Journal of Software Engineering Research and Development
a SpringerOpen Journal

RESEARCH Open Access

Personalized architectural documentation based on stakeholders' information needs

Matias Nicoletti*, Jorge Andres Diaz-Pace, Silvia Schiaffino, Antonela Tommasel and Daniela Godoy

*Correspondence:
matias.nicoletti@isistan.unicen.edu.ar
ISISTAN Research Institute, CONICET-UNICEN, Paraje Arroyo Seco, Campus Universitario, Tandil, Argentina

Abstract

Background: The stakeholders of a software system are, to a greater or lesser extent, concerned about its software architecture, as an essential artifact for capturing the key design decisions of the system. The architecture is normally documented in the Software Architecture Document (SAD), which tends to be a large and complex technical description, and does not always address the information needs of every stakeholder. Individual stakeholders are interested in different, sometimes overlapping, subsets of the SAD and they also require varying levels of detail. As a consequence, stakeholders are affected by an information overload problem, which in practice discourages the usage of the architectural knowledge and diminishes its value for the organization.

Methods: This work presents a semi-automated approach to recommend relevant contents of a given SAD to specific stakeholder profiles. Our approach assumes that SADs are hosted in Wikis, which not only favor communication and interactions among stakeholders, but also enable us to apply User Profiling techniques to infer stakeholders' interests with respect to particular documents.

Results: We have built a recommendation tool implementing our approach, which was tested in two experiments with Wiki-based SADs. The experiments aimed at assessing the performance reached by our tool when inferring stakeholders' interests. To this end, precision and recall metrics were used.

Conclusions: Although preliminary, the results have shown that the recommendations of the tool help to find the architectural documents that best match the stakeholders' interests.

Keywords: Stakeholders; Architectural documentation; Software architecture; Wikis; Personalization; Recommender systems

1 Contents

This article is organized as follows. Section 2 presents the introduction of this article. Section 3 discusses related work on architecture documentation. Section 4 provides background information about the V&B method, and then presents the details of our approach in terms of user profiles, NLP tools and similarity metrics. Section 5 is devoted to the experiments used to evaluate the approach. Section 6 discusses the results of the experiments of this study. Finally, Section 7 gives the conclusions and discusses future lines of work.

2 Background

Software Architecture is a useful model for describing the high-level structure of a system in terms of components, responsibilities allocated to those components, and relationships among them (Bass et al. 2012). The software architecture plays an important role in early development stages as the container of the main design decisions for satisfying the stakeholders' concerns. An example of decisions is the use of certain patterns, such as layers or client-server, to meet modifiability or performance qualities. In a development project, the software architecture is typically captured by the Software Architecture Documentation (SAD), which acts as a channel of communication and knowledge sharing among the stakeholders of the project (Clements et al. 2010). Along this line, a SAD must be clear in explaining: i) how the main functional requirements are addressed by the different software components, and ii) how the component structure satisfies the quality-attribute requirements of the system (e.g., performance, availability, modifiability). As indicated in the ISO Standard (ISO/IEC/IEEE 2011), a challenging aspect of the SAD is that is targeted to multiple readers (i.e., stakeholders such as managers, developers, architects, customers, testers, sub-contractors), which might have different backgrounds and information needs. Generally, each reader needs the architectural knowledge in order to understand specific parts of the system and perform tasks related to the project.

A common problem that stakeholders face when they consume information from a SAD is that of *information overload*. For instance, recent studies (Koning and Vliet 2006; Su 2010) have shown that many individual stakeholder's concerns are addressed by a fraction (less than 25%) of the SAD, but for each stakeholder a different (sometimes overlapping) SAD fraction is needed. In practice, the process of creating and maintaining a SAD tends to be a low-priority or underestimated activity in many projects (due to budget constraints, tight schedules, or pressures on developing features, among other reasons). A common approach is to produce a generic document loaded with development-oriented contents. However, this approach is not the most convenient solution from a multiple-stakeholder perspective. In addition, generic SADs are often extensive and complex, and therefore, stakeholders have difficulties in accessing the information needed for their tasks.

In this work, we aim at improving the ways by which stakeholders access and find relevant information in architectural documents, in the context of Wiki environments. In particular, we have investigated techniques to infer the interests of a user[a] as he/she works with a Wiki-based SAD, and then generate specific recommendations of SAD sections (i.e., Wiki documents) that might be relevant to that user. These kind of recommendations are useful when the size of the documentation is large, as it is common in architectural documentation. To this end, the techniques should determine the relevance of a given SAD section for each user by analyzing the characteristics of the SAD sections and the work context of the users. In other words, we need to characterize (or model) users' interests, preferences and goals with respect to the SAD. Our approach is based on the construction of *user profiles* (Schiaffino and Amandi 2009; Castro-Herrera et al. 2009). Initially, profiles for different stakeholder types are derived from the Views and Beyond (V&B) method for architectural documentation (Clements et al. 2003). As stakeholders browse the SAD, their profiles are enriched with information coming from Wiki documents via Natural Language Processing (NLP) techniques (Baeza-Yates and Ribeiro-Neto 2011) and implicit interest indicators (Al halabi et al. 2007; Claypool et al. 2001). We apply

the NLP techniques to perform a semantic analysis of the SAD textual contents through concept and tag mining (Nicoletti et al. 2012; Nicoletti et al. 2013a).

Our approach was initially presented in a previous work (Nicoletti et al. 2013b), and we employed a similarity function based on TF.IDF for matching Wiki documents against user profiles. We have extended that work by developing a recommendation tool for our approach, which works integrated into a Wiki-based system (DokuWiki[b]). In this article, we additionally investigate alternative similarity functions and assess their performance empirically. The preliminary results discussed in (Nicoletti et al. 2013b) for a given SAD are complemented with an additional experiment that uses a different SAD and different test subjects. The results of this second experiment confirmed the trends of the first experiment regarding the performance of our approach and usefulness of the recommendations. The results also shed light on the selection of similarity metrics (according on their performance) for a future deployment of our tool in real-life software development environments.

3 Related work

Several documentation methods for architectural knowledge have been proposed in the literature. Relevant examples include: *Kruchten*'s *4+1 View Model*, *Software Systems Architecture, Siemens 4 Views*, and *SEI Views & Beyond* (Clements et al. 2010). These methods prescribe a structure for the SAD (i.e., a template) and promote the use of separate architectural views. These methods provide few or no guidelines for generating the documentation corpus, and the *documenters* (in general, the architecture team) need to determine the *right contents* for each section of the template.

In particular, Views & Beyond (V&B) (Clements et al. 2003) is an appealing method for our work, since it proposes a role-based personalization of the SAD contents (a role here is a stakeholder type). The basic V&B principle states that documenting a software architecture involves documenting the *relevant views*, and then documenting information that applies to more than one view. A view is deemed as relevant if some stakeholder cares about it, or if the view addresses a key quality-attribute aspect of the system. Also, this method defines some documentation guidelines, such as the combination of views or the adjustment of the detail level for each view.

However, V&B still presents some drawbacks in practice. The method assumes that the stakeholder profiles are static (in time) and that their interests can be inferred just using the project role. In our opinion, the stakeholders' interests might change during the project lifetime, and they might be influenced by other factors, in addition to the stakeholder's role. A conditioning factor is the stakeholder's background. For instance, two stakeholders who share the developer role but work in different sub-systems might have different architectural interests (and thus, require different subsets of the SAD). Another factor is the stakeholder's reading history through the SAD (Su 2010). Also, V&B is often viewed by practitioners as a heavy-duty method, due to the amount of documentation to be generated in order to fulfill the SAD template. A contribution of our approach to fosters the applicability of V&B is the provision of more accurate profiles of interests regarding architectural documentation, thanks to the usage of User Profiling techniques. Furthermore, in our proposal, the contents of user profile can be adjusted as the user's working context changes during project life cycle.

To the best of our knowledge, only a few previous works (Castro-Herrera et al. 2009; Su 2010) have considered User Modeling techniques for architectural documentation. Su 2010 proposed an automated approach to deal with chunks of architectural information. These chunks are the result of specific exploration paths followed by a stakeholder (or user) when reading a SAD. The relevance of a given chunk is determined by factors such as the time spent by a reader on a section, or the access frequency of a section. The idea is that, when a new user is about to navigate the SAD, a tool can assist him/her to find relevant information by reusing previous similar exploration paths (from other users). A prototype tool has been recently developed (Su et al. 2011). Unfortunately, this approach still lacks an empirical performance evaluation, in contrast with our approach.

Related to (Su et al. 2011), de Boer and van Vliet (de Boer and van Vliet 2008) investigated the application of Latent Semantic Analysis (LSA) techniques to software documentation with auditing purposes. LSA is used for helping auditors in the search of specific architectural topics (e.g., terms like architecture, scenario, performance, SOA, etc.) across several documents by creating a "reading guide". In background, the LSA algorithm constructs a vector-space model for the documents. In this approach, the auditors must explicitly indicate the terms of interest (or "search query"), so as to steer the navigation through potentially-relevant documents. This approach is related to ours in the sense that documents are modeled with term-based representations. A difference with our approach is that we do not require explicit queries, because the user interests are inferred semi-automatically to generate personalized recommendations.

Castro-Herrera et al. 2009 proposed a user modeling approach to support the requirements elicitation process. A recommendation system is used to link relevant forums with the project stakeholders. For this task, the system builds user profiles by combining a term-based model (extracted with the help of NLP techniques) with context information, such as the stakeholder's role or his/her preferences for specific requirements or quality attributes. This technique is of particular interest for our work, since it is a possible strategy to solve the problem when the SAD is supported by collaborative tools.

The nature of the software architecting process makes it suitable for employing collaborative web-based tools, such as Wikis. In the last years, several successful experiences of Wiki-based tools applied to architecting tasks have been reported (Farenhorst and van Vliet 2008; Unphon and Dittrich 2010). A Wiki is an effective communication channel that improves the sharing of architectural knowledge among stakeholders. From the perspective of User Profiling, an advantage of hosting the documentation (e.g., a SAD) in a Wiki is that users' interactions can be monitored in order to gather information to build user profiles.

Graaf et al. 2012 presented an empirical study on semantic Wikis. In this research, there is a SAD based on a Wiki that supports semantic annotations and, in particular, includes an ontology of Software Architecture concepts. The authors argued that ontology-based SADs are more effective than traditional file-based approaches. Effectiveness here refers to the ability of user to find relevant information according to his/her interests. In a controlled experiment with a small group of software professionals, the authors showed evidence supporting their hypothesis. Although the objectives of this research are different to ours, our approach shares the usage of a Software Architecture ontology. which is part of a semantic dictionary (explained in Section 4.1).

4 Our approach

In order to share the architecture knowledge among the stakeholders, it must be adequately documented and communicated. The Software Architecture Document (SAD) is the usual artifact for capturing this knowledge. The SAD format can range from Word documents to a collection of Web pages hosted in a Wiki. The latter format is becoming common nowadays (de Graaf et al. 2012; Farenhorst and van Vliet 2008; Jansen et al. 2009). The SAD is generally structured around the concept of *architectural views*, which represent the many structures that are simultaneously present in software systems. A view presents an aspect or viewpoint of the system (e.g., static aspects, runtime aspects, allocation of software elements to hardware, etc.). Therefore, the SAD consists of a collection of documents (according to predefined templates) whose contents include views (e.g., module views, components-and-connectors views, allocation views) plus textual information about the views (e.g., system context, architectural drivers, key decisions, intended audience).

Stakeholders are important actors in the documentation process as they are the main consumers of the SAD. By *stakeholder* (Mitchell et al. 1997), we mean any person, group or organization that is interested in or affected by the architecture (e.g., managers, architects, developers, testers, end-users, contractors, auditors). We argue that the value of a SAD strongly depends on how its contents satisfy the *stakeholders' information needs*. As we mentioned in Section 3, the V&B method proposes a stakeholder-based strategy for organizing the SAD views and their contents (Clements et al. 2003). V&B characterizes several types of stakeholders and then links them to specific architectural views, based on the anticipated usage of the views by the stakeholders and their preferred level of detail for the views. This information is summarized in the matrix of Figure 1.

We see the V&B characterization of stakeholders as a basic form of *user profiles*, and then propose a semi-automated approach that leverages on these profiles (and enriches

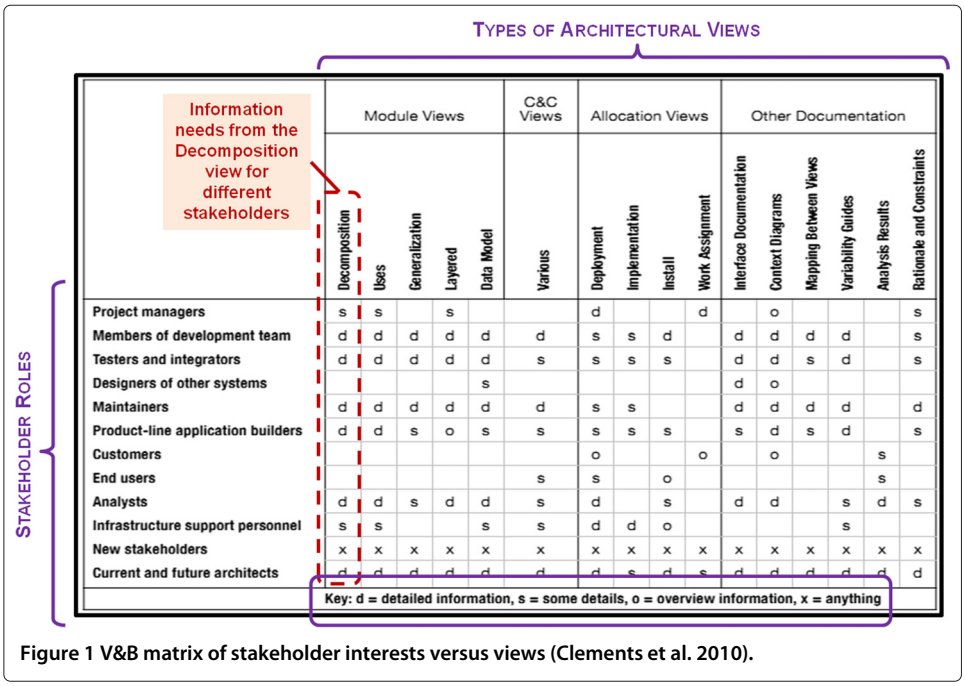

Figure 1 V&B matrix of stakeholder interests versus views (Clements et al. 2010).

them) for establishing links with relevant SAD documents. From this perspective, the SAD is personalized according to the stakeholders' information needs, as captured in his/her profile. The process requires a certain time to learn the user interests and build accurate profiles. This situation is known as the "cold start" problem (Schiaffino and Amandi 2009). Initially, the profiles are only based on the V&B matrix of stakeholders' preferences for architectural views. The "cold start" phase lasts until the system is able to gather additional information about stakeholders' interests so as to enrich the profiles. The user's browsing activity over a Web-based SAD is an example of such an information.

A key aspect of our approach is the granularity of SAD contents when mapped to Wiki pages. This granularity defines the "unit of recommendation" of our tool. In particular, we used one Wiki page per architectural view, plus one Wiki page per additional section (documentation beyond views) of the SEI's V&B template, as it has been suggested by other Wiki-based SADs based on V&B^c (Clements et al. 2003). However, this mapping choice is not mandatory.

A general schema of our profile-based recommendation approach is depicted in Figure 2. The design consists of a pipeline of processing units that: i) generates user profiles, ii) generates document representations, and iii) computes matching relationships among users and SAD documents. We refer to these relationships as *relevance links*, which are actually the recommendations provided by our tool. The relevance links are computed on the basis of the similarity between the user profiles and the document representations. A detailed description of our pipeline can be found on Section 4.3.

4.1 Inputs of the approach

The inputs needed to perform the analysis of stakeholders and construct their profiles are the following:

- *SAD textual contents*: The plain text from SAD documents (or sections) is automatically processed using NLP techniques (see Section 4.5) in order to generate documents models.
- *Interest indicators:* When users interact with Web pages, several interest indicators can be recorded (Al halabi et al. 2007; Claypool et al. 2001). In particular, we analyzed

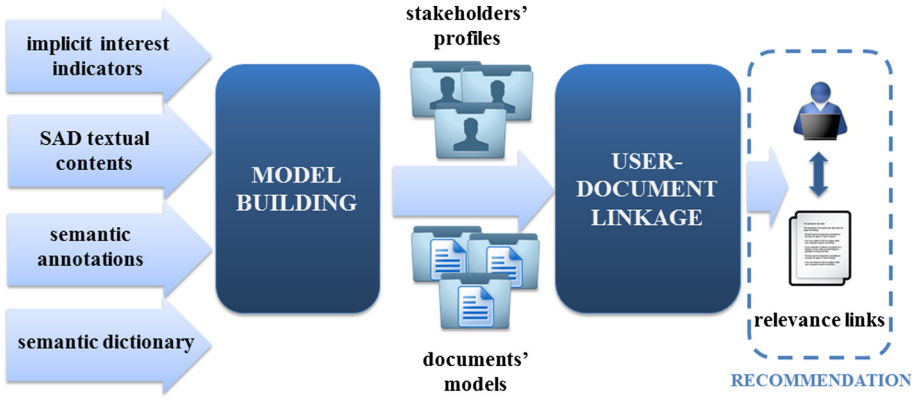

Figure 2 Overview of our recommendation approach.

indicators such as: time spent on reading a page, number of visits, mouse scrolls, mouse clicks, and the ratio between scrolls and time (which represents the frequency of scrolls while reading of Web page), among others (Claypool et al. 2001).

- *Semantic dictionary:* We considered a dictionary composed of a hierarchy of concepts and categories. Categories are concepts with a higher level of abstraction. This kind of semantic source of knowledge is derived from a previous work (Nicoletti et al. 2013a). Instead of using a general-purpose dictionary, we here customized it to consider only Software Architecture concepts along with their corresponding categories. This dictionary was built by combining concepts from an existing ontology for software architectures (de Graaf et al. 2012) with concepts described by the SEI's software architecture bibliography (Bass et al. 2012). We should note that thesauri commonly used for NLP tasks, such as WordNet or Wikipedia (Nicoletti et al. 2013a), are not specific enough in the Software Architecture domain of knowledge.
- *Semantic annotations:* Our approach needs to build a model (or representation) of each SAD document. We argue that these models can be enriched with explicit annotations provided by an expert (e.g., a member of the software architecture team), who, in general, will also generate the SAD contents. This expert is able to select those concepts or categories that best describe the semantics of each document. The annotations are considered part of the document representations. Moreover, the annotations are helpful to model documents/sections of the SAD template that are partially completed (or even empty), or to refine the representation of documents that are not accurately described by their textual contents (e.g., documents that contain many images and little text).

Our approach is regarded as semi-automated because the intervention of experts is required to input semantic annotations in the SAD documents. The expert also makes annotations on role types and, thus, incorporates V&B-related information. The initial stakeholders' profiles are mainly filled in with these annotations. For both annotation tasks, the expert uses the semantic dictionary as a "label catalog". The rest of the tasks and computations can be performed automatically.

4.2 Modeling users and documents

Both user profiles and documents are represented by the same structure. This structure comprises two parts: i) a set of semantic concepts and categories, which are extracted from the dictionary mentioned above, and ii) a set of tags (or keywords) (Schiaffino and Amandi 2009; Nicoletti et al. 2013a). For each item (i.e., concept, category or tag), the number of occurrences is recorded as the item frequency.

In our context, tags are keywords or non-trivial words often extracted with NLP techniques. Trivial words, such us pronouns or prepositions, are usually excluded. Concepts are basic units of meaning that serves humans to organize and share their knowledge. For instance, the English Wikipedia articles have been used as a source of concepts. Categories are concepts with a higher level of abstraction, which might be link to concrete concepts or to other categories with different level of abstraction. In our dictionary, for instance, *performance*, *fault tolerance* and *security* are examples of concepts within the category *quality atributes*, which, in turn, is linked to the high-level category *requirements*.

A user profile or a document model is a triple $M =< CON, CAT, TAG >$, in which:

- $CON = \{con_1, \ldots, con_n\}$ is a set of concepts, where con_i $(1 \leq \text{i} \leq \text{n})$ is a pair $< C, F >$ with C as the concept and F as its frequency.
- $CAT = \{cat_1, \ldots, cat_m\}$ is a set of categories, where cat_i $(1 \leq \text{i} \leq \text{m})$ is a pair $< C, F >$ with C as the category and F as its frequency.
- $TAG = \{tag_1, \ldots, tag_t\}$ is a set of tags, where tag_i $(1 \leq \text{i} \leq \text{t})$ is a pair $< T, F >$ with T as the tag and F as its frequency.

This representation is convenient to calculate similarities between users and documents, as well as for quickly describing the interests of a given user or the contents of a given document. For instance, Figure 3 shows how the model of user interests might look like. We also decided to combine both concepts and tags, since the SAD is generally composed of general concepts and problem-specific concepts. Some examples of problem-specific concepts are: names of software components, specific stakeholders' names, and tactics and patterns that are not included in the common catalogs, among others. The general concepts are defined in our semantic dictionary, whereas the problem-specific concepts are mined from the text.

4.3 The processing procedure

The process is divided into three main stages, as depicted in Figure 4. First, the *document representation generation* is performed. This stage runs a NLP semantic analysis of the documents (hosted in the Wiki), and afterwards merges the semantic annotations with the partial representation of those documents. We refer to a model (of a document or a user) as being "partial" when its constituents (e.g., tags) must be refined by running one or more processing units. First, the document annotations are included in the partial models of documents. A prefixed value (parameter N) denotes the weight (or frequency) that each annotation would have in the document model. Second, a NLP analysis of documents is performed. To this end, we have configured a sub-pipeline of NLP tasks that

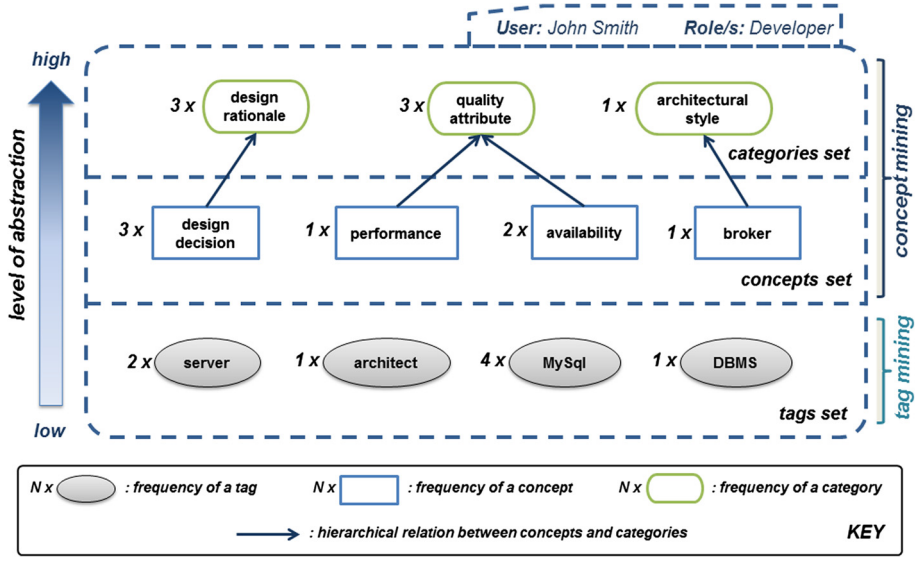

Figure 3 Graphical example of a user profile.

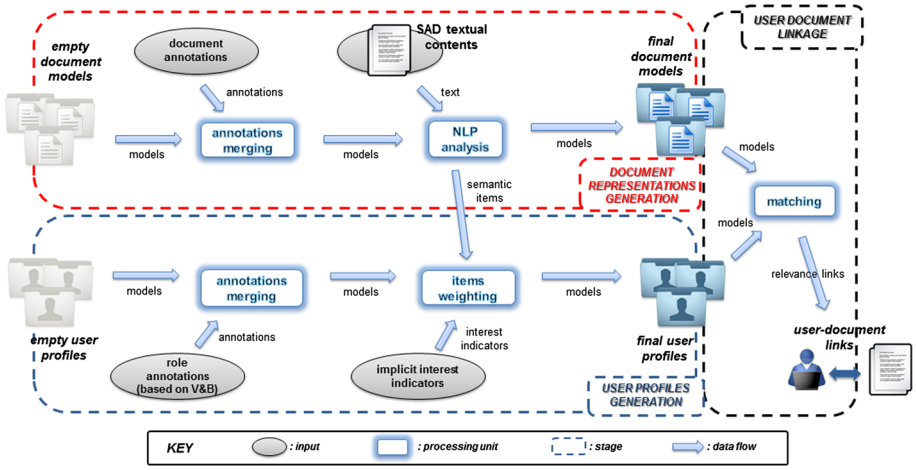

Figure 4 Detailed view of the pipeline.

produces a term-based representation of the documents. The details of this sub-pipeline are described in Section 4.5.

Second, the *user profiles generation* stage takes place. The initial user profiles, which are empty at this point, are enriched with semantic annotations coming from the roles (or stakeholder types) associated to each user. The role annotations and the user profiles are merged as described in the previous stage. Next, we perform what we call *semantic items weighting*: the semantic items that were extracted from the documents visited by a given user are added to that user profile. We assume that when a user accesses a SAD section, the contents of that section are likely to be relevant to that user. Therefore, we consider interest indicators from usage statistics to weight the relevance of the semantic items incorporated into user profiles. In particular, we used the number of visits as the frequency of the new items for the user profile. For example, if a user visited a given document N times, and that document contains a concept X and a tag T, then both X and T are incorporated to the user profile with a frequency of N. This indicator was prioritized over the others based on an empirical assessment of its relevance for inferring user interests (see Section 5).

Finally, the *user document linkage* stage is executed. In this stage, the models for both users and documents have already been generated. These two kinds of models are processed by an algorithm that determines the degree of matching (or similarity) between the models. In this article, we analyzed several metrics to compute the similarity between two models (see Section 4.4) and compared them empirically (see Section 5).

The output of the complete procedure is a set of weighted links between users and documents, in which the weights indicate the relevance of the documents for each user. The output is grouped by user and anked in descending order to select the most important links per user. A threshold k is used to establish the number of documents retrieved as relevant. For example, if we have N different sections in a SAD, with $k = 1/4$ we are considering the first $1/4N$ sections as relevant and the other $3/4N$ as irrelevant. Figure 5 shows an snapshot of our recommendation tool at work.

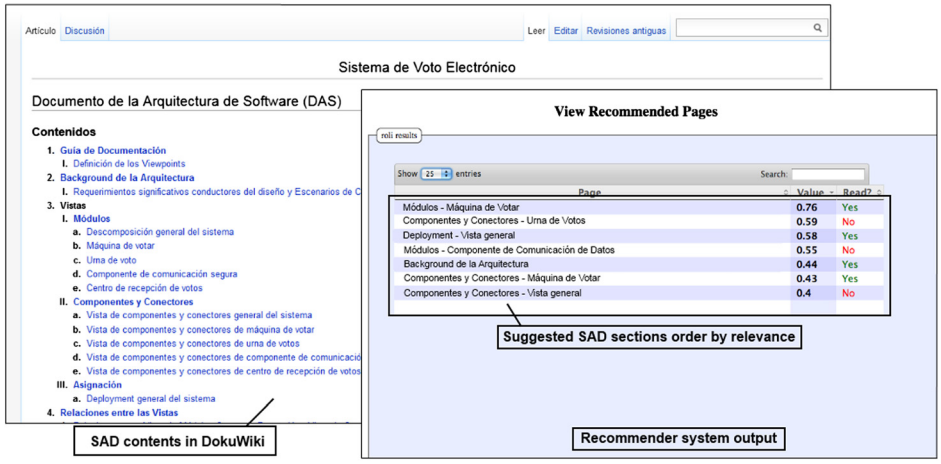

Figure 5 Our recommendation tool integrated in the DokuWiki environment action.

4.4 Computing similarity between models

In general, the strategies to compute the similarity (or distance) between two items map the similarity between the symbolic descriptions of two objects to a unique numerical value (Huang 2008). In this section we describe different strategies we used to compute the degree of matching between user profiles and documents models. For each strategy, we present a short description, its formal definition and, if necessary, some consideration for its usage. It is worth noting that for those strategies that compute the distance between two items, we consider the similarities as the inverse of such distance.

4.4.1 Euclidean distance

The Euclidean distance represents the distance between two points in space, given by the Phytagorean formula. It is one of the most widely used distances for numerical data (Deza and Deza 2006; 2009; Liu 2011). Equation 1 (Deza and Deza 2009) is the formal definition of this distance, where $\overrightarrow{t_a}$ and $\overrightarrow{t_b}$ are the vectors to be analyzed, and $w_{t,a}$ and $w_{t,b}$ are the weights associated to attributes t_a and t_b respectively. In this case, weights correspond to number of occurrences.

$$Euclidean\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \sqrt{\sum_{t=0}^{m} \left(w_{t,a} - w_{t,b}\right)^2} \tag{1}$$

4.4.2 Manhattan distance

This measure takes its name from its geometrical interpretation in the so called Taxicab geometry, in which the distance between two points is the sum of the absolute differences of their Cartesian coordinates. Its name allude to the grid layout of most streets on the island of Manhattan, which causes the shortest path a car could take between two intersections in the borough to have a length equal to the intersections distance in taxicab geometry. Equation 2 shows the formal definition of the Manhattan distance between two vectors $\overrightarrow{t_a}$ and $\overrightarrow{t_b}$.

$$Manhattan\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \sqrt{\sum_{k=0}^{n} \left|w_{t,a} - w_{t,b}\right|} \tag{2}$$

4.4.3 Chebyshev distance

This strategy defines the distance between two distributions considering the maximum difference between their attributes. It is a metric defined on a vector space where the distance between two vectors is the greatest of their differences along any coordinate dimension. Equation 3 shows its formal definition (Deza and Deza 2009).

$$Chebyshev\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = max\left\{\left|w_{1,a} - w_{1,b}\right|, \ldots, \left|w_{m,a} - w_{m,b}\right|\right\} \tag{3}$$

4.4.4 Cosine similarity

The representation of distributions as vectors enables us to measure the similarity between them as the correlation between the corresponding vectors (Huang 2008). Such similarity can be quantified as the cosine of the angle between them. The strategy is independent of the length of the distributions, and it is one of the most widely used in information retrieval systems. Equation 4 presents the formal definition of the cosine similarity of vectors $\overrightarrow{t_a}$ and $\overrightarrow{t_b}$, where $\overrightarrow{t_a} \cdot \overrightarrow{t_b}$ represents the inner product of these vectors, and $\left\|\overrightarrow{t_a}\right\|$ and $\left\|\overrightarrow{t_b}\right\|$ represent their norms (Deza and Deza 2009; Liu 2011).

$$Cosine\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \frac{\overrightarrow{t_a} \cdot \overrightarrow{t_b}}{\left\|\overrightarrow{t_a}\right\| \left\|\overrightarrow{t_b}\right\|} = \frac{\sum_{t=0}^{m}\left(w_{t,a} * w_{t,b}\right)}{\sqrt{\sum_{t=0}^{m} w_{t,a}^2} * \sqrt{\sum_{t=0}^{m} w_{t,b}^2}} \tag{4}$$

4.4.5 Cosine distance

Given the previous equation, the cosine distance in Equation 5 shows how to adapt it to compute the distance between two distributions.

$$CosineDistance\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = 1 - Cosine\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) \tag{5}$$

4.4.6 Kullback-Leibler divergence

This strategy is also known as information gain or relative entropy (Huang 2008), and it is defined as in Equation 6.

$$D_{KL}\left(\overrightarrow{t_a} || \overrightarrow{t_b}\right) = \sum_{t=1}^{m}\left[w_{t,a} * log\left(\frac{w_{t,a}}{w_{t,b}}\right)\right] \tag{6}$$

The formula presents an indetermination condition when $\frac{w_{t,a}}{w_{t,b}}$, and returns 0, that is when $w_{t,a}$ or $w_{t,b}$ are equal to 0. To avoid this indetermination and be able to compute the formula correctly, a correction is applied to those value equal to 0. These values are replaced by 10^{-6}, so that it does not affect the original distributions. Additionally, the strategy is not symmetrical, that is $D_{KL}\left(\overrightarrow{t_a} || \overrightarrow{t_b}\right) \neq D_{KL}\left(\overrightarrow{t_b} || \overrightarrow{t_a}\right)$. In this context, it cannot be used to measure distances. To overcome this difficulty, the average divergence is computed using the formula in Equation 7, where $\pi_1 = \frac{w_{t,a}}{w_{t,a}+w_{t,b}}$, $\pi_2 = \frac{w_{t,b}}{w_{t,a}+w_{t,b}}$ and $wt = \pi_1 * w_{t,a} + \pi_2 * w_{t,b}$.

$$AverageKullback\left(\overrightarrow{t_a} || \overrightarrow{t_b}\right) = \sum_{t=1}^{m}\left[\pi_1 * D_{KL}\left(w_{t,a} || w_t\right) + \pi_2 * D_{KL}\left(w_{t,b} || w_t\right)\right] \tag{7}$$

4.4.7 Dice-Sorensen similarity

The Dice-Sorensen coefficient is a statistic used for comparing the similarity of two samples (Dice 1945; Sørensen 1948). It is based on an analysis of the presence or absence of data in the samples considered. As compared to Euclidean distance, Sorensen distance

retains sensitivity in more heterogeneous data sets and gives less weight to outliers. Its formal definition is given by Equation 8 (Deza and Deza 2009).

$$\text{Dice-Sorensen}\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \frac{2 * \left|\overrightarrow{t_a} \cdot \overrightarrow{t_b}\right|}{\left\|\overrightarrow{t_a}\right\|^2 + \left\|\overrightarrow{t_b}\right\|^2} \tag{8}$$

4.4.8 Jaccard distance

The Jaccard distance is a statistic used for comparing the similarity and diversity of sample sets. This strategy is very useful to analyze text similarity in huge collections (Rajaraman and Ullman 2012). Equation 9 shows the formal definition of this strategy where t_a and t_b represent the attributes of distributions $\overrightarrow{t_a}$ and $\overrightarrow{t_b}$ respectively.

$$Jaccard\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = 1 - \frac{|t_a \cap t_b|}{|t_a \cup t_b|} = \frac{|t_a \cup t_b| - |t_a \cap t_b|}{|t_a \cup t_b|} \tag{9}$$

4.4.9 Overlap coefficient

The overlap coefficient, also known as Simpson similarity, is a similarity measure related to the Jaccard index that computes the overlap between two sets, which is defined as Equation 10 (Deza and Deza 2009).

$$Overlap\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \frac{|t_a \cap t_b|}{min\left\{\left|\overrightarrow{t_a}\right|, \left|\overrightarrow{t_b}\right|\right\}} \tag{10}$$

4.4.10 Pearson correlation coefficient

This strategy measures how related two distributions are. Equation 11 shows its formal definition, where $TF_a = \sum_{t=1}^{m} w_{t,a}$ and $TF_{ba} = \sum_{t=1}^{m} w_{t,b}$.

$$Pearson\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \frac{m * \sum_{t=1}^{m}\left(w_{t,a} \times w_{t,b}\right) - TF_a * TF_b}{\sqrt{\left[\left(m * \sum_{t=1}^{m} w_{t,a}^2 - TF_a^2\right) * \left(m * \sum_{t=1}^{m} w_{t,b}^2 - TF_b^2\right)\right]}} \tag{11}$$

This strategy gives as result a value in the range $[-1, +1]$, being 1 when $\overrightarrow{t_a} = \overrightarrow{t_b}$.

4.4.11 Pearson correlation coefficient distance

This strategy applies a change to Pearson correlation coefficient so that the value of the metric fits in the range $[0, +1]$ and hence, the strategy represents the distance between two distributions $\overrightarrow{t_a}$ and $\overrightarrow{t_b}$.

$$PearsonDistance\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \begin{cases} 1 - Pearson\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) & if\ Pearson\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) \geq 0 \\ \left|Pearson\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right)\right| & if\ Pearson\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) < 0 \end{cases} \tag{12}$$

4.4.12 Tanimoto distance

The Tanimoto distance can be defined as a variation of Jaccard distance (Huang 2008). It compares the weights of shared attributes with the weights of those attributes that belong to one of the distributions but are not shared between them. The strategy calculates the similarity between two distributions $\overrightarrow{t_a}$ and $\overrightarrow{t_b}$ giving a value in the range $[0, 1]$, being 1 when $\overrightarrow{t_a} = \overrightarrow{t_b}$ and 0 when the distributions are completely different. The formula of the distance is shown in Equation 14.

$$TanimotoSem\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \frac{\overrightarrow{t_a} \cdot \overrightarrow{t_b}}{\left\|\overrightarrow{t_a}\right\|^2 + \left\|\overrightarrow{t_b}\right\|^2 - \overrightarrow{t_a} \cdot \overrightarrow{t_b}} \tag{13}$$

$$Tanimoto\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = 1 - TanimotoSem\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) \tag{14}$$

4.4.13 TF.IDF-based similarity function

In addition to the previous similarity metrics, we propose a candidate function that specifically fits the problem of matching stakeholder profiles against architecture documents. Our function is based on the TF.IDF (Term Frequency x Inverse Document Frequency) metric of the Information Retrieval field (Baeza-Yates and Ribeiro-Neto 2011). The function is computed as indicated in Equation 15, in which U is a triple describing a user profile, D is a triple describing a document model (Section 4.2), N is the amount of concepts, M is the amount of categories, T is the amount of tags (all from the user profile), ConF.IDF(x) is the CF.IDF-value for user concept x, CatF.IDF(y) is the CF.IDF-value for user category y, and TF.IDF(t) is the TF.IDF-value for user tag t (Goossen et al. 2011; Nicoletti et al. 2012). This computation outputs a value in the range of $[0, +\infty]$, which is then normalized to the range $[0, 1]$. A high value represents a good similarity between the user and the document. If the value is close to 0, it means that there are few o none semantic items shared between the two models.

$$\text{TFIDF-based}(U, D) = \sum_{n \in N} \text{ConF.IDF}(con_n) + \sum_{m \in M} \text{CatF.IDF}(cat_m) + \sum_{t \in T} \text{TF.IDF}(tag_t) \tag{15}$$

4.5 NLP Semantic analysis

This analysis aims at extracting concepts and tags from a textual input. The analysis is executed on the raw text from the Wiki pages. This involves two processes: tag mining and concept mining. The sequence of tasks for tag mining is the following:

1. *Text parsing:* The input text from the SAD is parsed in order to remove custom annotations from the Wiki syntax as well as invalid characters.
2. *Sentence detection*: The parsed input text is split into a set of sentences. The OpenNLP[d] implementation was used for this task.
3. *Tokenizer:* The sentences are divided into tokens (terms). The OpenNLP implementation is again used here.
4. S*top-words removal:* Frequently used terms are removed. We use approximately 600 words for this task (a mixture of commonly-used stop-words).
5. *Stemming:* The terms are reduced to their root form to improve the keyword matching. Porter's Stemming algorithm[e] is used here.

The sequence of tasks for concept mining is the following:

1. *Text parsing:* Similar as done in tag mining (above).
2. *Sentence detection:* Similar as done in tag mining (above).
3. *Concept matching:* A set of concepts is associated with each sentence. Since the size of the concept dictionary is relatively small, we process the complete dictionary and try to match concepts with sentences. We apply stop-words removal and stemming (Porter's algorithm) to both concept names and sentence

text alike, aiming at improving the string matching algorithm. In case the match is positive, the concept is associated with the sentence.

4. *Categories matching:* The category hierarchy tree is built for each concept. We associate a set of intermediate level categories to each concept, which is already associated with each sentence, based on our previous work (Nicoletti et al. 2013a). The process is repeated for the upper level categories. In the resulting profile, we register the matching categories and their frequency.

5 Evaluation: methods and results

Our approach was empirically evaluated in two experiments[f] with real users of SADs. The evaluation pursued two main goals. The first goal was to assess the performance of the pipeline in terms of correctness of the recommendations (i.e., the relevance links between users and documents). The second goal was to compare the candidate similarity functions to compute the user-document links and thus determine the functions with the best performance. Additionally, we analyzed the benefits regarding to the effort reductions in stakeholders' tasks, if the people would have been assisted by our recommendation tool. At last, we assessed whether the interest indicators were relevant for inferring stakeholders' interests by means of an Information Gain analysis.

To accomplish the goals above, we asked different groups of users to work with Wiki-based SADs and perform specific architecture-related tasks. In background, we monitored the browsing activity of these users and collected Wiki usage statistics. We also had information about the actual users' interests on the SAD, as they provided us feedback during the experiment. This feedback allowed us to check, *postmortem*, if the SAD documents recommended by our tool could have been useful to these users (note that these users did not receive recommendations while working with the SAD).

In this study, inferring the interests of a user on a given document can be seen as a binary-class classification problem. We used standard Machine Learning metrics such as: precision, recall, and F-measure (Baeza-Yates and Ribeiro-Neto 2011). In our context, precision represents the percentage of recommended documents that were actually relevant to the user. Recall is the fraction of relevant documents that were suggested. F-measure is considered as the harmonic mean (or weighted average) of precision and recall. In particular, we used $F_{0.5}$ *measure* that is a variation of the regular metric that prioritizes precision over recall. We conducted two experiments, each one with a different SAD and separate groups of subjects, in order to analyze whether our approach exhibits similar performance trends with different experimental configurations.

5.1 Experiment #1: Electronic voting system

For this first experiment, we employed 77 test subjects, who were undergraduate and graduate students from a Software Architecture course taught at UNICEN University (Tandil, Argentina). The graduate students were practitioners taking software development courses for their Master degree. We organized the participants into 11 groups: 10 groups of 7 undergraduate students each, plus an additional group of 7 practitioners.

The materials used in the experiment were: a SAD describing an Electronic Voting System according to the V&B template[g], and predefined question sets for evaluating the quality of the architectural documentation[h]. The SAD documents were hosted on DokuWiki, and described the main design decisions and architectural views for the system. The SAD

contained 24 pages (documents) and 22 architectural diagrams, which are representative amounts of real-life SADs. After discarding those SAD pages containing general information, such us indexes, acronyms or definitions, the documentation corpus was reduced to 16 pages. The mapping of Wiki pages to SAD contents was as follows: one Wiki page per architectural view, and one Wiki page per additional section (i.e., documentation beyond views) of the V&B template.

The SAD contents were of acceptable quality, but still had some inconsistencies, omissions, and opportunities for improvement. The question sets were geared to discover these problems from the perspective of different types of stakeholders. A question set is a questionnaire designed with a specific stakeholder role in mind, so as to influence his/her navigation patterns through the Wiki. The questionnaire included: i) a set of quality-attribute scenarios to be evaluated with an ATAM-like design review (Bass et al. 2012), and ii) a set of role-specific questions (also ATAM-like) referring to the quality of the architectural descriptions (Nord et al. 2009). We decided to work with 4 common stakeholder roles, namely: manager, software architect, evaluator, end-user/client; and defined a question set for each role accordingly. Each person was asked to record the elapsed time per item, the difficulty of the task, and the SAD documents that supported his/her responses.

5.1.1 Experimental procedure

The experiment involved five main steps. First, an instance of DokuWiki with the SAD was deployed on a public-access server. This Wiki used a modified version of the software that included monitoring capabilities via PHP/JQuery scripts. Second, the subjects were asked to browse our Wiki for a week, so that they could familiarized with the software architecture of the Electronic Voting System as well as with the V&B templates. Then, we assigned a role to each user within the groups, and distributed the corresponding question sets. Third, the groups were given 3 weeks to go through their question sets and produce an assessment report. After the 3 weeks, we collected the usage statistics logged by the Wiki. Fourth, we generated a matrix with the real user interests on the SAD documents per group. That is, each user-document pair was labeled as relevant or not-relevant (1 or 0). These matrices of real interests were determined from: i) the question sets assigned to the user roles (indirect source), and ii) the answers of the users to those question sets (direct source). In the first source, each question set was designed in such a way it predetermined the types of SAD documents that a user should look at to answer it. Still, users playing the same role but in different groups might read different documents (of the same SAD). In the second source, each user explicitly said in his/her questionnaire what SAD documents he/she looked at. These matrices were actually the references for computing precision, recall, and F-measure (see Sections 5.1.3 and 5.2.2).

The explicit feedback reported by the subjects in their questionnaires also allowed us to analyze the difficulty and time required to solve each questionnaire item, which we called task-difficulty and resolution-time respectively (see Sections 5.1.4 and 5.2.3). Task-difficulty was measured in a categorical scale: low, medium and high. Resolution time was also binned in a categorical scale: low (0-20 minutes), medium (20-60 minutes) and high (more than 60 minutes). For each user-document pair, we recorded the maximum task-difficulty and the maximum resolution-time, which were computed as follows. If a

user indicated that a given item had a difficulty-value D and a resolution-time T, then we labeled the SAD documents associated to that item with a difficulty D and a time T only when: i) the documents had not been labeled before, or ii) the documents had been labeled before with lower values. This procedure was repeated for each item of the questionnaire, for every user's report.

As regards the semantic annotations (inputs of our approach), each Wiki document was annotated with semantic concepts and tags by mutual agreement among the authors (simulating an expert's opinion). These annotations were not visible to the subjects during the experiments. We annotated the roles considering the V&B model (Clements et al. 2003) and the topics involved in the questionnaire for each role. These role annotations constituted the initial stakeholders' profiles, which were later enriched by interest indicators and items coming from the Wiki pages visited by the users.

5.1.2 Relevance of interest indicators

Prior to the evaluation of our approach, we conducted an Information Gain (IG) analysis (Mitchell 1997) in order to assess the importance of interest indicators for inferring relevance labels of documents for each user. This analysis measures the quality of a given independent variable (e.g., an indicator such as the number of visits) for predicting a target dependent variable (e.g., the relevance label). The IG value for an indicator is in the range $[0, 1]$. A value of 0 means that the indicator is irrelevant, whereas a value of 1 means that it is highly relevant.

In our study, we employed the Wiki usage statistics to build a dataset (with $\simeq$ 900 samples) and computed the IG value of each interest indicator. The IG values for this experiment were: $IG_{visits} = 1$ (number of visits), $IG_{scroll\,time} = 0.80$ (scrolling time), $IG_{time} = 0.74$ (time spent reading a SAD document), which show that these 3 indicators are suitable to classify the user interests. Based on these results, we chose the number of visits as the main interest indicator in our NLP pipeline.

5.1.3 Performance of the recommender system (experiment #1)

We ran the NLP pipeline with the following inputs: i) the text contents from the SAD documents, ii) the number of visits for each SAD document, iii) the document annotations provided by experts, and iv) the role annotations for each user (i.e., the basic profiles per stakeholder type). We experimented with several values for the k parameter (see Section 4.3), in order to assess the performance of the recommender system. For each k-value, we firstly computed the measures per user, and then an average for the 77 users. In addition, we tested the 13 candidate similarity functions described in Section 4.4.

Figure 6 summarizes the precision, recall and F-measure in our approach for different configurations (i.e., k-values and similarity functions). Considering that we prioritized precision over recall, the best F-measure value was obtained for a k-value in the range $[0.3, 0.4]$ with very small variations depending on the similarity function being used. . In particular, the best performance was exhibited by the *Tanimoto* function with $k = 0.35$, with an F-measure of 0.66 that corresponded to a precision of 0.67 and a recall of 0.65). However, the maximum precision obtained was 0.7 for $k = 0.25$, but the recall dropped to 0.5 because of the natural trade-off between these metrics.

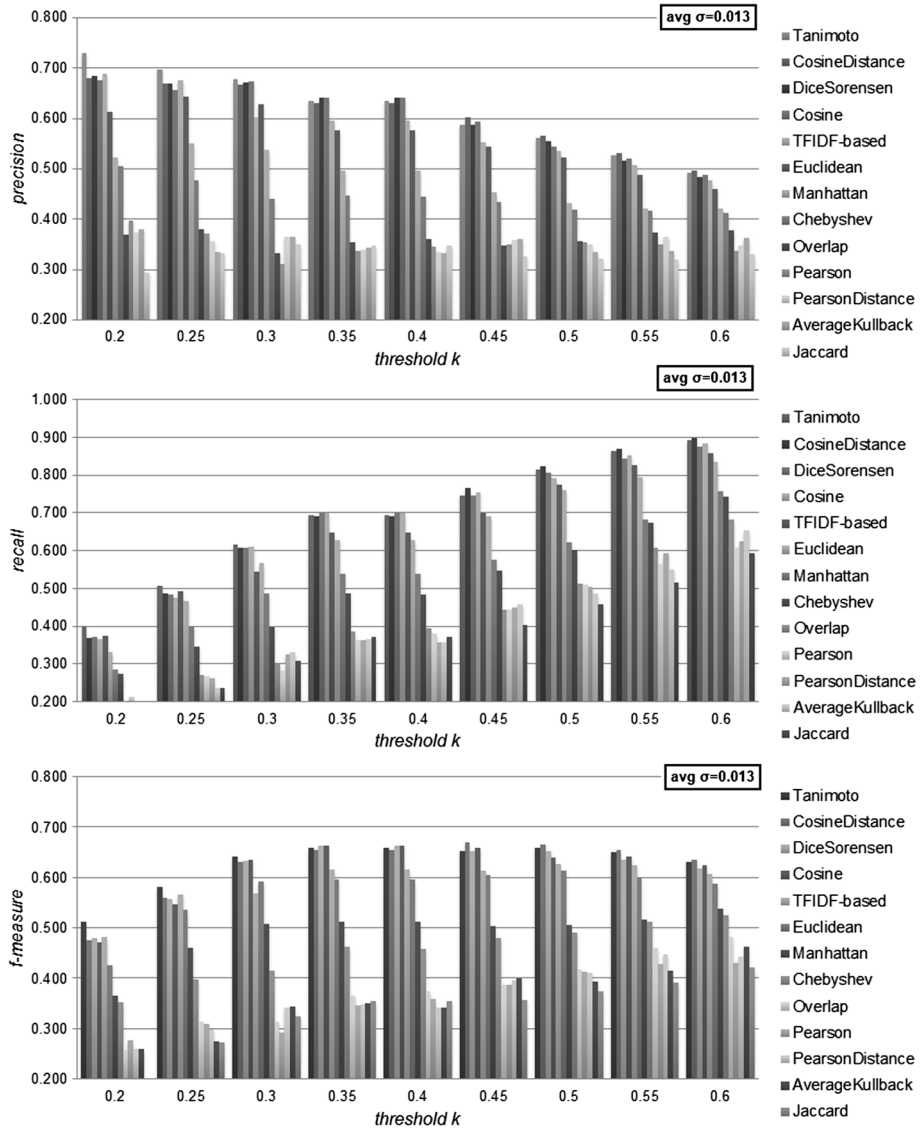

Figure 6 Performance of our approach for different k-values and similarity functions (experiment #1).

We additionally performed a statistical analysis of the similarity functions based on their F-measure values. To this end, we applied both the Student's t-test (two-sided) and the Mann-Whitney-Wilcoxon (MWW) test to the average F-measure obtained for the different 77 subjects. We used the MWW test in those cases in which normality of samples could not been verified with Shapiro-Wilk Normality test. For each pair of similarity functions, we tested the null hypothesis H_0 : *F-measure values of one function tend to be equal to those of the other function*, against the alternative hypothesis H_1 : *F-measure values of one function tend to be higher (or lower) than those of the other function*. For non-normal distributions, we used the notation H_0^* and H_1^*, respectively. In those cases were the Student's test was used (normal distributions), we were able to verify the homoscedasticity of the samples by using the F-test (two-samples) with a significance level of 0.95.

Figure 7 presents the results of this statistical analysis, in which each cell of the tablel represents the accepted hypothesis for each pair of functions with a significance level of 0.05. This table also includes the corresponding p-values for each statistical test. Notice that we can identify a subset of 3 functions (*Tanimoto, Cosine*, *andCosineDistance*) with the highest performance, which are also statistically different from the remaining functions, but similar to each other.

5.1.4 Usefulness of the recommendations (experiment #1)

To analyze the quality of the recommendations, we recorded the maximum of both task-difficulty and resolution-time associated with those SAD documents recommended by our tool. The set of recommended documents for a given user are those ranked in the first places (i.e., high similarity values), and the size of that set is determined by the k value. For example, if the system recommended N documents to a particular user, and they were linked to high-difficulty tasks, we categorized the recommendations as potentially useful. On the other hand, if the N sections were linked to tasks requiring little time to be solved, the recommendations were considered not as useful as in the previous case.

Figure 8 shows a comparative chart between the percentage of potentially useful recommendations (high or medium values) and the less useful ones (low values)regarding both task-difficulty and resolution-time. The results of this chart were computed with the *Cosine* function (one of the best performing functions) for the range of k values [0.3, 0.4]. The chart shows that a high percentage of the recommended documents are generally related to tasks with high or medium difficulty, as well as tasks requiring high or medium time to be solved.

In the case of task-difficulty, those tasks with high or medium difficulty were targeted by an average of 75% of the recommendations, whereas tasks with low difficulty were targeted by the remaining 25% recommendations. In the case of resolution time, those tasks requiring high or medium time (more than 20 minutes) were targeted by an average of 80%, whereas tasks that were quickly solved were targeted by the remaining 20%. Similar percentages were also observed for the other best performing functions, showing trend in the quality of recommendations, which seems independent of the similarity function chosen.

	TAN	**COS**	COSD	DICS	TFIDF	EUC	MAN	CHE	OVE	PEA	PEAD	KUL	JAC
TAN													
COS	**p=0.277 (H_0)**												
COSD	**p=0.362 (H_0)**	**p=0.917 (H_0)**											
DICS	p=0.011 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)										
TFIDF	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1*)									
EUC	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1*)	p=0.489 (H_0)								
MAN	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)							
CHE	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1*)	p≈0 (H_1)	p≈0 (H_1)	p=0.085 (H_0*)						
OVE	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1*)	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1*)	p≈0 (H_1)					
PEA	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p=0.043 (H_1*)				
PEAD	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p≈0 (H_1*)	p=0.005 (H_1*)	p=0.072 (H_0*)			
KUL	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1*)	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1*)	p≈0 (H_1)	p=0.004 (H_1)	p≈0 (H_1*)	p=0.026 (H_1*)		
JAC	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1*)	p≈0 (H_1)	p≈0 (H_1)	p≈0 (H_1*)	p≈0 (H_1)	p=0.005 (H_1)	p≈0 (H_1*)	p=0.028 (H_1*)	p=0.821 (H_0)	

Figure 7 Accepted hypothesis and p-values of statistical tests for each pair of functions.

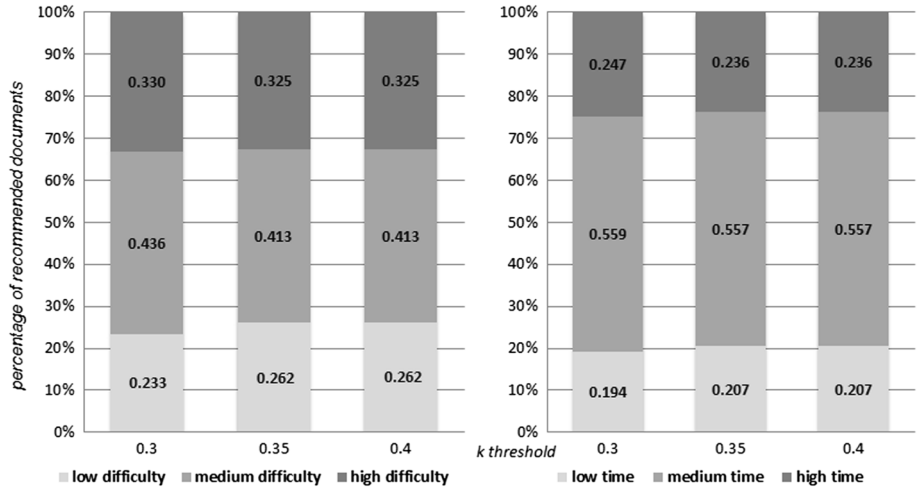

Figure 8 Useful (high + medium) versus less useful (low) recommendations, considering task-difficulty and resolution time (experiment #1).

5.2 Experiment #2: Clemson transit assistance system

This experiment was designed in a similar manner to the first experiment, but had a few differences. The participants were only 10 graduate students (who also were taking a Software Architecture post-graduate course at UNICEN University). As regard the materials, we used a different SAD hosted on DokuWiki, which materialized a solution for a model problem: the Clemson Transit Assistance System (CTAS)[i]. As this SAD was also open to improvements/suggestions, the students were asked to perform an ATAM-based evaluation activity. The matrix of real users' interests was computed similarly as in experiment #1. Despite the differences above, we were careful while designing the experiment #2 so as to make the results of both experiments comparable. According to Figure 2, we changed the following inputs of our approach: the interest indicators (different subjects browsing the Wiki), the SAD textual contents (we used a new SAD), and the semantic annotations. Actually, only those annotations related to problem-specific aspects of the new SAD were modified. The experimental procedure of experiment #2 was executed exactly in the same way as in experiment #1.

5.2.1 Relevance of interest indicators

An Information Gain analysis was performed with the Wiki-usage data from the new group of subjects. In this case, a dataset of $\simeq$ 300 samples was built. In particular, the IG values were the following: $IG_{visits} = 1$, $IG_{time} = 0.87$, $IG_{scroll\,time} = 0.72$. We notice that the same 3 interest indicators that showed the highest scores in experiment #1, were also the highest ones in this second experiment, although with some variations in IG_{time},and $IG_{scroll\,time}$. Like in experiment #1, we selected the number of visits as the main interest indicator for the NLP pipeline.

5.2.2 Performance of the recommender system (experiment #2)

The results of precision, recall and F-measure, derived from the matrix of users' interests are summarized in Figure 9. Similarly to experiment #1, we show a comparison across the candidate similarity functions. In this experiment, the *Cosine* function obtained the best

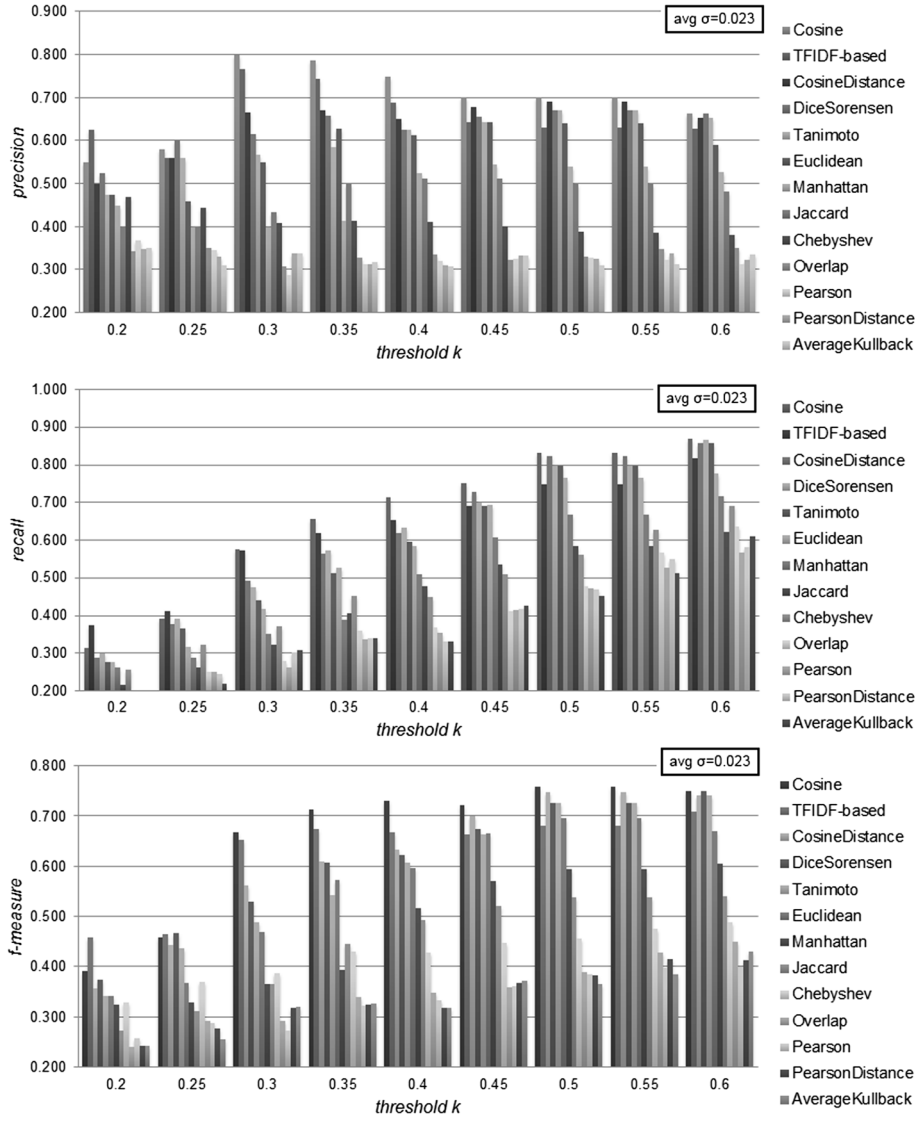

Figure 9 Performance of our approach for different k-values and similarity functions (experiment #2).

performance reaching an F-measure of $[0.73; 0.76]$ for $k = [0.4; 0.6]$, with a precision of $[0.70; 0.75]$ and a recall of $[0.75; 0.83]$. This performance was higher than the one obtained in experiment #1. Another interesting result is that the *TFIDF-based* function performed slightly better than in the previous experiment, with a F-measure of $[0.66; 0.68]$ for $k = [0.3; 0.4]$, reaching a precision of $[0.74; 0.76]$ and a recall of $[0.66; 0.7]$.

When comparing the performance results of both experiments, we observe that, in general, the overall performance of our approach does not significantly differ using a different set of inputs. This situation can be seen in Table 1, which shows the ranking of similarity functions for both experiments, using an average of F-measure (the standard deviation was ≈ 0.03). Although both rankings present several similarities, we may notice that the exact same ranking could not be verified. For instance, *Tanimoto* function moved from the first to the fifth place, whereas the *TFIDF-based* function (proposed by us) moved from the fifth to the second place.

Table 1 Rankings of performance (average F-measure) for the 13 similarity functions

Ranking for experiment #1	*Ranking for experiment #2*
Tanimoto (TAN)	**Cosine (COS)**
CosineDistance (COSD)	**TFIDF-based (TFIDF)**
Cosine (COS)	**CosineDistance (COSD)**
DiceSorensen (DICS)	**DiceSorensen (DICS)**
TFIDF-based (TFIDF)	**Tanimoto (TAN)**
Euclidean (EUC)	**Euclidean (EUC)**
Manhattan (MAN)	Manhattan (MAN)
Chebyshev (CHE)	Jaccard (JAC)
Overlap (OVE)	Chebyshev (CHE)
Pearson (PEA)	Overlap (OVE)
PearsonDistance (PEAD)	PearsonDistance (PEAD)
AverageKullback (KUL)	Pearson (PEA)
Jaccard (JAC)	AverageKullback (KUL)

We identified a subset of 6 out of 13 functions that exhibited twice a noticeably better performance than the rest (these 6 functions are highlighted in the table). This observation might indicate that certain functions are best suited to solve the profile similarity problem in our domain of study. Furthermore, we were able to confirm the statistical results of Section 5.1.3. However, only 2 of the functions with good performance could be verified, namely *Cosine* and *CosineDistance*. The *Tanimoto* function actually showed a considerably lower performance in experiment #2.

5.2.3 Usefulness of the recommendations (experiment #2)

Figure 10 presents results of the quality of recommendations of *Cosine* function for the range of best k values [0.4, 0.6]. We can appreciate that in average 72% of the recommendations targeted high or medium difficulty tasks, and that an average of 73% of recommendations targeted tasks requiring high or medium resolution time. These results are consistent with the ones of experiment #1, although the percentages of good-quality recommendations are slightly lower. However these minor differences might be explained

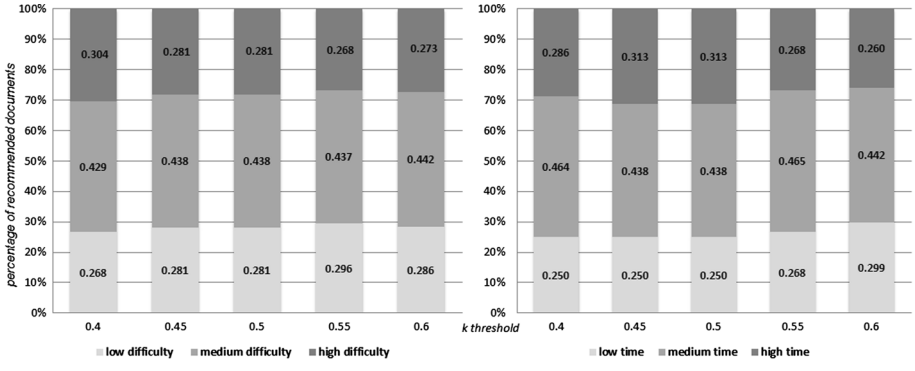

Figure 10 Useful (high + medium) versus less useful (low) recommendations, considering task-difficulty and resolution time (experiment #2).

by the judgment used by test subjects to classify questionnaire items regarding their difficulty. Also, the required time might differ according to the subjects' skills on architecture evaluation.

6 Discussion

From the analysis of the experiments above, we consider that our approach exhibited a good performance. For the Electronic Vote System (experiment #1), the best performances were obtained for values of k in the range $[0.3, 0.4]$ by the *Tanimoto* function with a F-measure of 0.66. The precision was 0.67, meaning that 67% of the recommendable documents were actually relevant to the users. The recall was 0.65, meaning that 65% of the documents actually relevant were effectively classified as relevant.

The second experiment (CTAS architecture) allowed us to verify the trends of the performance values measured in experiment #1. In fact, our pipeline performed even better than in the first experiment. For instance, the *Cosine* function achieved an F-measure of $[0.73; 0.76]$ for $k = [0.4; 0.6]$. We noticed that the highest performance was reached for higher k-values in experiment #2 with respect to the experiment #1. We lately associated this observation with the fact that the test subjects indicated a higher percentage of actually relevant documents in the experiment #2. The criterion for choosing k can not be generalized and depends on problem-specific settings.

For each experiment, we built a ranking that compared the 13 similarity functions based on their average performance. A statistical analysis allowed us to select 2 good-performance functions (*Cosine* and *CosineDistance*) with a confidence of 95%.

The results were also encouraging in terms of its potential for assisting stakeholders to deal with the information overload problem, since the recommended sections (in our experiments) were generally associated to difficult and time-consuming tasks. Anyway, a more rigorous evaluation on both usefulness of recommendations should be addressed in future experiments.

The results of this study, although preliminary, have important implications in the field of Software Architecture Documentation. First, our proposed approach was successfully evaluated with 2 experiments involving SAD users. To the best of our knowledge, there are not other similar approaches that had been empirically evaluated. Second, we proposed 2 alternative similarity functions to the function proposed in our previous work (Nicoletti et al. 2013b). These alternative functions have shown (statistically) higher performances values than the ones observed in previous results.

6.1 Threats to validity

We carefully designed the experiments in order to recreate a real software development scenario. However, there are still some threats to validity that should be considered (Wohlin et al. 2012).

6.1.1 Construct validity

To measure the time spent to solve each questionnaire item we let the subjects inform an approximate value, instead of recording the actual time between assignments. Although the categorical scale is a good approximation of resolution time, there are certainly more precise alternatives. The reason for our decision was that the experiments were not

designed as a "quiet room"[j]. A resolution procedure based on a quiet room would differ from a real scenario, in which stakeholders performs several concurrent tasks. Also, executing the experiments in a quiet room style would have made it impractical to obtain a significant sample of data.

Test subjects were not aware of the experimental objectives. Therefore, we minimized the chance that they had behaved in any particular way that may had biased the results.

6.1.2 Internal validity

Both SADs (e-Vote and CTAS) and the question sets were not part of the actual "working context" (e.g., artifacts, project tasks) of the subjects, so the values reported for task-difficulty and resolution-time might be affected by this condition. We also should note that the evaluation of SADs is not a common case in many architecture-centric developments, although some experiences have been reported as part of ATAM evaluations with checklists (Nord et al. 2009). Furthermore, we argue that the activity of evaluating a SAD is not very different from that of searching through the SAD during normal development activities, so the threat effect is reduced.

Another threat is the possible bias caused by some subjects having prior knowledge of the software architectures (documented by the SADs) or having performed this kind of practical assignment before. We were careful when selecting the subjects to check that they had neither participated in similar activities nor worked with the SADs before.

The Wiki pages browsed by the subjects and the V&B stakeholder types provide an approximation to the user profiles, but they are not the only information sources at work. Inter-personal conversations, questionnaires or interactions with other artifacts/tools (not covered by our approach) can reveal additional information about the interests of a user. We are planning to explore new information sources in future studies.

6.1.3 External validity

We are aware of the fact that an experiment with students in an academic environment might not be the same as an industrial context with seasoned software practitioners. Therefore, we cannot generalize the results of our study to an industrial scenario. To mitigate this threat, we designed our experimental environment to be as realistic as possible. First, the architecture documentation was representative of real-life SADs. Second, the test subjects were not tested in a "quiet room" style (as mentioned above). On the contrary, the subjects were free to solve the questionnaires at the university or at their homes, even working in groups. We believe this latter scenario is the closest one to a real development environment. In addition, most students were actually software practitioners working for local industries with 1-3 years of experience.

6.1.4 Conclusion validity

In contrast with external validity, the use of graduate and undergraduate students instead of industry practitioners allowed us to obtain a good sample (87 subjects) to support the validity of the results. Indeed, it would have been harder to gather a similar number of industry professionals with the time availability required by the experiments. As a downside, the level of heterogeneity of the test groups was low, since most subjects shared similar levels of knowledge and technical background. This situation reduced threats to conclusion validity, but it trades off with external validity.

7 Conclusions and future work

In this work, we have proposed an approach for discovering stakeholders' interests in a Wiki-based SAD that relies on user modeling and NLP techniques. We capture the stakeholders' interests in user profiles, and then look for SAD documents whose contents match those profiles. All the SAD documents are preprocessed in advance by an NLP pipeline. In this context, our tool can recommend specific SAD documents for each stakeholder. The ultimate goal is to improve the stakeholders' access to relevant architectural information, and thus, make the SAD a more effective artifact for communicating architectural concerns. A preliminary evaluation with simulated recommendations has shown the potential of our approach. Also, we empirically identified 2 (out of 13) candidate similarity functions that achieved good performance, with a F-measure value around 0.73, and precision values of 0.7 and recall of 0.75 (depending on the k threshold used). Nonetheless, experiments with other SADs and sets of stakeholders are required in order to validate these claims.

This research opens several lines of future work. In addition to the browsing activity of users, another source for inferring interests are the interactions between users (e.g., chats rooms, instant messaging systems, or voice over IP). In the short term, we will add a chat mechanism to the Wiki infrastructure, so as to monitor user conversations with respect to the SAD and apply our NLP pipeline on these conversations. The mined information will be incorporated to the user profiles.

Managing architectural knowledge in a software project involves both production and consumption of SAD contents. However, as the amount of documentation increases (and also its production costs), its value for the stakeholders tends to decrease. We believe that our approach can help to deal with the production-side of the process, i.e., the ways in which a documenter writes (or updates) SAD documents. Based on the user profiles, the documenter could document "just enough" of the SAD, by prioritizing those documents that maximize the stakeholders' overall satisfaction. In fact, we have recently developed a prototype tool (Diaz-Pace et al. 2013) to assist the documenter in this activity, although yet without user profiles. Finally, as a long-term goal, we want to investigate the pros and cons of personalization techniques when applied to other types of documents within an architecture-centric development process (e.g., technical manuals, requirements specifications, or API documentation).

Endnotes

[a] The terms user and stakeholder are considered synonyms.

[b] The DokuWiki project official Website might be found at: http://www.dokuwiki.org/.

[c] A Wiki-based SAD example provided by the SEI (Pittsburgh, EEUU) might be found at: http://wiki.sei.cmu.edu/sad.

[d] The OpenNLP official Website might be found at: http://opennlp.apache.org/.

[e] The Porter's Stemming algorithm official Website might be found at: http://snowball.tartarus.org/.

[f] We acknowledge that the experiments are in compliance with the Helsinki Declaration and were approved by the Professors responsible for the involved academic courses. The participants were neither negatively affected nor harmed in any way during the execution of the experiments.

[g] Examples of V&B templates might be found at: http://wiki.sei.cmu.edu/sad.

[h] More information about the resources used in our experiments can be found at the following article Website: http://mnicoletti.sites.exa.unicen.edu.ar/jserd2013.

[i] A reference document that describes the CTAS might be found at: http://people.cs.clemson.edu/~johnmc/courses/cpsc875/resources/Telematics.pdf.

[j] A quiet room is a term referring to a place used for experimentation with human subjects in which there are no distractions that may bias the results of tests.

Competing interests
The authors declare that they have no competing interests.

Acknowledgements
This work was partially supported by ANPCyT (Argentina) through PICT Project 2011 No. 0366 and PICT Project 2010 No. 2247, and also by CONICET (Argentina) through PIP Project No. 112-201101-00078. The authors would like to thank to the reviewers for their valuable feedback to improve the quality of this manuscript.

Received: 14 November 2013 Accepted: 13 May 2014
Published online: 21 August 2014

References

Al halabi WS, Kubat M, Tapia M (2007) Time spent on a web page is sufficient to infer a user's interest. In: Proceedings of the IASTED European Conference: Internet and Multimedia Systems and Applications (IMSA IASTED). ACTA Press, Anaheim, CA, USA, pp 41–46

Baeza-Yates R, Ribeiro-Neto B (2011) Modern Information Retrieval: The Concepts and Technology Behind Search. 2nd edn. Addison-Wesley Professional, Boston, USA

Bass L, Clements P, Kazman R (2012) Software Architecture in Practice. 3rd edn. Addison-Wesley Professional, Boston, USA

Castro-Herrera C, Cleland-Huang J, Mobasher B (2009) Enhancing stakeholder profiles to improve recommendations in online requirements elicitation. In: 17th IEEE International Requirements Engineering Conference (RE), Atlanta, USA, pp 37–46

Claypool M, Le P, Wased M, Brown D (2001) Implicit interest indicators In: Proceedings of the 6th International Conference on Intelligent User Interfaces (ICIUI). IUI '01. ACM, New York, NY, USA, pp 33–40

Clements P, Bachmann F, Bass L, Garlan D, Ivers J, Little R, Nord R, Stafford J (2003) A practical method for documenting software architectures. In: Proceedings of the International Conference on Software Engineering (ICSE). Portland, USA

Clements P, Bachmann F, Bass L, Garlan D, Ivers J, Little R, Merson P, Nord R, Stafford J (2010) Documenting Software Architectures: Views and Beyond (2nd Edition). 2nd edn. Addison-Wesley Professional, Boston, USA

de Boer RC, van Vliet H (2008) Architectural knowledge discovery with latent semantic analysis: Constructing a reading guide for software product audits. J Syst Softw 81(9):1456–1469

de Graaf KA, Tang A, Liang P, van Vliet H (2012) Ontology-based software architecture documentation In: Proceedings of Joint Working Conference on Software Architecture & 6th European Conference on Software Architecture (WICSA/ECSA). WICSA 2012. IEEE Computer Society, Helsinki, Finland, pp 315–319

Deza E, Deza M (2006) Dictionary of Distances. North-Holland Elsevier, Amsterdam, Netherlands

Deza MM, Deza E (2009) Encyclopedia of Distances. Springer, New York, USA

Dice LR (1945) Measures of the amount of ecologic association between species. Ecology 26(3):297–302

Diaz-Pace JA, Nicoletti M, Schiaffino S, Villavicencio C, Sanchez L (2013) A stakeholder-centric optimization strategy for architectural documentation. In: Cuzzocrea A, Maabout S. (eds.) Model and Data Engineering. Lecture Notes in Computer Science, Springer, New York, USA, pp 104–117

Farenhorst R, van Vliet H (2008) Experiences with a wiki to support architectural knowledge sharing. In: Proceedings of the 3rd Workshop on Wikis for Software Engineering (Wiki4SE), Porto, Portugal

Goossen F, IJntema W, Frasincar F, Hogenboom F, Kaymak U (2011) News personalization using the cf-idf semantic recommender. In: Proceedings of the International Conference on Web Intelligence, Mining and Semantics (ICWIMS). WIMS '11. ACM, New York, NY, USA, pp 10–11012

Huang A (2008) Similarity measures for text document clustering. In: Proceedings of the 6th New Zealand Computer Science Research Student Conference (NZCSRSC2008). Christchurch, New Zealand, pp 49–56

ISO/IEC/IEEE (2011) ISO/IEC/IEEE 42010: Systems and Software Engineering - Architecture Description. ISO/IEC/IEEE. International Organization for Standardization, number: 42010. http://www.iso-architecture.org/.

Koning H, Vliet HV (2006) Real-life it architecture design reports and their relation to ieee std 1471 stakeholders and concerns. Automated Softw Eng 13:201–223

Jansen A, Avgeriou P, van der Ven JS (2009) Enriching software architecture documentation. J Syst Softw 82(8):1232–1248. SI: Architectural Decisions and Rationale

Liu B (2011) Web Data Mining: Exploring Hyperlinks, Contents and Usage Data. 2nd edn. Data-Centric Systems and Applications. Springer, New York, USA

Mitchell T (1997) Machine Learning, 1st edn. McGraw-Hill Science/Engineering/Math, New York, USA

Mitchell RK, Agle BR, Wood DJ (1997) Toward a theory of stakeholder identification and salience: Defining the principle of who and what really counts. Acad Manag Rev 22:853

Nicoletti M, Diaz-Pace JA, Schiaffino S (2012) Towards software architecture documents matching stakeholders interests. In: Cipolla-Ficarra F, Veltman K, Verber D, Cipolla-Ficarra M, Kammuller F (eds.) Advances in New Technologies, Interactive Interfaces and Communicability. Lecture Notes in Computer Science. Springer, New York, USA, pp 176–185

Nicoletti M, Schiaffino S, Godoy D (2013a) Mining interests for user profiling in electronic conversations. Expert Syst Appl 40(2):638–645

Nicoletti M, Diaz-Pace JA, Schiaffino S (2013b) Discovering stakeholders' interests in wiki-based architectural documentation. In: Diego Vallespir MdOB (ed.) Proceedings of CIbSE 2013 (former IDEAS). XVI Ibero-American

Conference on Software Engineering, Montevideo, Uruguay. Universidad ORT Uruguay, Universidad de la Republica, Antel, pp 5–18

Nord RL, Clements PC, Emery DE, Hilliard R (2009) Reviewing architecture documents using question sets In: Proceedings of Joint Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture (WICSA/ECSA). IEEE, Cambridge, UK, pp 325–328

Rajaraman A, Ullman JD (2012) Mining of Massive Datasets. Cambridge University Press, Cambridge

Schiaffino S, Amandi A (2009) Intelligent user profiling. In: Bramer M (ed.) Artificial Intelligence: An International Perspective. Lecture Notes in Computer Science. Springer, New York, USA, pp 193–216

Su MT (2010) Capturing exploration to improve software architecture documentation. In: Proceedings of the 4th European Conference on Software Architecture (ECSA). ECSA '10. ACM, New York, NY, USA, pp 17–21

Su MT, Hosking J, Grundy J (2011) Capturing architecture documentation navigation trails for content chunking and sharing. In: 2011 9th Working IEEE/IFIP Conference on Software Architecture (WICSA), Boulder, USA, pp 256–259

Sørensen T (1948) A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. Biol Skr 5:1–34

Unphon H, Dittrich Y (2010) Software architecture awareness in long-term software product evolution. J Syst Softw 83(11):2211–2226

Wohlin C, Runeson P, Höst M, Ohlsson M, Regnell B (2012) Experimentation in Software Engineering, Vol. 978-3-642-29043-5. Springer, New York, USA

doi:10.1186/s40411-014-0009-3
Cite this article as: Nicoletti *et al.*: **Personalized architectural documentation based on stakeholders' information needs.** *Journal of Software Engineering Research and Development* 2014 **2**:9.