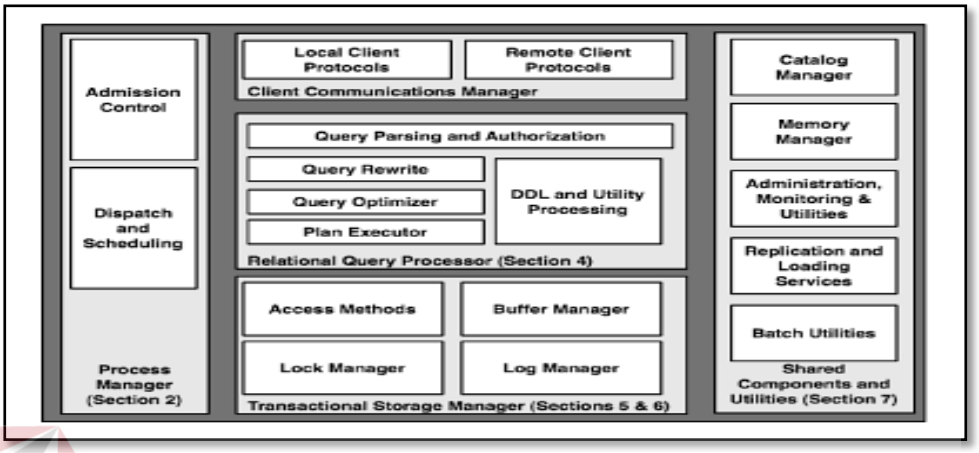
2.7 Konsep Dan Konfigurasi Database

*Database management system(*DBMS*)* adalah kumpulan program yang memungkinkan pengguna untuk membuat dan memelihara sebuah *database* (Robbins, 1995). Pada Gambar 2.4 akan digambar DBMS sebagai sistem *multy-layered.*



Gambar 2.4 DBMS sebagai *Multi-Layer*. Sumber (Robbins, 1995)

DBMS juga memiliki beberapa komponen utama, komponen utama DMBS dapat dilihat pada Gambar 2.5.



Gambar 2.5 Komponen Utama DBMS (Hellerstein, Stonebraker, & Hamilton, 2007)

2.7.1 MySQL

MySQL ada beberapa jenis yaitu *free license* dan *license*, untuk mendukung aplikasi ini berjalan cukup menggunakan MySQL versi *free license* dikarenakan kemudahan akses database dan support yang memudahkan *maintenance* aplikasi.

MySQL adalah *Relational Database Management Sistem* (RDBMS) yang didistribusikan secara gratis di bawah lisensi GPL (*General Public License*). Dimana setiap orang bebas untuk menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat *closed source* atau komersial (Prasetyo & Didik, 2003).

Berikut ini beberapa keistimewaan yang dimiliki oleh MySQL (Prasetyo & Didik, 2003) :

1. *Portability*
2. *Open Source*

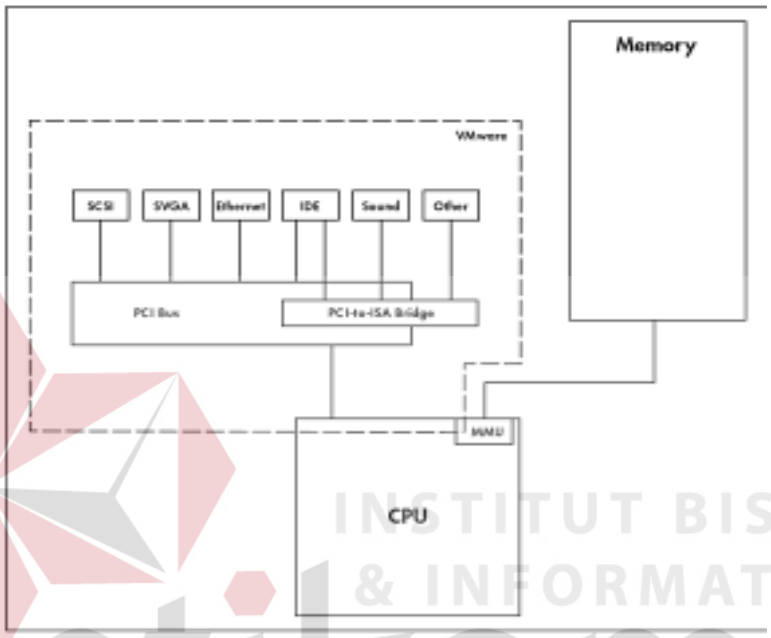
3. *Multiuser*
4. *Performance Tuning*
5. *Column Types*
6. *Command* dan *Functions*
7. *Security*
8. *Scalability* dan *Limits*
9. *Connectivity*
10. *Localisation*
11. *Interface*
12. *Lients* dan *Tools*

2.8 PC Router

Router adalah perangkat yang akan melewatkan paket IP dari suatu jaringan ke jaringan yang lain, menggunakan metode *addressing* dan *protocol* tertentu untuk melewatkan paket data tersebut. Router memiliki kemampuan melewatkan paket IP dari satu jaringan ke jaringan lain yang mungkin memiliki banyak jalur diantara keduanya. Router-router yang saling terhubung dalam jaringan internet turut serta dalam sebuah algoritma *routing* terdistribusi untuk menentukan jalur terbaik yang dilalui paket IP dari sistem ke sistem lain. Proses routing dilakukan secara *hop by hop*. IP tidak mengetahui jalur keseluruhan menuju tujuan setiap paket. IP routing hanya menyediakan IP *address* dari router berikutnya yang menurutnya lebih dekat ke *host* tujuan (Handriyanto, 2009).

2.9 VMware

VMware adalah sebuah komputer vitual yang berjalan diatas komputer dengan prosesor dan memory fisik atau sesungguhnya (Ward, 2002). Pada Gambar 2.6 akan ditunjukan *map* abstrak dari VMware Workstation PC.



Gambar 2.6 Map Abstrak Dari VMware Workstation PC. Sumber (Ward, 2002)

Berikut adalah beberapa *hardware* yang tersedia dalam VMware (Ward, 2002):

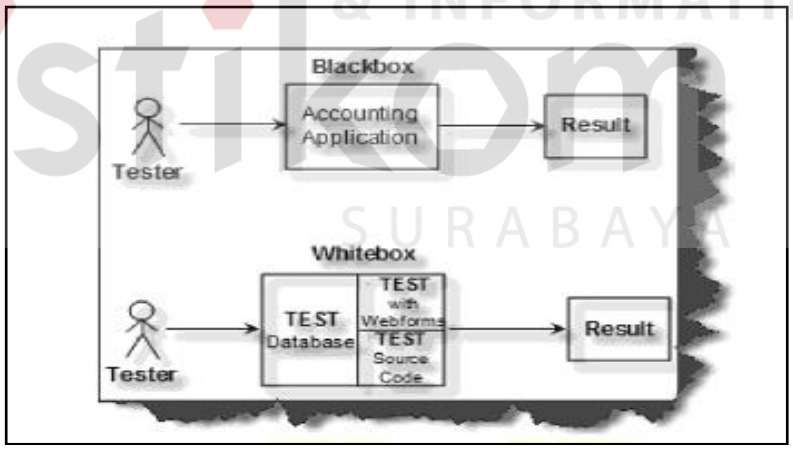
1. *IDE Disk and CD-ROM Drive*
2. *SCSI Disk*
3. *Floppy Drives*
4. *Ethernet Interface*
5. *Serial Ports*
6. *Parallel Ports*
7. *USB Interface*

8. *Graphics*
9. *Mouse*
10. *Sound Card*
11. *PC Bios*

2.10 Pengujian Sistem

Pengujian sistem untuk sistem ini menggunakan metode *black box*. *Black box testing* adalah strategi testing berbasis hanya pada spesifikasi dan kebutuhan. *Black box testing* tidak membutuhkan pengetahuan dari jalur internal, struktur ataupun implementasi dari software yang akan diuji (Koirala & Sbeikh, 2008).

Pada Gambar 2.7 akan dijelaskan bagaimana *white box* dan *black box testing* bekerja.



Gambar 2.7 White Box Dan Black Box Testing Bekerja. Sumber (Koirala & Sbeikh, 2008)

2.11 Object Oriented Design (OOD)

Object Oriented Design(OOD) adalah metode desain meliputi proses dekomposisi berorientasi objek dan notasi untuk menggambarkan sebuah model

baik secara fisik dan logis serta statis dan dinamis dari sebuah sistem pada desain (Booch, 1994).

2.12 UML (*Unified Modeling Language*)

Unified modelling language (UML) adalah bahasa standar yang digunakan untuk menjelaskan dan memvisualisasikan artifak dari proses analisis dan disain berorientasi objek. UML menyediakan standar pada notasi dan diagram yang bisa digunakan untuk memodelkan suatu sistem. Ada beberapa diagram yang disediakan dalam UML (Sholiq, 2006), antara lain:

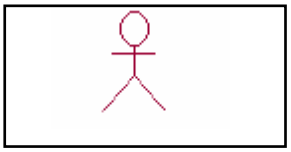
a. Diagram *use case (use case diagram)*
b. Diagram aktivitas *(activity diagram)*
c. Diagram sekuensial *(sequence diagram)*
d. Diagram kolaborasi *(collaboration diagram)*
e. Diagram kelas *(class diagram)*
f. Diagram *statechart (statechart diagram)*
g. Diagram komponen *(component diagram)*
h. Diagram *deployment (deployement diagram)*

Terdapat notasi pada UML, antara lain :

A. Actor

Actor adalah segala sesuatu yang berinteraksi dengan sistem aplikasi komputer. Jadi *actor* ini bisa berupa orang, perangkat keras, atau mungkin juga objek lain dalam sistem yang sama. Biasanya yang dilakukan oleh *actor* adalah

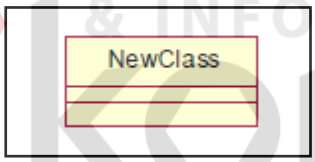
memberikan informasi pada sistem dan memerintahkan sistem untuk melakukan sesuatu. Pada Gambar 2.8 akan ditunjukan notasi aktor.



Gambar 2.8 Notasi Actor

B. Class

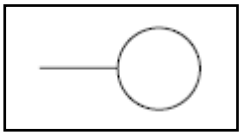
Class merupakan pembentuk utama dari sistem berorientasi objek karena *class* menunjukan kumpulan objek yang memiliki atribut dan operasi yang sama. *Class* digunakan untuk mengimplementasikan interface. Pada Gambar 2.9 akan ditunjukan notasi *class.*



Gambar 2.9 Notasi Class

C. Interface

Interface merupakan kumpulan operasi tanpa implementasi dari suatu *class*. Implementasi operasi dalam *interface* dijabarkan oleh operasi dalam *class*. Oleh karena itu keberadaan *interface* selalu disertai oleh *class* yang mengimplementasikan operasinya. *Interface* ini merupakan salah satu cara mewujudkan prinsip enkapsulasi dalam objek. Pada Gambar 2.10 akan ditunjukan notasi *interface*.



Gambar 2.10 Notasi Interface

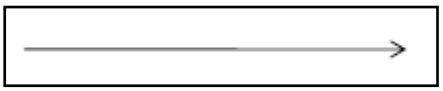
D. Use case

Use case menjelaskan urutan kegiatan yang dilakukan *actor* dan sitem untuk mencapai suatau tujuan tertentu. Walaupun menjelaskan kegiatan namun *use case* hanya menjelaskan apa yang dilakukan oleh *actor* dan sistem, bukan bagaimana *actor* dan sistem melakukan kegiatan tersebut. Pada Gambar 2.11 akan ditunjukan notasi *use case*.

Gambar 2.11 Notasi Use Case

E. Interaction

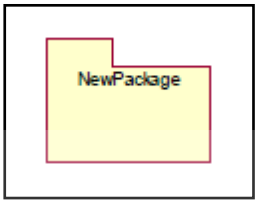
Interaction digunakan unutk menunjukan baik aliran pesan atau informasi antar objek. Biasanya *interaction* ini dilengkapi juga dengan teks bernama *operation signature* yang tersusun dari nama operasi, parameter yang dikirim dan tipe parameter yang dikembalikan. Pada Gambar 2.12 akan ditunjukan notasi *interaction.*



Gambar 2.12 Notasi Interaction

F. Package

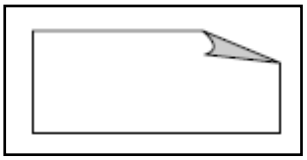
Package adalah kontainer atau wadah konseptual yang digunakan untuk mengelompokkan elemen-elemen dari sistem yang sedang dibangun, sehingga bisa dibuat model yang lebih sederhana. Tujuannya adalah untuk mempermudah penglihatan *(visibility)* dari model yang sedang dibangun. Pada Gambar 2.13 akan ditunjukan notasi *package.*



Gambar 2.13 Notasi Package

G. Note

Note digunakan untuk memberikan keterangan dan komentar tambahan dari suatu elemen sehigga bisa langsung terlampir dalam model. Note ini bisa ditempelkan ke semua elemen notasi yang lain. Pada Gambar 2.14 akan ditunjukan notasi *note.*

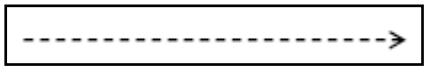


Gambar 2.14 Notasi Note

H. Dependency

Dependency merupakan relasi yang menunjukkan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain. Elemen yang ada di

bagian tanda panah adalah elemen yang tergantung pada elemen yang ada di bagian tanpa tanda panah. Pada Gambar 2.15 akan ditunjukan notasi *dependency.*



Gambar 2.15 Notasi Dependency

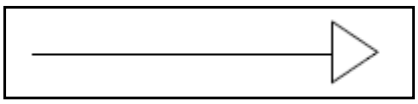
I. Association

Association menggambarkan navigasi antar *class (Navigation),* beberapa banyak objek lain yang bisa berhubungan dengan satu objek *(Multiplicity* antar *class),* dan apakah suatau *class* menjadi bagian dari *class* lainnya *(Aggregation).* Pada Gambar 2.16 akan ditunjukan notasi *association.*

Gambar 2.16 Notasi Association

J. Generalization

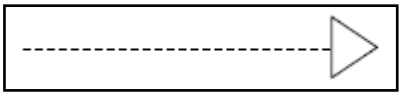
Generalization menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan *generalization, class* yang lebih spesifik *(subclass)* akan menurunkan atribut dan operasi dari *class* yang lebih umum *(superclass),* atau *"subclass is a superclass"*. Dengan menggunakan notasi *generalization* ini konsep *inheritance* dari prinsip hirarki dimodelkan. Pada Gambar 2.17 akan ditunjukan notasi Generalization.



Gambar 2.17 Notasi Generalization

K. Realization

Realization menunjukkan hubungan bahwa elemen yang ada di bagian tanpa panah akan merealisasikan apa yang dinyatakan oleh elemen yang ada di bagian dengan panah. Misalnya merealisasikan *package*, *component* merealisasikan *class* atau *interface*. Pada Gambar 2.18 akan ditunjukan Notasi *realization.*



Gambar 2.18 Notasi Realization

2.13 System Development Life Cycle

System development life Cycle (SDLC) adalah proses untuk memahami bagaimana sistem informasi (SI) dapat mendukung kebutuhan bisnis dengan merancang sebuah sistem, membangun, dan mengimplementasikan pada pengguna (Tegarden, Dennis, & Wixom, 2013). Ada beberapa tahap dalam SDLC:

1. Planning

   Tahap *planning* adalah proses mendasar dari pengertian kenapa sistem informasi harus dibuat dan menentukan bagaimana team proyek membuat sistem tersebut.

2. Analysis

   Tahap *analysis* adalah jawaban untuk pertanyaan siapa yang menggunakan sistem, apa yang akan sistem lakukan, dan dimana dan kapan sistem digunakan.

3. Design

Tahap *design* adalah tahapan untuk memutuskan bagaimana sistem akan beroperasi, dalam hal perangkat keras, *software*, dan infrastruktur jaringan juga dari *user interface*, *forms*, laporan dan spesifikasi program, *database*, dan *file* yang akan dibutuhkan.

4. Implementation

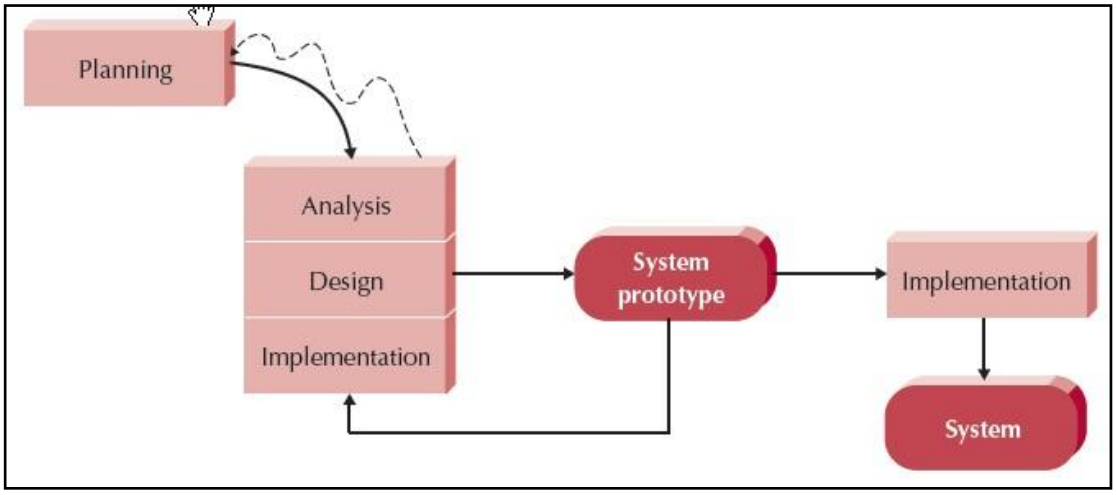
   Tahap akhir dari SDLC adalah *implementation*, selama sistem yang sebenarnya dibangun(atau dibeli, dalam hal desain paket perangkat lunak).

Dalam SLDC terdapat juga beberapa metodologi yang digunakan untuk pengembangan sebuah sistem, diantaranya:

1. Metode *Waterfall*
2. Metode *Parallel*
3. Metode *Raid Application Development*(RAD)
4. Metode *Phased*
5. Metode *Prototyping*
6. Metode *Throwaway Prototyping*
7. Metode *Agile*

2.14 Metode Prototyping

Metodologi berbasis *prototyping* melakukan fase analisis, desain, dan implementasi secara bersamaan, dan ketiga fase dilakukan secara berulang sampai sistem sempurna (Tegarden, Dennis, & Wixom, 2013). Berikut akan ditunjukan pada Gambar 2.19 *prototyping-based methology*.



Gambar 2.19 A Prototyping-Based Methodology

Keunggulan *prototyping* adalah :

1. Adanya komunikasi yang baik antara pengembang dan pelanggan.
2. Pengembang dapat bekerja lebih baik dalam menentukan kebutuhan pelanggan.
3. Pelanggan berperan aktif dalam pengembangan sistem.
4. Lebih menghemat waktu dalam pengembangan sistem.
5. Penerapan menjadi lebih mudah karena pemakai mengetahui apa yang diharapkannya

Sedangkan kelemahan *prototyping* adalah :

1. Pelanggan tidak melihat bahwa perangkat lunak belum mencerminkan kualitas perangkat lunak secara keseluruhan dan belum memikirkan peneliharaan dalam jangka waktu yang lama.
2. Pengembang biasanya ingin cepat menyelesaikan proyek sehingga menggunakan algoritma dan bahasa pemrograman sederhana.
3. Hubungan pelanggan dengan komputer mungkin tidak menggambarkan teknik perancangan yang baik.