ONTONOLOGICAL REPRESENTATION OF LEARNING OBJECTS IN SOFTWARE ENGINEERING AND RELATEDNESS BETWEEN THEM

Dr. Latha Parthiban and Shija K.R.
Department of Computer Science
Pondicherry Central University
Email: lathaparthiban@yahoo.com

Abstract: Concepts in any particular domain will be associated each other. This paper presents representation of concepts (Learning Objects in terms of e-learning) in Software engineering domain using ontology mechanism and introduces measure of annotation of a Learning Objects/concept must share some similar keywords with its related concepts. It calculates the semantic relatedness of Learning Objects (LO), based on their annotation similarity. This measure is unique in such a way that similarity based on similar keywords shared over annotations of concepts in Software Engineering (SE) ontology.

Key words: e-learning, learning objects, ontology, relatedness

I. INTRODUCTION

In this electronic era, prefix ‘e’ has being added with all popular areas like e-learning, e-shopping and e-banking etc. Learning or teaching something electronically is so called e-learning. E-learning is the most convenient way in learning and teaching without having to go to institute. This is expedient to those who have less time to go one place to another. The importance of e-learning is that it is student centered which views on students particular needs. Learning objects are collection of items. It can be contents, assessments and practice items (1).

This paper presents representation of concepts (LO) in software engineering domain using ontology mechanism and introduces measure of semantic relatedness of LO, based on their annotation similarity. Software engineering ontology consists concepts or LO and properties. It measures similarity by counting similar keywords that shared among annotations. This measurement is based upon content of LO. LOs can be represented in many ways. One method is representing LO using ontology mechanism. A brief description or comment can be given for each LO as its annotation while creating ontology. Annotation can be given in such a manner that related concepts must share some keywords as common.

A. Ontology

Ontology defines a common vocabulary for researchers who need to share information in a domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them (2). Ontologies are used to capture knowledge about some domain of interest.

It describes the concepts in the domain and also the relationships that hold between those concepts. Classes are the focus of most ontology. Classes describe concepts in the domain. Properties of each concept describing various features and attributes of the concept. Ontology is consisting of a vocabulary used to describe some domain. It specifies intended meaning of the vocabulary. Ideally, ontology should capture a shared understanding of a domain of interest (3) (10) (9).Fig 1 shows architecture of ontology generation.

Using OWL (Web Ontology Language) software engineering ontology has been developed. OWL ontology consists of three components that include Individuals, Properties, and Classes. Individuals are objects in the domain that we are interested in. Properties are binary relations on individuals which link two individuals together. Classes are sets that contain individuals.
2. RELATED WORKS

Software Engineering Ontology for Software Engineering Knowledge Management in Multi-site Software Development Environment (4) has provided a software engineering ontology which defines common sharable software engineering knowledge including particular project information. This software engineering ontology provides software engineering concepts for representing and communicating over software engineering knowledge and project information. These concepts provide common understanding of software engineering project information to all the distributed members of a development team in a multi-site development environment. The software engineering ontology contains abstractions of the software engineering domain concepts.

The Extended Gloss Overlapping as Measure of Semantic Relatedness (5) has provided a measure of semantic relatedness between concepts based on number of overlaps in their gloss. Glosses are brief description of each concept. According to a given concepts hierarchy, it extends the glosses of other concept to which they are related. Extended gloss overlaps, is a measure of semantic relatedness and is based on information from a machine readable dictionary such as SENSEVAL-2 lexical sample data and lexical data base WordNet. This measure takes advantage of hierarchies of concepts as found in resources such as the lexical database WordNet. It correlates human judgment reasonably.

WordNet::Similarity - Measuring the Relatedness of Concepts (6) provides three measures of relatedness and six measures of similarity, all of which are based on the lexical database WordNet. WordNet::Similarity is a freely available software package. It makes possibility to measure the semantic similarity or relatedness between a pair of concepts (or word senses). These measures are implemented as Perl modules which take as input two concepts, and return a numeric value that represents the degree to which they are similar or related.

Automatically creating datasets for measures of semantic relatedness (7) proposes a corpus-based system for automatically creating test datasets. Semantic relatedness is a special form of linguistic distance between words. Evaluating semantic relatedness measures is usually performed by comparison with human judgments. Previous test datasets had been created analytically and were limited in size. Resulting datasets cover all degrees of relatedness which experiments with human subjects. As a result of the corpus-based approach, test datasets cover all types of lexical-semantic relations and contain domain-specific words naturally occurring in texts.

KWSim: Concept Similarity Measure (8) is introduced the kwSim measure by examining the IS-A relation type and mapping concept to different medical thesauri. This measure is fully automated technique. The comparison of manually annotated medical images can be done in a lexical way by using the comparison of keywords or using the existing medical thesauri to calculate semantic similarity. This work provides a user assistant for the description and comparison of manually annotated medical images.

3. PROPOSED APPROACH

Proposed approach provides an efficient way to the learners by suggesting related LO of particular one. So that he/she can know what are the prerequisite and subsequent LOs with respect to the concept that he wants to learn. The measure is based on content. Relatedness is measured by comparing similar keywords in the comment that added with each LO while creating ontology. Software engineering ontology is the back bone of this method.
Steps involved in ontology development as provided in figure 2 is explained below

**Step 1: Determine the domain and scope of the ontology.**

Ontology development starts by defining its domain and scope. In this context software engineering is chosen.

**Step 2: Enumerate important terms in the ontology.**

Important terms can be identified after determining the domain. For instance, software engineering, life cycle, software engineering process etc.

**Step 3: Define the classes and the class hierarchy.**

There are several approaches in developing a class hierarchy. Among them, we have chosen top-down approach for defining class. It starts with most general concepts in the domain and subsequent specialization of concepts.

**Step 4: Define the properties of classes.**

For each property that has been identified, we must determine which class it describes. For example, is, hasType, hasLayer etc are some properties that has been identified.

**Step 5: Define the restriction of properties.**

Properties can have different restriction describing the value type, allowed values, the number of the values (cardinality) etc

**Step 6: Create instances**

The last step is creating individual instances of classes in the hierarchy. Defining an individual instance of a class requires (1) choosing a class, (2) creating an individual instance of that class, and (3) filling in the property values. Comment can be given while defining class and instances. Some LOs and respective comments are given in table above.

Table 1 also shows comparison of shared keyword of LO 'Testing' with other.

<table>
<thead>
<tr>
<th>Learning Object</th>
<th>Comment(12)</th>
<th>Shared keyword</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software</td>
<td>A systematic, disciplined and quantifiable process for development, operation</td>
<td>Yes</td>
<td>Related</td>
</tr>
<tr>
<td></td>
<td>and maintenance software. It includes several phases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Life cycle</td>
<td>Methodology for developing a software system and it includes well defined</td>
<td>Yes</td>
<td>Related</td>
</tr>
<tr>
<td></td>
<td>phases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Testing</td>
<td>One phase in SDLC. Testing is a verification and validation activity doing</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>to uncover errors.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>White box testing</td>
<td>White-box testing also called structural testing is a testing</td>
<td>Yes</td>
<td>Related</td>
</tr>
<tr>
<td></td>
<td>that takes into account the internal mechanism of a system.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>The product that software professionals build and then support over long</td>
<td>No</td>
<td>Not Related</td>
</tr>
<tr>
<td></td>
<td>term</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**A. Algorithm**

1. Get learning object as input
2. Get the corresponding annotation from ontology
3. Remove stop words from it and get key words alone
4. Do compare the keywords and comment each learning object
5. Return learning object if shared keywords is there

When learner enters a concept that he wants to learn, system will read and get its annotation from ontology. Applying stop word removal algorithm it will remove all stop words that contained in that annotation and store keywords alone. This will be compared with annotations of rest of the LOs. If shared keywords are found while comparing, the particular learning object will return. The LOs returned like will be the related concepts.

For example, the LO “Testing” from Table 1 shares keywords phase and testing with other LOs except software. It means that testing and other are related. But testing and software are not related so
4. ANALYSIS OF RESULT

When the comparison is done, the keywords from the comment of the particular LO given by the user are compared with rest of the LOs. This is done in order to find whether there are shared keywords in comment of given LO and other. Comments are given in such a way that related concepts must share some keywords. For highly related concepts, number of shared keywords will be more. Related concepts will share more than one keywords. For unrelated concepts number of shared keywords is zero.

Figure 3 shows the result analysis and it can be concluded that If the number of shared keywords is more, relatedness between them will be high.

5. CONCLUSION

This paper presents representation of LO in Software engineering domain using ontology mechanism and introduces measure of semantic relatedness of LOs, based on their annotation similarity. The relatedness is based on similar keywords that shared over annotation of LOs. Annotation can be given in such a way that annotation of a Learning Objects/concept must share some similar keywords with its related concepts. It calculates the similarity based on similar keywords shared over annotations of concepts in Software Engineering ontology. This provides a flexible way to find related LOs. This method can be extended for different domains providing an easy way for all e-learning systems.

REFERENCES