ArchVoc – Towards an Ontology for Software Architecture

Lenin Babu T
IIT-Kanpur, India.
lenin@cse.iitk.ac.in

Seetha Ramaiah M
IIT-Kanpur, India.
msram@cse.iitk.ac.in

Prabhakar T.V
IIT-Kanpur, India.
tvp@cse.iitk.ac.in

Rambabu D
Cisco Systems Pvt Ltd
rdudduku@cisco.com

Abstract

Knowledge management of any domain requires controlled vocabularies, taxonomies, thesauri, ontologies, concept maps and other such artifacts. This paper describes an effort to identify the major concepts in software architecture that can go into such meta knowledge. The concept terms are identified through two different techniques (1) manually, through back-of-the-book index of some of the major texts in Software Architecture (2) through a semi-automatic technique by parsing the Wikipedia pages. Only generic architecture knowledge is considered. Apart from identifying the important concepts of software architecture, we could also see gaps in the software architecture content in the Wikipedia.

1. Introduction

The exponential growth of the world's knowledge has made the job of organization and search very challenging and demanding. Ontology based techniques have gained acceptance as a means for tagging and performing semantic searches. In this paper we take the first step to construct an ontology for software architecture, generating a vocabulary. We talk about two methods of constructing the vocabulary: the first one using the back-of-the-book index from major books in software architecture; and the second using the Wikipedia. The first method is completely manual, where as the second one is a semi-automatic method which can be used for any domain.

Wikipedia has grown to be the largest encyclopedia on the Internet and is growing exponentially. The Wikimedia foundation provides a query interface which provides a way for user applications to query data directly from the MediaWiki servers. One or more pieces of information about the site and/or a given list of pages can be retrieved. Information can be queried in either a machine readable format (xml, json, php, yaml, wddx) or a human readable format. More than one piece of information may be requested with a single query.

The URL: *http://en.wikipedia.org/w/query.php* is the query interface to which the user or an application specifies certain parameters depending on the type of information one wants to get. Most important parameters of a query are: *format*, *what* and *titles*. A typical query is of the form:
http://en.wikipedia.org/w/query.php?format=...&what=...|...|...&titles=...|...|... & ...
A detailed discussion of the query interface can be found at [15].

The software architecture community in its short span has been able to come up with a body of knowledge which lends itself to enormous reuse - best practices like design patterns, tactics which seem to be more fine-grained than patterns, frameworks, factual information like benchmarks technology artefacts etc. Every organization also has its own expertise based on the design and development of problems they have solved earlier. This taxonomy or ontology of the software architecture enables the architect in understanding the existing best practices and the relationships between them and also provide a means to apply them to the new systems to be developed. This knowledge base will help the architect to analyze the system built.

In the process of creating the knowledge base first we have to identify the boundaries of software architecture. To identifying the boundary of the domain we have to create a controlled vocabulary which includes all the terms in this domain. Controlled vocabulary helps in eliminating meaningless terms, terms which are too broad or too narrow, preventing different authors from misspelling and slightly different form of the same term. From this controlled vocabulary we build the taxonomy, classification that arranges the terms in this controlled vocabulary into a hierarchy. Taxonomy helps by describing the subjects. Taxonomy defines *Broader term* and *Narrow term* relationships. Ontology will contain the terms and relationships between the terms. In ontology we can define many relationships like *Related term, Uses, Consists of etc.* In the process creating Ontology for this software architecture domain we have created taxonomy of the Vocabulary.

29th International Conference on Software Engineering Workshops(ICSEW'07)
0-7695-2830-9/07 $20.00 © 2007 IEEE

The rest of the paper is structured as follows: in the following section we describe the motivation behind our work. Section 3 describes related work done in this area. Section 4 describes the methodologies used to create the taxonomy. Section 5 puts in the picture the results obtained. Section 6 gives some comments and possible future work and finally section 7 gives a list of references.

2. Motivation

This section describes the main impetus behind our work. For an architect, thee taxonomy or ontology of the software architecture enables in understanding the existing best practices and the relationships between them. For the teacher and a student of Software Architecture, and ontology is also useful in knowing the important concepts and their relationships.

This ontology is also useful in situations like asset management. A document repository can be benefit through search enhanced with an ontology (semantic as against just syntactic).

Our fundamental motive is to come up with a better ontology which serves as a knowledge base for software architecture. We tried to compare two methods of ontology derivation, manual and semi automatic methods of deriving the vocabulary so that we can take the best derived vocabulary from both the approaches

3. Related Work

This section describes the work done by various people to classify different vocabulary terms in software architecture domain. In the Grady Booch Handbook of Software Architecture [16], large numbers of patterns are classified that allow comparisons across domains and architecture styles. Though it solves the domain expertise problem, there are no such relationships between Architecture Tactics and quality requirements mapped to the more real life problem domains. In [17], an ontology is described as that helps in reusing the architecture level documents with the help of a knowledge base management tool for maintaining documents. This paper mainly deals with viewpoints used in the documents and does not take care of the other architecture properties. In [18], the design patterns are classified. There is no emphasis on mapping these patterns to other architecture properties and design methodologies.

In SWEBOK [19] Software Engineering topics are classified using Bloom's taxonomy [20]. Bloom's taxonomy is well known and widely used classification of cognitive educational goals. Main classifications include *Knowledge, Comprehension, Application, Analysis, Synthesis, and Evaluation.* Each topic in Software Engineering domain is classified using the bloom's taxonomy.

In [23, 24, 26], an approach to annotate and search architecture documents and a preliminary vocabulary has been presented.

We used two methods for deriving the vocabulary for this ontology which we shall describe in detail in the next section.

4. Methodology

Here we describe the two methods of collecting the vocabulary of a given domain: first one using back-of-the-book index and the second one using Wikipedia.

4.1 Manual Method:

Using the back-of-the-book index from several books related to a certain domain, one can manually construct a controlled vocabulary of that domain. Given a domain, take as many books as possible related to that domain and write down all interesting terms from the back-side index of each book. Eliminate duplicates from the collected list of terms. And then try to form clusters of terms based on their conceptual relatedness, that perform an affinity analysis. Finally we will be left with a hierarchy of terms related to the given domain.

4.2 Method Using Wikipedia:

In a Wikipedia document, all the hyper-links pointing to other documents are favorable towards a controlled vocabulary. So, start with a seed term from the domain; get all the links present in the document whose title is the given seed term.

For example, to get all the links present in the Wikipedia document titled "Software architecture", use the URL:

http://en.wikipedia.org/w/query.php?format=xml&what=links&titles=Software_architecture

This query will return an XML file consisting of links present in the document titled "Software architecture". This XML file can be processed by some piece of code to get a list of terms. From this list, the user will select the terms which are of interest to his/her domain. Include these terms in the vocabulary. Considering each of these terms as seed, proceed recursively.

Implementing this method, we have built a tool called 'HyperOnto' in Python and Java. Figure 1 is a screen shot of our tool. Here, we are showing the second iteration in the process of collecting the controlled vocabulary of the "Software architecture" domain.

5. Results

We are interested in the terms/concepts related to software architecture knowledge which is generic that is not related to any application domain or a particular problem [22].

We have obtained 3 different result sets, two of which have been generated using the back-of-the-book index method and the third one using our tool. Result set-1 is collected using four books from the Software Engineering Institute (SEI) series: [1], [4], [5] and [12]. Result set-2 is collected using these four books plus some best selling books from the Amazon's list: [2], [3], [6], [7], [8], [9], [10], [11], [13] and [14]. Result set-3 has been generated from Wikipedia using our tool. Table 1 shows a comparison of these results. Table 3 gives the complete list.

The terms are classified into nine concept categories as shown in Figure 2. Table2 shows a break up of the vocabularies mentioned in Table 1 in these categories. Figure 3 highlights the categories we have covered from the SWEBOK guide [19].

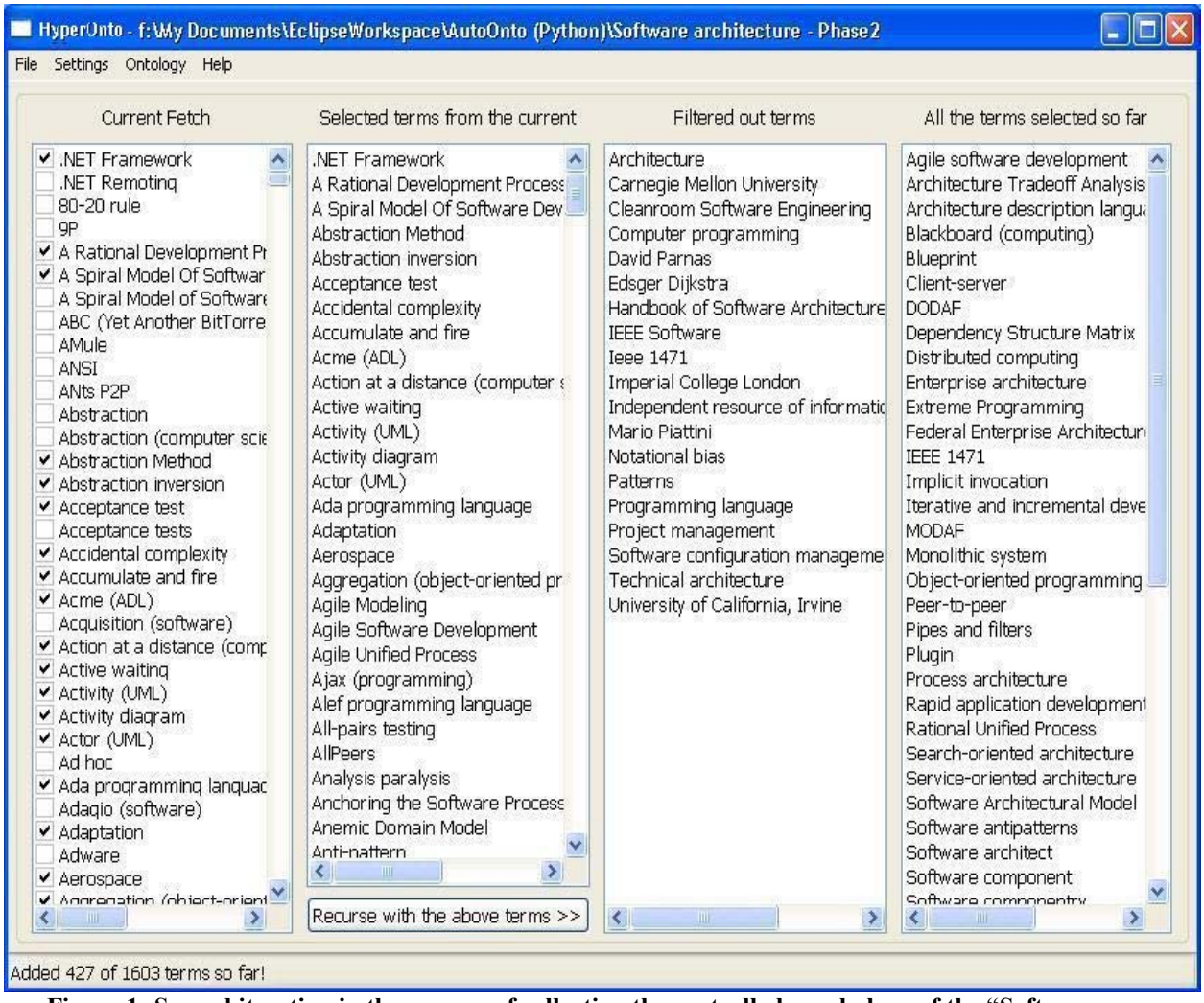

Figure 1: Second iteration in the process of collecting the controlled vocabulary of the "Software architecture" domain.

Result set #	Method Used	Source of Knowledge	Size of the Vocabulary
1	Back-of-the-book index	[1], [4], [5] and [12]	850
2	Back-of-the-book index	[1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13] and [14].	1650
3	HyperOnto	Wikipedia	480

Table 1 : Comparison of the result sets obtained by the two methods

S. No	Top-level category	From SEI series books	SEI series plus ten other books	Wikipedia
1	Antipatterns	0	47	77
2	Design patterns	41	208	58
3	Frameworks	4	9	5
4	Methodologies	10	11	17
5	Quality Attributes	18	53	24
6	Scenarios	16	18	0
7	Styles	20	70	7
8	Tactics	85	92	0
9	Views & View types	45	57	0

Table 2: Break-up of the vocabularies based on the count of the terms in each top-level category

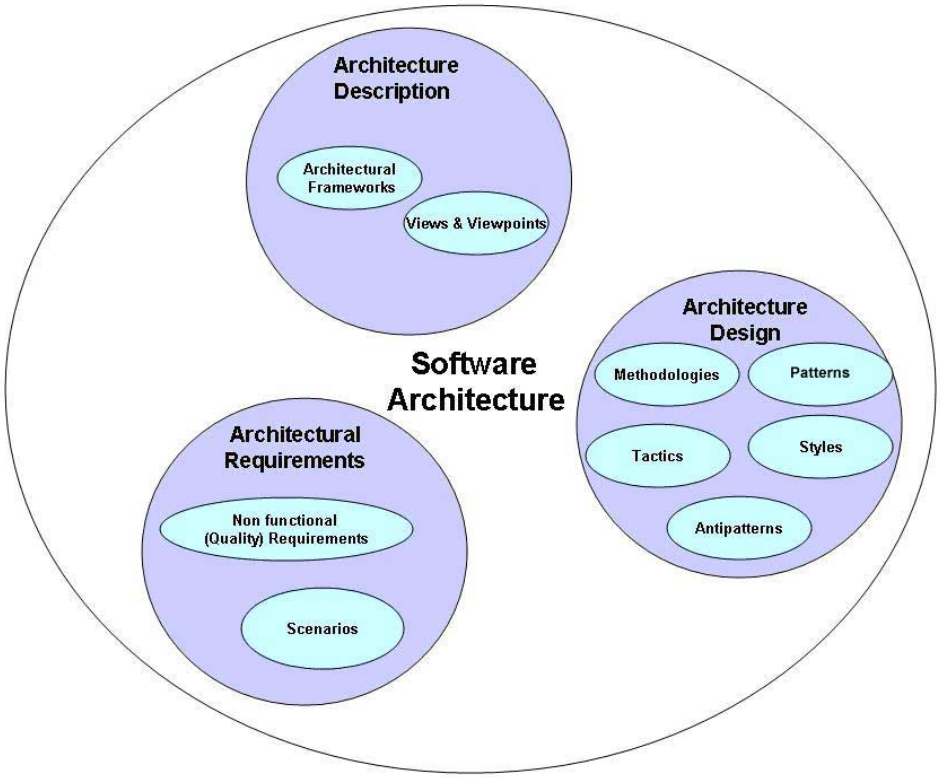

Figure 2: Concept categories of the "Software architecture" domain

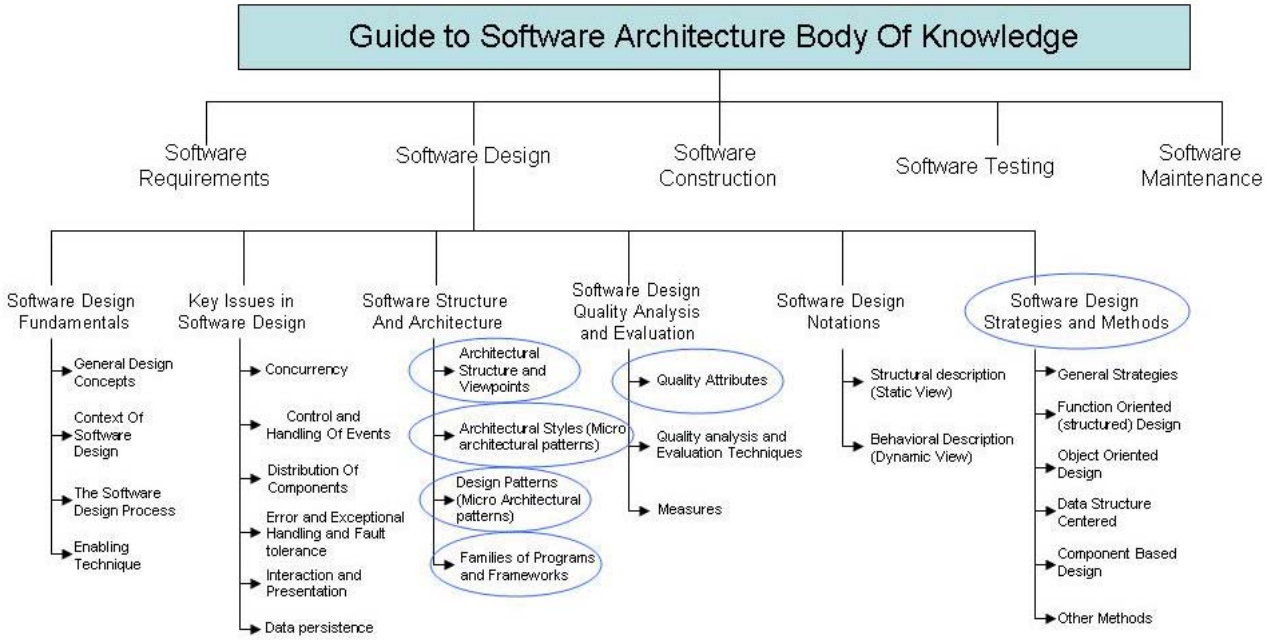

Figure 3: Knowledge Areas from SWEBOK which have contributed to the Vocabulary

6. Benefits of the Ontology

This ontology helps an architect in understanding of a new system and helps in its construction with existing best practices[26]. We have also said that this ontology can be used for educational purposes. For this we have made a Wiki entry http://wwwp.dnsalias.org/w/index.php/user:Thummalapalli. covering several architecture properties and concept maps showing the relationships between these architecture properties. We have also included the vocabulary derived from our methods so that users can evaluate the process and suggest more unbiased terms. We have developed the ontology as an Owl Standard file and is available to download from http://www.cse.iitk.ac.in/users/soft_arch/www/ontosoftarch/.

7. Comments and Future Work

This paper describes a first step towards constructing a full-fledged ontology for application generic software architecture knowledge. The vocabulary has to be substantially enhanced through consideration of other software architecture documents like books that have not been considered here, research publications and through expert reviews. The semantic relationships between these terms are lot more difficult to construct and requires more work. The HyperOnto tool may be extended to extract the relationships among the terms in the vocabulary using some NLP techniques and to help construct a full-fledged ontology of any domain.

There seem to be some gaps in the Wikipedia's current knowledge base for the Software Architecture domain, which can be seen in Table 2 of the results. There are no documents that talk about scenarios, tactics, views and view types. HyperOnto can be used to construct the controlled vocabulary of any domain. With slight modifications the same tool can be used for any online encyclopedia and can be used to compare several knowledge bases with regards to the coverage of the content related to a domain.

8. References

[1] Len Bass, Paul Clements, Rick Kazman, and Ken Bass. *Software Architecture in Practice*. 2nd Edition, Addison-Wesley, 2003.

[2] Jan Bosch. *Design and Use of Software Architectures*. 1st Edition, Addison-Wesley 2001.

[3] Stephen T. Albin. *The Art of Software Architecture: Design Methods and Techniques*, 1st Edition, Wiley, 2003.

[4] Paul Clements, Felix Bachmann, Len Bass, and David Garlan, *Documenting Software Architectures: Views and Beyond*. Addison-Wesley, 1st Edition, 2002.

[5] Paul Clements, Rick Kazman, and Mark Klein. *Evaluating Software Architectures: Methods and Case Studies* Addison-Wesley, 2002.

[6] Ian Gortan. *Essential Software Architecture,* 1st Edition, Springer, 2006.
[7] Gamma, Helm, Johnson, Vlissides.*Design Patterns: Elements of Reusable Object-Oriented Software*, 1st Edition, Edison Wesley, 1995.
[8] Frank Buschmann, Regine Meunier, Hans Rohnert, and Peter Sommerlad. *Pattern-Oriented Software Architecture, Volume 1: A System of Patterns*, 1st Edition, John Wiley & Sons, 1996.

[9] Douglas Schmidt, Michael Stal, Hans Rohnert, and Frank Buschmann. *Pattern-Oriented Software Architecture, Volume 2: Patterns for Concurrent and Networked Objects.,* John Wiley & Sons; 1 edition (September 14, 2000)
[10] Martin Fowler*, Patterns of Enterprise Application Architecture*, Addison-Wesley Professional; 1st edition (November 5, 2002)
[11] Brown, Malveau, and McCormick, *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis,* John Wiley & Sons; 1 edition (February 1, 2001)
[12] Paul Clements, Linda Northrop, and Linda M. Northrop, *Software Product Lines : Practices and Patterns*. 3Rev Ed edition ,Addison-Wesley, 2001.
[13] Raphael Malveau, Thomas J. Mowbray. *Software Architect Bootcamp*, Prentice Hall PTR, 2000
[14] Mary Shaw, David Garlan., *Software Architecture: Perspectives on an Emerging Discipline*, Prentice-Hall Engineering/Science/Mathematics, 1996
[15] *The Wikipedia query interface,.*
http://en.wikipedia.org/w/query.php
[16] *Handbook of Software Architecture*, Grady Booch.

[17] Christopher A., Welty David A. Ferrucci, *Formal Ontology for reuse of Software Architecture Documents,* Proceedings of the 14th IEEE international conference on Automated software engineering, 1999.
[18] Magnus Kardell, *A classification of object-oriented design patterns*
http://www.cs.umu.se/~jubo/ExJobbs/MK/patterns.htm
[19] *SWEBOK: A Guide to Software Engineering Body of knowledge,* 2004 version.
[20] *Bloom's Taxonomy*
http://www.officeport.com/edu/blooms.htm
[21] Philippe Kruchten, *A Taxonomy of Architectural Design Decision.,* Software Architecture Workshop Groningen, December 2-3, 2004.
[22] Lago, P., Avgeriou, P. *SIGSOFT Software Engineering Notes,* First ACM Workshop on SHAring and Keusing architectural Knowledge (SHARK)., Vol. 31(5), pp. 32-36. Sep. 2006.
[23] Rambabu Duddukuri, Prabhakar T.V., *On archiving architecture documents,* 12th Asia-Pacific Software Engineering Conference, 2005. APSEC '05
[24] Helping Architects In Retrieving Architecture Documents: A Semantic Based Approach, Rambabu Duddukuri, Prabhakar T.V. Semantic Matchmaking and Resource Retrieval: Issues and perspectives, Vldb Workshop, Seoul Korea 2006.
[25] CMAP tools: http://cmap.ihmc.us/
[26] Rambabu Duddukuri, Knowledge Management issues in software architecture documentation, M.Tech Thesis. http://www.cse.iitk.ac.in/gsdl/collect/cse/index/assoc/HASH013d.dir/doc.pdf