The Journal of Systems and Software 83 (2010) 803–814

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Measuring design complexity of semantic web ontologies

Hongyu Zhang [a,*], Yuan-Fang Li [b], Hee Beng Kuan Tan [c]

[a] School of Software, Tsinghua University, Beijing 100084, China
[b] School of ITEE, The University of Queensland, Brisbane, Australia
[c] School of EEE, Nanyang Technological University, Singapore 639798, Singapore

ARTICLE INFO

Article history:
Received 24 January 2009
Received in revised form 19 November 2009
Accepted 19 November 2009
Available online 1 December 2009

Index Terms:
Design complexity
Ontology
OWL
Ontology metrics
Software metrics

ABSTRACT

Ontology languages such as OWL are being widely used as the Semantic Web movement gains momentum. With the proliferation of the Semantic Web, more and more large-scale ontologies are being developed in real-world applications to represent and integrate knowledge and data. There is an increasing need for measuring the complexity of these ontologies in order for people to better understand, maintain, reuse and integrate them. In this paper, inspired by the concept of software metrics, we propose a suite of ontology metrics, at both the ontology-level and class-level, to measure the design complexity of ontologies. The proposed metrics are analytically evaluated against Weyuker's criteria. We have also performed empirical analysis on public domain ontologies to show the characteristics and usefulness of the metrics. We point out possible applications of the proposed metrics to ontology quality control. We believe that the proposed metric suite is useful for managing ontology development projects.

1. Introduction

The Semantic Web (Berners-Lee et al., 2001) is an envisioned extension of the current World Wide Web in which data is given well defined meaning so that software agents can autonomously process the data. It is also widely believed that Semantic Web ontologies provide a solution to the knowledge management and integration challenges (Searls, 2005; Auer et al., 2008; Smith et al., 2007; Ruttenberg et al., 2009). Ontology languages such as RDF Schema (Brickley and Guha, 2004) and OWL (Horrocks et al., 2003) provide essential vocabularies to describe domain knowledge, the underlying common model for data aggregation and integration.

A great deal of efforts are being invested in applying Semantic Web ontologies to create mutually agreeable and consistent vocabularies to describe domain knowledge from disparate sources. For example, the NCI Thesaurus Ontology[1] developed and actively curated by the National Cancer Institute is such an OWL ontology. It defines 60,000+ named classes, a roughly equal number of anonymous classes and 100,000+ connections (properties) from and to these classes. This ontology covers information about nearly 10,000 cancers and 8000 therapies. More recently, as the result of the Linked Data project[2], a large number of inter-connected RDF datasets, such as DBPedia[3], DBLP[4], FOAF[5], US census data, etc., are being generated and integrated. With more information being converted to RDF/OWL and integrated, we believe that properly designed OWL ontologies is essential to the effective management, reuse and integration of these data.

As ontologies grow in size and number, it is important to be able to measure their complexity quantitatively. It is well known that "You cannot control what you cannot measure" (DeMarco, 1986). Quantitative measurement of complexity can help ontology developers and maintainers better understand the current status of the ontology, therefore allowing them to better evaluate its design and control its development process. Research on human cognition shows that humans have limited capabilities in information processing (e.g., Miller, 1956; Simon, 1974). Experiences from the software engineering field also suggest that there are correlations between software complexity and quality (such as reusability and maintainability) (Li and Cheung, 1987; Wilde et al., 1993; Koru and Tian, 2003; Zhang et al., 2007). We believe such correlation exists between ontology complexity and quality too – in general, more complex ontologies tend to be more difficult for a human to comprehend, therefore more difficult to be maintained and reused.

In software engineering domain, the term software complexity is often defined as "the difficulty of performing tasks such as coding, debugging, testing and modifying the software" (Zuse, 1991).

* Corresponding author. Tel.: +86 10 62773275.
E-mail addresses: hongyu@tsinghua.edu.cn (H. Zhang), liyf@itee.uq.edu.au (Y.-F. Li), ibktan@ntu.edu.sg (H.B.K. Tan).

[1] http://www.cancer.gov/cancertopics/terminologyresources.
[2] http://linkeddata.org.
[3] http://dbpedia.org/.
[4] http://www.informatik.uni-trier.de/ley/db/.
[5] http://www.foaf-project.org.

0164-1212/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2009.11.735

Software metrics (Fenton and Pfleeger, 1998) are designed to quantify software products and processes. In the same spirit, we define ontology design complexity as the difficulty of performing tasks such as developing, reusing and modifying the ontology. This paper addresses the increasing needs for measuring the complexity of ontology designs by utilizing the concepts of software metrics.

We consider ontology complexity as a profile multidimensional construct (Law et al., 1998), which is formed as various combinations of dimensional characteristics and cannot be measured directly using a single metric. Therefore, we propose a suite of metrics (at both the ontology-level and class-level) to measure different aspects of the design complexity of ontologies. Together, these metrics help us gain a more complete understanding of ontology complexity.

Weyuker's criteria (Weyuker, 1988) is a set of properties for evaluating software metrics. We analyze the applicability of Weyuker's criteria in the context of ontology and analytically evaluate our proposed metrics against them.

We have also collected real-world ontologies from public domains to show the characteristics of the proposed metrics and to evaluate the usefulness of the metrics. By doing so, we seek to demonstrate the level of rigor required in the development of useful ontology metrics. An automated tool based on the Protégé-OWL API[6] has been developed to facilitate metric computation. We also point out how the proposed metrics can be applied to ontology quality control. Our proposed metrics are theoretically and empirically sound, are capable of revealing the internal structure of ontologies, and are useful for ontology engineering practices.

The rest of the paper is organized as follows: in Section 2 we introduce the background on complexity measures and related work. Section 3 introduces the problem of evaluating ontology complexity and formally defines the graphic-centric representation of OWL ontologies, for the discussion of complexity metrics. In Section 4, we describe our proposed metric suite. Sections 5 and 6 give analytical evaluation and empirical evaluation of the metrics, respectively. Section 7 discusses how the proposed metrics can be applied to ontology development practices. Finally, in Section 8 we conclude the paper and suggest future work directions.

2. Background and related work

Complexity has been a subject of considerable research. In cognitive psychology, a convenient metaphor treats human cognition as a computer-like information processor (Lindsay and Norman, 1977). Both of them involve similar concepts such as input/output, memory, processing power, and critical resources.[7] Like an information processor, it is believed that humans' problem solving and other complex cognitive processes have limited capabilities, which restrict the understanding and development of complex structures. For example, the seminal works on 7 ± 2 limits (Miller, 1956) and the size of a memory chunk (Simon, 1974) reveal that a human can only cope with limited information at a time via short-term memory, independent of information content. It is also discovered that the difficulty of a task can be measured by the number of cognitive resources required to perform the task (Sheridan, 1980).

In software engineering domain, researchers and engineers attempt to quantitatively understand the complexity of the software undertaken and to find the relationships between the complexity and the difficulty of development/maintenance task. Many software complexity metrics have been proposed over the years. Examples include cyclomatic complexity (McCabe, 1976), coupling metrics (Fenton and Melton, 1990) and the CK object-oriented design metrics (Chidamber and Kemerer, 1994). Many researchers have shown that complexity measures can be early indicators of software quality. For example, empirical evidence supporting the role of object-oriented metrics, especially the CK metrics, in determining software defects was provided in (Basili et al., 1996; Subramanyam and Krishnan, 2003).

An ontology is a specification of a conceptualization (Gruber, 1993), which can capture reusable knowledge in a domain. In software engineering area, object-oriented design also involves the conceptualization of domain knowledge, producing deliverables such as class diagrams. It has been shown that object-oriented modeling languages can be grounded on ontological theory (Opdahl and Henderson-Sellers, 2001). There is much similarity between object-oriented design and ontology development, suggesting that we may borrow the principles and methods from software metrics research to design ontology metrics. However, we cannot apply the metrics originally designed for software complexity to ontology without adaptation. Many software complexity metrics are based on program control flow or the number of methods. For example, three of the six CK metrics involves information about methods, which are not applicable to ontology. Therefore it is necessary to design a new suite of metrics for measuring complexity of ontologies.

In recent years, various metrics for measuring ontologies were proposed. For example, Yao et al. suggested three metrics (Yao et al., 2005) (namely the number of root classes, the number of leaf class, and the average depth of inheritance tree) to measure the cohesiveness of an ontology. Kang et al. (2004) proposed an entropy-based metric for measuring the structural complexity of an ontology represented as UML diagram. These efforts only focus on one or two aspects of structural complexity and lack sound theoretical or empirical validations.

Some researchers also proposed integrated frameworks for ontology measurement. For example, Gangemi et al. (2006) proposed a meta-ontology O^2 that characterizes ontologies as semiotic objects. Based on this ontology they identified three types of measures for ontology evaluation: structural measures, functional measures and usability-profiling measures. A large number of potential metrics were proposed as well. Some of these metrics cannot be automatically calculated, limiting their utility. It also did not provide an empirical analysis for the metrics.

In Wang et al. (2006), a large number ($\sim$1300) of OWL ontologies were collected and statistically analyzed. The main focus of that work is ontology expressivity (e.g., to which OWL species – Lite, DL or Full – an ontology belongs) and consistency characteristics. Besides expressivity, an analysis on the shape of ontology class hierarchy (a graph of subsumptions) was also presented. The authors compared the morphological changes between the classified and inferred (as in OWL reasoning) versions of a class hierarchy and suggested that it may be useful to determine which classes are over- or under-modeled. Their work on graph morphology of class hierarchies is similar to the intention of our tree impurity *TIP* metric that will be presented in Section 4.

Vrandečić and Sure (2007) proposed guidelines for creating ontology metrics based on the notions of "normalization". Their work laid out a set of principles for designing stable, semantic-aware metrics. They proposed five normalization steps, namely: (i) name anonymous classes, (ii) name anonymous individuals, (iii) materialize the subsumption hierarchy and unify names, (iv) propagate instances to deepest possible class or property within the hierarchy, and (v) normalize properties. The normalization process attempts to transform the ontology into a semantically-equivalent form to facilitate the creation of "semantic-aware" metrics. As we stated previously, the objective of our research is to measure

[6] http://protege.stanford.edu/overview/protege-owl.html.

[7] We should note that although we use this metaphor, we are not saying that human's brain functions like a Von Neumann computer.