Complexity metrics for ontology based information

Anthony Mark Orme*

Athens State University, Athens, Alabama 35611, USA
Fax: 256.233.8261
E-mail: Tony.Orme@athens.edu E-mail: aorme@cs.uah.edu
*Corresponding author

Haining Yao and Letha H. Etzkorn

University of Alabama in Huntsville,
Huntsville, Alabama 35899, USA
E-mail: hy77t@nih.gov E-mail: letzkorn@cs.uah.edu

Abstract: An ontology defines technical terms and shows interrelationships between terms for particular application areas. XML-based standards such as OWL and DAML provide mechanisms to produce XML based ontologies. Ontologies are used in service matching and dynamic web service composition, and are heavily used in bioinformatics and genomics to characterise the structure of living things. Our research focuses on complexity metrics for ontologies. These complexity metrics are compiled from semantic relationships in an ontology. These metrics will help select the best ontologies in several application areas, including bioinformatics and genomics.

Keywords: ontologies; ontology based systems; semantic web; metrics; complexity metrics.

Reference to this paper should be made as follows: Orme, A.M., Yao, H. and Etzkorn, L.H. (2009) 'Complexity metrics for ontology based information', *Int. J. Technology Management*, Vol. 47, Nos. 1/2/3, pp.161–173.

Biographical notes: Anthony M. Orme is an Assistant Professor at Athens State University in Athens, AL. His primary research interests are in software engineering (software metrics), distributed systems and knowledge engineering. He has PhD in Computer Science from the University of Alabama in Huntsville. He is a member of the ACM.

Haining (Irene) Yao is employed at the National Library of Medicine (NIM) at the National Institutes of Health (NIH), in Washington DC. Her research interests are in ontology development and metrics, knowledge representation, and software engineering (software reuse). She received her PhD Degree in Computer Science from the University of Alabama in Huntsville.

Letha H. Etzkorn is an Associate Professor at the University of Alabama in Huntsville. Her primary research interests are in software engineering (including program understanding, object-oriented software metrics and artificial intelligence applications to software engineering) and mobile agents. She received her PhD Degree in Computer Science from the University of Alabama in Huntsville.

1 Introduction

The astounding growth of the internet and the changes it has made to the global economy bring many challenges and opportunities to the development of new computing platforms and tools to support these platforms. These challenges include the need to provide solutions that are intelligent, adaptive and reactive to the changes in today's ever changing internet-based world. As the internet grew to become commercially available to the world, companies and the software industry quickly realised new strategies were needed to embrace the dawn of a new era in computing history.

Also, recent growth in the fields of bioinformatics and genomics has led to the need for new software solutions. For example, it is important in these areas to characterise the molecular structure of living things, particularly in ways that are easily manipulated via software.

In both the internet and bioinformatics, the use of ontologies to define technical terms in a particular application area and show the relationships between these terms has become common. On the internet, ontologies can be used in a web service description. Then when one web software application is trying to request a service from another web software application, the two web applications are speaking the same language. That is, an ontology is used to define terms so that when a web software application asks for a service, and the other application provides the service, both applications understand exactly what service is being requested, and what service is being provided.

In bioinformatics, as the molecular structure of living things is categorised, the abundance of information has led to the definition of a large number of new ontologies. Typically, although there may be overlap, each ontology serves a separate purpose, and it categorises a different set of living things. Sometimes, however, different ontologies represent similar sets of living things. In this situation, which ontology should be selected for use?

As technology has brought on many new challenges, it has also needed several new enabling technologies to answer many of these challenges. One such technology, Extensible Markup Language (XML) is a simple text-based format designed to allow the exchange of a wide variety of data on the internet (WC3, 2006). XML is now used in a variety of new architectures, and application specific specialised languages such as Web Services Description Language (WSDL), Ontology Web Language (OWL), Business Process Execution Language for Web Services (BPEL4WS) and many others. XML-based languages such as OWL are also used in bioinformatics and genomics.

This paper presents and analyses a set of complexity metrics, Number of Classes (NoC), Number of Fanouts (NoF) and Average Depth of Inheritance Tree of all Leaf Nodes (ADIT-LN). These metrics are collected using a standard XML based Document Object Model (DOM) parser. These metrics are validated by a proposed theoretical framework and using empirical validation by using statistical correlations. We analyse complexity metrics in this paper; previously we have analysed coupling metrics (Orme et al., 2006) and cohesion metrics (Yao et al., 2005).

The remainder of this paper is organised as follows: Section 2 provides the necessary background on OWL needed to understand web-based ontologies. Section 2 also presents a metrics validation framework developed by Kitchenham et al. (1995). We validate our set of metrics theoretically using this validation framework. Section 3 defines our ontology metrics both conceptually and in a formal mathematical notation. In Section 3, we also include example calculations of each metric. The theoretical and empirical validation of the ontology metrics is completed in Section 4 along with a complete summary of our work and results. Section 5 provides a brief discussion of our ontology metrics could be applied in bioinformatics and genomics. Section 6 provides conclusions and thoughts for future research in the area of ontology based computing.

2 Background

2.1 Ontology Web Language (OWL)

One of the most recognised definitions of an ontology as given by Gruber states

> "When the knowledge of a domain is represented in a declarative formalism, the set of objects that can be represented is called the universe of discourse. Those set of objects, and the describable relationship among them, are reflected in the representational vocabulary with which a knowledge-base program represents knowledge. Thus, in the context of AI, we can describe the ontology of a program by defining a set of representational terms. In such ontologies, definitions associate the names of entities in the universe of discourse (e.g., classes, relations, functions, or other objects) with human-readable text describing what the names mean, and formal axioms that constrains the interpretation and well-formed use of these terms. Formally, an ontology is the statement of a logical theory." (Corazzon, 2006)

The Web Ontology Language (OWL) is a semantic markup language for publishing and sharing ontologies on the World Wide Web. OWL is developed as an extension of Resource Description Framework (the RDF) and is based upon DAML + OIL OWL. OWL is intended to provide a language that can be used to describe the classes and relations between them that are inherent in web-based documents and applications. OWL aids in formalising a domain by defining classes and asserting properties of those classes. OWL allows reasoning about the classes to a degree that is supported by the formal semantics of the language.

An Ontology is defined using a mixture of both OWL and RDF tags, prefixed by owl: and rdf: respectively. In defining a class definition in an ontology, the tag OWL class tag is used to denote the beginning of a class definition along with the rdf:ID tag to denote the name of the class

```
<owl:Class rdf:ID = "CLASSNAME">
<rdfs:subClassOf rdf:resource = "SOMESUPERCLASS"/>
</owl:Class>
```

The end of a class definition is denoted by the </owl:Class> tag. Subclass relationships are denoted by the rdfs:subClassOf tag and a rdf:resource tag which point to the definition of the superclass. The superclass definition may be defined within the same ontology or defined in another ontology.

The following OWL example is drawn from three ontologies: agent.ont, foaf-basic.ont and location.ont, defined by Creative Commons (Chen, 2006). The agent ontology defines a class called Agent and a class called Person, wherein Person is defined to be a subclass of Agent by the statement <rdfs:subClassOf rdf:resource = "#Agent"/> in the definition of Person.

```
<owl:Class rdf:ID = “Agent”>
        <rdfs:label>Agent</rdfs:label>
        <rdfs:subClassOf rdf:resource = “&foaf; Agent”/>
        <rdfs:subClassOf
        rdf:resource = “&loc; ThingHasLocationContext”/>
</owl:Class>
<owl:Class rdf:ID = “Person”>
        <rdfs:label>Person</rdfs:label>
        <rdfs:subClassOf rdf:resource = “#Agent”/>
</owl:Class>
```

However, Agent is also defined as a subclass by the following statement in its definition:

```
<rdfs:subClassOf rdf:resource = “&foaf; Agent”/>
<rdfs:subClassOf rdf:resource = “&loc; ThingHasLocationContext”/>
```

stating the Agent in agent.ont is subclass from two classes found in ontologies defined in the other OWL documents referred to by ‘&foaf; Agent’ and “&loc; ThingHas LocationContext”.

2.2 Metrics validation framework

Kitchenham et al. (1995) proposed a framework for evaluating software metrics. In this framework, they described the structure of any measure as containing the entities being analysed, such as classes or modules; the attribute being measured, such as size; the unit used, such as lines of code; and the data scale: nominal, ordinal, interval, or ratio. Units are valid only for interval or ratio data, but they can be adapted for use with ordinal data. In order for a value to have any meaning, the entity, the attribute being measured, and the units must be specified. The measure must be defined over a specified set of permissible values (discrete or continuous).

In order to be valid, a measure must have (Kitchenham et al., 1995):

- *attribute validity*: the entity being analysed has the attribute
- *unit validity*: the unit is appropriate for the attribute
- *instrumental validity*: the underlying model is valid and the instrument was calibrated
- *protocol validity*: the protocol used for the measurement was valid and prevented errors such as double counting.

Furthermore, in order to be theoretically valid, a direct measure must have the following properties (Stein et al., 2004a, 2004b):

- the attribute has different values for different entities
- the measure works in a way that makes sense with respect to the attribute and its values for different entities

- any of the attribute's units can be used if the attribute is part of a valid measure
- the attribute can have the same value for different entities (Kitchenham et al., 1995).

For an indirect measure, the following properties apply:

- a model of relationships among entities' attributes is the basis for the measure
- no improper use of dimensionality occurs in the measure
- no unexpected discontinuities occur in the measure
- the units used are appropriate for the scale of data available (Kitchenham et al., 1995).

In addition to theoretical validation, Kitchenham et al. (1995) recommend empirical validation using statistical analysis of metric values compared to evaluators' assessment of software.

3 Complexity metric definitions

In this section we provide both the formal mathematical and a conceptual definitions for the NoC, NoF and ADIT metrics: Section 3.1 defines the common mathematical notation for the development of our metrics. Section 3.2 provides a conceptual representation along with the formal definition for the NoC, NoF and ADIT-LN metrics.

3.1 Common formal notation

In order to define metrics, we introduce the following formal notation. This formal notation is used in the mathematical definition of our metrics.

- Let $C_1, C_2, \ldots, C_m$ be the set of m classes defined in an ontology.
- Let $P_1, P_2, \ldots, P_n$ be the set of n properties define in an ontology.
- Let $F_{c1}, F_{c2}, \ldots, F_{cm}$ be the Fanout of each class C_i

  (see below for definition of Fanout)
- Let O_1 be the ontology of interest.
- Let $\rightarrow$ be a mapping from set C_i, to C_j such that $C_i \rightarrow C_j$ if class C_j is a subclass of class C_i.
- Let & be a mapping from set C_i to set P_j such that C_i and P_j if class C_i includes property P_j as part of its definition.

OWL defines many constructs that allow the definition of a tree-based graph to determine the relationship between entities defined in the ontology. The following terminology is used in this paper to describe the tree-based relationships in our paper.

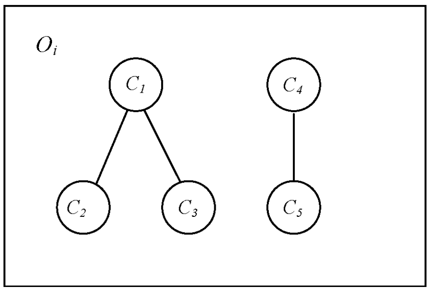
- Let A be a finite set, and let T be a relation on A. We say that T is a tree if there is a vertex V_0 in A with the property that there exists a unique path in T from V_0 to every other vertex in A (Kolman and Busby, 1987).
- Let A be a finite set, and let T be a relation on A. A vertex V_n has a Fanout of degree m if there exist m relationships to other vertices in A (Kolman and Busby, 1987).
- Let A be a finite set, and let T be a relation on A. A vertex V_q is called a leaf if it has Fanout of degree 0.

3.2 Metrics definitions

3.2.1 Definition of Number of Classes (NoC)

As suggested by its name, NoC, is the number of distinct classes defined in the ontology of interest O_i. Figure 1 shows a conceptual representation of NOC; in which the value of NOC = 5.

Figure 1 Number of Classes (NoC)



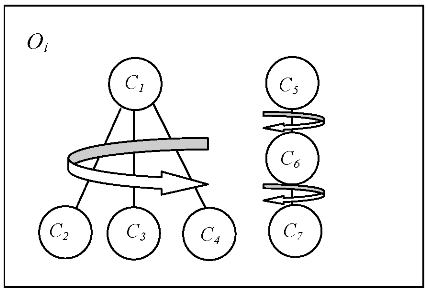
Mathematically, NoC is represented as follows:

$$\text{NoC}(O_i) = \sum C_j \text{ for } 1 \leq j \leq n \text{ (number of classess in } O_i\text{)}.$$

3.2.2 Definition of Number of Fanouts (NoF)

NoF, is the total number of fanout from all root, non-leaf nodes defined in the ontology of interest O_i. Figure 2 shows a conceptual representation of NoF in which the value of NoF = 5.

Figure 2 Number of Fanout (NoF)



Mathematically, NoF is represented as follows:

$$\text{NoC}(O_i) = \sum F_j \text{ for } 1 \le j \le n \text{ (number of classess in } O_i\text{).}$$

3.2.3 *Definition of Average Depth of Inheritance Tree of all Leaf Nodes (ADIT-LN)*

ADIT-LN is the sum of the length of all paths from a root to leaf nodes divided by the number of leaf nodes.

Mathematically, ADIT-LN is represented as follows:

$$\text{ADIT-LN}(O_i) = \sum D_j/n \text{ for all } D_j$$

(D_j is total number of nodes on each distinct path); $1 \le j \le n$ (number of paths in O_i.). The distinct paths are counted from a root class to each leaf. The root class is considered as level 1; therefore in Figure 3, the path from C_1 to C_2 is counted as 2, and so on for all distinct paths.

Figure 3 Average Depth of Inheritance Tree of all Leaf Nodes (ADIT-LN)

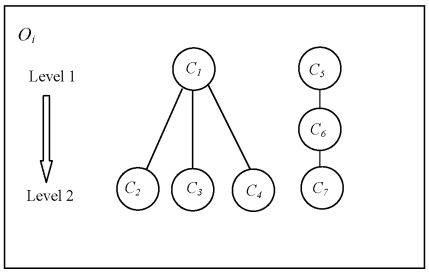


Figure 3 shows a conceptual representation of ADIT-LN; in which the value of ADIT-LN = 9/4 = 2.25.

4 Analysis of metrics

4.1 *Theoretical analysis*

4.1.1 *Analysis of Number of Classes (NoC)*

To analyse NoC within the framework proposed by Kitchenham et al. (1995), we first define the following:

- the entity being analysed as O_i, the ontology of interest
- the attribute being measured as the NoC
- the unit as the class
- the data scale is interval.

NoC is a direct measure of counting the NoC in the ontology. This measure meets Kitchenham et al.'s four properties for validity as follows:

- *attribute validity*: the entity (ontology) has the attribute (NoC), a measure of the total NoC defined in the ontology
- *unit validity*: the idea is an appropriate measure of NoC
- *instrumental validity*: the instrument is valid as long as our tool parses and counts correctly the NoC defined in the ontology
- *protocol validity*: the measurement as defined in the formal notation and shown conceptually given is consistent, and prevents double counting.

4.1.2 Analysis of Number of Fanouts (NoF)

To analyse NoF within the framework proposed by Kitchenham et al. (1995), we first define the following:

- the entity being analysed as O_i, the ontology of interest
- the attribute being measured as the total NoF
- the unit as the fanouts per class
- the data scale is interval.

NoF is a direct measure of counting the total NoF in the ontology. This measure meets Kitchenham et al.'s four properties for validity as follows:

- *attribute validity*: the entity (ontology) has the attribute (total NoF), a measure of the total NoF per classes defined in the ontology
- *unit validity*: the idea is an appropriate measure of total NoF
- *instrumental validity*: the instrument is valid as long as our tool parses and counts correctly the NoF defined in the ontology
- *protocol validity*: the measurement as defined in the formal notation and shown conceptually given is consistent, and prevents double-counting.

4.1.3 Analysis of Average Depth of Inheritance Tree of all Leaf Nodes (ADIT-LN)

To analyse ADIT-LN within the framework proposed by Kitchenham et al. (1995), we first define the following:

- the entity being analysed as O_i, the ontology of interest
- the attribute being measured as the ADIT-LN
- the unit as the depth of inheritance
- the data scale is interval.

ADIT-LN is a direct measure of counting the depth of the inheritance tree for all leaf nodes in the ontology. This measure meets Kitchenham et al.'s four properties for validity as follows:

- *attribute validity*: the entity (ontology) has the attribute (ADIT-LN), a measure of the average depth of inheritance in the ontology
- *unit validity*: the idea is an appropriate measure of ADIT-LN
- *instrumental validity*: the instrument is valid as long as our tool parses and counts correctly the depth of inheritance of each leaf node
- *protocol validity*: the measurement as defined in the formal notation and shown conceptually given is consistent, and prevents double counting.

4.2 Empirical analysis

To perform our empirical analysis of NoC, NoF and ADIT-LN we computed the metrics using 33 ontologies developed by Creative Commons (Chen, 2006). To perform our computation we developed Ontology Metrics Parser (OMP), a simple XML parser that uses Java API for XML Parsing (JAXP). OMP calculated NoC, NoF, and ADIT-LN for all 30 ontologies.

Then we assembled a panel of 18 evaluators to assess each of the 30 ontologies to determine each ontologys' complexity. Our evaluators' average experience in developing software projects was 3.5 years. The evaluators were given an electronic copy of each ontology and ask to rate the complexity of each. The evaluators rated the complexity of each of the 30 ontologies on the following scale:

- 0 = low
- 0.25 = moderately
- 0.50 = average
- 0.75 = high
- 1.0 = extremely.

First, using Minitab software we computed the Interrater reliability to determine how well our evaluators agree with one another; for example, did evaluator 1 give a rating of 0.25 and evaluator 2 give a rating of 1.0 on the same ontology O_i. Therefore, Interrater reliability addresses the consistency of the implementation of a rating system. Interrater reliability is expressed as a real number in the range of [0 … 1].

In our measure of Interrrater reliability we considered the responses of all 18 evaluators across all 33 ontologies. We used MiniTab's Two-Way Mixed Effect Model, considering the people effect to be random and the measurement effect to be fixed. The Interrater reliability for our evaluators is 0.9160, which shows that the agreement between our evaluators is consistent.

We then averaged the evaluator ratings for each of the ontologies and performed our statistical analysis to correlate the averaged evaluator data with the computed values NoC, NoF, ADIT-LN. Then we correlated the metrics, NoC, NoF, ADIT-LN with the evaluator data. We used Pearson's correlation coefficient with the following hypotheses for our correlations.

H0: $\rho = 0$ (There is no correlation between the metric value and the team's value).

H1: $\rho \neq 0$ (There is a correlation between the metric value and the team's value).

Correlation coefficients range from –1.0 to 1.0 with numbers closer to either end stating a higher degree of correlation, meaning that the values being correlated are not independent of one another. If our correlation coefficient is closer to 0 then the variables being correlated are more independent of one another.

To understand the meanings of these values, Cohen (1988) proposed the following scale to determine the meaning of our correlations.

- <0.01 = trivial
- 0.10–0.30 = minor
- 0.30–0.50 = moderate
- 0.50–0.70 = large
- 0.70–0.90 = very large
- 0.90–1.0 = almost perfect.

Table 1 summarises the results of our correlation, all of which show our metrics are statistically valid and correlate at 0.50 or greater.

Table 1 Comparison of metric to evaluators across creative commons ontologies

Correlated metrics	*Correlation coefficient*	*P-value*	*Significant at 0.05*
NoC to evaluators	0.885	0.000	Yes
NoF to evaluators	0.802	0.000	Yes
ADIT-LN to evaluators	0.588	0.001	Yes
NoC to NoF	0.946	0.000	Yes
NoC to ADIT-LN	0.705	0.000	Yes
NoF to ADIT -LN	0.819	0.000	Yes

5 Application of ontology complexity metrics in bioinformatics and genomics

Since the need to characterise huge numbers of living things has resulted in an explosion of the number of ontologies available for bioinformatics and genomics, the Standards and Ontologies for Functional Genomics organisation has developed standards for integrating controlled vocabularies and ontologies. However, even with this standard, there are a large number of ontologies intended for the same or similar purposes, that is, that are intended to characterise some of the same living things. Also, many different researchers develop these ontologies for many different purposes, so there is a wide variation in how accurate the ontologies are, how rich in description they are, and how detailed they are. For example, one ontology might define an animal called a mouse. Another ontology might go further and define particular kinds of mice, such as a Salt Marsh Harvest mouse (this is a particular kind of endangered mouse that lives near San Francisco Bay). To avoid potential confusion based on the particular examples we have selected here, we note that ontologies are not used in bioinformatics and genomics just to define living creatures, they are used in many other ways as well, such as to define characteristics of living creatures. For example, ‘Mouse Anatomy’ and ‘Human Anatomy’ are ontologies that have been used in the past (Orme et al., 2006).

Given two different ontologies characterising the same living things, which of the ontologies should be selected for use? That is, of course, dependent on the particular need. Our complexity metrics could be used to help in this selection. If two ontologies have relatively the same terms defined, then the ontology that shows as more complex using our metrics is probably more detailed and thus is richer in which living creatures are identified, and in how many relationships between living creatures are identified. For example, the NOC metric would give an indication of the number of different living creatures defined in the ontology, whereas the ADIT-LN metric would give an indication of how specific the definitions are. For NOC, the larger the NOC the more living creatures defined. From the Salt Marsh Harvest Mouse example given above, a larger value for ADIT-LN would be more likely to define living creatures down to the Salt Marsh Harvest Mouse level, whereas a lower value for ADIT-LN would be more likely to indicate the definitions stopped at the level of Mouse. The NoF metric is more related to NOC than to ADIT-LN; the NoF metric could be used to identify how many different families of creatures are defined in the ontology.

However, the selection is not as simple as just choosing the ontologies with higher values for NOC, NoF, or ADIT-LN. If the ontology is getting larger and more complex then that is desirable from the standpoint of completeness; but from a desire for lack of ambiguity it might not be desirable. If the area covered by the ontology gets too large, it is possible in some cases that the same term might be used for different things, which adds to confusion. For example, a ‘hopper’ might be used as a synonym for ‘rabbit’ but if insects are also included in the ontology, a ‘hopper’ might also be used as a synonym for ‘grasshopper’.

6 Conclusions

We have presented a set of metrics used to determine the complexity of ontological based systems. These metrics were theoretically validated against known criteria for the development of valid metric definitions. The empirical analysis was done by correlating our metrics with a panel of evaluators, the correlations where shown to be in the range of large to very large as defined by Cohen (1988).

The use of metrics to measure the discrete properties of ontologies has significant application in bioinformatics and genomics, as we illustrated in Section 5.

The use of ontology metrics also has Business-to-Business (B2B) and Business-to-Consumer (B2C) applications in terms of automatic service composition. In many B2B and B2C applications service composition relies on matching process capabilities between service providers. Our ontology based complexity metrics would allow the possibility of service composition without the partial graph matching used in some service composition schemes.

7 Future

In the future, using our complexity metrics, possibly some bioinformatics software systems could select appropriate ontologies at runtime, on the fly. This would allow comparison of many ontologies for a particular use, by looking at the quality of the output of the bioinformatics software itself.

Currently, most system complexity measures are determined by syntactical measure after the coding is complete. We believe that for ontological based systems that our metrics may be used to determine complexity prior to the coding of a system, therefore reducing system development and maintenance cost and promoting a greater code reuse.

Although we currently view these metrics in terms of system development, future research may include new uses for these metrics, such as, from an AI perspective can we say that in some respect our metrics measure intelligence? If so, how can these metrics be used to determine relative intelligence of systems or ontologies?

References

Chen, H. (2006) *Creative Commons*, Retrieved 10/13/06 from http://daml.umbc.edu/ontologies/cobra/

Cohen, J. (1988) *Statistical Power Analysis for the Behavioral Sciences*, 2nd ed., Lawrence Erlbaum Publishing Co., Mahwah, NH.

Corazzon, R. (2006) *Descriptive and Formal Ontology, a Resource Guide to Contemporary Research*, Retrieved 10/13/06 from http://www.formalontology.it/

Kitchenham, B., Pfleeger, S. and Fenton, N. (1995) 'Towards a framework for software measurement validation', *IEEE Transactions on Software Engineering*, Vol. 21, pp.929–944.

Kolman, B. and Busby, R.C. (1987) *Discrete Mathematical Structures for Computer Science*, 2nd ed., Prentice-Hall, Inc., Englewood Cliffs, NJ.

Orme, A.M., Yao, H. and Etzkorn, L. (2006) 'Coupling metrics for ontology-based systems', *IEEE Software*, Vol. 23, No. 2, pp.102–108.

Stein, C., Etzkorn, L., Cox, G., Farrington, P., Gholston, S., Utley, D. and Fortune, J. (2004a) 'A new suite of metrics for object-oriented software', *Proceedings from SAM '04: The First International Workshop on Software Audit and Metrics*, Springer-Verlag, Heidelberg, pp.49–58.

Stein, C., Etzkorn, L., Utley, D., Farrington, P., Cox, G., Fortune, J. and Gholston, S. (2004b) 'Computing software metrics from design documents', *Proceedings from ACMSE '06: The Forty-Second Annual ACM Southeast Conference*, Association for Computing Machinery Press, New York, pp.146–151.

WC3 (2006) XML.org, Retrieved 10/13/06 from http://www.w3.org/XML/#intro

Yao, H., Orme, A.M. and Etzkorn, L. (2005) 'Cohesion metrics for ontology design and application', *Journal of Computer Science*, Vol. 1, No. 1, pp.107–113.

Bibliography

Briand, L., Daly, J. and Wust, J. (1998a) 'A unified framework for cohesion measurement', *Empirical Software Engineering*, Vol. 25, No. 1, pp.65–117.

Briand, L., Daly, J., Porter, V. and Wust, J. (1998b) 'A comprehensive empirical validation of design measures for object-oriented systems', *Proceedings of the 5th International Software Metrics Symposium*, IEEE Computer Society Press, Los Alamitos, CA, pp.246–257.

Briand, L., Morasca, S. and Basili, B. (1996) 'Property-based software engineering management', *IEEE Transactions on Software Engineering*, Vol. 22, pp.68–86.

Chidamber, S. and Kemerer, C. (1994) 'A metrics suite for object oriented design', *IEEE Transactions on Software Engineering*, Vol. 20, pp.476–493.

Etzkorn, L. (1997) *A Metrics-based Approach to the Automated Identification of Object-oriented Reusable Software Components*, Doctoral Dissertation, University of Alabama in Huntsville, Huntsville, AL.

Etzkorn, L. and Delugach, H. (2000) 'Towards a semantic metrics suite for object-oriented design', *Proceedings from TOOLS '00: The Thirty-Fourth International Conference on Technology of Object-Oriented Languages and Systems*, IEEE Computer Society Press, Los Alamitos, CA, pp.71–80.

Li, W. (1998) 'Another metric suite for object-oriented programming', *Journal of Systems and Software*, Vol. 44, pp.155–162.

Li, W. and Henry, S. (1993) 'Object-oriented metrics which predict maintainability', *Journal of Systems and Software*, Vol. 23, pp.111–122.

Park, R. (1992) *Software Size Measurement: A Framework for Counting Source Statements*, Technical Report SEI-92-TR-20, Software Engineering Institute, Pittsburgh, PA, pp.136, 137.

Pressman, R. (2001) *Software Engineering: A Practitioner's Approach*, 5th ed., McGraw-Hill, Boston.

Sirin, E., Hendler, J. and Parsia, B. (2003) 'Semi-automatic composition of web services using semantic descriptions', *Proceedings of ICEIS '03: Workshop on Web Services: Modeling, Architecture and Infrastructure,* Angers, France.

Sowa, J. (1984) *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley, Reading, MA.

Stein, C. (2004) 'Fine-grained semantic metrics for object-oriented software', *Proceedings from SERP '04: The International Conference on Software Engineering Research and Practice*, SERP, Las Vegas, NV.

Weyuker, E. (1998) 'Evaluating software complexity measures', *IEEE Transactions on Software Engineering*, Vol. 14, pp.1357–1365.