Jump-starting a Body-of-Knowledge with a Semantic Wiki on a Discipline Ontology

Víctor Codocedo, Claudia López, and Hernán Astudillo

Universidad Técnica Federico Santa María,
Avenida España 1680, Valparaíso. Chile
{vcodocedo,clopez,hernan}@inf.utfsm.cl
http://www.usm.cl/

Abstract. Several communities have engaged recently in assembling a Body of Knowledge (BOK) to organize the discipline knowledge for learning and sharing. BOK ideally represents the domain, contextualizes assets (e.g. literature), and exploits the Social Web potential to maintain and improve it. Semantic wikis are excellent tools to handle domain (ontological) representations, to relate items, and to enable collaboration. Unfortunately, creating a whole BOK (structure, content and relations) from scratch may fall prey to the "white page syndrome"[1], given the size and complexity of the domain information. This article presents an approach to jump-start a BOK, by implementing it as a semantic wiki organized around a domain ontology. Domain representation (structure and content) are initialized by automatically creating wiki pages for each ontology concept and digital asset; the ontology itself is semi-automatically built using natural language processing (NLP) techniques. Contextualization is initialized by automatically linking concept- and asset-pages. The proposal's feasibility is shown with a prototype for a Software Architecture BOK, built from 1,000 articles indexed by a well-known scientific digital library and completed by volunteers. The proposed approach separates the issues of domain representation, resources contextualization, and social elaboration, allowing communities to try on alternate solutions for each issue.

Key words: semantic wiki, body of knowledge, automated domain ontology, digital assets contextualization

1 Introduction

In recent years, several professional and academic communities have undertaken to organize and systematize their knowledge with a "Body Of Knowledge" (BOK for short). BOK's have been created most famously for project management

[1] Colloquial name for writers' mental block when starting a new piece from scratch

(PMBOK [2] by the PMI [3]) and for software engineering (SWEBOK [4] [5]), but also for IT architecture (ITABOK [6] by IASA [7] [8] [9]), and other related disciplines.

Body-of-Knowledge (BOK) requirements typically include representing the domain, contextualizing resources (e.g. literature), and relying on Social Web members to maintain and improve it. Semantic wikis are excellent tools to handle domain (ontological) representations, to relate items, and to enable collaboration. Unfortunately, creating a whole BOK (structure, content and relations) from scratch may easily lead to the "white page syndrome", given the size and complexity of the domain information.

This article presents an approach that differs from most current BOK's in exploiting a formal discipline description to maintain the knowledge organization. It also presents several tools to automate the creation of a domain conceptualization (in concepts of a populated ontology), a semantic wiki to manage the domain representation and its assets, stylized wiki elements, and a timeline-based browser to explore the domain.

The reminder of the article is structured as follows: section 2 summarizes earlier related work; section 3 introduces the proposed approach for building a BOK; section 4 explains how the wiki structure, content and linking are initialized; section 5 describes the ConcepTion tools that implement the proposal; section 6 suggests some future work; 7 summarizes and concludes.

2 Related Work

Several strands of work are directly related to this approach.

2.1 Semantic Wiki

Semantic Wikis are designed to allow collaborative creation of content using a fixed syntax and semantics to improve searching and querying. In traditional wikis it is possible to find basic *building blocks* to create content (on most wikis only a set of pages each one with a set of links). Semantic wikis provides an expanded set of *building blocks* such as relations, entity types and RDF or OWL annotations [4].

[2] PMBOK - Project Management Body Of Knowledge: `www.pmi.org/Resources/Pages/Library-of-PMI-Global-Standards.aspx`
[3] PMI - Project Management Institute: `www.pmi.org/`
[4] SWEBOK - Software Engineering Body of Knowledge: `www.computer.org/portal/web/swebok`
[5] ACM - Association for Computing Machinery: `www.acm.org/`
[6] ITABOK - IT Architect Body of Knowledge: `www.iasahome.org/web/home/skillset`
[7] IASA - International Association of Software Architects: `www.iasahome.org/`
[8] EABOK - Enterprise Architecture Body Of Knowledge: `www.mitre.org/work/tech_papers/tech_papers_04/04_0104/index.html`
[9] CBK - Common Body Of Knowledge: `www.cissp.com/`

Semantic Media Wiki [11] is a semantic wiki implementation that supports semantic templates creation, allowing to create fixed representations for each concept of the BOK. Semantic Media Wiki is an extension of the popular Media Wiki project[10], the platform on which Wikipedia works on. By this reason it provides a large set of useful extensions like SIMILE Timeline [11], an interactive Timeline browser.

The Kiwi wiki [19] (a EU-funded project) is another semantic wiki implementation that provides some advanced semantic annotation features, allowing a deeper granularity of the information (this feature was inherited from its predecessor IkeWiki [18]). It also provides what they call "Content Versatility", which are different views over the same content implemented by different applications. Unfortunately, Kiwi does not provides as many extensions as Semantic Media Wiki does. By using Kiwi, we think that we will lose some time on building them.

2.2 Semantic Digital Libraries and Ontology-based Approaches

Angelo di Iorio et al. [9] proposed WikiFactory to automatically create a domain semantic wiki from a domain ontology. Their work is based on customizing a semantic wiki from an ontology definition to add the content afterwards.

Jerome DL [13] is a semantic digital library whose main requirements are: provide user-oriented browsing features and allow efficient searching using semantic tools. The description of resources is based on Dublin Core[12]and FOAF[13]. Unfortunately, this two ontologies are quite simple on their specification. In that way, documents cannot be contextualized to a domain specific categorization for searching purposes.

ScholOnto [20] is a discourse ontology for describing Digital Libraries designed to support searching, tracking and analyzing concepts from academic perspectives. It is focused on expressing the *claims* that authors make on their documents. Although this is an interesting perspective we realize that such an approach leads to the "white page syndrome" as authors lack on time and motivation to fill templates with this information.

2.3 Bodies of Knowledge

There is not a single, common structure for all BOK's:

- The SWEBOK [22] is organized into ten knowledge areas (KAs): requirements, design, construction, testing, maintenance, configuration management, engineering management, engineering process, engineering tools and

[10] http://www.mediawiki.org
[11] SIMILE: www.simile-widgets.org/timeline/
[12] Dublin Core: `www.dublincore.org/`
[13] FOAF - Friend of a Friend Project: `www.foaf-project.org/`

methods, and quality. The SWEBOK contents were authored under the guidance, coordination and editing of a committee, originally composed of members of several professional societies; and benefited from systematic revision by hundreds of individuals.
- The PMBOK [17] identifies 44 processes, organized into five process groups and nine knowledge areas; the process groups are: Initiating, Planning, Executing, Controlling and Monitoring, and Closing; and the knowledge areas are: Project Integration Management, Project Scope Management, Project Time Management, Project Cost Management, Project Quality Management, Project Human Resource Management, Project Communications Management, Project Risk Management, and Project Procurement Management.
- The ITABOK [14], also called The Aspiring Architect Skills Library, is organized around a taxonomy of IT architect skills, proposed by IASA as well; the taxonomy categories are: Bussiness Technology Strategy, Design, Human Dynamics, Infrastructure, IT Environment, Quality Attributes, and Software. The ITABOK holds several articles in each category; topics were defined by a Training Committee, and bid on by practitioners.

Clearly, there are alternative notions of what a BOK is and how it should be written. But some generalizations can be made:

- A BOK is not just another textbook (an authoritative view by an individual or a committee); if so, it runs the risk of quickly becoming (or being born already) obsolete.
- A BOK can be created from resource collections, but it is more than their sum; otherwise, an overall "big picture" does not emerge.

Although digital assets (e.g. papers, learning objects, Web sites...) are important, a BOK cannot be just a search engine for assets.

3 Proposal

Building a body of knowledge (BOK) is expensive in human resources and time: it demands not only defining concepts and relations among them, but also requires a management system capable of support a whole community that will collaborate to create knowledge and enable inexperienced members of the community to understand the domain. To simplify and speed-up these requirements, we propose an ontology-based BOK which is semi-automatically populated from authoritative documents (such as articles). The BOK is enriched socially using the wiki, and is presented on a timeline to help better understand topics evolution in the community.

[14] `www.iasahome.org/web/home/skillset`

3.1 Ontology-based Body of Knowledge

There is a link between ontologies and BOK's: an ontology is a knowledge representation in which concepts are organized in hierarchies and are related to each other through relations, and a BOK is also a knowledge organization in which a discipline is presented through definitions of concepts. (REFERENCIA A MAX VOLKEL). Both ontologies and BOK's are knowledge organizations, their difference being for whom they are constructed: ontologies are intended to be machine-readable whereas BOK's are intended to be used and understood by humans. It is not only a format difference that arises here (structured information v/s free text).

Our approach tries to balance the trade-off between representation accuracy and usability of the organization [1] by maintaining a simple ontology that represents the Software Architecture discipline. Thus, we benefit from the good representation given by ontologies and the "good" user experience provided by BOKs. The ontology is created from authoritative documents, and the BOK presented to the user is based on a software architecture thesaurus and the manual organization provided by Software Architects.

3.2 An Ontology for Software Architecture from the Literature

From a very simplistic point of view, the more papers of a given domain a researcher is able to read, the more understanding he will have of what is happening with that domain. It should be possible to aid this process by automating the analysis of publications, using basic *Information Extraction* [6] techniques and *Concept frequency analysis*. Although clearly the process of understanding a discipline is not yet automatable, current technologies allow to jump-start the creation of a knowledge model such as an ontology. For this work we used and extended SKOS ontology [15] to model the Concepts of a domain. We added a new Class called *DigitalAsset* that represents a digital artifact that contains *explicit knowledge* about a Concept (REFERENCIA A VOLKEL DE NUEVO). The simplicity of the ontology we chose owes much to the design criteria for *Minimal Ontological Commitment* [8].

The publication full body is not used for analysis since it would require a much more complex and expensive process for extracting information. Instead, we analyze publications' metadata since simple, structured and also freely available on Internet from Web sites such as DBLP[16], CiteSeer[17] or ScienceDirect[18].

[15] SKOS - Simple Knowledge Organization System: `www.w3.org/2004/02/skos/`
[16] `www.dblp.org`
[17] `www.citeseer.org`
[18] `www.sciencedirect.org`

Table 1. Papers per Concept

no.	Concept	Digital Assets Set	Frequency
1	Architecture Rationale	p1,p2,p3,p4,p5,p6,p7	7
2	Reusability	p0,p2,p4,p5,p6	4

Mining digital assets metadata to extract Concepts The following excerpt is a typical Bibtex[19]entry provided by ScienceDirect[20].

```
@article{Kazman2005511,
title = "From requirements negotiation to software architecture decisions",
year = "2005",
...
author = "Rick Kazman and Hoh Peter In and Hong-Mei Chen",
keywords = "Requirements negotiation", "Architecture analysis",...
abstract = "Architecture design and requirements..."}
```

Three main fields may contain information of the Software Architecture discipline: keywords, title and abstract. We use keywords as a primary data source, since it is the simplest information available (tags of no more than 3 words). The analysis is based on two properties of the keywords:

- Keyword Frequency: If a keyword is present on several papers (that is, a keyword was used to tag several papers) that keyword represent an important Concept for the discipline that is being analyzed.
- Co-occurrence: If a subset of keywords is present on several papers, all the keywords in the subset are likely to be related to each other.

We extended the analysis to the Abstract field, which contains a short text comprising the main ideas of the content of the document. This text was used as a search-base for the Keywords (processed with Named Entity Recognition [21]).

This analysis yields a thesaurus with Concepts related to each other but with no hierarchy among them.

Creating a hierarchy of Concepts Given two Concepts related by co-occurrence analysis, we would like to know which Concept is broader and which one is narrower in the discipline, to add semantics to their relation. We proposed to identify and compare all digital assets associated to the Concepts. Table 1 shows two Concepts, each with an associated collection of digital assets.

[19] Bibtex is a tool and file format to describe and process references - see `www.bibtex.org`

[20] ScienceDirect: `www.sciencedirect.com`

[21] Named Entity Recognition is an Information Extraction technique used to identify entities on texts

Both Concepts co-occur on 4 different digital assets so we could say that they are related by co-occurrence. However, an 80% of the digital assets of the Concept #2 are contained on the set of concept #1, and only a 57% of the digital assets of concept #1 are in the concept #2 set (we call these percentages *co-ocurrence factors*). We can make the simple assumption that 80% of the literature of the concept *Reusability* is part of the literature of the concept *Architecture Rationale* and thus, *Reusability* represents something in the subdomain of *Architecture Rationale*. Since we cannot know what is this "something" that it represents we use a shallow relation stating only that *Reusability* is a narrower concept than *Architecture Rationale* (actually, *Reusability* of design rationale documents is a major goal of *Architecture Rationale*).

Applying this technique to every pair of co-occurrent concepts yields a hierarchy that emerges from the flat thesaurus built by mining the digital assets metadata. We can choose the minimal *co-ocurrence factor* to create the "narrower" relation between two concepts. We call this the *co-ocurrence filter*. Notice that a concept is not constrained to be *narrower* of only one concept (*Reusability* also is narrower than *Non-functional requirement*).

Enriching Keywords with a thesaurus The ontology built is used as a backbone of the BOK. That means that it should be as complete as possible to cover all the main aspects of the discipline on research. Nevertheless, using only the keywords provided by the authors of papers yields some drawbacks:

- Ambiguous Concepts: Authors often get too creative to tag their documents. Ambiguity is a main problem of tagging as author s will tag using their own knowledge (different from shared knowledge) (*architecture design, architectural design*).
- Too Generic Concepts: Some Concepts are too generic for the discipline and may not appear in the collection of Keywords since they do not represent a good tag for categorization. For instance, the word *System* is never used as a Keyword to tag a Software Architecture paper.
- Too Specific Concepts: Many Keywords are too specific and do not add useful information that can be used on the BOK. For example, proper names, identificators, etc. These kind of Keywords add noise to the final ontology.

To overcome these issues, the initial dictionary of concepts to search on abstracts is created over a thesaurus (we use a Software Architecture thesaurus presented by Fraga et al.[7]). The thesaurus plays a triple role in the process:

- Using tools such as lemmatization, we can anchor different tags to a single concept within the thesaurus ($\{$*architecture design, architectural design*$\} \Rightarrow \{$*Software Architecture Design*$\}$) reducing ambiguity.
- It adds words that, for being too generic, will not appear as Keywords on papers (*System* is a main concept in the thesaurus).

Too Specific Concepts need to be managed on a different way. We cannot just simply ignore all Keywords from papers' metadata and use only those on the

hand-made thesaurus because we would lose the capacity to discover information or new trends and topics. Specific concepts that cause noise are avoided by filtering them by the frequency they have. The idea is simple, the more specific a concept is, the less frequency it will have. Only concepts that appear in more than X papers will be used. We called X the *frequency filter*.

4 Use of Semantic Wiki for a BOK

The configuration of a wiki for the identified metamodel implies creating two kinds of pages: those representing domain concepts, and those representing digital assets. Both kinds of pages make use of specialized Infoboxes, allowing a standardized visual representation of the (concept or asset) attributes. The relationship between assets and concepts is represented by inter-pages referencing.

4.1 Discipline Exploration: Page per Concept

The ontology is later used on a semantic wiki, where a single wiki page is created for each concept (a little program in java was used to do such labor). At this point is necessary to understand that the we are providing a jump-start approach for the SABOK, but of course, the definitions and contents of this knowledge representation remains in the hands of the Software Architecture Community. Of course, some information is provided on the semantic wiki for each concept: Broader Concepts, Narrower Concepts, Associated Digital Assets, and Topic Category. According to the properties of the concept, we have created specific types of topics.

The semantic wiki allows the community to create and maintain content collaborative, populating and enriching the SABOK; explaining such tools is out of the scope of this work. Searching concepts can be done either with the free-text searching tool provided by the wiki framework, or by browsing the thesaurus used to build the ontology.

4.2 Resource Contextualization: Page per Asset

Since each concept on the SABOK has several digital assets associated (the same used to build the ontology), it can be used as a digital asset search tool as well. The ontology behind the SABOK allow us to use inference on answering queries. We have identified two inference levels: basic, and based on concepts.

Basic transitivity. Since the concepts are arranged on a hierarchy we can provide transitivity inference level for digital assets associated on a branch of concepts. For instance, all digital assets associated to *Reusability* will be answered to the query *"digital assets for Architecture Rationale"*.

As it can be seen, the SABOK besides from organizing the discipline knowledge, provides a searching capability of Digital Assets associated to each concept based on inference powered by its ontology.

4.3 Subject-based Exploration with Timelines

The generated BOK can be browsed with a timeline-based tool, which shows the evolution of concepts and how they relate to each other. A timeline-based visualization tool can show which concepts concite attention currently. Crosscuting concepts can be visually identified because they have a constant participation in the timeline over the years. Users can access the community-created information of concepts and the wiki itself to edit and manage it.

The information that tool requires resides as *year of publication* in the digital assets information (see section 3.2). In the timeline, the concepts are presented with the dates of the first and (currently) last publication that use it.

The timeline can also be used to present Digital Assets evolution around a concept. This should be really useful for researchers looking for the last publications according certain subject, for example.

Finally, new Digital Assets can be added to the SABOK, such as lessons, presentations, posters, video, etc.

5 Case Study: A Software Architecture BOK

The proposed approach has been implemented in a system named *ConcepTion* [22], composed of three main tools: a Miner, a Hierarchizer, and a Visualizer.

The approach was validated with a case study for the Software Architecture domain.

5.1 Software Architecture(s) Descriptions

Several efforts have been carried out to build a vocabulary for Software Architecture (SA). However, most of them are not intended to describe the entire Software Architecture discipline but systems and parts thereof (i.e. the discipline subject matter, not the discipline itself).

The SA community has recently focused on describing and recording architecture knowledge (AK) that supports the architecting process (e.g. adopted and discarded decisions, rationale, tradeoffs), and several metamodels and ontologies has been proposed to systematize it (PAKME [2], ADDSS [5], Archium [10], AREL [21], NDR [16], [12], among others). Also, Liang et al. [15] tackled the measuring of semantic distance among several proposals to describe AK, and defined a set of characteristics to categorize all AK concepts.

Unfortunately, only a couple of articles have proposed a broader description of the entire software architecture discipline. Babu et al. [14] introduced ArchVoc, the most cited software architecture ontology, which was generated with combined manual and semi-automatic techniques to identify software architecture concepts. The manual technique used the back-of-the-book index of major software architecture books, and the semi-automatic technique parsed

[22] `www.toeska.cl/conception/wiki/`

architecture-related Wikipedia[23]pages. The first approach yield 480 concepts, and the second one, 1650 concepts; they were organized into 9 overall categories, which were also sorted according to architecting phases.

Fraga et al. [7] also employed both an automatic and a manual technique to generate a software architecture thesaurus. The corpus of both generation techniques were the back-of-the-book index of major software architecture books (in 2005). The manual process yield a 500-concept thesaurus, and the automatic technique generated a 1200-concept thesaurus. Both thesauri were combined yielding 27 top-level concepts.

Although these two thesauri are good vocabularies to classify existing SA knowledge, there are several challenges that have not been already tackled in building a software architecture discipline vocabulary:

- Both thesauri have been manually manipulated to better classify SA knowledge, so their hierarchies and relationships are usually very influenced by existing conceptual frameworks present in the discipline. This aspect certainly helps to create good thesauri for information search, but it usually hampers its ability to describe real connections among concepts. For example, they group "fault-tolerance", "performance" and "usability" into a single category ("Quality Requirements" or "Non-Functional Requirements"), but in practice all three concepts are rarely present in the same article; indeed, most papers (and communities) focus on only one of them. Also, "fault-tolerance" is more frequently related to "validation" and "formal methods" than to any other quality requirement.
- The starting corpus of these thesauri did not include published research or industry articles, either; they used back-of-the-book indices and/or SA-related Wikipedia pages. This corpus selection reduces the vocabulary scope to those topics already published in books, omitting new trending topics or novel techniques that might be being discussed in major refereed SA conferences or journals. For example, none of these thesauri mention "design rationale" or "software architecture rationale", both dealt with in several recent mainstream articles.

5.2 Mining

The ontology was populated using 1,000 Bibtex files (including abstract) returned by ScienceDirect [24] for the "Software Architecture" search concept. Extracted metadata was stored in RDF [25]. Table 2 shows some statistics generated by the Miner.

Over 10% of the articles do not have an abstract in their Bibtex file, so we can only rely on the keywords that the authors used to tag them. Interestingly, only

[23] Wikipedia: `www.wikipedia.org`

[24] ScienceDirect: www.sciencedirect.com

[25] RDF: Resource Description Framework, the industry standard to store Semantic Information; see `www.w3.org/RDF/`.

Table 2. Statistics from ConcepTion Miner

Name	Value
Quantity of Papers	1000
Quantity of Papers with Abstract	886
Quantity of unique Concepts	2203
Concepts over 50%	47

47 tags account for more than the 50% of the matches produced by comparing searching dictionary concepts in abstracts. These are the most important and which we focused on.

5.3 Hierarchizer

The Hierarchizer compares every pair of concepts and calculates a co-occurrence factor between them, (see section 3.2). We can lower the *co-ocurrence filter* to find more relations among concepts, but of course, the lower it is, the more false positives we will find. We have found empirically that a *co-ocurrence filter* of 80% is appropriate to discover new relations and maintain false positives on a low level.

The *co-occurrence filter* and *frequency filter* (see section 3.2) are the two parameters that can be used to adjust the quality of the hierarchy obtained, and thus, the ontology's instances.

After creating the hierarchy, it can be visualizated with Graphviz[26]to draw the concepts and their relations, allowing Software Architecture experts to audit it and manually filter false-positives. Some samples of hierarchies can be found on Toeska's Website[27].

5.4 The prototype SABOK

The prototype SABOK was implemented using the semantic wiki platform Semantic Media Wiki [28] (SMW). A simple ad-hoc tool adds a wiki page for each concept in the hierarchy.

A timeline browser was also built with the MIT SIMILE Timeline[29]allowing to use HTML and JavaScript to use XML data, namely, a Knowledge Base with the ontology created.

Figure 1 shows a screenshot of the prototype SABOK. The evolution of the Concept *Architecture* is shown. Each line represents a narrower Concept displayed from the year of the first paper published with this Concept to the last paper. Figure 2 shows the wiki page for the Concept *Reusability*. Along with

[26] `www.graphviz.org/`
[27] Toeska Research Group, Universidad Técnica Federico Santa María: `www.toeska.cl`
[28] `www.semantic-mediawiki.org`
[29] SIMILE Project: `http://simile.mit.edu/timeline/`

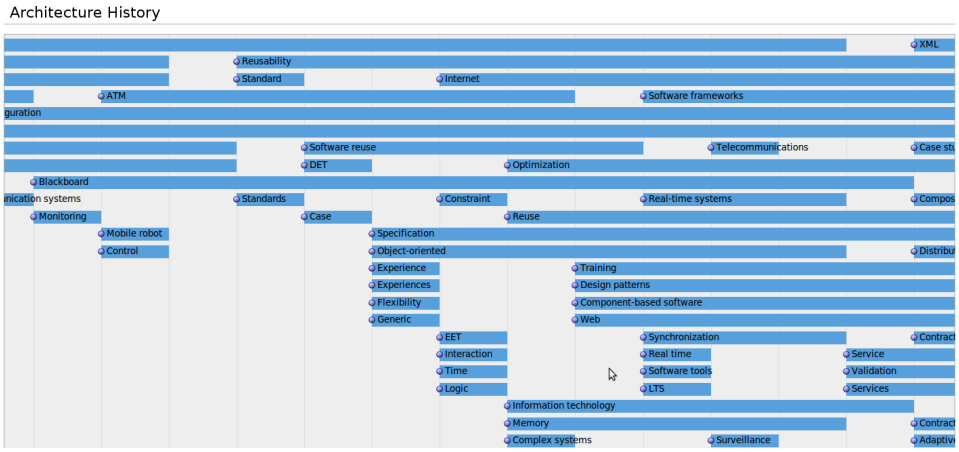

Fig. 1. Screenshot of ConcepTion SABOK Timeline - Architecture Concept Evolution

the information of broader concepts and narrower Concepts a Timeline of the publications using this Concept is provided. The Timeline is fully interactive and allow user to browse research papers. Figure 3 shows two infoboxes: Digital Asset and Concept. Digital Asset's infobox displays useful information such as title, author and Concepts used on this paper. It also provides information of inferenced Concepts related to the paper. Concept's infobox the upper and lower concepts in the hierarchy. It also displays inferred concepts and Digital Assets associated to the concept.

6 Further Work

Along with adding more advanced NLP tools and adding more papers to the analysis to improve our hierarchy, we believe that there are two topics that could add a lot of value to the SABOK presented.

- Cluster Analysis: Through cluster analysis we can understand better which are the areas the discipline is divided into. Also, it should be possible to acknowledge some useful intersections of areas and define them as different elements in the ontology to improve searching capability. We think that using Formal Concept Analysis tools would allow us to find this clusters of information by identifying classes of concepts as shown on PACTOLE methodology [3].
- Emerging Topics Tracking: With our approach is possible to find which are the most newer topics in the discipline and how they are related to each other. However, that does not mean that these are emerging topics. We think that emerging topics have a low frequency and thus, they will not emerge on our hierarchy. Besides that, we think that emerging topics appears on publications with a high impact factor and that's how we think that they

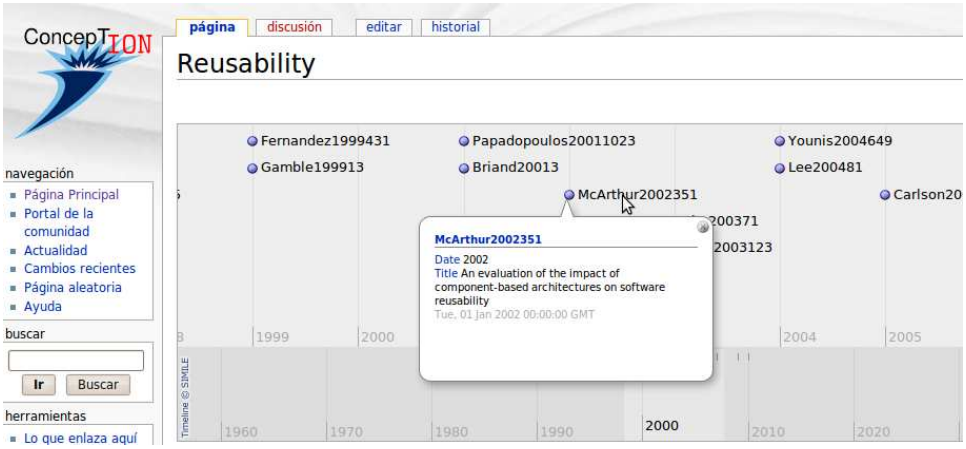

Fig. 2. ConcepTion wiki - Concept Reusability

should be identified. Though, that kind of information is not available on bibtex files and should be obtained on a different way.

Although we think the best validation for our SABOK should be made by the community, we are planning on making validation tests with Software Architects and Software Engineering students in the following months.

7 Conclusions

This article has presented a novel method to jump-start the creation of an ontology-based Body of Knowledge (BOK).

Using authorative documents from a community, we can mine and extract information about a discipline to hierarchize it and create an ontology. The ontology is used to organize the BOK and search Digital Assets (research publications in our example) using inference. The resulting BOK provides contextualization allowing document discovering and search inference.

The *ConcepTion* set of tools allows to extract, mine, hierarchize and display a BOK using a semantic wiki to manage information and a timeline tool to show evolution of topics in the discipline. The community is then asked to feed the BOK with definitions and their own Digital Assets. Future work will be focused on improving the quality of the resulting BOK and adding more features.

References

1. H. Astudillo. Maximizing object reuse with a biological metaphor. *TAPOS*, 3(4):235–251, 1997.

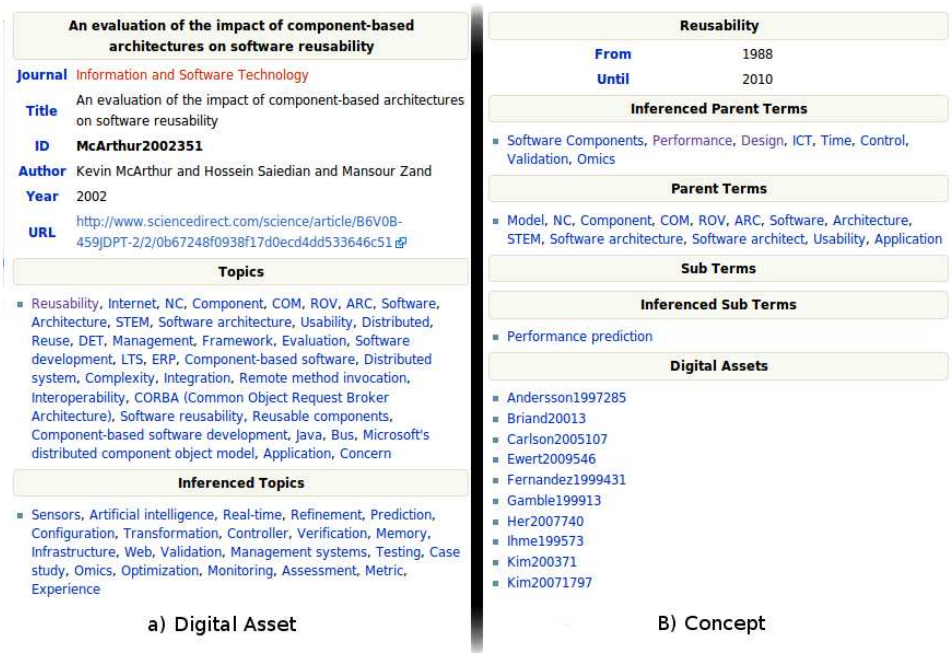

Fig. 3. Screenshot of ConcepTion Infoboxes

2. M. A. Babar, I. Gorton, and B. Kitchenham. Rationale management in software engineering. In *A Framework for Supporting Architecture Knowledge and Rationale Management*, pages 237–254. Springer Berlin Heidelberg, 2007.
3. R. Bendaoud, Y. Toussaint, and A. Napoli. Pactole: A methodology and a system for semi-automatically enriching an ontology from a collection of texts. 5113:203–216, 2008.
4. F. Bry, M. Eckert, J. Kotowski, and K. A. Weiand. What the user interacts with: Reflections on conceptual models for semantic wikis. In C. L. 0002, S. Schaffert, H. Skaf-Molli, and M. Völkel, editors, *SemWiki*, volume 464 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.
5. R. Capilla, F. Nava, S. Pérez, and J. C. Dueñas. A web-based tool for managing architectural design decisions. *SIGSOFT Softw. Eng. Notes*, 31(5):4, 2006.
6. H. Cunningham. *Encyclopedia of Language and Linguistics*, chapter Information Extraction, Automatic, pages 665–677. 2nd edition, 2005.
7. A. Fraga, S. Sánchez-Cuadrado, J. Lloréns, and H. Astudillo. Knowledge representation for software architecture domain by manual and automatic methodologies. *CLEI Electron. J.*, 9(1), 2006.
8. T. R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *Int. J. Hum.-Comput. Stud.*, 43(5-6):907–928, 1995.
9. A. D. Iorio, V. Presutti, and F. Vitali. Wikifactory: An ontology-based application for creating domain-oriented wikis. In Y. Sure and J. Domingue, editors, *ESWC*, volume 4011 of *Lecture Notes in Computer Science*, pages 664–678. Springer, 2006.
10. A. Jansen, J. van der Ven, P. Avgeriou, and D. K. Hammer. Tool support for architectural decisions. In *WICSA '07: Proceedings of the Sixth Working IEEE/IFIP*

Conference on Software Architecture, page 4, Washington, DC, USA, 2007. IEEE Computer Society.
11. M. Krötzsch, D. Vrandecic, M. Völkel, H. Haller, and R. Studer. Semantic wikipedia. *J. Web Sem.*, 5(4):251–261, 2007.
12. P. Kruchten, P. Lago, and H. van Vliet. Building up and exploiting architectural knowledge. In *QoSA'05: Second International Conference on Quality of Software Architectures*, pages 43–58. Springer Berlin / Heidelberg, 2006.
13. S. R. Kruk, M. Cygan, A. Gzella, T. Woroniecki, and M. Dabrowski. Jeromedl: The social semantic digital library. In S. R. Kruk and B. McDaniel, editors, *Semantic Digital Libraries*, pages 139–150. Springer, 2009.
14. B. T. Lenin, S. R. M., P. T. V., and R. D. ArchVoc-Towards an ontology for software architecture. In *SHARK-ADI '07: Proceedings of the Second Workshop on SHAring and Reusing architectural Knowledge Architecture, Rationale, and Design Intent*, page 5, Washington, DC, USA, 2007. IEEE Computer Society.
15. P. Liang, A. Jansen, and P. Avgeriou. Selecting a high-quality central model for sharing architectural knowledge. *Quality Software, International Conference on*, 0:357–365, 2008.
16. C. López, P. Inostroza, L. M. Cysneiros, and H. Astudillo. Visualization and comparison of architecture rationale with semantic web technologies. *Journal of Systems and Software*, 82(8):1198–1210, 2009.
17. Project Management Institute. *A Guide to the Project Management Body of Knowledge (PMBOK Guide) - Third Edition, Paperback*. Project Management Institute, 2004.
18. S. Schaffert. Ikewiki: A semantic wiki for collaborative knowledge management. In *WETICE '06: Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pages 388–396, Washington, DC, USA, 2006. IEEE Computer Society.
19. S. Schaffert, J. Eder, S. Grünwald, T. Kurz, and M. Radulescu. Kiwi — a platform for semantic social software (demonstration). In *ESWC 2009 Heraklion: Proceedings of the 6th European Semantic Web Conference on The Semantic Web*, pages 888–892, Berlin, Heidelberg, 2009. Springer-Verlag.
20. S. B. Shum, E. Motta, and J. Domingue. Scholonto: an ontology-based digital library server for research documents and discourse. *Int. J. on Digital Libraries*, 3(3):237–248, 2000.
21. A. Tang, Y. Jin, and J. Han. A rationale-based architecture model for design traceability and reasoning. *J. Syst. Softw.*, 80(6):918–934, 2007.
22. L. L. Tripp. *Guide to the Software Engineering Body of Knowledge: 2004 Version*. 2005.