BAB 2
TINJAUAN PUSTAKA

2.1. Media Pembelajaran

Kata “media” berasal dari kata latin, merupakan bentuk jamak dari kata “medium”. Secara harfiah kata tersebut mempunyai arti perantara atau pengantar. Banyak pakar dan juga organisasi yang memberikan batasan mengenai pengertian media. Beberapa diantaranya mengemukakan bahwa media pembelajaran merupakan wadah dari pesan, dan materi yang ingin disampaikan adalah pesan pembelajaran, sedangkan tujuan yang ingin dicapai ialah proses pembelajaran. Penggunaan media secara kreatif akan memperbesar kemungkinan bagi penggunanya untuk belajar lebih banyak, mencamkan apa yang dipelajari lebih baik, dan meningkatkan penampilan dalam melakukan keterampilan sesuai dengan yang menjadi tujuan pembelajaran.

Media pembelajaran yang sedang berkembang saat ini yaitu media pembelajaran interaktif, dimana pengguna tidak memperhatikan objek atau media saja, tetapi pengguna dapat berinteraksi selama mengikuti pembelajaran. Didalam media pembelajaran sendiri terdapat beberapa aspek penting yaitu audio, visual, dan informasi. Dalam hal ini *game* juga bisa sebagai media pembelajaran, dikarenakan *game* mengandung ketiga aspek tersebut. Jadi permainan pendidikan dan simulasi yang berorientasikan pada masalah memiliki potensi untuk memberikan pengalaman belajar yang merangsang minat dan realistis. Oleh karena itu media pembelajaran dianggap sebagai sumber terbaik dalam hal media pembelajaran.

2.2. *Game*

Game merupakan salah satu media hiburan yang paling popular untuk semua kalangan usia. Sejak pertama kali ditemukan sampai saat sekarang, teknologi *game* telah mengalami kemajuan yang terbilang sangat pesat. Hal ini ditandai dengan berkembangnya jenis, produk, alat, dan jenis interaksi *game* dengan penggunaan yang semakin beragam bentuknya.

2.2.1. Pengertian *Game*

Game berasal dari kata bahasa Inggris yang memiliki arti dasar Permainan. Permainan dalam hal ini merujuk pada pengertian "kelincahan intelektual" (*intellectual playability*). Permainan terdapat berbagai aturan yang harus dipahami oleh penggunanya dan didalam *game* perlu adanya skenario dan tujuan agar alur permainan jelas dan terarah. Skenario bisa meliputi *setting*, map, *level*, alur cerita, hingga efek yang ada dalam *game* tersebut. Pengertian permainan dikemukakan oleh John von Neumann and Oskar Morgenstern [3] yang berisi:

"Permainan terdiri atas sekumpulan peraturan yang membangun situasi bersaing dari dua sampai beberapa orang atau kelompok dengan memilih strategi yang dibangun untuk memaksimalkan kemenangan sendiri atau pun untuk meminimalkan kemenangan lawan. Peraturan-peraturan menentukan kemungkinan tindakan untuk setiap pemain, sejumlah keterangan diterima setiap pemain sebagai kemajuan bermain, dan sejumlah kemenangan atau kekalahan dalam berbagai situasi".

2.2.2. Sejarah Singkat *Game*

Electronic Game atau selanjutnya dapat disebut sebagai *video game*, pertama kali ditemukan oleh Thomas T. Goldsmith Jr. dan Estle Ray Mann. Penemuan ini dipatenkan pada januari 1947. Penemuan yang mendasari perkembangan saat ini adalah ketika para penemu *video game* menggunakan

Cathode-Ray Tube, yaitu sebuah tabung *vacuum* yang digunakan sebagai media untuk membuat simulasi kecepatan tembakan dan arah tembakan sebuah roket. Pada Februari 1951, Christopher Strachey memulai perkembangan kearah pemrograman yang mulai menggunakan memori di mana aplikasinya diterapkan untuk kebutuhan para pilot. Penemuan baru terus berkembang hingga tahun 1959 dan memunculkan berbagai perusahaan yang bisa disebut sebagai *developer game*.

Saat ini perkembangan *game* di komputer sangat cepat. Para pengelola industri *game* berlomba-lomba untuk menciptakan *game* yang lebih nyata dan menarik untuk para pemainnya. Hal inilah yang membuat perkembangan *game* di komputer sangat cepat. Sehingga *game* bukan hanya sekedar permainan untuk mengisi waktu luang atau sekedar hobi. Melainkan sebuah cara untuk meningkatkan kreatifitas dan tingkat intelektual para penggunanya.

2.2.3. Jenis-Jenis *Game*

Berikut ini akan dijabarkan beberapa jenis *game* berdasarkan cara pembuatannya, cara pemasarannya, mesin yang menjalankannya. Jenis *game* diantaranya adalah [7]:

1. *Game* Arcade

   Arcade *game*, yaitu yang sering disebut ding-dong di Indonesia, biasanya berada di daerah / tempat khusus dan memiliki box atau mesin yang memang khusus di design untuk jenis video *game* tertentu dan tidak jarang bahkan memiliki fitur yang dapat membuat pemainnya lebih merasa berada di dalam *game*, seperti pistol, kursi khusus, sensor gerakan, sensor injakkan dan stir mobil.

2. *Game* PC

   Game yang dimainkan pada PC (*Personal Computer*) yang memiliki kelebihan yaitu memiliki tampilan antarmuka yang baik untuk input maupun output, output visual kualitas tinggi karena layar

computer biasanya memiliki resolusi yang jauh lebih tinggi dibandingkan dengan layar televisi biasa. Kekurangannya adalah spesifikasi computer yang sangat bervariasai antara satu computer dengan computer lainnya menyebabkan beberapa *game* dapat ditampilkan dengan baik pada satu computer, tetapi tidak berjalan dengan baik pada computer yang lainnya

3. *Game* Konsol

   Game yang dijalankan pada suatu mesin spesifik yang biasanya tersedia di rumah pribadi, seperti Microsoft Xbox 360, Nintendo Wii dan sebagainya

4. *Game Handheld*

   Handheld game, yaitu yang dimainkan di *console* khusus video *game* yang dapat dibawa kemana-mana, contoh Nintendo DS dan Sony PSP.

5. *Game Mobile*

   Mobile game, yaitu yang dapat dimainkan atau khusus untuk mobile phone, PDA, dan Smartphone

2.2.4. *Genre Game*

Game dibagi-bagi lagi ke dalam beberapa *genre*, diantaranya yaitu [7] :

1. Aksi – Shooting

   Shooting merupakan salah satu *genre* dalam *game* yang biasanya mensimulasikan tentang tembak-tembakan, atau hajar-hajaran bisa juga tusuk-tusukan, tergantung cerita dan tokoh di dalamnya, video *game* jenis ini sangat memerlukan kecepatan refleks, koordinasi mata-tangan, juga timing, inti dari *game* jenis ini adalah tembak

   Termasuk didalam-nya :

a. *First person shooting* (FPS)
b. Drive n' shoot
c. Shoot em' up
d. Beat 'em up
e. Light gun shooting

2. Fighting (pertarungan)

Game jenis pertarungan merupakan *game* yang mengusung *gameplay* tentang bertarung. *Game* ini memang memerlukan kecepatan refleks dan koordinasi mata-tangan, tetapi inti dari *game* ini adalah penguasaan *gameplay* kontrol, pengenalan karakter dan *timing* sangatlah penting, *combo*-pun menjadi esensial untuk mengalahkan lawan. Seri Street Fighter, Tekken, Mortal Kombat, Soul Calibur dan King of Fighter adalah contohnya.

3. Petualangan.

Video g*ame* murni petualangan lebih menekankan pada jalan cerita dan kemampuan berpikir pemain dalam menganalisa tempat secara visual, memecahkan teka-teki maupun menyimpulkan rangkaian peristiwa dan percakapan karakter hingga penggunaan benda-benda tepat pada tempat yang tepat. Termasuk didalamnya:

a. Petualangan dengan teks atau sistem tunjuk dan klik, contoh: Kings Quest, Space Quest, Heroes Quest, Monkey Island, Sam and Max,
b. Novel atau film interaktif, seperti *game* "dating" yang banyak beredar di jepang, Dragons Lair dan Night Trap.

4. Simulasi, Konstruksi dan manajemen.

   Video Game jenis ini seringkali menggambarkan dunia di dalamnya sedekat mungkin dengan dunia nyata dan memperhatikan dengan detil berbagai faktor. *Video Game* jenis ini membuat pemain harus berpikir untuk mendirikan, membangun dan mengatasi masalah dengan menggunakan dana yang terbatas. Contoh: Sim City, The Sims, Tamagotchi.

5. *Role Playing*.

   Video game jenis ini sesuai dengan terjemahannya, bermain peran, memiliki penekanan pada tokoh/peran perwakilan pemain di dalam permainan, yang biasanya adalah tokoh utamanya, dimana seiring memainkannya, karakter tersebut dapat berubah dan berkembang ke arah yang diinginkan pemain dalam berbagai parameter yang biasanya ditentukan dengan naiknya level, baik dari status kepintaran, kecepatan dan kekuatan karakter, secara kebudayaan. Contohnya Final Fantasy, Dragon Quest dan Xenogears. Sedangkan pengembang *game* RPG Eropa, cenderung membuat karakter kita bebas memilih jalan cerita sendiri secara non-linear, seperti Ultima, Never Winter Nights, baldurs gate, Elder Scroll, dan Fallout.

6. Strategi.

   Jenis *video game* strategi kebalikan dari *video game* jenis aksi. Dimana *game* aksi yang berjalan cepat dan perlu reflek yang cepat, *video game* strategi biasanya memberikan pemain atas kendali tidak hanya satu orang tapi minimal sekelompok orang dengan berbagai jenis tipe kemampuan, kendaraan, hingga pembangunan berbagai bangunan, pabrik dan pusat pelatihan tempur, tergantung dari tema

ceritanya. Pemain *game* strategi melihat dari sudut pandang lebih meluas dan lebih kedepan dengan waktu permainan yang biasanya lebih lama dan santai dibandingkan *game* aksi.

Game jenis ini terbagi atas:

a. *Real time Strategy*
b. *Turn based Strategy*

7. Puzzle.

   Video game jenis ini sesuai dengan namanya yang ber-intikan mengenai pemecahan teka-teki, sering pula permainan jenis ini mengandung unsur permainan dalam *video game* petualangan maupun *game* edukasi. Contohnya seperti *game* Tetris, Minesweeper, Bejeweled, Sokoban dan Bomberman.

8. Olahraga.

   Game jenis ini permainannya diusahakan serealistik mungkin walau kadang ada yang menambah unsur fiksi seperti NBA JAM. Contohnya pun jelas, Seri Winning Eleven, seri NBA, seri FIFA, John Madden NFL, Lakers vs Celtics, Tony hawk pro skater.

9. *Game* Edukasi

   Game edukasi adalah sebuah *game* yang bertujuan untuk sebuah pembelajaran. Biasanya *game* edukasi sering dipakai oleh anak-anak, tapi para remaja juga suka memainkan *game* ini. Anak-anak lebih suka bermain daripada belajar, maka dari itu banyak vendor menciptakan sebuah *game* edukasi.

10. *Game Entertainment*

Game entertainment merupakan permainan yang lebih memprioritaskan tingkat hiburan bagi para penggunanya, sehingga aspek hiburan terhadap pemain sangat diprioritaskan.

2.3. Algoritma *Branch and Bound*

Algoritma *Branch and Bound* (B&B) pertama kali dikemukakan oleh A. H. Land dan A. G. Doig pada tahun 1960 untuk menyelesaikan masalah pemrograman linear. Pemrograman linear meliputi seluruh masalah optimasi dan fungsi-fungsi linear, yang dapat dinyatakan sebagai persamaan linear dengan batasan-batasan tertentu. Pemrograman linear umumnya digunakan untuk menemukan cara untuk memperoleh hasil optimum, misalnya keuntungan maksimal atau biaya minimal, dengan sejumlah sumber daya (misalnya waktu, uang, dan tempat) yang diberikan. Algoritma B&B merupakan metode pencarian di dalam ruang solusi secara sistematis.

Algoritma B&B mengorganisasikan ruang solusi ke dalam pohon ruang status. Berbeda dengan algoritma runut-balik, yang membentuk pohon ruang status dengan menggunakan skema BFS (*Breadth First Search*), algoritma B&B membentuk pohon status dengan menggunakan skema DFS (*Depth First Search*). Simpul hidup pada pohon ruang status yang dibentuk oleh algoritma B&B memiliki ongkos tertentu, yang dinotasikan dengan simbol c(i), i menyatakan simbol simpul, c(i) menyatakan nilai batas (bound) dari simpul i.

Tiga jenis simpul yang terdapat dalam pohon ruang status :

1. Simpul hidup, yaitu simpul yang belum diperluas (di-expand)
2. Simpul mati, yaitu simpul yang tidak dapat diperluas lagi
3. Simpul-E, yaitu simpul hidup dengan ongkos terbaik (umumnya terbesar atau terkecil), yang akan diperluas.

Algoritma B&B memiliki tiga buah prosedur yang digunakan dalam membentuk pohon ruang status, antara lain :

1. Branching, atau perluasan

   Branching adalah proses pembentukan pohon ruang solusi dengan mengperluas simpul-E yang terdapat pada pohon ruang solusi.
2. Bounding, atau pemberian nilai batas

   Bounding adalah proses pemberian nilai batas pada semua simpul hidup yang terdapat pada pohon ruang solusi.
3. Pruning, atau penghapusan

   Pruning adalah penghapusan simpul hidup dari daftar simpul yang akan diperluas, membuat simpul tersebut menjadi simpul mati.

Pruning dapat dikatakan sebagai proses mematikan simpul, misalnya simpul x, dimana program menganggap simpul x tidak akan menghasilkan status akhir (solusi), sehingga simpul x tidak akan diperluas oleh program. Umumnya program akan memperluas simpul yang memiliki ongkos terbesar atau terkecil, dan melakukan pruning pada simpul-simpul sisanya. Proses pruning ini mengakibatkan kompleksitas waktu algoritma B&B jauh lebih baik dibandingkan algoritma runut-balik, yang sama-sama membangun pohon ruang solusi untuk memperoleh penyelesaian dari permasalahan.

Ongkos dari setiap simpul hidup dihasilkan dalam proses bounding. Notasi umum dari ongkos :

$$c(i)=f(i)+g(i) \quad (2.1)$$

Keterangan :

f(i) = ongkos untuk mencapai simpul i dari akar

g(i) = ongkos untuk mencapai simpul tujuan dari akar

c(i) = ongkos total simpul i

Simpul-E adalah simpul hidup yang memiliki ongkos terbesar.

Tahap-tahap pencarian solusi :

1. Memperluas simpul-E untuk setiap kemungkinan kombinasi penempatan pasien pada ruangan
2. Menghitung ongkos untuk setiap simpul hidup Simpul hidup adalah kondisi di mana setidaknya satu pasien telah selesai diobati.
3. Menentukan simpul-E, yaitu simpul hidup dengan ongkos terbesar
4. Mematikan semua simpul hidup dengan ongkos lebih kecil daripada simpul-E

Mengulangi tahap 1 s/d 4 hingga mencapai solusi permainan

Algoritma B&B memiliki kompleksitas waktu lebih baik dibandingkan dengan algoritma runut-balik, BFS murni, maunpun DFS murni, karena adanya fungsi pembatas, yang akan dijelaskan pada bagian berikutnya. Berdasarkan penyelidikan *Optimization Technology Center*, algoritma *Branch and Bound* merupakan algoritma yang paling banyak diimplementasikan di dalam perangkat lunak. Algoritma B&B telah terbukti cukup tepat untuk menyelesaikan masalah praktis.

2.4. Metode yang Digunakan

2.4.1. OOP (*Object Oriented Programming*)

Pemrograman berorientasi objek merupakan pemrograman yang menjadikan objek sebagai komponen utama dalam sistem. Pemrograman berorientasi objek merupakan satu ide luar biasa dalam bidang pemrograman yang setelah diterapkan ternyata melahirkan bahasa yang lebih maju seperti C++, Java, Phyton, dsb. Perkembangan dari OOP inipun melahirkan konsep pemrograman yang lain berbasis Windows atau GUI *(Graphical User Intarface)* yang kemudian popular sebai bahasa visual, seperti Visual Basic, Delphi, Visual C, Visual Java, dsb. [8]

Jadi untuk menyelesaikan suatu masalah dengan kolaborasi antar objek-objek yang ada karena setiap objek memiliki deskripsi tugasnya sendiri Pemrograman orientasi-objek menekankan konsep berikut:

1. **Kelas (Class)** — kumpulan atas definisi data dan fungsi-fungsi dalam suatu unit untuk suatu tujuan tertentu. Sebagai contoh 'class of plants' adalah suatu unit yang terdiri atas definisi-definisi data dan fungsi-fungsi yang menunjuk pada berbagai macam perilaku/turunan dari tanaman. Sebuah class adalah dasar dari modularitas dan struktur dalam pemrograman berorientasi objek. Sebuah class secara tipikal sebaiknya dapat dikenali oleh seorang non-programmer sekalipun terkait dengan domain permasalahan yang ada, dan kode yang terdapat dalam sebuah class sebaiknya (relatif) bersifat mandiri dan independen (sebagaimana kode tersebut digunakan jika tidak menggunakan OOP). Dengan modularitas, struktur dari sebuah program akan terkait dengan aspek-aspek dalam masalah yang akan diselesaikan melalui program tersebut. Cara seperti ini akan menyederhanakan pemetaan dari masalah ke sebuah program ataupun sebaliknya.
2. **Objek (Object)** - membungkus data dan fungsi bersama menjadi suatu unit dalam sebuah program komputer. Objek merupakan dasar dari modularitas dan struktur dalam sebuah program komputer berorientasi objek
3. **Abstraksi (Abstract)** - Kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti. Setiap objek dalam sistem melayani sebagai model dari "pelaku" abstrak yang dapat melakukan kerja, laporan dan perubahan keadaannya, dan berkomunikasi dengan objek lainnya dalam sistem, tanpa mengungkapkan bagaimana kelebihan ini diterapkan. Proses, fungsi atau metode dapat juga dibuat abstrak, dan beberapa teknik digunakan untuk mengembangkan sebuah pengabstrakan
4. **Enkapsulasi (Encapsulation)** - Memastikan pengguna sebuah objek tidak dapat mengganti keadaan dalam dari sebuah objek dengan cara yang tidak layak hanya metode dalam objek tersebut yang diberi ijin

untuk mengakses keadaannya. Setiap objek mengakses interface yang menyebutkan bagaimana objek lainnya dapat berinteraksi dengannya. Objek lainnya tidak akan mengetahui dan tergantung kepada representasi dalam objek tersebut

5. **Polimorfisme (Polimorfism)** melalui pengiriman pesan. Tidak bergantung kepada pemanggilan subrutin, bahasa orientasi objek dapat mengirim pesan; metode tertentu yang berhubungan dengan sebuah pengiriman pesan tergantung kepada objek tertentu di mana pesa tersebut dikirim. Contohnya, bila sebuah burung menerima pesan "gerak cepat", dia akan menggerakan sayapnya dan terbang. Bila seekor singa menerima pesan yang sama, dia akan menggerakkan kakinya dan berlari. Keduanya menjawab sebuah pesan yang sama, namun yang sesuai dengan kemampuan hewan tersebut. Ini disebut polimorfisme karena sebuah variabel tungal dalam program dapat memegang berbagai jenis objek yang berbeda selagi program berjalan, dan teks program yang sama dapat memanggil beberapa metode yang berbeda di saat yang berbeda dalam pemanggilan yang sama. Hal ini berlawanan dengan bahasa fungsional yang mencapai polimorfisme melalui penggunaan fungsi kelas-pertama
6. **Inheritas (Inheritance)**- Mengatur polimorfisme dan enkapsulasi dengan mengijinkan objek didefinisikan dan diciptakan dengan jenis khusus dari objek yang sudah ada objek-objek ini dapat membagi (dan memperluas) perilaku mereka tanpa haru mengimplementasi ulang perilaku tersebut (bahasa berbasis-objek tidak selalu memiliki inheritas)

2.4.2. UML (Unified Modeling Language)

UML adalah bahasa untuk mendokumentasikan, memodelkan, dan membangun sistem perangkat lunak dalam bentuk diagram visual. Berikut ini adalah jenis-jenis diagram yang umum digunakan :

1. *Use Case Diagram*

   Mendeskripsikan apa yang sistem lakukan tanpa mendeskripsikan bagaimana sistem menyelesaikannya. Dibuat berdasarkan interaksi dan relasi dari individual use case yang berisi aktor, event, dan use case.

2. *Use Case Scenario*

   Merupakan hasil penjelasan dari setiap use case. Terbagi menjadi tiga bagian, yaitu: identifikasi dan inisiasi, step perfomed, kondisi akhir.

3. *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 antar 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.

4. *Activity* Diagram

   Menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses yang dipakai untuk memperlihatkan urutan aktifitas proses bisnis. Struktur diagram ini mirip flowchart atau Data Flow Diagram pada perancangan terstruktur.

5. *Class* Diagram

   Class Diagram menggambarkan keadan suatu sistem dengan menjelaskan keterhubungan antara suatu *class* dengan *class* yang lain yang terdapat pada sistem tersebut. *Class Diagram* bersifat statis, di dalam *class diagram* digambarkan relasi dari masing - masing *class*

tetapi tidak menggambarkan apa yang terjadi ketika *class* tersebut berelasi.

2.5. Tools yang Digunakan

2.5.1. Java

Java adalah sebuah bahasa pemrograman, Java dapat membuat seluruh bentuk aplikasi, desktop, web dan lainnya, sebagaimana dibuat dengan menggunakan bahasa pemrograman konvensional yang lain. Java adalah bahasa pemrograman yang berorientasi objek (OOP) dan dapat dijalankan pada berbagai platform sistem operasi. Perkembangan Java tidak hanya terfokus pada satu sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat open source.java sudah memiliki library database yang disebut JDBC (Java Database Connectivity).

Java Platform adalah sebuah platform software yang berbeda dari berbagai platform lainnya seperti Microsoft Windows, Macintosh, OS / 2, UNIX dan NetWare. Java dirancang untuk menghasilkan dan berjalan sangat interaktif, dinamis, dan applet yang aman dan aplikasi-aplikasi pada sistem jaringan komputer. Aplikasi yang ditulis dalam bahasa Java untuk mengkompilasi arsitektur netral JVM (Java Virtual Machine) *bytecodes*, yang tidak sekedar kode mesin normal. Keuntungan utamanya adalah agar memungkinkan untuk menjalankan aplikasi Java pada sistem apapun, selama masih menerapkan sistem JVM. Java Platform ini terdiri dari dua bagian utama, yaitu Java Virtual Machine (JVM) dan Java API yang akan dibahas sebagai berikut :

1. Java Virtual Machine (JVM)

   JVM adalah suatu mesin virtual yang menginterpretasikan *bytecode* java ke dalam bahasa mesin. *Bytecode* java adalah adalah file hasil kompilasi kode java ekstension filenya adalah.class. Jika sutu program java bernama ContohJava, maka java dikompilasi maka

hasilnya adalah: ContohJava.class inilah yang disebut bytecode java. Paradigma pemrogramman dengan cara menginterpretasikan bytecode menurut para ahli kurang bagus dari sisi kecepatan, akan tetapi JVM mencoba mengatasi masalah ini dengan menerapkan teknik *Just In Time* (JIT) compilation yaitu java bytecode langsung dikompilasi menjadi bahasa mesin untuk kode-kode program yang dijalankan secara berulang-ulang.

JVM ada dalam paket yang disebut JRE (Java Run Time Environtment), yaitu paket lingkungan yang dibutuhkan ketika menjalankan aplikasi java. JRE inilah yang menterjemahkan Byte code hasil kompilasi, menjadi native pada platform yang akan menjalankan aplikasi. JRE inilah bersifat spesifik platform sehingga terdapat : jre untuk linux, jre untuk windows dan sebagainya. Salah satu komponen penting dari JRE adalah JVM (Java Virtual Machine).

2. Application Programming Interface (API).

   API adalah sekumpulan libraries komponen software kelas-kelas dan interface-interface java siap pakai yang memiliki berbagai kegunaan dan kemampuan yang berbeda-beda. Sekumpulan kelas-kelas dan interface-interface yang saling berkaitan diorganisasikan dalam suatu library. Library ini dikenal dengan sebutan paket. Paket inilah untuk menambahkan fungsionalitas yang sudah dibuat dan dapat dirubah, sehingga menghemat waktu coding.

Berikut adalah kumpulan library dasar di java :

a. Java.lang

   Java.lang adalah paket yang berisi kelas-kelas elemen dasar untuk melakukan pemrogramman. Jika menggunakan kelas-kelas ini dalam kode maka tidak perlu melakukan import lagi.

b. Java.io

Java.io adalah paket yang berisi kelas-kelas untuk melakukan manipulasi masukan dan keluaran termasuk penanganan masalah file.

c. Java.util

Java.util adalah paket yang berisi kelas-kelas pelengkap seperti kelas struktur data dan kelas penanggalan.

d. Java.net

Java.net adalah paket yang berisi kelas-kelas yang memungkinkan komputer saling berinteraksi dengan komputer yang lainnya melalui jaringan TCP/IP.

e. Java.awt

Java.awt adalah paket yang berisi kelas-kelas yang memungkin programmer untuk membuat aplikasi dengan antarmuka grafis, Graphical User Interface (GUI).

f. Java.awt.peer

Java.awt.peer merupakan paket yang berisi keseluruhan definisi interfece. Setiap antarmuka java.awt.peer berkaitan dengan salah satu atau java.awt komponen menu komponen kelas. Interface dalam paket ini menentukan metode yang harus didukung oleh komponen GUI pada *platform* tertentu.

g. Java.applet

Java.applet adalah paket yang berisi kelas-kelas yang memungkin programmer untuk membuat antarmuka yang interaktif di web. Semua program yang ditulis dalam java pasti memanfaatkan API Java ini.

2.5.2. NetBeans

NetBeans IDE adalah sebuah lingkungan pengembangan atau sebuah alat untuk pemrogram menulis, mengompilasi, mencari kesalahan dan menyebarkan program. Netbeans IDE ditulis dalam Java dan dapat mendukung bahasa pemrograman lain seperti : JavaScript, PHP, Python, Ruby, Groovy, C, C++, Scala, Clojure, dan lain-lain. NetBeans juga mengacu pada kedua *platform* kerangka untuk aplikasi desktop Java, dan sebuah lingkungan pengembangan terpadu untuk pengembangan dengan Java.

NetBeans IDE dapat digunakan jika JVM diinstal, termasuk Windows, Mac OS, Linux, dan Solaris. Sebuah JDK (Java Development Kit) diperlukan untuk pengembangan fungsionalitas Java, tetapi tidak diperlukan untuk pembangunan di bahasa pemrograman lain. Komponen-komponen dapat dipakai ulang sehingga mendukung RAD (Rapid Application development).

1. Jendela utama

   Disebut jendela utama karena merupakan pusat perintah pada IDE. Disini kitatemukan menu bar, beberapa toolbar, componen palette dan tab woskspace.

2. Workspaces

   Sebuah workspaces adalah kumpulan-kumpulan dari jendela-jendela yang salingberhubungan erat untuk melakukan tugas-tugas tertentu. Jenis-jenis workspaces default, yaitu:

   a. Editting

      Editting terdiri dari jendela Explorer dan Propertis. Saat membuka sebuah file, source edittor secara otomatis terbuka. Source Edittor digunakan untuk mengeditfile-file java, HTML dan Plaint Text.

   b. GUI Editting

      Workspaces GUI Editing digunakan untuk merancang dan membangun GUI. Didalamnya terdapat Explorer, Form Editor, Component Inspector.

1) Form Editor

   Jendela Form Editor merupakan daerah utama untuk membuat dan memodifikasi sebuah GUI

2) Component Inspector

   Component Inspector memungkinkan kita untuk melihat komponenkomponen apa saja, baik yang tampak maupun yang tidak tampak. Disini kita mengubah properti dari komponen-komponen tersebut.

c. Browsing

   Didalamnya terdapat Object Browser yang terdiri atas tiga panel dan jendela Properties.

d. Running

   Didalamnya terdapat jendela Execution View dan Output. Saat mengeksekusiprogram, maka IDE akan secara otomatis ke Running Workspaces.

e. Debugging

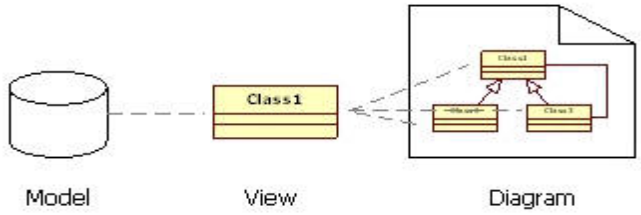
   Debugging terdiri dari jendela-jendela Debugger dan Output. Jendela-jendela ini mempunyai tab panel untuk menset breakpoint, memonitor thread dan mengawasi nilai variabel-variabel. Jendela output menampilkan pesan dari Debugger.

2.5.3. Star UML

StarUML adalah software permodelan yang mendukung UML (Unified Modeling Language). Berdasarkan pada UML version 1.4 dan dilengkapi 11 macam diagram yang berbeda, mendukung notasi UML 2.0 dan juga mendukung pendekatan MDA (Model Driven Architecture) dengan dukungan konsep UML. StarUML dapat memaksimalkan pruduktivitas dan kualitas dari suatu softwareproject, konsep Dasar dari StarUml:

1. Model, View and Diagram

   StarUML membuat perbedaan konseptual yang lebih jelas antara models, views and diagrams. Model adalah elemen yang memuat informasi untuk model software. View adalah suatu ekpresi visual dari informasi di dalam model dan Diagram adalah suatu koleksi dari elemen yang memberikan pemikiran user didalam mendesaian secara spesifik.



Gambar 2.1 Gambar Model, View dan Diagram

2. Project dan Unit

   a. Project

   Project adalah unit manajemen dasar didalam StarUML. Suatu project dapat mengatur satu atau lebih model software. Project merupakan top-level package yang selalu ada didalam beberapa model software. Secara umum, satu project disimpan dalam satu file.

   b. File Project

   File project disimpan kedalam format XML dengan extension ".UML" . Semua model, views dan diagrams yang dibuat dengan StarUML disimpan dalam satu file project.

   c. Units

   Ada beberapa kasus dimana satu project perlu disimpan didalam beberapa file file kecilsehingga para pengembang dapat bekerja didalam satu project secara bersamaan. Didalamkasus ini suatu project dapat mengatur bermacam – macam unit. Suatu unit

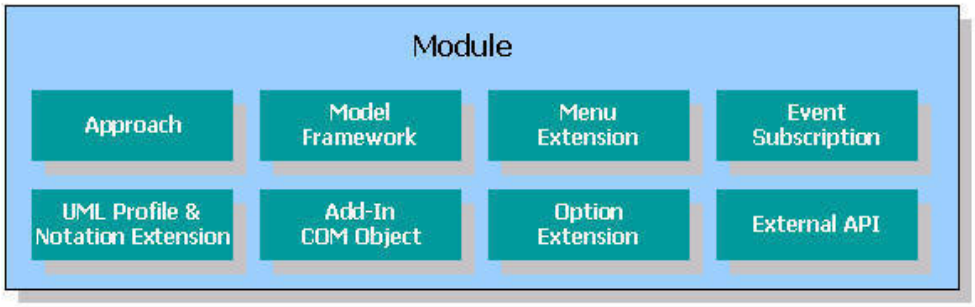
mempunyai3struktur hirarki dan berisikan beberapa sub-unit. Unit disimpan sebagai ".UML " file dan beberapa mengacu pada file project (.UML) atau unit file lainnya (.UNT).

d. Komposisi Unit

Hanya package, subsystem dan elemen model yang dapat membentuk satu unit. Semuaelemen dibawah jenis elemen package ini disimpan didalam masing – masing file unit (.UNT).

3. *Module*

Modul adalah suatu package yang menyediakan fungsi – fungsi baru dan feature sebagaiperluasan dari StarUML. Modul dapat dibuat sebagai kombinasi dari beberapa elemen –elemen extension dan juga membuat beberapa jenis elemen – elemen didalam suatu modul.



Gambar 2.2 Gambar Module

2.5.4. Kuesioner

Kuesioner adalah salah satu teknik pengumpulan data yang dilakukan dengan cara memberi seperangkat pertanyaan atau pernyataan tertulis kepada responden untuk dijawab. Kuesioner cocok digunakan bila jumlah responden cukup besar dan tersebar di wilayah yang luas.[11] Kuesioner dapat berupa pertanyaan/pernyataan tertutup atau terbuka. Terdapat beberapa prinsip dalam penulisan kuesioner yaitu: isi dan tujuan pertanyaan, bahasa yang digunakan, tipe dan bentuk pertanyaan, pertanyaan tidak mendua, tidak

menanyakan yang sudah lupa, pertanyaan tidak menggiring, panjang pertanyaan, urutan pertanyaan, prinsip pengukuran, penampilan fisik kuesioner. Supaya kuesioner menghasilkan data kuantitatif yang akurat maka digunakan skala pengukuran, skala pengukuran adalah kesepakatan yang digunakan sebagai acuan untuk menentukan panjang pendeknya interval yang ada dalam alat ukur sehingga alat ukur tersebut menghasilkan data kuantitatif.[11]

Skala untuk mengolah kuesioner adalah skala likert, skala likert digunakan untuk mengukur sikap, pendapat dan persepsi seseorang tentang fenomena sosial. Fenomena sosial ini telah ditetapkan secara spesifik oleh peneliti, dengan skala likert, maka variabel yang akan diukur dijabarkan menjadi kriteria variabel. Kriteria tersebut dijadikan sebagai titik tolak untuk menyusun item-item instrumen yang dapat berupa pernyataan atau pertanyaan.

Jawaban setiap instrumen yang menggunakan skala likert memiliki gradasi dari positif sampai sangat negatif, yang dapat berupa kata-kata sebagai berikut:

1. Sangat Setuju
2. Setuju
3. Cukup Setuju
4. Kurang Setuju
5. Tidak Setuju

Jawaban tersebut diberikan skor untuk keperluan analisis kuantitatif, yaitu:

1. Sangat Setuju diberi 5
2. Setuju diberi 4
3. Cukup Setuju diberi 3
4. Kurang Setuju diberi 2
5. Tidak Setuju diberi 1

Instrumen penelitian yang menggunakan skala likert dapat dibuat dalam bentuk *checklist* ataupun pilihan ganda. contoh bentuk pilihan ganda

Berilah salah satu jawaban terhadap pertanyaan berikut sesuai dengan pendapat anda, dengan cara memberi tanda lingkaran pada nomor jawaban yang tersedia.

prosedur kerja yang baru itu akan segera diterapkan dilembaga anda?

1. Sangat Setuju
2. Setuju
3. Cukup Setuju
4. Kurang Setuju
5. Tidak Setuju

Bentuk jawaban pilihan ganda dapat diletakan pada tempat yang berbeda-beda, untuk jawaban di atas "tidak setuju" diletakan pada jawaban nomor pertama, untuk item selanjutnya jawaban "tidak setuju" dapat diletakan pada jawaban nomor terakhir. Penyusunan instrumen untuk variabel tertentu, sebaiknya butir-butir pertanyaan dibuat dalam bentuk kalimat positif, netral atau negatif, sehingga responden dapat menjawab dengan serius dan kosisten contoh:

1. saya mencintai mobil diesel karena hemat bahan bakar (positif)
2. mobil diesel banyak diproduksi di jepang (netral)
3. mobil diesel sulit dihidupkan di tempat dingin (negatif)

Cara ini membuat kecenderungan responden dalam menjawab pada kolom tertentu dapat dikurangi dan responden akan selalu membaca pertanyaan setiap item instrumen dan juga jawabannya. Pada benuk cheklist, sering jawaban tidak dibaca karena letak jawaban sudah menentu, tetapi keuntungan dalam hal ini yaitu singkat dalam pembuatannya, hemat kertas, mudah mentabulasikan data, dan secara visual lebih menarik. Data yang diperoleh dari skala tersebut adalah data interval.

SS = Sangat Setuju diberi 5

ST = Setuju diberi 4

CS = Cukup Setuju diberi 3

KS = Kurang Setuju diberi 2

TS = Tidak Setuju diberi 1

Kemudian dengan teknik pengumpulan data kuesioner, misalnya instrumen tersebut diberikan kepada 100 responden yang diambil secara random, dari 100 responden setelah dilakukan analisis di dapatkan:

25 orang menjawab SS

40 orang menjawab ST

5 orang menjawab CS

20 orang menjawab KS

10 orang menjawab TS

Berdasarkan data tersebut 65 orang (40+25) atau 65% responden menjawab setuju dan sangat setuju. Jadi kesimpulannya mayoritas responden setuju dengan metode kerja baru.

Data interval tesebut juga dapat dianalisis dengan menghitung rata-rata jawaban berdasarkan skoring setiap jawab dari responden. Berdasarkan skor yang telah ditetapkan dapat dihitung dan dilihat pada Tabel 2.1

Tabel 2.1 Perhitungan Skoring Skala Likert

Jumlah skor untuk 25 orang yang menjawab SS =	25x5 =	125
Jumlah skor untuk 40 orang yang menjawab ST =	40x4 =	160
Jumlah skor untuk 5 orang yang menjawab CS =	5x3 =	15
Jumlah skor untuk 20 orang yang menjawab KS =	20x2 =	40
Jumlah skor untuk 10 orang yang menjawab TS =	10x1 =	10
jumlah total =		350

$$P = \frac{S}{skorideal} x100\% \qquad (2.2)$$

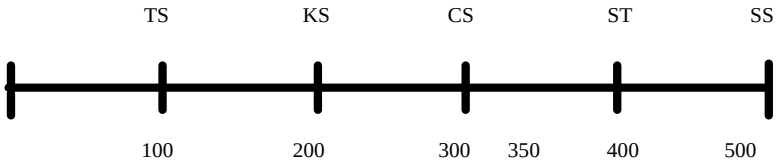
Keterangan :

P = Nilai persentase yang dicari

S = Jumlah frekuensi dikalikan dengan nilai yang di tetapkan jawaban

Skorideal = Nilai tertinggi dikalikan dengan jumlah sampel

Jumlah skor ideal (kriterium) untuk seluruh item = 5x100 = 500 (seandainya semua menjawab SS). Jumlah skor yang diperoleh dari penelitian = 350. Jadiberdasarkan data itu maka tingkat persetujuan terhadap metode kerja baru itu =(350:500)x100% = 70% dari yang diharapkan (100%). Secara kontinum dapat dilihat pada gambar 2.3



Gambar 2.3 Skala Kontinum

Jadi berdasarkan data yang diperoleh dari 100 responden makar rata-rata 350 terletak pada daerah setuju.