BAB II

LANDASAN TEORI

2.1 Teknologi TAGME

TAGME merupakan sebuah "*topic annotator*" yang dapat mengidentifikasi urutan kata yang berarti dalam sebuah teks singkat dan menghubungkannya ke halaman Wikipedia yang bersangkutan (Ferragina, 2009). TAGME menggunakan *anchor text* yang ada pada halaman Wikipedia untuk mengidentifikasi urutan *term* bermakna (*spot*) pada teks masukan, kemudian menggunakan setiap halaman Wikipedia yang bersesuaian dengan *anchor text* tersebut sebagai kandidat *sense* yang menjelaskan *term* yang bersangkutan. Pemilihan halaman Wikipedia pada *spot* yang memiliki lebih dari satu *sense* diselesaikan dengan menggunakan fungsi-fungsi yang secara cepat dan akurat menghasilkan kesepakatan kolektif antara seluruh pemetaan *spot* ke halaman Wikipedia (Ferragina & Scaiella, 2010).

Ferragina & Scaiella (2010) menjelaskan bahwa sebuah *anchor* untuk sebuah halaman Wikipedia *p* adalah sebuah teks yang digunakan pada halaman Wikipedia lain untuk menunjuk ke *p*. Dalam Wikipedia, hal ini dapat berupa judul dari *p*, sinonim, akronim, atau bahkan dapat tersusun dari frasa yang secara sintaks berbeda dengan judul dari *p*.

Anchor a yang sama, yang muncul secara berulang dalam Wikipedia dapat menunjuk pada banyak halaman yang berbeda. Himpunan ini ditunjukkan dengan $Pg(a)$, yaitu himpunan halaman Wikipedia yang ditunjuk oleh *anchor a*. Selain itu, TAGME juga menggunakan notasi $freq(a)$ untuk menyatakan jumlah teks *a* terjadi dalam Wikipedia, baik sebagai *anchor* maupun tidak dan $link(a)$ untuk

menyatakan jumlah teks *a* terjadi dalam Wikipedia sebagai *anchor*. Selain itu, TAGME juga menggunakan notasi $lp(a) = link(a) / freq(a)$ yang menunjukkan probabilitas (*link probability*) dari kejadian sebuah teks *a* sebagai sebuah *anchor* yang menunjuk ke satu atau beberapa halaman Wikipedia dan $\Pr(p \mid a)$ untuk menyatakan *prior-probability* suatu *anchor a* menunjuk ke sebuah halaman spesifik $p \in Pg(a)$. $\Pr(p \mid a)$ disebut juga sebagai *commonness* dari *p* (Ferragina & Scaiella, 2010).

Anotasi dari sebuah *anchor a* ke beberapa halaman $p \in Pg(a)$ ditunjukkan dengan $a \mapsto p$. Jika *a* memiliki lebih dari satu *sense*, yaitu $|Pg(a)| > 1$, *disambiguation* merupakan proses memilih satu dari beberapa *sense* untuk *a* yang terdapat pada $Pg(a)$. Selain itu, TAGME melakukan pemangkasan (*pruning*) anotasi dengan membuat pemetaan tiruan dari *anchor a* ke sebuah halaman palsu *NA*, yaitu $a \mapsto NA$ (Ferragina & Scaiella, 2010).

TAGME menggunakan tiga struktur data dan melalui tiga tahap utama, yaitu *anchor parsing*, *disambiguation*, dan *pruning*. Berikut ini dijelaskan anatomi dari TAGME (Ferragina & Scaiella, 2010).

2.1.1 Preprocessing dan Membangun Index

Terdapat tiga struktur data yang digunakan oleh TAGME, yaitu sebagai berikut.

1. *Anchor Dictionary*

   TAGME mengambil seluruh *anchor* yang terdapat pada halaman Wikipedia, kemudian menggabungkannya dengan judul dari seluruh halaman pengalihan. *Anchor* yang terdiri dari hanya satu karakter atau angka akan dikeluarkan dari

dictionary. Selain itu, *anchor a* yang memiliki nilai frekuensi absolut ($link(a) < 2$) atau frekuensi relatif ($lp(a) < 0.1\%$) cukup kecil akan dikeluarkan dari *dictionary*.

2. *Page Catalog*

   TAGME mengambil seluruh halaman Wikipedia dan mengeluarkan halaman *disambiguation*, halaman daftar, dan halaman pengalihan.

3. *In-link Graph*

   Hal ini merupakan *directed graph* dengan himpunan halaman dalam *page catalog* merupakan himpunan *vertex* dan himpunan *link* antar halaman tersebut merupakan himpunan *edge*.

2.1.2 Anchor Parsing

TAGME menerima masukan berupa teks singkat, melakukan *tokenization*, dan mendeteksi *anchor* dengan melakukan *query* pada *anchor dictionary* yang memiliki panjang sampai enam kata. Apabila *anchor* a_1 merupakan *substring* dari *anchor* a_2, TAGME akan membuang a_1 jika $lp(a_1) < lp(a_2)$. Di sisi lain, dapat terjadi kasus dimana $lp(a_1) > lp(a_2)$. Jika diketahui $freq(a_1) \geq freq(a_2)$, kasus ini dapat terjadi jika $link(a_1) >> link(a_2)$. Dalam kasus ini TAGME tetap mempertahankan kedua *anchor*.

2.1.3 Anchor Disambiguation

Diberikan sebuah himpunan dari *anchor* A_T, yang dideteksi dalam fragmen teks *T*, TAGME mencoba untuk men-*disambiguate* setiap *anchor* $a \in A_T$ dengan menghitung sebuah nilai (*score*) untuk setiap kemungkinan *sense* p_a dari

a ($p_a \in Pg(a)$). Nilai ini merupakan bentuk kesepakatan kolektif antara *sense* p_a (halaman Wikipedia) dan *sense* dari seluruh *anchor* lain yang terdeteksi pada *T*. Nilai dari $a \mapsto p_a$ dievaluasi dengan menggunakan skema *voting* yang menghitung nilai *vote* dari setiap *anchor* lain $b \in A_T \setminus \{a\}$ untuk anotasi tersebut. Jika *b* memiliki banyak *sense* ($|Pg(b)| > 1$), TAGME menghitung *vote* yang diberikan sebagai *average relatedness* antara setiap *sense* p_b dari *b* dan *sense* p_a yang ingin diasosiasikan ke *a*. Nilai *relatedness* antara dua halaman Wikipedia p_a dan p_b dapat dihitung dengan menggunakan persamaan 2.1 berikut ini.

$$rel(p_a, p_b) = \frac{\log(\max(|in(p_a), in(p_b)|)) - \log(|in(p_a) \cap in(p_b)|)}{\log(W) - \log(\min(|in(p_a), in(p_b)|))} \quad \text{.... Persamaan 2.1}$$

Pada persamaan 2.1, $in(p)$ adalah himpunan halaman Wikipedia yang menunjuk ke halaman *p* dan *W* adalah jumlah halaman dalam Wikipedia. Nilai *vote* yang diberikan oleh *anchor b* ke anotasi $a \mapsto p_a$ dapat dihitung dengan menggunakan persamaan 2.2 berikut ini.

$$vote_b(p_a) = \frac{\sum_{p_b \in Pg(b)} rel(p_a, p_b) \cdot \Pr(p_b \mid b)}{|Pg(b)|} \quad \text{... Persamaan 2.2}$$

Nilai total dari anotasi $a \mapsto p_a$ dihitung dengan menjumlahkan nilai *vote* dari seluruh *anchor* lain *b* yang dideteksi pada *T*, seperti ditunjukkan pada persamaan 2.3 berikut ini.

$$rel_a(p_a) = \sum_{b \in A_T \setminus \{a\}} vote_b(p_a) \quad \text{... Persamaan 2.3}$$

Nilai yang dihasilkan dari persamaan 2.3 kemudian dikombinasikan dengan *commonness* dari *sense* p_a untuk *a*. Proses kombinasi kedua nilai ini dilakukan dengan pendekatan yang disebut *Disambiguation by Threshold* (DT). Pendekatan ini dilakukan dengan menentukan ε *sense* dengan nilai *relatedness* (hasil persamaan 2.3) tertinggi dalam $Pg(a)$, kemudian menganotasi *a* dengan *sense* p_a yang memiliki nilai *commonness* $\Pr(p_a \mid a)$ tertinggi. Nilai ε yang dipilih adalah 30%. Karena kecepatan merupakan perhatian utama, seluruh *sense* yang memiliki nilai *commonness* lebih rendah dari *threshold* τ akan dikeluarkan dari proses kombinasi. Jika τ terlalu besar, *precision* berkurang karena banyak *sense* bersangkutan yang akan dieliminasi. Namun, jika τ terlalu kecil, kecepatan dan *recall* berkurang. Nilai τ yang dipilih adalah 2%.

Persamaan 2.1 yang didefinisikan oleh Milne & Witten (2008b) mengembalikan sebuah nilai antara nol dan satu, dengan nol menunjukkan keterkaitan yang kuat antara dua halaman Wikipedia dan satu menandakan kedua halaman Wikipedia tersebut tidak berkaitan (Cilibrasi & Vitanyi, 2007). Pada penelitian ini, digunakan persamaan 2.4 sebagai berikut (Ceccarelli, Lucchese, Orlando, Perego, & Trani, 2013).

$$rel(p_a, p_b) = 1 - \frac{\log\left(\max\left(\left|in(p_a)\right|, \left|in(p_b)\right|\right)\right) - \log\left(\left|in(p_a) \cap in(p_b)\right|\right)}{\log(W) - \log\left(\min\left(\left|in(p_a)\right|, \left|in(p_b)\right|\right)\right)} \quad \text{.... Persamaan 2.4}$$

Persamaan 2.4 mengembalikan sebuah nilai antara nol dan satu, dengan satu menunjukkan keterkaitan yang kuat antara dua halaman Wikipedia dan nol menandakan kedua halaman Wikipedia tersebut tidak berkaitan. Hal ini menjadikan persamaan 2.4 lebih cocok digunakan pada pendekatan

Disambiguation by Threshold yang mengambil 30% *sense* dengan nilai total *relatedness* tertinggi (hasil persamaan 2.3).

2.1.4 Anchor Pruning

Tahap *disambiguation* menghasilkan sebuah himpunan dari kandidat anotasi, satu untuk setiap *anchor* yang dideteksi pada teks masukan *T*. Anggota dari himpunan ini perlu dikurangi (*pruned*) dalam rangka menghapus anotasi yang tidak berarti. Hal ini dilakukan dengan menggunakan fungsi *scoring* sederhana yang memperhitungkan dua *feature*, yaitu *link probability* $lp(a)$ dari *anchor a* dan *coherence* antara kandidat anotasi $a \mapsto p_a$ dari *anchor* tersebut dan kandidat anotasi dari *anchor* lain dalam *T*. Jika *S* adalah himpunan dari *sense* yang berasosiasi dengan *anchor* dalam *T* yang dihasilkan pada tahap *disambiguation* ($|S| > 1$), nilai *coherence* dapat dihitung dengan menggunakan persamaan 2.5 berikut ini.

$$coherence(a \mapsto p_a) = \frac{1}{|S|-1} \sum_{p_b \in S \setminus \{p_a\}} rel(p_b, p_a) \quad \text{... Persamaan 2.5}$$

Setiap *pruner* menghitung *pruning score* $\rho(a \mapsto p)$ untuk setiap kandidat anotasi, kemudian membandingkannya dengan *threshold* ρ_{NA}. Jika $\rho(a \mapsto p) < \rho_{NA}$, anotasi tersebut dibuang dengan melakukan $a \mapsto NA$. Parameter ρ_{NA} digunakan untuk membuat keseimbangan antara *precision* dan *recall*. *Pruning score* dihitung dengan mengambil rerata dari $lp(a)$ dan $coherence(a \mapsto p_a)$, seperti ditunjukkan pada persamaan 2.6 berikut ini.

$$\rho_{AVG}(a \mapsto p_a) = \frac{lp(a) + coherence(a \mapsto p_a)}{2} \quad \text{... Persamaan 2.6}$$

2.2 Wikipedia

Wikipedia merupakan ensiklopedia, yang berisi seluruh pengetahuan umum, yang dapat disunting oleh siapapun. Wikipedia telah menjadi sumber informasi dan sasaran pilihan pertama bagi banyak orang yang mencari pengetahuan di internet (Lih, 2009). Wikipedia merupakan ensiklopedia elektronik terbesar dan paling banyak digunakan (Medelyan, Milne, Legg, & Witten, 2009).

Lih (2009) mengungkapkan bahwa artikel pada Wikipedia ditulis dengan menampilkan fakta penting terlebih dahulu, yaitu di bagian atas artikel. Hal ini disebut juga dengan pola penulisan "piramida terbalik". Kemudian, dengan menggunakan WikiMarkup, teks dalam artikel dikelola, tiap subjudul dipisahkan secara visual, dan daftar isi dari artikel dibuat secara otomatis. WikiMarkup merupakan sebuah bahasa *markup* dasar untuk Wikipedia.

Terdapat beberapa fitur struktural dari Wikipedia, yaitu sebagai berikut (Medelyan, Milne, Legg, & Witten, 2009).

1. Artikel

   Artikel merupakan unit dasar informasi dalam Wikipedia. Artikel ditulis dalam bentuk teks bebas yang mengikuti seperangkat petunjuk editorial dan struktural untuk mempromosikan konsistensi dan kohesi.

2. Halaman *Disambiguation*

   Halaman ini berisi daftar judul artikel yang memiliki kesamaan terminologi, tetapi masing-masing artikel tersebut menjelaskan konsep yang berbeda. Halaman disambiguasi merupakan sumber informasi yang dapat menolong pengguna jika suatu *term* merupakan homonim.

3. Pengalihan (*Redirects*)

   Halaman pengalihan (*redirects*) hanya berisi suatu *link* yang mengarah ke suatu halaman target.

4. *Hyperlink*

   Artikel-artikel Wikipedia saling berhubungan menggunakan *hyperlink*.

5. Struktur Kategori

   Setiap artikel berasosiasi dengan beberapa kategori. Struktur kategori pada Wikipedia tidak membentuk struktur pohon, tetapi berupa *graph*.

6. *Templates* dan *Infoboxes*

   Template merupakan struktur yang dapat digunakan kembali untuk menambahkan informasi ke halaman lain dengan cara yang efisien. Sebuah *infoboxes* merupakan jenis khusus dari *template* yang menampilkan informasi faktual secara terstruktur dengan format yang seragam.

7. Halaman Diskusi

   Halaman ini merepresentasikan sebuah forum untuk mendiskusikan bagaimana suatu halaman dikritik, diperbaiki, dan dikembangkan.

8. Riwayat Penyuntingan

   Halaman ini berisi hal-hal terkait riwayat penyuntingan suatu artikel.

2.3 Entity Annotation

Entity annotation merupakan sebuah pendekatan yang bertujuan untuk mengatasi kekurangan pendekatan "*bag-of-words*" dalam memberikan representasi semantik untuk suatu dokumen teks. Ide utama dari pendekatan ini adalah untuk mengidentifikasi urutan kata yang berarti (disebut juga *mention*) pada teks masukan, kemudian menganotasikannya dengan suatu *identifier* (disebut

juga *entity*) yang diambil dari suatu katalog, seperti Wikipedia. Proses dari *entity annotation* melibatkan tiga langkah utama, yaitu sebagai berikut (Cornolti, Ferragina, & Ciaramita, 2013).

1. Mendeteksi kandidat *mention* dan menghubungkannya ke seluruh *entity* yang memiliki kemungkinan menjelaskan *mention* tersebut.
2. Memilih *entity* yang tepat dan paling baik dalam mendeskripsikan setiap *mention* tersebut.
3. Membuang *mention* dan *entity* yang terhubung jika dinilai tidak terkait dengan interpretasi semantik dari dokumen teks.

Cornolti, Ferragina, & Ciaramita (2013) memperkenalkan hierarki dari permasalahan *entity annotation* dengan menggunakan terminologi sebagai berikut.

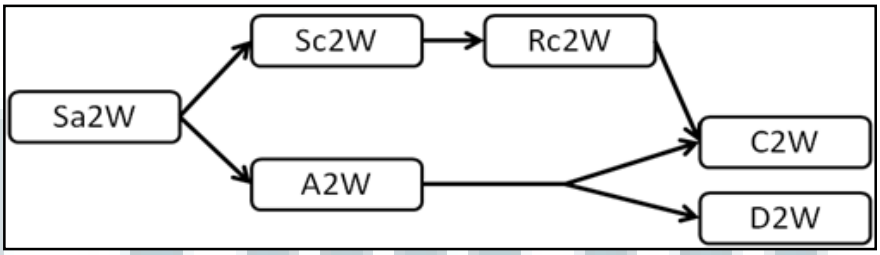
1. Sebuah *entity* (konsep atau topik) adalah sebuah artikel Wikipedia yang diidentifikasi secara unik oleh nomor *ID* halamannya.
2. Sebuah *mention* adalah terjadinya urutan *term* yang terdapat dalam sebuah teks. Hal ini dikodekan oleh pasangan bilangan bulat (p,l), yang mana p adalah posisi dari kejadian dan l adalah panjang dari *mention*.
3. Sebuah *annotation* adalah hubungan dari sebuah *mention* ke sebuah *entity*. Hal ini dikodekan oleh pasangan bilangan bulat (m,e), yang mana $m = (p,l)$ adalah *mention* dan e adalah nomor *ID* halaman dari *entity* yang terhubung.
4. Sebuah *tag* adalah *annotation* dari sebuah teks dengan sebuah *entity* yang secara eksplisit menangkap sebuah topik dari teks masukan.
5. Sebuah *score* s adalah sebuah nilai *real* antara nol dan satu yang diberikan ke sebuah *annotation* atau sebuah *tag* untuk mengindikasikan kebenarannya.

Permasalahan yang berhubungan dengan *entity annotation* dapat dikelompokkan menjadi dua kelas utama, yaitu mengidentifikasi *annotation* (kemungkinan juga nilainya) yang berarti menemukan pasangan *mention-entity* dan menemukan *tag* (kemungkinan juga nilai atau peringkatnya) yang berarti hanya memperhitungkan *entity*. Terdapat enam permasalahan yang dapat dijelaskan sebagai berikut (Cornolti, Ferragina, & Ciaramita, 2013).

1. *Disambiguate to* Wikipedia (D2W). Permasalahan ini memiliki masukan berupa teks atau himpunan dari *mention* dan keluaran berupa himpunan dari *annotation* yang relevan. Penyelesaian dilakukan dengan menghubungkan setiap *mention* ke *entity* yang sesuai.
2. *Annotate to* Wikipedia (A2W). Permasalahan ini memiliki masukan berupa teks dan keluaran berupa himpunan *annotation* yang relevan. Penyelesaian dilakukan dengan mengidentifikasi *mention* yang relevan pada teks masukan dan menghubungkan setiap *mention* tersebut ke *entity* yang bersangkutan.
3. *Scored-annotate to* Wikipedia (Sa2W). Permasalahan ini memiliki masukan berupa teks dan keluaran berupa himpunan *annotation* yang relevan dan memiliki *score*. Penyelesaian dilakukan dengan cara yang sama pada A2W, tetapi dalam hal ini setiap *annotation* diberikan sebuah *score* yang merepresentasikan kemungkinan *annotation* tersebut benar.
4. *Concepts to* Wikipedia (C2W). Permasalahan ini memiliki masukan berupa teks dan keluaran berupa himpunan *tag* yang relevan. Penyelesaian dilakukan dengan mengambil himpunan *entity* yang sesuai sebagai himpunan *tag*, yang mana setiap *entitiy* tersebut berpasangan dengan *mention* pada teks masukan.

5. *Scored concepts to* Wikipedia (Sc2W). Permasalahan ini memiliki masukan berupa teks dan keluaran berupa himpunan *tag* yang relevan dan memiliki *score*. Penyelesaian dilakukan dengan cara yang sama pada C2W, tetapi dalam hal ini setiap *tag* diberikan sebuah *score* yang merepresentasikan kemungkinan *annotation* yang dilakukan benar.
6. *Ranked-concepts to* Wikipedia (Rc2W). Permasalahan ini memiliki masukan berupa teks dan keluaran berupa himpunan *tag* berperingkat yang relevan. Penyelesaian dilakukan dengan mengidentifikasi *entitiy* yang berasosiasi dengan *mention* pada teks masukan dan menentukan peringkat dari setiap *entity* berdasarkan relevansinya dengan topik pada teks masukan.

Cornolti, Ferragina, & Ciaramita (2013) menjelaskan bahwa permasalahan paling umum dan sulit adalah Sa2W. Hal ini dikarenakan permasalahan lain merupakan penyederhanaan dari permasalahan Sa2W. Gambar 2.1 berikut merupakan DAG (*Directed Acyclic Graph*) dari penyederhanaan antara keenam permasalahan *entity annotation*. Panah menunjuk dari permasalahan yang lebih sulit ke permasalahan yang lebih mudah.



Gambar 2.1 DAG dari Penyederhanaan Permasalahan *Entity Annotation*
(Cornolti, Ferragina, & Ciaramita, 2013, dengan perubahan)

2.4 WordPress Sebagai Content Management System

Content Management System (CMS) merupakan sebuah *software package* yang digunakan untuk membangun dan mengelola konten *website*. Hal ini dapat

dilakukan secara cepat dan mudah, bahkan oleh pelaku non-*technical* (Ghorecha & Bhatt, 2013). CMS dapat digunakan untuk menulis, mengedit, dan mempublikasi suatu pekerjaan secara *online* (Hedengren, 2010).

WordPress adalah sebuah *Content Management System* (CMS), walaupun pada mulanya digunakan hanya untuk mempublikasikan *blog*. Pada dasarnya, WordPress dapat digunakan untuk mengelola konten tertulis, gambar, suara, dan video (Hedengren, 2010). WordPress merupakan CMS yang didukung oleh navigasi halaman, manajemen pengguna, pembuatan *blog*, dan berbagai perangkat pengelolaan (Olinik & Armitage, 2011).

WordPress merupakan pilihan CMS yang baik, terutama jika membangun sebuah *editorial site*. WordPress menjadi pilihan jika memerlukan CMS yang yang memiliki karakteristik sebagai berikut (Hedengren, 2010).

1. *Open source* dan tidak berbayar.
2. Cepat dan mudah digunakan.
3. Mudah dikembangkan.
4. Mudah didesain dan dikembangkan *plugin*-nya.
5. Bekerja secara baik dengan teks.
6. Cukup baik dengan gambar-gambar.

WordPress memiliki sistem yang sederhana dan struktur *file* yang mudah dipahami. Selain itu, WordPress memisahkan antara komponen yang umum dimodifikasi dan *core installation*-nya. Pada pengembangan di sisi *client*, WordPress bekerja secara baik dengan HTML, CSS, dan JavaScript. WordPress juga mendukung pengembangan di sisi *server* dengan PHP 5 dan MySQL 5.

Dengan demikian, dapat dipahami bahwa penyusun utama dari WordPress adalah HTML, CSS, JavaScript, PHP, dan MySQL (Olinik & Armitage, 2011).

2.5 Plugin WordPress

Plugin WordPress digunakan ketika ingin mengembangkan fungsionalitas WordPress dengan fitur-fitur tambahan (Hedengren, 2010). *Plugin* WordPress memungkinkan proses modifikasi, pengaturan, dan peningkatan sebuah *blog* WordPress dilakukan dengan mudah tanpa memodifikasi pemrograman inti dari WordPress. Secara mendasar, *plugin* WordPress didefinisikan sebagai sebuah program, atau kumpulan dari satu atau lebih fungsi, ditulis menggunakan bahasa *scripting* PHP, yang menambahkan sebuah himpunan fitur atau layanan spesifik ke WordPress *weblog*, yang dapat dengan mudah diintegrasikan dengan *weblog* menggunakan *access point* dengan metode-metode yang disediakan oleh WordPress *Plugin Application Program Interface* (API) (WordPress.org, 2015).

Sebuah WordPress *plugin* harus memiliki minimal sebuah *file* PHP. Bagian atas *file* PHP utama *plugin* tersebut harus mengandung sebuah *header* informasi *plugin* standar. *Header* ini memungkinkan WordPress mengenali kehadiran suatu *plugin* dan menambahkannya ke layar pengelolaan Plugins sehingga dapat diaktifkan (WordPress.org, 2015). Gambar 2.2 menunjukkan format *header* yang digunakan WordPress untuk mengidentifikasi suatu *plugin*. Informasi minimum yang dibutuhkan WordPress untuk mengenali suatu *plugin* adalah baris "Plugin Name". Selain itu, urutan dari baris tidak menjadi hal yang penting (WordPress.org, 2015).

```
<?php
/**
 * Plugin Name: Name of the plugin, must be unique.
 * Plugin URI: http://URI_Of_Page_Describing_Plugin_and_Updates
 * Description: A brief description of the plugin.
 * Version: The plugin's version number. Example: 1.0.0
 * Author: Name of the plugin author
 * Author URI: http://URI_Of_The_Plugin_Author
 * Text Domain: Optional. Plugin's text domain for localization. Example: mytextdomain
 * Domain Path: Optional. Plugin's relative directory path to .mo files. Example: /locale/
 * Network: Optional. Whether the plugin can only be activated network wide. Example: true
 * License: A short license name. Example: GPL2
 */
```

Gambar 2.2 Format *Header Plugin* WordPress
(WordPress.org, 2015)

2.6 Extensible Markup Language (XML)

Extensible Markup Language (XML) merupakan sebuah format teks sederhana dan fleksibel yang diturunkan dari SGML (ISO 9979) dan memiliki peranan penting dalam pertukaran berbagai jenis data pada *web* (World Wide Web Consortium, 2015). Selain itu, XML dapat dengan mudah dibaca, baik oleh manusia, maupun oleh perangkat lunak (Fawcett, Quin, & Ayers, 2012).

Salah satu tujuan dari penggunaan XML adalah untuk memisahkan antara data dan tampilan dari data tersebut. Hal ini berarti ketika data berpindah, misalnya melalui jaringan, *bandwidth* tidak terbuang oleh informasi mengenai tampilan dari data. Selain itu, XML dapat digunakan untuk menampilkan data yang bersifat hierarki dengan tetap mempertahankan kemudahan untuk dipahami oleh manusia dan perangkat lunak (Fawcett, Quin, & Ayers, 2012).

XML merupakan pilihan yang baik jika ingin melakukan pertukaran data antara aplikasi yang berbeda. Dengan demikian, data yang dihasilkan oleh satu aplikasi dapat digunakan oleh aplikasi lainnya dengan mudah (Fawcett, Quin, & Ayers, 2012).