

BAB 2

TINJAUAN PUSTAKA

Pada bab ini dijelaskan hasil tinjauan pustaka yang dilakukan penulis dalam tugas akhir ini. Hal yang dibahas adalah *e-learning*, *personalized e-learning*, teori *learning style* Felder-Silverman, *Semantic Web*, ontologi, dan TANGRAM *project*.

2.1 E-learning

E-learning adalah suatu istilah umum yang berarti pembelajaran yang dilakukan dengan komputer, biasanya terhubung ke jaringan sehingga kita dapat melakukan pembelajaran kapanpun dan dimanapun [5]. Dengan *e-learning*, proses pembelajaran tidak terbatas dengan interaksi temu muka.

E-learning bertujuan untuk mempermudah interaksi antara pembelajar dengan materi pembelajaran, pembelajar dengan pengajar, atau antara sesama pembelajar. Pembelajar dapat saling berbagi informasi dan dapat mengakses materi pembelajaran kapan pun dan berulang kali sehingga diharapkan dapat membantu pembelajar untuk lebih menguasai materi pembelajaran. Dengan adanya *e-learning*, pengajar diharapkan akan lebih mudah untuk melakukan pemutakhiran materi pembelajaran, lebih mudah untuk melakukan penelitian untuk meningkatkan wawasannya, dan lebih mudah untuk mengontrol kegiatan belajar mengajar.

E-learning memiliki berbagai tipe dan seringkali merupakan kombinasi dari tipe-tipe tersebut. *E-learning* dapat bersifat *pure online* atau *blended learning*. *Pure online* adalah *e-learning* yang tidak melibatkan proses tatap muka antara pengajar dan pembelajar atau dalam kata lain seluruh proses pembelajaran dilakukan secara *online*. Tipe *E-learning* juga dapat berupa *self study* (belajar mandiri) atau grup yang dipimpin instruktur. *Blended learning* adalah jenis *e-learning* yang

mengkombinasikan sistem *online* dan tatap muka. Jenis *E-learning* juga dapat berupa *web-based*, *computer based* (menggunakan CD atau DVD ROM), atau jenis *video/audio tape*.

2.2 Personalized E-learning

Personalized Learning adalah pembelajaran yang metode pembelajarannya dirancang untuk menyesuaikan kebutuhan dan minat dari individu pembelajar. *Personalized Learning* adalah pendekatan kepada pendidikan yang menghargai dan memperhatikan bakat unik, kemampuan, hasrat, dan atribut dari setiap pembelajar [11]. Pada *personalized learning* proses pembelajaran menggunakan metode yang dianggap paling sesuai dengan karakteristik seorang pembelajar. Dengan cara ini diharapkan proses pembelajaran akan lebih efektif dan dapat makin memaksimalkan potensi unik setiap pembelajar. Teknologi Informasi dan Komunikasi dapat menjadi medium yang sangat berguna bagi *personalized learning* oleh karena dapat membantu pembelajar untuk mengakses informasi, dan menyediakan mekanisme untuk komunikasi, debat, dan merekam hasil pembelajaran.

Personalized e-learning dapat kita definisikan sebagai sistem *e-learning* yang mendukung *personalized learning*. *Personalized e-learning* juga dapat berarti sistem *e-learning* yang memiliki fitur personalisasi metode pembelajaran untuk setiap individu pembelajar. Personalisasi memberikan pembelajar kesempatan untuk belajar dengan cara yang paling sesuai dengan gaya belajar mereka.

Berdasarkan fungsi personalisasinya, fitur dari suatu *personalized e-learning* dapat dikelompokkan menjadi tiga [9], yaitu:

1. Rekomendasi Mata Kuliah.

Personalized e-learning berfungsi sebagai sistem yang merekomendasikan mata kuliah yang paling cocok untuk dipelajari oleh pembelajar. Selain itu *personalized e-learning* juga menandai mata kuliah yang belum dapat atau

belum cocok diambil pembelajar. Mata kuliah yang belum dapat diambil itu dikarenakan pembelajar belum memenuhi prasyarat kuliah tersebut.

2. Personalisasi berdasarkan *learning style*.

Personalized e-learning berfungsi sebagai sistem yang menyajikan materi dan metoda pembelajaran berdasarkan *learning style* pembelajar. *Learning style* dianggap sebagai parameter yang penting untuk menentukan metode pembelajaran yang paling cocok untuk seorang pembelajar. Ada berbagai teori yang memodelkan *learning style*, salah satunya adalah teori *learning style* Felder-Silverman. Ontologi *user-model* pada pengerjaan tugas akhir ini menggunakan teori *learning style* Felder-Soloman. Teori *learning style* Felder-Silverman akan dijelaskan pada sub-bab 2.4.

3. Personalisasi berdasarkan *performance*

Personalized e-learning berfungsi sebagai sistem yang menyajikan materi dan metoda pembelajaran berdasarkan performa belajar. Parameter personalisasi berdasarkan *performance* ini dapat berupa nilai yang didapat dari ujian, nilai tugas, tingkat pengetahuan atas mata kuliah terkait, dan lain-lain.

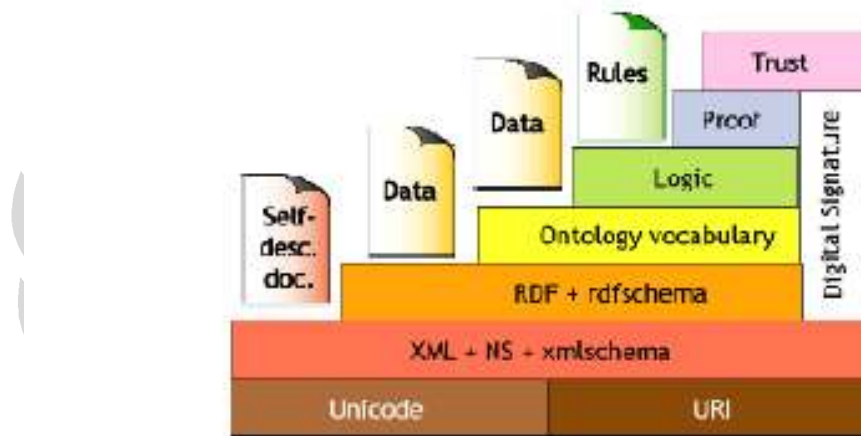
Untuk mengembangkan *personalized e-learning* diperlukan *engine* yang dapat mengerti dan mengolah informasi tentang pembelajar dan juga diperlukan *engine* yang dapat melakukan dekomposisi *learning content* dan kemudian dapat merangkai *learning content* tersebut kembali sesuai dengan personalisasi tiap pembelajar. Namun untuk membangun *engine-engine* tersebut kita perlu membangun infrastruktur informasi sehingga sistem dapat dengan mudah mengerti informasi dan relasi antar informasi. Dengan dibangun infrastruktur seperti itu diharapkan sistem lebih mudah untuk mengambil dan mengolah informasi sehingga lebih akurat dalam menyajikan personalisasi. Oleh karena itu teknologi *Semantic Web* banyak digunakan untuk pengembangan *personalized e-learning*. Teknologi *Semantic Web* didasarkan pada pemikiran akan adanya sistem *web* yang lebih cerdas yang dapat mengerti makna dan relasi dari setiap informasi seperti halnya manusia dapat mengerti makna dan relasi dari suatu informasi. *Semantic Web* akan dijelaskan lebih lanjut pada subbab 2.3.

Universitas Indonesia

2.3 Semantic Web

2.3.1 Definisi *Semantic Web*

Semantic Web adalah pemikiran Sir Tim Berners-Lee, penemu WWW, URI, HTTP, dan HTML. Sir Tim Berners-Lee mendefinisikan *Semantic Web* sebagai pengembangan dari *web* yang ada saat ini dengan informasi yang memiliki makna yang didefinisikan dengan baik (*welldefined meaning*), lebih memungkinkan komputer dan manusia untuk bekerja sama [3]. *Semantic Web* adalah cara yang efisien untuk merepresentasikan data pada *World Wide Web*, atau sebagai *database* yang terhubung secara global.



Gambar 2.1 *Semantic Web Layer* [10]

Dilihat secara konseptual, teknologi *Semantic Web* terbagi dalam beberapa *layer* arsitektur seperti yang dapat dilihat pada Gambar 2.1. Teknologi ini terdiri dari:

- Unicode dan URI
Unicode adalah standard representasi karakter komputer sedangkan URI (*Uniform Resources Identifier*) adalah standard untuk lokasi dan identitas suatu *resource*.
- XML + *Namespace* (NS) + XMLschema

XML(*Extensible Markup Language*), Namespace, dan XMLschema merupakan aturan sintaks yang berfungsi untuk menyajikan struktur data pada *web*.

- RDF + RDF(S)

RDF(*Resource Description Framework*) merupakan model dalam format *triple* yang dapat direpresentasikan dalam bentuk *graph* untuk menjelaskan *resource* dan relasinya. Sedangkan RDF Schema(RDFS) adalah definisi kosakata yang digunakan pada RDF.

- *Ontology vocabulary*

Bahasa ontologi yang direkomendasikan oleh W3C pada 10 Februari 2004 adalah *Web Ontology Language*(OWL). Bahasa OWL lebih kaya dan kompleks dari RDF untuk mendeskripsikan *resource*. OWL menggunakan format *triple*, sama seperti RDF.

- *Logic dan Proof*

Layer ini berupa *rule* dan sistem untuk melakukan reasoning pada ontologi.

- *Trust*

Layer terakhir dari *Semantic Web* yang memungkinkan pengguna *web* untuk mempercayai suatu informasi pada *web*.

2.3.1 RDF

Di dalam *Semantic Web*, kita merujuk hal-hal yang ada di dunia sebagai *resources*. *Resource* dapat merupakan apapun yang ingin seseorang bahas. Joko, kuliah, “nilai dari X”, dan “semua anjing di UI” adalah contoh hal-hal yang seseorang mungkin dibahas dan dapat menjadi *resource* dalam *Semantic Web*. *Resource* dapat diibaratkan seperti entitas atau benda pada konsep lain. *Resource* merupakan kata yang digunakan pada standard *Semantic Web*. RDF (*Resource Description Framework*) merupakan teknologi dasar dari *Semantic Web* yang berfungsi sebagai standard untuk mendeskripsikan *resource* [1].

Pada RDF, sebuah fakta atau *statement* yang ingin direpresentasikan dinyatakan dalam bentuk *triple*, dimana terdapat *subject*, *predicate*, dan *object*. Misalkan ada

Universitas Indonesia

fakta: “Stella seorang wanita”, maka “Stella” adalah *subject*, “seorang” adalah *predicate*, dan “wanita” adalah *object*. Ketiga kata ini adalah *resources*. “Stella” adalah suatu *resource*, “seorang” adalah *resource*, dan “wanita” juga merupakan *resource*. Penamaan ini bersifat global sehingga ketika misalnya ada *statement* “Stella mempunyai tas”, komputer akan tahu bahwa “Stella” pada “Stella seorang wanita” adalah sama dengan “Stella” pada “Stella mempunyai tas”. Oleh karena itu untuk membedakan setiap *resource* digunakan URI (*Unified Resource Identifier*). URI adalah suatu sistem penamaan yang *globally unique* dan merupakan generalisasi dari URL sehingga memiliki format yang mirip dengan URL pada *website*. Misalkan *statement* “Stella mempunyai tas” ingin kita representasikan dalam URI akan tampil seperti ini:

- Stella → <http://www.ta-user-model.com/kuncung/oal50#Stella>;
- mempunyai → <http:// www.ta-user-model.com/kuncung/oal50#mempunyai>;
- tas → <http:// www.ta-user-model.com/kuncung/oal50#tas>.

Jika kita ingin merepresentasikan *statement* “Stella mempunyai tas” dalam bentuk *triple* URI kita perlu menulis

```
<http://www.ta-user-model.com/kuncung/oal50#Stella
http://www.ta-user-model.com/kuncung/oal50#mempunyai
http://www.ta-user-model.com/kuncung/oal50#tas>.
```

Representasi RDF *triple* pada teks atau disebut juga *serialization syntax for RDF* yang umum dipakai ada tiga macam. Bentuk-bentuk itu adalah *N-triples*, *Notation3*, dan *XMLschema*.

Bentuk pertama yang merupakan bentuk yang paling sederhana adalah *N-Triples*. *N-triples* berkorespondensi langsung dengan *triple* RDF. *N-triples* menunjuk ke suatu *resource* dengan URI yang tidak dipersingkat sama sekali. Setiap URI ditulis diantara kurung siku (< dan >). Tiga *resource* diekspresikan dalam susunan *subject-predicate-object* ditutup dengan titik (.). Berikut ini contoh *N-triples*.

```
<http://www.ta-user-model.com/kuncung/oal50#Stella>
<http:// www.ta-user-model.com/kuncung/oal50#mempunyai>
<http:// www.ta-user-model.com/kuncung/oal50#tas>.
```

Bentuk kedua adalah *Notation3* RDF atau disingkat menjadi N3. N3 dikembangkan oleh Tim Berners-Lee[1]. N3 mengkombinasikan tampilan *N-triples* dengan *qnames*. *Qnames* adalah kombinasi *namespace* dan *identifer* suatu *resource*. N3 menggunakan *prefix* untuk merepresentasikan *local* URI suatu *resource*. Misalkan kita mendefinisikan suatu *prefix* sebagai berikut:

```
@prefix oal50:
<http://www.ta-user-model.com/kuncung/oal50#>
```

Dengan menggunakan *prefix* “oal50:” tampilan RDF dalam bentuk *N-triple* yang dijelaskan sebelumnya dapat ditampilkan dengan bentuk N3 sebagai berikut:

```
oal50:Stella;
oal50:mempunyai;
oal50:tas.
```

Bentuk selanjutnya adalah RDF/XML. Bentuk ini banyak digunakan untuk *web infrastructure*¹ oleh karena representasinya dalam bentuk XML.

2.3.2 OWL

OWL adalah *Web Ontology Language*. Bahasa-bahasa ontologi pendahulu OWL biasanya digunakan untuk mengembangkan *tools* dan ontologi untuk komunitas pengguna tertentu seperti komunitas sains. Bahasa-bahasa pendahulu tersebut biasanya tidak didefinisikan untuk *compatible* dengan arsitektur *World Wide Web* secara umum dan *Semantic Web* secara khusus [19].

OWL menggunakan URI untuk penamaan dan menggunakan RDF untuk deskripsi *framework* untuk *Web* untuk menambahkan kapabilitas ontologi sebagai berikut [19]:

- Kemampuan untuk didistribusi melalui banyak sistem
- Skalabilitas untuk keperluan *Web*
- Kompatibilitas dengan standard *Web* untuk aksesibilitas dan internasionalisasi
- *Openess* dan *Extensibility*

¹ Detail lengkap mengenai *syntax* RDF/XML dapat dilihat pada <http://www.w3.org/TR/rdf-syntax-grammar/>

OWL dibangun dari RDF dan RDF Schema dan menambahkan *vocabulary* untuk lebih mendeskripsikan *property* dan *classes*. OWL merupakan bahasa ontologi untuk *Semantic Web* yang direkomendasikan oleh *World Wide Web Consortium* (W3C).

OWL dibagi dalam 3 kelompok *sublanguage* berdasarkan ekspresi bahasanya yang digunakan sesuai kebutuhan [18]:

- *OWL Lite*
Mendukung pengguna yang memerlukan klasifikasi berdasarkan hirarki dan *constraint* sederhana. Misalkan untuk *constraint* kardinalitas, *OWL Lite* hanya memungkinkan nilai kardinalitas 0 atau 1. *OWL Lite* merupakan *sublanguage* OWL yang memiliki ekspresi bahasa yang paling sederhana, memiliki formalitas bahasa yang lebih rendah namun lebih tinggi dari *RDF Schema*. Kita dapat menyebut *OWL Lite* sebagai ekstensi dari RDFS.
- *OWL DL (Description Logic)*
Mendukung pengguna yang menginginkan ekspresi maksimal sambil tetap mempertahankan kelengkapan komputasional (segala konklusi dijamin dapat dikomputasi) dan *decidability* (semua komputasi akan selesai pada waktu yang terbatas). *OWL DL* menambahkan beberapa fitur dari yang terdapat pada *OWL Lite*, antara lain membuat operasi himpunan seperti *unionOf*, *intersectionOf*, *complementOf*. *OWL DL* dinamakan dari korespondensinya dengan *description logics*, bidang penelitian yang mempelajari *logics* yang membentuk pondasi formal dari OWL.
- *OWL Full*
Ditujukan untuk pengguna yang menginginkan ekspresi maksimum dan kebebasan *syntax* RDF tanpa jaminan komputasional. Sebagai contoh, pada *OWL Full* sebuah *class* dapat dianggap sebagai koleksi individual dan juga sebagai individual itu sendiri. *OWL Full* merupakan *sublanguage* yang paling kompleks dengan batasan *syntax* yang begitu kecil dibanding *sublanguage* lainnya. Namun hal ini menyebabkan tak adanya jaminan komputasi *complete*.

Misalnya pada OWL *Full*, suatu *reasoning* tidak dapat dijamin dapat selesai pada waktu yang *finite* (terbatas).

Setiap *sublanguage* ini merupakan ekstensi dari pendahulunya yang lebih sederhana [18]. Relasi berikut ini berlaku (inversenya tidak):

- Setiap ontologi OWL *Lite* legal adalah ontologi OWL DL legal juga.
- Setiap ontologi OWL DL legal adalah ontologi OWL *Full* legal juga.
- Setiap konklusi OWL *Lite* yang valid adalah konklusi OWL DL yang valid juga.
- Setiap konklusi OWL DL yang valid adalah konklusi OWL *Full* yang valid juga.

Pengembang ontologi yang menggunakan OWL perlu mempertimbangkan *sublanguage* yang paling cocok dengan kebutuhannya. Pilihan antara OWL *Lite* dan OWL DL bergantung pada cukup tidaknya ekspresi yang terdapat pada OWL *Lite* sehingga memerlukan ekspresi dari OWL DL. Pilihan antara OWL DL dan OWL *Full* bergantung pada lebih penting melakukan *automated reasoning* atau memberikan ekspresi yang lebih tinggi pada model seperti memberikan *meta classes (classes of classes)* [18].

2.4 Ontologi

Menurut Tom Grueber (1993), ontologi adalah spesifikasi eksplisit formal dari konseptualisasi [8]. Konseptualisasi adalah suatu gambaran abstrak dari sesuatu di dunia yang ingin kita representasikan. Ontologi menyediakan *shared vocabulary* yang dapat digunakan untuk memodelkan suatu domain, yaitu tipe suatu obyek dan/atau konsep yang ada, dan *property* dan relasinya [2]. Ontologi digunakan pada kecerdasan buatan (*artificial intelligence*), *Semantic Web*, rekayasa perangkat lunak, informatika biomedis, dan pada ilmu informatika sebagai bentuk dari representasi pengetahuan.

Ontologi menjelaskan konsep dasar pada sebuah domain dan mendefinisikan relasi diantaranya [14]. Dasar dari rancangan ontologi terdiri dari:

- *Classes* atau konsep
- *Properties* dari setiap konsep menjelaskan berbagai fitur dan atribut dari konsep
- *Restrictions* pada *properties*

Sebuah ontologi bersama dengan sekumpulan *individual instances* dari *classes* membentuk sebuah *knowledge base* [14].

Beberapa alasan mengapa kita perlu membuat suatu ontologi adalah [16]:

- Untuk mengetahui pemahaman umum mengenai struktur informasi di antara orang-orang atau *software agents*
- Untuk memungkinkan *reuse* dari *domain knowledge*
- Untuk membuat asumsi domain eksplisit
- Untuk memisahkan *domain knowledge* dari *operational knowledge*
- Untuk menganalisis *domain knowledge*

Dengan ontologi, *knowledge* dapat direpresentasikan dalam bahasa yang dapat dimengerti oleh mesin (*machine readable*). Selain itu ontologi memungkinkan terjadinya komunikasi antar *software agent* karena adanya unsur semantik atau makna yang dimengerti *agent-agent* tersebut.

Ontologi dapat dibedakan menjadi beberapa tipe sebagai berikut [4].

1. *Upper-level ontology*

Merupakan ontologi berupa suatu model umum yang merepresentasikan apa yang ada dunia. Saat ini ada SUO (*Standard Upper Ontology*) yang sedang dikembangkan oleh IEEE. Namun sangat sulit untuk mencapai kesepakatan dalam menetapkan ontologi yang demikian umum. Beberapa kandidat untuk SUO adalah SUMO (*Suggested Upper Merged Ontology*) dan *Cyc upper ontology* (*OpenCyc*). NASA juga mengembangkan *upper level ontology* yang disebut SWEET (*Semantic Web for Earth and Environmental Terminology*).

2. *Domain ontology*

Universitas Indonesia

Merupakan ontologi yang merepresentasikan suatu domain tertentu saja. Banyak penelitian yang mengembangkan ontologi di bidang kesehatan atau biologi, seperti *Gene Ontology*, *Cancer Ontology*, *Medical Ontology*.

3. *Application and Task ontology*

Merupakan ontologi yang khusus menyatakan *application* dan *task* yang independen terhadap domain. Contoh ontologi tipe ini adalah PROTON yang digunakan untuk *knowledge management system* dan selanjutnya dikembangkan pula untuk *automatic entity recognition* dan *information extraction* dari teks.

4. *Heavyweight ontology and lightweight ontology*

Berdasarkan level formalitasnya, ontologi dibedakan menjadi *heavyweight* dan *lightweight*. Hirarki topik pada Yahoo! dapat dikategorikan sebagai *lightweight* karena terdiri atas banyak konsep namun sedikit relasi dan aksioma. Sedangkan *heavyweight ontology* menyediakan definisi yang jauh lebih lengkap.

Pada ilmu komputer dan ilmu informatika, ontologi dikembangkan menggunakan bahasa ontologi. Jenis bahasa ontologi yang pernah dikembangkan antara lain: F-Logic, OKBC, LOOM, Ontolingua, SHOE (*Simple HTML Ontology Extension*), OML (*Ontology Markup Language*), RDF, RDFS, dan OWL (*Ontology Web Language*). OWL merupakan bahasa ontologi yang direkomendasikan oleh W3C sebagai standard bahasa ontologi untuk *Semantic Web*.

Pengembangan ontologi biasanya merupakan proses iteratif. Pembuatan ontologi dimulai dengan rancangan kasar. Kemudian dilakukan revisi untuk memperbaiki ontologi dan menambahkan detail. Pengembangan ontologi itu sendiri terdiri dari[12]:

1. Mendefinisikan *classes* pada ontologi
2. Menyusun *classes* dalam hirarki *subclass-superclass*
3. Mendefinisikan *properties* dan menjelaskan nilai yang diperbolehkan untuk *properties* ini
4. Mengisi nilai dari *properties* untuk *instances*.

Ada banyak kemungkinan ontologi untuk setiap *domain* tertentu. Sebuah ontologi tertentu hanyalah satu cara untuk menunjukkan struktur konsep dan relasinya. Ada beberapa prinsip sederhana yang dapat membantu kita membuat keputusan perancangan dalam banyak kasus, yaitu [13]:

- Tak ada satu cara yang benar untuk memodelkan suatu *domain*, selalu ada alternatif lain. Solusi terbaik selalu bergantung pada aplikasi yang ada dalam pikiran kita dan penambahan yang kita antisipasi.
- Pengembangan ontologi adalah proses yang iteratif.
- Konsep dalam ontologi sebaiknya mendekati *objects* dan relasi di dalam *domain of interest* kita. Ini akan seperti kata benda (*objects*) atau kata kerja (relasi) dalam kalimat yang menjelaskan *domain* kita.

Pada saat kita ingin mengembangkan ontologi, kita dapat memulai dengan menentukan untuk apakah ontologi itu nanti digunakan, dan seberapa detail atau khususkah ontologi itu akan dibuat. Diantara beberapa alternatif yang mungkin, kita akan ingin menentukan yang manakah yang paling cocok untuk proyek kita. Ingatlah ontologi adalah model dari *real domain* di dunia sehingga konsep dari ontologi sebaiknya merefleksikan kenyataan. Setelah kita mendefinisikan versi awal dari ontologi, kita dapat mengevaluasinya dan *debug* dengan menggunakannya dalam aplikasi atau metode *problem-solving* atau dengan mendiskusikannya dengan ahli di bidangnya. Kita akan hampir pasti perlu merevisi ontologi awal. Proses perancangan iteratif ini akan terus berlanjut selama siklus hidup dari ontologi tersebut [15].

2.5 Teori *Learning Style* Felder-Silverman²

Model *learning style* yang dikembangkan oleh Richard Felder dan Linda Silverman menggabungkan lima dimensi, dua dimensi tersebut merupakan

² Penjelasan mengenai Teori *Learning Style* Felder-Silverman diunduh dari <http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Papers/LS-1988.pdf>

replikasi dari model Myers-Briggs³ dan Kolb⁴. Dimensi *Perception* (*sensing/intuitive*) analog dengan *Perception* pada Myers-Briggs dan Kolb; Dimensi *Processing* (*active/reflective*) juga ditemukan pada model Kolb. Felder-Silverman menambah tiga dimensi: *Input* (*visual/verbal*), *Organization* (*inductive/deductive*), dan *Understanding* (*sequential/global*) [6]. Tabel 2.1 menunjukkan dimensi dari *learning style* Felder-Silverman.

<i>Learning Style</i>	Dimensi	Keterangan
<i>Sensing</i> <i>Intuitive</i> }	<i>Perception</i>	Berkaitan dengan bagaimana pembelajar melakukan persepsi informasi
<i>Visual</i> <i>Verbal</i> }	<i>Input</i>	Jenis <i>input</i> informasi seperti apa yang mudah diterima pembelajar
<i>Inductive</i> <i>Deductive</i> }	<i>Organization</i>	Jenis organisasi informasi yang paling cocok
<i>Active</i> <i>Reflective</i> }	<i>Processing</i>	Bagaimana pembelajar memproses informasi
<i>Sequential</i> <i>Global</i> }	<i>Understanding</i>	Bagaimana pembelajar mencapai pengertian

Tabel 2.1 Dimensi *Learning Style* Felder-Silverman

Learning style dari seorang pembelajar dapat diketahui secara garis besar dengan menjawab lima pertanyaan berikut [6]:

1. Apakah tipe informasi yang lebih mudah dirasakan pembelajar:
sensing (eksternal) → pemandangan, suara, sensasi fisik, atau
intuitive (internal) → kemungkinan, pengertian, firasat?
2. Melalui jalur sensor yang manakah informasi dari luar lebih efektif diterima pembelajar:

³ <http://chat.carleton.ca/~tblouin/MBTI/myers.html#Introduction>

⁴ <http://chat.carleton.ca/~tblouin/Kolb%27s%20Leaning%20Styles%20Model/kolb.html#Introduction-kolb>

visual → gambar, diagram, grafik, demonstrasi, atau

verbal → kata-kata, kalimat, suara, percakapan?

3. Dengan organisasi informasi seperti apakah pembelajar paling nyaman:

inductive → fakta dan observasi sudah diberikan, prinsip dasar disimpulkan, atau

deductive → prinsip diberikan, konsekuensi dan aplikasi ditarik kesimpulan?

4. Bagaimanakah pembelajar memproses informasi:

active → melalui aktifitas fisik atau diskusi, atau

reflective → melalui introspeksi, merenung, memikirkan sendiri dahulu?

5. Bagaimanakah pembelajar mencapai pengertian:

sequential → dalam langkah tahap-pertahap, atau

global → dengan lompatan besar, secara holistik, melihat gambaran besar dahulu?

Sensing dan *intuition* berkaitan dengan bagaimana kecenderungan seseorang merasakan dunia. *Sensing* termasuk observasi, melihat atau mendengar langsung; *intuition* termasuk persepsi tidak langsung dari bawah sadar seperti spekulasi, imajinasi, dan firasat. Setiap orang menggunakan kedua hal tersebut, *sensing* dan *intuition*, namun biasanya seseorang punya kecenderungan pada satu hal dibanding yang lain [6].

Berikut ini dijelaskan ciri-ciri dari pembelajar *sensing* dan *intuitive* [7].

- Pembelajar *sensing* (*sensor*) cenderung suka mempelajari fakta, data, dan eksperimen; pembelajar *intuitive* (*intuitor*) lebih suka prinsip dan teori dan seringkali lebih suka menemukan kemungkinan-kemungkinan dan hubungan.
- *Sensor* suka menyelesaikan permasalahan dengan metode yang sudah baku dan tidak suka komplikasi dan kejutan; *intuitor* suka inovasi dan tidak suka repetisi. *Sensor* akan lebih mungkin marah dibandingkan dengan *intuitor* jika mendapat ujian yang materinya tidak dijelaskan secara eksplisit di dalam kelas.
- *Sensor* cenderung sabar dengan detail dan bagus dalam mengingat fakta dan melakukan pekerjaan tangan; *Intuitor* dapat lebih baik dalam memahami

Universitas Indonesia

konsep baru dan seringkali lebih nyaman dengan abstraksi dan formula matematika dibandingkan dengan *sensor*.

- *Sensor* cenderung lebih praktikal dan hati-hati dibanding *intuitor*; *intuitor* cenderung bekerja lebih cepat dan lebih inovatif dibanding *sensor*.
- *Sensor* tidak suka kuliah yang tidak punya hubungan jelas dengan dunia nyata; *intuitor* tidak suka dengan kuliah yang melibatkan banyak hapalan dan kalkulasi rutin.

Pembelajar *visual* mengingat dengan baik apa yang mereka lihat – gambar, diagram, film, demonstrasi. Pembelajar *verbal* mendapatkan lebih dari kata-kata – baik penjelasan tertulis atau ucapan. Pembelajar *visual* mungkin dapat mudah melupakan kata-kata yang orang lain ucapkan. Pembelajar *verbal* mendapatkan banyak informasi dari diskusi dan belajar efektif dengan cara menjelaskan kepada orang lain. Kebanyakan orang pada usia kuliah dan lebih tua bertipe *visual* namun kebanyakan pengajaran kuliah adalah *verbal* – informasi yang disajikan lebih dominan *verbal* seperti ceramah atau representasi *visual* dari informasi *verbal* (kata-kata dan simbol matematika tertulis di buku, *handout*, atau papan tulis).

Induction atau induksi adalah tahapan pemikiran yang dimulai dari hal khusus (observasi, pengukuran, data) ke hal umum (teori, hukum-hukum). *Deduction* atau deduksi dimulai pada arah yang berlawanan. Pada *induction* seseorang menyimpulkan prinsip, pada *deduction* seseorang dapat menarik suatu konsekuensi/akibat.

Induction adalah *learning style* alami seorang manusia. Seorang bayi tidak datang ke dunia dengan sejumlah prinsip umum namun lebih cenderung mengobservasi dunia sekitarnya dan mengambil kesimpulan: “jika aku lempar botolku dan menangis dengan kencang pasti akan ada orang yang datang.” Kebanyakan hal yang kita pelajari sendiri berasal dari situasi nyata yang perlu diatasi, tidak dalam prinsip umum.

Pada sisi lain, *deduction* adalah gaya mengajar (*teaching style*) alami seorang manusia, paling tidak pada tingkatan kuliah. Menjelaskan prinsip umum dan kemudian maju kepada aplikasi adalah cara yang efisien dan elegan untuk mengorganisasi dan mempresentasikan materi yang sudah dimengerti. Konsekuensinya, kebanyakan kuliah biasanya dimulai dengan prinsip dan aplikasi kemudian. Hal ini menunjukkan adanya ketidakcocokan pada gaya belajar dan gaya mengajar pada umumnya. Survey informal yang dilakukan Felder-Silverman menunjukkan bahwa kebanyakan mahasiswa teknik adalah pembelajar *inductive* sedangkan kebanyakan profesor teknik mengajar dengan gaya *deductive*.

Proses mental dimana informasi dikonversi menjadi pengetahuan dapat dikelompokkan dalam dua kategori, yaitu *active experimentation* dan *reflective observation* [6]. *Active experimentation* adalah melakukan aktivitas dengan informasi seperti mendiskusikannya, menjelaskannya, atau mengujinya dengan berbagai cara. *Reflective observation* adalah menguji dan memanipulasi informasi secara introspektif.

Pembelajar *active* adalah seseorang yang merasa nyaman atau lebih baik dengan *active experiment* dibandingkan dengan *reflective observation*. Begitu pula sebaliknya untuk pembelajar *reflective*. Berikut ini dijelaskan karakteristik dari pembelajar *active* dan *reflective* [7].

- Pembelajar *active* cenderung menguasai dan mengerti informasi dengan melakukan suatu aktifitas dengannya – mendiskusikan, menjelaskan, atau mengujinya. Pembelajar *reflective* lebih suka memikirkannya sendiri dengan tenang terlebih dahulu.
- “Mari coba dan lihat apa yang terjadi” adalah kata-kata seorang pembelajar *active*. “Mari pikirkan terlebih dahulu” adalah respon dari seorang pembelajar *reflective*.
- Pembelajar *active* cenderung suka kerja kelompok dibandingkan pembelajar *reflective* yang lebih suka bekerja sendiri.

Kebanyakan pendidikan formal menggunakan bahan presentasi dalam urutan progresif, bab demi bab yang berurutan. Ketika seluruh materi sudah dibahas, pembelajar diuji penguasaannya dan kemudian maju ke tingkatan selanjutnya. Beberapa pembelajar nyaman dengan sistem ini, belajar secara *sequential*, menguasai materi lebih atau kurang sesuai dengan yang diajarkan. Yang lain, tidak dapat belajar dengan cara seperti itu. Mereka mungkin ketinggalan dari yang lain hingga berhari-hari atau berminggu-minggu, bahkan tidak dapat menyelesaikan soal yang mudah. Hingga pada suatu saat mereka “mengerti maksudnya”, mereka sudah mendapatkan gambarannya. Setelah itu mungkin mereka akan lebih mengerti materi dan menyelesaikan persoalan lebih baik dari pembelajar *sequential*. Mereka inilah yang disebut pembelajar *global*.

Berikut ini dijelaskan karakteristik dari pembelajar *sequential* dan *global*.

- Pembelajar *sequential* cenderung mendapat pengertian pada tahapan linier. Pembelajar *global* cenderung belajar pada lopatan besar, menerima materi secara acak tanpa melihat hubungannya hingga suatu saat mereka mengerti maksudnya.
- Pembelajar *sequential* cenderung mengikuti alur bertahap dalam menemukan solusi. Pembelajar *global* dapat menyelesaikan permasalahan yang kompleks dengan cepat ketika mereka sudah mendapatkan gambaran besarnya, namun mereka mungkin mendapat kesulitan menjelaskan bagaimana mereka melakukannya.

2.6 Overview TANGRAM

TANGRAM adalah sistem *personalized e-learning* yang dijadikan referensi oleh penulis untuk perancangan ontologi *user-model*. Subbab ini akan menjelaskan *overview* singkat mengenai TANGRAM.

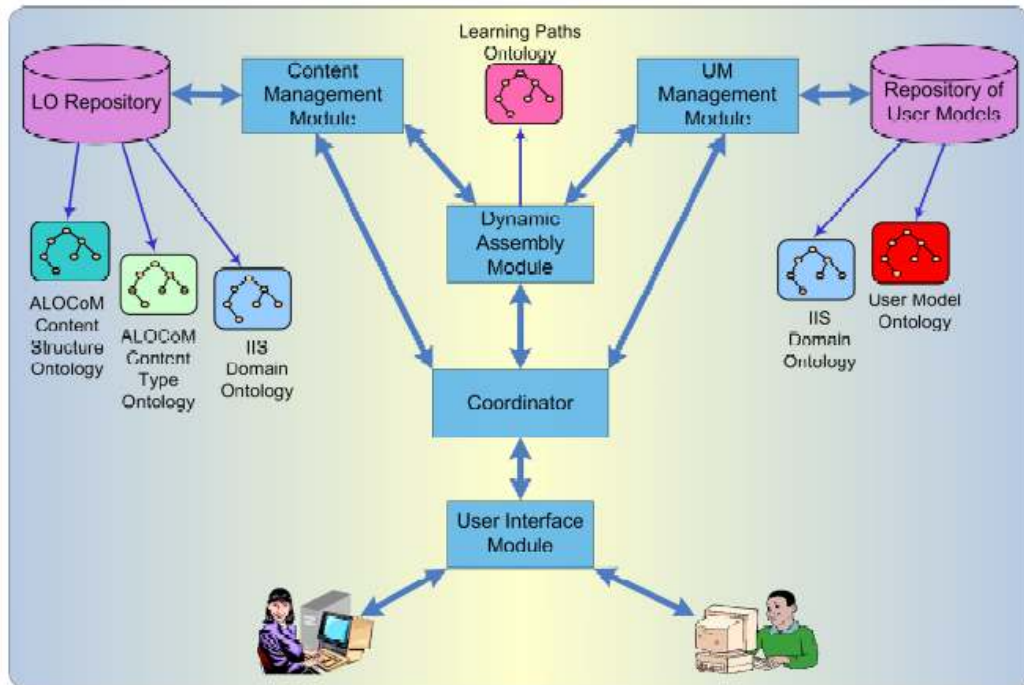
TANGRAM adalah suatu sistem *e-learning*, suatu *intelligent learning environment* untuk domain *Intelligent Information Systems*. TANGRAM menggunakan pendekatan berbasis ontologi untuk melakukan secara otomatis

Universitas Indonesia

dekomposisi *learning objects* (LO) ke dalam *reusable content units*, dan perangkaian dinamis (*dynamic reassembly*) dari unit-unit tersebut kedalam *personalized learning content*. Dengan sejumlah ontologi, TANGRAM memungkinkan dekomposisi LO kedalam *content units* yang lebih kecil, yang kemudian selanjutnya dapat dirangkai menjadi LO baru yang di-personalisasi kepada domain *knowledge*, *preference*, dan *learning styles* dari pengguna [9].

Gambar 2.2 mengilustrasikan arsitektur TANGRAM. Seperti pada gambar tersebut, terdapat empat modul utama pada TANGRAM yang dikoordinasikan dengan modul *Coordinator*, yaitu [9]:

- *Content Management Module*, bertanggung-jawab untuk menangani *uploaded* LO dan memanipulasi repositori LO TANGRAM. Fungsionalitas utama dari modul ini adalah:
 - a) Dekomposisi *uploaded* LO kedalam *content units* yang tingkatannya lebih rendah, berdasarkan ontologi *content structure*;
 - b) Pemberian anotasi kepada *content units* secara otomatis;
 - c) Penyimpanan LO kedalam format yang sesuai dengan ontologi *content structure*;
 - d) Pencarian repositori dan pengambilan *content units* dari jenis dan tingkatan yang berbeda.
- *User Model (UM) Management Module*, bertanggung jawab menangani segala jenis *request* untuk mengakses dan/atau meng-*update* repositori dari *user models*.
- *Dynamic Assembly Module*, bertanggung jawab untuk secara dinamis menghasilkan *personalized learning content* untuk pengguna spesifik (mis. pembelajar). Modul ini mengetahui bagaimana cara untuk mengkombinasikan *content units* (didapat dari *Content Management Module*) untuk membentuk *learning content* yang paling sesuai untuk pembelajar tertentu (cth. Informasi mengenai pembelajar yang dimiliki sistem didapat dari *UM Management Module*).
- *User Interface Module* menangani interaksi diantara sistem dan pengguna.



Gambar 2.2 Arsitektur TANGRAM [9]

Arsitektur ini juga terdiri dari dua repositori:

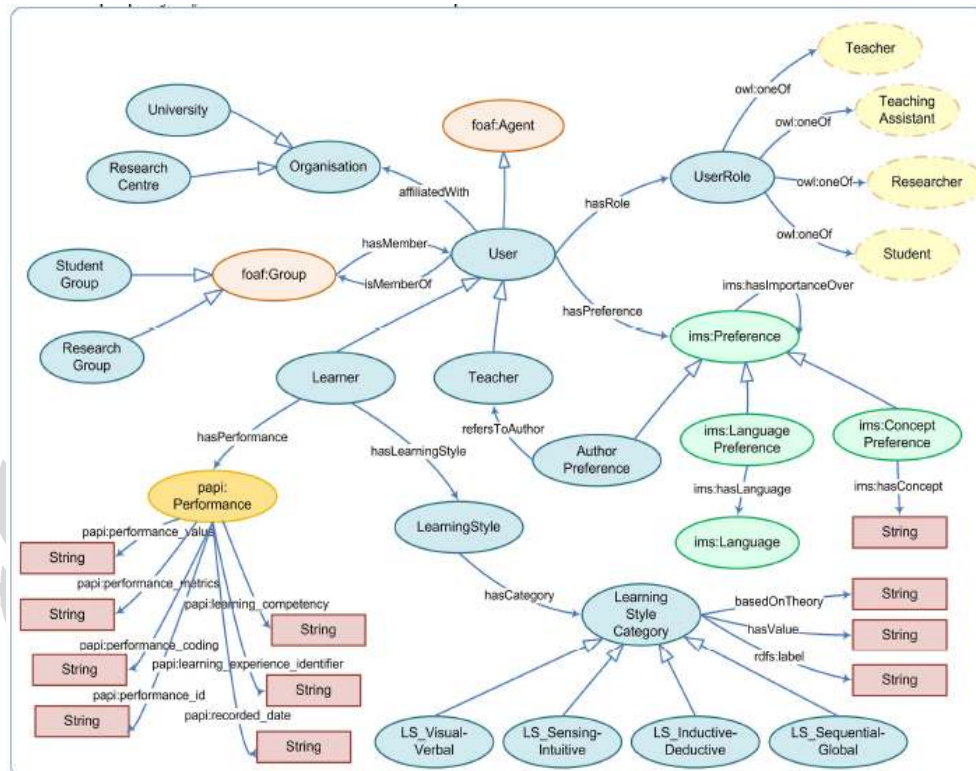
- 1) Repositori dari LO (disimpan dalam format yang sesuai dengan ontologi *content structure*) dan *metadata*-nya (berdasarkan TANGRAM LOM *profile*);
- 2) Repositori dari profil pengguna yang direpresentasikan sesuai dengan ontologi *User Model* TANGRAM.

TANGRAM dibangun dari ontologi-ontologi berikut:

1. Ontologi berbasis ALOCoM, sebagai ontologi *content structure*. ALOCoM merupakan singkatan dari *Abstract Learning Object Content Model* [1]. Ontologi ini mendefinisikan konsep dan relasi yang memungkinkan definisi formal struktur LO.
2. Ontologi *Domain*, digunakan untuk mendeskripsikan taksonomi dan skema klasifikasi topik dalam kuliah.
3. Ontologi *Learning Path*, berguna untuk mendefinisikan secara formal jalur pembelajaran melalui topik di ontologi *domain*. Pada ontologi ini terdapat

property seperti *lp:requiresKnowledgeOf*, *lp:isPrerequisiteFor*, dan *lp:hasKnowledgePonder*.

4. Ontologi *User-Model*, dikembangkan untuk merepresentasikan informasi relevan pengguna TANGRAM secara formal. Gambar 2.3 mengilustrasikan representasi grafik ontologi *user-model* TANGRAM.



Gambar 2.3 Ontologi *User-Model* TANGRAM