BAB II
LANDASAN TEORI

II.1 *Web Mining*

Menurut B. Liu [6], *Web mining* adalah suatu teknik yang digunakan untuk menemukan informasi yang berguna atau pengetahuan dari *web hyperlink structure*, *page content*, dan *usage data*. Sedangkan menurut O. Maimon dan L. Rokach [7] *web mining* adalah teknik yang bertujuan menemukan dan mengekstrak informasi relavan yang tersembunyi di *web* yang berhubungan dengan data, khususnya pada data dokumen *hypertext* yang dipublikasikan ke *web*.

Berdasarkan penjelasan di atas, dapat disimpulkan bahwa *web mining* adalah sebuah cara atau metode yang digunakan untuk menemukan atau menggali informasi yang bermanfaat pada *web*. Di dalam *web* terdapat tiga tipe data, yaitu *data on the web* (*content*), *web log data* (*usage*) dan *web structure data*. Dari tiga tipe data tersebut maka *web mining* dapat dikelompokkan menjadi tiga jenis yaitu *Web Content Mining*, *Web Structure Mining*, dan *Web Usage Mining*.

II.1.1 *Web Content Mining* (WCM)

Metode *Web content mining* (WCM) yang akan digunakan dalam tugas akhir ini merupakan metode yang berguna untuk mencari informasi yang berguna yang bersumber dari halaman *web*. Informasi tersebut bisa berupa teks, gambar, suara, video, *metadata* dan *hyperlinks*. Beberapa informasi tersebut didapatkan dari sumber data yang bersifat semi-terstruktur seperti dokumen HTML, atau yang lebih terstruktur seperti data pada tabel atau *database* yang diambil dari halaman HTML. Tetapi kebanyakan dari data tersebut berbentuk data teks yang tidak terstruktur. Ada dua strategi yang umum digunakan pada WCM, pertama langsung melakukan *mining* terhadap data dan kedua melakukan pencarian serta meng*improve* pencarian seperti *search engine* atau *information retrieval*.

WCM dapat dibedakan menjadi dua sudut pandang, yaitu pendekatan berbasis agen dan pendekatan dengan *database*. Pendekatan pertama bertujuan

untuk meningkatkan pencarian dan penyaringan informasi dan bisa ditempatkan ke dalam tiga kategori:

1. *Intelligent Search Agents*

   Search agent atau agen pencari ini untuk mencari informasi yang relavan menggunakan karakteristik domain dan profil pengguna untuk mengatur dan manafsirkan informasi yang ditemukannya.

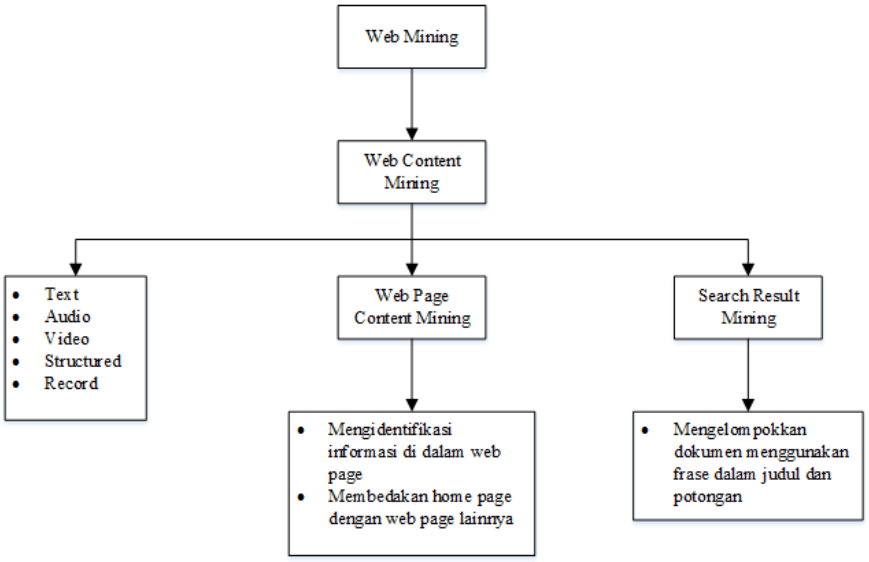
2. *Information Filtering / Categorization Agent*

   Agen ini menggunakan metode *Information Retrieval* (IR) dan karakteristik *hypertext* dokumen *web* untuk memperoleh, menyaring, dan mengelompokkannya secara otomatis.

3. *Personalized Web Agents*

   Agen ini digunakan untuk menemukan informasi berdasarkan ketertarikan dari pengguna.

Pendekatan kedua bertujuan untuk memodelkan data pada *web* ke dalam bentuk terstuktur dengan menerapkan mekanisme *query database* dan aplikasi *data mining* untuk analisisnya.



Gambar II.1 Arsitektur WCM

II.1.1.1 Teknik *Web Content Mining*

Fokus utama dari skripsi ini adalah menerapkan metode WCM. Konsep WCM melibatkan beberapa teknik untuk *summarizing* (meringkas), *classification*

(klasifikasi) dan *clustering* (mengelompokkan) dalam konten web. Berikut adalah penjelasan teknik-teknik WCM untuk tipe data teks [8]:

1. *Unstructured Text Data Mining Techniques*

   Data *web content* sebagian besar merupakan data teks yang tidak terstruktur. Teknik-teknik yang digunakan untuk data teks yang tidak terstruktur yang digunakan dalam WCM dapat dilihat pada tabel berikut :

Tabel II.1 Teknik *Unstructured Text Data*

Teknik	Input	Metode	Output
Topic Tracking	Database	Topik *filtering* dari *database*, Topik *re-ranking*	Topik-topik yang berhubungan
Summariza-tion	Dokumen	Metode *Extractive* dan *Abstractive*	Ringkasan yang menyampaikan isi sebuah dokumen
Categoriza-tion	Dokumen	Menghitung jumlah kata dari tema atau topik dokumen dan dilakukan *ranking*	Dokumen yang berhubungan berdasarkan *ranking*
Clustering	Koleksi Dokumen	Mengelompok-kan dokumen ke kelompok/ klaster yang terbaik	Kelompok/ klaster yang relavan terbaik yang berisi dokumen yang diinginkan

2. *Structured Text Data Mining Techniques*

   Data teks tersruktur biasanya merupakan data *record* yang diperoleh dari *database* dan ditampilkan di *web pages*. Data *record* tersebut dapat ditampilkan ke dalam bentuk tabel atau *form*. Teknik-teknik yang digunakan untuk data teks terstruktur yang digunakan dalam WCM dapat dilihat pada tabel berikut :

Tabel II.2 Teknik *Structured Text Data*

Teknik	Metode	Digunakan	Contoh
Web Crawler	*Crawling* atau menelusuri *hyperlink*, Membuat indeks ke *database*	*Search Engine*, *Textual Analysis*, *Access Market*, *Trend* topik	Google, Yahoo
Wrapper	Menghubungkan ke bermacam *search engine*	Meta *Search Engine*	Visual Web Ripper, iMacros, Screen Scrapper

II.1.2 *Web Structure Mining* (WSM)

WSM adalah metode yang digunakan untuk menemukan pengetahuan yang berguna dari *hyperlinks* yang memperlihatkan struktur dari suatu *web*. Kebanyakan dari *tools* yang digunakan untuk mencari informasi di *web* hanya menggunakan informasi yang berbentuk teks dan mengabaikan *link* dari informasi yang bisa menjadi sangat bernilai. Tujuan dari WSM adalah menghasilkan ringkasan yang terstruktur tentang *website* dan *webpage*. Secara teknik, WCM hanya menfokuskan pada struktur dari dokumen, sedangkan WSM mencoba untuk menemukan struktur *link* dari *hyperlinks* yang terdapat pada dokumen. Berdasarkan pada topologi dari *hyperlinks*, WSM akan mengklasifikasikan *webpages* dan mengambil informasi, seperti persamaan dan hubungan antara *website* yang berlainan.

II.1.3 *Web Usage Mining* (WUM)

Web usage mining (WUM) merupakan metode untuk menemukan informasi yang bermanfaat dari data sekunder yang di dapat dari interaksi pengguna yang melakukan *browsing* di *web*. Metode ini menfokuskan pada bagaimana mengetahui perilaku dari pengguna internet saat mereka berinteraksi dengan *web*. Tujuan dari metode ini antara lain untuk mengetahui perilaku pengguna terhadap suatu *website*, membandingkan antara harapan dan kenyataan dari pengguna suatu *website* dan menyesuaikan isi *website* berdasarkan keinginan dari pengguna.

II.2 *Text Mining*

Menurut R. Feldman dan J. Sanger [9], *Text mining* adalah sebagai suatu proses menggali informasi dimana seorang pengguna berinteraksi dengan sekumpulan dokumen menggunakan *tools* analisis. Sedangkan menurut O. Maimon dan L. Rokach [7], *Text mining* adalah suatu proses ekstraksi pola tertentu dari *database* dokumen teks yang besar yang bertujuan untuk menemukan pengetahuan.

Berdasarkan penjelasan di atas, dapat disimpulkan bahwa *Text mining* adalah sebuah cara atau metode untuk menemukan atau menggali informasi dari kumpulan dokumen teks yang besar. *Text mining* memberikan solusi dari permasalahan seperti pemrosesan, pengorganisasian atau pengelompokkan dan menganalisa *unstructured text* dalam jumlah besar.

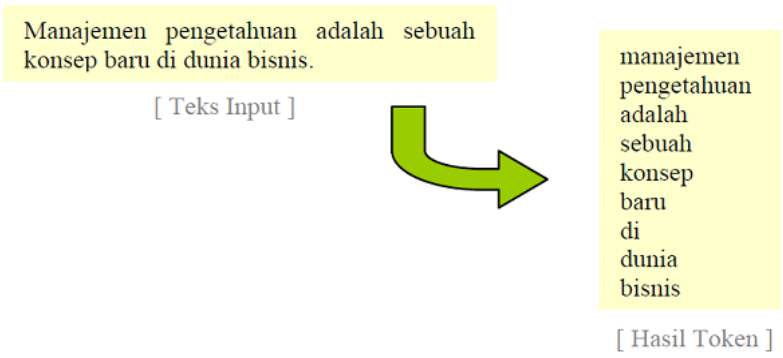
Tujuan utama dari *text mining* adalah untuk mendukung proses *knowledge discovery* pada koleksi dokumen yang besar. Sumber data yang digunakan pada *text mining* adalah sekumpulan teks yang memiliki format yang tidak terstruktur atau minimal semi-terstruktur. Adapun kegiatan yang dilakukan dari *text mining* antara lain pengkategorisasian teks (*text categorization*) dan pengelompokkan teks (*text clustering*). *Text mining* mengembangkan banyak teknik untuk beberapa bidang lain, seperti *Data Mining*, *Information Retrieval*, Statistik dan Matematika, *Machine Learning*, *Linguistic*, *Natural Languange Processing* (NLP), dan *Visualization*.

II.2.1 Tahap *Text Mining*

Dalam *text mining* memiliki lima tahapan dalam pemrosesan data teks tetapi pada tugas akhir ini akan hanya menggunakan 4 tahap yang antara lain :

1. Tahap *Tokenizing*

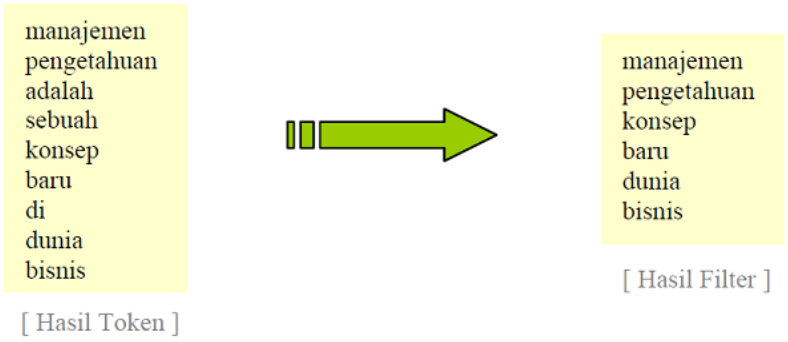
   Merupakan tahap pemotongan *string input* berdasarkan tiap kata yang menyusunnya. Contoh dari tahap ini adalah sebagai berikut



Gambar II.2 Contoh Tahap *Tokenizing*

2. Tahap *Filtering*

   Filtering adalah tahap mengambil kata-kata penting dari hasil token. Tahap ini biasanya juga disebut tahap *stopword removal*. Contoh dari tahap ini adalah sebagai berikut :



Gambar II.3 Contoh Tahap *Filtering*

3. Tahap *Stemming*

   Stemming adalah tahap mentransformasi kata-kata hasil *filtering* ke kata-kata akarnya (*root word*) atau kata dasar dengan menggunakan aturan-aturan tertentu. Proses *stemming* pada teks berbahasa Indonesia berbeda dengan *stemming* pada teks berbahasa Inggris. Pada teks berbahasa Inggris, proses yang diperlukan hanya proses menghilangkan sufiks. Sedangkan pada teks berbahasa Indonesia selain sufiks, prefiks dan konfiks juga dihilangkan.

Contoh penggunaan *stemming* pada teks berbahasa Indonesia, kata bersama, kebersamaan, menyamai, jika dkenakan proses *stemming* ke bentuk kata dasarnya yaitu “sama”.

4. Tahap *Analyzing*

   Tahap ini merupakan tahap penentuan seberapa jauh keterkaitan antar kata-kata pada dokumen/inputan yang ada. Pada tahap *anlyzing* akan digunakan rumus TF-IDF untuk mengambil sebuah informasi dari sebuah dokumen. Kata-kata yang umum dalam sebuah dokumen cenderung memiliki nilai tinggi dalam perhitungan TF-IDF.

Sebelum melakukan tahap *text mining* ini¸ ada tahap *parser* atau tahap pemisahan tag *script* dari *webpage* berhubung datanya diambil langsung dari *website*.

II.3 *Clustering* Dokumen

Penelitian tentang *clustering document* telah banyak dilakukan. Tujuan *clustering* dokumen adalah untuk memisahkan dokumen-dokumen yang berelavan dengan yang memiliki kemiripan satu sama lain dari pada dokumen yang tidak berelavan, sehingga dapat dikelompokkan ke dalam suatu klaster atau kelompok. Pada algoritma *clustering*, dokumen akan dikelompokkan menjadi klaster-klaster berdasarkan kemiripan satu data dengan yang lain. Prinsip dari *clustering* adalah memaksimalkan kesamaan antar anggota satu klaster dan meminimumkan kesamaan antar anggota klaster yang berbeda.

Ada banyak jenis teknik-teknik untuk *clustering* yang bisa digunakan untuk dokumen yang berasal dari *web* :

1. *Text Based Clustering*.
2. *Partitional Clustering*.
3. *Hierarchical Clustering*.
4. *Graph Based Clustering*.
5. *Neural Network Base Clustering*.
6. *Fuzzy Clustering*.
7. *Probabilistic Clustering*.

II.4 *K-Means*

K-Means merupakan salah satu teknik atau metode *partitional clustering* yang melakukan partisi data yang ada ke dalam bentuk satu atau lebih klaster atau kelompok. Teknik ini mempartisi data ke dalam klaster sehingga data yang memiliki karakteristik yang sama akan dikelompokkan ke dalam satu klaster dan data yang mempunyai karakteristik yang berbeda dikelompokkan ke dalam kelompok yang lain.

Dari beberapa teknik *clustering* yang paling sederhana dan umum dikenal adalah *clustering k-means*. Dalam teknik ini, akan dikelompokkan obyek ke dalam *k* kelompok atau klaster. Untuk melakukan *clustering* ini nilai *k* harus ditentukan dahulu. Biasanya pengguna telah mempunyai informasi awal mengenai obyek yang dipelajarinya, termasuk berapa jumlah klaster yang paling tepat. Adapun algoritma dari *k-means* yang dengan sebagai berikut :

1. Tentukan k sebagai jumlah *cluster* yang ingin dibentuk.
2. Pilih secara acak vektor dokumen yang akan dijadikan k *centroid*.
3. Cari *centroid* yang paling dekat dari setiap dokumen.
4. Hitung ulang untuk menentukan *centroid* baru dari dokumen-dokumen yang terletak pada *centroid* yang sama.
5. Lakukan langkah 3 hingga tidak ada lagi tempat yang akan ditandai sebagai *cluster* baru.

Berikut ini merupakan contoh implementasi dari algoritma K-Means ke dalam bentuk *pseudocode* [10].

```
Input: E = {e1,e2,...,en} (set entitas menjadi cluster)
       k (jumlah cluster)
       MaxIters (batas iterasi)
Output: C = {c1,c2,...,ck} (set centroid cluster}
        L = {l(e)|e = 1,2,...,n} (set label cluster untuk
             setiap E)
foreach ci ∈ C do
      ci ← ej ∈ E ; (random data)
end
foreach ei ∈ E do
      l(ei) ← argminDistance(ei,cj)j ∈ {1...k};
end

changed ← false;
iter ← 0;
repeat
      foreach ci ∈ C do
            UpdateCluster(ci);
      end
      foreach ei ∈ E do
            minDist ← argminDistance(ei,cj)j ∈ {1...k};
            if minDist ≠ l(ei) then
                  l(ei) ← minDist;
                  changed ← true;
            end
      end
      iter++;
until changed = true and iter ≤ Maxiters;
```

II.5 *Web Crawler*

Web crawler adalah sebuah sistem yang menjelajahi struktur *hyperlink* dari *web*, dari sebuah alamat awal atau *seed* dan mengunjungi alamat *web* di dalam *webpage* secara berkala. *Web crawler* juga dikenal dengan *web robot* atau *spider*. *Search engine* yang merupakan salah satu contoh sistem besar yang menggunakan *crawler* untuk melintasi internet secara terus-menerus dengan tujuan menemukan dan mengambil *webpage* sebanyak mungkin. Selain digunakan untuk *search engine*, *web crawler* juga digunakan untuk penelusuran khusus atau tertentu, seperti penelusuran alamat *email*.

Banyaknya sistem untuk *web crawler*, pada prinsipnya secara fungsional semuanya sama. Berikut ini proses yang dilakukan *web crawler* [11] :

1. Mengunduh *webpage*.
2. Mem-*parsing webpage* yang diunduh dan mengambil semua *link*.
3. Untuk setiap *link* yang diambil, ulangi proses.

Dalam langkah pertama, sebuah *web crawler* mengambil URL dan mengunduh halaman dari internet berdasarkan URL yang diberikan. Biasanya halaman yang diunduh disimpan ke sebuah file atau ditempatkan di *database*. Dengan menyimpan *webpage*, maka *crawler* atau sistem yang lain dapat memanipulasi *webpage* tersebut untuk diindeks (dalam kasus *search engine*) atau untuk pengarsipan yang digunakan oleh pengarsip otomatis.

Pada langkah kedua, *web crawler* mem-*parsing* keseluruhan *webpage* yang diunduh dan mengambil *link-link* ke halaman lain. Pada penelitian ini, *web crawler* selain mengambil *link-link* juga akan mengambil konten informasi pada *webpage*. Tiap *link* dalam halaman didefinisikan dengan sebuah penanda HTML yang dapat dilihat seperti berikut :

```
<A HREF=http://www.detik.com/directory/file.html>Link</A>
```

Setelah *crawler* mengambil *link* dari *webpage*, tiap *link* ditambahkan ke sebuah daftar untuk di-*crawl*.

Pada langkah ketiga, *web crawler* akan mengulangi proses. Semua *crawler* bekerja dengan rekursif atau bentuk perulangan yang dilakukan dengan menerapkan algoritma *web crawler*.

II.5.1 Algoritma *Web Crawler*

Algoritma *web crawler* memiliki dua jenis yang umum digunakan, yaitu *Breadth-First* dan *Depth-First*. Prinsip dari dua jenis algoritma *web crawler* tersebut sama dengan algoritma pencarian, yaitu *Breadth-First Search* (BFS) dan *Depth-First Search* (DFS). Konsep dari algoritma *Breadth-First* [11] adalah dengan mengecek setiap *link* dalam suatu *webpage* sebelum berpindah ke *webpage* yang lain. *Breadth-First* akan menelusuri setiap *link* pada *webpage* pertama, lalu menelusuri setiap *link* pada *webpage* dari *link* pertama di *webpage* pertama, dan seterusnya. Penelusuran ini dilakukan sampai tidak ada lagi *link* baru yang dapat ditelusuri atau jika jumlah *webpage* yang ditelusuri sudah mencapai batas maksimal yang ditentukan. Sedangkan algoritma *Depth-First* kebalikan dari

Breadth-First, dimana *Depth-First* akan menelusuri semua kemungkinan jalur dari suatu *link* sampai mencapai suatu dasar sebelum melanjutkan ke *link* berikutnya.

Berdasarkan penjelasan di atas, dapat diketahui bahwa *Depth First* akan bekerja dengan cara mengunjungi dari satu cabang sampai mencapai dasar dari cabang tersebut. Cara ini tidak bagus jika tujuan akhirnya berada pada cabang yang lain karena tidak ada jaminan dasar dari satu cabang tersebut yang memiliki level dasar tidak terhingga. Selain itu juga, algoritma ini menggunakan prinsip LIFO (*Last In First Out*) di mana dia akan mengunjungi level yang paling terakhir terlebih dulu. Sedangkan *Breadth First* yang menerapkan prinsip FIFO (*First In First Out*) akan lebih berhati-hati dengan memeriksa semua *link* di setiap cabang, meski membutuhkan waktu yang lama. Berdasarkan perbandingan tersebut, maka pada tugas akhir ini akan menggunakan algoritma *Breadth First* karena *web crawler* ini hanya digunakan sebagai pengambilan *link* dari situs berita yang bisa ditentukan berdasarkan kedalaman levelnya.

II.5.1.1 *Breadth First Crawler*

Seperti yang dijelaskan pada sub bab II.5.1 tentang algoritma web crawler bahwa algoritma *Breadth First Crawler* sama dengan *Breadth First Search*. Adapun langkah-langkah dari *Breadth First* dengan sebagai berikut:

1. Memasukkan URL yang akan dijadikan *seed* atau akar.
2. Untuk setiap *link* yang diambil dari URL *seed* dimasukkan ke dalam antrian atau *queue*.
3. Selama *queue* tidak kosong, hapus *link* pertama dari *queue* lalu melakukan *parsing* dari *link* pertama yang dihapus, dan masukkan ke dalam antrian.

Berikut ini merupakan prosedur dari algoritma *Breadth First* ke dalam bentuk *pseudocode* [12].

```
Breadth-First (starting_urls) {
      foreach link (starting_urls) {
            enqueue(frontier, link);
      }
      while (visited < MAX_PAGES) {
            link := dequeue_link(frontier);
            page := fetch(link);
            enqueue(frontier, extract_links(page));
            if (frontier > MAX_BUFFER) {
                  dequeue_last_links(frontier);
            }
      }
}
```

II.5.2 *Robot Protocol* [13]

Proses *crawling* secara terus-menerus dapat menyebabkan beban berlebih pada server suatu *web*. Beberapa halaman yang diunduh pada suatu waktu dari *webpage* tidak beratus-ratus atau ribuan secara berurutan. Selain itu juga, beberapa *website* terdapat bagian *web* yang diharap tidak dimasuki ke dalam proses *crawling*. Untuk mengatasi masalah ini, maka dibuatlah sebuah protokol yang disebut *Robot Protocol*. *Robot protocol* ini menspesifikasikan bahwa *website* membatasi area tertentu atau halaman dari *crawling* yang disimpan dalam sebuah *file* berformat teks yang diberi nama robots.txt. Dalam *file* itu juga dapat dispesifikasikan *web crawler* apa saja yang dapat menelusuri *web server* tersebut.

II.6 *Tools*

Pada sub bab ini akan menjelaskan *tools* yang dibutuhkan untuk memodelkan, membangun perangkat lunak, dan pengujian perangkat lunaknya dalam penelitian ini.

II.6.1 PHP [14]

PHP atau PHP *Hypertext Preprocessor* merupakan suatu bahasa pemrograman yang digunakan untuk membangun suatu *website* yang dinamis. PHP menyatu dengan kode HTML. HTML digunakan sebagai pembangun atau pondasi dari kerangka *layout web*, sedangkan PHP digunakan sebagai prosesnya. PHP berjalan pada sisi *server* sehingga PHP disebut sebagai bahasa *Server Side Scripting*, yang berarti bahwa dalam setiap / untuk menjalankan PHP wajib

adanya *web server*. PHP ini bersifat *open source* sehingga dapat digunakan oleh semua *programmer* dan mampu digunakan di semua platform atau sistem operasi, seperti Windows dan Linux. PHP juga dibangun sebagai modul pada *web server* Apache dan sebagai *binary* yang dapa berjalan sebagai CGI.

II.6.1.1 Berbagai Macam Versi PHP

Saat ini PHP telah mencapai versi 6. Pada sub bab ini akan diuraikan versi PHP pertama hingga saat ini.

1. PHP/FI

   PHP pertama kali diciptakan oleh Rasmus Lerdof yang sebelumnya bernama FI. PHP atau FI ini sudah digunakan untuk mengolah *form web* yang kemudian versi pertama ini mulai banyak dikembangkan sejak dirilis kepada secara umum.

2. PHP Versi 2.0

   Pada versi ini, PHP sudah diintegrasi dengan bahasa pemrograman lain seperti C dan dilengkapi dengan modul sehingga kualitas PHP juga sudah meningkat.

3. PHP Versi 3.0

   Pada versi ini, dilakukan perubahan dengan nama PHP (*PHP* : *Hypertext Preprocessor*).

4. PHP Versi 4.0

   Versi ini paling banyak digunakan pada awal abad 21 karena sudah mampu membangun *web* secara kompleks juga disertai dengan stabilitas kecepatan yang tinggi.

5. PHP Versi 5.0

   Pada versi ini telah terjadi perubahan yang sangat besar, di mana PHP kini telah berubah menjadi sesuatu dengan pendekatan objek. Versi ini dikembangkan oleh perusahaan bernama Zend. Selain dengan perubahan tersebut, perusahaan tersebut melahirkan *framework-framework* PHPyang ada sekarang.

6. PHP Versi 6.0

Versi ini PHP sudah mendukung Unicode, perbaikan keamanan, memiliki fitur dan konstruksi yang lebih baru.

II.6.2 *Unified Modelling Language* (UML) [15]

Unified Modeling Language (UML) adalah bahasa spesifikasi standar untuk mendokumentasikan, menspesifikasikan, dan membangun sistem perangkat lunak. UML adalah metodologi untuk mengembangkan sistem OOP dan sekelompok perangkat *tool* untuk mendukung pengembangan sistem tersebut. UML juga sebagai dasar bagi perangkat desain berorientasi objek dari IBM.

UML dikembangkan sebagai suatu alat untuk analisis dan desain berorientasi objek oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson. Namun demikian UML dapat digunakan untuk memahami dan mendokumentasikan setiap sistem informasi. Penggunaan UML dalam industri terus meningkat. Ini merupakan standar terbuka yang menjadikannya sebagai bahasa pemodelan yang umum dalam industri peranti lunak dan pengembangan sistem.

II.6.2.1 Diagram UML

Berikut akan dijelaskan empat macam diagram yang akan digunakan dalam penelitian skripsi ini, yaitu *use case* diagram, *activity* diagram, *class* diagram dan *sequence* diagram.

1. *Use Case* Diagram

Use Case diagram digunakan untuk memodelkan bisnis proses berdasarkan perspektif pengguna sistem. *Use Case* diagram terdiri atas diagram untuk *use case* dan *actor*. *Actor* mempresentasikan orang yang akan mengoperasikan atau orang yang berinterkasi dengan sistem aplikasi. *Use Case* mempresentasikan operasi-operasi yang dilakukan oleh aktor. *Use Case* digambarkan berbentuk elips dengan nama operasi dituliskan di dalamnya. *Actor* yang melakukan operasi dihubungkan dengan garis lurus ke *use case*.

2. *Activity* Diagram

Activity diagram merupakan diagram yang menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Diagram ini memiliki struktur diagram yang mirip dengan *flowchart* atau data *flow* diagram pada perancangan

terstruktur. Manfaat dari diagram ini yaitu dapat membantu dalam memahami proses secara keseluruhan. *Activity* diagram dibuat berdasarkan sebuah atau beberapa *use case* pada *use case* diagram.

3. *Class* Diagram

Class diagram merupakan diagram yang selalu ada di permodelan sistem berorientasi objek. *Class* diagram menunjukkan hubungan antar *class* dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan.

4. *Sequence* Diagram

Sequence diagram menjelaskan secara detil urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari *use case*: interaksi yang terjadi antar *class*, operasi apa saja yang terlibat, urutan antar operasi, dan informasi yang diperlukan oleh masing-masing operasi.

II.6.3 Pengujian Validasi

Pengujian validasi digunakan untuk mengetahui apakah sistem yang dibangun sudah benar dan sesuai dengan yang dibutuhkan. Bagian-bagian yang telah dirumuskan dalam daftar kebutuhan fungsional dan hasil analisis kebutuhan akan menjadi acuan untuk melakukan pengujian validasi. Pengujian validasi menggunakan metode pengujian *Black Box*.

Black Box Testing atau pengujian *black box* [16] merupakan pendekatan pengujian dimana program sebagai suatu '*black-box*' atau kotak hitam. Pengujian *black box* berfokus pada persyaratan fungsional perangkat lunak. Metode ini memungkinkan *software developer* untuk mendapatkan serangkaian kondisi input yang mempergunakan semua persyaratan fungsional program.

Pengujian *black box* berusaha menemukan kesalahan dalam beberapa kategori, diantaranya :

a. Fungsi-fungsi yang tidak benar atau hilang.
b. Kesalahan *interface*.
c. Kesalahan dalam struktur data atau akses *database* eksternal.
d. Kesalahan dan kinerja.
e. Inisialisasi dan kesalahan terminasi.

Salah satu jenis pengujian *black box* yang akan digunakan pada tugas akhir ini adalah *Equivalence Partitioning*. *Equivalence Partitioning* merupakan metode pengujian *black box* yang membagi domain input dari suatu program ke dalam kelas data dari mana *test case* dapat dilaksanakan. Input dan output data terdapat di *class* yang berbeda yang sesuai dengan *class* inputnya. Masing-masing *class* pada *equivalence partition* diproses dimana program akan memproses anggota *class-class* tersebut secara ekuivalen.

II.6.4 Pengujian Akurasi

Pada tugas akhir ini, untuk pengujian akurasi dari hasil pengelompokkan yang diambil dari tiga situs berita digunakan suatu standar yang disebut *Confusion Matrix*. *Confusion matrix* merupakan sebuah matrik yang memberikan gambaran mengenai tingkat kesalahan atau *error rate* dan kualitas prediksi sebuah model. Ukuran dimensi *confusion matrix* adalah L x L di mana L adalah jumlah nilai yang ingin dibandingkan [17]. *Confusion matrix* ini akan digunakan untuk mengetahui apakah topik informasi berita yang ditentukan sudah dalam satu *cluster* atau sebaliknya yang dilakukan secara manual dan oleh sistem. Tabel II.3 pada di bawah ini merupakan contoh sebuah matriks yang menggunakan dua kelas, yaitu "*cluster"* dan "bukan *cluster*".

Tabel II.3 *Confusion Matrix*

Actual Class / Predicted Class		Sistem	
		Cluster	Bukan *Cluster*
Manual	*Cluster*	a	b
	Bukan *Cluster*	c	d

Setiap kolom dari *confusion matrix* mewakili contoh di kelas yang dilakukan oleh sistem, sedangkan setiap baris mewakili contoh di kelas yang dilakukan secara manual. Setelah diketahui nilai a, b, c, dan d maka selanjutnya melakukan perhitungan nilai *precision* dan akurasinya. *Precision* merupakan jumlah nilai suatu kelas yang bernilai "benar" yang berhasil diprediksi secara tepat, sedangkan akurasi merupakan jumlah nilai yang diprediksi secara tepat

terhadap jumlah seluruh nilai/sampel. Berikut persamaan dari *precision* dan akurasi.

$$\text{Akurasi} = \frac{a+d}{a+b+c+d} \qquad \text{Persamaan II-1}$$

II.6.5 Pengujian *Beta*

Pengujian *Beta* merupakan pengujian yang dilakukan secara objektif dimana pengujian dilakukan secara langsung terhadap pengguna. Pengujian *beta* ini menggunakan wawancara atau interview yang dilakukan secara langsung terhadap responden yang akan diwawancara. Wawancara merupakan teknik pengumpulan data dimana pewawancara atau peneliti mengumpulkan data dengan mengajukan pertanyaan kepada yang diwawancara atau responden [18].