Aalborg Universitet

AALBORG UNIVERSITY

DENMARK

Personalization Component for KiWi Framework

Durao, Frederico; Xu, Guandong; Kotowski, Jakub; Dolog, Peter

D4.4

Personalization Component for KiWi Framework

Project title:	**Knowledge in a Wiki**
Project acronym:	KIWI
Project number:	ICT-2007.4.2-211932
Project instrument:	EU FP7 Small or Medium-scale Focused Research Projects (STREP)
Project thematic priority:	Information and Communication Technologies (ICT)
Document type:	D (deliverable)
Nature of document:	R/P (report and prototype)
Dissemination level:	PU (public)
Document number:	ICT211932/AAU/D4.4/D/PU/b1
Responsible editors:	Peter Dolog
Authors:	Frederico Durao[1], Guandong Xu[1], Peter Dolog[1] and Jakub Kotowski[2]
Reviewers:	Thomas Schandl (Semantic Web Company) and Tobias Burger (SRFG)
Contributing participants:	AAU, LMU-Munich
Contributing workpackages:	WP4
Contractual delivery:	30 August 2010
Actual delivery:	30 August 2010

Abstract

In this report we present the personalization component implemented for KiWi addressing diverse aspects such as motivation scenarios, configuration management, formal models, implementations and some real case scenarios on how personalization can be applied to support software project management. The personalization component is responsible for addressing and delivering information to individuals regarding their roles and preferences, explicitly or implicitly inferred from their activities in the system. Personalization is a essential for KiWi since this in a multi-user environment and individuals has different interests and needs when using the system.

Keyword List

personalization, user model, social software, customization, semantic web

www.kiwi-project.eu

Personalization Component for KiWi Framework

Frederico Durao[1], Guandong Xu[1], Peter Dolog[1] and Jakub Kotowski[2]

[1] Department of Computer Science, Aalborg University
Email: `fred,dolog,xu@cs.aau.dk`

[2] Institute for Informatics, University of Munich
Email: `Jakub.Kotowski@ifi.lmu.de`

30 August 2010

Abstract

In this report we present the personalization component implemented for KiWi addressing diverse aspects such as motivation scenarios, configuration management, formal models, implementations and some real case scenarios on how personalization can be applied to support software project management. The personalization component is responsible for addressing and delivering information to individuals regarding their roles and preferences, explicitly or implicitly inferred from their activities in the system. Personalization is a essential for KiWi since this in a multi-user environment and individuals has different interests and needs when using the system.

Keyword List

personalization, user model, social software, customization, semantic web

Contents

1 Introduction

In essence, to personalize social systems implies taking decision based on what is the most useful for a particular user or a community of collaborating users in specific situations. In this work we are going to present the personalization component implemented for KiWi addressing diverse aspects such as motivation scenarios, configuration management, formal models, implementations and some real case scenarios on how personalization can be applied to support software project management. The personalization component is responsible for addressing and delivering information to individuals regarding their roles and preferences, explicitly or implicitly inferred from their activities in the system. The result of personalization can be seen as a decision on variable knowledge, content and annotation according to characteristics relevant to users or his involvement in actions as well as different groups. Personalization can be employed at various levels such as content composition or restriction, query rewriting, link recommendation and restriction and so on. Personalization is an essential component for KiWi since it is a multi-user environment and individuals have different needs when using the system. The content of this deliverable is organized as follows:

- Section 1 introduces personalization concepts and the common personalization model that bases the provided algorithm.

- Section 2 presents the recommendation methods implemented for KiWi.

- Section 3 presents the personalized search implemented in KiWi.

- Section 4 introduces the concept of social capital and the model for measuring the social capital from WiKi pages as a factor to enhance the performance of tag-based recommendations.

- Section 5 presents how users in KiWi can customize the information they want to visualize at the interface, and presents a clustering mechanism for better organizing the recommendations in a category presentation way.

- Section 6 addresses a case study on how personalization supports software project development.

- Section 7 presents the configuration and installation requirements for using in KiWi.

- Section 8 concludes the work and points out future works.

To ease the reading and overall understanding, the reader is welcomed to read also other KIWI deliverables such as the revision of state of the art in personalization at [8] and on the specification and extension of the concept and model document in conformance with [5].

1.1 Personalization in KiWi

Most of personalization in KiWi is conceived by observing user's **tagging** activity aiming at building tag-based user models and thereby delivering the appropriate information to right users. KiWi like popular social bookmarking systems such as *del.ici.ous* allows users to express their preference by tagging items of interests. The *collaborative* tagging activity produces a considerable amount of user generated meta-data, representing potential means for personalization. We leverage user tagging information to *personalize recommendations*, *group users* and

personalize search. In addition to the user tagging activity, we analyze the *social activity* within KiWi pages in order to evaluate which pages are preferable to be recommended over the others. In other words, we look at the *social capital value* derived from collaborative work to measure the importance of pages to the whole community.

We also provide facilities for KiWi to *customize information* in KiWi according to their personal preferences. KiWi users can explicitly choose among the many different sorts of recommendations available to be displayed, allocate widgets and organize visual information when logged in the system.

1.2 General Tag-Based Personalization Model

As previously said most of personalization in KiWi is based on user's tagging activity. We see tags as potential source for learning user's interests. Our general personalization model (based on [9]) realizes tagging systems as hyper graphs where the set of verticals is partitioned into sets:

$U = \{u_1, ..., u_k\}, R = \{r_1, ..., r_m\}$, and $T = \{t_1, ..., t_n\}$, where U, R, and T correspond to users, resources, and tags. A tag annotation, i.e. a resource tagged by a user, is an element of set Y, where: $Y \subseteq U \times R \times T$. The final hyper graph formed by a tagging system is defined as G with: $G = \langle V, E \rangle$ with vertices $V = U \cup R \cup T$, and edges $E = \{\{u, r, t\} \mid (u, r, t) \in Y\}$. Particularly to understand the interests of a single user, our models concentrate on the tags and resources that are associated with this particular user, i.e. in a *personal* part of the hyper graph G. We then define the set of interests of a user as $P = (T_u, R_u, Y_u)$, where Y_u is the set of tag annotations of the user: $Y_u = \{(t, r) \mid (u, t, r) \in Y\}$, T_u is the tag set of the user: $T_u = \{t \mid (t, r) \in Y_u\}$, R_u is the set of resources: $R_u = \{r \mid (t, r) \in Y_u\}$.

These tagging notations will serve as a basis for describing our personalized model in the following sections.

1.3 Quality Factors

On top of the user tagging model, we employ quality factors to privilege tags that are popularly used and best represent the user's preferences. The quality factors will build a more precise ranking and increasing the quality personalization model as a whole. The quality factors are:

- *Tag Popularity – pop(t)*. It indicates how often a tag $t \in T$ is being assigned to the pages in the system. We calculate $pop(t)$ as $\frac{n_t}{|R|}$, where n_t is the amount of occurrences of a tag $t \in T$ and $|R|$ is the amount of existing resources in the system.

- *Tag Representativeness – rep(t, r)*. It indicates how important a tag $t \in T$ is to a resource $r \in R$ based on the amount of occurrences of t as a term (or word) in r. For calculating $rep(t)$, we utilize the *term frequency* metric [1], with the belief that the tags that most appear as *term* in a page can better describe it. Note that this factor is limited to textual documents.

- *Affinity between User and Tag – aut(u, t)*. It measures how important is a tag $t \in T_u$ for a user $u \in U$. We calculate $aut(u, t)$ as $\frac{n_t}{|T_u|}$, where n_t is the amount of occurrences of a tag $t \in T_u$ and $|T_u|$ is the amount of tags of a user $u \in U$.

The quality factors are combined in the $QualityFactor(r, u)$ function defined as:

$$\text{QuanlityFactor(r,u)} = \sum_{|t \in T_r|} pop(t) \times rep(t,p) \times aut(t,u),$$

where $|t \in T_r|$ is the amount of tags assigned to a page $r \in R$. This generic personalization model is applied to most of all implemented features plus some minor variants such as multifactor, group and social capital recommendation and personalized search.

1.4 Personalization Services

The personalization functionalities are implemented as services so that can be accessed from other internal components in KiWi. For instance, all recommendations are placed at in the *RecommendationService.java* interface. It defines how each recommendation must be called including the required parameters and a description for each method available. In that way, any other component willing to access one of our personalized recommendations just need to invoke this service. For example, group services could utilize recommendations from the *RecommendationService.java* interface. Figure 1 shows multi factor recommendations being consumed by a group service method. As seen, the class *GroupServiceImpl.java* has a method:

```
List<ContentItem> recommendItemToGroup (ContentItem contentItem);
```

that asks for a multi factor recommendation from the *RecommendationService.java* interface. Note that the recommendation interface must be injected as a private attribute in the class body as follows:

```
private RecommendationService recommendationService;
```

Any other personalization services aimed to be consumed must be accessed in the same way as shown in this example.

1.5 Recommendation Web Services

Some personalization services such as recommendations are also available for external access. We published some recommendation methods as REST Web Services, to which external applications only need to make a HTTP request to have desired recommendations performed. In general, the Recommendation REST Web Services call returns a JSON object representing a list of recommendations (RecommendationJSON). The web service is called using the basic URL path:

```
http://KiWi/seam/resource/service/widgets/recommendations
```

The path however must be completed with a specific recommendation method:

- */personal* - returns personalized recommendations for a given user taking into account similarity of user tags and tags assigned to pages in the system.

- */standard* - returns related recommendations for a given page taking into account similarity of its tags and tags assigned to other pages in the system.

3

Figure 1: Multi factor Recommendations being consumed by a group service method.

- $/multifactor$ - returns personalized recommendations for a given user taking into account similarity of user tags and tags assigned to pages in the system. Further, this is a score weighed by the different quality factors.

  All RESTful services must provide as an input parameter the user identification so that the recommendations can be personalized. This information must be enclosed to the URL as a suffix such as $/user/id$.

2   The Recommendation Methods

This section describes various recommendations approaches that partially or completely fulfill the specification and extension of concept and model for personalization [5]. In Kiwi, tag-based recommendations play a very important role by revealing additional content in addition to the currently viewed page. In addition to that, the recommendations support users in making a choice among a large number of possible alternatives once they are ranked according to the degree of similarity with the currently visited page. Further, social recommendations arise as another means of communication between users in KiWi where they can exchange information and share knowledge.

2.1   Standard Tag-based Recommendation

In this approach, all pages that share tags with the currently viewed page are recommended. In this standard approach no further similarity processing is carried out therefore the list of recommendation is not ranked. The advantage of this approach is the performance since the recommendations relies simply on a data retrieval task. On the other hand, a single tag shared by pages may not be sufficient enough to determine which page is the most similar one to the viewed page. Subfigure 2a shows the standard recommendations in KiWi followed by their respective authors.
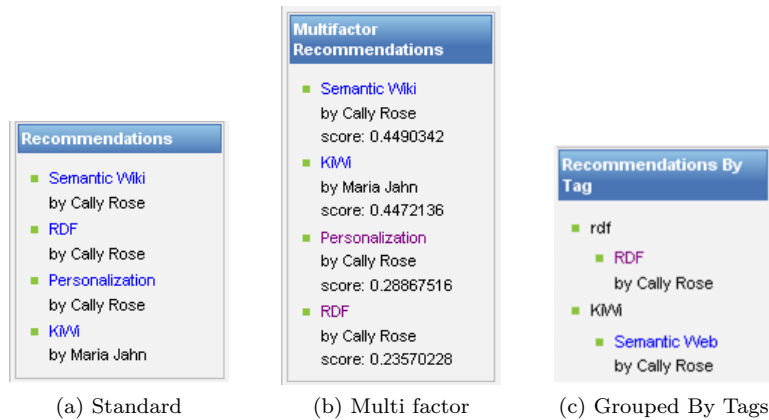
4

(a) Standard   (b) Multi factor   (c) Grouped By Tags

Figure 2: Recommendation Methods

How to use it: Minimally, two users must exist in the system and have assigned tags to two distinct pages with at least one tag in common. Important: users will only receive recommendations of pages they don't have authored or tagged before. In order to receive recommendations, one of the users must be logged in and access the page he knows. The recommendations will be displayed on the left side of the currently viewed page.

2.2 Multi-factor Recommendation

The multi-factor recommendation approach computes the similarity between pages considering multiple factors. Besides cosine similarity, we consider tag popularity, tag representativeness and the affinity between user and tag (see quality factors at section 1.3) [3, 7]. The immediate difference from previous standard approach is the ranking. Multi-factor recommendations are ranked by a similarity score considering the quality factors. The most related recommendations should be at the top of the list, which is derived from the calculated combined similarity scores. Sub figure 2b shows the same recommendations as in sub figure 2a however sorted differently by the multi-factor recommendation model. Pages ranked higher are more likely to satisfy the user's tastes and closer to the content discussed in the currently viewed page.

Comparing the performance of this approach with the previous standard, we noticed the multi-factor approach is worse, however the ranked list provides a more precise selection. Although the recommendations are more effective than the standard approach, its applicability indeed depends on the further performance analysis in different KiWi scenarios. A evaluation of performance of these recommendation mechanisms can be seen at [4].

How to use it: Minimally, two users must exist in the system and have assigned tags to two distinct pages with at least one tag in common. It is recommended that each user have annotated 10 to 15 tags along the pages and some tags are repeated. The pages must have sufficient content (about 50 lines of text) and the tags must appear in the content as ordinary terms. Important: users will only receive recommendations of pages they don't have authored or tagged before. In order to receive recommendations, one of the users must be logged in and access the page he had tagged or authored. The recommendations will be displayed on the left side of the currently viewed page ranked by the quality factors.

5

Figure 3: User "test" recommending page entitled "Start Page" to user "Lala".

2.3 Recommendations Grouped by Tags

In this approach, the recommendations are grouped by the tags which are assigned to the currently viewed page. Similarly to the standard approach, no further similarity processing is undertaken and the list of recommendation is not ranked. On the other hand, the user can go directly to the recommended wiki page just following the tag he/she is interested. The tag-based distribution explicitly provides a justification why the recommendations were generated and assists users to find related specific wiki pages. The disadvantage however is the possibility of existing duplicated recommendations since two different pages can share two distinct tags as well. Sub figure 2c shows two tags *RDF* and *KiWi* with their respective linking recommendations.

How to use it: Minimally, two users must exist in the system and have assigned tags to two distinct pages with at least one tag in common. Important: users will only receive recommendations of pages they don't have authored or tagged before. In order to receive recommendations, one of the users must be logged in and access the page he knows. The recommendations will be displayed on the left side of the currently viewed page and grouped by tags assigned to this page.

2.4 Content Recommendations Sent by User to Friends

Any page displayed in KiWi can be recommended by the currently logged user to his friends. The recommended items will be displayed at the social recommendation panel in the Dashboard. The ones in the group who receive recommendations can either visit the recommended page and/or delete them. Figure 3 shows the simplicity of the recommendation process.

How to use it: Minimally, two users and one page must exist in the system. These users must be friends, which is defined in friend recommendation process. Tags are not required. In order to recommend a page, one user must sign in, select the other user and click on the "Recommend" button placed in the head of the page. The receiver must be logged in KiWi to see the recommended items in the recommendations panel at the Dashboard.

2.5 Content Recommendations Sent by User to Groups

Any page visualized in KiWi can be recommended by the currently logged user to his groups. The difference from recommendation to friends is that all persons who are member of a group will receive this recommendation. The recommended items will be displayed at the group recommendation panel in the Dashboard. Those one who receive recommendations can either visit the recommended page and/or delete them.

How to use it: Minimally, two users, one page and one group must exist in the system. The users must be part of the same group. Tags are not required. In order to recommend a page to group, one of the users must sign in, browse the page to be recommended, select the preferred
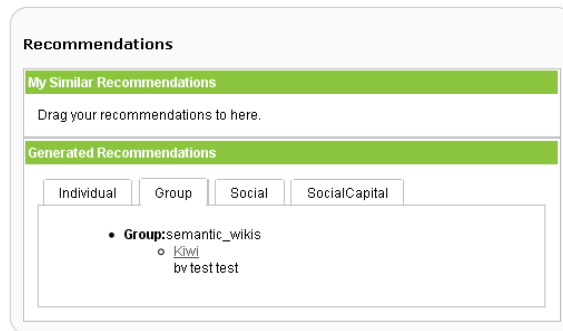
Figure 4: Page entitled "KiWi" recommended to user group "Semantic_Wikis", where user "Lala" is a member of.

group and click on the "Recommend" button placed in the head of the page. The remaining group users must be logged in KiWi to see the recommended item in the recommendations panel at the Dashboard, more precisely at the group tab.

2.6   Group Recommendation

In KiWi, users with similar roles or interest can be organized in groups. Groups can be tagged so that this information provides additional information about the group purpose. We utilize this tagging information to find related pages that might be interesting for a group. We then generate recommendations for groups by measuring the similarity of tags assigned to groups against tags assigned to existing pages in the system. Figure 4 shows the group recommendations being generated at the recommendation panel in the Dashboard.

How to use it: Minimally, one user, one group and one page must exist in the system. This user must be a member of this group. The page and the group must share some tags. In order to visualize the recommendation to the group, the user must sign and browse the group tab of recommendation panel at the Dashboard.

2.7   Friendship (contact) Recommendations

Usually users in KiWi make new friendships just by explicitly choosing their friends among all active users in the system. In order to help users be connected with users who share similar interest, we recommend potential friends to be part of the user's contact list. The recommendations take into account the similarity of tags between the currently logged user and other users in the system. Once the friendship suggestion is processed, the logged KiWi user just needs to add the new friend to his contact list if he agrees. Figure 5 shows a contact recommendation for the currently logged user "Test".

How to use it: Minimally, two users and one page must exist in the system. These two users must not be friends before. They must tag the same page with some tags in common. In order to visualize the friendship recommendation, one user must sign in and visit the contact page at the Dashboard. The friendships suggested will be listed there. As future work, the friendship recommendation would benefit of an explanation for the user, i.e. this person has been recommended to you because you have many social ties in common.

7
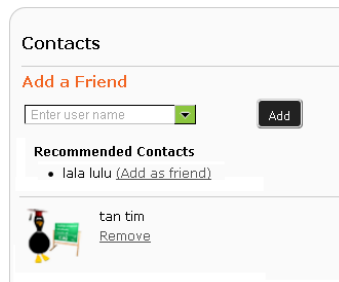
Figure 5: Potential user "Lala" recommended to be a friend of the currently logged user "Test". User "Tan" already belongs to "Test"'s contact list.
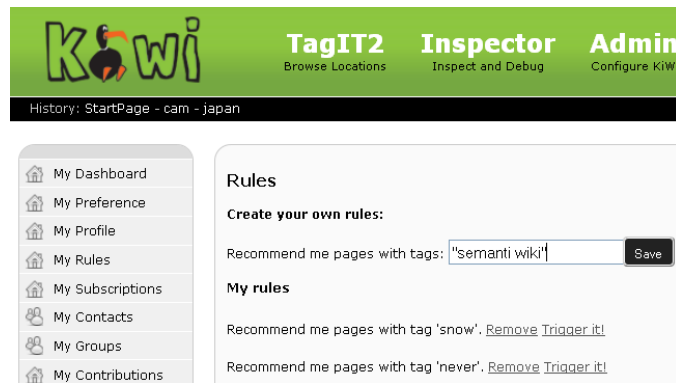
Figure 6: Creation of rules through user interface

2.8   Rule-Based Recommendations

Rule-based recommendation is an outcome of integration between personalization and reasoner components in KiWi. Through rule-based recommendations users can explicitly customize recommendations they want to receive. Users tell the system explicitly what they would like to see. The system then generates rules according to this specification and recommendations are subsequently generated whenever conditions of these rules are satisfied. Because rule-based recommendations use the reasoner, they also transparently benefit from the reason-maintenance and explanation components. Reason maintenance updates facts derived by rules according to changes in the triple store. This means that rule-based recommendations are generated and removed according to the current state of the triple store. The explanation component provides explanations for derived facts. Rule-based recommendations are derived by the reasoner and therefore they can be explained using the explanation component.

How to use it: At the Dashboard, we provide a user interface which allows users to easily specify bodies of their rules; users do not have to know the rule language, in fact they are not aware that they are creating rules. Once a body is specified, the system creates the respective rule which is then listed in the "My rules" panel. If a user is not satisfied with a rule he or she can remove it and create another one. Important: once created, rules still need to be enabled. Figure 6 shows a rule being created "Recommend me pages with tags 'semantic wiki'".

8

2.8.1 Reasoner Task

The reasoner currently used by KiWi is a forward chaining rule based reasoner. It means that the reasoner derives entailed facts as soon as new facts are entered – i.e. it materializes derived facts. Querying then amounts to a simple retrieval of certain facts which are already materialized in the knowledge base. In contrast, a backward chaining reasoner derives entailed facts at query time and therefore querying may be slower with a backward chaining reasoner (while updating the knowledge base is faster than with a forward chaining reasoner).

The reasoner uses the enabled personalization rules and materializes all facts that they entail. The derived facts are then retrieved by SPARQL queries and shown at the recommendation panel in the Dashboard. Note, that this would not be possible using only SPARQL without a reasoner because SPARQL queries only explicit, materialized facts; it does not query entailed facts.

Changes to the knowledge base trigger reasoner and reason maintenance to update materialized facts accordingly; i.e. materialized facts which are not entailed anymore are removed and newly derivable facts are materialized. For the personalization rules it means that if a condition of a personalized rule is not satisfied (i.e. a tag 'semantic wiki' is removed), the inferred facts representing recommendations are removed by reason maintenance and therefore these recommendations disappear from the recommendation panel.

The current KiWi reasoner was built as a prototype to experiment with various forward chaining and reason maintenance techniques for rule languages with different interesting features (e.g. value invention), therefore it has limitations in terms of practical use. For example no form of negation is currently supported except for constraint rules. Also aggregation is not supported and there is only limited support for changing rules in the running system. Rules can be added but removal of a rule is not implemented yet. Implementation of these features is technically feasible but would require significant additional effort. Note: In the current stage, only rules like "Recommend me pages with tags x,y" are implemented. However it is planned as a future work to provide a number of template rules so that individuals are assisted to build their rules by simply filling out a few entry points.

3 The Personalized Search: A Multi-Factor Tag-based Approach

Normally the same query for different users provides the same results. With the personalized search we aim at optimizing the search results to reflect the user's real intention when running a query and reduce the amount of unwanted information. To achieve personalization, we learn a user's preference automatically based on his tagging activity. We utilize this information to re-rank the original search result by calculating the similarity of the set of a user's tags and the tags assigned to the retrieved items. The set of user's tag is composed by different factors such as his own tags; tags assigned to his favorite or most visited pages.

3.1 User Preference Representation

The user preference representation is slightly different from the tag-based user model introduced previously. Instead of relying on a single indicator (i.e. the tagging activity), we now represent the user's preference as a multi-factor user model. For example, we could learn users' preferences

based on the most frequent queries a user perform and also from the tags assigned to his most visited pages. In this way, tags from different factors serve as our learn units of user preferences. Although considering various factors, it is normal that some tags are preferred over another, meaning that the frequency of usage of a given tag can denote its affinity with the tagger. In this sense, we define the tag preference set, for each factor, as a tuple $(tag, tagFreq(tag))$, where $tagFreq(tag)$ is a function that measures the user's degree of interest in that tag. Formally, we define this set for a particular user u and factor $f \in F$ (let F be the set of all possible factors) as:

$$T_f = \{(t, tagFreq(t)) \mid t \in T_f\},$$

where $tagFreq(t) = \frac{n_t}{|T_f|}$, n_t is the number of occurrences of the tag $t \in T_f$ and $|T_f|$ is the amount of tags in a given factor f. The set T_f is normalized such that $\sum_{i=1}^{|T_f|} tagFreq(i) = 1$. To illustrate the user tagging preference representation, suppose a user has only two tags in one particular factor: "semantic web" and "data mining", and the first has been utilized three times while the second has been utilized only once. This means the user has been interested in "semantic web" three times as much as he has been interested in "data mining". Then, the tag preference set of the user for that factor will be represented as {("semantic web",0.75),("data mining",0.25)}.

The composition of our multi-factor tag-based user model T'_u extends the traditional set of user tags T_u (as defined in section 1.2), with a disjoint union of tag sets:

$$T'_u = \bigsqcup_{f \in F} T_f,$$

where T_u is one of the T_f, i.e. $T_u \subseteq T_f$ and T_f is the set of tags assigned to each factor $f \in F$. Next section explains how the tag-based multi-factor approach is applied to personalize search results.

3.2 Tag-based Personalization Approach

The tag-based personalization approach decides which resource $r \in R$ is relevant to each user $u \in U$ based on his preferences established in the multi-factor tag-based user model. Indeed we re-rank the search results by measuring the similarity of tags that denote user preference and the tags assigned to the retrieved items. With this, we promote the items closer to user's preferences to the first positions in the collection of search results. Technically, we calculate the cosine similarity [1] between *each* vector of tag frequencies $\overrightarrow{T}_f \subset \overrightarrow{T}'_u$ from $T_f \subset T'_u$ and the vector of tag frequencies $\overrightarrow{T}_r$ from the retrieved resources T_r of a given user query q. Further, we weigh each vector $\overrightarrow{T}_f$ with a coefficient, α_f that determines its degree of importance over others factors in the model. We incorporate the coefficients because different users may rely on tag factors differently. As a consequence, the importance of each factor may vary accordingly. The coefficient values for each factor are automatically estimated using the Ordinary Least Square (OLS) linear regression. The goal of using OLS is to minimize the sum of squared distances between the observed tag frequencies of each factor in the dataset and the best coefficient values predicted by linear approximation [13].

Once the coefficient values are estimated, the Tag-based Similarity Score (TSS) can be calculated as:
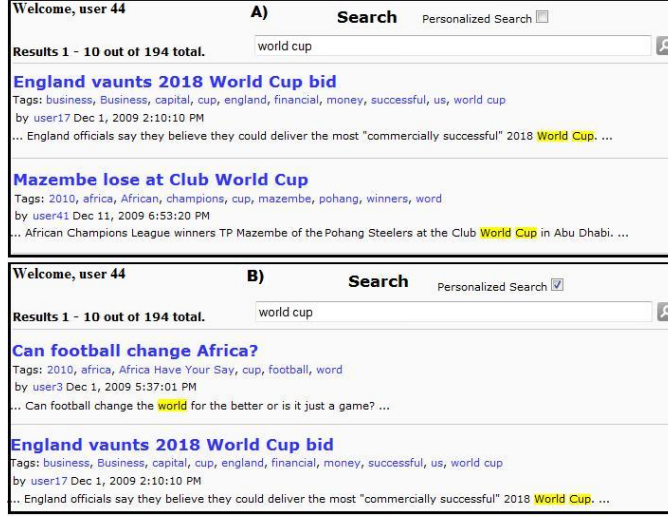
Figure 7: Excerpt of effects of the tag-based personalized search.

$$TSS(\overrightarrow{T}_r, \overrightarrow{T}'_u) = \sum_{f=1}^{|F|} \alpha_f * \frac{\overrightarrow{T_r} \cdot \overrightarrow{T_f}}{|\overrightarrow{T_r}| \times |\overrightarrow{T_f}|}$$

The TSS value is then utilized to weigh the ordinary search score and thereby promote the items matching the user interests to higher positions in the search ranking. The personalized search score (persScore) over a resource r triggered by a query q in the set of resources R is then defined as follows:

$$\text{Personalized Search Score}(q, r, u) = \sum_{t \in q} \text{tf-idf}_{t,r} * TSS(\overrightarrow{T}_r, \overrightarrow{T}'_u).$$

At the user interface, the personalized search score eventually re-ranks the search results that are browsed.

Figure 7 illustrates the effects of the tag-based personalized search for the logged user *user44*. Scenario a) shows the result set of a non-personalized search for the query "world cup" whereas scenario b) shows the result set of the tag-based personalized search for the same query. Each item, in both scenarios, is followed by its respective tags and additional metadata. Note that the search results are differently browsed when the personalized search is activated in scenario b). In particular, the retrieved item *England vaunts 2018 World Cup bid* is surpassed by *Can football change Africa?* whereas the item *Mazembe lose at Club World Cup* looses the second position in the ranking. This happens because user *user44* preference is represented by tags "world cup", "africa", "2010". The second item although is related to world cup, it concerns with the event taking place in 2018.

How to use it: Minimally, one user must exist in the system and have assigned tags to two distinct pages. The content of these two pages must contain share some terms to be retrieved in the same query. The index of terms must be processed so that the search returns results. The
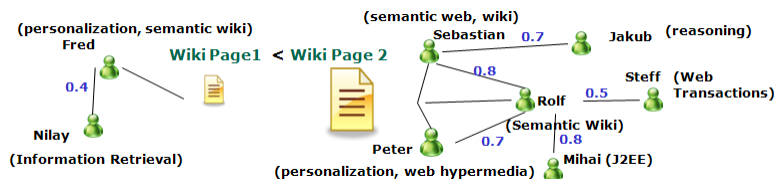
Figure 8: Social capital behind of Wiki pages

user must have one or many tags in one of pages. The other page must have none or very few tags assigned by this user. In order to realize the personalize search, the user may be logged in the system and go to the search area in KiWi. The query input must be one of the shared terms common to both pages. In order to realize the changes in the ranking, the searcher only needs to tick the "personalized search button".

4   Using Social Capital from Collaborative Work for Personalization

The amount of data produced through collaborative activities such as commenting, rating, tagging, editing, determine its social capital — *the ability of social networks to transfer information to share/solve issues for a common benefit*. Due to the simplicity of usage, KiWi becomes a perfect means for the development of *social capital* so that individuals contribute to the overall content building. Based on these premises and results obtained from the analysis of social and behavioral aspects of tag-based recommender system [6], we believe that recommendations of content items can be ranked according to the social capital value they carry. In order to evaluate the *social capital*, we analyze the user expertise and user participation when building a Wiki page. The user expertise and participation looks at the production and consumption of an individual within the community besides his activities such as editing, tagging, rating, commenting, etc. We consider the social capital as another factor in addition to the ones discussed in section 3.

As a motivation scenario, Figure 8 presents two Wiki pages **Page 1** and **Page 2** followed by their authors. It also shows the social ties of the authors besides the user skills and degrees of interactivity (ranging from 0 to 1) between them. Although both pages are eligible to be recommended based on a given criteria (details omitted for now), one is likely more preferable over the other because the social capital it carries. The social capital at the end is expected to bring more constructive and useful information to the readers. According to our model, **Page 2** is preferred for recommendation over **Page 1** due to the higher social capital. The proposed model for measuring the social capital on Wiki pages takes into account the amount of contributors of **Page 2** (*Sebastian*, *Peter* and *Rolf*) over **Page 1** (only *Fred*). The model takes into account the sum of collaborative work on **Page 2**, which is enriched by diverse expertise inherited from direct and indirect ties (users *Jakub*, *Steff* and *Mihai*) and high interactivity within the social network. We believe that the higher is the participation to develop a page, the worthier is the content of the final product since it increases considerably the chances of having discussions among the authors. Although this fact does not assure the quality of the content addressed, we understand that the collaborative work forces a mutual agreement and a convergence of different opinions about the content of the page. In this sense, we intend

to recommend pages with the highest social capital given by the collaborative work of their authors.

4.1 The collaborative work

In the context of KiWi, we decompose the collaborative *work* into a set of activities including *editing*, *rating*, *tagging* and *commenting* that are realized in a shared space that enables multi-user editing of Web pages. The model does not limit the addition of others collaborative activities though. A formal description of the collaborative work W is defined by the tuple $W = \{U, R, A, C\}$, where: $U = \{u_1, ..., u_k\}, R = \{p_1, ..., p_m\}$, and $A = \{a_1, ..., a_n\}$, where U, R, and A correspond to users, pages, and activities including commenting, editing and tagging. A collaboration is an element of set C, where: $C \subseteq U \times R \times A$. Particularly to understand the collaborations of single user, we concentrate on the activities and pages that are associated with this particular user. We then define the set of individual collaborations as $I_u = (C_u, R_u, A_u)$, where C_u is the set of collaborations of the user: $C_u = \{(a, r) \mid (u, a, c) \in C\}$, A_u is the user's set of collaborative activities: $A_u = \{a \mid (a, r) \in C_u\}$, R_u is the set of pages: $P_u = \{r \mid (a, r) \in C_u\}$. More specifically, the collaborations of user u on the pages created by user w is denoted by $C_{u,w} = \{(a, r) \mid (u, a, c) \in C_u, w \in U\}$ and the collaborations of a user u a user on the particular page p is denoted by $C_{u,r} = \{(a, r) \mid (u, a, c) \in C_u, r \in R_w\}$. These notations will be utilized to describe the social capital model.

4.2 Measuring Social Capital in KiWi

Given the basis of the collaborative work, we introduce the model for calculating the social capital value.

4.2.1 Social Capital Value

The social capital of a pages built under collaboratively must contemplate the entire contributions (those who edit, tag, rate or comment). Thus, we define the social capital value $scv(p)$ of a page as a summation of contributions to that page weighted by the individual expertise.

$$scv(r) = \sum^{|u|} |C_{u,r}| \times userExp(u)' \mid u \in U, r \in R,$$

where $|C_{u,r}|$ is the total amount of collaborative activities performed by a user u on a particular page r, weighted by the user expertise calculated from $userExp(u)'$. The set of users is represented by U and R represents the set of existing pages in the system. The social capital value obtained with the function $scv(r)$ is utilized as an additional factor in our recommendation model to privilege those pages where the collaborative work is higher. In the following, we detail the pillars who sustain this model: the user expertise and social interactivity between social ties.

4.2.2 User Expertise

In addition to the collaborative work, we believe that the social capital value may vary according to the expertise of individuals who develop it. In this sense, we incorporate the user expertise of each individual as a weighting factor in our social capital model. Likewise the real world where

individual knowledge is built up from life experiences and academic education, in collaborative environments, the knowledge a person acquires can be derived from his contributions and participations within the community. All produced material and consumed information can be used as an indicator of user expertise. Based on this premise, we envisage a model to infer user's expertise from the pages that demonstrate user activity. Unlike approaches that rely on self-evaluation, we define the set of user expertise E as the most frequent terms that appear in the pages from R_u. For calculating E, we applied the *tf – idf* metric (term frequency – inverse document frequency) [1]. The importance of each expertise $e \in E$ is proportional to the term frequency and inversely proportional to the document frequency in the corpus, which is expressed as:

$$userExp(u) = \sum_{i=1}^{|R_u|} \sum_{j=1}^{|E|} tf(e_j, r_i) \times idf(t_j) \mid e_j \in T, r_i \in R_u, u \in U,$$

where $|R_u|$ is the total amount of pages created, edited, tagged, rated or commented by a given user u, r_i is a particular page, and e_j is a particular expertise (or term), $/E/$ is amount of expertise (top terms), R is a set of pages and U corresponds to a set of users. Figure 9 shows the user expertise of two users followed by its score. The set of terms that define a user expertise is automatically validated by set of stop words that reject senseless terms and also by an expert, which explicitly remove terms not related to expertise.

4.2.3 Interaction between Social Ties

Besides the own expertise, we believe that individuals may inherit knowledge from their social ties. This premise is motivated by [12], which claims that strong ties created between the members of a virtual team are essential for community's knowledge performance. Such knowledge inheritance however depends on the degree of interactivity with others in the network. As mostly individuals interact to each other, they contribute to build highly skilled relationships besides improving the overall social capital within community [10]. In this sense, we weight the inheritance of knowledge considering the interactivity between individuals as:

$$inter(u, w) = \frac{|C_{u,w}|}{|A| \times |R_w|} \mid u, w \in U,$$

where $|C_{u,w}|$ is the total amount of collaborations performed by user u on the pages created by user w, $|A|$ corresponds to the total amount of *possible* collaborative activities in the system, $|R_w|$ corresponds to the amount of pages created by user w, and U is the set of users. Provided this definition, we re-define the user expertise based on his interactivity as:

$$userExp(u)' = userExp(u) \times \frac{\sum^{|s|} userExp(s).inter(u, s)}{|s|} \mid u, s \in U,$$

where $|s|$ is the total amount of social ties, s is a particular tie of user u, and U is the set of users. Figure 9 the outcome of computing the user interactivity between social ties and the expertise per user.
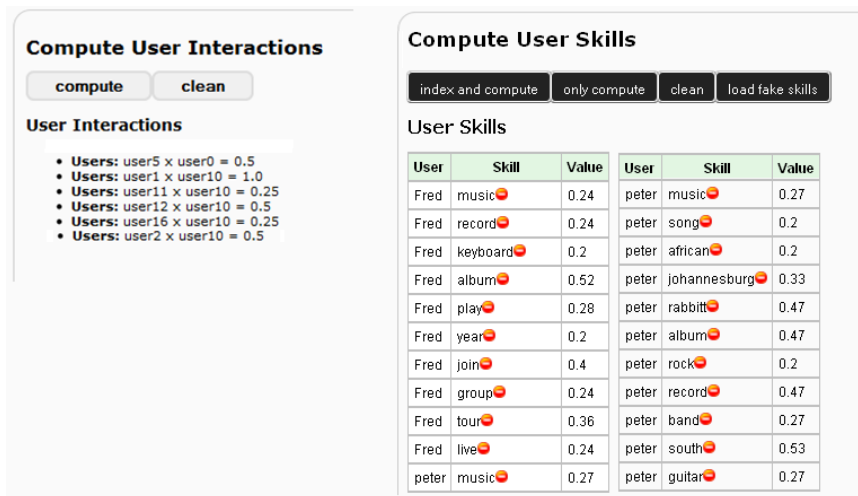
Figure 9: User interactivity and user expertise.

4.3 Requirements to use

An example scenario is visualized when several users (say authors of 4 other pages here) arbitrarily visit, edit, tag, rate, or comment these others 4 existing pages. Another user with Admin privileges visit the administration area in KiWi explicitly computing the expertise and interactivity between users (there are specific menu for both operations). In order to show how the social capital facilitates recommendations, the same requirements of standard recommendations must be fulfilled . As a matter of comparison, the logged user could enable both recommendation types, the standard and social capital based, so that he perceives that the ranking of recommendations has changed. The pages with higher social capital, i.e. where the collaborative place was more intense should appear at the top.

5 User Interface Customization

KiWi users can customize the information they want to visualize in the KiWi interface. In particular, they are allowed to choose between the different sorts of recommendations, widgets allocation and set navigation preferences. The settings are accessible in the Dashboard, through menu *My UI Preference*. There three tabs will be displayed containing all possible interface customizations.

5.1 Choosing Recommendations

Although different sort of recommendations can be processed by KiWi, users may have the option to decide which ones should be displayed. Either they enable all recommendations or choose those recommendations they prefer. In addition, it's possible to set the amount of recommendations per view. Figure 10 shows the interface where users can set up their recommendation preferences to be displayed.
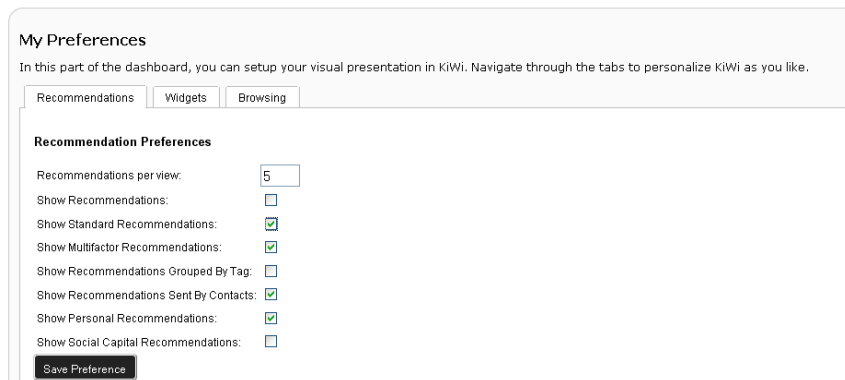
Figure 10: User interface preference.

5.2 Widget Allocation

The KiWi interfaces are composition of widgets that provide different services in parallel. For instance, at the dashboard, the KiWi user has four widgets containing recommendations, user history, tag clouds and the stream of activities. He might then customize the widgets according to his preferences. In this way, we provide a customization space where he decides which to be displayed.

5.3 Navigation Preference

Users also have the possibility of choosing among different perspective of navigation in KiWi.

- *Redirect to the last visited page*: After login users can opt between being redirected to Start Page or the last visited page.

- *Fill up the history menu with my last visited pages*: After login users can have their last visited pages in the history menu, otherwise the history menu is reset.

- *Remember the last searches*: After running a query on the search engine, users are reminded about their last searches so that they don't need to type again. Rather they just need to click on the remembered queries.

- *Watch only my friend activities*: In the Dashboard, users can see their activities mixed up with their friends' activities However, he can opt to filter the visualization only by his friends' activities.

5.4 My Recommendations

Among all content recommendations a user receive, some naturally might be more important than other. In this sense, we allow users to select the recommendations he prefers more just by dragging and dropping them from the recommendation panel to "My recommendations" panel. The recommendations at "My Recommendations" panel also can be removed in case the user decides to reorganize the items he wants to see in that list. Figure 11 shows a item being dragged to "My recommendations" panel.
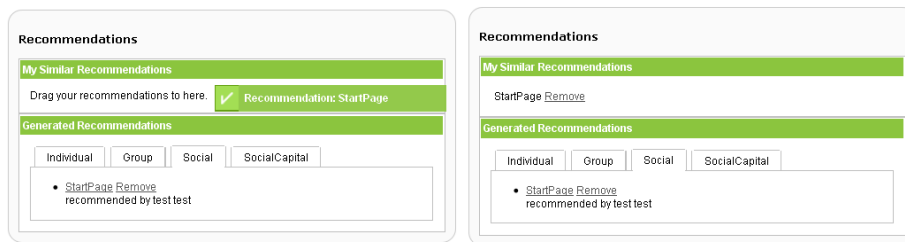
Figure 11: Recommendation being dragged to "My recommendations" panel. Afterward it appears in the list in the panel.

How to use it: At least one of the previous recommendation type must be generated. The currently logged user, must access the recommendation panel in the Dashboard and simply drag it to "My recommendations" panel.

5.5 Using Spectral Clustering for KiWi Page Organization

In KiWi system, we also implement clustering for better organizing the recommendations in a category presentation way. Although tag is an important metadata in social tagging systems, it possesses the inherent problems of free style term, uncontrolled and not well-defined vocabulary, less semantics, ambiguity and redundancy. It would be helpful and practical if we could capture the aggregation property of tags, which also partially tackles the aforementioned challenges. As such, we developed a cluster-based visualization function to fulfill this task by tag clustering.

The tag clusters provide a practical way to reveal the aggregated structures of tags associated with conceptual organizations of articles in Kiwi systems. And upon the discovered tag clusters, we are able to further envision a structured and visualized scheme to present the significant documents returned by the system. The recommendation scenario could work like this: when the user uses the tag-based personalization functionality, the existing KiWi system will provide a list of possible results to the user using the developed personalized search strategies. And we then reorganize the personalized search results in a grouped way, which helps retrieve the needed documents based on the tag clusters. Thus the additional clustering-based component of KiWi system is able to provide such a possibility.

To fulfill the aims, we intend to utilize clustering mining. First we conduct clustering on the total document set based on tag vector expression. Then each obtained tag clusters are formed. Then in personalization stage, each returned document is processed individually and classified to corresponding groups based on tag clusters. The documents in each document cluster are ranked in an order based on the tag frequency occurring in the documents. To have an optimized cluster presentation, we choose the top-N documents in each cluster to display. As a consequence, eventually we obtained a set of tag clusters and a set of corresponding documents clusters. The tag clusters and document clusters provide an additional option for users to visualize the personalization results.

When there are substantially more tags generated in the KiWi system, the user with Admin privilege can call the clustering mining again to get the updated tag and document clusters. The clustering-based visualization could be enabled and disable in *User Interface Preference*. Figure 12 depicts a snapshot of tag and document cluster presentation of personalization.
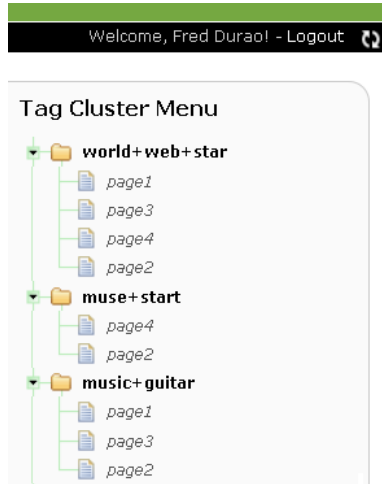
17

Figure 12: The cluster presentation of tags and corresponding documents

5.5.1 Spectral Clustering Algorithm

In KiWi system, we introduce the Spectral Clustering Algorithm in our work. Spectral clustering refers to a class of techniques which rely on the eigenvalues of the adjacency similarity matrix; it can partition all of the points into disjoint clusters. The points that have high similarity will be classified under the same cluster, while the points have low similarity with the points from the other clusters. The spectral clustering is based on the graph partition. The basic idea of spectral clustering is to map the original inherent relationships of subjects onto a new "spectral" space, on which each document is projected. The mathematical characteristics of algebra theory guarantee the new projected dimensional space is able to accurately measure the mutual similarity of data objects, in turn, resulting in a good quality partition. After the projection, the whole document profiles are simultaneously partitioned into disjoint clusters with minimum cut optimization.

Compared to traditional clustering algorithms, spectral clustering has many fundamental advantages. Results obtained by spectral clustering often outperform the traditional approaches, it is very simple to implement and can be executed in a much optimized way, e.g. dimensionality reduction.

To perform spectral clustering we first formulate the following Laplacian matrix:

$$L = I - D^{-1/2}SD^{-1/2}$$

where D is the diagonal matrix of tags similarity matrix, which are defined as: $D(i,i) = \sum_{j=1}^{N} Sim(D_i, D_j), i = 1, \cdots, N$.

According to the spectral graph theory, the k singular vectors, DV of the Laplacian matrix L present a best approximation to the projection of tag vectors on the new spectral space. In this case, running a classic clustering algorithm, e.g. k-means algorithm, on the DV matrix will result in the k-partition of the graph with the minimum normalized cut, and thus obtaining k tag clusters [14]. We then save the k tag clusters for the purpose of differentiating the search results into various groups.

5.5.2 How to use it

Minimally it is expected to have four pages in the system and each page must be tagged with at least 5 tags. Some tags must repeat over the pages. The tag clustering run in background after a predefined number of tags is added or explicitly by any user who is granted permission. On the 'Admin' interface, there is a menu for calculating the tag cluster. The user only needs to call computing the tag clustering and the results will be seen at any KiWi page. In case there are problems during the computation the user is notified about that running error. The tag clustering menu is also integrated with customization of user interface. Users can decide whether this component should be part of his view or not while browsing pages.

6 Personalization in KiWi Supporting Software Project Development: A Case Study

Software development is a highly collaborative activity that is usually performed by one or more teams, in a longer period of time, and in several stages. In every stage, team members need to share knowledge about actions they have performed solutions to the problems they have introduced, and so on. Adoption of tools and techniques intended to support collaborative project development requires knowledge and skills so that these tools can save personal time and reduce costs during the software development process.

Software applications supposed to assist members in the organization must comprise knowledge about designing management and development processes, planning and monitoring projects, managing requirements, quality assurance, configuration management, and measurement and analysis. This knowledge is necessary to overcome traditional obstacles evidenced over the years in software development environments such as lack of collaboration, technical deficiency, and communication problems. In addition, usually it is tricky to find all the relevant information when solving a specific task, sometimes even not knowing that relevant knowledge exists.

Aware to the above mentioned needs, personalization in KiWi aims at supporting collaboration of all parties involved so that tasks can be realized in parallel and synchronized with the project expectations. Particularly, we show how personalization features described early in this report will be utilized for presenting relevant content to project participants and allowing them to organize the workplace according to their preferences. In order to illustrate how personalization supports software project development, we proposed a real case scenario followed by sub scenarios where the participants involved benefit from the personalization features implemented in KiWi.

6.1 Real Case Scenario: Project Management for a Security System

A project manager is requested to coordinate the activities of a team who is developing a security system. He decides to utilize KiWi to ease the knowledge management and coordinate the tasks of his team. The system must provide different security techniques including permission, identification, authorization, cryptography and integration with public ids for thirty party users. In the first phase of the project, the entire functional features will be documented in requirement and use case documents.

The personalization goal is to assist the project manager as well as the development team to carry out the project activities and reduce the effort for accomplishing the tasks. Particularly,

personalization in KiWi must be able to support the project manager in monitoring activities, visualizing the work progress, finding potential experts and find pre-existing knowledge. The team must benefit from personalization by having an easy visualization of their designated tasks, personalized access to desired information and increasing the knowledge expertise by expanding their social connection with other security experts in the system.

6.2 Scenario 1. Project manager sets up the workplace and teamwork

The purpose of this scenario is to demonstrate how the project manager can benefit from personalization when he is setting up the workplace and managing teamwork.

1. The project manager logs in the KiWi system and creates tasks for his team. Each task can be represented by a new page containing the description of the each requirement to be implemented.

2. To each page, he assigns specific tags to categorize the document as well as to find it later on. Pre-defined tags are suggested by a software development ontology containing the pertinent vocabulary for project management. For loading the vocabulary, he will utilize the software development ontology as described [2].

3. He entitles one of the page as "Functional Requirement 1 (FR1): Single sign-on (SSO)" and assigns such tags "todo", "sso", "login", "requirement document", "functional requirement". Further, the other page is entitled as "Functional Requirement 2 ((FR2): Privacy Management" and is tagged with "todo", "privacy", "individual", "roles", "requirement document", "functional requirement".

4. Aiming at facilitating the communication with the group, the project manager adds the team members to his contact list. For managing his contact list, he will utilize the contact feature described in section 2.7.

5. Additionally, he creates a group "SWP-Security" and add all developers to this group. For creating a group and adding them to the group, he will utilize feature described in section 2.6.

6. The created group is tagged with "project", "security", "management", "identity", "privacy". Automatically, pages that share same tags will be recommended to all members of group "SWP-Security". The group recommendation feature also refers to section 2.6.

6.3 Scenario 2. Assigning tasks to team members

The purpose of this scenario is to demonstrate how recommendations in KiWi can help a project manager to distribute task to his team.

1. Once the workplace and teamwork is setup, the project manager then assigns tasks to the team members.

2. In order to assign the tasks to the team members, he visits the page and recommends it to responsible developer. For recommending the page that describes a task, he will utilize the contact feature described in section 2.7.

3. The target developers can visualize the assigned tasks in the recommendation panel in the Dashboard. For visualizing the recommendations, the team members will utilize the feature described in section 2.7.

6.4 Scenario 3. Task Execution and Content utilization

The purpose of this scenario is to demonstrate how personalization features can help project members to leverage content in KiWi.

1. The developers can access their tasks by clicking on the recommended items.

2. Just after browsing the assigned tasks, related recommendations will be displayed on the left side of the currently viewed page. These recommendations will assist the developers to fulfill their tasks by leveraging pre-existing content from previous projects, for instance. Related recommendations are generated by standard, multi factor or grouped based features visualized in the respective sections to section 2.1, 2.2 or 2.3.

3. The developers visit the recommended pages, assign more tags and rate them according to their preferences. After browsing all the pages, naturally their knowledge has expanded. The system then tries to infer what each developer is interested in. This computation of user expertise refers to the feature described in section 4.2.2.

4. Not fully satisfied with the recommendations, the developer responsible for the requirement FR1, searches for additional contents in KiWi. Let's say he queries for "authentication mechanism". The actual search results are presented differently depending on its type (e.g. pages, users and tags);

5. If the user is satisfied with the search results he naturally leaves the screen by clicking on the appropriate item in the search results.

6. If the user is not satisfied he can either enter a different query into the input field, or he can choose the option to personalize the search to change search results ordering. Instead of specifying a new query, he ticks the personalized search option and the retrieved results seem to be more interesting. The personalized search feature is described in section 3.

7. He then visits the first 5 pages retrieved by the personalized search. To each page visited he assigns more tags and rates them. The system again improves his expertise for the additional pages he had visited. This computation of user expertise refers to the feature described in section 4.2.2.

6.5 Scenario 4. Monitoring Activities

The purpose of this scenario is to demonstrate how project managers could benefit from recommendations to monitor activities of his team.

1. In order to monitor the team activities, the project manager then accesses his Dashboard page and chooses the option "show only my contact's activities". Immediately his view is switched so that he sees the stream of activities of each team member. This feature is described in section 5.3.

2. Meanwhile, the team members can ask for revision of their tasks by tagging the page with "needsRevision" and recommending the page to the project manager. For sending the page, he will utilize the contact recommendation feature described in section 2.7.

3. Alternatively, the project manager could create a personal rule to be recommended pages tagged with "needsRevision". The rule based recommendations are described in section 2.8. Still in the Dashboard, he can quickly visualize all tasks under revision.

4. In order to assess the work progress, he can rate the current stage, edit the content or leave comments.

5. Still in the Dashboard he can always visualize his last activities in the system including visited, edited, tagged and searched pages. This feature is described in section 5.3.

6.6   Scenario 5. Organizing the workspace

The purpose of this scenario is to demonstrate how information overload could be managed so that project member could focus only on the information they need.

1. As long as the work progress evolves, it is normal that the amount of information produced might distract the team members from their target activities. In order to reduce the information overload, the developers might need to adjust the interface according to their preferences. The user interface preference can be set in the features described in section 5.

2. The project manager and the team members, for instance, could choose to accept only recommendations sent by their contacts and items recommended by their predefined rules. The recommendation preferences are set in section 5.1.

3. KiWi also could assist the project manager by grouping similar pages and suggest a label to each grouping. The project manager however can update the group labels to a preferred name. The clustering features for grouping similar items are described in section 5.5.

4. Regarding the recommendations, the project manager could modify the visualization of his recommendations from the standard view to the tag-based view. Additionally, he could reset the amount of recommendations per view from 10 to 5. This feature is described in section 5.3.

5. The recommendations that are more attractive are dragged to "My Recommendations" panel to be quickly visualized once he returns to the Dashboard. This feature is described in section 5.4.

6. Since the team members are focused on a specific task, they could also set the interface preference to always get redirected to the last visited page before logout. The navigation preferences can be seen in section 5.3.

7. The project manager and the team members still can choose how the history menu will be setup. Either displaying the first visited page or presenting the last visited pages from last session. The visualization preferences can also be set following the feature described in section 5.3.

6.7 Scenario 6. Networking and Managing Contacts

The purpose of this scenario is to demonstrate project members could expand their social network or being in contact with experts from related areas of interest.

1. He also could add those persons to the group "SWP-Security" where the new group members could immediately recommend content to the group they think will be helpful. The recommendation to group feature is described in section 2.6

2. In the contact page, at the Dashboard, KiWi also recommend to the project manager potential contacts who are also working with security issues. The recommendation of contact can be visualized in the feature described in section 2.7.

7 Personalization: Installation and Configuration

Most of personalization in KiWi is placed at the Dashboard, a personal space where a user can define his profile information, visualize all ongoing activities, manage social network and customize the interface preferences. The other personalization features such as recommendations are integrated to the KiWi core as extension services. Enabling or disabling personalization in KiWi implies therefore in switch on/off the Dashboard component or extension services for personalization.

7.1 Installation

Basically, the KiWi system is formed from two kind of components: the core component and extensions. The Dashboard is one of the extensions and to be installed in KiWi, it must appear in both files:

- $kiwi - dir/resources/META - INF/application.xml$ - this file contains all extensions as external components that will be integrated to the core system.

- $kiwi - dir/build.xml$ - this file must contain only those extensions what will appear in the next deployment.

Figure 13 shows the Dashboard components published in both files respectively.

7.2 Enabling/Disabling Personalization

Once the Dashboard extension is installed it can be visualized in the upper menu. At the interface level, users still can enable/disable through the Admin interface. Figure 14 shows the Admim interface where Dashboard is switched on/off. By disabling the Dashboard, the upper menu is updated by removing the Dashboard menu. The more technical information on how to create extensions and integrate them into KiWi will not be discussed in this report but can be seen at [11].

Interface preferences are set at the Dashboard through the "My UI Preferences" menu. There, users can choose many sorts of recommendations, widget organizations and navigation preferences. A more detailed explanation will be discussed at section 5.

Figure 13: Dashboard components published in the application.xml and added to the build.xml to be deployed
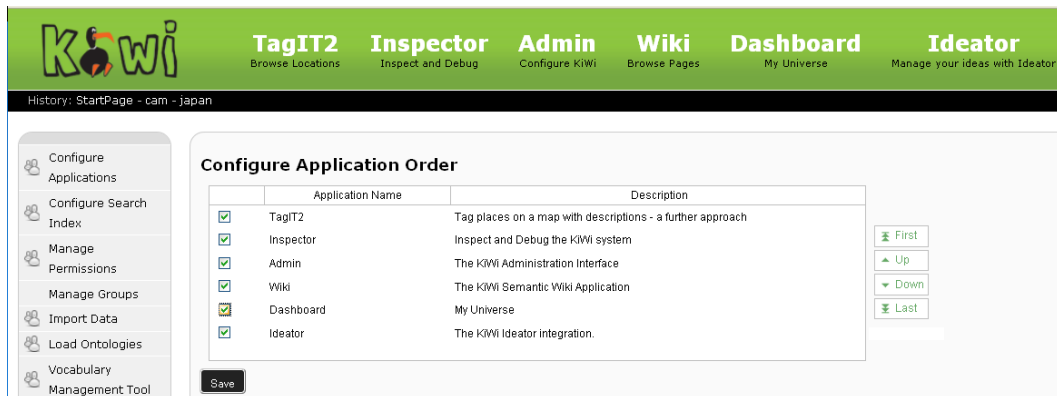

Figure 14: Admim interface where Dashboard is switched on/off

8 Conclusion and Future Works

In this report we presented the personalization component implemented for KiWi addressing diverse aspects such as motivation scenarios, formal models, implementations and the minimal requirements necessary to install and realize the benefits of personalization in KiWi. Further we presented a number of scenarios on how personalization could be applied to support software project management and project development.

As shown, most of personalization in KiWi is conceived by observing user's **tagging** activity aiming at building tag-based user models and thereby delivering the appropriate information to right users. We leverage user tagging information to *personalize recommendations*, *group users* and *personalize search*. In addition to that, we analyze the *social activity* within KiWi pages in order to evaluate which pages are preferable to be recommended over the others. Furthermore we provide a set of facilities so that users can customize the interface preferences and define which piece of information should be displayed while they are using the system.

The future works could be along the following directions: we plan to improve the recommendation model taking into account not only the syntax similarity between tags but also the semantic relatedness. This enhancement could be realized by analyzing tag relationships across to domain ontologies or statistically considering tag neighbors from the perspective of co-occurrence. As a common ground research, we aim at enhancing our tag-based models by considering the tag decay. The goal is to perform a temporal analyzes and filter the results according to the actual users' preferences. In addition, the social capital model should consider the structure of social network aim at targeting, for instance, individuals not very active in the community. Regarding the rule-based recommendations, a number of template rules could be provided to users so that individuals are assisted to build their rules by simply filling out a few entry points. Besides, some recommendations could be provided of an explanation. For instance, the friendship recommendation would benefit of a explanation for the user, i.e. this person has been recommended to you because you have many social ties in common. From the user's interface perspective, we plan to increase the user's interactivity with the system. Users could for instance change manually the ranking of search results or recommendations. Further, users could drag and drop widgets to arrange the interface they prefer to work. Each of these ideas are promising features to KiWi and based on a realistic analysis on what could bring value to the KiWi system in terms of personalization.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, May 1999.

[2] P. Dolog, F. Durao, D. Grolin, K. Jahn, P. A. Nielsen, A. Munk-Madsen, and K. Pedersen. Knowledge model: Project knowledge management. Technical report.

[3] F. Durao and P. Dolog. A personalized tag-based recommendation in social web systems. In *International Workshop on Adaptation and Personalization for Web 2.0 (AP-WEB 2.0 2009)*, volume 485. CEUR.org, 2009.

[4] F. A. Durão and P. Dolog. Analysis of tag-based recommendation performance for a semantic wiki. In *4th Semantic Wiki Workshop (SemWiki 2009) at the 6th European Semantic Web Conference (ESWC 2009), Hersonissos, Greece, June 1st, 2009. Proceedings*, 2009.

[5] F. A. Durao and P. Dolog. Personalisation: Concept and model. Technical report, April 2009. KIWI project FP7, Project Number: ICT-2007.4.2-211932 Document number: ICT211932/SRFG/D2.8/D/PU/b1.

[6] F. A. Durão and P. Dolog. Social and behavioral aspects of a tag-based recommender system. In *Ninth International Conference on Intelligent Systems Design and Applications, ISDA 2009, Pisa, Italy , November 30-December 2, 2009*, pages 294–299, 2009.

[7] F. A. Durão and P. Dolog. Extending a hybrid tag-based recommender system with personalization. In *Proceedings of the 2010 ACM Symposium on Applied Computing (SAC), Sierre, Switzerland, March 22-26, 2010*, pages 1723–1727, 2010.

[8] F. A. Durao, P. Dolog, and K. Jahn. State of the art: Personalization. Technical report, 2008. KIWI project FP7, Project Number: ICT-2007.4.2-211932 Document number: ICT211932/SFRG/D2.7/D/PU/b1.

[9] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Information retrieval in folksonomies: Search and ranking. In Y. Sure and J. Domingue, editors, *The Semantic Web: Research and Applications*, volume 4011 of *Lecture Notes in Computer Science*, chapter 31, pages 411–426. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[10] A. Nazir, S. Raza, and C.-N. Chuah. Unveiling facebook: a measurement study of social network based applications. In *IMC '08: Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*, pages 43–56, New York, NY, USA, 2008. ACM.

[11] S. Schaffert, R. Sint, S. Grunwald, and S. Stroka. The kiwi architecture. Technical report, October 2008. KIWI project FP7, Project Number: ICT-2007.4.2-211932 Document number: ICT211932/SRFG/D3.1/D/PU/b1.

[12] L. Striukova and T. Rayna. The role of social capital in virtual teams and organisations: Corporate value creation. *Int. J. Netw. Virtual Organ.*, 5(1):103–119, 2007.

[13] G. Williams. *Linear Algebra With Applications*. Jones and Bartlett Publishers, Inc., USA, 2007.

[14] G. Xu, Y. Zong, P. Dolog, and Y. Zhang. Co-clustering of web log using bipartite spectral clustering. In *in the Proceedings of 14th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems*, Cardiff, Wales, UK, 8-10 September 2010.