

UNIVERSITAS GUNADARMA

FAKULTAS ILMU KOMPUTER



**PERBANDINGAN *TOOL* UNTUK MEMBANGUN
ONTOLOGY BERBASIS *RDF/OWL* DAN ILUSTRASI
IMPLEMENTASINYA**

Disusun Oleh :

Nama Kris Triyantio
NPM 12101167
Jurusan Sistem Informasi
Pembimbing DR.-Ing .Adang Suhendra, SSi,SKom,MSc
 Lintang Yuniar Banowosari, SKom, Msc

Diajukan Guna Melengkapi Sebagian Syarat
Dalam Mencapai Gelar Sarjana Strata Satu (S1)

Jakarta
2006

ABSTRACT

Kris Triyantio, 12101167

Perbandingan *Tool* Untuk Membangun *Ontology* Berbasis *RDF/OWL* dan Ilustrasi Implementasinya

Tugas Akhir . Fakultas Ilmu Komputer. 2006

Kata Kunci : *Ontology, OWL, RDF, Tool*

(xi + 108 + Lampiran)

Dengan berkembangnya teknologi informasi saat ini maka sifat informasi sekarang ini juga mengalami perkembangan. Perkembangan ini mendorong sumber informasi menjadi semakin dinamis, otonomi, beragam dan berukuran besar. Keragaman yang terjadi tidak saja pada tingkat teknis tapi juga pada tingkat representasi informasi. Perbedaan ini bisa terjadi pada sintaktis, skematis dan semantik level.

Karena keragaman ini, maka untuk pertukaran informasi antar berbagai sumber akan menjadi kendala. Untuk mengatasi kendala itu maka dikembangkan sebuah metode. Metode tersebut adalah *Semantic Web* dengan menggunakan pendekatan *ontology*. Karena metode tersebut baru berkembang, sehingga *tool* yang dapat digunakan untuk mengembangkan teknologi tersebut masih terbatas dan belum *mature*.

Melalui penulisan ini akan diuji coba beberapa *tool* yang dapat digunakan untuk pengembangan *ontology* berbasis *RDF/OWL*. Diharapkan dengan dilakukannya uji coba ini maka akan dapat memberikan kontribusi untuk para pengembang *ontology* dalam memilih *tool* yang sesuai dengan tujuan dan spesifikasi *ontology* yang akan dibuatnya. Dalam penulisan ini juga akan dibuat sebuah ilustrasi pemanfaatan *ontology* dalam informasi interoperabilitas.

LEMBAR PENGESAHAN

Komisi Pembimbing

No	Nama	Kedudukan
1	Lintang Yuniar Banowosari, SKom, MSc	Ketua
2	Sulistyo Puspitodjati, SSI, MSc	Anggota
3	Dr. Yuhilza Hanum	Anggota

Panitia Ujian

No	Nama	Kedudukan
1	Dr. Ravi Ahmad Salim	Ketua
2	Prof. Dr. Wahyudi Priyono	Sekretaris
3	Lintang Yuniar Banowosari, SKom, MSc	Anggota
4	Sulistyo Puspitodjati, SSI, MSc	Anggota
5	Dr. Yuhilza Hanum	Anggota

Tanggal Lulus : 15 Juli 2006

Mengetahui

Jakarta, 20 April 2006

Pembimbing

Bagian Sidang Sarjana

(Lintang Yuniar Banowosari, SKom, MSc)

(Drs. Edi Sukirman, MM)

KATA PENGANTAR

Dengan mengucapkan puji dan syukur kehadiran Allah Yang Maha Kuasa atas selesainya Tugas Akhir ini dengan baik.

Tugas Akhir yang berjudul "Perbandingan *Tool* Untuk Membangun *Ontology* Berbasis *RDF/OWL* Serta Ilustrasi Penggunaanya", disusun guna melengkapi sebagian syarat mencapai gelar Sarjana Strata Satu di Universitas Gunadarma.

Penulis ingin mengucapkan terima kasih kepada semua pihak yang secara langsung maupun tidak langsung telah membantu penulis dalam pembuatan Penulisan Ilmiah ini, yaitu :

1. Prof. Dr. E. S. Margianti, SE, MM selaku Rektor Universitas Gunadarma.
2. Bambang Wahyudi, SKom, MMSi selaku Dekan Fakultas Ilmu Komputer.
3. Dr.Ing. Adang Suhendra, SSi, SKom, MSc selaku Ketua Jurusan Sistem Informasi dan juga sebagai dosen pembimbing
4. Lintang Yuniar Banowosari, SKom, MSc yang telah memberikan waktunya untuk membimbing penulis dan memberikan ilmunya dengan ikhlas
5. Dr. I Wayan Simri Wicaksana, SSi, MEng yang juga telah banyak membantu memberikan saran dan ilmunya untuk keberhasilan tugas akhir ini
6. Kepada Ayah, Ibu tersayang yang dengan kasih sayangnya membantu memberikan dukungan hingga penulisan ini berhasil
7. Kepada Kakak tercinta yang dengan kemampuannya memberikan bantuan yang sangat diperlukan dalam penyusunan penulisan ini
8. Kepada yang ku sayang Viona Helena Putri yang juga dengan sabar memberikan semangat dan dorongan hingga penulisan ini dapat selesai
9. Seluruh teman-teman di PT.Radiant Centra Nusa yang memberikan fasilitasnya untuk penyelesaian penulisan ini

10. Semua pihak yang telah membantu dalam Tugas Akhir ini yang tidak dapat disebutkan satu persatu.

Namun demikian, penulis menyadari bahwa Tugas Akhir ini masih banyak terdapat kekurangan dan kelemahannya. Oleh karena itu penulis mengharapkan kritik dan saran untuk penyempurnaan tulisan ini untuk masa yang akan datang.

Akhirnya penulis berharap semoga dengan selesainya tulisan ini akan bermanfaat bagi kita semua.

Jakarta, 21 Mei 2007

Penulis

Daftar Isi

ABSTRACT	ii
Lembar Pengesahan	iii
Kata Pengantar	iv
Daftar Isi	vii
Daftar Gambar	vii
Daftar Gambar	ix
Daftar Tabel	ix
Daftar Tabel	x
1. Pendahuluan	1
1.1 Latar Belakang Masalah	1
1.2 Rumusan Masalah	3
1.3 Batasan Masalah	3
1.4 Tujuan Penulisan	3
1.5 Metode Penulisan	4
1.6 Sistematika Penulisan	4
2. <i>Interoperabilitas Informasi</i>	5
2.1 <i>Ontology</i>	5
2.1.1 Definisi <i>Ontology</i>	5
2.1.2 Komponen <i>Ontologi</i>	7
2.1.3 Metode Pengembangan <i>Ontology</i>	8
2.2 Bahasa <i>Ontology</i>	14
2.2.1 <i>XML</i>	16
2.2.1.1 Fungsi dan Tujuan <i>XML</i>	16
2.2.1.2 Sintaksis dan Elemen <i>XML</i>	17
2.2.2 <i>RDF</i>	18
2.2.2.1 Pengertian <i>RDF</i>	18
2.2.2.2 Sintaksis <i>XML</i> untuk <i>RDF</i>	21
2.2.2.3 Query <i>RDF</i>	24
2.2.3 <i>OWL</i>	25
2.2.3.1 Pengertian <i>OWL</i>	25
2.2.3.2 Sub Bahasa <i>OWL</i>	28
2.2.3.3 Rancangan <i>OWL</i>	30
2.2.3.4 Sintaksis <i>OWL</i>	32
2.3 Web <i>Semantic</i>	33
2.3.1 Definisi	33
2.3.2 Komponen Web <i>Semantic</i>	34
2.3.3 Fungsi Web <i>Semantic</i>	35

3. Perancangan	36
3.1 Taksonomi Tumbuhan di Pulau Jawa	36
3.1.1 Tumbuhan Lumut	37
3.1.2 Tumbuhan Berpembuluh	38
3.1.2.1 Tumbuhan Paku	38
3.1.2.2 Tumbuhan Berbiji	39
3.1.3 Pendeskripsian	41
3.2 Pendefinisian Komponen <i>Ontology</i>	43
3.2.1 <i>Class</i> dan Hirarki <i>Class</i>	43
3.2.2 Property <i>Class</i>	46
3.2.3 Ruang Lingkup Property	47
4. Perbandingan Tool <i>Ontology</i>	48
4.1 Metode Pengujian	48
4.1.1 Skenario Pengujian	48
4.1.2 Parameter Perbandingan	48
4.1.2.1 Instalasi	48
4.1.2.2 Kemudahan Penggunaan	50
4.1.2.3 Fasilitas	50
4.1.2.4 Faktor Lain	51
4.1.3 <i>Tool</i> yang diuji	53
4.1.3.1 <i>Protégé</i>	53
4.1.3.2 <i>Altova</i>	54
4.1.3.3 <i>SWOOP</i>	55
4.2 Pengujian	56
4.2.1 Hasil Pengujian	56
4.2.2 Analisis Hasil Pengujian	75
5. Web <i>Semantic</i> dan <i>Ontology</i>	82
5.1 Penggunaan Dalam Informasi Interoperabilitas	82
5.2 Contoh Implementasi	86
5.3 Pemanfaatan Hasil <i>Ontology</i> Tumbuhan	91
6. Penutup	95
6.1 Ringkasan dan Kesimpulan	95
6.2 Rencana Kedepan	97
Bibliografi	98
Lampiran	a

Daftar Gambar

2.1	Uschold Metodologi, bersumber dari [5]	9
2.2	Grüninger dan Fox Metodologi, bersumber dari [5]	10
2.3	KACTUS Metodologi	11
2.4	METHONTOLOGY Metodologi, bersumber dari [5]	12
2.5	Sensus Metodologi, bersumber dari [9]	12
2.6	On-To-Knowledge Metodologi, bersumber dari [30]	13
2.7	Ontology Layer, bersumber dari [23]	15
2.8	Contoh Skema <i>RDFS</i> , bersumber dari [27]	19
2.9	Contoh Graph <i>RDFS</i> , bersumber dari [28]	22
2.10	Contoh Graph Satu Jalur, bersumber dari [28]	23
3.1	Skema Tumbuhan	45
3.2	Skema Habitat	46
3.3	Skema Habitus	46
4.1	Protégé, bersumber dari [20]	54
4.2	Altova, bersumber dari [1]	56
4.3	SWOOP, bersumber dari [24]	57
5.1	Halaman web tentang Tanaman Jahe, H1. Dari [7]	87
5.2	Halaman web tentang Tanaman Jahe, H2. Dari [7]	88
5.3	Halaman web tentang Harga Jahe, H1. Dari [12]	88
5.4	Halaman web tentang Harga Jahe, H2. Dari [12]	89
5.5	Skema Semantic Web Domain Tumbuhan	92
5.6	Penggunaan Ontology Tumbuhan dengan Semantic Bank	94
1	Skema <i>OWL</i> dengan Protege L4	r

2	Skema <i>RDF</i> untuk Tumbuhan dengan Altova Semantic Web L5	s
3	Skema <i>RDF</i> untuk Habitat dan Habitus dengan Altova Semantic Web L6	t
4	Survei <i>tool</i> Halaman 1 L7	t
5	Survei <i>tool</i> Halaman 2 L8	u

Daftar Tabel

2.1	<i>Property</i>	21
2.2	<i>Classes</i>	21
4.1	Hasil Penilaian Protégé	58
4.2	Hasil Penilaian Altova Semantic Work	64
4.3	Hasil Penilaian SWOOP	70
4.4	Rangkuman Penilaian <i>tool</i>	75

Bab 1

Pendahuluan

1.1 Latar Belakang Masalah

Berbagai kegiatan saat ini, sangat membutuhkan berbagai informasi, seperti untuk perencanaan, pengambilan keputusan, evaluasi dan sebagainya. Sumber informasi saat ini semakin beragam dan banyak. Terlebih dengan semakin berkembangnya teknologi informasi dan internet, membuat sistem tidak ada batasan geografi dan waktu. Hal ini yang mendorong semakin memudahkan dalam pertukaran informasi, selain itu juga timbul keragaman sumber informasi.

Sumber informasi yang telah tersedia dewasa ini memiliki metode penyajian yang berbeda-beda walaupun tujuan dari pembuatan atau penyajian *web* tersebut sama. Ini dikarenakan teknologi *web* memiliki tingkat autonomi yang tinggi. Hal ini juga yang membutuhkan metode penafsiran yang berbeda-beda untuk masing-masing *web* tersebut walupun pada akhirnya akan memiliki hasil yang sama.

Sebuah halaman *web*, dapat menyajikan informasi dalam berbagai format, seperti gambar, data, suara, video. Kemudahan ini juga dapat berakibat sebuah informasi dapat berubah isinya atau koneksinya setiap waktu tanpa ada yang dapat mengaturnya.

Informasi di internet juga terbuka sehingga memudahkan setiap orang untuk mengaksesnya, sehingga isu keamanan menjadi sangat penting. Walaupun sebuah halaman *web* telah dilindungi dengan berbagai macam metode keamanan, tetap saja informasi yang terdapat di dalamnya mudah untuk diakses orang. Sehingga sebuah informasi dapat diakses oleh orang yang tidak berhak.

Setiap orang yang menginginkan sebuah informasi dapat dengan mudah

didapat melalui media internet, walaupun informasi tersebut tidak sepenuhnya relevan dengan apa yang diinginkan oleh orang tersebut, oleh karena itu dibutuhkan sebuah metode pencarian informasi yang dapat setidaknya mendekati atau menyaring informasi yang diinginkan oleh orang tersebut.

Keragaman yang terjadi juga ada ditingkat semantic. Contoh sederhana adalah adanya perbedaan pemahaman akan sebuah konsep. Konsep bisa dikatakan seperti pemahaman akan sebuah istilah. Perbedaan istilah yang berarti setiap istilah akan memiliki perbedaan arti tergantung penggunaan istilah tersebut dan siapa yang menggunakan istilah tersebut. Perbedaan tersebut dapat berupa sebuah istilah memiliki arti yang berbeda, atau istilah yang berbeda tetapi memiliki arti yang sama.

Hal lain pada tingkat semantik adalah taksonomi istilah. Dalam penyimpanan data juga dikenal yang namanya katalog, dalam dunia nyata katalog tersebut memiliki berbagai versi atau metode penyimpanannya tergantung pemakai katalog tersebut. Pengelompokan akan terjadi beragam, sebagai contoh ular bisa digolongkan sebagai hewan berbisa, kelompok lain menggolongkan sebagai hewan melata. Perbedaan taksonomi atau katalog ini akan menjadikan masalah dalam akses informasi.

Karena masalah keragaman tersebut, seperti yang telah dijelaskan di atas maka untuk pertukaran informasi akan menjadi suatu kendala yang sulit untuk dipertemukan tanpa ada solusi yang efektif. Salah satu pendekatan yang memungkinkan untuk menjembatani masalah ini adalah menggunakan *Semantic Web* yang memanfaatkan teknologi *Ontology*.

1.2 Rumusan Masalah

Disisi lain pengembangan *Semantic Web* yang menggunakan pendekatan *Ontology* masih memiliki kendala seperti [30] :

1. Prosedur atau pengembangan *Ontology* masih belum sempurna jika dibandingkan dengan menggunakan metode pemrograman tradisional.
2. *Tools* yang digunakan untuk mengembangkan *Ontology* masih banyak yang memiliki kelemahan, karena metode ini baru dimulai sejak tahun 1995

1.3 Batasan Masalah

Dalam penulisan ini akan dibahas perbandingan *tools* yang digunakan untuk pembuatan *Ontology* pada *RDF/OWL*. Pengevaluasian pembatasan masalah menekankan kepada penggunaan dalam kasus taksonomi tanaman. Dan juga melihat berbagai indikator diantaranya:

1. Proses Instalasi;
2. User Interface;
3. Kemudahan Penggunaan;
4. Fasilitas;
5. Lisensi.

1.4 Tujuan Penulisan

Penulisan skripsi ini memiliki berbagai tujuan, yaitu membandingkan beberapa *tool* yang digunakan untuk pengembangan *ontology*. Kemudian menyajikan hasil pengujian perbandingan tersebut.

Setelah didapat hasil pengujian *tool ontology* maka diharapkan hasil pengujian ini akan digunakan oleh pengguna lain untuk dapat memilih *tool* yang tepat untuk mengembangkan *ontology*.

1.5 Metode Penulisan

Metode yang digunakan dalam penulisan ini adalah:

1. *Study Literatur*;
2. Penentuan *ontology* yang akan digunakan untuk uji coba;
3. Penentuan *class* dalam *ontology*;
4. Penentuan Parameter pengujian;
5. Pengujian terhadap masing *tool*;
6. Pengambilan kesimpulan terhadap hasil tes yang dilakukan.

1.6 Sistematika Penulisan

Bab satu menguraikan latar belakang, rumusan masalah, batasan masalah dan tujuan penulisan. Bab dua memberikan latar belakang teori yang mendasari *ontology* dan beberapa bahasa di *ontology*, Bab tiga menjelaskan *ontology* yang akan digunakan untuk pengujian, termasuk taxonomi yang digunakan. Sistem pengujian pada *tool ontology* dan hasil pengujian akan diuraikan pada bab empat. Bab lima memberikan sebuah gambaran sederhana pemanfaatan *ontology* untuk pertukaran informasi pada semantik web. Dan terakhir pada bab enam yang merupakan bab penutup akan merangkum hasil pengujian dan juga memberikan kesimpulan serta rencana penelitian kedepan.

Bab 2

Interoperabilitas Informasi

2.1 *Ontology*

2.1.1 Definisi *Ontology*

Pengertian *ontology* sangat beragam dan berubah sesuai dengan berjalannya waktu, ada beberapa definisi *ontology* yang didefinisikan oleh Benjamins [3] yaitu:

Neches dan rekannya [21] memberikan definisi awal tentang *ontology* yaitu "Sebuah *ontology* merupakan definisi dari pengertian dasar dan relasi vokabulari dari sebuah area sebagaimana aturan dari kombinasi istilah dan relasi untuk mendefinisikan vakabulari".

Kemudian Gruber [10] memberikan definisi yang sering digunakan oleh beberapa orang, definisi tersebut adalah " *Ontology* merupakan sebuah spesifikasi eksplisit dari konseptualisme". Berdasarkan definisi Gruber tersebut banyak orang yang mengemukakan definisi tentang *ontology* diantaranya Guarino dan Giaretta [11] yang pada tahun 1995 mengumpulkan hingga tujuh definisi yang berkoresponden dengan *syntactic* dan *semantic* interpretasi. sedangkan pada tahun 1997, Borst [4] melakukan penambahan dari definisi Gruber dengan mengatakan "Sebuah *ontology* adalah spesifikasi formal dari sebuah konseptual yang diterima (*share*)".

Kemudian oleh Studer [22] mencoba mengemukakan definisi tentang *ontology* yang mengambil acuan dari definisi yang dikemukakan oleh Gruber dan Borst, definisi tersebut adalah : "Konseptualisasi mengacu kepada sebuah model abstrak dari beberapa fenomena di dunia dengan memiliki identifikasi konsep yang relevan dari fenomena tersebut. Yang dimaksud dengan eksplisit adalah tipe dari konsep yang digunakan, dan batasan dari eksplisit yang di-

gunakan. *Shared* adalah merefleksikan sebuah *ontology* mencoba menangkap pengetahuan secara konsesus yang tidak merupakan hal yang hanya terkait pada individu tetapi diterima oleh sebuah group/domain".

Barnaras [2] pada proyek KACTUS memberikan definisi *ontology* yang berdasarkan pada pengembangan *ontology*. Definisi yang diberikan adalah : "Sebuah *ontology* memberikan pengertian untuk penjelasan secara eksplisit dari konsep terhadap representasi pengetahuan pada sebuah *knowledge base*". Proyek SENSUS [26] juga memberikan definisi : "Sebuah *ontologi* adalah sebuah struktur hirarki dari istilah untuk menjelaskan sebuah *domain* yang dapat digunakan sebagai landasan untuk sebuah *knowledge base*".

Ada buku yang memberikan definisi tentang *Ontology*, salah satunya adalah "The Semantic Web" [6], definisi dari *Ontology* adalah :

1. Salah satu cabang metafisika yang terfokus pada alam dan hubungan antara makhluk hidup;
2. Teori tentang sifat alami makhluk hidup.

Ontology merupakan suatu teori tentang makna dari suatu objek, *property* dari suatu objek, serta relasi objek tersebut yang mungkin terjadi pada suatu *domain* pengetahuan. Pada tinjauan filsafat, *ontology* adalah studi tentang sesuatu yang ada. Selain itu *ontology* adalah sebuah konsep yang secara sistematis menjelaskan tentang segala sesuatu yang ada atau nyata. Dalam bidang *Artificial Intelligence* (AI) *ontology* memiliki dua pengertian yang berkaitan. Pertama *ontology* merupakan kosakata representasi yang sering dikhususkan untuk *domain* atau subyek pembahasan tertentu. Kedua, sebagai suatu *body of knowledge* untuk menjelaskan suatu bahasan tertentu.

Literature yang berisi tentang *Artificial Intelligence* banyak menjelaskan tentang definisi *ontology*, banyak yang bertentangan satu dengan yang lainnya.

Tetapi dapat diambil kesimpulan, *ontology* adalah sebuah uraian formal yang menjelaskan tentang sebuah konsep dalam sebuah *domain* tertentu (*Classes*, terkadang disebut *concepts*), *properties* dari masing-masing konsep menjelaskan bermacam-macam corak dan *atribut* dari sebuah *concept* (*Slots*, terkadang disebut *roles* atau *properties*), dan batasan-batasan (*facets*, terkadang disebut *role restrictions*). Sebuah *ontology* bersama dengan beberapa set *instances* dari *class* membentuk sebuah *Knowledge base*.

Secara umum, *ontology* digunakan pada Artificial Intelligence (AI) dan persentasi pengetahuan. Segala bidang ilmu yang ada di dunia, dapat menggunakan metode *ontology* untuk dapat berhubungan dan saling berkomunikasi dalam hal pertukaran informasi antara sistem-sistem yang berbeda.

2.1.2 Komponen *Ontologi*

Ontology menggunakan banyak variasi struktur, tergantung dari penggunaan bahasa *ontology* termasuk sintaksis yang digunakan. Perlu diingat adalah *ontology* tidak melakukan apapun, fungsi perhitungan dan lainnya yang memproses *ontology* tidak hanya tergantung dari data yang terdapat dalam *ontology* tersebut, tetapi juga tergantung kepada aplikasi yang digunakan.

Ontology memiliki beberapa komponen yang dapat menjelaskan *ontology* tersebut, diantaranya [30]:

- Konsep (*Concept*)

Digunakan dalam pemahaman yang luas. Sebuah konsep dapat sesuatu yang dikatakan, sehingga dapat pula merupakan penjelasan dari tugas, fungsi, aksi, strategi, dan sebagainya.

Concept juga dikenal sebagai *classes*, *object* dan *catagories*.

- Relasi (*relation*)

Merupakan representasi sebuah tipe dari interaksi antara konsep dari sebuah domain. Secara formal dapat didefinisikan sebagai subset dari sebuah produk dari n set, $R:C_1 \times C_2 \times \dots \times C_n$. Sebagai contoh dari relasi binari termasuk *subclass-of* dan *connected-to*.

- Fungsi (*functions*)

Adalah sebuah relasi khusus dimana elemen ke n dari relasi adalah unik untuk elemen ke $n-1$. $F:C_1 \times C_2 \times \dots \times C_{n-1} \rightarrow C_n$, contohnya adalah *Mother-of*.

- Aksiom (*axioms*)

Digunakan untuk memodelkan sebuah *sentence* yang selalau benar.

- *Instances*

Digunakan untuk merepresentasikan elemen.

2.1.3 Metode Pengembangan *Ontology*

Proses pengembangan *ontology* lebih merupakan "Kerajinan Tangan", dibandingkan kegiatan "*engineering*". Setiap kelompok mengembangkan metode yang dikembangkan sendiri, baik dalam kriteria, fase, maupun tujuan pada pengembangan *ontology*. Belum adanya persetujuan atau kesepakatan mengenai metode tertentu untuk pengembangan *ontology* kerap menyulitkan untuk mencapai tujuan dari *ontology* dalam ketentuan "*reuse*".

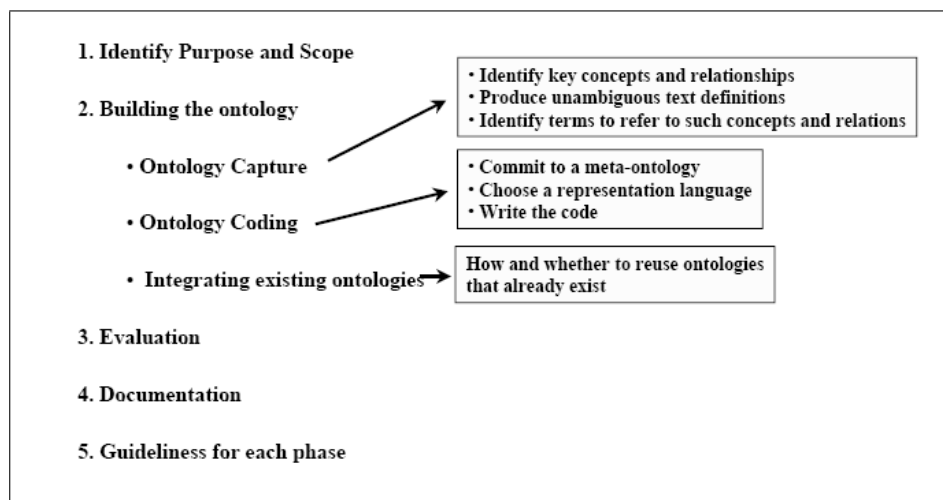
Beberapa metode pengembangan *ontology* yang telah berkembang selama ini dan telah digunakan oleh banyak pihak yang bersumber dari beberapa paper dari Benjamins [3], kemudian dari Noy [16], Wache [29], dan Sure [23]. Metodologi yang digunakan untuk pengembangan *ontology* adalah :

- Metodologi Uschold

Metode ini memiliki empat fase utama, yaitu :

1. Mendefinisikan tujuan dan cakupan dari *ontology*;
2. Membangun *ontology* dengan langkah *ontology capture* yang merupakan pengumpulan pengetahuan, *ontology coding* membangun model konsep dan mengintegrasikan *ontology* yang telah ada (*reuse*);
3. Evaluasi dengan verifikasi dan validasi;
4. Petunjuk setiap fase dan dokumentasi.

Fase yang digunakan oleh Uschold dapat digambarkan pada skema 2.1.



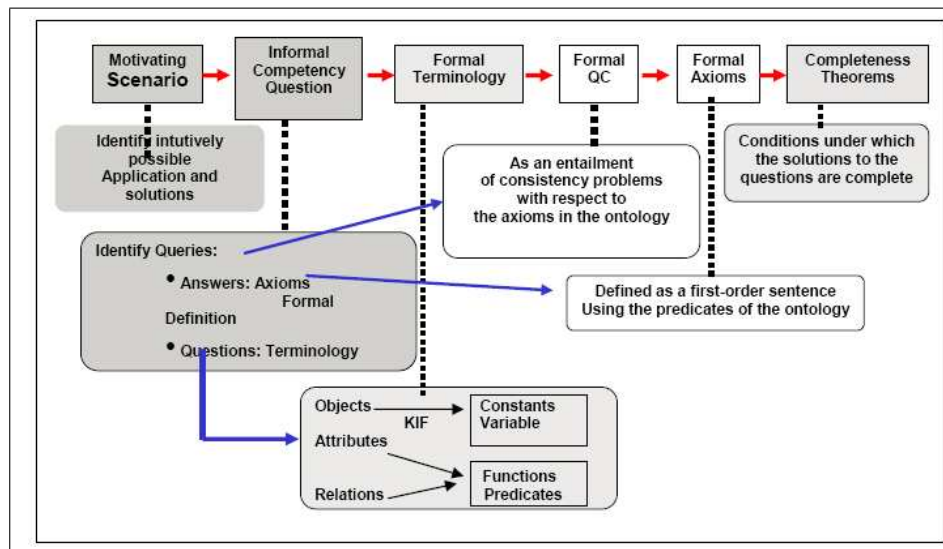
Gambar 2.1: Uschold Metodologi, bersumber dari [5]

- Metodologi Grüninger dan Fox

Grüninger dan Fox yang mengembangkan metode berdasarkan pengalaman ketika membangun *ontology* pada proyek TOVE. Hal yang paling utama adalah pembuatan model logik dari *ontology*, model ini tidak dibangun secara langsung. Pertama adalah motivasi dengan skenario pada aplikasi. Pendeskripsian dan formalisasi berdasarkan *first-order*

kalkulus. Dengan komposisi dan de-komposisi mekanisasi, akan membantu dalam integrasi *ontology*.

Metodologi yang dikembangkan oleh Grüninger dan Fox memiliki model seperti gambar 2.2 dibawah ini :



Gambar 2.2: Grüninger dan Fox Metodologi, bersumber dari [5]

- Metodologi KACTUS

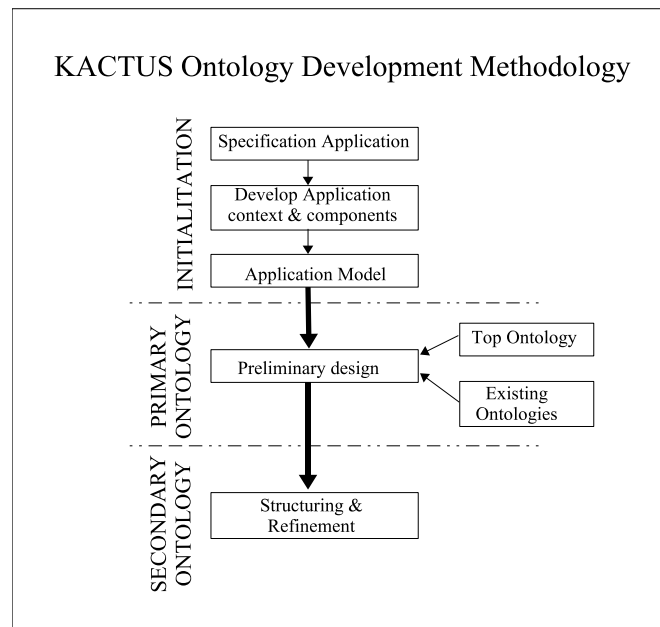
Metode ini memiliki tahapan umum sebagai berikut :

1. Spesifikasi dari aplikasi;
2. Design awal berdasarkan pada katagori *top-level ontology*;
3. Penyempurnaan dan restrukturung *ontology*.

Gambar 2.3 merupakan skema dari methodology KACTUS.

- Metodologi *Methontology*

Metodologi *Methontology* yang dikemukakan Gomez-Perez (1996) termasuk:



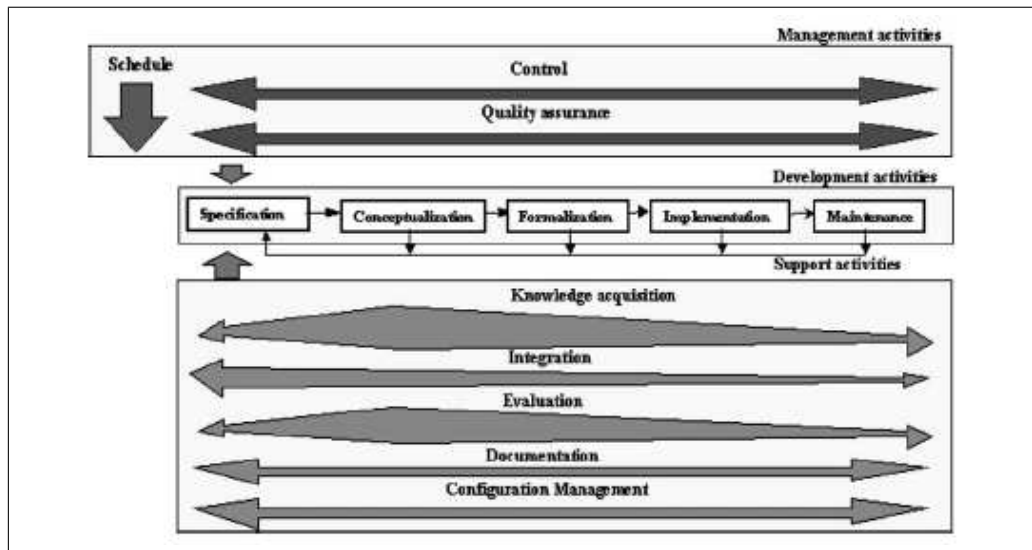
Gambar 2.3: KACTUS Metodologi

1. Mengidentifikasi proses pengembangan *ontology*;
2. Menentukan langkah-langkah yang akan dilalui oleh *ontology* tersebut selama masa hidupnya;
3. Langkah yang dilakukan untuk setiap aktifitas, teknik pendukung dan masa evaluasi;
4. Menyiapkan ORSD untuk mengetahui kebutuhan *ontology* agar sesuai dengan kebutuhan perangkat lunak;

Metodologi *Methontology* memiliki skema seperti gambar 2.4 dibawah ini.

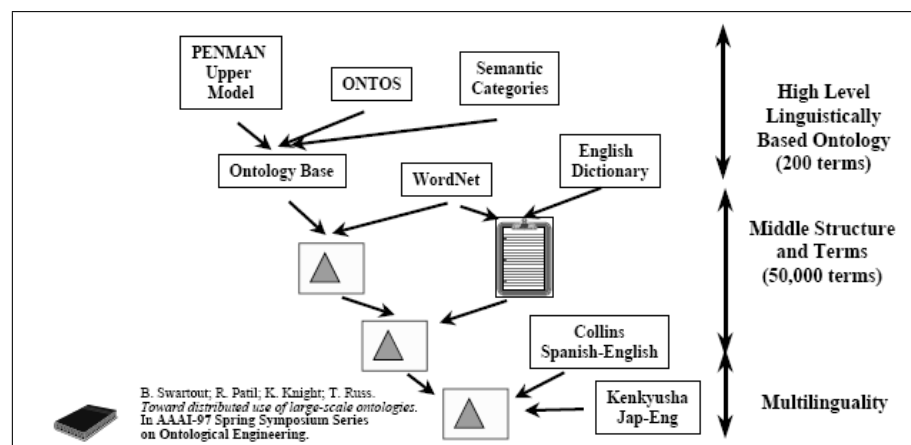
- Metodologi Sensus

Melakukan pengembangan *ontology* dengan langkah-langkah sebagai berikut :



Gambar 2.4: METHONTOLOGY Metodologi, bersumber dari [5]

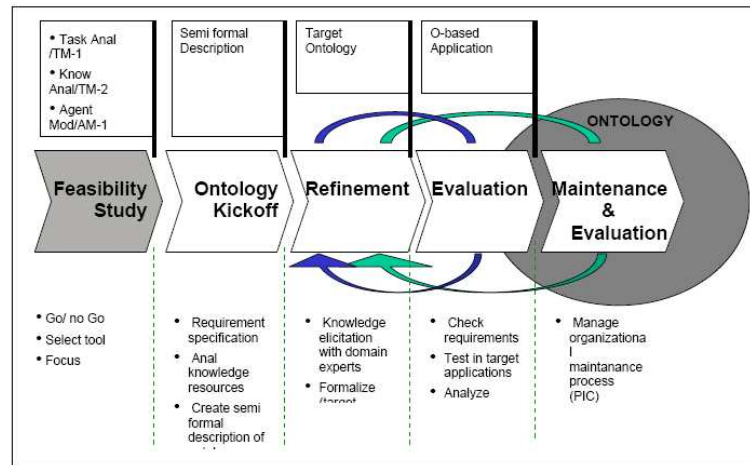
1. Mengidentifikasi istilah yang penting ('seed');
2. Melakukan link terminologi ke Sensus secara manual;
3. Memasukan node kedalam path ke root;
4. Menambahkan subtree dengan aturan heuristik jika banyak node dalam sebuah subtree relevan.



Gambar 2.5: Sensus Metodologi, bersumber dari [9]

- Metodologi On-To-Knowledge (OTK)

Metode ini memiliki tahapan dan proses umpan balik seperti pada gambar 2.6 ini



Gambar 2.6: On-To-Knowledge Metodologi, bersumber dari [30]

Tahapan pada OTK adalah studi kelayakan (*feasibility study*), penentuan kelanjutan (*ontology kickoff*), penyempurnaan (*refinement*), evaluasi, pemeliharaan evolusi.

Sedangkan untuk *software engineering* yang berdasarkan **IEEE Standar 1074-1995** meliputi:

1. Model Proses pada software life cycle;
2. Proses proyek management (*Planing, control dan quality management*);
3. Proses berorientasi pada pengembangan (*development*) yang dirinci dengan tahapan :
 - (a) Proses *pre-development* (studi lingkungan dan kelayakan);

- (b) Proses *development* (persyaratan/*requirement*, design, implementasi);
 - (c) Proses *post-development* (instalasi, operasi, dukungan, perawatan, keberlanjutan);
4. Proses terintegrasi (evaluasi, dokumentasi, konfigurasi dan training).

2.2 Bahasa *Ontology*

Untuk dapat digunakan, sebuah *ontology* harus diekspresikan dalam notasi yang nyata. Sebuah bahasa *ontology* adalah sebuah bahasa formal dari sebuah pembuatan *ontology*. Beberapa komponen yang menjadi struktur *ontology*, antara lain :

- *XML*

Menyediakan sintaksis untuk *output* dokumen terstruktur, tetapi belum dipaksakan untuk dokumen *XML* menggunakan *semantic constrains*.

- *XML Schema*

Bahasa untuk pembatasan struktur dari dokumen *XML*.

- *RDF*

Model data untuk objek (*'resources'*) dan relasi diantaranya, menyediakan *semantic* yang sederhana untuk model data tersebut, dan data model ini dapat disajikan dalam sintaks *XML*.

- *RDF Schema*

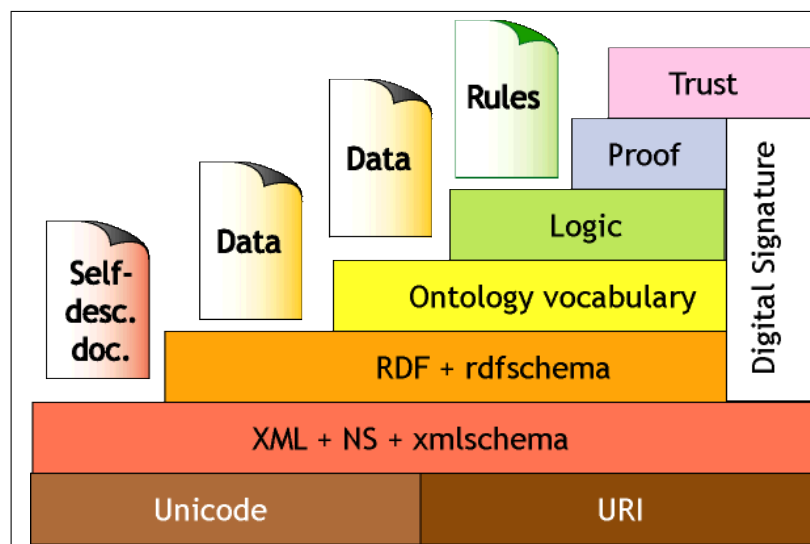
Adalah kosa kata untuk menjelaskan *properties* dan *classes* dari sumber *RDF*, dengan sebuah *semantics* untuk hirarki penyamarataan dari *properties* dan *classes*.

- *OWL*

Manambahkan beberapa kosa kata untuk menjelaskan *properties* dan *Classes*, antara lain : relasi antara *classes* (misalkan *disjointness*), kardinalitas (misalkan 'tepat satu'), *equality*, berbagai tipe dari *properties*, karakteristik dari *properties* (misalkan *symmetry*), menyebutkan satu persatu *classes*.

Berbagai bahasa yang dijelaskan di atas akan dijelaskan kembali pada section berikutnya secara terperinci.

Berbagai bahasa yang menyusun *ontology*, seperti yang telah dijelaskan di atas memiliki kedudukan tertentu dalam struktur *ontology*. Struktur layer *ontology* ditunjukkan seperti gambar 2.7. Setiap layer akan memiliki fungsi tambahan dan kompleksitas tambahan dari layer sebelumnya. Pengguna atau User yang memiliki fungsi pemrosesan layer paling rendah dapat memahami walaupun tidak seluruh *ontology* yang terletak di layer atasnya.



Gambar 2.7: Ontology Layer, bersumber dari [23]

Dalam setiap layer tersebut, masing-masing bagian memiliki fungsi masing-masing [15]:

- *XML* memiliki fungsi menyimpan isi halaman web
- *RDF* adalah layer untuk merepresentasikan semantik dari isi halaman tersebut
- *Ontology* layer untuk menjelaskan *vocabulary* dari domain
- *Logic Layer* memungkinkan untuk mengambil data yang diinginkan

2.2.1 *XML*

Extensible Markup Language (XML) adalah sebuah format text yang sederhana yang berdasarkan SGML(ISO 8879), yang didesain untuk mempermudah berbagai macam sumber informasi dalam dunia web.

2.2.1.1 Fungsi dan Tujuan *XML*

XML sudah dikenal oleh banyak orang, dan adalah dasar untuk pengembangan Software yang meningkat dengan pesat. *XML* adalah dokumen yang menyimpan data dalam struktur-struktur yang dapat berubah-ubah, hal ini berbeda dengan html yang didesain untuk dokumen *hypertext* dengan struktur yang baku. Struktur *XML* yang baik menciptakan struktur yang berbentuk hirarki terstruktur yang memiliki pasangan *tags* awal dan akhir, yang dapat terdiri dari beberapa atribut yang berpasangan. Tidak ada aturan kosakata *tags* yang baku atau pasangan *tags* yang diperbolehkan, jadi hal ini diatur di setiap aplikasi.

XML digunakan untuk beberapa alasan, antara lain :

- Menyamakan atau mempertemukan *syntax* untuk berbagai bahasa *mark-up* lain. Sebagai contoh, *Synchronized Multimedia Integration Language (SMIL)* kalau diperhatikan *syntax*-nya maka akan menyerupai *XML DTD*; karena menyediakan struktur dari dokumen SMIL.
- Memisahkan Form dengan data, sebuah XML dapat digunakan pada halaman web dengan *XLS style sheet* untuk memecah elemen-elemen yang ada pada dokumen XML dengan tepat.
- *Uniform data-exchange format*, sebuah dokumen XML dapat juga digunakan untuk mentransfer object data antara dua aplikasi.

2.2.1.2 Sintaksis dan Elemen XML

Sintak dokumen XML yang sederhana terdiri dari deklarasi XML dan elemen puncak. Deklarasi XML merupakan tempat untuk menyatakan Versi dari XML dan encoding untuk dokumen tersebut. Untuk dokumen XML standar versi yang tersedia adalah versi "1.0" dan menggunakan ISO-8859-1 (Latin-1/West European) sebagai encodingnya.

Bagian yang selanjutnya adalah elemen-elemen yang menyusun dokumen XML tersebut. Setiap elemen tersebut memiliki *tags* penutup. *Tags* dalam XML memperhatikan penggunaan huruf atau dalam arti *Case Sensitive*.

Dalam dokumen XML setiap elemen harus tersusun dengan benar, dalam arti setiap elemen harus benar-benar terkurung. sebuah dokumen XML harus memiliki elemen utama. Sedangkan setiap nilai atribut dari elemen tersebut harus menggunakan tanda kutip dua (""). Berikut ini adalah contoh sintak XML yang sederhana :

```
<?xml version="1.0"?> <note date=12/11/99>
  <to>Tove</to>}
```

```

<from>Jani</from>}
<heading>Reminder</heading>}
<body>Don't forget me this weekend!</body>}
</note>}

```

2.2.2 *RDF*

2.2.2.1 Pengertian *RDF*

Resource Description Framework (RDF) merupakan sebuah model sederhana untuk mendeskripsikan hubungan antara sumber-sumber daya yang merupakan *properties* dan *values*. *RDF properties* dapat sebagai atribut dari sebuah sumber daya. *RDF Properties* dapat merepresentasikan hubungan antara sumber daya. *RDF Data Model* dapat disusun dari sebuah diagram *entity-relationship*, tetapi tidak menyediakan mekanisme untuk mendeskripsikan *properties*-nya dan tidak dapat menyediakan mekanisme untuk menjelaskan hubungan antara *properties* tersebut dengan sumber lain. *RDF Vocabulary* menyediakan bahasa untuk mendeskripsikan *classes* dan *properties* yang dapat digunakan untuk menjelaskan *classes* dan *properties* lain.

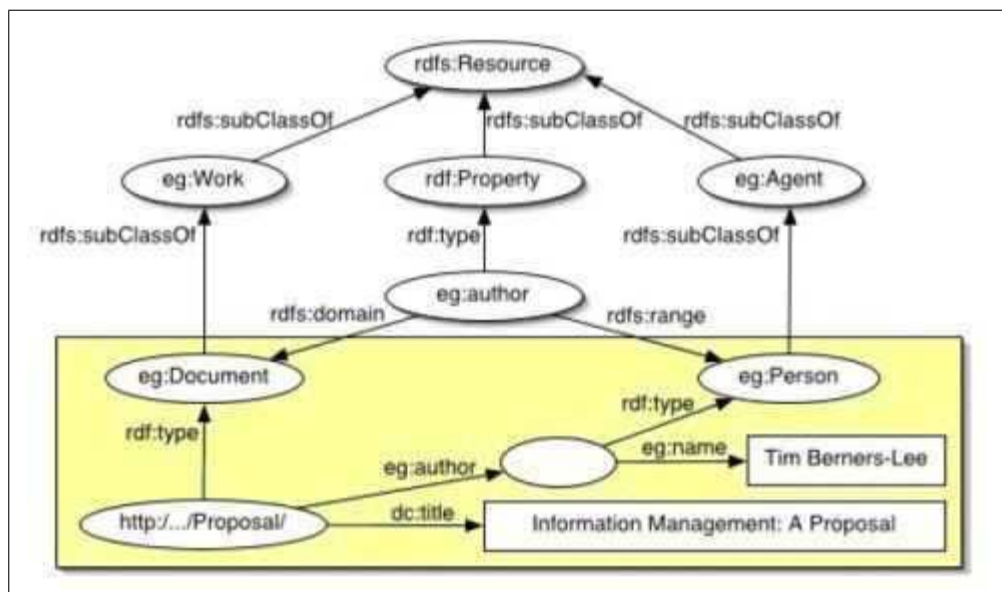
Sesungguhnya 'arti' kata menerangkan dalam *RDF* atau *RDFS* tergantung oleh beberapa faktor, termasuk peraturan sosial, bahasa natural atau penghubung ke dokumen lain. Banyak diantara arti-arti tersebut tidak dapat diakses oleh mesin.

RDF tidak memaksakan pembatasan *logic* pada *domain* dan *range* dari *properties*. Dalam praktiknya sebuah *properties* dapat diaplikasikan ke dalam dirinya sendiri. Sebagai contoh, diperbolehkan sebuah *class* 'universal' untuk menampung *class*-nya sendiri sebagai anggota, secara umum terdapat pada klasifikasi teratas.

Pendeklarasian *properties* (*atribut*) dan semantic yang berhubungan didefin-

isikan dalam konteks *RDF* adalah dalam skema *RDF*. Sebuah skema tidak hanya mendefinisikan *properties* dari sebuah sumber (contohnya: 'judul', 'pengarang', 'subjek', 'ukuran') tetapi juga menjelaskan jenis sumber daya yang sedang dijelaskan.

Kosakata *RDF* yang digunakan sebagai bahasa penggambaran sesuatu, skema *RDF*, secara khusus digunakan pada model dasar informasi pada *RDF* atau sebagai sebuah struktur grafik yang menjelaskan sumber daya dan *properties*. Semua kosakata *RDF* membagi struktur dasar yang sering digunakan, mereka menjelaskan *classes* dari sumber daya dan tipe hubungan antara sumber daya tersebut. Skema *RDF* yang merupakan kosakata penjelasan memperbolehkan perancang kosakata untuk merepresentasikan *classes* dan *properties* dalam *World Wide Web*. Contoh di bawah ini menggambarkan penggunaan kosakata skema *RDF* untuk menjelaskan *classes* dan *properties*, dan hubungan ke data pada tingkat aplikasi.



Gambar 2.8: Contoh Skema *RDFS*, bersumber dari [27]

Contoh di atas menggambarkan bagaimana *RDF* dapat digunakan un-

tuk mendeskripsikan sesuatu yang nyata, termasuk *classes* mana mereka, dan *properties* yang digunakan untuk menghubungkan anggota dari *classes* tersebut.

Bahasa yang digunakan merupakan koleksi dari sumber *RDF* yang dapat digunakan untuk mendeskripsikan *properties* dari sumber *RDF* yang lain yang mendefinisikan kosakata *RDF* untuk spesifikasi aplikasi. Kosakata inti yang didefinisikan dalam namespace dikenal sebagai '*rdfs*', dan diidentifikasi dengan referensi URI '<http://www.w3.org/2000/01/rdf-schema#>'. Spesifikasi ini juga menggunakan prefik '*rdf*' untuk merujuk ke namespace inti dari *RDF* '<http://www.w3.org/1999/02/22-rdf-syntax-ns#>'.

Sistem *class* dan *properties* pada skema *RDF* hampir sama dengan bahasa pemrograman tipe *object oriented programming* seperti Java. Tetapi ada beberapa perbedaan antara *RDF* dengan bahasa pemrograman tersebut yaitu dalam mendefinisikan *class* dalam sebuah *property*. Sebuah skema *RDF* akan mendefinisikan *properties* dalam *class* mana dia diaplikasikan. Hal tersebut adalah tugas dari *rdfs:domain* dan *rdfs:range*. Sebagai contoh, kita akan mendefinisikan *property* "*author*" akan memiliki *domain* "*Document*" dan sebuah *range* "*Person*", dimana dalam sistem *Object Oriented* akan didefinisikan sebuah *class* "*book*" dengan sebuah atribut yang dinamakan "*author*" dari tipe "*Person*". Jika menggunakan pendekatan *RDF*, sangatlah mudah untuk menambahkan *property* tambahan dengan *domain* dari dokumen atau *range* dari "*Person*". Hal tersebut dapat dilakukan dengan tanpa mendefinisikan ulang deskripsi *original* dari *class* tersebut.

Skema *RDF* dapat mendeskripsikan sebuah hubungan antara kosa kata dari skema yang tidak saling berhubungan. Sejak referensi URI digunakan untuk mendefinisikan *class* dan *properties* pada *Web*, sangatlah mungkin untuk

menciptakan *properties* baru yang mempunyai nilai dari *domain* dan *range* adalah sebuah class yang didefinisikan dari *namespace* lain.

Table di bawah ini adalah sebagian dari kosa kata dasar dalam *RDF*, digunakan bersama kosakata-kosakata tambahan dari model *RDF* dan sintaksis khusus untuk *class* dan *properties*.

Tabel 2.1: *Property*

Nama <i>Property</i>	Pengertian
<i>rdfs:Resource</i>	<i>Class</i> sumber daya
<i>rdfs:Class</i>	Konsep <i>Class</i>
<i>rdf:Property</i>	Konsep dari <i>property</i>
<i>rdfs:Literal</i>	Merepresentasikan bagian dari nilai literal
<i>rdf:Statement</i>	<i>Class</i> dari statement <i>RDF</i>
<i>rdfs:Container</i>	Merepresentasikan penyimpanan
<i>rdf:Bag</i>	Koleksi yang tidak terstruktur
<i>rdf:Seq</i>	Koleksi yang terstruktur
<i>rdf:Alt</i>	Koleksi alternative

Tabel 2.2: *Classes*

Nama <i>Property</i>	Pengertian	Domain	Range
<i>rdfs:isDefinedBy</i>	Nama space dari sumber	<i>rdfs:Resource</i>	<i>rdfs:Resource</i>
<i>rdf:subject</i>	Subjek dari statement <i>RDF</i>	<i>rdf:Statement</i>	<i>rdfs:Resource</i>
<i>rdf:predicate</i>	Prediket dari statement <i>RDF</i>	<i>rdf:Statement</i>	<i>rdf:Property</i>

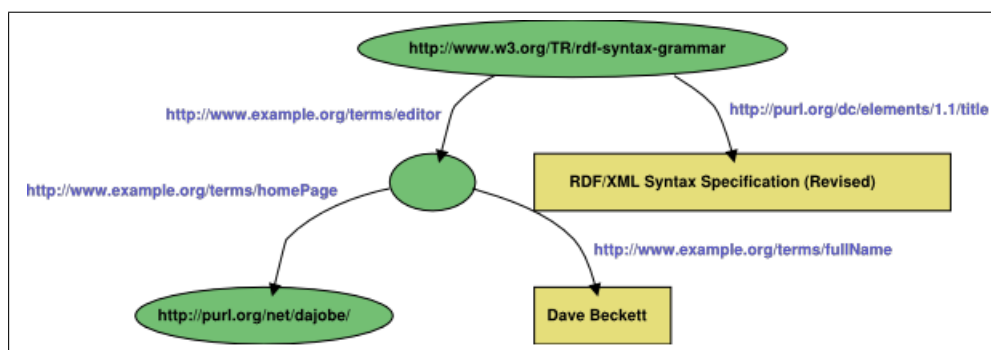
2.2.2.2 Sintaksis *XML* untuk *RDF*

Untuk menterjemahkan grafik *RDF* ke dalam *XML*. *Nodes* dan *predicate* harus direpresentasikan dalam istilah *XML* termasuk nama elemen, nama atribut, isi elemen dan nilai atribut. *RDF/XML* menggunakan *Qnames XML*

sebagai pendefinisian *namespace* dalam *XML* untuk merepresentasikan URI *RDF*. Semua *QNames* memiliki nama *namespace* yang merupakan referensi URI dan sebuah nama local.

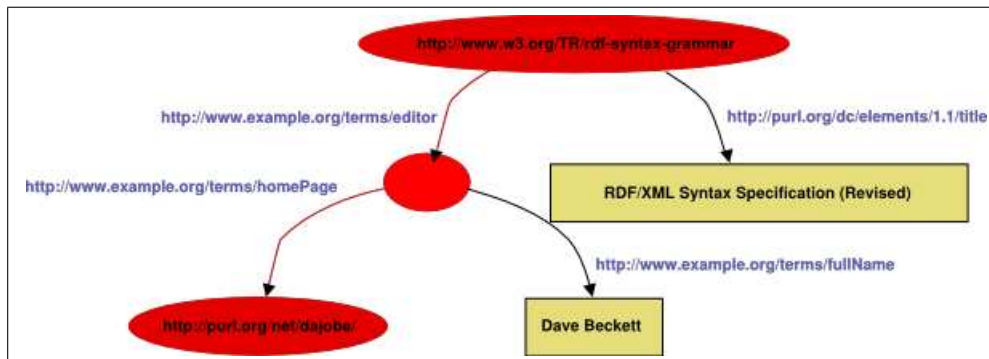
Sebuah graph dapat berupa sebuah koleksi jalur dengan bentuk *nodes*, *predicate arcs*, *nodes*, *predicate arcs*,... Node yang sudah termasuk keseluruhan *graph*. Dalam *RDF/XML* hal tersebut berubah menjadi sesuatu yang berulang-ulang dari elemen dalam elemen yang merupakan *alternative* antara elemen untuk *nodes* dan *predicate arcs*. *Nodes* yang berada di urutan pertama dari perulangan menjadi elemen yang terluar, *predicate arcs* selanjutnya menjadi elemen anak, dan seterusnya. Secara umum dimulai dari atas dari sebuah dokumen *RDF/XML* dan selalu dimulai dengan *nodes*.

Berikut ini adalah beberapa contoh graph *RDF/XML*:



Gambar 2.9: Contoh Graph *RDFS*, bersumber dari[28]

Dalam graph *RDF* di atas *nodes* digambarkan dengan oval dan menampung referensi *RDF* URI, semua *predicate arcs* diberi label dengan *RDF* URI references dan *nodes plain literal* ditulis dalam persegi. Jika kita mengikuti jalur satu *node*, *predicate arcs* ..., node maka gambar graph akan seperti contoh di bawah ini :



Gambar 2.10: Contoh Graph Satu Jalur, bersumber dari [28]

Bagian graph sebelah kiri menggambarkan *nodes/predicate* dengan ketentuan :

1. *Node* dengan referensi *RDF* URI *http://www.w3.org/TR/rdf-syntax-grammar*
2. *Predicate arcs* dengan label referensi *RDF* /URI '*http://example.org/terms/editor*'
3. *Node* dengan tanpa referensi *RDF* /URI
4. *Predicate arcs* dengan label referensi *RDF* /URI '*http://example.org/terms/homepage*'
5. *Node* dengan referensi *RDF* /URI '*http://purl.org/net/dajobe/*'

dalam *RDF/XML*, urutan dari 5 *nodes* dan *predicate arcs* yang terdapat pada sebelah kiri dari gambar 2.10 bertanggung jawab pada penggunaan lima elemen *XML* dari dua tipe, untuk *graph nodes* dan *predicate arcs*. Secara tradisional hal tersebut dikenal dengan nama "*element nodes*" dan "*property element*". Dalam garis yang ditunjukkan pada gambar 2.9, *rdf:Description* adalah elemen *nodes* (digunakan tiga kali untuk tiga *nodes*) dan *ex:editor* dan *ex:homepage* adalah dua *element property*. Berikut ini adalah contoh *RDF/XML* (*nodes* dan *predicate arcs*):

```

<rdf:Description>
  <ex:editor>
    <rdf:Description>
      <ex:homePage>
        <rdf:Description>
          </rdf:Description>
        </ex:homePage>
      </rdf:Description>
    </ex:editor>
  </rdf:Description>

```

2.2.2.3 Query *RDF*

Sebuah grafik *RDF* terdiri dari tiga serangkai, tiga rangkai itu merupakan Subject, Prediket, dan Object. Tiga rangkaian itu dapat diambil dari berbagai sumber, mereka dapat bersumber dari sebuah dokumen *RDF* atau merupakan suatu ekspresi dari penyimpanan data dalam format lain, seperti *XML* atau Database Relational.

Untuk mendapat sebuah informasi dari sebuah grafik *RDF* diperlukan sebuah bahasa query. Query yang cocok untuk masalah ini adalah SPARQL, SPARQL menyediakan fasilitas untuk :

- Mengambil informasi dari URI, *Blank Nodes, Plain*;
- Memecah *subgraph* dari *RDF*;
- Membangun sebuah grafik *RDF* berdasarkan informasi yang didapat dari hasil eksekusi query.

Bahasa query SPARQL mencocokkan struktur grafik. Struktur grafik

termudah adalah struktur *triple*, yang menyerupai *RDF Triple*, tetapi dengan variabel yang mungkin untuk masing-masing komponen triple tersebut. Menggabungkan *triple* struktur tersebut akan memberikan pola dasar dari grafik tersebut. Berikut ini adalah contoh sederhana dari SPARQL :

Data:

```
<http://example.org/people/people1>
<http://purl.org/dc/element/1.1/name> "Doni Kusuma".
```

Query:

```
SELECT?name
WHERE
{
  <http://example.org/people/people1>
  <http://purl.org/dc/element/1.1/name> ?name.
}
```

Query Result:

name
"Doni Kusuma"

2.2.3 OWL

2.2.3.1 Pengertian OWL

OWL adalah bahasa *ontology* yang baru untuk sebuah *semantic web*, dikembangkan oleh *World Wide Web Consortium* (W3C) kelompok kerja *Web Ontology*. Pada mulanya *OWL* didesain untuk merepresentasikan informasi tentang katagori dari sebuah objek dan bagaimana objek tersebut berhubungan. *OWL* dapat juga menyediakan informasi tentang objek itu sendiri.

Sebagai hasil usaha yang dilakukan oleh kegiatan *Semantic Web W3C*, *OWL* harus dapat cocok dengan visi *Semantic Web* yaitu bahasa stack bersama-

sama dengan *XML* dan *RDF*. *OWL* yang diharapkan menjadi salah satu bahasa *ontology*, harus dapat merepresentasikan bagian-bagian yang berguna dalam sebuah *ontology*.

Dalam usahanya untuk mendukung kemampuan dan skenario yang telah disepakati, *OWL* mengambil kemampuan *RDF* untuk penjabaran statis dan kemampuan struktur *class* dan *property* dari skema *RDF* dan menyisipkan mereka ke dalamnya. *OWL* dapat mendeklarasikan *class*, dan mengorganisasikan *class* tersebut ke dalam hirarki *sub-class*, sama seperti skema *RDF*. *Class* *OWL* dapat dijelaskan sebagai kombinasi logikal (irisan, gabungan, komplemen) dari *class* lainya, atau sebagai penjelasan satu-persatu dari objek yang dimaksud, melebihi kemampuan skema *RDF*. *OWL* dapat juga mendeklarasikan *property*, mengorganisasikan *property* tersebut ke dalam hirarki "*subproperty*", dan menyediakan *domain* dan *range* untuk *property* tersebut, seperti pada skema *RDF*. *Domain* dari *property* *OWL* adalah *class* dalam *OWL*, dan *range* dapat berupa *class* dalam *OWL* atau tipe data yang dideklarasikan dari luar seperti *string* atau *integer*. *OWL* dapat menetapkan bahwa *property* tersebut adalah *transitive*, *asymmetric*, *functional*, atau bertolak belakang dengan *property* lainnya.

OWL dapat mengekspresikan objek (dapat juga disebut '*individu*') mana yang dimiliki oleh *class* yang mana, dan apa nilai *property* untuk sebuah individu. Pernyataan yang sama dapat dibuat pada *class* dan *property*, pernyataan *disjoint* dapat dibuat pada *class*, dan *equality* dan *inequality* dapat juga disisipkan antara individu.

Kemampuan *OWL* yang lebih dari *RDF*, adalah kemampuannya untuk menyediakan pembatasan pada bagaimana posisi *property* terhadap *class*. *OWL* dapat mendefinisikan *class* mana yang mempunyai *property* terbatas yang

membuat semua nilai untuk *property* tersebut maka semua nilai untuk *property* dalam *instances* harus dimiliki oleh *class* tertentu (atau tipe data); setidaknya satu nilai harus datang dari *class* tertentu (atau tipe data); setidaknya terdapat nilai tertentu; dan harus setidaknya satu atau lebih angka tertentu yang berbeda dari sebuah nilai. Sebagai contoh, dengan menggunakan skema *RDF* kita dapat:

- Mendeklarasikan *class*, seperti 'negara', 'orang', 'mobil';
- Menyatakan bahwa 'murid' adalah *sub-class* dari 'orang';
- Menyatakan bahwa 'Indonesia' dan 'Jerman' adalah anggota *class* 'negara';
- Mendeklarasikan 'bangsa' sebagai *property* yang menghubungkan *class* 'orang' (sebagai *domain*) dan *class* 'negara' (sebagai *range*);
- Menyatakan bahwa 'umur' adalah *property*, dengan 'orang' sebagai *domain* dan *integer* sebagai *range*;
- Menyatakan bahwa 'Budi' sebagai anggota dari *class* 'Indonesia', dan 'umur' yang dimilikinya memiliki nilai '48'.

Dengan *OWL* dapat ditambahkan:

- Menyatakan bahwa 'negara' dan 'orang' adalah *class* yang *disjoint*;
- Menyatakan bahwa 'Canada' dan 'Jerman' adalah *individu* yang berbeda;
- Mendeklarasikan 'memiliki warga' sebagai kebalikan dari *property* 'bangsa';

- Menyatakan bahwa *class* 'tidak ada negara' dibuat untuk semua anggota dari *class* 'orang' yang tidak memiliki nilai untuk *property* 'bangsa'.

OWL mempunyai suatu penukar sintaksis *RDF/XML* dan suatu abstrak sintaksis seperti *frame*, dan itu telah memiliki tiga nama *sub languages*. Keanekaragaman ini adalah hasil berusaha yang langsung untuk mencukupi sejumlah besar kebutuhan yang berlawanan dan pengaruh.

2.2.3.2 Sub Bahasa *OWL*

OWL menyediakan tiga sub bahasa yang berbeda tingkatan bahasanya yang dirancang untuk berbagai kebutuhan tertentu dari pengguna, antara lain :

- *OWL Lite*

Mendukung bagi pengguna yang memerlukan hirarki klasifikasi dan batasan yang sederhana. Sebagai contoh, hanya mendukung batasan *Cardinality*, dengan nilai untuk *Cardinality* sebesar 0 atau 1. Haruslah lebih sederhana untuk menyediakan alat pendukung untuk *OWL Lite* dibanding yang lebih ekspresif, dan *OWL Lite* menyediakan suatu alur migrasi cepat untuk thesauri dan taksonomi lain. *OWL Lite* juga mempunyai suatu kompleksitas formal yang lebih rendah dibanding *OWL DL*. *OWL Lite* didesain untuk kemudahan penerapan dan untuk memberikan user dengan fungsi subset yang akan membawa mereka ke permulaan dalam penggunaan *OWL*.

- *OWL DL*

Mendukung pengguna yang menginginkan ekspresi yang maksimum disaat menahan perhitungan yang lengkap (semua kesimpulan dijamin menjadi dapat diperhitungkan) dan ketepatan (semua perhitungan akan diselesaikan waktu tertentu). *OWL DL* meliputi semua bahasa konstruksi

dalam *OWL*, tetapi mereka dapat digunakan hanya di bawah batasan tertentu (sebagai contoh, selama suatu *class* menjadi suatu *subclass* dari banyak kelas, suatu kelas tidak bisa menjadi suatu *instance* dari kelas yang lain). *OWL DL* di beri nama dalam kaitan dengan *Description Logic*, suatu bidang riset yang telah belajar logika yang membentuk pondasi bagi yang formal pada *OWL*. *OWL DL* memang dirancang untuk mendukung deskripsi logikal dari segmen bisnis dan untuk menyediakan bahasa subset yang memberikan *property* untuk perhitungan dalam sebuah sistem.

- *OWL Full*

Sangat berguna untuk pengguna yang menginginkan ekspresi maksimum dan kebebasan sintaksis dari *RDF* tanpa ada jaminan perhitungan. Sebagai contoh, dalam *OWL Full* sebuah class dapat diperlakukan berturut-turut sebagai kumpulan individu dan sebagai individu dengan haknya sendiri. *OWL Full* memperbolehkan ontologi untuk meningkatkan arti dari kosa kata yang belum digambarkan (*RDF* atau *OWL*). Tidaklah mungkin untuk semua pembuat *software* untuk mendukung secara penuh kemampuan yang ada pada *OWL Full*.

OWL Full dan *OWL DL* menyediakan konstruksi bahasa *OWL* yang sama. Perbedaan mereka hanya terletak pada pembatasan dalam penggunaan fungsi dan dalam penggunaan fungsi dalam *RDF*. *OWL Full* memperbolehkan pencampuran dari *OWL* dengan skema *RDF* dan, seperti skema *RDF*, tidak memaksakan peraturan yang memisahkan *classes*, *properties*, *individuals*, dan nilai data. *OWL DL* memberikan pembatasan dalam penggabungan dengan *RDF* dan membutuhkan *disjoint* dari *classes*, *properties*, *individuals*, dan nilai data. *OWL Lite* adalah sub bahasa dari *OWL DL* yang hanya mendukung

subset dari konstruksi bahasa *OWL*. *OWL Lite* secara umum ditujukan sebagai alat untuk membangun sebuah *ontology* bagi orang-orang yang ingin menggunakan *OWL*, tetapi ingin memulainya dengan struktur atau fungsi bahasa yang sederhana.

Pengembang *ontology* yang mengadopsi *OWL* harus menentukan sub bahasa mana yang tepat untuk kebutuhannya. Pemilihan antara *OWL Lite* dan *OWL DL* tergantung kebutuhan dalam penggunaannya oleh user yang membutuhkan konstruksi yang lebih ekspresif di sediakan oleh *OWL DL*. Pemilihan antara *OWL DL* dan *OWL Full* tergantung juga kepada kebutuhan dalam penggunaannya yang user membutuhkan fasilitas *meta-modeling* dari skema *RDF* (contohnya, mendefinisikan *class* dari *class*, atau mengkaitkan *properties* dengan *class*).

OWL Full dapat dipandang sebagai ekstensi dari *RDF*, *OWL Lite* dan *OWL DL* dapat dipandang sebagai ekstensi dari tampilan yang terbatas dari *RDF*. Setiap dokumentasi *OWL (Lite, DL Full)* adalah dokumen *RDF*, dan setiap dokumen *RDF* adalah dokumen *OWL Full*, tetapi hanya sebagian dokumen *RDF* akan menjadi dokumen *OWL Lite* atau *OWL DL*. Karena hal tersebut, perhatian yang lebih harus diberikan user yang ingin bermigrasi dari dokumen *RDF* ke *OWL*.

2.2.3.3 Rancangan *OWL*

Untuk berbagai alasan, pendeskripsian dalam bagian proses, terdapat dua tipe penggunaan *OWL*. Tipe yang pertama, bagian dari *OWL DL* dan *OWL Lite*, hanya konstruksi tertentu yang diperbolehkan untuk dideskripsikan, dan konstruksi tersebut hanya dapat dikombinasikan dengan cara-cara tertentu. Keunggulan dalam pembatasan ini meliputi menentukan kesimpulan dan kemungkinan berpikir dari *OWL* di dalam suatu lingkungan yang standar, uta-

manya sebagai suatu ekspresi *Description Logic*. Di dalam tipe yang kedua, yang terdapat dalam *OWL Full*, semua grafik *RDF* diijinkan. Kelebihan tipe ini meliputi total peningkatan kecocokan dengan *RDF* dan suatu ekspresif yang lebih besar.

Walaupun versi terbatas dari *OWL* memiliki beberapa perbedaan dari standar *Description Logic*. Perbedaan ini memindahkan versi *OWL* tersebut dari dunia *Description Logic* ke dunia *Semantic Web*. Beberapa penjelasan tentang *OWL* :

- *OWL* menggunakan referensi URI sebagai nama, dan membangun URI tersebut menggunakan kondisi yang sama yang digunakan dalam *RDF*. Alasan inilah dalam *OWL* untuk menggunakan nama yang singkat untuk referensi URI, contohnya menggunakan '*owl:Thing*' untuk referensi URI;
- *OWL* mengumpulkan semua informasi ke dalam *ontology*, yang secara umum disimpan sebagai dokumen Web yang ditulis dalam *RDF/XML*. *Ontology* dapat memasukan *ontology* lain, menambahkan informasi dari *ontology* lain ke dalam *ontology* sendiri;
- Walaupun tipe *DL/Lite* dalam menggunakan *OWL* memperbolehkan catatan *property RDF* digunakan untuk menyisipkan informasi ke dalam *class*, *properties*, dan *ontology*, seperti '*owl:DeprecatedClass*'. Penyisipan ini adalah *RDF triples*, dan hal tersebut digunakan dalam *semantic* yang lebih besar. Mereka tidak dapat diperlakukan sebagai komentar tanpa ada arti sesungguhnya. Hal ini memecahkan perbedaan dalam *Description Logic* antara *individual* dan *classes* dan *properties*;
- *OWL* menggunakan fasilitas dari tipe data *RDF* dan skema tipe data *XML* untuk menyediakan tipe data dan nilai data;

- Versi DL dan Lite dari *OWL* memiliki frame seperti sintaksis abstrak, walaupun *RDF/XML* adalah sintaksis pengganti yang resmi untuk semua *OWL*.

2.2.3.4 Sintaksis *OWL*

Sebuah *OWL ontology* adalah sebuah grafik *RDF*, yang berbentuk set dari *RDF Triples*. Seperti grafik *RDF*, grafik *OWL ontology* dapat ditulis dalam berbagai macam bentuk sintaksis yang berbeda. Arti dari *OWL ontology* dapat digambarkan dengan menggunakan grafik *RDF*. Tetapi dapat dimungkinkan untuk menggunakan berbagai bentuk sintaksis *RDF/XML* yang berbeda, selama hasil yang dikeluarkan memiliki kecocokan dengan set dari *RDF Triples*. Berikut ini adalah sedikit contoh sintaksis dalam *OWL* yang memiliki arti sama dengan yang dibuat dengan menggunakan *RDF Triples* :

```
<owl:class rdf:ID="Continent"/>
```

Sintak dalam *RDF/XML* :

```
<rdf:Description rdf:about="#Continent">
<rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
<rdf:Description>
```

Jika kedua coding tersebut di encode, maka akan dibangkitkan arti yang sama. Contoh sederhana dokumen *OWL* adalah :

```
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#Eurasia"/>
```

```

    <owl:Thing rdf:about="#Africa"/>
    <owl:Thing rdf:about="#NorthAmerica"/>
    <owl:Thing rdf:about="#SouthAmerica"/>
    <owl:Thing rdf:about="#Australia"/>
    <owl:Thing rdf:about="#Antarctica"/>
  </owl:oneOf>
</owl:Class>

```

2.3 Web *Semantic*

2.3.1 Definisi

Web *Semantic* adalah sekumpulan informasi yang dikumpulkan dengan metode tertentu agar dapat dengan mudah diproses oleh mesin, dalam skala yang besar. Ini seperti cara yang efisien dari representasi data pada *World Wide Web*, atau sebagai database global yang saling terhubung.

Web *Semantic* dikembangkan oleh sebuah tim di *World Wide Web consortium*. Hingga saat ini Web *Semantic* masih dalam tahap pengembangan dan penyempurnaan, karena teknologi ini masih baru digunakan dan tim masih mengembangkan metode masing-masing untuk mengembangkan Web *Semantic* [18].

Kemudian apakah keuntungan menggunakan Web *Semantic*? Dengan metode tradisional data-data disimpan pada halaman web tersebut sangat beragam. Sehingga ini masih mungkin digunakan untuk skala pemakai terbatas. Tetapi jika akan digunakan dalam skala yang luas maka akan menjadi kesulitan, karena tidak ada sistem yang global yang dapat digunakan untuk merepresentasikan data dengan cara tersebut yang dapat di proses oleh setiap pemakai. Sebagai contoh ada informasi mengenai olah raga, cuaca, dan lain-

lain, kesemua informasi tersebut masing-masing jumlahnya jutaan dan dibuat oleh pembuat yang berbeda-beda, yang masing-masing memiliki bahasa dan metode tersendiri untuk menyimpan informasi tersebut dan kesemua informasi tersebut ditampilkan dalam halaman HTML, Hal tersebut sangat sulit dilakukan kalau menggunakan metode tradisional.

Seperti halaman web biasa yang memiliki service seperti mesin pencari, yang menggabungkan berbagai macam halaman kedalam satu koleksi yang sama. Web *Semantic* juga memiliki hal yang sama, perbedaanya terletak pada metode pencarian halaman web yang diinginkan. Jika pada halaman web biasa hanya hanya dapat mencari halaman web yang memiliki sebuah atau beberapa kata yang menjadi bahan pencarian, sedangkan dalam Web *Semantic* dapat melakukan pencarian dengan lebih terstruktur, pertanyaan yang spesifik (selama hal tersebut di tulis kedalam bentuk yang dimengerti oleh mesin).

Web *Semantic* tidak hanya tentang bagaimana mengajarkan mesin untuk dapat mengerti bahasa manusia atau memproses bahasa alami dan juga tidak semata-mata untuk membuat sebuah *Artificial Intelligence*, tetapi tujuan utama adalah untuk mempermudah mengumpulkan data-data, lebih diutamakan untuk data yang besar.

2.3.2 *Komponen Web Semantic*

Sebuah web *Semantic* tidak berdiri sendiri, terdiri dari berbagai macam komponen-komponen yang saling berhubungan satu sama lainnya. Komponen yang terdapat dalam sebuah web *semantic* antara lain [32]:

1. *XML*, menyediakan sintaksis untuk dokumen yang terstruktur;
2. *XML Schema*, adalah bahasa untuk membatasi struktur dari dokumen

XML;

3. *RDF*, model data sederhana yang berhubungan dengan object ("resource") dan bagaimana mereka berhubungan. Sebuah model data *RDF* dapat ditulis dengan sintaksis *XML*;
4. Skema *RDF*, adalah vocabulary untuk mendeskripsikan *property* dan *class* dari *RDF*;
5. *OWL*, menambahkan beberapa kosa kata untuk menjelaskan *property* dan *class*, antara lain: hubungan antara *class*, kardinalitas, persamaan, karakteristik dari *property*.

2.3.3 Fungsi Web *Semantic*

Seperti telah dijelaskan di atas bahwa tujuan utama dalam penerapan web *Semantic* adalah untuk menemukan informasi yang tepat dan cepat dalam kumpulan informasi yang tersebar luas dalam dunia internet. Dengan melihat tujuan tersebut maka web *semantic* lebih tepat untuk penggunaan di dalam perusahaan yang biasanya membutuhkan informasi dalam waktu yang cepat, dan informasi tersebut mengambil referensi dari banyak sumber.

Dalam sebuah perusahaan web *semantic* dapat digunakan untuk [6]:

- *Decision Support*
- *Business Development*;
- *Information Sharing and Knowledge*;
- *Administration and Automation*.

Bab 3

Perancangan

3.1 Taksonomi Tumbuhan di Pulau Jawa

Setiap spesies tumbuhan terdiri dari sejumlah individu sehingga seluruh spesies terdiri dari berjuta-juta individu. Antara satu spesies dengan spesies yang lain memiliki sejumlah perbedaan, antara lain ukuran, umur, bentuk tubuh, pola warna, dan jenis kelamin. Karena perbedaan dan keanekaragaman tersebut dibuatlah suatu sistem klasifikasi makhluk hidup. Pada sistem ini makhluk hidup dibedakan atau dikelompokkan secara sistematis dan bertahap.

Mahluk hidup terutama tumbuhan dikelompokkan berdasarkan persamaan dan perbedaan cirinya. Klasifikasi tersebut secara umum terdiri dari tiga jenis, yaitu klasifikasi alami yaitu pengelompokan makhluk hidup berdasarkan ciri morfologi, anatomi, dan fisiologi makhluk tersebut. Klasifikasi kedua adalah klasifikasi filogeni yang memperhatikan sejarah evolusi suatu makhluk hidup. Klasifikasi buatan adalah pengelompokan yang berdasarkan ciri yang mudah dilihat.

Jenjang taksonomi menunjukkan bahwa setiap kelompok besar makhluk hidup terdiri dari kelompok kecil makhluk hidup, dengan kata lain, kelompok kecil makhluk hidup dengan kesamaan ciri tertentu membentuk kelompok makhluk hidup yang besar.

Dalam dunia tumbuhan, ada dua pengelompokan besar, kelompok yang pertama adalah tumbuhan tidak berpembuluh atau non-Tracheophyta dan kelompok yang kedua adalah kelompok tumbuhan berpembuluh atau Tracheophyta. Perbedaan antara kedua kelompok tersebut hanyalah pada perbedaan ada atau tidaknya organ pengangkut bahan makan yang biasanya terletak pada batang tumbuhan.

Pada tumbuhan tidak berpembuluh organ pengangkutnya hanyalah sebuah cairan, biasanya tumbuhan yang tidak memiliki pembuluh adalah tumbuhan rendah atau tumbuhan yang memiliki ukuran yang mikro atau kecil, hanya sebagian kecil dari tumbuhan tidak berpembuluh yang dapat dilihat dengan pengelihatian normal. Tetapi tumbuhan tidak berpembuluh walaupun berukuran kecil, tetapi masih disebut tumbuhan karena pada salah satu bagian tubuhnya memiliki zat hijau daun atau klorofil.

Pada tumbuhan berpembuluh memiliki organ pengangkut pada batangnya. Organ pengangkut tersebut dibagi menjadi dua kelompok atau bagian, bagian yang pertama adalah organ yang mengangkut bahan-bahan makanan dari akar ke daun. Sedangkan bagian yang satu lagi bertugas mengangkut hasil-hasil proses pemasakan yang dilakukan oleh daun untuk disebarkan ke seluruh pohon. Pada kelompok tumbuhan ini biasanya merupakan tumbuhan tingkat tinggi atau berukuran besar.

Yang termasuk kelompok tumbuhan tidak berpembuluh adalah kelompok lumut, sedangkan yang termasuk kelompok tumbuhan berpembuluh adalah tumbuhan paku-pakuan dan tumbuhan berbiji.

Informasi yang digunakan untuk penyusunan taxonomi tumbuhan diatas diambil dari sebuah buku yang diterbitkan oleh Universitas Gajah Mada [25].

3.1.1 Tumbuhan Lumut

Tumbuhan lumut merupakan suatu tumbuhan darat yang tubuhnya tidak dapat dibedakan antara akar, batang, daun. Yang paling penting adalah tumbuhan lumut tersebut tidak memiliki pembuluh pengangkut, namun memiliki klorofil sehingga masuk kedalam kerajaan tumbuhan. Ada beberapa jenis lumut yang tubuhnya masih berupa lembaran (*Talus*) dan ada yang sudah memiliki bagian tubuh yang mirip dengan akar, batang, daun. Oleh karena

itu tumbuhan lumut dianggap sebagai tumbuhan peraluan.

Tumbuhan lumut paling cocok tumbuh pada daerah yang lembab dan teduh, hampir seluruh daerah di Indonesia merupakan daerah habitat lumut yang baik.

Pengelompokan berbagai jenis lumut menghasilkan tiga kelas, yaitu lumut hati (*Hepaticopsida*), lumut tanduk (*Anthocerotopsida*), dan lumut daun (*Bryopsida*).

- Lumut hati (*Hepaticopsida*)

Lumut hati memiliki ciri tubuh berbentuk talus. Contoh dari Lumut hati adalah *Riccia nutans*, *Marchantia*, dan *Lunularia*.

- Lumut Tanduk (*Anthocerotopsida*)

Lumut tanduk memiliki tubuh seperti lumut hati yaitu berupa talus. Sel lumut tanduk hanya memiliki satu kloroplas. Yang merupakan contoh lumut tanduk adalah *Anthoceros*

- Lumut Daun (*Bryopsida*)

Lumut daun disebut juga lumut sejati karena tubuhnya berbentuk tumbuhan kecil dengan bagian akar, batang, dan daun yang dapat dibedakan dengan jelas. Lumut daun hidup berkelompok membentuk hamparan tebal seperti beludru. Yang termasuk lumut daun adalah *Polytrichum* dan *Sphagnum*.

3.1.2 Tumbuhan Berpembuluh

3.1.2.1 Tumbuhan Paku

Tumbuhan paku adalah anggota kerajaan tumbuhan yang memiliki akar, batang, daun sejati, serta memiliki pembuluh pengangkut. Ada beberapa jenis

tumbuhan paku yang tidak mempunyai akar dan daun sejati. Tumbuhan paku ada yang hidup mengapung di atas air misalnya *Azolla pinnata* dan *Marseilea crenata*. Namun umumnya tumbuhan paku adalah tumbuhan darat. Persebaran tumbuhan paku lebih melimpah pada daerah hutan hujan tropis seperti Indonesia. Tumbuhan paku diklasifikasikan berdasarkan ciri tubuhnya menjadi empat subdivisi yaitu :

- Paku Purba
- Paku Kawat
- Paku Ekor Kuda
- Paku Sejati

3.1.2.2 Tumbuhan Berbiji

Tumbuhan berbiji atau *Spermatophyta* merupakan kelompok tumbuhan yang mempunyai ciri khas, yaitu adanya suatu organ yang berupa biji. Biji merupakan bagian yang berasal dari bakal biji dan di dalamnya mengandung calon individu baru yaitu lembaga. Lembaga akan terjadi setelah terlebih dahulu terjadi peristiwa penyerbukan atau persarian yang diikuti pembuahan. Tumbuhan berbiji merupakan tumbuhan yang paling dominan dibandingkan tumbuhan lain.

Tumbuhan berbiji memiliki ukuran yang relative besar jika dibandingkan dengan jenis tumbuhan lain. Habitus atau perawakan tumbuhan berbiji sangat bervariasi, yaitu [1] pohon, misalnya jati, duku, kelapa, cemara, beringin. [2] perdu, misalnya mawar, kembang merak, kembang sepatu. [3] semak, misalnya arbei. [4] herba, misalnya sayur-sayuran, bunga krokot.

Tumbuhan berbiji kebanyakan hidup di darat. Namun ada beberapa jenis tumbuhan berbiji yang hidup mengapung di air, misalnya teratai. Divisi tumbuhan berbiji dibedakan menjadi dua subdivisi, yaitu tumbuhan berbiji terbuka (*Gymnospermae*) dan tumbuhan berbiji tertutup (*Angiospermae*).

Tumbuhan berbiji terbuka dibagi menjadi beberapa kelas, yaitu :

1. Kelas *Cycadinae*

habitus menyerupai palem, berkayu, dan daun tersusun dalam roset batang. Contohnya adalah pakis haji.

2. Kelas *Coniferae*

Anggota kelas ini mempunyai habitus semak, perdu atau pohon dengan tajuk menyerupai kerucut dan daun berbentuk jarum. Contohnya adalah damar, pinus.

3. Kelas *Gnetinae*

Anggota kelas ini merupakan tumbuhan berkayu yang batangnya bercabang atau tidak. Contoh kelas ini adalah melinjo.

4. Kelas *Ginkgoinae*

Berupa pohon yang mempunyai tunas pendek, daun bertangkai panjang dan berbentuk pasak atau kipas, dengan tulang daun yang bercabang-cabang menggarpu. Contoh tumbuhan dari kelas ini adalah Ginkgo biloba.

Subdivisi tumbuhan berbiji tertutup dibagi kedalam dua kelas, yaitu :

1. Kelas *Dikotil*

Tumbuhan yang termasuk kedalam kelas ini meliputi tumbuhan yang mempunyai habitus herba, semak, perdu, maupun pohon. Yang merupakan contoh tumbuhan dikotil adalah cemara laut, nangka, lada, kenanga.

2. Kelas *Monokotil*

Anggota tumbuhan yang termasuk golongan tumbuhan monokotil adalah tumbuhan yang memiliki habitus herba, semak, perdu, atau pohon. Contoh dari tumbuhan monokotil adalah padi, tebu, kunyit, eceng gondok, anggrek.

3.1.3 Pendeskripsian

Dalam *ontology* ini *terminology* yang akan dibahas adalah yang berhubungan dengan taksonomi tumbuhan termasuk habitat alaminya, habitat tersebut dibatasi hanya di daerah pulau jawa. Tetapi untuk Domain habitat untuk tumbuhan hanya dibagi menjadi dua, yaitu daratan dan permukaan air. Taksonomi tumbuhan tersebut dimulai dari kerajaan tumbuhan sebagai *superclass* atau tingkatan hirarki tertinggi, sampai subdivisi-subdivisi dari masing-masing kelas. Setiap subdivisi-subdivisi tersebut ditambah dengan beberapa contoh dan masing-masing habitat. Sebagai tambahan, Habitus dan Habitat tumbuhan tersebut dimasukan kedalam suatu kelas tersendiri.

Dalam taksonomi tumbuhan yang akan dimasukan ke dalam *ontology* ini dimulai kerajaan tumbuhan (*plantae*), pada kerajaan ini yang dimasukan hanya yang memiliki klorofil di dalam anggota tubuhnya. Dalam kerajaan tersebut dibagi menjadi dua bagian, yaitu tumbuhan berpembuluh dan tumbuhan tidak berpembuluh. Dari kelompok tumbuhan tidak berpembuluh hanya ada satu bagian yaitu kelompok Lumut, sedangkan untuk tumbuhan

berpembuluh dibagi menjadi dua kelompok yaitu tumbuhan paku dan tumbuhan berbiji.

Pada kelompok tumbuhan lumut, dibagi menjadi tiga kelas yaitu lumut hati, lumut tanduk, lumut daun. Masing-masing kelompok tersebut memiliki nama-nama tumbuhan yang masuk kedalam kelas tersebut, berikut dengan habitat alaminya.

Pada kelompok tumbuhan paku yang merupakan bagian dari tumbuhan berpembuluh memiliki empat subdivisi, yaitu Paku purba, Paku kawat, Paku ekor kuda, Paku sejati. Masing-masing subdivisi tersebut memiliki contoh tumbuhan yang termasuk ke dalam tersebut, serta memiliki herba atau bentuk tumbuhan dan memiliki habitat alaminya.

Dari kelompok tumbuhan berbiji dibagi menjadi dua subdivisi, yaitu tumbuhan berbiji terbuka dan tumbuhan berbiji tertutup. Pada tumbuhan berbiji terbuka terdapat empat kelas Kelas *Cycadinae*, Kelas *Coniferae*, Kelas *Gnetinae*, Kelas *Ginkgoinae*. Sedangkan pada tumbuhan berbiji tertutup terdapat dua kelas, yaitu monokotil dan dikotil. Pada kelompok tumbuhan berbiji tipe herba atau perawakan dari jenis tumbuhan tersebut memiliki beragam tipe, yaitu semak, pohon, perdu, herba.

Dari ruang lingkup di atas, yang termasuk ke dalam dalam *ontology* ini adalah :

- Tumbuhan
- Tumbuhan Lumut
- Tumbuhan Berpembuluh
- Tumbuhan Paku-pakuan
- Tumbuhan Berbiji

- Tumbuhan Berbiji Terbuka
- Tumbuhan Berbiji Tertutup
- Tumbuhan Berkeping Satu
- Tumbuhan Berkeping Dua
- Lumut Hati (*Hepaticopsida*)
- Lumut Daun (*Bryopsida*)
- Lumut Tanduk (*Anthocerotopsida*)
- Paku Kawat
- Paku Ekor Kuda
- Paku Purba
- Paku Sejati
- Habitat
- Habitus

Masing-masing di atas akan menjadi *class* tersendiri. Dan akan menjadi *subclass* dari *owl:thing* di dalam *OWL*, sedangkan jika di dalam *RDF* akan menjadi *subclass* dari *:THING*

3.2 Pendefinisian Komponen *Ontology*

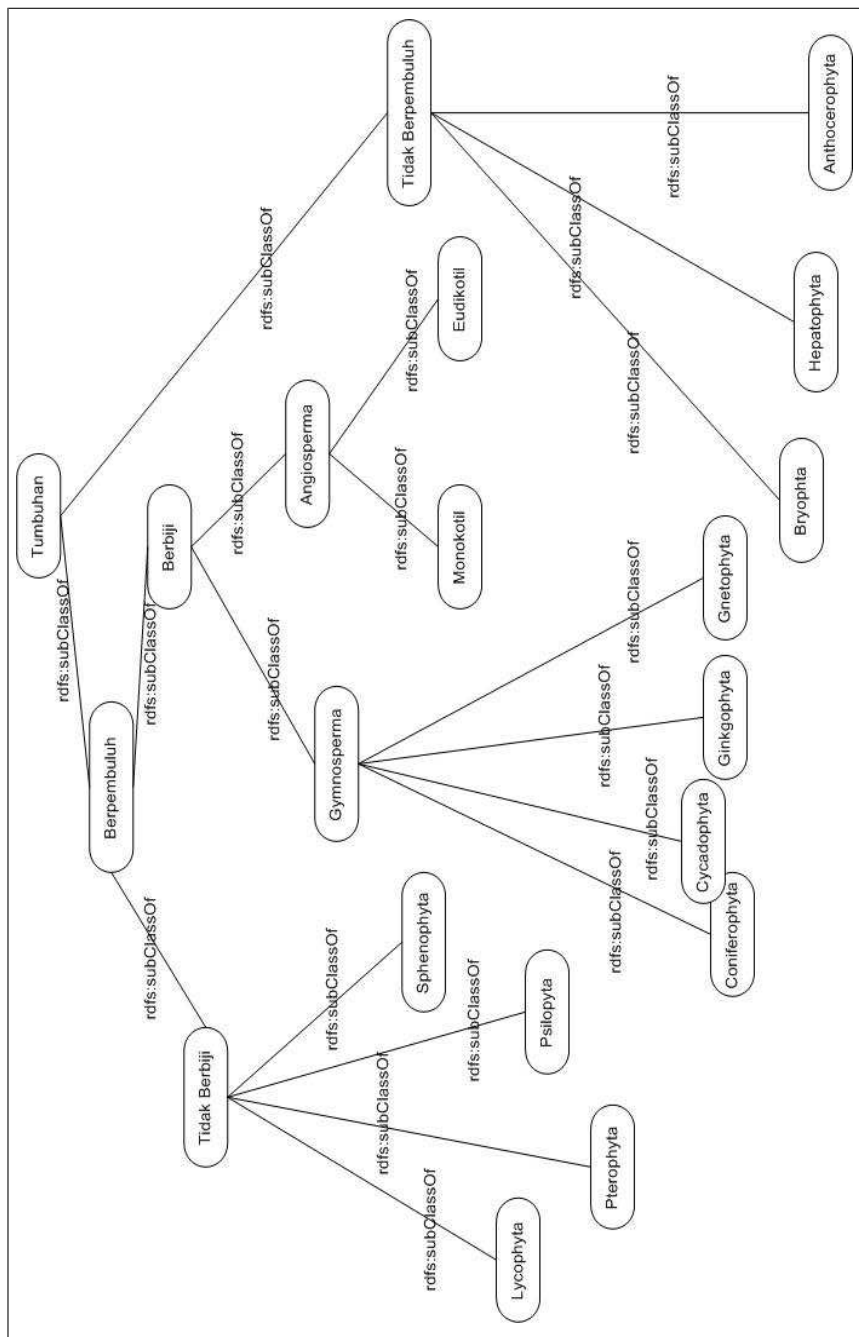
3.2.1 *Class* dan Hirarki *Class*

Dalam bab ini penentuan hirarki class menggunakan metode top-down, yaitu menentukan super class terlebih dahulu dari class-class yang telah ditentukan pada subbab sebelumnya.

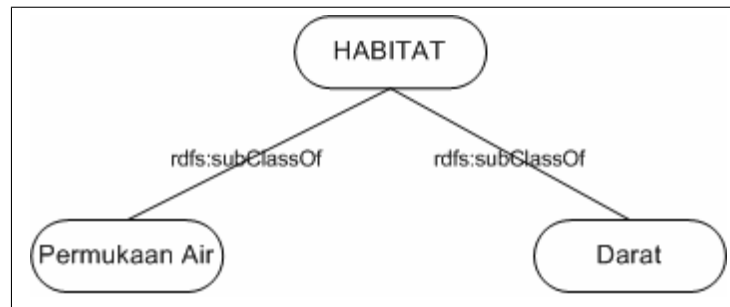
Dalam kasus ini yang menjadi super-class adalah "Tumbuhan", karena "Tumbuhan" adalah tingkatan paling tinggi dari hirarki dalam kerajaan tumbuhan. Di bawah *class* "tumbuhan" terdapat dua *class* yang masih memiliki anggota atau *subclass*, kedua *class* tersebut adalah "Berpembuluh" dan "tidak berpembuluh".

Class "tidak berpembuluh" hanya memiliki satu *subclass* yaitu "lumut", oleh karena *subclass* dari *class* "tidak berpembuluh" hanya satu yaitu "lumut" maka *class* "lumut" tidak akan ditampilkan atau hanya akan mengikuti *class* "tidak berpembuluh". Di bawah *class* "tidak berpembuluh" akan memiliki tiga *subclass* yaitu "*Anthocerotopsida*", "*Bryopsida*" dan "*Hepaticopsida*".

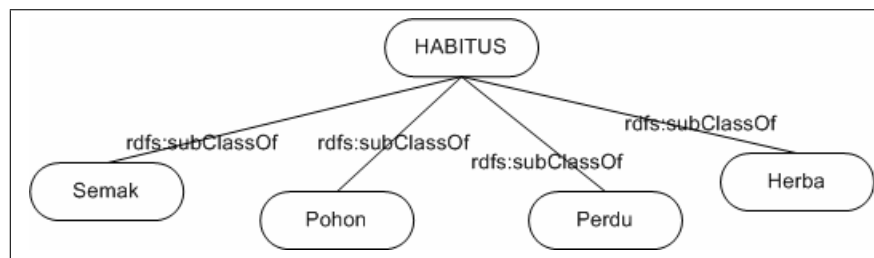
Sedangkan untuk *class* "berpembuluh" memiliki dua *subclass* yaitu "Paku-pakuan" dan "Berbiji". *Class* "Paku-pakuan" memiliki empat *subclass*, keempat *subclass* tersebut adalah Paku purba, Paku kawat, Paku ekor kuda, Paku sejati. sedangkan untuk *class* "Berbiji" memiliki dua *subclass* yakni *class* "Berbiji Terbuka" dan *class* "Berbiji Tertutup". Untuk *class* "berbiji tertutup" memiliki dua *class* yakni *class* "Monokotil" dan *class* "Dikotil". Sedangkan untuk *class* "Berbiji terbuka" memiliki empat *class*, yakni Kelas *Cycadinae*, Kelas *Coniferae*, Kelas *Gnetinae*, Kelas *Ginkgoina*. Class-class tersebut dapat dilihat hirarkinya pada skema *RDF* di bawah ini :



Gambar 3.1: Skema Tumbuhan



Gambar 3.2: Skema Habitat



Gambar 3.3: Skema Habitus

3.2.2 Property Class

Sebuah *class* jika berdiri sendiri tidak akan memberikan informasi yang cukup untuk apa yang kita inginkan dari sebuah *ontology* ini. Sekali kita mendefinisikan sebuah atau beberapa kelas, maka kita harus mendefinisikan atau menjelaskan struktur internal dari konsep yang akan kita bangun dari *class-class* tersebut.

Kita telah menentukan *class-class* yang akan kita gunakan. Sebagian atribut dari masing *class* tersebut adalah bentuk, habitat. Untuk masing-masing kelas masih memiliki perbedaan dalam isi dari propertinya.

Untuk *SubClass* "lumut" memiliki *Property* habitat dengan nilai "Daratan" dan "Permukaan Air", sedangkan untuk *property* "Habitus" *class* tersebut memiliki nilai *false*. Untuk *subclass* "Paku-pakuan" memiliki nilai untuk *prop-*

erty "Habitat" "Daratan" dan untuk nilai dari *property* "Habitus" "true" karena masing-masing *subclass* tersebut memiliki hampir semua jenis habitus yang ada. Sama dengan *subclass* dari "paku-pakuan" *subclass* "Berbiji Terbuka" dan "Berbiji Tertutup" memiliki semua nilai dari *property* tersebut.

3.2.3 Ruang Lingkup Property

Property dapat memiliki berbagai penjelasan mengenai tipe nilai, nilai yang diperbolehkan, dan nilai kardinalitas.

Kardinalitas menjelaskan berapa banyak nilai yang dimiliki, ada kardinalitas yang memiliki nilai satu dan ada yang memiliki nilai lebih dari satu. Kardinalitas yang mungkin untuk *ontology* ini adalah bernilai satu atau lebih. Sebagian besar *property* memiliki nilai satu dan hanya beberapa yang memiliki nilai lebih dari satu.

Type nilai adalah jenis nilai yang dapat dimasukkan kedalam masing-masing *property* tersebut. Nilai yang mungkin untuk kasus *RDF* di atas adalah *String* dan *Boolean*.

Bab 4

Perbandingan Tool *Ontology*

4.1 Metode Pengujian

4.1.1 Skenario Pengujian

Pengujian ini dilakukan dengan melakukan pembuatan sebuah *ontology* untuk persebaran botani di Indonesia. Masing-masing Tool akan diberikan tugas untuk membuat sebuah *ontology* dengan tiga domain yaitu Tumbuhan, Habitat, Habitus.

Masing-masing domain tersebut akan dibuat model *RDF* dan *OWL*. Untuk melihat kemampuan masing-masing *tool* tersebut. Kemudian hasil dari masing-masing *tool* tersebut akan di bandingkan dan di uji dengan *RDF* validator.

4.1.2 Parameter Perbandingan

4.1.2.1 Instalasi

Instalasi memegang peranan dalam pemanfaatan sebuah perangkat lunak. Kemudahan dalam instalasi dan *setup* akan memberikan nilai awal yang baik dari sebuah *tool*. Parameter yang digunakan akan mengacu pada beberapa faktor di bawah ini :

- **Mendapatkan Tool**, Penilaian ini akan mempertimbangkan kemudahan dalam mendapatkan *tool* tersebut, jika *tool* tersedia minimal dalam CD dan Internet maka akan mendapatkan nilai 1, sedangkan jika hanya dalam CD atau Internet akan mendapatkan nilai 0,5;
- **Ukuran File**, Ukuran file untuk proses instalasi juga akan mempengaruhi penilaian, semakin besar ukuran file instalasi akan mengurangi

penilaian total. Untuk kemudahan dalam perhitungan maka untuk file instalasi yang di bawah 100 MB akan mendapatkan nilai 1, file instalasi dengan ukuran 100 MB - 300 MB akan mendapatkan nilai 0,5, dan di atas 300 MB akan mendapatkan nilai 0.

- **Dependency File**, penilaian ini meliputi kebutuhan akan file-file pendukung untuk menjalankan *tool* tersebut dan apakah file pendukung tersebut disediakan oleh pembuat *tool* atau pihak di luar pembuat *tool* tersebut. Jika informasi file pendukung disediakan termasuk file tersebut maka akan mendapatkan nilai 1, jika hanya informasi saja yang disediakan maka akan mendapatkan nilai 0,5, jika tidak disediakan baik informasi file pendukung dan file pendukung itu sendiri maka akan diberikan nilai 0.
- **Perangkat Keras**, Seberapa besar sumber daya yang dibutuhkan untuk menjalankan aplikasi tersebut. Jika dapat dijalankan di bawah Pentium III dengan memory 128 MB, dan hardisk 200 MB maka akan mendapatkan nilai 1, sedangkan jika hanya dapat dijalankan dengan minimal Pentium IV dan memory 256 MB dan hardisk 400 MB akan mendapatkan nilai 0,5. Sedangkan jika di atas spesifikasi tersebut akan mendapatkan nilai 0.
- **Buku Manual**, jika buku manual saat proses instalasi tersedia maka akan bernilai 1, jika tidak maka bernilai 0.
- **Prosedur**, kerumitan yang timbul pada saat proses instalasi. Penilaian ini akan berdasarkan pada survei.
- **Multiplatform**, kemampuan *tool* tersebut untuk dapat beroperasi pada sistem operasi dan perangkat keras. Jika mendukung lebih dari tiga

sistem operasi dan perangkat keras maka akan bernilai 1, jika hanya 2 maka akan bernilai 0,5, jika hanya satu akan bernilai 0.

4.1.2.2 Kemudahan Penggunaan

Setiap aplikasi baik yang berbasis Web atau yang berbasis windows diharapkan dapat memberikan kemudahan dalam penggunaannya, baik itu tampilan awal atau kemudahan mendapatkan bantuan atau Help. Parameter yang akan menjadi bahan penilaian untuk bagian ini adalah :

- **User Interface**, penilaian yang digunakan adalah kemudahan pengguna dalam menggunakan aplikasi tersebut. Penilaian akan berdasarkan survei.
- **Help**, ketersediaan panduan atau help untuk penggunaan *tool* tersebut. Jika tersedia maka akan mendapatkan nilai 1, jika tidak akan mendapatkan nilai 0.

4.1.2.3 Fasilitas

Salah satu penilaian suatu *tool* adalah kemampuannya untuk melakukan berbagai macam kegiatan dalam satu aplikasi, apakah itu proses export atau import. Untuk bagian ini yang menjadi bahan penilaian adalah :

- **Format File**, kemampuan *tool* tersebut untuk dapat membuka berbagai format file yang berhubungan dengan *ontology*. Untuk *tool* yang dapat membuka lebih dari tiga format file maka akan mendapatkan nilai 1, untuk yang hanya dapat membuka dua format file akan mendapatkan nilai 0,5, sedangkan yang hanya dapat membuka satu format file maka mendapatkan nilai 0.

- **Eksport/Import**, kemampuan *tool* tersebut untuk melakukan transformasi atau perubahan format ontology ke berbagai bentuk. Untuk yang dapat melakukan eksport dan import mendapatkan nilai 1, untuk yang hanya eksport atau import mendapatkan nilai 0,5, sedangkan untuk yang tidak keduanya mendapatkan nilai 0.
- **Validator**, ketersediaan untuk melakukan validator atau pengecekan terhadap *ontology* yang telah dibuat. Jika tersedia fasilitas validator maka akan mendapatkan nilai 1, sedangkan jika tidak tersedia akan mendapatkan nilai 0.
- **Plug-In**, terdapatnya fungsi-fungsi tambahan yang menambah kemampuan *tool* tersebut, fungsi tersebut tidak secara otomatis terdapat dalam *tool* tersebut. Jika tersedia *Plug-in* maka akan mendapatkan nilai 1, sedangkan jika tidak 0.

4.1.2.4 Faktor Lain

Ada beberapa faktor pendukung yang juga mempengaruhi penilaian *tool* tersebut, antara lain:

- **Lisensi**, apakah lisensi yang dimiliki oleh *tool* tersebut. Jika Lisensi tersebut free maka akan mendapatkan nilai 1, jika tidak maka 0.
- **Komunitas**, ketersediaannya sekumpulan user pengguna *tool* tersebut, atau biasa disebut milis. Jika tersedia milis maka mendapatkan nilai 1, sedangkan jika tidak akan mendapatkan nilai 0.

Untuk penilaian yang membutuhkan survei, akan dilakukan penilaian sendiri. Penilaian yang membutuhkan proses survei adalah "Procedure" dan "Grafical User Interface". Survei akan dilakukan dengan mengambil sampel

dari sepuluh orang yang melakukan penilaian dengan menjalankan *tool* yang akan diuji. Dari hasil penilaian yang melalui survei akan dilihat peringkat *tool* yang tertinggi untuk dua kriteria penilaian tersebut. Penilaian Survei ini disarikan dari Web Tim Pandu [19].

Untuk penilaian "Procedure" ada beberapa sub penilaian, yaitu :

1. Proses Instalasi, apakah proses instalasi dilakukan dengan Command Line, GUI, atau kedua-duanya?. Penilaian ini memberikan persentase 30 % kepada total penilaian untuk "Procedure";
2. Keterangan Proses, apakah ada petunjuk untuk setiap langkah-langkah instalasi?. Memberikan persentase 30 % untuk total penilaian "Procedure";
3. Langkah, Berapa langkah yang harus dilewati untuk dapat menginstall *tool* tersebut. Penilaian ini memberikan 15 % dari total nilai;
4. Proses Otomatis, apakah proses instalasi dilakukan dengan otomastis seperti penentuan folder aplikasi. Hal ini akan menyumbangkan 15 % untuk penilaian "Procedure".
5. Help, apakah terdapat bantuan untuk memberikan panduan dalam proses instalasi. Penilaian ini memberikan 10 % dari total nilai.

Sedangkan untuk penilaian "GUI" ada beberapa sub penilaian, yaitu :

1. Komposisi Warna, warna yang digunakan untuk *User Interface*. Sub penilaian ini akan memberikan persentase nilai 20 %;
2. Perubahan Warna, apakah warna dapat dirubah sesuai keinginan *user*?. Akan memberikan nilai dengan persentase 20 % dari nilai total "GUI";

3. Letak Icon, apakah letak icon memudahkan *user* untuk menggunakannya?. Penilaian ini memberikan 10 % dari total nilai untuk "GUI";
4. Gambar Icon, apakah gambar yang digunakan untuk simbol pada icon dapat mudah dipahami?. Akan memberikan nilai 15 % untuk penilaian "GUI";
5. Letak Windows, letak windows memberikan kemudahan bagi *user*. 15 % akan diberikan dari penilaian ini.
6. Pengaturan Letak Windows, apakah letak windows dapat diatur oleh *user*?. Nilai 20 % akan disumbangkan dari penilaian ini.

Dari masing-masing sub penilaian tersebut akan dilakukan perhitungan total untuk masing-masing penilaian tersebut.

4.1.3 *Tool yang diuji*

4.1.3.1 *Protégé*

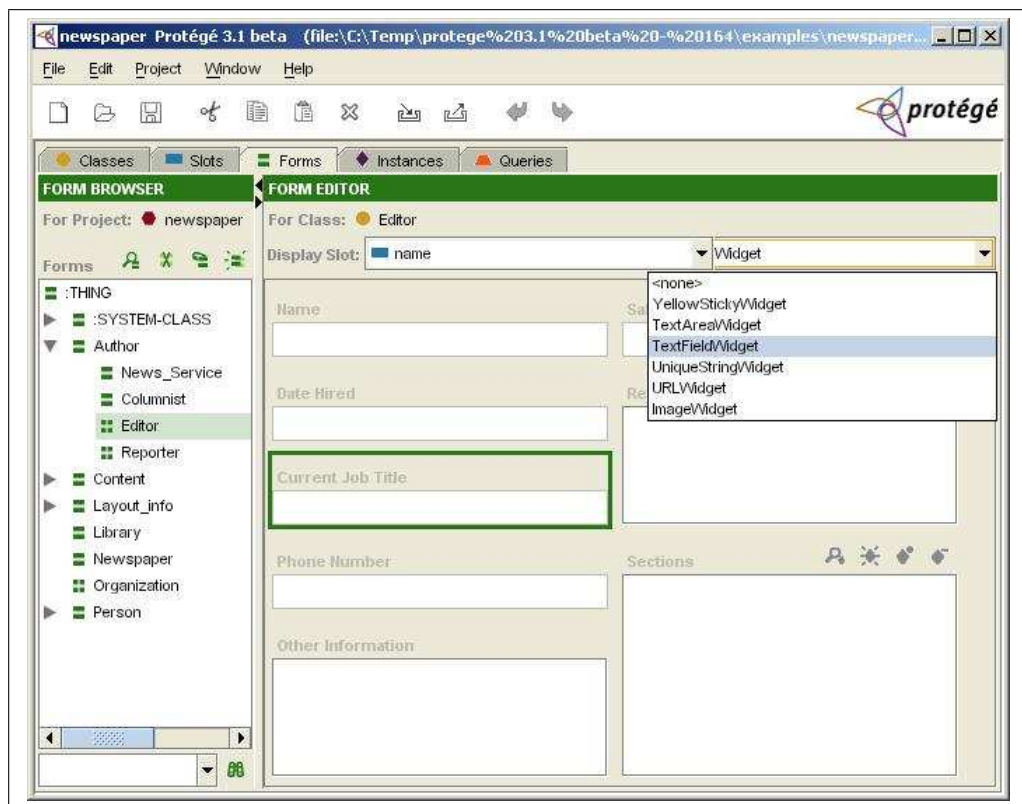
Protégé adalah sebuah alat bantu yang berbentuk perangkat lunak yang digunakan untuk pengembang system untuk mengembangkan *Knowledge-Base System*. Aplikasi yang dikembangkan dengan Protégé digunakan dalam pemecahan masalah dan pembuat keputusan dalam sebuah domain.

Protégé dikembangkan oleh sebuah organisasi yang bernaung di bawah *Stanford*, yang mengambil spesialisasi dibidang *ontology*. Segala sesuatu yang berhubungan dengan Protégé dapat dilihat pada alamat <http://Protege.stanford.edu/>, termasuk tutorial dan komunitas pengguna Protégé

Protégé merupakan sebuah alat yang digunakan untuk membuat sebuah domain *ontology*, menyesuaikan form untuk entry data, dan memasukan data. Berbagai format penyimpanan seperti *OWL*, *RDF*, *XML*, dan *HTML*.

Protégé menyediakan kemudahan plug and play yang membuatnya fleksibel untuk pengembangan *prototype* yang berkembang.

Protégé dibuat dengan menggunakan bahasa pemrograman Java. Semua alat-alat dalam Protégé dapat digunakan melalui *Graphical User Interface* (GUI) dengan menyediakan Tab untuk masing-masing bagian dan fungsi standar. *Class* Tab dalam editor *ontology* berfungsi untuk mendefinisikan *class* dan hirarki *class*, *property* dan nilai *property* tersebut, relasi antara *class* dan *property* dari relasi tersebut [20].



Gambar 4.1: Protégé, bersumber dari [20]

4.1.3.2 *Altova*

Tool yang kedua untuk membuat *ontology* adalah Altova Semantic Work yang dibuat oleh sebuah perusahaan pembuat software Altova. Dengan meng-

gunakan Altova Semantic Work, pengembangan *ontology* dilakukan dengan gambar-gambar. Yang dapat dilakukan pembuatan dan perubahan adalah *RDF*, *RDFS* dan *OWL* termasuk pemeriksaan sintaksis. Semua yang berhubungan dengan Semantic Work dapat dilihat pada http://www.altova.com/products_semanticwork.

Altova Semantic Work menyediakan beberapa fungsi, antara lain [1]:

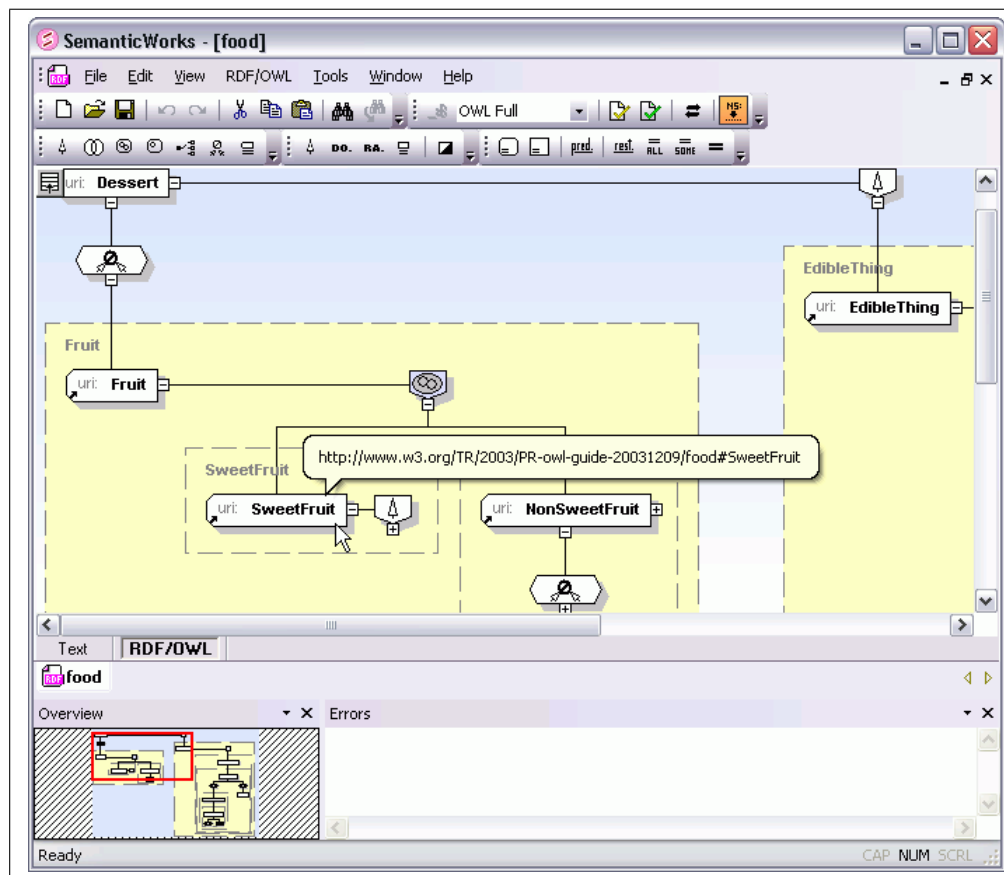
- Pembuatan dan perubahan secara visual dari *RDF*, *RDFS*, *OWL*.
- Pemeriksaan sintaksis untuk menyesuaikan kemampuan dengan spesifikasi *RDF/XML*.
- *Auto Generated RDF/XML* dan format N-triples berdasarkan rancangan *RDF/OWL*.
- Mencetak desain *RDF/OWL* yang berbentuk gambar untuk membuat dokumentasi web *semantic*

4.1.3.3 *SWOOP*

Ada sebuah *tool* lagi yang akan diuji, yaitu SWOOP. keduanya berasal dari Mindswap yang merupakan organisasi yang bergerak dibidang semantic web. SWOOP dibuat dengan menggunakan bahasa pemrograman Java, yang berbasis *Windows Base Application*. SWOOP dapat diperoleh di dan <http://www.mindswap.org/2004/SWOOP>, termasuk segala yang berhubungan dengannya.

SWOOP dirancang untuk melakukan development *ontology*. Dari *tool* ini ada memiliki beberapa kemampuan yaitu [24]:

- Tampilan Browser untuk *ontology* menyerupai tampilan pada browser untuk halaman web.



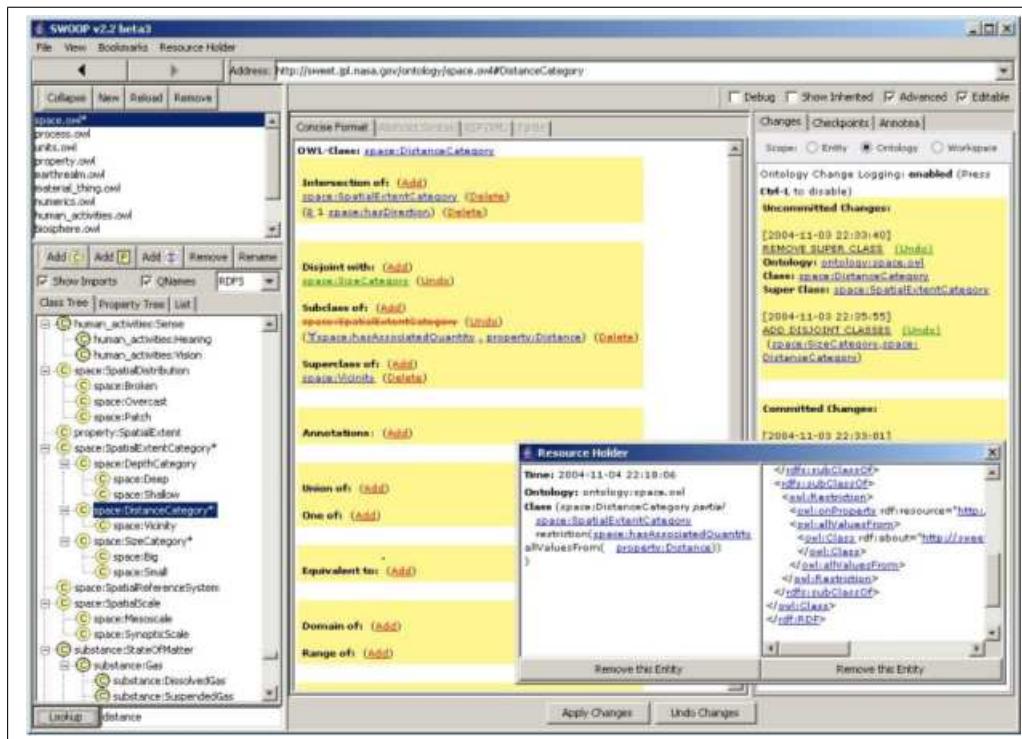
Gambar 4.2: Altova, bersumber dari [1]

- Perubahan *ontology* dilakukan dengan metode inline, yaitu semua perubahan yang dilakukan akan diikuti *class-class* yang mengikutinya.
- Dirancang memang untuk mengakomodasi kebutuhan *OWL*, termasuk *RDF*, *N3*.

4.2 Pengujian

4.2.1 Hasil Pengujian

Setiap *tool* yang dilakukan pengujian akan memberikan hasil yang berbeda-beda untuk tiap bagian penilaian. Nilai yang didapat akan diakumulasikan un-



Gambar 4.3: SWOOP, bersumber dari [24]

tuk mendapatkan nilai total yang akan memberikan kesimpulan yang berbeda-beda. Berikut ini adalah hasil yang didapat dari pengujian yang dilakukan untuk masing-masing *tool*:

1. Protégé

Berikut ini adalah hasil penilaian yang dilakukan pada *tool* Protégé dapat dilihat pada tabel 4.1, berikut ini adalah keterangan penilaian tersebut :

Keterangan:

(a) Mendapatkan *Tool*

Tabel 4.1: Hasil Penilaian Protégé

No	Jenis Penilaian	Nilai
1	Mendapatkan <i>tool</i>	0,5
2	Ukuran File	1
3	Dependency File	1
4	Perangkat Keras	1
5	Buku Manual	1
6	Prosedur	1
7	Multiplatform	1
8	User Interface	1
9	Help	1
10	Format File	1
11	Eksport/Import	1
12	Validator	1
13	Plug-In	1
14	Lisensi	1
15	Komunitas	1
	Total	14,5
	Rata-rata	0,967

Untuk mendapatkan Protégé dapat dilakukan dengan cara *Download* dari web penyedia *Tool*, alamat web tersebut adalah <http://Protege.stanford.edu/>. Langkah pertama yang harus dilakukan adalah melakukan pendaftaran, kemudian anda akan diarahkan untuk menuju ke halaman *download*. Karena hanya dengan menggunakan WEB untuk mendapatkan *tool* Protégé, maka untuk penilaian ini akan diberikan nilai 0,5.

(b) **Ukuran File**

Ukuran file instalasi untuk *tool* Protégé tergantung pada versi yang diinginkan dan juga tergantung termasuk atau tidaknya SDK Java. Untuk Protégé versi 3.2 sesuai yang digunakan untuk penilaian ini, ukuran file yang di dalamnya termasuk SDK Java adalah 61.6

MB. Karena ukuran file instalasi hanya 61.6 MB dan ukuran itu termasuk kedalam penilaian untuk ukuran <100 MB, maka untuk *tool* Protégé mendapatkan nilai 1 untuk kriteria "Ukuran File".

(c) ***Dependency File***

Untuk *tool* Protégé hanya membutuhkan SDK Java, terdapat dua pilihan yaitu apakah SDK Java termasuk kedalam file instalasi atau tidak. Karena kebutuhan file untuk menjalankan Protégé terdapat dua pilihan, maka untuk penilaian ini mendapatkan nilai 1.

(d) **Perangkat Keras**

Tool Protégé dapat berjalan pada komputer dengan spesifikasi Pentium III, memory 128Mb walaupun dengan proses yang agak lama. Dengan spesifikasi tersebut maka Protégé mendapatkan nilai 1 untuk penilaian "Perangkat Keras".

(e) **Buku Manual**

Dalam halaman Web Protégé disediakan instruksi untuk melakukan proses instalasi. Manual tersebut berupa instruksi yang terdapat dalam sebuah dokumen web. Penilaian ketersediaan Buku manual dalam proses instalasi memberikan nilai 1 untuk *tool* Protégé.

(f) **Prosedur**

Dari empat sub penilaian tentang "Procedure", ada beberapa kriteria yang membuat protégé mendapatkan nilai 1, yaitu :

- Proses instalasi Protégé sudah menggunakan tampilan grafik atau tidak lagi menggunakan proses Command Line.
- Disetiap proses atau langkah instalasi terdapat keterangan yang menjelaskan langkah tersebut.

- Langkah yang dibutuhkan untuk dapat selesai meng-*install* Protégé adalah 8 langkah.
- Proses *instalasi* dilakukan dengan otomatis atau bantuan *User* dapat dihilangkan.
- Help atau bantuan tidak disediakan dalam proses instalasi protégé.

(g) **Multiplatform**

Tool Protégé dapat berjalan diberbagai platform operating system, antara lain Windows, Mac OS, Solaris, Linux, HP-UX, Unix, AIX. Karena Protégé dapat dijalankan pada berbagai platform operating system maka untuk penilaian ini Protégé mendapatkan nilai 1.

(h) **User Interface**

Untuk penilaian "User Interface", Protégé mendapatkan nilai 1 karena beberapa hal berikut ini, yaitu :

- Komposisi warna untuk *tool* Protégé, melalui survei didapat 6 orang memberikan penilaian "biasa", 3 orang memberikan penilaian menarik, dan 1 orang memberikan penilaian "Buruk". Dari komposisi tersebut maka diberikan penilaian 1 untuk sub nilai "Komposisi Warna".
- Pengaturan Warna, untuk *tool* Protégé, komposisi warna dapat diatur tapi pengaturan tersebut mengikuti komposisi yang diberikan oleh Protégé.
- Letak Icon, ada 8 orang *user* yang memilih icon dalam protégé pada posisi yang tepat, sedangkan sisanya atau 2 orang mengatakan bahwa icon-icon tersebut tidak pada posisi yang memudahkan *user* untuk menggunakannya.

- Arti Icon, seluruh sukarelawan memberikan penilaian bahwa gambar untuk icon tidak dengan mudah dimengerti oleh *user*. Karena hal tersebut maka untuk sub penilaian ini tidak memberikan nilai apapun.
- Letak Windows, 7 orang mengatakan bahwa letak windows mempermudah pengoperasian *tool* Protégé, sedangkan sisanya atau 3 orang mengatakan bahwa window protégé tidak terletak pada posisi yang kurang tepat. Karena mayoritas memberikan penilaian baik, maka nilai untuk sub "Letak Windows" adalah 1.
- Pengaturan Windows, windows pada protégé dapat diatur, tetapi dengan kemampuan yang terbatas. Nilai yang diberikan adalah 1.

(i) **Help**

Untuk *tool* Protégé terdapat referensi yang dapat digunakan oleh user untuk melakukan pembuatan *ontology* maupun penjelasan tentang *User Interface* yang terdapat dalam Protégé. Semua file bantuan tersebut terdapat dalam halaman Web Protégé yaitu pada <http://Protege.stanford.edu/>. Karena terdapat banyak bantuan untuk melakukan pembuatan *ontology* dengan menggunakan protégé, maka *tool* ini mendapatkan nilai 1.

(j) **Format File**

Tool Protégé dapat membuka berbagai macam format file, ada tiga format file umum yang dapat dibuka dengan protégé, yaitu *XML*, *RDF*, dan *OWL*. Untuk dapat membuka format file tersebut hal yang perlu dilakukan adalah dengan membuat sebuah project baru

pada protégé, project tersebut akan memiliki format file .pprj, karena protégé tidak menyediakan fasilitas untuk dapat membuka secara langsung format file tersebut kecuali untuk format *OWL*. Karena dapat membuka tiga format file maka untuk penilaian ini *tool* Protégé mendapatkan nilai 1.

(k) **Eksport/Import**

Setelah pembuatan sebuah *ontology*, file *ontology* dapat dilakukan export ke berbagai format antara lain *RDF*, *OWL*, *N3*, *TURTLE*, *HTML*, dan *CLIPS*. Eksport yang dapat dilakukan adalah untuk format *ontology* dan untuk format file project Protégé.

Untuk melakukan import *ontology* dapat dilakukan dengan cara menambahkannya pada tab "Metadata". Import dapat dilakukan dari berbagai sumber antara lain dari local file dan dari *URI*.

Karena untuk fasilitas Eksport dan Import *ontology* dapat dilakukan dengan menggunakan *tool* Protégé maka untuk penilaian ini akan mendapatkan nilai 1.

(l) **Validator**

Di dalam Protégé terdapat fasilitas validator. Hanya jika ingin melakukan validator untuk *ontology* yang telah dibuat dibutuhkan sebuah aplikasi tambahan yang berupa server. Protégé merekomendasikan untuk menggunakan Racer [8].

(m) **Plug-In**

Protégé juga menyediakan *Plug-In* yang dapat digunakan untuk mempermudah penerapan dan pengembangan *ontology*. Pengembangan yang lebih banyak adalah penggambaran *ontology* yang telah dibuat dalam bentuk hirarki *class*.

Semua *Plug-In* yang dapat digunakan di dalam Protégé dapat di *Download* di web Protégé [20]

Nilai yang didapatkan oleh Protégé untuk penilaian ini adalah 1.

(n) **Lisensi**

Protégé menggunakan lisensi *Mozilla Public license (MPL)*, yang berarti setiap user diberikan kebebasan untuk mendapatkan *tool* Protégé tanpa harus memikirkan lisensi yang membutuhkan biaya. Karena Protégé tidak menggunakan lisensi berbayar maka untuk penilaian ini mendapatkan nilai 1.

(o) **Komunitas**

Protégé juga menyediakan komunitas untuk melakukan tanya jawab kesulitan dalam pengembangan menggunakan protégé. Komunitas tersebut di dukung oleh GMANE ([http : //www.gmane.org/](http://www.gmane.org/)).

Ketersediaan komunitas Protégé tersebut memberikan nilai 1 untuk penilaian "Komunitas" ini.

2. **Altova Semantic Work**

Berikut ini adalah hasil uji coba yang dilakukan dengan menggunakan Altova Semantic work dapat dilihat pada tabel 4.2, berikut ini adalah keterangan :

Keterangan:

(a) **Mendapatkan *Tool***

Tabel 4.2: Hasil Penilaian Altova Semantic Work

No	Jenis Penilaian	Nilai
1	Mendapatkan <i>tool</i>	0,5
2	Ukuran File	1
3	Dependency File	0
4	Perangkat Keras	1
5	Buku Manual	0
6	Prosedur	1
7	Multiplatform	0,5
8	User Interface	1
9	Help	1
10	Format File	1
11	Eksport/Import	1
12	Validator	1
13	Plug-In	0
14	Lisensi	0
15	Komunitas	1
	Total	10
	Rata-Rata	0,667

Tool Altova Semantic Work hanya dapat diperoleh dengan *download* dari halaman web Altova Semantic Web [1].

Karena untuk mendapatkan *tool* Altova Semantic work hanya dapat dilakukan dengan menggunakan Web, maka nilai yang diberikan dalam penilaian ini adalah 0,5.

(b) Ukuran File

File instalasi untuk Altova Semantic Work adalah 5,9 Mb. File instalasi tersebut sudah termasuk segala sesuatu yang dibutuhkan untuk menjalankan *tool* Altova Semantic Work. Dengan ukuran file instalasi yang di bawah 100 Mb memberikan nilai untuk Altova Semantic Work sebesar 1.

(c) *Dependency File*

Altova Semantic Work tidak memberikan penjelasan tentang kebutuhan akan aplikasi atau file-file pendukung yang digunakan untuk menjalankan *tool* Altova Semantic Work. Pada penilaian ini *tool* Altova Semantic Work mendapatkan nilai 0.

(d) **Perangkat Keras**

Altova Semantic Work membutuhkan Perangkat keras yang tidak besar, dengan minimal PC yang memiliki Processor Pentium III dan kapasitas memory 128Mb sudah dapat menjalankan *tool* Altova Semantic Work. Dengan spesifikasi minimal seperti ini, maka Altova Semantic Work mendapatkan nilai 1.

(e) **Buku Manual**

Dalam proses instalasi *tool* Altova Semantic Work, tidak disediakan manual yang memadai. Penilaian Altova Semantic Work untuk bagian "Buku Manual" adalah 0.

(f) **Prosedur**

Dari empat sub penilaian tentang "Procedure", ada beberapa kriteria yang membuat Altova Semantic Work mendapatkan nilai 1, yaitu :

- Proses instalasi Altova Semantic Work sudah menggunakan tampilan grafik atau tidak lagi menggunakan proses Command Line.
- Disetiap proses atau langkah instalasi terdapat keterangan yang menjelaskan langkah tersebut.
- Langkah yang dibutuhkan untuk dapat selesai meng-*install* Altova Semantic Work adalah 6 langkah.

- Proses *instalasi* dilakukan dengan otomatis atau bantuan *User* dapat dihilangkan.
- Help atau bantuan disediakan dalam proses instalasi Altova Semantic Work.

(g) **Multiplatform**

Altova Semantic Work hanya dapat dijalankan pada operating system Windows, Mac OS, dan Linux. Tetapi Altova merekomendasikan penggunaan Windows untuk penggunaan *tool* Altova Semantic Work. Kemampuan Altova Semantic Work yang dapat dijalankan pada tiga operating system memberikan nilai 1 untuk penilaian "Multiplatform".

(h) **User Interface**

Untuk penilaian "User Interface", Altova Semantic Work mendapatkan nilai 1 karena beberapa hal berikut ini, yaitu :

- Komposisi warna untuk *tool* Altova Semantic Work, melalui survei didapat 5 orang memberikan penilaian "biasa", 3 orang memberikan penilaian menarik, dan 2 orang memberikan penilaian "Buruk". Dari komposisi tersebut maka diberikan penilaian 1 untuk sub nilai "Komposisi Warna".
- Pengaturan Warna, untuk *tool* Altova Semantic Work, komposisi warna tidak dapat diatur oleh *user*.
- Letak Icon, ada 9 orang *user* yang memilih icon dalam Altova Semantic Work pada posisi yang tepat, sedangkan sisanya atau 1 orang mengatakan bahwa icon-icon tersebut tidak pada posisi yang memudahkan *user* untuk menggunakannya.

- Arti Icon, hanya 6 orang yang mengatakan bahwa gambar icon yang digunakan dapat dimengerti oleh orang awam. Sedangkan sisanya atau 4 orang mengatakan tidak dengan mudah dapat memahami gambar icon yang digunakan. Karena terdapat 50 % lebih yang mengatakan bahwa arti icon dapat dimengerti, maka nilai yang diberikan adalah 1.
- Letak Windows, 6 orang mengatakan bahwa letak windows mempermudah pengoperasian *tool* Altova Semantic Work, sedangkan sisanya atau 4 orang mengatakan bahwa window Altova Semantic Work tidak terletak pada posisi yang kurang tepat. Karena mayoritas memberikan penilaian baik, maka nilai untuk sub "Letak Windows" adalah 1.
- Pengaturan Windows, windows pada Altova Semantic Work dapat diatur. Nilai yang diberikan adalah 1.

(i) **Help**

Bantuan(Help) secara default disediakan dalam *tool* Altova, termasuk di dalam aplikasi Altova Semantic Work. Maka dengan tersedianya bantuan(Help) dalam *tool* Altova Semantic Work maka untuk penilaian ini akan mendapatkan nilai 1.

(j) **Format File**

Altova Semantic Work dapat mendukung berbagai macam format file, format tersebut adalah *RDF*, *RDFS*, *OWL*, dan *triple fi*. Dengan Altova Semantic Work semua format tersebut dapat dibuka secara langsung tanpa harus membuat sebuah project tersendiri.

Dengan kemampuan Altova Semantic Work untuk membuka berbagai format file tersebut, yang jumlahnya tiga format maka pada pe-

nilaian ini akan mendapatkan nilai 1.

(k) **Eksport/Import**

Altova Semantic Work hanya menyediakan fasilitas eksport ke dua format yaitu Triple file dan *XML*. Sedangkan untuk fasilitas import disediakan oleh Altova Semantic Work, tetapi dokumentasi untuk proses import tersebut tidak lengkap dan memiliki prosedur yang cukup rumit.

Dengan tersediannya fasilitas import dan eksport maka *tool* Altova Semantic Web mendapatkan nilai 1 untuk penilaian "Eksport/Import".

(l) **Validator**

Altova Semantic Work menyediakan validator untuk menentukan apakah *RDF/OWL* yang telah dibuat. Validator tersebut tidak membutuhkan aplikasi lain di luar Altova Semantic Work.

Dengan tersedianya fasilitas validator maka Altova Semantic Work mendapatkan nilai 1 untuk penilaian "Validator".

(m) **Plug-In**

Altova Semantic Work tidak menyediakan *Plug-in* untuk menunjang pembuatan *ontology*, maka nilai yang diberikan adalah 0.

(n) **Lisensi**

Tool Altova Semantic Work memiliki lisensi yang tidak gratis. Untuk setiap Altova Semantic Work yang di *download* merupakan versi trial atau uji coba. Versi uji coba tersebut memiliki batas waktu pemakaian 30 hari. Untuk mendapatkan versi full, lisensi yang dibutuhkan seharga kurang lebih USD 311. Dengan tidak bebasnya lisensi Altova Semantic Work maka nilai yang diberikan adalah

0.

(o) **Komunitas**

Dalam halaman web Altova Semantic Web [1] disediakan forum untuk melakukan diskusi. Nilai yang diberikan untuk penilaian "Komunitas" adalah 1.

3. **SWOOP**

Hampir sama dengan yang dilakukan *tool* sebelumnya, pada Swoop juga dilakukan pengujian dengan menggunakan Domain Taxonomi tumbuhan sebagai contoh kasusnya. Dengan menggunakan Domain tersebut, akan dilakukan uji coba pembuatan *RDF* dan *OWL*. Berikut ini adalah hasil pengujian tersebut yang terdapat dalam tabel 4.3, berikut ini adalah keterangan dari penilaian tersebut :

Keterangan:

1. **Mendapatkan *Tool***

Untuk mendapatkan *tool* SWOOP diperlukan pendaftaran dahulu, pendaftaran yang dilakukan tidak membutuhkan pembayaran hanya data diri secara umum.

SWOOP hanya dapat diperoleh melalui halaman Web atau melalui *download*, tidak dapat melalui CD yang didistribusikan. Karena untuk mendapatkan *tool* SWOOP hanya melalui halaman Web, maka nilai yang diperoleh adalah 0,5.

Tabel 4.3: Hasil Penilaian SWOOP

No	Jenis Penilaian	Nilai
1	Mendapatkan <i>tool</i>	0,5
2	Ukuran File	1
3	Dependency File	0
4	Perangkat Keras	1
5	Buku Manual	0
6	Prosedur	0
7	Multiplatform	1
8	User Interface	0,5
9	Help	0
10	Format File	1
11	Eksport/Import	0
12	Validator	0
13	Plug-In	0
14	Lisensi	1
15	Komunitas	1
	Total	7
	Rata-rata	0,467

2. Ukuran File

Setelah dilakukan pendaftaran, maka akan di arahkan ke halaman *download*. Untuk dapat menggunakan SWOOP tidak diperlukan file instalasi, hanya sebuah paket yang telah di kompresi yang berukuran 10,3Mb.

File yang dapat di *download* tersebut merupakan file-file yang diperlukan untuk menjalankan *tool* SWOOP. Ukuran tersebut memberikan nilai kepada *tool* SWOOP sebesar 1.

3. *Dependency File*

Untuk dapat menjalankan *tool* SWOOP diperlukan SDK Java, karena *tool* SWOOP berjalan dengan platform java. Didalam halaman web SWOOP tidak dijelaskan kebutuhan akan SDK Java.

Dan sesungguhnya yang paling utama, tapi sering terlewat adalah kebutuhan akan sebuah aplikasi untuk membuka kompresi dari file yang telah didapatkan dari halaman web SWOOP.

Karena kebutuhan akan file tambahan yang tidak dipublikasikan, maka nilai untuk penilaian "**Dependency File**" adalah 0.

4. **Perangkat Keras**

Memang tidak dijelaskan secara rinci kebutuhan perangkat keras untuk menjalankan *tool* ini. Tetapi setelah dicoba di komputer Pentium III dan dengan memory 128Mb sudah dapat berjalan dengan normal, hanya cukup lambat prosesnya. Dengan kemampuannya tersebut maka untuk penilaian ini *tool* SWOOP mendapatkan nilai 1.

5. **Buku Manual**

Untuk dapat menggunakan SWOOP tidak diperlukan sebuah proses instalasi, karena setelah *tool* SWOOP didapatkan dapat langsung dijalankan dengan menggunakan file bat yang telah disediakan. Maka dalam penilaian ini *tool* SWOOP mendapatkan nilai 0

6. **Prosedur**

Karena untuk dapat menjalankan SWOOP tidak diperlukan sebuah proses instalasi maka dalam penilaian ini, yang dilakukan penilaian adalah proses menjalankan *tool* SWOOP. Hasil penilaian tersebut adalah :

- Proses instalasi SWOOP membutuhkan command line untuk dapat menjalankannya, tetapi oleh SWOOP disediakan *shortcut* untuk dapat langsung menjalankan SWOOP. Karena masih menggunakan

command line maka nilai yang diperoleh SWOOP untuk penilaian ini adalah 0.

- Tidak terdapat keterangan yang dilakukan untuk setiap langkah dalam menjalankan SWOOP. Nilai yang didapatkan adalah 0.
- Langkah yang ditempuh untuk dapat menjalankan SWOOP adalah satu langkah. Karena masih di bawah 5 langkah maka nilai yang diperoleh adalah 1.
- Untuk dapat menjalankan SWOOP, tidak dapat dengan otomatis. tetapi masih dibuthkan campur tangan *user* secara keseluruhan. Nilai yang didapat untuk penilaian ini adalah 0.
- Help atau bantuan tidak disediakan dalam proses instalasi SWOOP.

Karena penilaian tersebut maka nilai yang didapatkan oleh SWOOP untuk penilaian "Procedure" adalah 0.

7. Multiplatform

Tool SWOOP dapat dijalankan pada platform Windows, Linux, dan Mac OS. Karena kemampuannya untuk berjalan pada tiga operating system maka dalam penilaian ini *tool* SWOOP mendapatkan nilai 1.

8. User Interface

Untuk penilaian "User Interface", SWOOP mendapatkan nilai 1 karena beberapa hal berikut ini, yaitu :

- Komposisi warna untuk *tool* SWOOP, melalui survei didapat 8 orang memberikan penilaian "biasa", dan 2 orang memberikan penilaian "Buruk". Dari komposisi tersebut maka diberikan penilaian 0.5 untuk sub nilai "Komposisi Warna".

- Pengaturan Warna, untuk *tool* SWOOP, komposisi warna tidak dapat dirubah oleh *user*.
- Letak Icon, karena pada SWOOP tidak terdapat icon, maka untuk penilaian ini tidak memberikan nilai kepada SWOOP.
- Arti Icon, hal yang sama juga berlaku pada penilaian ini, yaitu tidak memberikan penilaian apapun pada SWOOP.
- Letak Windows, 5 orang memberikan nilai pada tata letak windows yang memberikan kemudahan dalam penggunaan SWOOP. Dan sisanya atau 5 orang mengatakan bahwa letak windows pada SWOOP tidak pada posisi yang bagus. Karena hasil untuk penilaian ini seimbang, maka nilai yang didapat adalah 0,5.
- Pengaturan Windows, pada SWOOP tidak dapat dilakukan pengaturan windows.

Karena hasil tersebut maka SWOOP mendapatkan nilai 0,5 untuk penilaian "User Interface".

9. Help

Tool SWOOP tidak menyediakan bantuan atau referensi yang dapat mempermudah pengembangan *ontology*. Karena tidak teesedianya fasilitas *Help* atau bantuan, maka untuk penilaian ini *tool* SWOOP mendapatkan nilai 0.

10. Format File

Format file yang dapat dibuka dengan *tool* SWOOP adalah *RDF*, *OWL* dan *XML*. Dan SWOOP memiliki file tambahan yaitu *.SWO* yang merupakan file standar buatan SWOOP. Untuk penilaian ini, *tool* SWOOP

mendapatkan nilai 1.

11. **Eksport/Import**

Tool SWOOP tidak menyediakan fasilitas Eksport atau Import. Maka mendapatkan nilai 0.

12. **Validator**

Tool SWOOP juga tidak menyediakan fasilitas Validator, yang ada hanyalah penjelasan singkat mengenai link yang dapat digunakan untuk validator. Oleh karena itu *tool* SWOOP mendapatkan nilai 0 untuk penilaian "Validator".

13. **Plug-In**

Tidak dijelaskan secara jelas ketersediaan *Plug-in*. Disini diasumsikan tidak tersedianya *Plug-in* untuk *tool* SWOOP. Maka dalam penilaian ini akan mendapatkan nilai 0.

14. **Lisensi**

Lisensi untuk *tool* SWOOP adalah MIT, yang berarti *tool* ini bebas untuk di dapatkan atau di copy serta di umumkan dan di lakukan perbaikan atau penambahan. Lisensi untuk *tool* SWOOP dapat dilihat di file kompresi yang didapatkan serta di [http : //www.mindswap.org/2004/SWOOP/license/](http://www.mindswap.org/2004/SWOOP/license/). Karena *tool* ini mendapatkan nilai 1.

15. **Komunitas**

Terdapat dua jenis komunitas, yaitu komunitas untuk pengguna biasa dan komunitas untuk para Developer. Komunitas tersebut tersedia di [http : //lists.mindswap.org/mailman/listinfo/swoop – devel](http://lists.mindswap.org/mailman/listinfo/swoop-devel) untuk

Developer dan [http : //lists.mindswap.org/mailman/listinfo/swoop](http://lists.mindswap.org/mailman/listinfo/swoop) untuk Pengguna Biasa. Dengan tersedianya komunitas untuk *tool* SWOOP maka akan mendapatkan nilai 1.

Dari hasil penilaian yang telah dilakukan terhadap masing-masing *tool* tersebut, kemudian akan dilakukan analisa terhadapnya.

4.2.2 Analisis Hasil Pengujian

Setelah dilakukan penilaian terhadap masing-masing *tool* tersebut, didapatkan hasil penilaian yang beda-beda untuk *tool* tersebut. Rangkuman hasil penilaian dapat dilihat pada gambar di bawah ini :

Tabel 4.4: Rangkuman Penilaian *tool*

No	Jenis Penilaian	Protégé	Altova Semantic Work	SWOOP
1	Mendapatkan <i>tool</i>	0,5	0,5	0,5
2	Ukuran File	1	1	1
3	Dependency File	1	0	0
4	Perangkat Keras	1	1	1
5	Buku Manual	1	0	0
6	Prosedur	1	1	0
7	Multiplatform	1	0,5	1
8	User Interface	1	1	0,5
9	Help	1	1	0
10	Format File	1	1	1
11	Eksport/Import	1	1	0
12	Validator	1	1	0
13	Plug-In	1	0	0
14	Lisensi	1	0	1
15	Komunitas	1	1	1
	Total	14,5	10	7
	Rata-rata	0,967	0,667	0,467

Dari penilaian yang dilakukan oleh masing-masing *tool* tersebut, akan diambil kesimpulan untuk masing-masing kriteria penilaian. Hasil kesimpulan tersebut sebagai berikut :

1. Mendapatkan *Tool*

Dari masing-masing *tool* tersebut memiliki persamaan dalam hal mendapatkan *tool* tersebut, yaitu hanya melalui *website*. Yang menjadi perbedaan adalah proses untuk mendapatkannya, untuk *tool* Protégé dan SWOOP mengharuskan setiap user yang akan menggunakan *tool* tersebut untuk mendaftar terlebih dahulu. Sedangkan untuk *tool* Altova Semantic Work tidak memerlukan langkah pendaftaran. Untuk penilaian ini semua *tool* memiliki cara mendapatkan yang sama yang membedakan adalah langkah-langkah sampai bisa *download* *tool* tersebut, dari penilaian ini dari ketiga *tool* tersebut yang terbaik adalah Altova Semantic Work karena tidak membutuhkan prosedur yang sulit untuk mendapatkannya.

2. Ukuran File

Setiap *tool* yang diuji memiliki ukuran file instalasi yang relatif sama, hanya untuk *SWOOP* file yang didapatkan adalah merupakan file-file untuk menjalankan *tool* SWOOP dan bukan merupakan file instalasi. Jika dilihat dari ukuran file, maka untuk *tool* Protégé berada di urutan terendah, karena memiliki ukuran file instalasi yang terbesar yakni 61,6Mb, dan yang terkecil adalah Altova Semantic Work yang hanya sebesar 5,9Mb. Dari perbedaan tersebut terlihat *tool* yang terbaik dalam hal ukuran file instalasi adalah Altova Semantic Work.

3. *Dependency File*

Dari penilaian ini yang menjelaskan tentang kebutuhan perangkat lunak lain hanyalah *tool* Protégé, dan Protégé juga memberikan perangkat lunak tersebut di dalam file instalasi yang dapat digunakan bersama atau mengambil dari luar. Untuk Altova Semantic Work tidak ada penjelasan mengenai kebutuhan perangkat lunak lain dan pada saat proses instalasi tidak ada penjelasan atau keterangan yang menjelaskan hal tersebut, sedangkan untuk SWOOP tidak menjelaskan kebutuhan perangkat lunak di luar *tool* tersebut, tetapi pada saat dijalankan membutuhkan SDK Java sebagai platform *tool* tersebut. Dari keterangan tersebut *tool* yang terbaik untuk penilaian ini adalah Protégé.

4. **Perangkat Keras**

Semua *tool* yang mengalami pengujian dapat dijalankan pada komputer dengan spesifikasi processor Pentium III, memory 128Mb. Dari ketiga *tool* yang dilakukan penilaian tersebut tidak ada yang lebih baik diantara ketigannya.

5. **Buku Manual**

Hanya Protégé yang memberikan bantuan atau langkah-langkah dalam melakukan proses instalasi. Sedangkan kedua *tool* lainnya tidak memberikan bantuan atau penjelasan dalam proses instalasi. *Tool* yang terbaik dalam hal pemberian "Buku Manual" dalam proses instalasi adalah Protégé.

6. **Prosedur**

Untuk penilaian "Prosedur" terdapat dua *tool* yang menempati posisi yang sama, yaitu Protégé dan Altova Semantic Work. Hanya kalau dilihat sub penilaian maka *tool* Altova Semantic Work memiliki nilai tert-

inggi daripada Protégé. Maka urutan tertinggi adalah Altova Semantic Work, kemudian Protégé, dan yang terakhir adalah SWOOP.

7. **Multiplatform**

Semua *tool* yang dilakukan pengujian memiliki kemampuan untuk berjalan di atas berbagai macam sistem operasi. Ketigannya dapat berjalan di atas tiga sistem operasi yaitu Windows, Linux, MAC OS. Untuk *tool* Protégé dapat berjalan di atas enam sistem operasi, maka untuk penilaian ini Protégé direkomendasikan untuk digunakan.

8. **User Interface**

Untuk penilaian "User Interface", *tool* yang mendapatkan nilai tertinggi adalah Protégé, peringkat kedua adalah Altova Semantic Work, dan yang ketiga adalah SWOOP.

9. **Help**

Dalam penilaian ini yang direkomendasikan adalah *tool* Altova Semantic Work dan Protégé dalam hal ketersediaan file bantuan atau *tutorial* yang dapat digunakan secara umum, baik yang termasuk di dalam aplikasi tersebut atau yang di luar aplikasi.

10. **Format File**

Walaupun ketiga *tool* yang diuji dapat mendukung minimal tiga format file, tetapi hanya Altova Semantic Work yang tidak membutuhkan metode yang rumit untuk membuka format file tersebut. Karena kedua *tool* lainnya yakni SWOOP dan Protégé membutuhkan sebuah format file tambahan atau yang disebut file project, yang digunakan untuk membuka format file yang tidak dibuat dengan *tool* tersebut. Karena

kemampuan yang dimilikinya Altova Semantic Work direkomendasikan untuk dimiliki jika dilihat dari kemudahan dalam membuka berbagai macam format file *ontology*.

11. **Eksport/Import**

Altova Semantic Work dan Protégé memiliki fasilitas untuk melakukan Eksport/Import. Dan SWOOP tidak menyediakan fasilitas tersebut. Dari kedua *tool* yang memiliki fasilitas Eksport/Import tersebut hanya Protégé yang direkomendasikan, karena memiliki langkah yang mudah jika dibandingkan dengan Altova Semantic Work.

12. **Validator**

Hanya *tool* SWOOP yang tidak menyediakan fasilitas validator untuk mengetahui apakah *ontology* yang telah dibuat sudah benar atau tidak. Melalui penilaian ini *tool* Altova Semantic Work dan Protégé direkomendasikan untuk pengembangan *ontology* jika dilihat dari ketersediaannya fasilitas validator.

13. **Plug-In**

Hanya *tool* Protégé yang menyediakan *Plug-in* untuk tambahan fungsi dalam hal pengembangan *ontology*. Kedua *tool* lainnya, yaitu Altova Semantic Work dan SWOOP tidak menyediakan fasilitas *Plug-in*. *Tool* Protégé direkomendasikan dalam penilaian "*Plug-In*".

14. **Lisensi**

Untuk *tool* Protégé dan SWOOP memiliki lisensi yang sama, yaitu lisensi MPL yang tidak mengharuskan penggunaannya untuk membayar. Sedangkan untuk *tool* ALtova Semantic Work memerlukan lisensi yang meng-

haruskan penggunaanya untuk membayar jika ingin menggunakan *tool* tersebut lebih dari tiga puluh 30 hari. *Tool* Protégé dan SWOOP direkomendasikan dalam penilaian ini karena keduanya tidak memerlukan pembayaran lisensi dalam penggunaanya.

15. Komunitas

Semua *tool* yang dilakukan pengujian memiliki komunitas yang dapat saling bertukar permasalahan dalam pengembangan *ontology*.

Secara keseluruhan penilaian yang memberikan nilai terbaik adalah Protégé, karena dari segi cara mendapatkan *tool*, fasilitas yang ditawarkan, dan berbagai faktor pendukung memberikan nilai sempurna. Kecuali untuk penilaian "User Interface", masih ada yang lebih bagus penilaiannya yaitu Altova Semantic Work.

Didalam Protégé terdapat berbagai fungsi yang membuatnya mendapatkan nilai lebih, seperti berbagai macam format file yang dapat didukung oleh Protégé. Protégé juga memberikan fungsi grafik yang menggambarkan skema yang dibuat, tetapi untuk fungsi standarnya tidak disediakan fungsi grafik, fungsi grafiknya dapat ditambahkan dengan penambahan *Plug-in* kedalam Protégé. Protégé juga memiliki fungsi Reasoning terhadap *ontology* yang dibuat, walaupun masih membutuhkan perangkat lunak tambahan.

Mengenai "User Interface", Protégé hanya memiliki empat kombinasi warna untuk tampilan muka dan warna-warna yang dapat dipilih tersebut bagi orang yang dilakukan survei masih kurang membuatnya menarik untuk sebuah perangkat lunak komputer. Namun yang lebih membuatnya lebih baik adalah dengan tersediannya bantuan yang berupa file-file help atau yang berupa komunitas yang memberikan ruang untuk para developer untuk mengembangkan *ontology*.

Untuk Altova Semantic Work menempati peringkat kedua, karena ada beberapa penilaian yang tidak dimiliki oleh Altova Semantic Work seperti ketidaktersediannya fungsi *Plug-in*, kompatibilitas dengan hanya beberapa operating system, dan lisensi untuk dapat menggunakan Altova Semantic Work yang membutuhkan pembayaran atau tidak didapatkan secara gratis.

Masalah lisensi tersebutlah yang membuat Altova Semantic Work menempati posisi yang rendah, karena lisensi tersebut menjadi faktor penentu dari sebuah perangkat lunak dewasa ini. Salah satu yang membuat Altova Semantic Work memiliki nilai yang baik adalah "User Interface" yang dimilikinya memiliki tampilan yang baik menurut survei, serta fungsi grafik yang menjadi standar dalam Altova Semantic Work. Namun dengan adanya bantuan dan komunitas yang membantu dalam memberikan solusi, juga memberikan tambahan nilai yang membuat Altova Semantic Work menempati peringkat ke dua.

SWOOP tidak direkomendasikan untuk digunakan karena hampir setiap penilaian yang dilakukan terhadapnya tidak memberikan tambahan nilai untuk perhitungan nilai total. Yang memberikan tambahan nilai hanyalah cara mendapatkan file SWOOP yang menggunakan media internet, ukuran file untuk menjalankan SWOOP yang relatif kecil, perangkat keras yang dapat didukungnya yang relatif tidak membutuhkan spesifikasi yang besar, cukup banyak format file *ontology* yang dapat didukungnya, lisensi yang tidak membutuhkan biaya atau *tool* SWOOP merupakan *tool* yang gratis, serta ketersediaannya komunitas yang dapat membantu pengembangan *ontology* menggunakan SWOOP. SWOOP merupakan project sample dari Mindswap yang hanya memberikan kemampuan yang terbatas dan hanya cocok untuk orang yang baru dalam pengembangan *ontology*.

Bab 5

Web *Semantic* dan *Ontology*

5.1 Penggunaan Dalam Informasi Interoperabilitas

Dalam informasi Interoperabilitas terdapat berbagai macam model penjabaran informasi yang berkembang dengan sangat pesat. Setiap orang yang memiliki kebebasan untuk menerapkan metode tertentu untuk menampilkan informasi yang dimilikinya. Selain itu dalam waktu yang relatif singkat jumlah informasi yang tersebar didunia internet semakin tidak terprediksi.

Selain itu setiap orang memiliki penjabaran atau penggambaran masing-masing mengenai sebuah informasi. Setiap penjabaran tersebut sudah tentu memiliki pengertian, pengertian tersebut ada yang memiliki penjabaran sama dengan orang lain atau berbeda sama sekali.

Jika masih menggunakan metode tradisional maka pencarian informasi dalam dunia maya tersebut akan menemukan kesulitan, seperti perbedaan skema untuk struktur informasi tersebut dan juga perbedaan bahasa yang digunakan. Kalau diperhatikan dengan seksama dalam pencarian sebuah informasi masih dijumpai ketidakcocokan antara informasi yang diinginkan dengan informasi yang didapatkan.

Pendekatan tradisional memiliki fokus pada standarisasi dan arsitektur. Hal tersebut yang membuat pendekatan tradisional masih digunakan hingga saat ini. Tetapi pendekatan tradisional tidak disiapkan untuk menyelesaikan masalah yang timbul dalam dunia internet yaitu masalah interoperabilitas. Web *Semantic* diperkenalkan untuk mencoba menyelesaikan masalah interoperabilitas tersebut. *Ontology* menjadi inti dari pendekatan *semantic* tersebut.

Pada awal pendekatan *Semantic* digunakan, masih berbasis pada penggunaan *thesauri* untuk mengartikan kosakata yang digunakan. Pendekatan ini

sangat tergantung pada domain dari *thesauri* tersebut, solusi yang ditawarkan pada saat itu adalah penggunaan *ontology* yang terintegrasi secara global.

penggunaan metode *semantic* yang salah satunya menggunakan teknologi *ontology* akan memiliki struktur atau hirarki dari sebuah domain yang dapat digunakan untuk mencari sumber informasi yang relevan dengan yang diinginkan.

Ontology tersebut memiliki suatu domain yang khusus untuk suatu ilmu atau masalah tertentu. Dari setiap domain tersebut memiliki keterhubungan dengan *Representation Ontology*, yang berfungsi sebagai muara dari berbagai *ontology* yang terdapat di dunia. Jadi setiap aplikasi atau halaman web yang menggunakan metode *semantic* sebagai pendekatan untuk proses searching akan terhubung ke *Representation Ontology* tersebut untuk mencari domain-domain yang diinginkan.

Setiap halaman web yang memiliki sumber informasi tertentu untuk dapat disaring diwajibkan memiliki tambahan metadata, yang biasa disebut Anotation. Anotation ini merupakan sebuah penghubung yang menghubungkan antara web tersebut dengan sebuah domain *ontology*. Anotation tersebut memiliki kriteria tertentu untuk penggunaannya, karena Anotation tersebut menjadi ciri dari sebuah halaman web yang di dalamnya menjelaskan kandungan informasi web tersebut.

Pencocokan *ontology* yang akan digunakan merupakan tantangan terbesar dari pemetaan *ontology*. Menurut NOY [31], terdapat dua arsitektur yang digunakan untuk pemetaan antara *ontology*, yaitu menggunakan metode *shared ontology* dan menggunakan metode yang mengartikan atau menggunakan bahasa yang dapat dimengerti oleh mesin atau yang disebut dengan *Heuristics-based*. *Shared ontology* merupakan *upper ontology* yang telah dise-

tujui oleh beberapa pengembang yang dapat digunakan untuk berbagai macam aplikasi. *Heuristics-based* menggunakan karakteristik yang berbagai macam dari sebuah *ontology*, seperti strukturnya atau konsepnya untuk menemukan pemetaan yang tepat.

Untuk mengakses sebuah database diperlukan sebuah query. Query dalam sistem database yang *heterogeneous* dapat di ekspresikan pada komponen database itu sendiri atau menggunakan sistem tersendiri yang khusus memproses bahasa query. Proses pemrosesan query memerlukan mekanisme tertentu untuk *dekomposisi*, dan untuk mengartikan query tersebut hingga menjadi sub-sub query.

Sama seperti model tradisional, model semantic juga menggunakan query untuk mengambil atau *re-trieve* data agar menampilkan data yang sesuai dengan keinginan pencari informasi. Query yang diinginkan dikirimkan ke sebuah server yang berfungsi untuk menerjemahkan query. di dalam server tersebut terdapat sebuah *wrappers* yang berfungsi mencari domain yang tepat sebagai langkah selanjutnya untuk mencari informasi tersebut.

Untuk melakukan query pada sebuah *semantic web* membutuhkan query tambahan yang tidak termasuk kedalam spesifikasi *Knowledge Bases* yang berguna untuk menjawab query tersebut atau yang lebih tepat untuk mengartikan query yang dikirimkan agar dapat dimengerti oleh mesin.

Setelah sebuah query dikirimkan dan diterjemahkan kedalam bahasa yang dapat dimengerti oleh mesin, maka hasil terjemahan query tersebut akan menjadi bahan acuan untuk mencari domain yang dimaksud. Pencarian informasi tersebut berdasarkan pada *upper ontology* yang tersedia dan mencari sub-sub domain yang relevan denganya. Setelah mendapat domain yang relevan maka untuk mencari halaman web yang cocok akan memanfaatkan annotation

yang terdapat dalam metadata web tersebut.

Dalam sebuah halaman web data-data ditampilkan dalam format HTML. Metode yang digunakan untuk dapat mengekstrak data dari HTML biasanya menggunakan sebuah modul yang disebut *wrappers*. Pada awalnya proses *wrappers* menggunakan metode manual. Permasalahan yang timbul jika menggunakan metode tradisional adalah terkadang menyulitkan untuk memahami isi dari masing-masing halaman web tersebut serta sangat membutuhkan tenaga kerja yang banyak.

Data yang ditampilkan dalam sebuah halaman Web merupakan data yang tidak terstruktur atau yang semi terstruktur. Permasalahan yang timbul adalah bagaimana melakukan proses *wrapping* untuk data yang tidak dan semi terstruktur tadi. Hasil yang didapat dari proses *wrapping* biasanya berbentuk sebuah dokumen yang terstruktur seperti *XML*. Ada tiga proses yang dilakukan dalam kegiatan *wrapping* [13], yaitu menerima halaman web, kemudian mengekstrak informasi dari halaman web, dan yang langkah terakhir adalah menempatkan informasi yang telah diekstrak kedalam bentuk *XML*.

Walaupun demikian sudah banyak halaman web yang memiliki sumber informasi dari data yang terstruktur, seperti *Web Service* dan Database [14]. Tetapi tidak mendukung banyak informasi pendukung yang dapat mempermudah proses pencarian dengan menggunakan metode semantic.

Pemetaan dari informasi yang telah dipecah dari sebuah halaman web masih ada yang menggunakan DTD untuk menempatkan informasi pada posisi yang tepat, terkadang penentuan posisi yang tepat menggunakan DTD masih membutuhkan proses yang lama dan merupakan proses yang sulit. Setelah pemetaan dan penempatan dilakukan dengan benar, maka informasi tersebut dapat ditampilkan ke halaman web baru atau ke database sebagai sebuah

informasi yang berguna atau sesuai dengan yang di inginkan *user*.

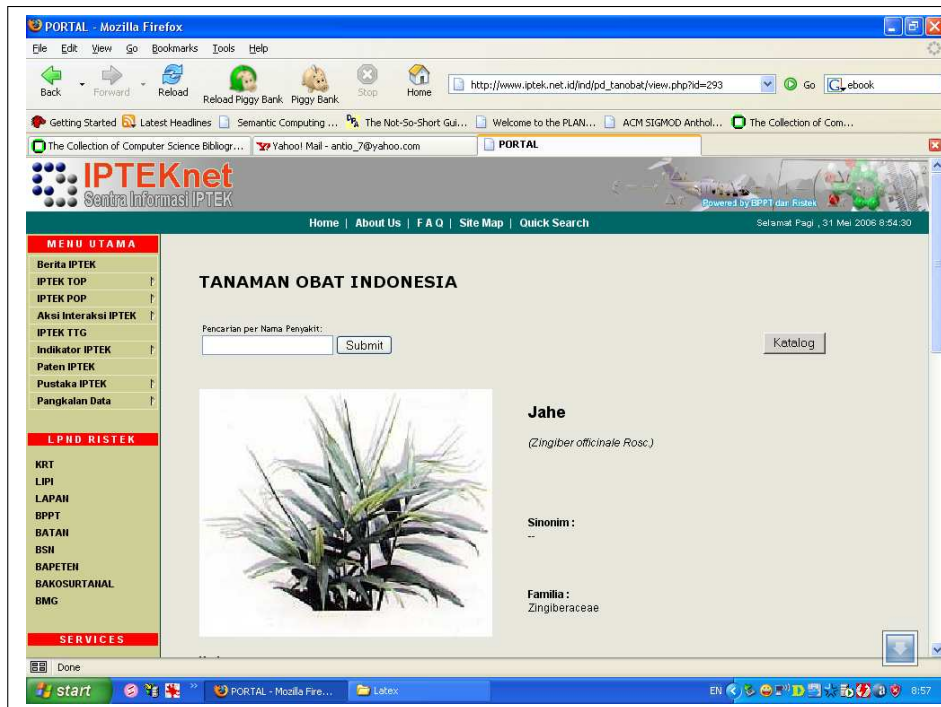
Hal tersebut di atas baru mendapatkan informasi dari sebuah halaman web, bagaimana jika informasi tersebut ingin didapatkan dari beberapa halaman yang memiliki isi yang menyerupai atau sama? Secara tidak langsung dibutuhkan sebuah penghubung atau jika dalam sebuah model data RDBMS disebut dengan *Relationship*, untuk dapat mencari hubungan yang dimiliki oleh beberapa halaman web atau sumber informasi yang memiliki isi atau kandungan informasi yang sama.

5.2 Contoh Implementasi

Baru dari segi penjabaran informasi sudah memiliki perbedaan, sedangkan dari segi penafsiran juga memiliki kendala yang cukup kompleks dan dapat berakibat fatal. Sebagai ilustrasi akan diperlihatkan contoh Interoperabilitas informasi dalam bidang tumbuhan.

Ada dua buah perusahaan yang berbeda tujuan bisnisnya, masing-masing menampilkan informasi yang dimilikinya dengan cara mereka masing-masing. Sebagai contoh akan diperlihatkan sebuah perusahaan industri obat-obatan akan memproduksi sebuah tipe obat baru, dan yang menjadi sumber bahan bakunya adalah tanaman obat. Tanaman obat tersebut yang akan diambil sebagai bahan baku obatnya adalah tanaman jahe. Perusahaan tersebut membutuhkan data-data mengenai tanaman jahe tersebut, apakah itu data mengenai kandungan kimia, kegunaan atau khasiat tanaman obat tersebut, harga, serta dimana memperolehnya untuk jumlah yang banyak. Ada beberapa Web yang menyediakan informasi tersebut, antara lain informasi tentang kegunaan dan kandungan dari tanaman jahe adalah dari *http : //www.iptek.net.id/ind/pd_tanobat/view.php?id = 147*, halaman web tersebut dapat dilihat pada

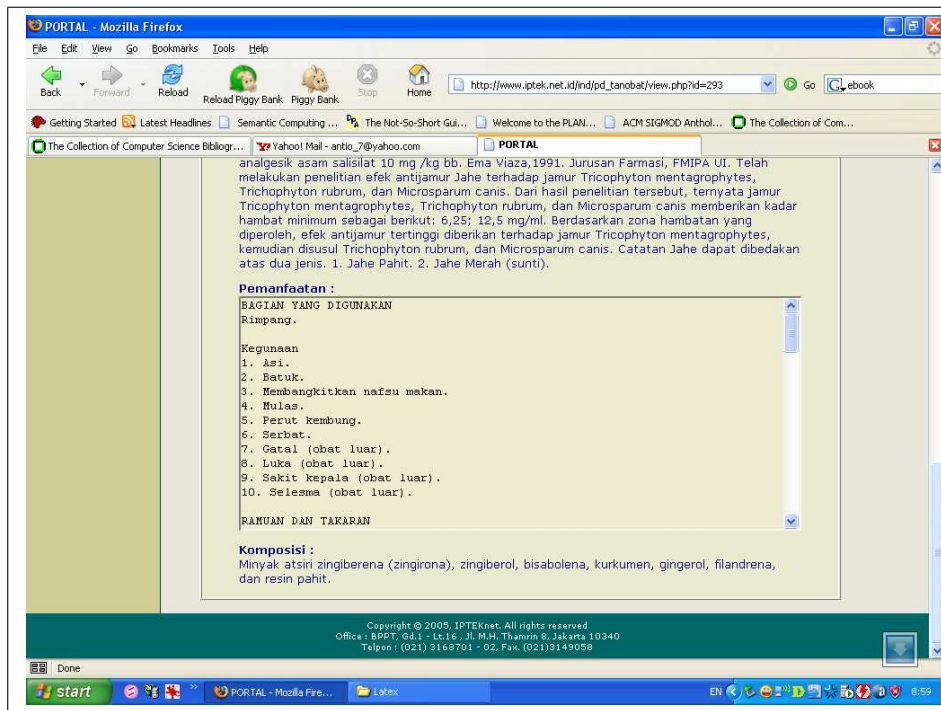
gambar 5.2 dan 5.2



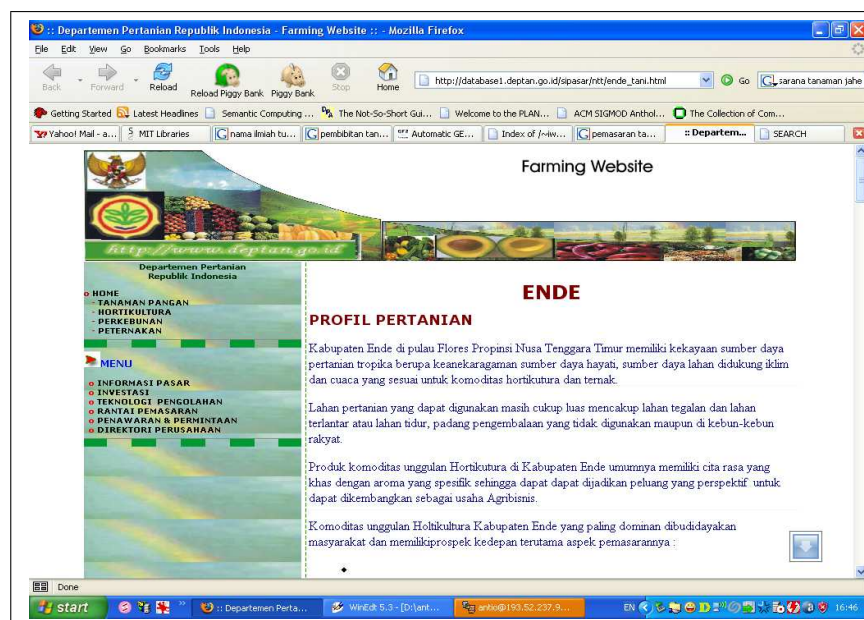
Gambar 5.1: Halaman web tentang Tanaman Jahe, H1. Dari [7]

Sedangkan untuk yang menampilkan informasi tentang harga salah satunya terdapat pada halaman web yang dibuat oleh Departemen Pertanian Republik Indonesia pada http://database1.deptan.go.id/sipasar/ntt/ende_tani.html. Halaman Web dari Departemen Pertanian tersebut dapat dilihat pada gambar 5.3 dan 5.4

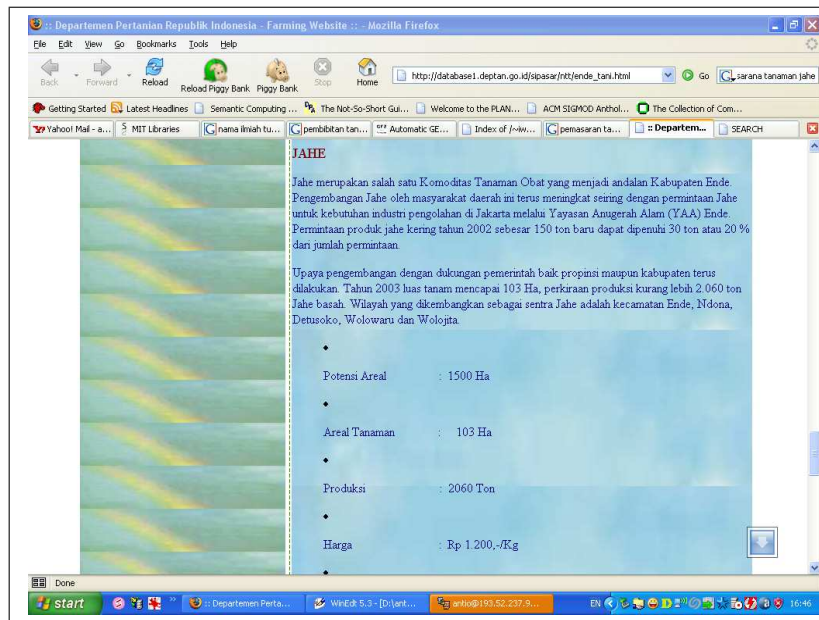
Kalau dilihat dari kedua halaman web tersebut terdapat perbedaan yang cukup mendasar yaitu terdapat perbedaan cara penyampaian informasi yang dimilikinya, selain perbedaan inti subjek yang ingin ditampilkan. Pada halaman web Departemen Pertanian komoditas jahe hanya ditampilkan tanaman jahe tanpa ada nama ilmiah, sedangkan pada halaman web yang dibuat oleh BPPT komoditas jahe selain nama latinnya juga terdapat nama ilmiahnya. Dan terdapat perbedaan dalam hal pemahaman tentang komoditas jahe itu



Gambar 5.2: Halaman web tentang Tanaman Jahe, H2. Dari [7]



Gambar 5.3: Halaman web tentang Harga Jahe, H1. Dari [12]



Gambar 5.4: Halaman web tentang Harga Jahe, H2. Dari [12]

sendiri, pada halaman web Departemen Pertanian komoditas jahe dianggap sebagai salah satu tanaman obat yang menjadi komoditas industri sedangkan dalam halaman web dari BPPT komoditas jahe dianggap sebagai tanaman obat saja.

Selain masalah tersebut di atas masih ada permasalahan yang timbul dengan adanya keragaman informasi dan penerapannya dalam dunia internet yang interoperabilitas. Dari contoh di atas yaitu dari kedua halaman web tersebut terdapat perbedaan, jika pada halaman web yang dibuat oleh IPTEK merupakan web yang semi terstruktur, karena data-data yang ditampilkan tersusun berdasarkan urutan yang teratur. Sedangkan pada halaman web yang dibuat oleh Departemen Pertanian menerapkan tampilan informasi yang tidak terstruktur, karena di dalamnya tidak diuraikan penjelasan atau keterangan perbedaan data-data yang ditampilkan.

Informasi yang diinginkan oleh perusahaan tadi merupakan gabungan

dari kedua halaman web tersebut. Untuk mendapatkan informasi yang diinginkan, perusahaan tersebut harus melakukan pemilihan dan menampilkan informasi tersebut yang sesuai dengan yang dibutuhkan oleh perusahaan tersebut.

Bagaimana caranya agar informasi tersebut didapatkan dari kedua sumber tersebut? Itu baru dari dua sumber, bagaimana kalau dari banyak sumber seperti yang terdapat di dunia maya. Belum lagi dengan adanya perbedaan penggunaan bahasa yang juga harus dipertemukan persamaannya.

Pengambilan informasi yang dibutuhkan tersebut jika dilakukan dengan cara tradisional dengan menggunakan query RDBMS biasa akan menemukan banyak kesulitan karena perbedaan dalam penggunaan istilah tadi dan tidak adanya keterhubungan antara web yang memiliki informasi yang relevan tersebut.

Metode yang dianggap paling baik dalam hal pengambilan informasi, paling baik disini adalah yang mendekati tepat informasi yang diduplikatnya dan dengan waktu yang relatif cepat, adalah menggunakan metode Semantic. Semantic menggunakan teknologi *ontology* untuk mendapatkan hubungan antara domain-domain yang berhubungan.

Proses pengambilan informasi dari kedua halaman web tersebut baik yang menerapkan data terstruktur atau yang tidak terstruktur dapat menggunakan Wrapper. Wrapper tersebut menggunakan struktur *ontology* yang telah ditetapkan sebelumnya. Di dalam web tersebut juga terdapat beberapa informasi yang dapat menyebabkan kerancuan atau kebingungan bagi komputer dalam memproses query yang dikirimkan oleh user, contohnya perbedaan kandungan dan komposisi dari tanaman jahe tersebut. Untuk menghindari kerancuan tersebut digunakan teknologi semantic yang gunanya mengenali

pengertian dan pemahaman dari suatu data agar dapat di sesuaikan dengan kebutuhan dari user tersebut.

Di dalam sebuah database terdapat berbagai pemahaman mengenai struktur data itu sendiri, dalam RDBMS *field* dikenal sebagai sebuah kolom yang berisikan data. Sedangkan dalam *ontology* kolom dikenal dengan nama *class*. Perbedaan pemahaman ini dapat juga membuat bingung orang yang akan menekuni Web Semantic khususnya yang menggunakan *ontology* sebagai model pengembangannya.

5.3 Pemanfaatan Hasil *Ontology* Tumbuhan

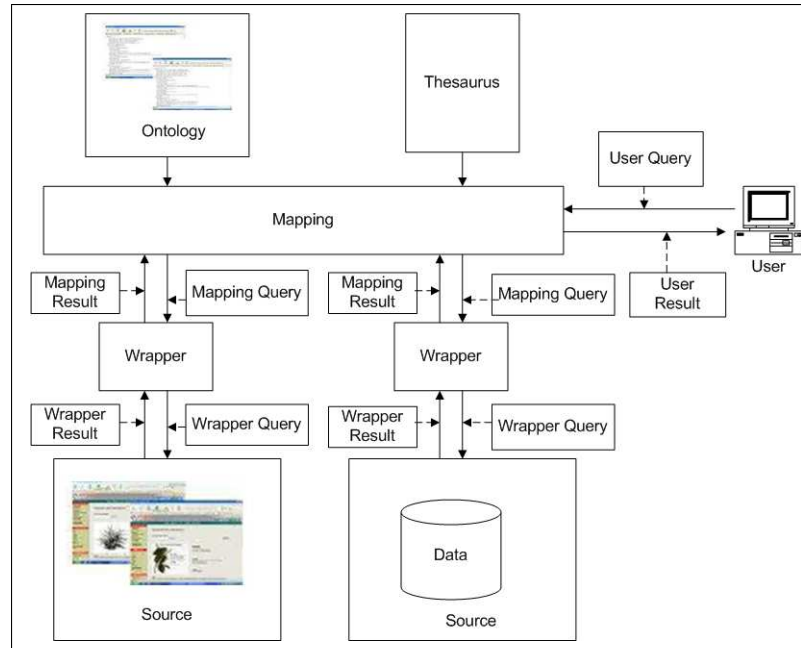
Dalam bidang botani banyak yang dapat menjadi sumber informasi bagi orang-orang yang membutuhkan. Hampir sebagian besar membutuhkan informasi tentang botani, khususnya bidang akademik yang sangat membutuhkan informasi tentang tumbuhan.

Ontology tumbuhan yang menjadi bahan praktik dapat menjadi salah satu domain dari seluruh domain tentang tumbuhan. Tumbuhan dapat terdiri dari berbagai macam domain yang masing-masing memiliki satu sumber yaitu tumbuhan atau *plant*.

Domain tersebut dapat lebih terlihat fungsinya jika digabungkan dengan domain lain yang masih memiliki satu area tumbuhan. Dari domain yang dibuat untuk bahan uji coba jika diletakkan disebuah URI (*Universal Resource Identifier*) dapat dipergunakan oleh orang lain yang ingin menggunakan *ontology* tumbuhan tersebut sebagai domain *ontology*.

Secara umum skema yang menggambarkan pemanfaatan domain tumbuhan yang digunakan untuk ujicoba tersebut, digambarkan pada gambar 5.5, skema ini menggunakan metode *Semantic Web* dengan *ontology* sebagai basis

teknologinya.



Gambar 5.5: Skema Semantic Web Domain Tumbuhan

Dalam skema tersebut digambarkan, User yang ingin mendapatkan sebuah informasi tentang tumbuhan, yang perlu dia lakukan hanyalah mengisi kriteria apa yang diinginkan. Kemudian *Web Client* akan mengirimkan kriteria tersebut ke sebuah server, di dalam server tersebut akan dibuat query untuk mendapatkan domain yang cocok dengannya. Query tersebut akan dilempar ke sebuah URI yang di dalamnya terdapat *ontology* yang isinya merupakan domain dari tumbuhan.

Setelah domain yang tepat didapatkan maka hasil query tersebut akan digunakan untuk melakukan *Wrapping* terhadap halaman web yang memiliki isi yang relevan dengan hasil query, untuk dapat melakukan *Wrapping* dibutuhkan Annotation di setiap halaman web tersebut agar proses *Wrapping* dapat menghasilkan informasi yang dibutuhkan.

Setelah informasi yang diinginkan dapat maka langkah selanjutnya adalah

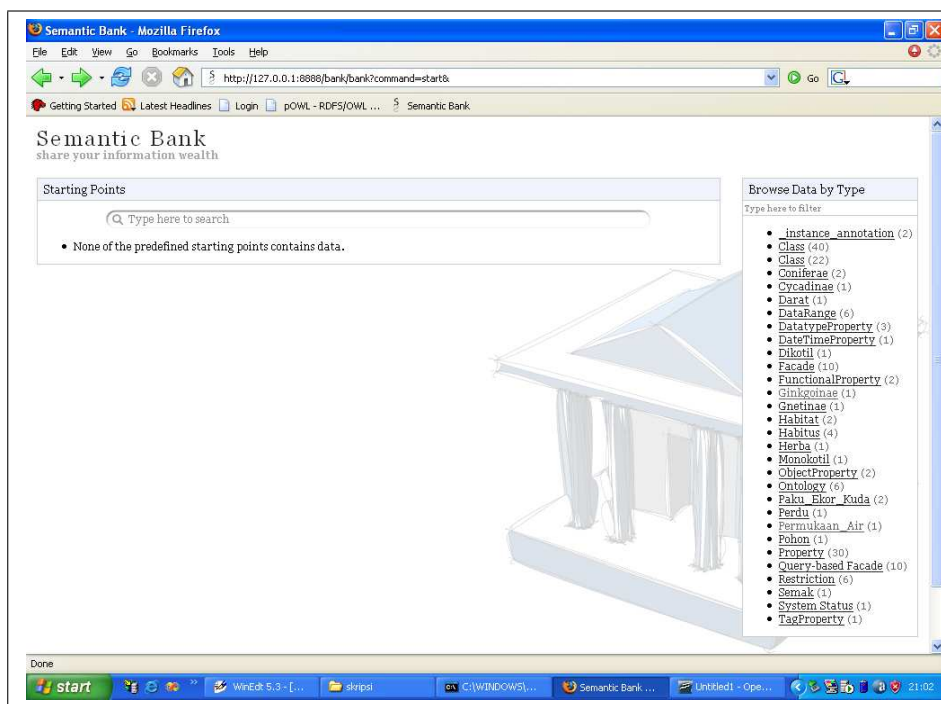
memetakan informasi tersebut kedalam sebuah struktur XML. Struktur XML tersebut yang nantinya akan menjadi bahan informasi baru yang telah terstruktur agar dapat dengan mudah ditampilkan kedalam halaman web baru.

Sebelum proses pengambilan informasi di atas dilakukan ada beberapa proses yang dilakukan terlebih dahulu, yaitu pembuatan skema export dari berbagai macam sumber informasi, membuat wrapper yang nantinya digunakan untuk mengambil informasi dair dalam halaman web, kemudian memetakan kedalam *Knowledge Ontology* dengan menggunakan Thesaurus.

Sebagai ujicoba pemanfaatan *ontology* tumbuhan akan digunakan sebuah project dari SIMILE [17], *tool* yang digunakan dari SIMILE tersebut adalah Semantic Bank. Semantic Bank merupakan sebuah *tool* yang bebas untuk didapatkan, dan Semantic Bank merupakan sebuah web server dengan menggunakan fungsi *ontology* yang di query untuk menghasilkan data yang yang di inginkan, Semantic Bank hampir menyerupai sebuah browser sebuah *ontology*.

Untuk dapat menggunakan Semantic Bank dengan *ontology* tumbuhan yang telah dibuat, *ontology* yang digunakan dapat menggunakan *ontology* yang dibuat dengan Protégé, Altova Semantic Work, atau SWOOP. Untuk Penggunaan kali ini akan digunakan *ontology* yang dibuat dengan *tool* Protégé. *Ontology* yang dibuat dengan menggunakan Protégé diletakkaa disebuah folder yang terdapat dalam Semantic Bank, untuk ujicoba ini digunakan *browser* Firefox dari Mozilla. Setelah dijalankan akan terlihat seperti gambar 5.6

Di dalam contoh yang digunakan tersebut terdapat sebuah *text box* untuk mengisi kriteria yang diinginkan. Setelah diisi maka kemudian akan tampil informasi yang relevan dengan yang diinginkan tersebut. Dalam contoh tersebut tidak dilakukan pengambilan informasi dari berbagai halaman web, tetapi hanya informasi yang terdapat dalam *RDF/OWL*.



Gambar 5.6: Penggunaan Ontology Tumbuhan dengan Semantic Bank

Bab 6

Penutup

6.1 Ringkasan dan Kesimpulan

Dengan semakin berkembangnya informasi menjadi lebih beragam atau yang disebut dengan Interoperabilitas informasi, maka penggunaan metode tradisional akan dirasakan semakin kurang dirasakan hasilnya, karena metode pencarian tradisional tidak dapat mengikuti perkembangan informasi dalam dunia internet yang semakin berkembang dengan cepat dan memiliki penjabaran masing-masing.

Salah satu metode yang saat ini dirasakan dapat mengatasi masalah yang dihadapi jika masih menggunakan metode tradisional, adalah penggunaan metode *semantic*. Walaupun metode *semantic* tersebut masih terus berkembang dan metode pengembangannya masih belum matang seperti metode tradisional, karena metode *semantic* mulai dikembangkan pada tahun 1995, hal ini sangat berbeda dengan metode tradisional yang telah memiliki model pengembangan dan kemampuan yang sudah tidak diragukan lagi.

Salah satu teknologi yang dikembangkan untuk menunjang penggunaan metode *semantic* adalah penggunaan *ontology* sebagai dasar struktur informasi yang digunakan untuk proses pencarian sumber-sumber informasi yang relevan dengan yang diinginkan. Sama seperti metode *semantic*, teknologi *ontology* juga masih belum memiliki metode pengembangan yang pasti. Setiap pengembang memiliki metode atau cara tersendiri untuk mengembangkan dan membuat sebuah aplikasi yang menggunakan metode *semantic* dan juga termasuk *ontology*.

Karena hal tersebut, maka pengembangan *tool* untuk mengembangkan *ontology* belum memiliki metode standar. Dan setiap orang bebas mengelu-

arkan atau membuat *tool-tool* yang dapat digunakan oleh setiap orang untuk pengembangan *ontology*. Sekarang yang bisa dilakukan hanyalah sebatas membandingkan kemampuan dari masing-masing *tool* tersebut untuk agar pengembangan *ontology* dan juga termasuk pengembangan metode *semantic* menjadi lebih mudah.

Dengan alasan tersebut maka dilakukan perbandingan terhadap tiga *tool* pengembangan *ontology*, yaitu Protégé, Altova Semantic Work, dan SWOOP, dalam kemampuannya untuk pembuatan sebuah *ontology* dengan mengambil domain "Tumbuhan". Setelah dilakukan uji coba, maka masing-masing *tool* memiliki kelebihan dan kekurangan masing-masing.

Dari hasil penilaian yang dilakukan kepada masing-masing *tool* dan dengan kriteria yang sama, dapat diambil kesimpulan bahwa *tool* yang menjadi peringkat pertama adalah *tool* Protégé yang memiliki total nilai 14,5. Sedangkan untuk peringkat kedua adalah Altova Semantic Work, dengan nilai total 10. Dan untuk yang menempati peringkat ke tiga adalah *tool* SWOOP dengan nilai total 7.

Dengan hasil penilaian tersebut, maka direkomendasikan untuk menggunakan Protégé sebagai *tool* untuk pengembangan *ontology*. Selain itu juga direkomendasikan untuk pilihan kedua adalah Altova Semantic Work, walaupun Altova Semantic Work merupakan *tool ontology* yang memiliki lisensi berbayar tapi fungsi dan fasilitas yang disediakan termasuk lengkap dan metode yang digunakan untuk pengembangan *ontology* dengan Altova Semantic Work adalah yang berbasis grafik atau gambar.

6.2 Rencana Kedepan

Dengan dilakukannya penilaian terhadap *tool* pengembangan dan pembuatan *ontology* diharapkan dapat memberikan referensi bagi orang-orang yang akan membuat *ontology*, walaupun *tool* yang diuji hanya tiga.

Pada pengujian ini hanya dilakukan pengujian teknis terhadap masing-masing *tool* tersebut, tetapi belum dilakukan pengujian terhadap proses implementasi pembuatan *ontology* tersebut. Diharapkan untuk kedepan diharapkan dapat dilakukan penelitian mengenai proses perancangan dan pembuatan *ontology*.

Pengujian terhadap *tool* pengembangan *ontology* dapat juga dilakukan dengan melakukan penilaian dari segi proses pengembangan dari sebuah *ontology* ke sebuah web semantic, walaupun masih web semantic yang dikembangkan pada langkah awal.

Lebih bagus jika juga dilakukan pengujian mengenai proses pemeliharaan dan perubahan dari *ontology* yang sudah tersedia. Walaupun belum terlalu banyak *tool* yang dapat menjadi bahan uji coba tetapi ada beberapa *tool* yang dapat dijadikan sebagai bahan ujicoba tambahan yaitu KAON ([http : //kaon.semanticweb.org/](http://kaon.semanticweb.org/)) atau KIT ([http : //www.caboodlenetworks.com/](http://www.caboodlenetworks.com/)) selain daripada tiga *tool* yang sudah diuji di atas.

Walaupun pada pengujian ini hanya dilakukan pengujian secara teknis perangkat lunak *tool ontology*, tetapi secara langsung dapat memberikan panduan untuk melakukan pemilihan *tool* mana yang terbaik dari segi teknis perangkat lunak.

Bibliografi

- [1] Altova. *http : //www.altova.com/products_semanticworks.html*, 2005.
- [2] A Barnaras, L Laresgoiti, and J Corera. Building and Reusing Ontologies for Electical Network Application. In *12th European Conference on Artificial Intelligence*, pages 298–302, 1996.
- [3] V Richard Benjamins and Assunción Gómez-Pérez. Knowledge System Technology: Ontologies and Problem-Solving Methods. 5 2004. <www.swi.psy.uva.nl/usr/richard/pdf/kais.pdf>.
- [4] Willem Nico Borst. *Construction of Engineering Ontologies for Knowledge Sharing and Reuse*. PhD thesis, University of Twente, Netherland, 5 September 1997. SIKS The Dutch Graduate School.
- [5] Óscar Corcho, Mariano Fernández-López, and Asunción Gómez-Pérez. Methodologies, tools and languages for building ontologies: Where is their meeting point? *Data Knowl. Eng*, 46(1):41–64, 2003.
- [6] Michael C Daconta, Leo J Obrst, and Kevin T Smith. A Guide to the Future of XML, Web Services, and Knowledge Management. Wiley Publishing, Indianapolis, Indiana, 2003.
- [7] Badan Pengkajian dan Penerapan Teknologi. *http : //www.iptek.net.id/ind/pd_tanobat/*, 2006.
- [8] Racer Systems GmbH and Co. KG. *http : //www.racer – systems.com/*, 2005.
- [9] Asuncion Gomez-Perez. Ontological Engineering (Slide). In *Proc. of IJCAI'99: Tutorial on Ontological Engineering*, 1999.

- [10] T Gruber. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. *Int. Journal of Human-Computer Studies*, 43:907–928, 1995.
- [11] N. Guarino and P. Giaretta. *Ontologies and Knowledge Bases: Towards a Terminological Clarification*, chapter Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing, pages 25–32. IOS Press, Amsterdam, 1995.
- [12] Departemen Pertanian Republik Indonesia.
http://database1.deptan.go.id/sipasar/ntt/ende_tani.html, 2004.
- [13] Sabine Jabbour and Anne-Marie Vercoistre. Wrapping Web Pages into xml Documents. *CMIS Technical Report 01/199*, 2001.
- [14] Craig A Knoblock. Bringing Semantics to the Web. 2005.
- [15] Vladimir Kolovski and John Galletly. Towards E-Learning via the Semantic Web. In *International Conference on Computer Systems and Technologies-CompSysTech'2003*, page 2, 2003.
- [16] Natalya F. Noy and Deborah L. McGuinness. *Ontology Development 101: A Guide to Creating Your First Ontology*, 2000.
- [17] Massachusetts Institute of Technology.
<http://simile.mit.edu/semantic-bank/>, 2004.
- [18] Sean B Palmer. <http://infomesh.net/2001/swintro/>, 2001.
- [19] Tim Pandu. <http://www.pandu.org/solusi/office-application/>, 1999.
- [20] Protege. <http://protege.stanford.edu/>, 2005.

- [21] T. Finin T. R. Gruber T. Senator R. Neches, R. E. Fikes and W. R. Swartout. Enabling Technology for Knowledge Sharing. 1991. AI Magazine.
- [22] V. R. Benjamins R. Studer and D. Fensel. *Knowledge Engineering, Principles and Methods.*, chapter Data and Knowledge Engineering, pages 25(1–2):161–197. 1998.
- [23] York Sure and Rudi Studer. Towards the Semantic Web: Ontology-driven Knowledge Management, 2003.
- [24] SWOOP and SMORE. <http://www.mindswap.org/2004/swoop/> and <http://www.mindswap.org/2004/smored/>, 2006.
- [25] Gembong Tjitrosoepomo. *Taksonomi Tumbuhan : schizophyta, thallophyta, bryophyta, pteridophyta.* Yogyakarta : Universitas Gadjah mada Press, 2005.
- [26] K. Knight W. Swartout, R. Patil and T. Russ. *Toward Distributed use of large-scale Ontologies.*, chapter Spring Symposium Series on Ontological Engineeringg, pages 33–40. AAAI Press, 1997.
- [27] W3C. <http://www.w3.org/tr/2002/wd-rdf-schema-20020430/>, 2 2006.
- [28] W3C. <http://www.w3.org/tr/2004/rec-rdf-syntax-grammar-20040210/>, 2 2006.
- [29] H. Wache, T. Vogele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hubner. Ontology-based Integration of Information

- a Survey of Existing Approaches. In *Proceedings of IJCAI-01 Workshiop: Ontologies and Informaton Sharing*, pages 108–117, Seattle, WA, USA.
- [30] I Wayan Simri Wicaksana. Survei dan Evaluasi Metode Pengembangan Ontologi (Survey and Evaluation of Methodology of Ontology Development). In *Proc. of KOMMIT 2004*, Jakarta&Depok, 24 2004. University Gunadarma.
- [31] I Wayan Simri Wicaksana. *A Peer-to-Peer (P2P) Based Semantic Agreement Approach for Spatial Information Interoperability*. Doctor of information technology, University Gunadarma, Jl. Margonda Raya 100, Depok, Indonesia, 23 2006.
- [32] WikiPedia. <http://en.wikipedia.org/wiki/semanticweb>, 2006.

=====

LAMPIRAN

Sintaksis *OWL* untuk Domain Tumbuhan L1

```

<?xml version="1.0"?> <!DOCTYPE rdf:RDF [
  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>
<rdf:RDF xmlns="http://www.owl-ontologies.com/Tumbuhan.owl#"
  xml:base="http://www.owl-ontologies.com/Tumbuhan.owl"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:owl="http://www.w3.org/2002/07/owl#">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Angiospermae">
    <rdfs:subClassOf rdf:resource="#Spermatophyta"/>
    <owl:disjointWith rdf:resource="#Gymnospermae"/>
    <rdfs:label rdf:datatype="&xsd:string">Angiospermae</rdfs:label>
    <rdfs:comment rdf:datatype="&xsd:string"
      >Tumbuhan Berbiji Terbuka</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="Anthocerotopsida">
    <rdfs:subClassOf rdf:resource="#non-Tracheophyta"/>
    <owl:disjointWith rdf:resource="#Bryopsida"/>
    <owl:disjointWith rdf:resource="#Hepaticopsida"/>
    <rdfs:label rdf:datatype="&xsd:string"
      >Anthocerotopsida</rdfs:label>
    <rdfs:comment rdf:datatype="&xsd:string"

```

```

        >Lumut yang memiliki tubuh seperti talus
        tetapi hanya memiliki satu kloroplas</rdfs:comment>
</owl:Class>
<owl:Class rdf:ID="Bryopsida">
    <rdfs:subClassOf rdf:resource="#non-Tracheophyta"/>
    <owl:disjointWith rdf:resource="#Hepaticopsida"/>
    <owl:disjointWith rdf:resource="#Anthocerotopsida"/>
    <rdfs:label rdf:datatype="xsd:string">Bryopsida</rdfs:label>
    <rdfs:comment rdf:datatype="xsd:string"
        >Tumbuhan Lumut yang dapat dibedakan antara batang,
        daun dan akar</rdfs:comment>
</owl:Class>
<owl:Class rdf:ID="Coniferae">
    <rdfs:subClassOf rdf:resource="#Gymnospermae"/>
    <owl:disjointWith rdf:resource="#Cycadinae"/>
    <owl:disjointWith rdf:resource="#Ginkgoinae"/>
    <owl:disjointWith rdf:resource="#Gnetinae"/>
    <rdfs:label rdf:datatype="xsd:string">Coniferae</rdfs:label>
    <rdfs:comment rdf:datatype="xsd:string"
        >Tumbuhan yang memiliki tajuk berbentuk kerucut dan
        daun berbentuk jarum</rdfs:comment>
</owl:Class>
<owl:DatatypeProperty rdf:ID="Contoh">
    <rdfs:domain rdf:resource="#Tumbuhan"/>
    <rdfs:range rdf:resource="xsd:string"/>
    <rdfs:label rdf:datatype="xsd:string">Contoh</rdfs:label>
    <rdfs:comment rdf:datatype="xsd:string"
        >Untuk memberikan contoh kepada amasing-masing
        Class</rdfs:comment>
</owl:DatatypeProperty>
<owl:Class rdf:ID="Cycadinae">
    <rdfs:subClassOf rdf:resource="#Gymnospermae"/>
    <owl:disjointWith rdf:resource="#Gnetinae"/>
    <owl:disjointWith rdf:resource="#Coniferae"/>
    <owl:disjointWith rdf:resource="#Ginkgoinae"/>
    <rdfs:label rdf:datatype="xsd:string">Cycadinae</rdfs:label>
    <rdfs:comment rdf:datatype="xsd:string"

```

```

        >Tumbuhan yang menyerupai Palem dan daun yang tersusun
        dalam roset batang</rdfs:comment>
</owl:Class>
<Coniferae rdf:ID="Coniferae_27">
    <Contoh rdf:datatype="&xsd:string">damar</Contoh>
    <Mempunyai_Habitat rdf:resource="#Habitat_16"/>
    <Mempunyai_Habitus rdf:resource="#Habitus_12"/>
</Coniferae>
<Habitat rdf:ID="Habitat_16">
    <Jenis_Habitat rdf:datatype="&xsd:string">Darat</Jenis_Habitat>
</Habitat>
<owl:Class rdf:ID="Dikotil">
    <rdfs:subClassOf rdf:resource="#Angiospermae"/>
    <owl:disjointWith rdf:resource="#Monokotil"/>
    <rdfs:label rdf:datatype="&xsd:string">Dikotil</rdfs:label>
    <rdfs:comment rdf:datatype="&xsd:string"
        >Tumbuhan Yang di Dalamnya memiliki dua keping
        biji</rdfs:comment>
</owl:Class>
<Monokotil rdf:ID="Monokotil_32">
    <Contoh rdf:datatype="&xsd:string">eceng gondok</Contoh>
    <Mempunyai_Habitat rdf:resource="#Habitat_17"/>
    <Mempunyai_Habitus rdf:resource="#Habitus_15"/>
</Monokotil>
<owl:Class rdf:ID="Ekor_Kuda">
    <rdfs:subClassOf rdf:resource="#Pteridophyta"/>
    <owl:disjointWith rdf:resource="#Kawat"/>
    <owl:disjointWith rdf:resource="#Sejati"/>
    <owl:disjointWith rdf:resource="#Purba"/>
    <rdfs:label rdf:datatype="&xsd:string">Paku Ekor Kuda
        </rdfs:label>
    <rdfs:comment rdf:datatype="&xsd:string"
        >Tumbuhan Paku Ekor Kuda</rdfs:comment>
</owl:Class>
<Ginkgoinae rdf:ID="Ginkgoinae_30">
    <Contoh rdf:datatype="&xsd:string">Ginkgo biloba</Contoh>
    <Mempunyai_Habitat rdf:resource="#Habitat_16"/>

```

```

    <Memunyai_Habitus rdf:resource="#Habitus_13"/>
  </Ginkgoinae>
  <owl:Class rdf:ID="Ginkgoinae">
    <rdfs:subClassOf rdf:resource="#Gymnospermae"/>
    <owl:disjointWith rdf:resource="#Coniferae"/>
    <owl:disjointWith rdf:resource="#Cycadinae"/>
    <owl:disjointWith rdf:resource="#Gnetinae"/>
    <rdfs:label rdf:datatype="xsd:string">Ginkgoinae</rdfs:label>
    <rdfs:comment rdf:datatype="xsd:string"
      >Pohon yang menyerupai tunas pendek, daun bertangkai
      panjang dan berbentuk pasak atau kipas</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="Gnetinae">
    <rdfs:subClassOf rdf:resource="#Gymnospermae"/>
    <owl:disjointWith rdf:resource="#Ginkgoinae"/>
    <owl:disjointWith rdf:resource="#Cycadinae"/>
    <owl:disjointWith rdf:resource="#Coniferae"/>
    <rdfs:label rdf:datatype="xsd:string">Gnetinae</rdfs:label>
    <rdfs:comment rdf:datatype="xsd:string"
      >Tumbuhan berkayu yang batangnya bercabang atau
      tidak</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="Gymnospermae">
    <rdfs:subClassOf rdf:resource="#Spermatophyta"/>
    <owl:disjointWith rdf:resource="#Angiospermae"/>
    <rdfs:label rdf:datatype="xsd:string">Gymnospermae</rdfs:label>
    <rdfs:comment rdf:datatype="xsd:string"
      >Tumbuhan Berbiji Tertutup</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="Habitat">
    <owl:disjointWith rdf:resource="#Tumbuhan"/>
    <owl:disjointWith rdf:resource="#Habitus"/>
    <rdfs:label rdf:datatype="xsd:string">Habitat</rdfs:label>
    <rdfs:comment rdf:datatype="xsd:string"
      >Merupakan Tempat Berkembang biak atau tempat
      tumbuhan biasa atau dapat hidup</rdfs:comment>
  </owl:Class>

```

```

<owl:Class rdf:ID="Habitus">
  <owl:disjointWith rdf:resource="#Habitat"/>
  <owl:disjointWith rdf:resource="#Tumbuhan"/>
  <rdfs:label rdf:datatype="xsd:string">Habitus</rdfs:label>
  <rdfs:comment rdf:datatype="xsd:string"
    >Adalah Bentuk Tumbuh Sebuah Pohon yang merupakan
      proses berkembangnya sebuah tumbuhan</rdfs:comment>
</owl:Class>
<owl:Class rdf:ID="Hepaticopsida">
  <rdfs:subClassOf rdf:resource="#non-Tracheophyta"/>
  <owl:disjointWith rdf:resource="#Anthoceratopsida"/>
  <owl:disjointWith rdf:resource="#Bryopsida"/>
  <rdfs:label rdf:datatype="xsd:string">Hepaticopsida</rdfs:label>
  <rdfs:comment rdf:datatype="xsd:string"
    >Lumut Yang memiliki Ciri Tubuh Berbentuk Talus</rdfs:comment>
</owl:Class>
<Habitus rdf:ID="Habitus_15">
  <Jenis_Habitus rdf:datatype="xsd:string">Herba</Jenis_Habitus>
</Habitus>
<owl:DatatypeProperty rdf:ID="Jenis_Habitat">
  <rdfs:domain rdf:resource="#Habitat"/>
  <rdfs:range>
    <owl:DataRange>
      <owl:oneOf>
        <rdf:List>
          <rdf:first rdf:datatype="xsd:string"
            >Darat</rdf:first>
          <rdf:rest>
            <rdf:List>
              <rdf:first rdf:datatype="xsd:string">
                Permukaan_Air</rdf:first>
              <rdf:rest rdf:resource="&rdf:nil"/>
            </rdf:List>
          </rdf:rest>
        </rdf:List>
      </owl:oneOf>
    </owl:DataRange>
  </rdfs:range>
</owl:DatatypeProperty>

```

```

</rdfs:range>
<rdfs:label rdf:datatype="&xsd:string">Jenis_Habitat</rdfs:label>
<rdfs:comment rdf:datatype="&xsd:string"
  >Berbagai Macam Jenis Habitat Yang tersedia</rdfs:comment>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Jenis_Habitus">
  <rdfs:domain rdf:resource="#Habitus"/>
  <rdfs:range>
    <owl:DataRange>
      <owl:oneOf>
        <rdf:List>
          <rdf:first rdf:datatype="&xsd:string"
            >Pohon</rdf:first>
          <rdf:rest>
            <rdf:List>
              <rdf:first rdf:datatype="&xsd:string"
                >Perdu</rdf:first>
              <rdf:rest>
                <rdf:List>
                  <rdf:first rdf:datatype="&xsd:string"
                    >Semak</rdf:first>
                  <rdf:rest>
                    <rdf:List>
                      <rdf:first rdf:datatype="
                        &xsd:string"
                        >Herba</rdf:first>
                      <rdf:rest rdf:resource="
                        &rdf:nil"/>
                    </rdf:List>
                  </rdf:rest>
                </rdf:List>
              </rdf:rest>
            </rdf:List>
          </rdf:rest>
        </rdf:List>
      </owl:oneOf>
    </owl:DataRange>
  </owl:DatatypeProperty>

```

```

    </rdfs:range>
    <rdfs:label rdf:datatype="&xsd:string">Jenis_Habitus</rdfs:label>
    <rdfs:comment rdf:datatype="&xsd:string"
        >Berbagai Macam JenisHabitus Yang tersedia</rdfs:comment>
</owl:DatatypeProperty>
<owl:Class rdf:ID="Kawat">
    <rdfs:subClassOf rdf:resource="#Pteridophyta"/>
    <owl:disjointWith rdf:resource="#Sejati"/>
    <owl:disjointWith rdf:resource="#Purba"/>
    <owl:disjointWith rdf:resource="#Ekor_Kuda"/>
    <rdfs:label rdf:datatype="&xsd:string">Paku Kawat</rdfs:label>
    <rdfs:comment rdf:datatype="&xsd:string"
        >Tumbuhan Paku Kawat</rdfs:comment>
</owl:Class>
<Gnetinae rdf:ID="Gnetinae_29">
    <Contoh rdf:datatype="&xsd:string">Melinjo</Contoh>
    <Memunyai_Habitat rdf:resource="#Habitat_16"/>
    <Memunyai_Habitus rdf:resource="#Habitus_14"/>
</Gnetinae>
<owl:ObjectProperty rdf:ID="Memunyai_Habitat">
    <rdfs:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:domain rdf:resource="#Tumbuhan"/>
    <rdfs:range rdf:resource="#Habitat"/>
    <rdfs:label rdf:datatype="&xsd:string"
        >Memunyai Habitat</rdfs:label>
    <rdfs:comment rdf:datatype="&xsd:string"
        >Tempat Tumbuhan Biasa berkembang biak atau hidup</rdfs:comment>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Memunyai_Habitus">
    <rdfs:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:domain rdf:resource="#Tumbuhan"/>
    <rdfs:range rdf:resource="#Habitus"/>
    <rdfs:label rdf:datatype="&xsd:string"
        >Memunyai Habitus</rdfs:label>
    <rdfs:comment rdf:datatype="&xsd:string"
        >Tumbuhan Memiliki Bentuk Tumbuh </rdfs:comment>
</owl:ObjectProperty>

```

```

<owl:Class rdf:ID="Monokotil">
  <rdfs:subClassOf rdf:resource="#Angiospermae"/>
  <owl:disjointWith rdf:resource="#Dikotil"/>
  <rdfs:label rdf:datatype="&xsd:string">Monokotil</rdfs:label>
  <rdfs:comment rdf:datatype="&xsd:string"
    >Tumbuhan Yang di Dalamnya Memiliki Satu Keping Biji</rdfs:comment>
</owl:Class>
<Dikotil rdf:ID="Dikotil_31">
  <Contoh rdf:datatype="&xsd:string">nangka</Contoh>
  <Mempunyai_Habitat rdf:resource="#Habitat_16"/>
  <Mempunyai_Habitus rdf:resource="#Habitus_12"/>
</Dikotil>
<owl:Class rdf:ID="non-Tracheophyta">
  <rdfs:subClassOf rdf:resource="#Tumbuhan"/>
  <owl:disjointWith rdf:resource="#Tracheophyta"/>
  <rdfs:label rdf:datatype="&xsd:string"
    >non-Tracheophyta</rdfs:label>
  <rdfs:comment rdf:datatype="&xsd:string"
    >Tumbuhan yang tidak memiliki pembuluh untuk
      mengangkut sari-sari makanan</rdfs:comment>
</owl:Class>
<Cycadinae rdf:ID="Cycadinae_26">
  <Contoh rdf:datatype="&xsd:string">Pakis Haji</Contoh>
  <Mempunyai_Habitat rdf:resource="#Habitat_16"/>
  <Mempunyai_Habitus rdf:resource="#Habitus_12"/>
</Cycadinae>
<Habitus rdf:ID="Habitus_13">
  <Jenis_Habitus rdf:datatype="&xsd:string">Perdu</Jenis_Habitus>
</Habitus>
<Habitat rdf:ID="Habitat_17">
  <Jenis_Habitat rdf:datatype="&xsd:string">Permukaan_Air</Jenis_Habitat>
</Habitat>
<Coniferae rdf:ID="Coniferae_28">
  <Contoh rdf:datatype="&xsd:string">Pinus</Contoh>
  <Mempunyai_Habitat rdf:resource="#Habitat_16"/>
  <Mempunyai_Habitus rdf:resource="#Habitus_12"/>
</Coniferae>

```

```
<Habitus rdf:ID="Habitus_12">
  <Jenis_Habitus rdf:datatype="&xsd:string">Pohon</Jenis_Habitus>
</Habitus>
<owl:Class rdf:ID="Pteridophyta">
  <rdfs:subClassOf rdf:resource="#Tracheophyta"/>
  <owl:disjointWith rdf:resource="#Spermatophyta"/>
  <rdfs:label rdf:datatype="&xsd:string">Pteridophyta</rdfs:label>
  <rdfs:comment rdf:datatype="&xsd:string"
    >Tumbuhan Tidak Berbiji atau Tumbuhan Paku-pakuan</rdfs:comment>
</owl:Class>
<owl:Class rdf:ID="Purba">
  <rdfs:subClassOf rdf:resource="#Pteridophyta"/>
  <owl:disjointWith rdf:resource="#Kawat"/>
  <owl:disjointWith rdf:resource="#Sejati"/>
  <owl:disjointWith rdf:resource="#Ekor_Kuda"/>
  <rdfs:label rdf:datatype="&xsd:string">Paku Purba</rdfs:label>
  <rdfs:comment rdf:datatype="&xsd:string"
    >Tumbuhan Paku Purba</rdfs:comment>
</owl:Class>
<owl:Class rdf:ID="Sejati">
  <rdfs:subClassOf rdf:resource="#Pteridophyta"/>
  <owl:disjointWith rdf:resource="#Purba"/>
  <owl:disjointWith rdf:resource="#Kawat"/>
  <owl:disjointWith rdf:resource="#Ekor_Kuda"/>
  <rdfs:label rdf:datatype="&xsd:string">Paku Sejati</rdfs:label>
  <rdfs:comment rdf:datatype="&xsd:string"
    >Tumbuhan Paku Sejati</rdfs:comment>
</owl:Class>
<Habitus rdf:ID="Habitus_14">
  <Jenis_Habitus rdf:datatype="&xsd:string">Semak</Jenis_Habitus>
</Habitus>
<owl:Class rdf:ID="Spermatophyta">
  <rdfs:subClassOf rdf:resource="#Tracheophyta"/>
  <owl:disjointWith rdf:resource="#Pteridophyta"/>
  <rdfs:label rdf:datatype="&xsd:string">Spermatophyta</rdfs:label>
  <rdfs:comment rdf:datatype="&xsd:string"
    >Tumbuhan Berbiji</rdfs:comment>
```

```

</owl:Class>
<owl:Class rdf:ID="Tracheophyta">
  <rdfs:subClassOf rdf:resource="#Tumbuhan"/>
  <owl:disjointWith rdf:resource="#non-Tracheophyta"/>
  <rdfs:label rdf:datatype="&xsd:string">Tracheophyta</rdfs:label>
  <rdfs:comment rdf:datatype="&xsd:string"
    >Tumbuhan Yang memilki Alat Pengangkut pada
      Batangnya</rdfs:comment>
</owl:Class>
<owl:Class rdf:ID="Tumbuhan">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#Mempunyai_Habitat"/>
      <owl:someValuesFrom rdf:resource="#Habitat"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#Mempunyai_Habitus"/>
      <owl:someValuesFrom rdf:resource="#Habitus"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="&owl;Thing"/>
  <owl:disjointWith rdf:resource="#Habitat"/>
  <owl:disjointWith rdf:resource="#Habitus"/>
  <rdfs:label rdf:datatype="&xsd:string">Tumbuhan</rdfs:label>
  <rdfs:comment rdf:datatype="&xsd:string"
    >Segala macam mahluk hidup yang memiliki klorofil di
      dalam daunnya yang berfungsi untuk mengubah bahan
      makanan menjadi energi</rdfs:comment>
</owl:Class>
</rdf:RDF>

```

Sintaksis Skema *Class RDF* untuk Domain Tumbuhan L2

```

<?xml version='1.0' encoding='UTF-8'?> <!DOCTYPE rdf:RDF [
  <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
  <!ENTITY a 'http://protege.stanford.edu/system#'>
  <!ENTITY rdf_ 'http://protege.stanford.edu/rdf'>
  <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
]> <rdf:RDF xmlns:rdf="&rdf;"
  xmlns:rdf_="&rdf_;"
  xmlns:a="&a;"
  xmlns:rdfs="&rdfs;">
<a:owl_class rdf:about="&rdf_;Angiospermae"
  rdfs:comment="Tumbuhan Berbiji Tertutup"
  rdfs:label="Angiospermae">
<rdfs:subClassOf rdf:resource="&rdf_;Spermatophyta"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Anthocerotopsida"
  rdfs:label="Anthocerotopsida">
<rdfs:comment>Lumut yang memiliki tubuh seperti
  talus tetapi hanya memiliki satu kloroplas</rdfs:comment>
<rdfs:subClassOf rdf:resource="&rdf_;non-Tracheophyta"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Bryopsida"
  rdfs:label="Bryopsida">
<rdfs:comment>Tumbuhan Lumut yang dapat dibedakan
  antara batang, daun dan akar</rdfs:comment>
<rdfs:subClassOf rdf:resource="&rdf_;non-Tracheophyta"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Coniferae"
  rdfs:label="Coniferae">
<rdfs:comment>Tumbuhan yang memiliki tajuk berbentuk
  kerucut dan daun berbentuk jarum</rdfs:comment>
<rdfs:subClassOf rdf:resource="&rdf_;Gymnospermae"/>
</a:owl_class> <a:owl_datatypeproperty rdf:about="&rdf_;Contoh"
  rdfs:comment="Untuk memberikan contoh kepada amasing-masing Class"
  rdfs:label="Contoh">
<rdfs:domain rdf:resource="&rdf_;Tumbuhan"/>
<rdfs:range rdf:resource="&rdfs;Literal"/>

```

```

</a:owl_datatypeproperty> <a:owl_class rdf:about="&rdf_;Cycadinae"
  rdfs:label="Cycadinae">
  <rdfs:comment>Tumbuhan yang menyerupai Palem dan
    daun yang tersusun dalam roset batang</rdfs:comment>
  <rdfs:subClassOf rdf:resource="&rdf_;Gymnospermae"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Dikotil"
  rdfs:comment="Tumbuhan Yang di Dalamnya memiliki dua keping biji"
  rdfs:label="Dikotil">
  <rdfs:subClassOf rdf:resource="&rdf_;Angiospermae"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Ekor_Kuda"
  rdfs:comment="Tumbuhan Paku Ekor Kuda"
  rdfs:label="Ekor_Kuda">
  <rdfs:subClassOf rdf:resource="&rdf_;Pteridophyta"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Ginkgoinae"
  rdfs:label="Ginkgoinae">
  <rdfs:comment>Pohon yang menyerupai tunas pendek, daun
    bertangkai panjang dan berbentuk pasak atau kipas</rdfs:comment>
  <rdfs:subClassOf rdf:resource="&rdf_;Gymnospermae"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Gnetinae"
  rdfs:comment="Tumbuhan berkayu yang batangnya bercabang atau tidak"
  rdfs:label="Gnetinae">
  <rdfs:subClassOf rdf:resource="&rdf_;Gymnospermae"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Gymnospermae"
  rdfs:comment="Tumbuhan Berbiji Tertutup"
  rdfs:label="Gymnospermae">
  <rdfs:subClassOf rdf:resource="&rdf_;Spermatophyta"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Habitat"
  rdfs:label="Habitat">
  <rdfs:comment>Merupakan Tempat Berkembang biak atau
    tempat tumbuhan biasa atau dapat hidup</rdfs:comment>
  <rdfs:subClassOf rdf:resource="&a;owl_thing"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Habitus"
  rdfs:label="Habitus">
  <rdfs:comment>Adalah Bentuk Tumbuh Sebuah Pohon
    yang merupakan proses berkembangnya sebuah tumbuhan</rdfs:comment>
  <rdfs:subClassOf rdf:resource="&a;owl_thing"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Hepaticopsida"

```

```

        rdfs:comment="Lumut Yang memiliki Ciri Tubuh Berbentuk Talus"
        rdfs:label="Hepaticopsida">
        <rdfs:subClassOf rdf:resource="&rdf_;non-Tracheophyta"/>
</a:owl_class> <a:owl_datatypeproperty
rdf:about="&rdf_;Jenis_Habitat"
        rdfs:comment="Berbagai Macam Jenis Habitat Yang tersedia"
        rdfs:label="Jenis_Habitat">
        <rdfs:domain rdf:resource="&rdf_;Habitat"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:owl_datatypeproperty> <a:owl_datatypeproperty
rdf:about="&rdf_;Jenis_Habitus"
        rdfs:comment="Berbagai Macam JenisHabitus Yang tersedia"
        rdfs:label="Jenis_Habitus">
        <rdfs:domain rdf:resource="&rdf_;Habitus"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:owl_datatypeproperty> <a:owl_class rdf:about="&rdf_;Kawat"
        rdfs:comment="Tumbuhan Paku Kawat"
        rdfs:label="Kawat">
        <rdfs:subClassOf rdf:resource="&rdf_;Pteridophyta"/>
</a:owl_class> <a:owl_objectproperty
rdf:about="&rdf_;Mempunyai_Habitat"
        rdfs:comment="Tempat Tumbuhan Biasa berkembang biak atau hidup"
        rdfs:label="Mempunyai_Habitat">
        <rdfs:range rdf:resource="&rdf_;Habitat"/>
        <rdfs:domain rdf:resource="&rdf_;Tumbuhan"/>
</a:owl_objectproperty> <a:owl_objectproperty
rdf:about="&rdf_;Mempunyai_Habitus"
        rdfs:comment="Tumbuhan Memiliki Bentuk Tumbuh "
        rdfs:label="Mempunyai_Habitus">
        <rdfs:range rdf:resource="&rdf_;Habitus"/>
        <rdfs:domain rdf:resource="&rdf_;Tumbuhan"/>
</a:owl_objectproperty> <a:owl_class rdf:about="&rdf_;Monokotil"
        rdfs:comment="Tumbuhan Yang di Dalamnya Memiliki Satu Keping Biji"
        rdfs:label="Monokotil">
        <rdfs:subClassOf rdf:resource="&rdf_;Angiospermae"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Pteridophyta"
        rdfs:comment="Tumbuhan Tidak Berbiji atau Tumbuhan Paku-pakuan"

```

```

    rdfs:label="Pteridophyta">
    <rdfs:subClassOf rdf:resource="&rdf_;Tracheophyta"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Purba"
    rdfs:comment="Tumbuhan Paku Purba"
    rdfs:label="Purba">
    <rdfs:subClassOf rdf:resource="&rdf_;Pteridophyta"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Sejati"
    rdfs:comment="Tumbuhan Paku Sejati"
    rdfs:label="Sejati">
    <rdfs:subClassOf rdf:resource="&rdf_;Pteridophyta"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Spermatophyta"
    rdfs:comment="Tumbuhan Berbiji"
    rdfs:label="Spermatophyta">
    <rdfs:subClassOf rdf:resource="&rdf_;Tracheophyta"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Tracheophyta"
    rdfs:comment="Tumbuhan Yang memilki Alat Pengangkut pada Batangnya"
    rdfs:label="Tracheophyta">
    <rdfs:subClassOf rdf:resource="&rdf_;Tumbuhan"/>
</a:owl_class> <a:owl_class rdf:about="&rdf_;Tumbuhan"
    rdfs:label="Tumbuhan">
    <rdfs:comment>Segala macam makhluk hidup yang memiliki
        klorofil di dalam daunnya yang berfungsi untuk mengubah
        bahan makanan menjadi energi</rdfs:comment>
    <rdfs:subClassOf rdf:resource="&rdf_;_:A13"/>
    <rdfs:subClassOf rdf:resource="&rdf_;_:A16"/>
    <rdfs:subClassOf rdf:resource="&a;owl_thing"/>
</a:owl_class> <a:owl_somevaluesfromrestriction
rdf:about="&rdf_;_:A13"
    rdfs:label="Mempunyai_Habitat some Habitat"/>
<a:owl_somevaluesfromrestriction rdf:about="&rdf_;_:A16"
    rdfs:label="Mempunyai_Habitus some Habitus"/>
<a:owl_class rdf:about="&rdf_;non-Tracheophyta"
    rdfs:label="non-Tracheophyta">
    <rdfs:comment>Tumbuhan yang tidak memiliki pembuluh
        untuk mengangkut sari-sari makanan</rdfs:comment>
    <rdfs:subClassOf rdf:resource="&rdf_;Tumbuhan"/>
</a:owl_class> </rdf:RDF>

```

Sintaksis *RDF* untuk Domain Tumbuhan L3

```

<?xml version='1.0' encoding='UTF-8'?> <!DOCTYPE rdf:RDF [
  <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
  <!ENTITY a 'http://protege.stanford.edu/system#'>
  <!ENTITY rdf_ 'http://protege.stanford.edu/rdf'>
  <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
]> <rdf:RDF xmlns:rdf="&rdf;"
  xmlns:rdf_="&rdf_;"
  xmlns:a="&a;"
  xmlns:rdfs="&rdfs;">
  <a:owl_ontology rdf:about="&rdf_;"
    rdfs:label="Ontology(http://www.owl-ontologies.com/Tumbuhan.owl)"/>
  <rdf_:Coniferae rdf:about="&rdf_;Coniferae_27"
    rdf_:Contoh="damar"
    rdfs:label="damar">
    <rdf_:Mempunyai_Habitat rdf:resource="&rdf_;Habitat_16"/>
    <rdf_:Mempunyai_Habitus rdf:resource="&rdf_;Habitus_12"/>
  </rdf_:Coniferae> <rdf_:Coniferae rdf:about="&rdf_;Coniferae_28"
    rdf_:Contoh="Pinus"
    rdfs:label="Pinus">
    <rdf_:Mempunyai_Habitat rdf:resource="&rdf_;Habitat_16"/>
    <rdf_:Mempunyai_Habitus rdf:resource="&rdf_;Habitus_12"/>
  </rdf_:Coniferae> <rdf_:Cycadinae rdf:about="&rdf_;Cycadinae_26"
    rdf_:Contoh="Pakis Haji"
    rdfs:label="Pakis Haji">
    <rdf_:Mempunyai_Habitat rdf:resource="&rdf_;Habitat_16"/>
    <rdf_:Mempunyai_Habitus rdf:resource="&rdf_;Habitus_12"/>
  </rdf_:Cycadinae> <rdf_:Dikotil rdf:about="&rdf_;Dikotil_31"
    rdf_:Contoh="nangka"
    rdfs:label="nangka">
    <rdf_:Mempunyai_Habitat rdf:resource="&rdf_;Habitat_16"/>

```

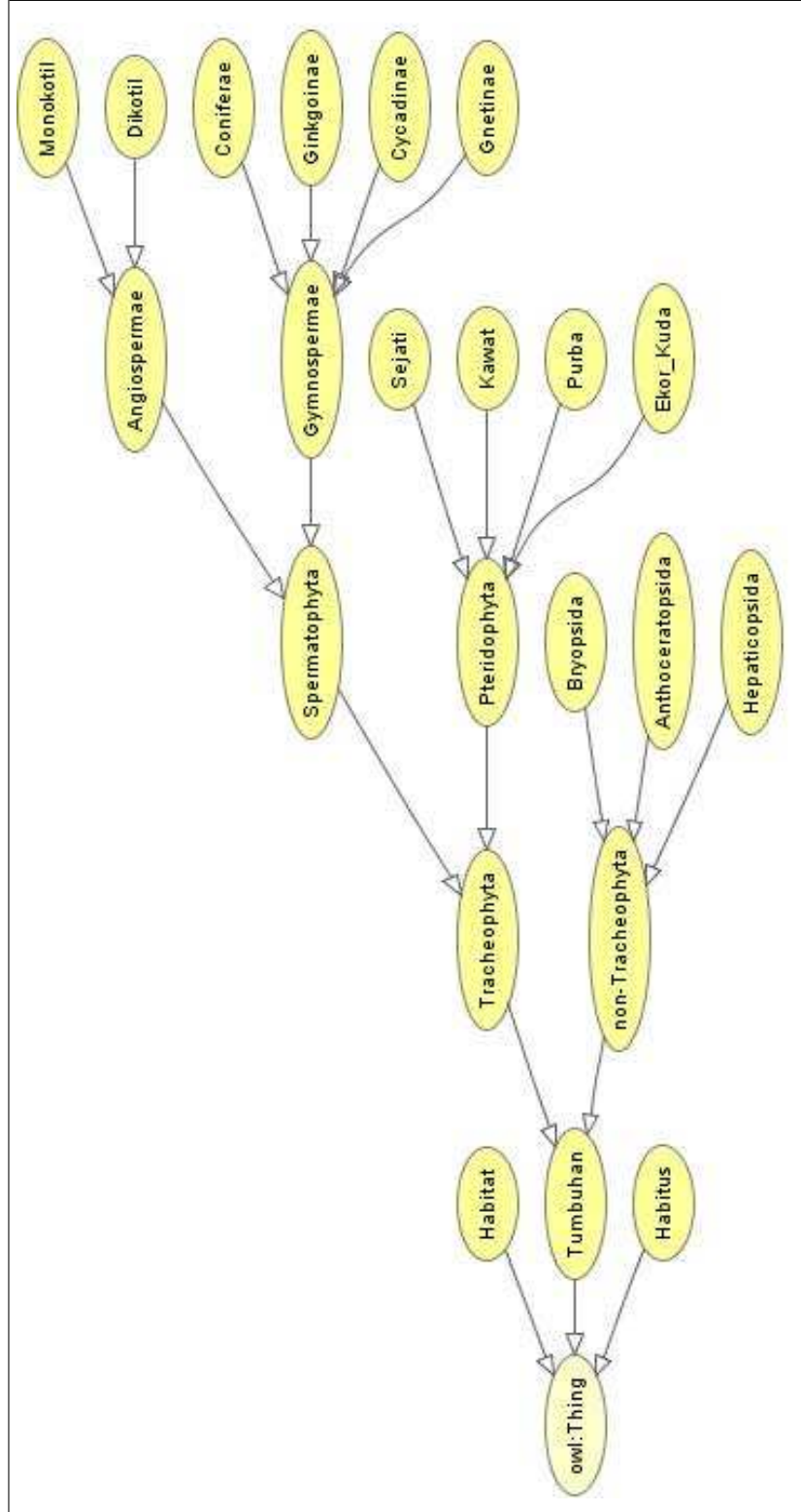
```

    <rdf_:Mempunyai_Habitus rdf:resource="&rdf_;Habitus_12"/>
</rdf_:Dikotil> <rdf_:Ginkgoinae rdf:about="&rdf_;Ginkgoinae_30"
  rdf_:Contoh="Ginkgo biloba"
  rdfs:label="Ginkgo biloba">
  <rdf_:Mempunyai_Habitat rdf:resource="&rdf_;Habitat_16"/>
  <rdf_:Mempunyai_Habitus rdf:resource="&rdf_;Habitus_13"/>
</rdf_:Ginkgoinae> <rdf_:Gnetinae rdf:about="&rdf_;Gnetinae_29"
  rdf_:Contoh="Melinjo"
  rdfs:label="Melinjo">
  <rdf_:Mempunyai_Habitat rdf:resource="&rdf_;Habitat_16"/>
  <rdf_:Mempunyai_Habitus rdf:resource="&rdf_;Habitus_14"/>
</rdf_:Gnetinae> <rdf_:Habitat rdf:about="&rdf_;Habitat_16"
  rdf_:Jenis_Habitat="Darat"
  rdfs:label="Darat"/>
<rdf_:Habitat rdf:about="&rdf_;Habitat_17"
  rdf_:Jenis_Habitat="Permukaan_Air"
  rdfs:label="Permukaan_Air"/>
<rdf_:Habitus rdf:about="&rdf_;Habitus_12"
  rdf_:Jenis_Habitus="Pohon"
  rdfs:label="Pohon"/>
<rdf_:Habitus rdf:about="&rdf_;Habitus_13"
  rdf_:Jenis_Habitus="Perdu"
  rdfs:label="Perdu"/>
<rdf_:Habitus rdf:about="&rdf_;Habitus_14"
  rdf_:Jenis_Habitus="Semak"
  rdfs:label="Semak"/>
<rdf_:Habitus rdf:about="&rdf_;Habitus_15"
  rdf_:Jenis_Habitus="Herba"
  rdfs:label="Herba"/>
<rdf_:Monokotil rdf:about="&rdf_;Monokotil_32"
  rdf_:Contoh="eceng gondok"
  rdfs:label="eceng gondok">
  <rdf_:Mempunyai_Habitat rdf:resource="&rdf_;Habitat_17"/>
  <rdf_:Mempunyai_Habitus rdf:resource="&rdf_;Habitus_15"/>
</rdf_:Monokotil> <a:owl_datarange rdf:about="&rdf_;__:A105"
  rdfs:label='owl:oneOf{"Darat" "Permukaan_Air"}'/>
<a:rdf_list rdf:about="&rdf_;__:A106"

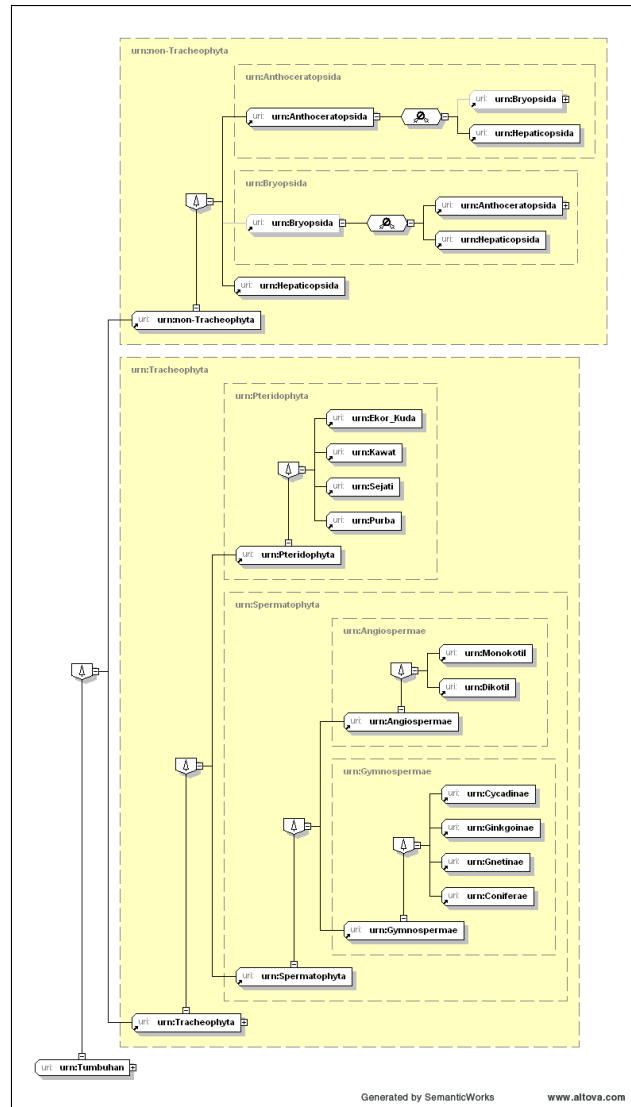
```

```
        rdfs:label="rdf:List (Darat, Permukaan_Air)"/>
<a:rdf_list rdf:about="&rdf;__:A107"
        rdfs:label="rdf:List (Permukaan_Air)"/>
<a:owl_datarange rdf:about="&rdf;__:A94"
        rdfs:label='owl:oneOf{"Pohon" "Perdu" "Semak" "Herba"}'/>
<a:rdf_list rdf:about="&rdf;__:A95"
        rdfs:label="rdf:List (Pohon, Perdu, Semak, Herba)"/>
<a:rdf_list rdf:about="&rdf;__:A96"
        rdfs:label="rdf:List (Perdu, Semak, Herba)"/>
<a:rdf_list rdf:about="&rdf;__:A97"
        rdfs:label="rdf:List (Semak, Herba)"/>
<a:rdf_list rdf:about="&rdf;__:A98"
        rdfs:label="rdf:List (Herba)"/>
</rdf:RDF>
```

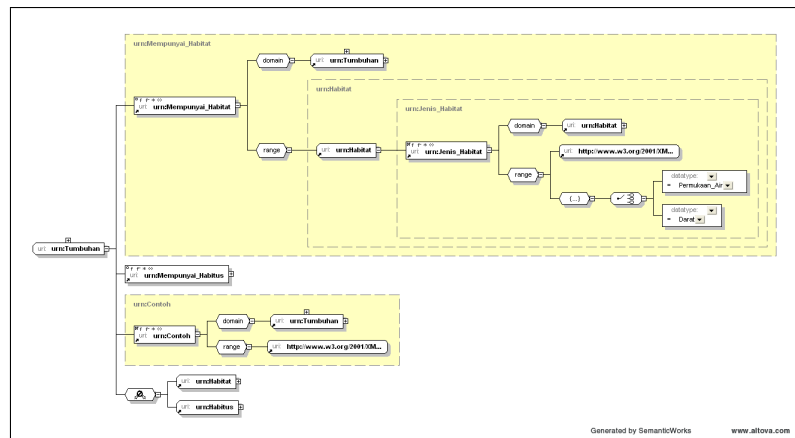
Gambar 1: Skema OWL dengan Protege L4



Gambar 2: Skema *RDF* untuk Tumbuhan dengan Altova Semantic Web L5



Gambar 3: Skema *RDF* untuk Habitat dan Habitus dengan Altova Semantic Web L6



Gambar 4: Survei *tool* Halaman 1 L7

Survei Tool Ontology
Prosedur Instalasi dan *User Interface*

Penguji : I
 Software :
 Nama :
 Versi :
 Penilaian :

- Prosedur Instalasi
 - Apakah proses instalasi dilakukan dengan ?
 - Command Line
 - GUI (Grafical User Interface)
 - Command Line dan GUI
 - Apakah ada petunjuk/keterangan setiap proses instalasi?
 - Ya
 - Tidak
 - Berapa langkah yang dibutuhkan untuk proses instalasi?
 - <5
 - 5 – 10
 - >10
 - Apakah proses instalasi dilakukan dengan otomatis atau tidak (seperti penentuan folder aplikasi) ?
 - Ya
 - Tidak

Gambar 5: Survei *tool* Halaman 2 L8

- Apakah tersedia bantuan atau Help selama proses instalasi?
 - Ya
 - Tidak
- Grafical User Interface
 - Komposisi warna ?
 - Menarik
 - Biasa
 - Buruk
 - Apakah komposisi warna dapat diatur/dirubah?
 - Ya
 - Tidak
 - Apakah letak icon mempermudah penggunaan ?
 - Ya
 - Tidak
 - Apakah gambar icon mudah dimengerti atau tidak ?
 - Ya
 - Tidak
 - Apakah tata letak window (area kerja) mempermudah penggunaan ?
 - Ya
 - Tidak
 - Apakah susunan window (area kerja) dapat diatur?
 - Ya
 - Tidak