BAB II

TINJAUAN PUSTAKA

II.1. Perancangan

Perancangan sistem adalah sebuah teknik pemecahan masalah yang saling melengkapi (dengan analisis sistem) yang merangkai kembali bagian-bagian komponen menjadi sistem yang lengkap-harapannya, sebuah sistem yang diperbaiki. Hal ini melibatkan penambahan, penghapusan, dan perubahan-perubahan bagian relatif pada sistem awal (Hanif Al Fatta; 2008 : 44).

Desain atau perancangan dalam pengembangan perangkat lunak merupakan upaya untuk mengkontruksi sebuah sistem yang memberikan kepuasan (mungkin informal) akan spesifikasi kebutuhan fungsional memenuhi target, memenuhi kebutuhan secara implisit dan eksplisit dari segi performansi maupun penggunaan sumber daya, kepuasan batasan pada proses desain dari segi biaya, waktu, dan peranggkat. (Rosa A.S, M. Shalahuddin ; 2011: 20)

II.2. Aplikasi

Aplikasi merupakan program komputer yang dibuat untuk menolong manusia melaksanakan tugas tertentu. Aplikasi berbeda dengan Sistem Operasi (yang menjalankan komputer), *Utility* (yang melaksanakan perawatan/tugas-tugas umum) dan Bahasa (yang digunakan untuk membuat program komputer), tergantung dari tujuan pekerjaan yang dimaksudkan, suatu aplikasi dapat memanipulasi teks, angka, kombinasi dari unsur-unsur tersebut.

(Jamroni ; 2005 : 14)

Aplikasi merupakan perangkat lunak yang dimasukan atau terdapat dalam komputer dan memiliki fungsi-fungsi khusus (Dendy Sugono; 2013: 85).

II.3. Game

Game adalah kata dari bahasa inggris yang berarti permainan atau pertandingan. *Game* bisa diartikan sebagai aktivitas terstruktur atau semi setruktur, yang biasanya dilakukan untuk bersenang-senang dan kadang digunakan sebagai alat pembelajaran. Adapun teori dari J.von neuman dan O. Morgenstern yang mengatakan "permainan terdiri atas sekumpulan peraturan yang membangun situasi situasi bersaing dari dua sampai beberapa orang atau kelompok dengan memilih strategi yang dibangun untuk memaksimalkan kemenangan sendiri atau pun meminimalkan kemenangan lawan. Peraturanperaturan menentukan kemungkinan tindakan untuk setiap pemain, sejumlah keterangan diterima setiap pemain sebagai kemajuan bermain, dan sejumlah kemenangan atau kekalahan dalam berbagai situasi". (Kurniawan Parbowo ; 2010 : 5)

Game adalah aktifitas terstruktur atau seni terstruktur, yang biasanya dilakukan untuk bersenang-senang dan kadang juga digunakan sebagai alat pembelajaran. Sebuah *game* bisa dikarakterisasi dari "apa yang pemain lakukan ", hal-hal tersebut adalah sebagai berikut :

a. Peralatan, seperti bola, kartu, papan dan pion, atau sebuah komputer
b. Peraturan, biasanya menentukan giliran permainan, hak dan keharusan masing-masing pemain dan tujuan permainan

c. *Skill*, strategi, dan keberuntungan. *Game* dengan *skill* misalnya dengan kekuatatan fisik seperti gulat, menembak, kekuatan mental seperti catur

d. *Single player game*, permainan tunggal ini adalah permainan dengan keahlian, berpacu dengan waktu atau keberuntungan

Game juga terbagi dalam beberapa jenis sebagai berikut:

a. Olahraga atau *sport*

b. Permainan papan atau *board games*

c. Permainan kartu atau *card games*

d. *Video* games dengan komputer atau alat elektronik untuk *game* lainnya. *Game* degan media komputer disebut juga komputer *game*.

(Lukman Hakim ; 2008 : 9)

II.4. Pac-Man

Pac-Man adalah sebuah permainan video *arcade* yang dikembangkan oleh Namco dan dipublikasikan oleh Midway. *Game* ini pertama kali dirilis di Jepang pada 22 Mei 1980. Pada awalnya, *Pac-Man* hanya dirilis untuk *arcade* saja. Namun dalam perkembangannya, *Pac-Man* yang masih populer hingga kini dirilis pula dalam *platform* lainnya seperti *Game Boy*, SNES, bahkan *Playstation* dan konsol *game* modern. Perancang permainan ini adalah Toru Iwatani, yang merupakan karyawan Namco. Pemain harus mengontrol tokoh berwarna kuning bernama *Pac-Man* dan membawanya mengelilingi labirin sambil "memakan" *pac-dots* (titik-titik). Pada saat permainan dimulai, terdapat empat hantu yang akan keluar dari

persembunyiannya, lalu berkeliling di labirin tersebut untuk menangkap *Pac-Man*. Sang pemain dapat menyelesaikan satu level (tingkat) jika berhasil memakan seluruh *pac-dots*, baik yang berjenis standar maupun spesial. Selain itu, di labirin pun terdapat item yang sesekali muncul untuk menambah skor jika dimakan. (Emeraldy Widiyadi ; 2011 : 1)

II.5. Pemrograman Game

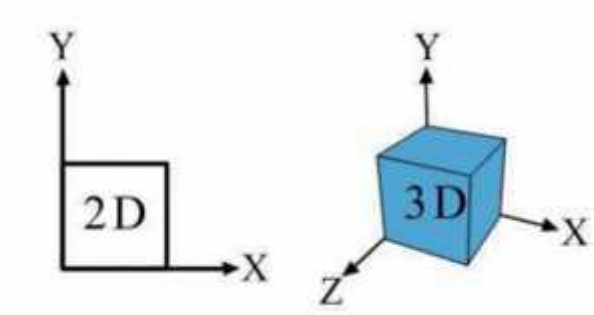
Pemrograman *game* adalah sebuah upaya untuk merancang sebuah aplikasi *game* dengan menggunakan *tools* tertentu. Dalam pemrograman *game* diperlukan *compiler* bahasa pemrograman yang bersifat *general purposes* seperti *C++, Java, Visual Basic, Delphi* dan lain-lain. Dalam pemrograman *game* diperlukan pemahaman yan sangat baik dalam bidang matematika dan fisika. Beberapa materi dalam bidang matematika seperti garis dan titik, geometri, trigonometri, operasi vektor, operasi matriks, transformasi, konversi unit, pergerakan dalam satu dimensi, pendekatan derivatif dalam gerakan satu dimensi, serta pergerakan dalam dua dan tiga dimensi. Selain matematika, bidang fisika juga sangat banyak dibutuhkan dalam pemrograman *game*, materi-materi di bidang fisika yang harus dikuasai oleh seorang pemrogram *game* antara lain hukum newton, energi, tabrakan dan momentum, serta perputaran gerak (Lukman Hakim ; 2008 : 10)

II.6. 3 Dimensi

Dimensi yakni ukuran, matra, terdiri atas satuan panjang, lebar, tinggi.2 Dimensi berarti hanya terdiri atas panjang dan lebar, sedangkan 3 dimensi berarti terdiri atas panjang, lebar dan tinggi .(Dendy Sugono, dkk, 2008 : 354).

Animasi 3 Dimensi adalah Animasi objek dalam bentuk 3 dimensi . Animasi 3D mudah untuk di deskripsikan, tapi lebih sulit untuk dikerjakan. *Properties* 3D model didefinisikan dengan angka-angka. Dengan merubah angka bisa merubah posisi objek, rotasi, karakteristik permukaan, dan bahkan bentuk. 3D adalah dimensi yang memiliki ruang. Jika kita merujuk pada objek yang memiliki 3D, artinya objek tersebut memiliki ruang dan volume. Objek 3D juga memiliki lokasi pada koordinat x, y dan z.

(http://power.lecture.ub.ac.id/files/2011/03/Animasi-3-Dimensi.pdf)



Gambar II.1. Bidang 2 Dimensi Dan 3 Dimensi
(Sumber : http://power.lecture.ub.ac.id/files/2011/03/Animasi-3-Dimensi.pdf)

II.7. Visual C++

Jauh sebelum *Visual C++.Net* dibuat, sekitar tahun 1970-an, Dennis Ritchie membuat bahasa pemrograman yang bernama bahasa *C*. salah satu keunggulan *C* dibanding bahasa yang lain, adalah kemampuan untuk mengakses

memori secara manual. Bahasa *C* terus mengalami perbaikan dan pengembangan, hingga pada sekitar tahun 1980-an dibuatlah bahasa pemrograman *C++*. Perubahan terbesar pada *C++* adalah bahasa ini telah menerapkan konsep pemrograman berorientasi objek. Dalam kurun waktu 1992-1998, *Microsoft* mengeluarkan *Visual C++* yang dimulai dari versi *Visual C++* versi 1.0 sampai dengan versi 6.0. pada tahun 2002, *Microsoft* mengeluarkan versi *Visual C++* yang telah menggunakan *.Net Framework*. Dengan menggunakan *framework* ini, program yang dibuat pada suatu *platform*, bisa dijalankan pada suatu *platform* yang lain, selama terdapat *.Net Framework* yang sesuai. Pada tahun 2003, *Visual C++.Net* 2003 dirilis dalam paket *Microsoft Visual Studio* 2003. Dan dua tahun kemudian, *Visual Studio* 2005 dirilis dengan menyertakan *Visual C++.Net* 2005 dan *.Net Framework* versi 2.0. Pada November 2007, telah dirilis *Visual C++.Net* 2008 dan *.Net Framework 3.5* (Suharian Ramadi; 2009 : 2)

II.8. Pemodelan UML

Pemodelan adalah gambaran dari realita yang simpel dan dituangkan dalam bentuk pemetaan dengan aturan tertentu. (Rosa A.S, M. Shalahuddin; 2011: 116)

Pada perkembangan teknik pemrograman beroientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Languange* (UML). *UML* hanya bergungsi untuk melakukan pemodelan. Jadi penggunaan *UML* tidak terbatas pada metodologi tertentu,

meskipun pada kenyataannya *UML* paling banyak digunakan pada metodologi berorientasi objek. (Rosa A.S, M. Shalahuddin; 2011: 118)

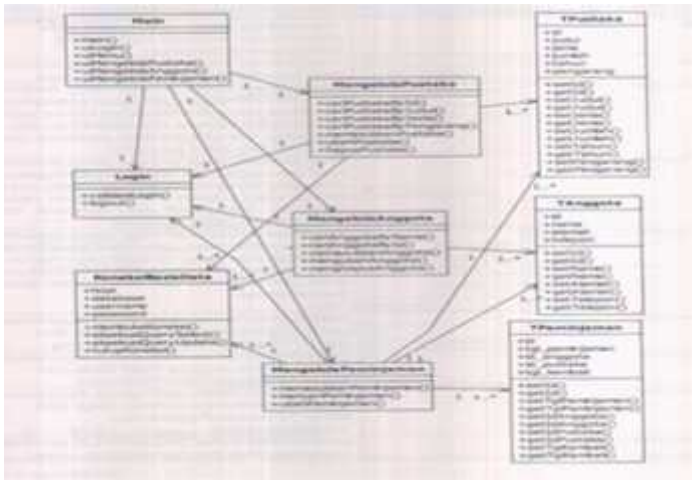
Diagram *UML* terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori yaitu :

1. Structure Diagram

Yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. Yang termasuk dalam structure diagram adalah sebagai berikut:

a. Class diagram

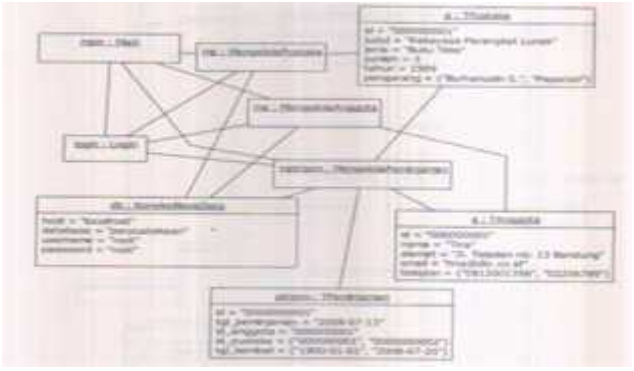
Digaram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefenisian kelas-kelas yang akan dibuat untuk membangun sistem. (Rosa A.S, M. Shalahuddin; 2011: 122). Contoh *class diagram* dapat dilihat pada gambar II.2.



Gambar II.2. Contoh *Class* Diagram
Sumber : (Rosa. A.S. dan M. Shalahuddin ; 2011 : 161)

b. Object diagram

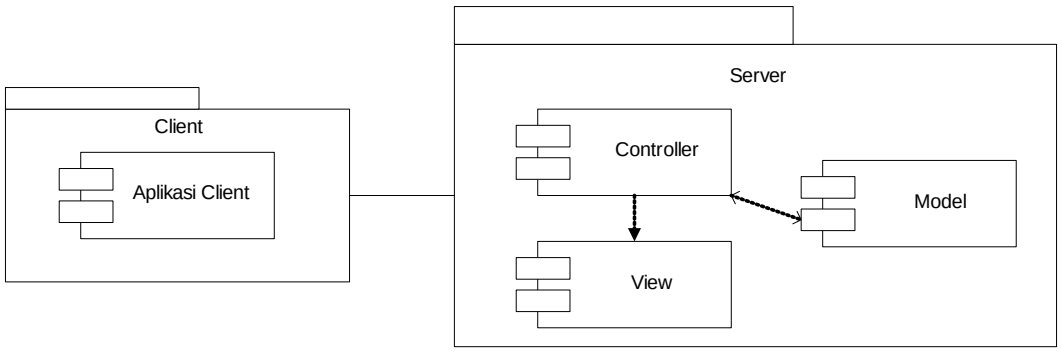
Diagram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. (Rosa A.S, M. Shalahuddin; 2011: 124)



Gambar II.3. Contoh *Object* Diagram
Sumber : (Rosa A.S. dan M. Shalahuddin ; 2011 : 163)

c. Component diagram

Diagram komponen atau *component diagram* dibuat untuk menunjukkan organisasi dan ketergantungan di antara kumpulan komponen di dalam sebuah sistem. (Rosa A.S, M. Shalahuddin; 2011: 125)



Gambar II.4. Contoh *Component* Diagram
Sumber : (Rosa A.S, M. Shalahuddin; 2011: 125)

d. Composite diagram

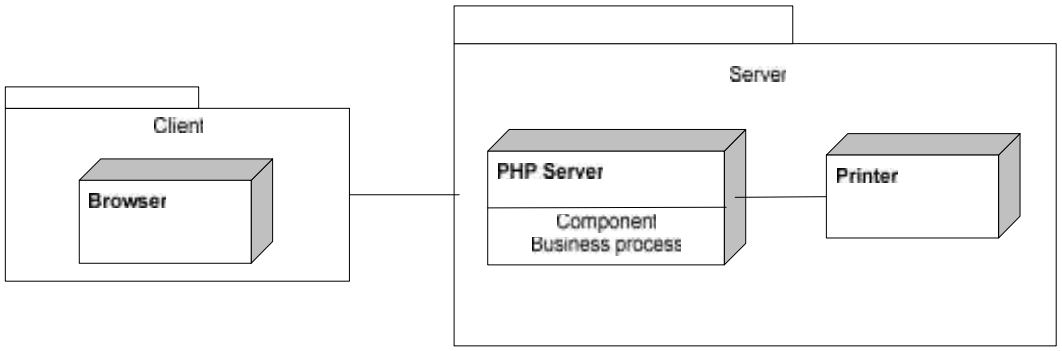
Composite strukture diagram baru mulai ada pada UML versi 2.0, pada versi 1.x diagram ini belum muncul. Digaram ini dapat digunakanuntuk menggambarkan struktur dari bagian-bagian yang saling terhubung maupun mendeskripsikan struktur pada saat berjalan (*runtime*) dari *instance* yang saling terhubung. (Rosa A.S, M. Shalahuddin; 2011: 127)

e. Package diagram

Diagram ini meyediakan cara mengumpulkan elemen-elemen yang saling terkait dalam diagram UML. (Rosa A.S, M. Shalahuddin; 2011: 128)

f. Deployment diagram

Diagram deployment atau *deployment diagram* menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi. (Rosa A.S, M. Shalahuddin; 2011: 129)



Gambar II.5. Contoh *Deployment* Diagram

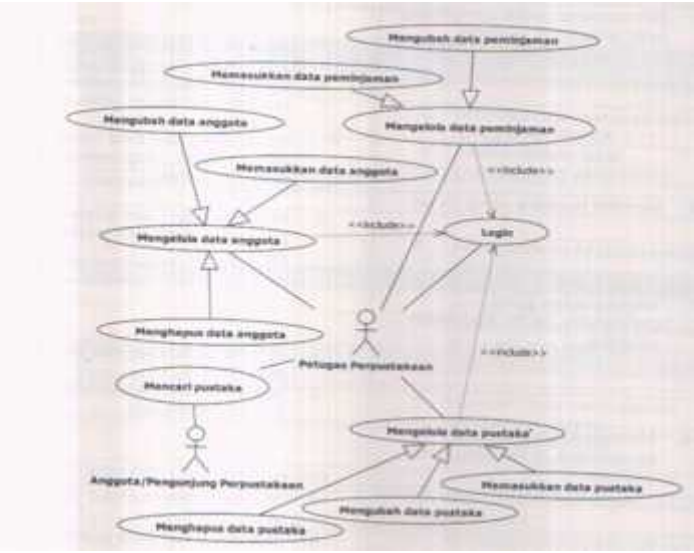
Sumber : (Rosa A.S, M. Shalahuddin; 2011: 129)

2. Behavior Diagram

Yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. Yang termasuk dalam *bahvior* diagram adalah sebagai berikut :

a. Use case

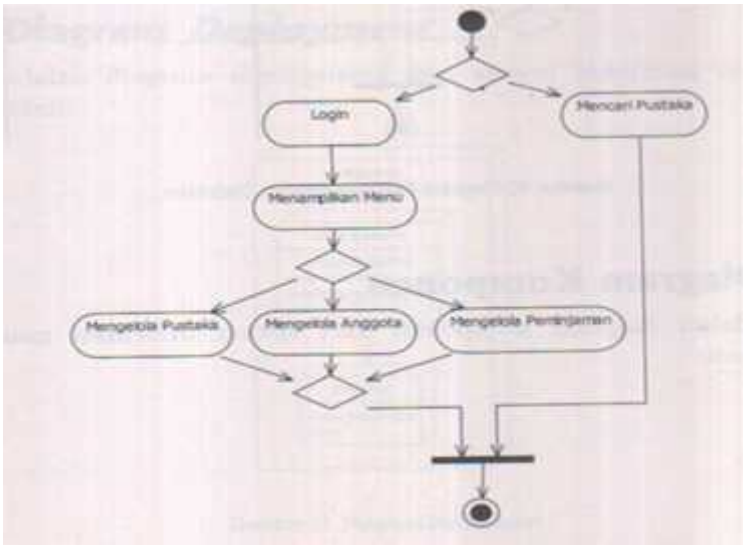
Use case atau diagram *use case* merupakan pemodelan untuk kelakuan (*bahavior*) sistem informasi yang akan dibuat. (Rosa A.S, M. Shalahuddin; 2011: 130). Lihat gambar II.6



Gambar II.6. Contoh *Use Case* Diagram
Sumber : (Rosa A.S. dan M. Shalahuddin ; 2011 : 160)

b. Activity diagram

Digaram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. (Rosa A.S, M. Shalahuddin; 2011: 134). Lihat gambar II.7

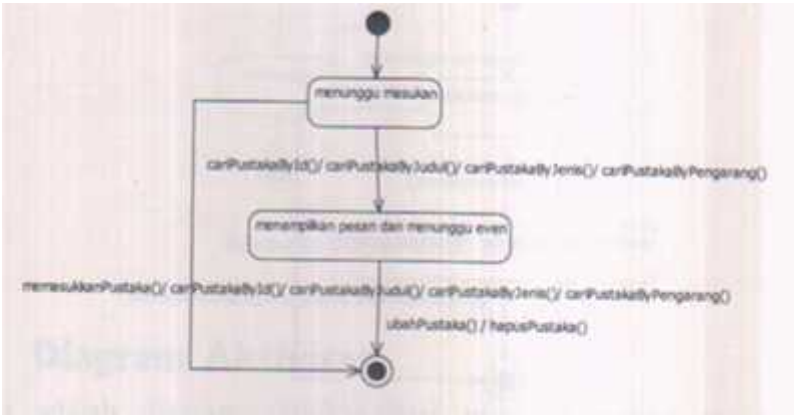


Gambar II.7. Contoh *Activity* Diagram

Sumber : (Rosa A.S. dan M. Shalahuddin ; 2011 : 177)

c. State machine diagram

Diagram mesin status digunakan untuk menggambarkan perubahan status atau transisi status dari sebuah mesin atau sistem. (Rosa A.S, M. Shalahuddin; 2011: 136).



Gambar II.8. Contoh *State Machine* Diagram

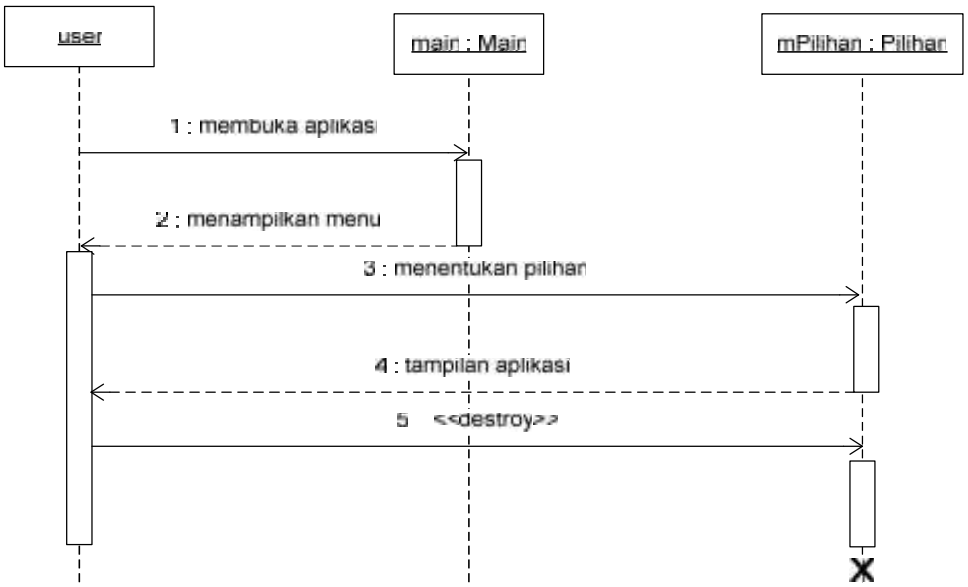
Sumber : (Rosa A.S, M. Shalahuddin; 2011: 174)

3. Interaction Diagram

Yaitu kumpulan diagram yang digunkan untuk menggambarkan interaksi antar sub sistem pada suatu sistem. Yang termasuk dalam *interaction* diagram adalah sebagai berikut :

a. Sequence diagram

Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. (Rosa A.S, M. Shalahuddin; 2011: 137). Lihat gambar II.9



Gambar II.9. Contoh *Sequence* Diagram

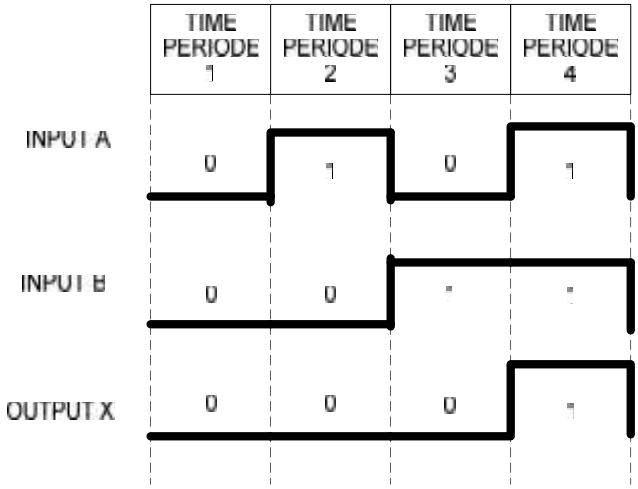
Sumber : (Rosa A.S, M. Shalahuddin; 2011: 164)

b. *Communication diagram*

Communication diagram atau diagram komunikasi menggambarkan interaksi antar objek /bagian dalam bentuk urutan pengiriman pesan. (Rosa A.S, M. Shalahuddin; 2011: 140).

c. *Timing diagram*

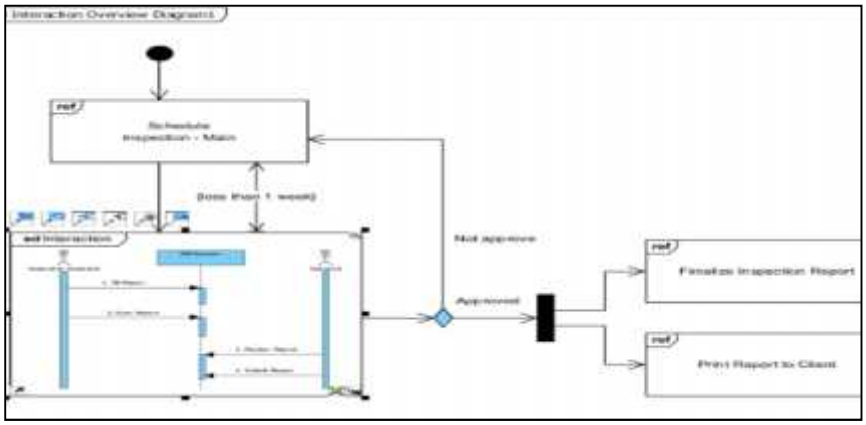
Timing diagram merupakan diagram yang fokus pada penggambaran terkait batasan waktu. (Rosa A.S, M. Shalahuddin; 2011: 141). Lihat gambar II.9



Gambar II.10. Contoh *Timing* Diagram
Sumber : (Rosa A.S, M. Shalahuddin; 2011: 142)

d. *Interaction overview diagram*

Diagram ini mirip dengan diagram aktivitas yang berfungsi untuk menggambarkan sekumpulan urutan aktivitas. (Rosa A.S, M. Shalahuddin; 2011: 143)



Gambar II.11. Contoh *Interaction Overview* Diagram

Sumber : (Rosa A.S, M. Shalahuddin; 2011: 145)

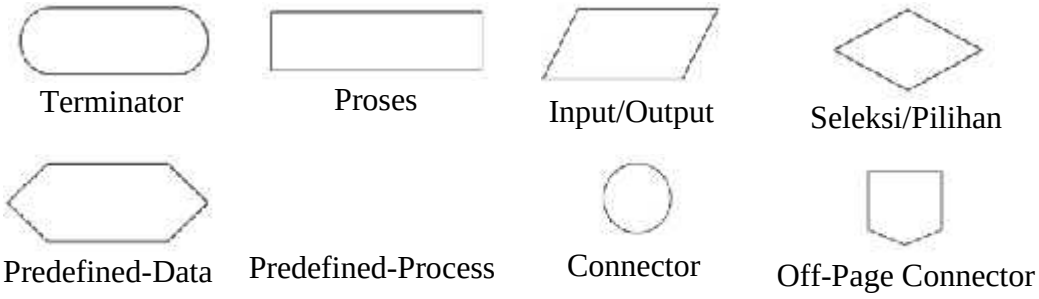
II.9. Flowchart

Tujuan utama dari pengunaan *flowchart* adalah untuk menggambarkan suatu tahapan penyelesaian masalah secara sederhana, terurai, rapi dan jelas dengan menggunakan simbol-simbol yang standar. (Budi Sutejo dan Michael AN, 2004 : 46)

II.9.1 Simbol-simbol *Flowchart*

Simbol-simbol *flowchart* dapat diklasifikasikan menjadi simbol untuk program dan simbol untuk sistem (perlatan *hardware*)

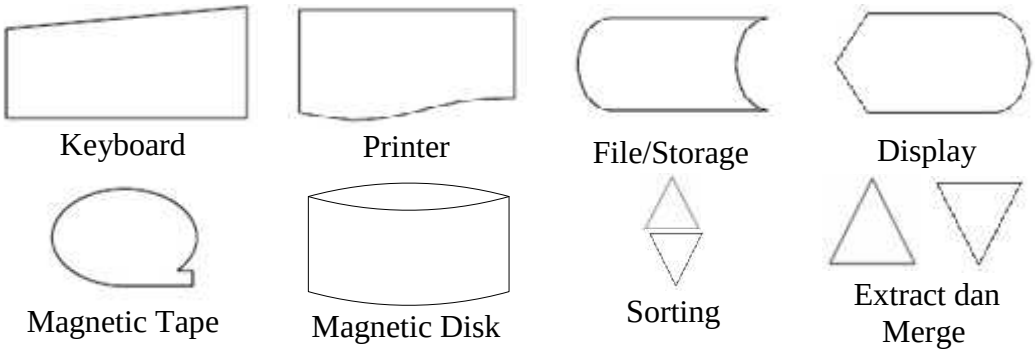
1) Program *Flowchart*



Gambar II.12. Simbol Program Flowchart

Sumber : (Suarga, 2006 : 6)

2) *System Flowchart*



Gambar II.13. Simbol Program Flowchart

Sumber : (Suarga, 2006 : 7)