BAB II

TINJAUAN PUSTAKA

II.1 Plagiarisme

II.1.1 Pengertian Plagiarisme

Plagiarisme adalah tindakan penyalahgunaan, pencurian /perampasan, penerbitan, pernyataan, atau menyatakan sebagai milik sendiri sebuah pikiran, ide, tulisan, atau ciptaan yang sebenarnya milik orang lain. (Ridhatillah, 2006:64).

Plagiat dapat dianggap sebagai tindak pidana karena mencuri hak cipta orang lain. Di dunia pendidikan, pelaku plagiarisme akan mendapat hukuman berat seperti dikeluarkan dari sekolah / universitas. Pelaku plagiat disebut sebagai plagiator. Sistem pendeteksi plagiarisme dapat dikembangkan untuk :

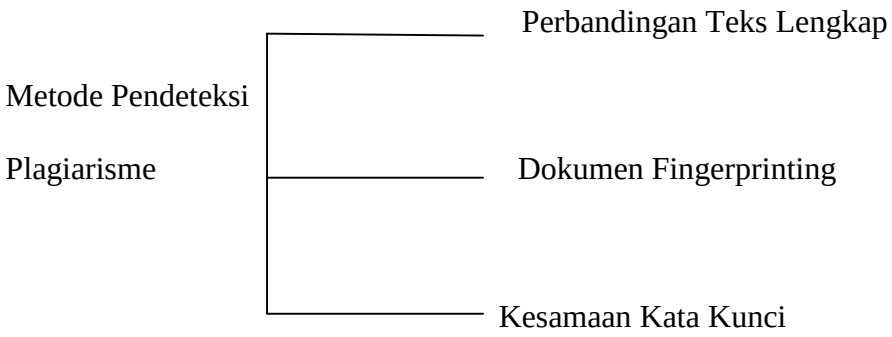
1. Data teks seperti *essay*, artikel, jurnal, penelitian dan sebagainya.
2. Dokumen teks yang lebih terstruktur seperti bahasa pemrograman

Beberapa tipe plagiarisme yaitu :

1. *Word-for-word plagiaris*m adalah menyalin setiap kata secara langsung tanpa diubah sedikitpun.
2. *Plagiarism of authorship* adalah mengakui hasil karya orang lain sebagai hasil karya sendiri dengan cara mencantumkan nama sendiri menggantikan nama pengarang yang sebenarnya.
3. *Plagiarism of ideas* adalah mengakui hasil pemikiran atau ide orang lain. *Plagiarism of sources*, jika seorang penulis menggunakan kutipan dari penulis lainnya tanpa mencantumkan sumbernya. (Parvatti, 2006 :57)

II.1.2 Metode Pendeteksi Plagiarisme

Metode pendeteksi plagiarisme dibagi menjadi tiga bagian yaitu metode perbandingan teks lengkap, Metode dokumen *fingerprinting*, dan metode kesamaan kata kunci. Metode pendeteksi plagiarism dapat dilihat pada gambar II.1 : (Stein, 2006: 15)



Gambar II.1 : Metode Pendeteksi Plagiarisme

(Sumber : Stein, 2006 : 16**)**

Berikut ini penjelasan dari masing-masing metode dan algoritma pendeteksi plagiarisme :

1. Perbandingan Teks Lengkap.

   Metode ini diterapkan dengan membandingkan semua isi dokumen. Dapat diterapkan untuk dokumen yang besar. Pendekatan ini membutuhkan waktu yang lama tetapi cukup efektif, karena kumpulan dokumen yang diperbandingkan adalah dokumen yang disimpan pada penyimpanan lokal. Metode perbandingan teks lengkap tidak dapat diterapkan untuk kumpulan dokumen yang tidak terdapat pada dokumen lokal. Algoritma yang

digunakan pada metode ini adalah algoritma *Brute Force*, algoritma *edit distance*, algoritma *Boyer Moore* dan algoritma *lavenshtein distance*

2. Dokumen *Fingerprinting*. Dokumen *fingerprinting*

merupakan metode yang digunakan untuk mendeteksi keakuratan salinan antar dokumen, baik semua teks yang terdapat di dalam dokumen atau hanya sebagian teks saja. Prinsip kerja dari metode dokumen *fingerprinting* ini adalah dengan menggunakan teknik *hashing*. Teknik *hashing* adalah sebuah fungsi yang mengkonversi setiap string menjadi bilangan. Misalnya Rabin-Karp, Winnowing dan Manber

3. Kesamaan Kata Kunci.

Prinsip dari metode ini adalah mengekstrak kata kuncidari dokumen dan kemudiandibandingkan dengan kata kunci pada dokumen yang lain. Pendekatan yang digunakan pada metode ini adalah teknik dot.

(**Sumber** : Stein, 2006 : 16)

II.2. Teks Mining

II.2.1 Pengertian Teks *Mining*

Data *mining* adalah suatu istilah yang digunakan untuk menguraikan penemuan pengetahuan di dalam database. Data *minning* adalah proses yang menggunakan teknik statistic, matematika, kecerdasan buatan, dan *machine learning* untuk mengekstraksi dan mengindetifikasi informasi yang bermanfaat dan pengetahuan yang terakit dari berbagai database besar (Kusrini. 2009 : 3).

Kemajuan luar biasa yang terus berlanjut dalam bidang data *mining* didorong oleh beberapa factor, antara lain :

1. Pertumbuhan yang sangat cepat dalam pengumpulan data.
2. Penyimpanan data dalam *data warehouse*, sehingga seluruh perusahaan memiliki akses kedalam database yang handal.
3. Adanya peningkatan akses data melalui navigasi web dan internet.
4. Tekanan kompetisi bisnis untuk meningkatan penguasaan pasar dalam globalisasi ekonomi.
5. Perkembangan teknologi perangkat lunak untuk data *mining*
6. Perkembangan yang hebat dalam kemampuan komputasi dan pengembangan kapasitas media penyimpanan (Kusrini, 2009 : 5)

Teks *mining*, yang juga disebut sebagai Teks *Data Mining* (TDM) atau *Knowledge Discovery in Text* (KDT), secara umum mengacu pada proses ekstraksi informasi dari dokumen-dokumen teks tak terstruktur (*unstructured*). Teks *mining* dapat didefinisikan sebagai penemuan informasi baru dan tidak diketahui sebelumnya oleh komputer, dengan secara otomatis mengekstrak informasi dari sumber-sumber teks tak terstruktur yang berbeda. Kunci dari proses ini adalah menggabungkan informasi yang berhasil diekstraksi dari berbagai sumber (Tan, 2007: 17).

Tujuan utama teks *mining* adalah mendukung proses *knowledge discovery* pada koleksi dokumen yang besar. Pada prinsipnya, teks *mining* adalah bidang ilmu multidisipliner, melibatkan *information retrieval* (IR), *text analysis*, *information extraction* (IE), *clustering*, *categorization*, *visualization*, *database*

technology, *natural language processing* (NLP), *machine learning*, dan *data mining*. Dapat pula dikatakan bahwa teks *mining* merupakan salah satu bentuk aplikasi kecerdasan buatan *(artificial intelligence* / AI).

Teks *mining* mencoba memecahkan masalah *information overload* dengan menggunakan teknik-teknik dari bidang ilmu yang terkait. Teks *mining* dapat dipandang sebagai suatu perluasan dari *data mining* atau *knowledge-discovery in database* (KDD), yang mencoba untuk menemukan pola-pola menarik dari basis data berskala besar. Namun teks *mining* memiliki potensi komersil yang lebih tinggi dibandingkan dengan *data mining*, karena kebanyakan format alami dari penyimpanan informasi adalah berupa teks. Teks *mining* menggunakan informasi teks tak terstruktur dan mengujinya dalam upaya mengungkap struktur dan arti yang tersembunyi di dalam teks.

Perbedaan mendasar antara teks *mining* dan *data mining* terletak pada sumber data yang digunakan. Pada *data mining*, pola-pola diekstrak dari basis data yang terstruktur, sedangkan di teks *mining*, pola-pola diekstrak dari data tekstual (*natural language*). Secara umum, basis data didesain untuk program dengan tujuan melakukan pemrosesan secara otomatis, sedangkan teks ditulis untuk dibaca langsung oleh manusia (Hearst, 2006:23).

II.2.2 Ruang Lingkup Teks *mining*

Teks *mining* merupakan suatu proses yang melibatkan beberapa area teknologi. Namun secara umum proses-proses pada teks *mining* mengadopsi proses *data mining*. Bahkan beberapa teknik dalam proses teks *mining* juga

menggunakan teknik-teknik *data mining*. Ada empat tahap proses pokok dalam teks *mining*, yaitu pemrosesan awal terhadap teks (*text preprocessing*), transformasi teks (*text transformation*), pemilihan fitur (*feature selection*), dan penemuan pola (*pattern discovery*). (Even-Zohar, 2006:45).

a. Text Preprocessing

Tahap ini melakukan analisis semantik (kebenaran arti) dan sintaktik (kebenaran susunan) terhadap teks. Tujuan dari pemrosesan awal adalah untuk mempersiapkan teks menjadi data yang akan mengalami pengolahan lebih lanjut. Operasi yang dapat dilakukan pada tahap ini meliputi *part-of-speech* (PoS) *tagging*, menghasilkan *parse tree* untuk tiap-tiap kalimat, dan pembersihan teks.

b. Text Transformation

Transformasi teks atau pembentukan atribut mengacu padaproses untuk mendapatkan representasi dokumen yang diharapkan. Pendekatan representasi dokumen yang lazim digunakan adalah model *bag of words* dan model ruang vector (*vector space model*). Transformasi teks sekaligus juga melakukan pengubahan kata-kata ke bentuk dasarnya dan pengurangan dimensi kata di dalam dokumen. Tindakan ini diwujudkan dengan menerapkan *stemming* dan menghapus *stopwords*.

c. *Feature Selection*

Pemilihan fitur (kata) merupakan tahap lanjut dari pengurangan dimensi pada proses transformasi teks. Walaupun tahap sebelumnya sudah

melakukan penghapusan kata-kata yang tidak deskriptif (*stopwords*), namun tidak semua kata-kata di dalam dokumen memiliki arti penting. Oleh karena itu, untuk mengurangi dimensi, pemilihan hanya dilakukan terhadap katakata yang relevan yang benar-benar merepresentasikan isi dari suatu dokumen. Ide dasar dari pemilihan fitur adalah menghapus kata-kata yang kemunculannya di suatu dokumen terlalu sedikit atau terlalu banyak.

Algoritma yang digunakan pada teks *mining*, biasanya tidak hanya melakukan perhitungan pada dokumen saja, tetapi juga pada *feature*. Empat macam *feature* yang sering digunakan:

1. *Character*, merupakan komponan individual, bias huruf, angka, karakter spesial dan spasi, merupakan *block* pembangun pada level paling tinggi pembentuk semantik *feature*, seperti kata, *term* dan *concept*. Pada umumnya, representasi *character-based* ini jarang digunakan pada beberapa teknik pemrosesan teks.
2. *Words*.
3. *Terms* merupakan *single word* dan *multiword phrase* yang terpilih secara langsung dari *corpus*. Representasi *term-based* dari dokumen tersusun dari *subset term* dalam dokumen.
4. *Concept*, merupakan *feature* yang di-*generate* dari sebuah dokumen secara manual, *rule-based*, atau metodologi lain.(Triawati, 2009: 23)

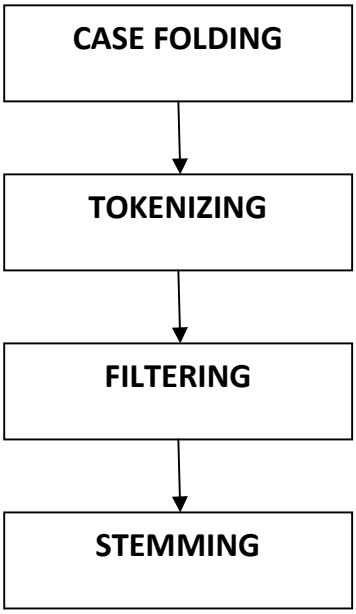
d. *Pattern Discovery*

Pattern discovery merupakan tahap penting untuk menemukan pola atau pengetahuan (*knowledge*) dari keseluruhan teks. Tindakan yang lazim dilakukan pada tahap ini adalah operasi teks *mining*, dan biasanya menggunakan teknik-teknik *data mining*. Dalam penemuan pola ini, proses teks *mining* dikombinasikan dengan proses-proses *data mining*. Masukan awal dari proses teks *mining* adalah suatu data teks dan menghasilkan keluaran berupa pola sebagai hasil interpretasi atau evaluasi. Apabila hasil keluaran dari penemuan pola belum sesuai untuk aplikasi, dilanjutkan evaluasi dengan melakukan iterasi ke satu atau beberapa tahap sebelumnya. Sebaliknya, hasil interpretasi merupakan tahap akhir dari proses teks *mining* dan akan disajikan ke pengguna dalam bentuk visual. (Even-Zohar, 2008: 56)

II.2.3 Ekstraksi Dokumen

Teks yang akan dilakukan proses teks *mining*, pada umumnya memiliki beberapa karakteristik diantaranya adalah memiliki dimensi yang tinggi, terdapat *noise* pada data, dan terdapat struktur teks yang tidak baik. Cara yang digunakan dalam mempelajari suatu data teks, adalah dengan terlebih dahulu menentukan fitur-fitur yang mewakili setiap kata untuk setiap fitur yang ada pada dokumen. Sebelum menentukan fitur-fitur yang mewakili, diperlukan tahap *preprocessing* yang dilakukan secara umum dalam teks *mining* pada dokumen, yaitu *case*

folding, tokenizing, filtering, stemming, tagging dan *analyzing*. Gambar II.2 adalah tahap dari *preprocessing:* (Triawati, 2009: 56)

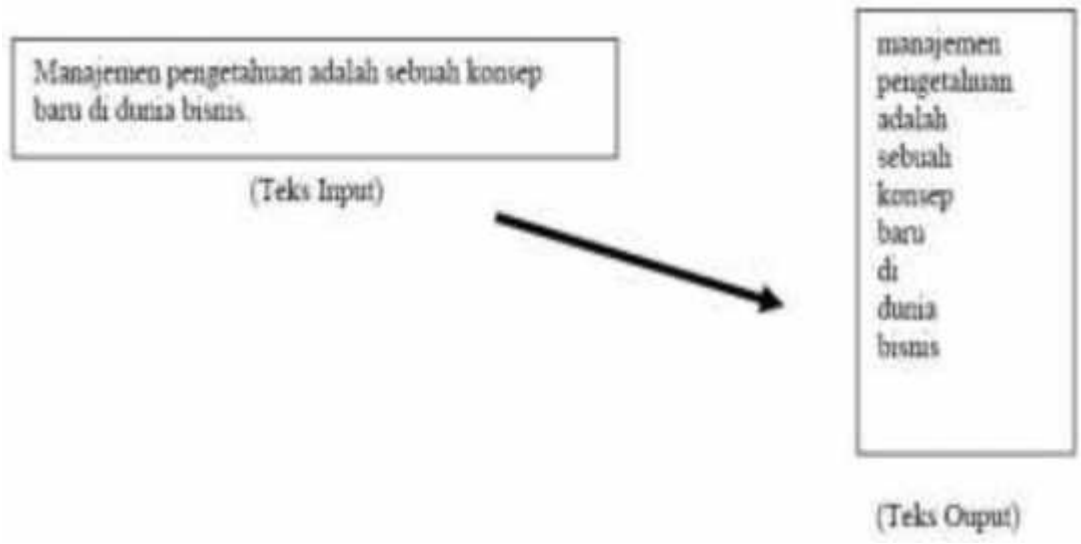


Gambar 2.2 Tahap *preprocessing*

Sumber **:** Triawati, 2009: 56

II.2.3.1 *Case folding* dan *Tokenizing*

Case folding adalah mengubah semua huruf dalam dokumen menjadi huruf kecil. Hanya huruf ‗a‘ sampai dengan ‗z‘ yang diterima. Karakter selain huruf dihilangkan dan dianggap *delimiter*. Tahap *tokenizing / parsing* adalah tahap pemotongan string input berdasarkan tiap kata yang menyusunnya. Gambar II.3 adalah proses *tokenizing*: (Triawati, 2009:59).

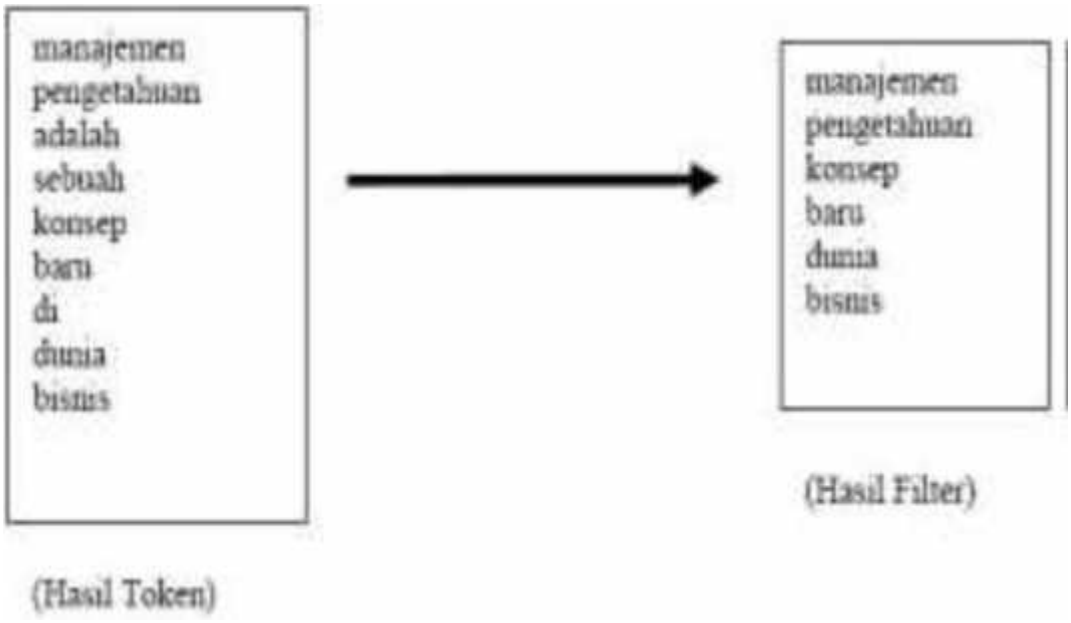


Gambar II.3 *Tokenizing*

Sumber: Triawati, 2009 :61

II.2.3.2 *Filtering*

Filtering adalah tahap mengambil kata-kata penting dari hasil token. Bisa menggunakan algoritma *stoplist* (membuang kata yang kurang penting) atau *wordlist* (menyimpan kata penting). *Stoplist / stopword* adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan *bag-of-words*. Contoh *stopwords* adalah —yang, —dan, —di, —dari dan seterusnya. Contoh dari tahapan ini dapat dilihat pada gambar II.4: (Triawati, 2009:61)

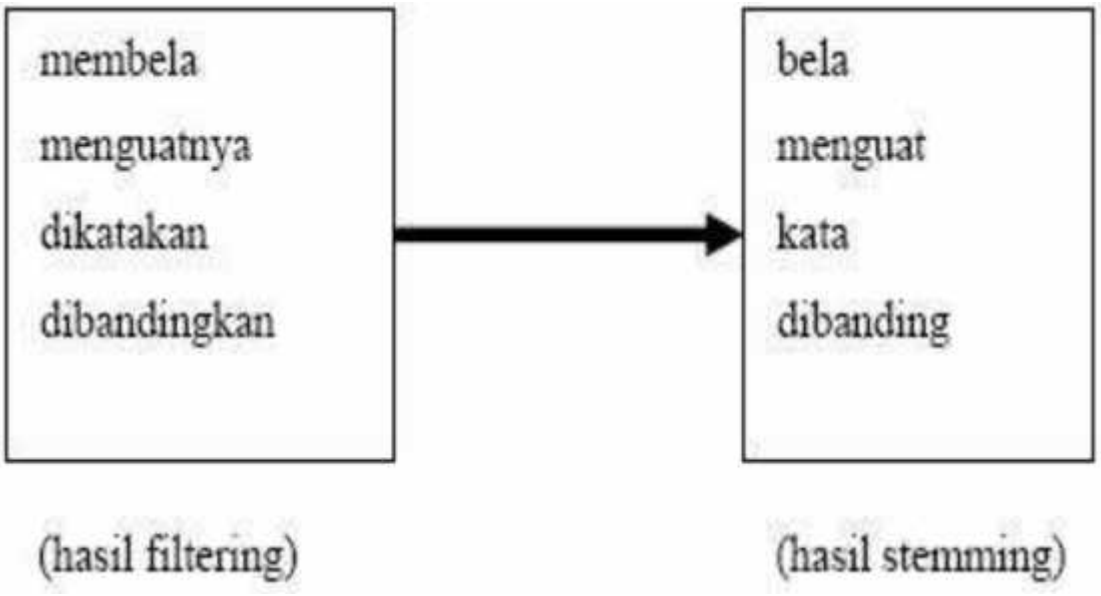


Gambar II.4 *Filtering*

Sumber : Triawati, 2009:61

II.2.3.3 *Stemming*

Tahap *stemming* adalah tahap mencari *root* kata dari tiap kata hasil *filtering.* Pada tahap ini dilakukan proses pengembalian berbagai bentukan kata ke dalam suatu representasi yang sama. Tahap ini kebanyakan dipakai untuk teks berbahasa nggris dan lebih sulit diterapkan pada teks berbahasa Indonesia. Hal ini dikarenakan bahasa Indonesia tidak memiliki rumus bentuk baku yang permanen. Contoh dari tahapan ini pada teks adalah seperti gambar II.5: (Triawati, 2009:62)



Gambar II.5 *Stemming*

Sumber : Triawati, 2009:62

II.3 Sekilas Sejarah Java

Java adalah bahasa pemrograman yang disusun oleh James Gosling yang dibantu oleh rekan – rekannya seperti Patrick Naugton, Chris Warth, Ed Frank, dan Mike Sheridan di suatu peruasahaan perangkat lunak yang bernama *Sun Microsystems.* Pada tahun 1991. Bahasa pemrograman ini mula – mula diinisialisasi dengan nama “Oak”, namun pada tahun 1995 diganti namanya menjadi ”java”. (Budi Rahardjo. 2010 : 1-2)

Alasan utama pembentukan bahasa java adalah untuk membuat aplikasi – aplikasi yang dapat diletakkan diberbagai macam perangkat elekronik, seperti *microwave oven* dan *remote control*, sehingga java harus bersifat *portable* atau yang sering disebut dengan *platform independent* (tidak tergantung pada *platform).* Itulah yang menyebabkan dalam dunia pemrograman java, dikenal

adanya istilah "*write once"*, "*run everywhere"*, yang berarti kode program hanya ditulis sekali, namun dapat dijalankan dibawah *flatform* manapun, tanpa harus melakukan perubahan kode program. (Budi Rahardjo. 2010 : 2).

Oleh karena itu adanya keharusan sebuah pemrograman yang kecil, menghasilkan kode yang kecil pula dan harus platform independen (tidak terikat pada platform) membuat tim pada proyek tersebut terinspirasi ole hide pemrograman yang sama yang telah ditemukan oleh Niklaus Wirth, penemu pascal. Jadi penwemu pascal memiliki pemikiran tentang sebuah software bahasa pemrograman portable dan tidak tergantung pada sebuah platform atau mesin. bahas pemrograman komersial yang disebut UCSD pascal tersebut menghasilkan kode intermediate yang diperuntukkan bagi sebuah mesin virtual. Jadi, kode asli dari bahasa pemrograman tersebut tidak tergantung pada mesin ataupun platform system operasi karena USCB pascal menghasilkan intermediate code yang selanjutnya akan dikopilasi atau diterjemahkan oleh mesin virtual ke kode mesin dimana kode tersebut dijalankan. (Sumber: Sulistiani, Sri. *Membangun GUI dengan JAVA Netbean 6.5* : 2010:2).

Pada tahun – tahun tersebut sangat pesat. Namun saat itu browser juga masih jarang ditemui. Pada tahun 1994 kebanyakan orang menggunakan mosaic, yaitu sebuah *browser* nonkomersil yang dibuat oleh Marc Anderseen pada tahun 1993 disupercomputing center universitas Illinois. Pada pertengahan tahun 1994 para pengembang java menyadari bahwa mereka dapat saja membangun sebuah *browser* yang lebih *flexible* dari pada yang lainnya. Selanjutnya dibuatlah HotJava *Browser* yang dikerjakan oleh Patrick Naughton dan Jonanthan Payne. Tujuan

utama dari pembuatan *browser* tersebut tidak lain adalah untuk mempromosikan bahasa java dan memamerkan kekuatannya. Java juga memiiliki kekuatan pada aplikasi yang disebut applet yang disebut juga berhubungan dengan *browser*. (Sumber: Sulistiani, Sri. *Membangun GUI dengan Java Netbean 6.5* : 2010: 2)

II.3.1 Java Language Spesification, API, JDK dan IDE

Java language specification adalah teknis dari pemprograman java yang didalamnya terdapat aturan penulisan sintaks dan semantic java. Referensi lengkap tentang *Java language specification* dapat dilihat pada website resmi java, yaitu http://java.sun.com/docs/books/jl.

API adalah *Aplication Programming Interface* yaitu sebuah layer yang berisi class – class yang sudah didefinisikan dan antarmuka pemrograman yang akan membantu para pengembangan aplikasi dalam perancangan sebuah aplikasi. Pada saat ini dikenal ada tiga macam API dari java, yaitu :

a. J2SE, yaitu java 2 Standard Edition adalah sebuah API yang digunakan untuk mengembangkan aplikasi – aplikasi yang bersifat *client – side standalone* atau *applet.*
b. J2EE, yaitu java 2 Enterprise Edition adalah API yang digunakan untuk melakukan pengembangan aplikasi – aplikasi yang bersifat server – side seperti Java Servlet, dan Java Server Pages.
c. J3ME, yaitu java 2 Micro Edition adalah API yang merupakan subnet dari J2SE tetapi memiliki kegunaan untuk mengembangkan aplikasi pada

handheld device seperti smart phone atau PDA tentu saja yang didalamnya telah ditanamkan interpreter java. (Sumber : Sulistiani, Sri. *Membangun GUI dengan JAVA Netbean 6.5* : 2010 :4)

IDE (*Integreted Development Environment),* yaitu sebuah lingkungan pengembangan aplikasi lengkap dan dapat membantu proses pengembangan sebuah aplikasi menjadi lebih cepat.

II.4 Bagan Alir (Flowchart)

Dalam mendeskripsikan algoritma, seringkali dijumpai kesulitan jika hanya menggunakan kata – kata atau kalimat- kalimat saja. Sehingga deskripsinya cenderung sulit dipahami dan memungkinkan timbulnya kesalahan interpretasi bagi orang lain. Selain itu, diperllukan banyak keterangan – keterangan yang sering kali justru sulit dimengerti. Belum lagi, setiap orang tentu memiliki cara – cara tersendiri yang berbeda sehingga tidak pernah ditemukan keserangaman dalam format penulisannya.

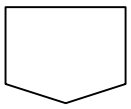
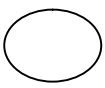
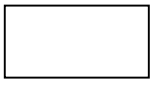
Untuk alasan tersebut, maka penggunaan *flowchart* akan banyak membantu dan menguntungkan. Dengan *flowchart,* langkah prosedur penyelesaian permasalahan dapat diekspresikan dengan serangkaian symbol grafis yang baku dan lebih mudah digunakan. Penggunaan *flowchart* akan mennghindarkan sejak dini timbulnya kesalahan interpretasi bagi orang lain yang merupakan awal kegagalan suatu prosedur yang dikembangkan.

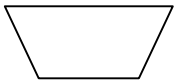
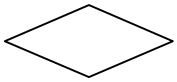
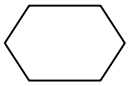
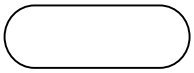
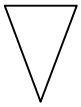
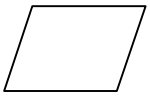
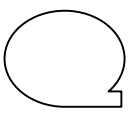
Banyak manfaat akan diperoleh apabila kita sering menggunakan *flowchart* dalam mengembangkan prosedur pemecahan masalah komputasi.

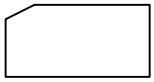
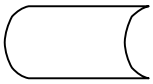
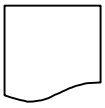
Pertama, kita akan terbiasa berfikir secara sistematis dan terstruktur dalam setiap kesempatan. Kedua, dengan menggunakan *flowchart*, kita akan lebih mudah mengecek dan menemukan bagian – bagian prosedur yang tiidak valid dan bertele – tele. Selain itu, prosedur yang dikembangkan akan lebih mudah dipahami oleh orang lain, sehingga tidak menimbulkan kesalahan interpretasi apabila mau menerapkan prosedur yang kita kembangkan.

Flowchart dapat diartikan sebagai suatu alat atau sarana yang menunjukkan langkah – langkah yang harus dilaksanakan dalam menyelesaikan suatu permasalahan untuk komputasi dengan cara mengekspresikan kedalam serangkaian symbol – symbol grafis khusus. Beberapa symbol yang sering digunakan dalam *flowchart* adalah seperti ditunjukkan pada Gambar II.6.

(Sumber : Edhy Sutanta, 2005: 99)

No	Simbol	Arti	Keterangan
1		*Symbol Off-line Connector*	Simbol untuk keluar/masuk *prosedure* atau proses dalam lembar/halaman yang lain.
2		*Symbol Connector*	Simbol untuk keluar/masuk prosedur atau proses dalam lembar/halaman yang sama.
3		*Symbol Process*	Simbol yang menunjukkan pengolahan yang dilakukan oleh komputer

4		*Symbol Manual Operation*	Simbol yang menunjukkan pengolahan yang tidak dilakukan oleh komputer.
5		*Symbol Decision*	Simbol untuk kondisi yang akan menghasilkan beberapa kemungkinan jawaban/aksi.
6		*Symbol Predefined Process*	Simbol untuk mempersiapkan penyimpanan yang akan digunakan sebagai tempat pengolahan di dalam *storage*
7		*Symbol Terminal*	Simbol untuk permulaan atau akhir dari suatu program
8		*Symbol Off-line Storage*	Simbol yang menunjukkan bahwa data di dalam symbol ini akan disimpan
9		*Data Input Reader Operation*	Simbol operasi dengan membaca data input dari sistem atau bagian lain
10		*Symbol magnetic-tape unit*	Simbol yang menyatakan input berasal pita magnetic atau output disimpan ke pita

			magnetic
11		*Symbol punched card*	Simbol yang menyatakan input berasal dari kartu atau output ditulis ke kartu
12		*Symbol disk and on-line storage*	Simbol untuk menyatakan input berasal dari disk atau output disimpan ke *disk*
13		*Symbol display*	Simbol yang menyatakan peralatan output yang digunakan yaitu layar, *plotter*, printer, speaker dan sebagainya
14		*Symbol transmittal tape*	Simbol untuk menyatakan input berasal dari mesin jumlah/hitung
15		*Symbol document*	Simbol yang menyatakan input berasal dari dokumen dalam bentuk kertas atau output dicetak ke kertas

Tabel II.1. Simbol – Simbol *Flowchart*

(**Sumber** : Edhy Sutanta, 2005: 99)