BAB II

LANDASAN TEORI

2.1 Android

Android adalah sistem operasi yang terdapat pada piranti bergerak yang berbasis Linux dan bersifat terbuka atau *open source* (Elian dkk., 2012). Sebelum diakuisisi oleh Google pada tahun 2005, Android dikembangkan oleh perusahaan yang bernama Android Inc.

Google dan beberapa perusahaan yang tergabung dalam *Open Handset Alliance,* seperti Intel, Texas Instrument, dan NVidia, mengembangkan sistem operasi Android dan resmi menjadi *open source* pada tahun 2007. Sistem operasi Android memiliki beberapa versi yang sudah diluncurkan ke pasar, dimulai dari versi 1.0 sampai yang terbaru, yaitu versi 5.0 (Lollipop).

2.2 Sistem Pencarian

Sistem adalah suatu kesatuan yang terdiri dari dua atau lebih komponen atau sub sistem yang saling berinteraksi untuk mencapai suatu tujuan (Jogiyanto, 2005). Dari pengertian tersebut, dapat disimpulkan bahwa sistem pencarian merupakan sekumpulan elemen yang saling berinteraksi satu sama lain yang dibangun untuk melakukan pencarian data.

Beberapa teknik pencarian yang telah dikembangkan, antara lain pencarian teks penuh, pencarian dengan metadata, dan pencarian dengan jaringan semantik (Faisal, 2009). Pencarian teks penuh adalah tipe pencarian dokumen yang dilakukan komputer dengan menelusuri keseluruhan isi sebuah dokumen, sedangkan pencarian metadata merupakan tipe pencarian dokumen yang

dilakukan komputer dengan menelusuri metadata dokumen (Sarno dan Rahutomo, 2008).

2.3 Semantic Web

Semantic web merupakan pengembangan dari WWW, dimana *content* yang ditampilkan tidak hanya dalam bahasa manusia yang umum, tetapi juga dalam format yang dapat dibaca oleh mesin (Ibrahim, 2007). Menurut Zebua dan Mustikasari (2012), *semantic web* dapat berisi informasi dalam jumlah sangat besar di *world wide web* yang terhubung secara global dengan suatu cara tertentu, dan dapat dimengerti oleh mesin, sehingga dapat diproses secara langsung oleh mesin menjadi *knowledge* untuk ditampilkan kepada pengguna.

Pembuatan *semantic web* dimungkinkan dengan adanya sekumpulan standar, seperti XML, XML *Schema, Resource Description Framework* (RDF), RDF *Scheme*, *Ontology Web Language* (OWL), dan SPARQL. Ada beberapa model yang dapat digunakan untuk pencarian dengan jaringan semantik, antara lain taksonomi (Daconta, dkk., 2003), *weighted tree similarity* (Yang, 2005), dan ontologi (Yi, dkk., 2004).

2.4 Ontologi

Ontologi adalah representasi simbolis tentang pengetahuan objek, kelas objek, properti objek, dan relasi antar objek untuk merepresentasikan suatu pengetahuan tentang domain aplikasi (Afif, 2013). Ontologi menunjukkan bagaimana konsep saling berelasi, misalnya konsep SEDAN dan TRUK memiliki relasi yang dekat, yakni keduanya merupakan kendaraan. Ontologi mempunyai

struktur bahasa formal (terdefinisi) agar dapat digunakan, antara lain sebagai berikut.

1) XML (*Extensible Markup Language*)

   XML mirip dengan HTML, tetapi *tag*-nya dapat didefinisikan oleh *user*. XML menyediakan sintaksis untuk keluaran dokumen terstruktur, tetapi dokumen belum dipaksakan untuk menggunakan batasan-batasan semantik.

2) XML Schema

   XML Schema merupakan bahasa yang membatasi struktur yang didefinisikan pada dokumen XML.

2.4.1 Resource Description Framework dan RDF Schema

Resource Description Framework (RDF) merupakan model data untuk objek dan relasi di antaranya. RDF menyediakan semantik yang sederhana untuk model data tersebut, dan data model tersebut dapat disajikan dalam sintaksis XML. RDF memiliki bagian-bagian standar yang digunakan untuk membentuknya, antara lain *resource* yang digunakan untuk menggambarkan apa saja yang dapat dimiliki oleh sebuah *Uniform Resource Identifier* (URI), *property* berisikan nama *property-property* yang ada dalam *resource*, *property value* yang berisikan nilai dari sebuah *property*. Dari bagian-bagian tersebut, terbentuk sebuah *statement* yang terdiri dari *resource*, *property*, dan *property value* atau yang disebut *triple* (Newman, 2010). *Triple* berfungsi untuk menyimpan data dan relasi antar data.

```
<?xml version="1.0">
<RDF>
    <Description about="http://www.w3schools.com/rdf">
        <author>Jan Egil Refsnes</author>
        <homepage>http://www.w3schools.com</homepage>
    <Description>
</RDF
```

Gambar 2.1 Contoh Bentuk RDF
(Atqiya dan Hariguna, 2014)

Pada Gambar 2.1 dapat dilihat contoh bentuk RDF yang memiliki beberapa komponen, antara lain sebagai berikut.

1) Http://www.w3schools.com/rdf sebagai *resource* yang berupa URI.
2) *Author*, dan *homepage* sebagai *property*.
3) Jan Egil Refsnes sebagai *property value* untuk *property author*, dan http://www.w3schools.com sebagai *property value* untuk *property homepage*.

Dari komponen-komponen yang terlihat pada Gambar 2.1, dapat dibentuk menjadi beberapa *statement*, antara lain sebagai berikut.

1) *Statement* 1: http://www.w3schools.com/rdf memiliki *author*, yaitu Jan Egil Refsnes.
2) *Statement* 2: http://www.w3schools.com/rdf memiliki *homepage*, yaitu http://www.w3schools.com.

RDF *Schema* merupakan kosakata untuk menjelaskan *properties* dan *classes* dari sumber RDF, dengan sebuah semantik untuk hierarki penyamarataan dari *properties* dan *classes*.

2.4.2 Ontology Web Language

Ontology Web Language (OWL) merupakan bahasa untuk sebuah *web* yang dikembangkan oleh W3C kelompok kerja *Web Ontology* (Gunawan dan Halim, 2014). OWL menambahkan beberapa kosakata untuk menjelaskan *properties* dan *classes*, antara lain relasi antar kelas, kardinalitas, persamaan, dan karakteristik dari properti.

Gunawan dan Halim (2014) juga menjelaskan terdapat beberapa elemen dalam OWL, antara lain sebagai berikut.

1) *Class*

OWL mendefinisikan *root* dari semua yang ada dengan owl:Thing. Semua kelas yang dibuat secara implisit merupakan subkelas dari owl:Thing. Pembuatan kelas menggunakan owl:Class dan menyatakan subkelas dengan rdfs:subClassOf. Pada Gambar 2.2, dapat dilihat contoh pendefinisian *class Teacher* yang merupakan *subclass* dari *class Person.*

```
<owl:Class rdf:ID="Teacher">
    <rdfs:subClassOf rdf:resource="#Person"/>
</owl:Class>
```

Gambar 2.2 Contoh Pendefinisian *Class* (Kurniawan, 2012)

2) *Individual*

Individual atau disebut juga *instance* adalah anggota (*member*) dari kelas. *Instance* ini dapat dipandang sebagai objek yang ada pada domain yang dibahas. Sama seperti owl:Class yang menjadi *meta level* untuk kelas, begitu pula kelas yang telah didefinisikan menjadi *meta level* untuk *instance*.

3) *Property*

Property merupakan *binary relation*. Ada dua jenis *property* pada OWL, yaitu *Object Property* (relasi antara *instance* dari dua kelas) dan *Datatype Property* (relasi antara *instance* dengan RDF *literal* dan tipe data XML Schema). Sama halnya seperti kelas yang dapat dinyatakan secara hierarki, *property* dapat dinyatakan sebagai subPropertyOf dengan rdfs:subPropertyOf. Untuk memberikan batasan pada suatu *property* dapat digunakan rdfs:domain dan rdfs:range yang disebut juga sebagai *global restriction* karena berlaku untuk umum dan tidak terbatas pada kelas tertentu. Gambar 2.3 merupakan contoh pendefinisian *Object Property* isTaughtBy.

```
<owl:ObjectProperty rdf:ID="isTaughtBy">
    <owl:domain rdf:resource="#course"/>
    <owl:range rdf:resource="#academicStaffMember"/>
</owl:ObjectProperty>
```

Gambar 2.3 Conton Pendefinisian *Object Property* (Kurniawan, 2012)

Pada Gambar 2.4, dapat dilihat contoh pendefinisian *Datatype Property* ageYear.

```
<owl:DatatypeProperty rdf:ID="ageYear">
    <rdfs:domain rdf:resource="#Person"/>
    <rdfs:range rdf:resource=
    "http://www.w3.org/2001/XLMSchema#nonNegativeInteger"/>
</owl:DatatypeProperty>
```

Gambar 2.4 Contoh Pendefinisian *Datatype Property* (Kurniawan, 2012)

2.5 RDF API for PHP

RDF API for PHP (RAP) merupakan *tools web* semantik untuk bahasa PHP yang memiliki fitur untuk memanipulasi, menyimpan, melakukan *query*, serta membangun *graph* RDF. RAP diawali sebagai proyek *open source* oleh

Freie Universitas Berlin pada 2002 dan telah mengalami perkembangan di komunitas web semantik. Inti dari RAP meliputi dua implementasi atas penyimpanan *statement*, yaitu apakah menyimpan *graph* RDF dalam *memory* sistem maupun dalam *database relational* (Awaludin, 2009).

2.6 SPARQL

Model data RDF berupa suatu *statement* dalam bentuk *triple* yang terdiri dari subjek, predikat, dan objek. Untuk mendapatkan informasi dari suatu *graph* RDF dibutuhkan suatu *query*. SPARQL merupakan *query* untuk RDF/OWL yang digunakan untuk mengambil data yang ditulis dengan menggunakan RDF/OWL. Bahasa *query* SPARQL hampir sama dengan *query* SQL biasa. Menurut Mukhmad Nurkamid (2009), klausa yang digunakan dalam *query* SPARQL, antara lain sebagai berikut.

1) PREFIX

   Statement PREFIX merupakan sebuah metode yang digunakan sebagai penunjuk yang membawa informasi dalam suatu halaman *web*. Pada dasarnya, PREFIX digunakan untuk menyingkat sebuah *resource*, dalam hal ini dapat diwakili oleh *Uniform Resource Identifier* (URI).

2) SELECT

   Statement SELECT didefinisikan sebagai sebuah daftar variabel-variabel yang akan dikembalikan sebagai hasil dari eksekusi *query*. Setiap variabel diawali dengan notasi tanda tanya (?).

3) WHERE

Statement WHERE didefinisikan sebagai sederetan *triple pattern* yang harus dimiliki oleh setiap hasil *query* yang valid. Seluruh pola yang merepresentasikan suatu kalimat RDF harus sesuai dengan RDF *triple*, yaitu terdiri dari subjek, predikat dan objek. Ketiga RDF *triple* tersebut dapat direpresentasikan oleh URI atau sebuah *variable* dan *literal*.

4) OPTIONAL

Statement OPTIONAL digunakan untuk mengatasi ketidakcocokan struktur pola *query* dengan pola yang ada pada *graph* RDF.

```
PREFIX abc: <http://mynamespace.com/exampleOntologie#>
SELECT ?capital ?province
WHERE {
    ?x abc:cityname ?capital.
    ?y abc:provincename ?province.
    ?x abc:isCapitalOf ?y.
    ?y abc:isInCountry abc:indonesia.
}
```

Gambar 2.5 Contoh SPARQL (Ibrahim, 2007)

Pada Gambar 2.5 dapat dilihat contoh SPARQL yang mengambil nilai dari variabel ?capital dan ?province. Variabel ?capital merupakan nilai dari *property* cityname, dan variabel ?province merupakan nilai dari *property* provincename.

2.7 Skala Likert

Skala Likert merupakan suatu skala psikometrik yang banyak digunakan pada kuesioner maupun survei dengan menggunakan beberapa tingkatan skala, misalnya lima tingkatan dengan skala: sangat setuju, setuju, netral, tidak setuju dan sangat tidak setuju, dengan masing-masing bobot untuk setiap tingkatnya

(Tantri, 2015). Bobot dari setiap tingkatan berguna untuk menganalisis dan melakukan penarikan kesimpulan.

Menurut Rizkiyani (2013), terdapat dua bentuk pernyataan dalam skala Likert, yaitu pernyataan positif yang berfungsi untuk mengukur sikap positif dan pernyataan negatif yang berfungsi untuk mengukur sikap negatif suatu objek. Skor pernyataan positif dimulai dari 1 untuk sangat tidak setuju, 2 untuk tidak setuju, 3 untuk netral, 4 untuk setuju, dan 5 untuk sangat setuju. Skor pernyataan negatif adalah kebalikan dari skor pernyataan positif, yaitu 1 untuk sangat setuju, 2 untuk setuju, 3 untuk netral, 4 untuk tidak setuju, dan 5 untuk sangat tidak setuju.