2010 International Conference on Global Software Engineering

Distributed Requirements Elicitation Using A Spatial Hypertext Wiki

Carlos Solís
Lero, the Irish Software Engineering Research Centre
University of Limerick
Limerick, Ireland
Email:Carlos.Solis@lero.ie

Nour Ali
Lero, the Irish Software Engineering Research Centre
University of Limerick
Limerick, Ireland
Email:Nour.Ali@lero.ie

***Abstract*—In Global Software Development (GSD), distributed stakeholders (e.g. team members, customers, etc) have to collaborate and communicate in an efficient and effective way to share, create and discuss knowledge. Nowadays, a challenge is to provide integrated collaborative tools that implement creativity techniques which allow distributed stakeholders to externalize their knowledge through brainstorming and share and store knowledge in a common repository. The Requirements Elicitation (RE) process is a clear example where this kind of support is needed in the software development process. This paper presents the Spatial Hypertext Wiki as a collaborative tool for supporting creativity in the RE process. The Spatial Hypertext characteristics of the wiki provide a virtual board where distributed stakeholders can share, brainstorm, negotiate, or prioritize the knowledge involved in RE.**

***Keywords*-Requirements elicitation, spatial hypertext wiki, ShyWiki, global software development**

I. Introduction

In the current competitive and dynamic economy, software companies are becoming virtual organizations that distribute their projects on multi-sites seeking for specific skills or expertise, or to reduce costs [1]. However, many challenges exist in this kind of software development, essentially the ones related to coordination over distance [2].

In any project, Requirements Elicitation (RE) is a critical phase due to the fact that most software project failures are caused from inadequate requirements [3]. The RE phase is the first step in the requirements engineering process, in which the requirements or needs that a system has to satisfy are discovered [4]. Requirements elicitation is a creative process in which all stakeholders collaborate in the creation of the needs that describe a new system [5]. The stakeholders involved in the requirements elicitation process must understand a domain, and the problems that the different stakeholders want to solve using a software system. Some of the proposed needs will become system requirements after their negotiation and prioritization [4]. In requirements elicitation, diverse methods are used such as interviews, workshops, brainstorming, and protocol analysis [6].

In a global software development context, the distribution of the stakeholders adds additional difficulties to the requirements elicitation. When collocated software development is followed, requirements elicitation is a face to face activity that can be performed using interviews or requirements workshops. However, in the case of geographically distributed teams, face to face communication cannot be possible due to distance or time differences. Teams have used asynchronous communications such as emails to overcome these problems. However, interchanging huge volume of emails is difficult to track [7]. Requirements negotiation in global software development is an essential challenge to be overcome [2]. Tools have to be provided in order to support distributed RE in a similar way to the traditional RE methods in collocated environments such as brainstorming, and workshops.

Wikis are tools for distributed and collaborative work, and they can be used to solve problems behind distributed requirements elicitation [8]. A wiki is a web based software that allows the collaborative and incremental creation of hyperlinked web pages [9]. Wikis are based on the principles of easy of use, incremental content creation, open structure for editing and evolution, and self organized structure [10]. The content in a wiki page is defined by using a simple mark-up language, which allows the user to format the content and create hyperlinks. In this way, users do not have to be technical experts in the edition and design of hypertext. The creation of new wiki pages is achieved simply by navigating through a link. In addition, most wikis provide versioning facilities, which permit reviewing previous versions of wiki pages or the roll-back of unwanted changes. A good example of the success of the wiki concept in the web is Wikipedia [11].

Wikis have been used in global software development projects to communicate, coordinate, track, discuss, test and document the work [7], [12], [13]. For example, in Open Source Software (OSS) projects where the development is performed by a community which is often distributed, wikis and forums are becoming the most used tools for requirements elicitation [14]. Wikis permit requirements and domain languages to be captured. According to Asmari and Yu, wikis are easier to use, more reliable and cheaper than other tools for communication and coordination in distributed development [12].

978-0-7695-4122-8/10 $26.00 © 2010 IEEE
DOI 10.1109/ICGSE.2010.35

A basic technique in the elicitation process is brainstorming. However, wikis are not well suited for supporting collaborative brainstorming because they lack of a virtual blackboard. An essential feature to be provided for brainstorming is the capability of allowing participants to collaborate by reorganizing spatially notes (which contain ideas, or certain knowledge). Without this characteristic, the emergence of ideas and creativity is limited [15].

This paper presents a spatial hypertext wiki (ShyWiki) for requirements elicitation. The wiki principles provides to ShyWiki easy to use characteristics. Spatial hypertext [16] facilitates the externalization of tacit knowledge and permits to manipulate the requirements in a spatial blackboard. Thus, each wiki page can be seen as a virtual board where distributed stakeholders can add, move, or group notes. The requirements elicitation process supported by ShyWiki is based on the KJ externalization method [17], and EasyWinWin [18] elicitation process which provides support to the negotiation and prioritization of requirements.

This paper is structured as follows: section II presents the background consisting of an explanation of what spatial hypertext is, and a brief description of the WinWin [19] and EasyWinWin [18] requirements elicitation methods. Section III explains why the spatial manipulation of the requirements is important in brainstorming elicitation. Section IV gives an overview of the Spatial Hypertext Wiki. Section V explains ShyWiki support to requirements elicitation. Section VI presents the related works. Section VII discusses some properties of ShyWiki, and how the requirements elicitation process is supported by ShyWiki in comparison to the EasyWinWin tool. Finally section VIII gives the conclusions and future work.

II. Background

A. Spatial Hypertext

In 1968, Engelbart implemented the first hypertext system [20] called the "oN-Line System" (NLS) which provided hyperlinks that connected documents. This kind of hypertext is called document centred. However, using this kind of navigation in large networks of documents, users can get lost in the hyperspace [21]. A way to solve such problem is by using map based hypertext, which shows explicitly the relations among hypertext documents [22]. Another way, is by using implicit relations.

Spatial hypertext [16] is a kind of hypermedia that is based on using visual and spatial characteristics to define relations among hypertext elements, which are seen as "sticky notes" or bibliographic cards that can hold hypermedia content (text, images, hyperlinks, etc.). Spatial hypertext can represent implicit hypertext structures, which are interpreted depending on the note's spatial context [23]. In this way, the relations that are explicit in the map based approach are represented implicitly by using visual and spatial characteristics, therefore, hyperlinks become implicit.

The relations of elements in spatial hypertext can be represented in several ways: the notes in the document can be positioned to form lists, stacks or only being near each other. Also, notes of the same type can be represented by sharing the same visual and spatial characteristics: colour, borders, font types, adornments, layout, position, proximity, geometric relations, etc. The notes can be contained inside other notes, creating collections.

Spatial hypertext systems have special facilities in their user interface [16]. For example, users can handle and move notes from one place to another in a hypertext document, or can change their visual properties or their size. This way of organizing the information allows users to describe complex relations among notes.

B. WinWin and EasyWinWin

The WinWin Method [19] is a requirements negotiation approach where each stakeholder expresses her or his system needs as a list of winning conditions. When a winning condition has conflicts with the winning conditions proposed by other stakeholders, a win-lose situation happens [24]. WinWin provides principles and practices for finding win-win conditions shared among stakeholders of the project.

In WinWin, a conflict among winning conditions of different stakeholders is called an issue. An issue is associated with the conflicting winning conditions, and has a description of the conflict. For solving an issue, the stakeholders propose alternative solutions which are called options. Then, the stakeholders have to evaluate the options and select or reject some of them. After several iterations over the options list, the stakeholders can have an agreement about the solution adopted. During the negotiation process, domain terms appear in the descriptions of the winning conditions, issues, options and agreements. These domain terms have to be defined in a glossary, and structured in domain taxonomy.

EasyWinWin [18] is a lightweight WinWin method that has lower entry barriers for all the stakeholders. In EasyWinWin stakeholders define their winning conditions through brainstorming. The stakeholders collaboratively define their winning conditions in an electronic brainstorming tool. In this way, they can share and view the different available winning conditions. The win conditions that are similar to others can be merged. As a result, a new win condition is created. The previous brainstorming statements are attached to the winning condition in order to preserve the rationale. In addition, the winning conditions can be organized in a taxonomy. For instance, a winning condition can be a refinement of others.

In the next step, the stakeholders prioritize the winning conditions in order to define their importance. The prioritization has two perspectives, the business perspective, and the ease of realization. The first one defines the importance for the business organization, and the second the perceived difficulty of achievement. Both perspectives are measured in

the scale from 1 to 10, being 1 unimportant and difficult, and 10 very important and easy. After, the prioritization step it can be observed which tasks are important and easy to accomplish, which are important and hard, which are easy to do but without importance, and which tasks will not be performed for being difficult and without importance.

The EasyWinWin tool is a collaborative groupware system, which has been used in more than 50 projects [25], and it has succeeded in capturing the initial requirements of those projects. However, EasyWinWin lacks flexibility when the requirements evolve, cannot link other information resources in a local or distributed repository, is difficult to share relevant information for the requirements, and is not as easy to use as wikis [26].

III. The importance of space in Brainstorming

The concept of tacit knowledge was defined by Polanyi as knowledge that cannot be easily shared. Tacit knowledge is composed of intuitions, unarticulated mental models, or technical skills [27], [28]. Tacit knowledge is personal, context specific, and hard to communicate to others. Tacit knowledge has individual cognitive elements such as mental maps, beliefs, paradigms and viewpoints, and concrete technical elements know-how, and context specific skills [29]. Tacit knowledge in organizations is related to undocumented work practices that workers use to take decisions. In addition, much of the knowledge about a future system is tacit [28].

On the other hand, explicit knowledge is articulated, codified, and can be communicated in natural or symbolic languages [28]. When the stakeholders try to express their requirements, they are converting part of their tacit knowledge of the system into explicit. Requirements have to capture this undocumented knowledge in order to have a complete specification of the system.

The use of creativity techniques is indispensable in the requirements elicitation process, because creative stakeholders can think in an innovative way [30]. Creative stakeholders are an important factor in the innovation of the companies [31]. That fact has been confirmed by researches that have applied the creativity techniques during the elicitation process [32]. According to Mich *et.al.* [33], brainstorming is most the used creativity technique for requirements elicitation.

Externalization is the process in which tacit knowledge of an expert is transformed into explicit [28]. A technique commonly used for externalization is the Kawakita Jiro (KJ) method [17] which is used by Japanese companies to evaluate and organize information [34]. The KJ method consists of the following phases: In the first phase, ideas are generated and written on adhesive notes without any evaluation or critique. In the second phase, ideas are grouped. If the ideas are related, they create a spatial group of ideas, which is created by moving related notes to the same space. In the third phase, the goal is to create a consensus about the solution to be adopted. In this phase, the groups of ideas are categorized, i.e., ideas of a group are considered to be the best solution, then the second, and so on. For example, when an idea on an adhesive note describes a better solution than the others, it can be put over the other notes. The nominal group technique [35] adds a voting step to the brainstorming, where each participant votes about the risks in the realization of the ideas.

Cox and Greenberg define collaborative interpretation as the activity of transforming information fragments into coherent descriptions, and where emergence of ideas happens [15]. Externalization is a collaborative interpretation task. Software tools that support emergence of ideas and collaborative interpretation need to satisfy the following requirements [15]: provide a spatial visual workspace, let people express relations among data using spatial proximity, allow free-form annotation of the underlying space, and allow the free creation and movement of data in the space.

None of the above mentioned requirements are supported by current wiki technologies. On the other hand, collaborative interpretation tools such as Gungen DX II [36] do not provide wiki features. Wikis can be adapted to the specific needs of the business, provide a common information repository and can use hyperlinks for structuring information. An integrative collaborative tool which combines the features of collaborative interpretation tools and wikis can improve collaboration in global software development.

IV. Overview of ShyWiki

ShyWiki [37], [38] is a wiki which uses spatial hypertext for representing its content. On a ShyWiki page, each element is similar to an adhesive note. The content of the wiki pages is spatially organized: notes may be placed in different regions, moved around, and can be of different sizes and colors (see Figure 1). Each of these notes can contain elements of formatted text, images or other types of media. The purpose of the notes is to define the attributes that characterize the concept represented by a wiki page. Properties of notes such as their colors or positions are used for relating them.

Figure 2 summarizes the ShyWiki Hypertext Model. The ShyWikiWeb is composed of information and knowledge stored in WikiPages connected by hyperlinks. WikiPages are identified by a unique name and are composed of notes. The AbstractNote class includes the properties which are common to other kinds of notes: position (x,y), width, height, color, etc. ContentNotes hold content of different hypermedia types and can be composed of other ContentNotes. A TranscludedNote is a note whose content is defined by another note, it is a reference to another note.

ShyWiki can represent structured data by means of templates, and labeled hyperlinks [39]. A template is a wiki page that can be instantiated into many wiki pages, hence

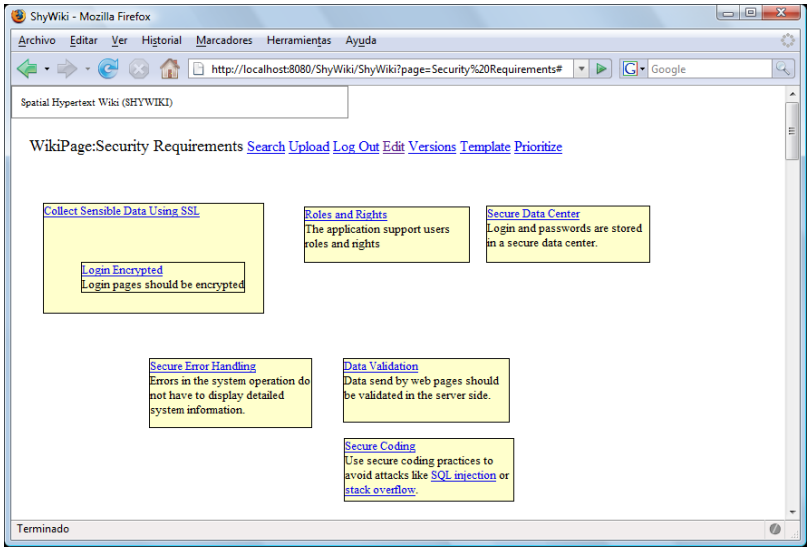

Figure 1. A wiki page in ShyWiki

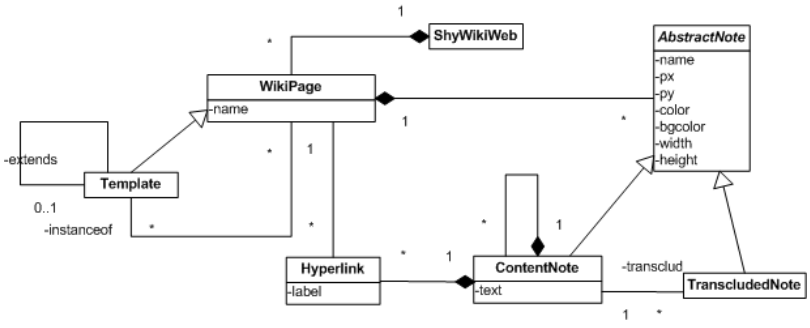

Figure 2. ShyWiki Model

providing reusability. Notes defined in a template, are also created in its wiki page instances. In addition, during the definition of a template, the associations that instances can have with other concepts can be indicated as well as their cardinalities. A labeled hyperlink is a link with a type. In this way, users can define the semantics of the association represented by a hyperlink.

ShyWiki supports the model in Figure 2 by providing basic operations to create or modify wiki pages. In the edition mode, a user can perform the following actions: creating wiki pages, creating, editing, moving, grouping and transcluding notes, and creating and instantiating templates. In addition, ShyWiki has a versioning component that permits previous versions of a wiki page to be tracked, as well as a search component for locating text or phrases.

V. Requirements Elicitation in ShyWiki

The elicitation process supported by ShyWiki is based on the KJ method [17], the Nominal Group Technique [35], and EasyWinWin [18]. Figure 3 presents the activities flow in the ShyWiki elicitation process: initial setup, requirements generation, grouping, capturing domain language, prioritization, and refinement. Although Figure 3 shows that the process ends after refinement, ShyWiki does not restrict this. Brainstorming can be performed many times before the RE process ends. These activities are explained in the following.

A. Initial setup

The initial setup consists of identifying the initial stakeholders, and requirements categories (in EasyWinWin they are called win categories). The identification of stakeholders helps participants to understand the social context in which the project will take place. In the initial session, the software system is divided into categories of negotiation topics. For example, if WinWin method is followed, the five requirement categories can be included [19]: project win condition, capability win condition, system interface win condition, level of service win condition, and evolutionary win condition. The categories to include in a project can be negotiated in a previous brainstorming session.

ShyWiki provides a setup page that builds a wiki page for the project, a wiki page with the stakeholders list,

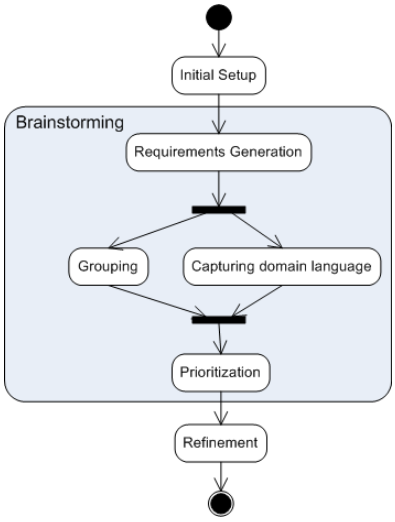

Figure 3. ShyWiki Elicitation Activitities

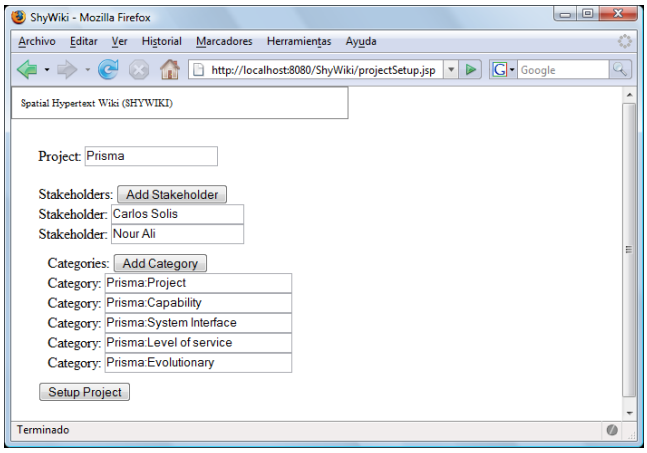

Figure 4. Project setup interface

a wiki page for each stakeholder, a wiki page with the initial requirements categories, and a glossary wiki page (see Figure 4). The wikipage of each stakeholder is an instance of a template called stakeholder. The stakeholder template includes the following information: name, email, phone, and role in the project.

B. Requirements Generation

ShyWiki supports externalizing stakeholders' interest through the brainstorming technique. A brainstorming session has to be performed for each requirement category. In this step, the system to develop can be seen as a problem to solve, and the requirements are the solution to that problem. A brainstorming has a divergent phase where the possible ideas are expressed without criticism. The goal of this phase is to collect as many ideas as possible.

ShyWiki has many advantages over traditional brainstorming and brainwriting. By using ShyWiki, there is no need for a room with a blackboard or paper notes. The session participants can be geographically distributed. ShyWiki can support different brainstorming sessions in parallel, one for each team working on a particular problem or category. In addition, an idea in a brainstorming session can be expanded in other brainstorming sessions in order to reduce the complexity and produce ideas with better quality [40]. Therefore, if there are many complex requirements, the problems related to them can be analyzed using nested brainstorming sessions.

The participants can easily enrich other notes, by adding new content to the original ones, or by adding annotations to them. The drag and drop facilities aid in grouping ideas. In addition, ideas can be classified using background or border colours. As a result, the relations that can be expressed among ShyWikis' notes are richer and easier to manipulate than the blackboard or paper based techniques. ShyWiki can automatically preserve a brainstorming session, which can be reconstructed if required later on. Furthermore, the use of the versioning facility of a ShyWiki page can enable users to track the path followed by a team to make a decision. For example, Figure 1 shows a brainstorming session about the security requirements of a web application.

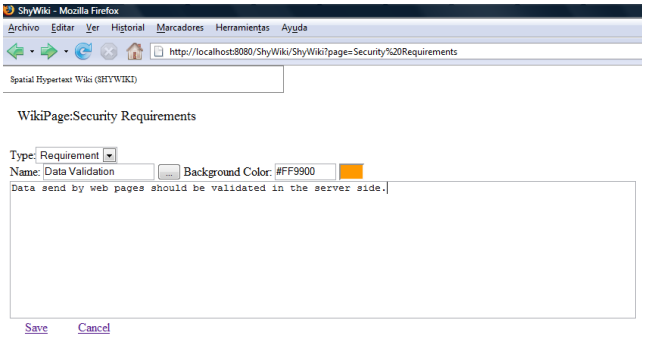

Figure 5. Adding a requirement note

Any wiki page in ShyWiki can be a blackboard used for brainstorming. If a new requirement has to be added to the board, the stakeholders have to add a new note of type *requirement* (see figure 5), and define the name and description of the requirement.

C. Grouping

In this step, the stakeholders that participate in the brainstorming session, have to eliminate redundant and ambiguous requirements. With the remaining requirements, groups of related requirements are made by finding out whether a requirement can be part of another one and merging similar requirements together. Each group of requirements has to receive a name. If a new requirement occurs to someone, then it can also be added to a group. Stakeholders can use the grouping and spatial properties of ShyWiki in order to organize the clusters and hierarchies of requirements.

D. Capturing domain language

The project wiki page includes a hyperlink to the glossary of terms that will be used in the communication process of the stakeholders. Stakeholders might use the same word with different meanings. This is an essential step to be taken since understanding the accurate meaning of terms helps in reducing ambiguity and thus reducing misunderstandings and miscommunication. In this way, the different stakeholders can be aware of the terms, and can speak a common domain language.

The first step is to add the new terms to the project glossary. The definition of the new terms is explained with detail in the wiki page corresponding to the term name. The open nature of wikis permits any stakeholder to participate in the definition of a term. In addition, the versioning capability of the wiki allows the stakeholders to observe the evolution of the definition. The definition of the terms can be performed incrementally during the grouping step. In the post-it notes of the brainstorming sessions, some terms of the domain language are mentioned. For each term in the domain language, a wiki link could be added to the post it note to allow the users to navigate to the definition and check the meaning of the terms. For example, in Figure 1 the note about *secure coding* has hyperlinks to the terms *SQL injection* and *stack overflow*.

E. Prioritization

ShyWiki allows stakeholders to vote about the importance and difficulty of a requirement. A stakeholder can assign a value from 1 to 10 to them. The voting action can only be performed in wiki pages that are instances of the *requirement template*. The voting interface is shown in Figure 6.

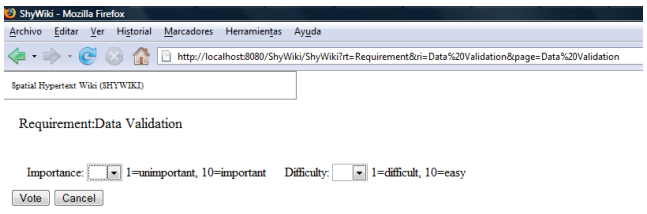

Figure 6. Voting the priority of a requirement

In ShyWiki, any wiki page that includes notes which represent requirements (instance of the requirement template) has a hyperlink that permits visualizing their priority. As a result, a web page that shows the list of requirements contained in the wiki page ordered by priority is available (see Figure 7). The list is ordered according to the requirements importance to the business, and by their difficulty. If the importance or difficulty is greater than 5, then they are shown in green colour. Otherwise, they are shown in red.

Figure 7. Prioritization of a set of requirements

F. Refinement

The refinement of a requirement represented by a note, can be performed in the wiki page associated to a note. In this step, the requirements are defined with more precision and detail, which is often called requirements modeling [3]. We have defined a set of templates for refining requirements. Figure 8 shows the UML model of the templates defined and their relations. The templates are: *requirement*, *project*, *stakeholder*, *option*, and *agreement*, and the empty templates *status*, *priority* and *difficulty*. The *status* template has the instances *accepted*, *rejected*, and *pending*. The *priority* and *difficulty* templates have the instances *low*, *medium* and *high*.

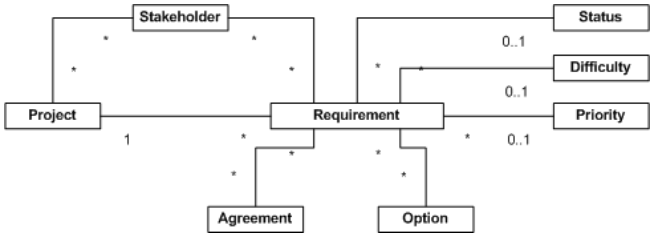

Figure 8. UML model of the RE templates and their relations

Figure 9 shows the requirement template, which includes the following properties: *name*, *number*, *description*, and *comments*. The *name* property indicates a very short description of the requirement. The *number* property is to provide a requirement identifier. In the *description* property the user has to indicate a detailed explanation of the requirement providing its rationale, concerns, and conditions to satisfy. The *comments* property is used to indicate relevant issues about the requirement. The template also shows the following relations: *status*, *priority*, *difficulty*, *stakeholder*, *options*, and *agreements*. The relations *status*, *priority*, *difficulty* are 1 to 0..1, while the relations *stakeholder*, *options*, and *agreements* are 1 to 0..N.

Instances creation is associated with users navigation. For example, Figure 5 shows how the user added the *data validation* requirement to the *security requirements* wiki

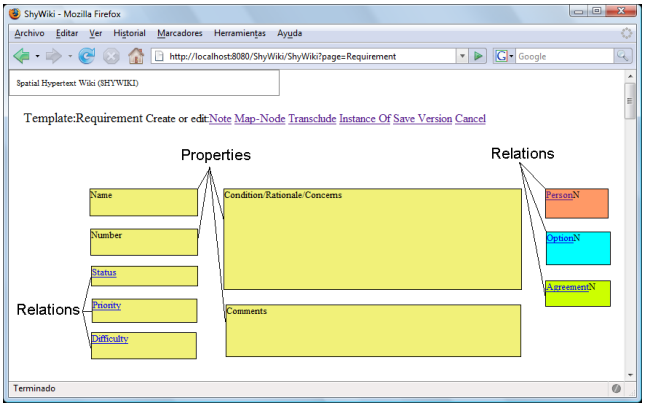

Figure 9. Requirement Template

page in Figure 1. When the hyperlink that points to the *data validation* wiki page is navigated for the first time, the wiki page is created as an instance of the *requirement* template. Figure 10 shows the instance of the requirement *data validation*. It can also be observed that the requirement instances show the average importance, and difficulty resulting of stakeholders' votes. The stakeholders involved in this requirement, *Cristobal* and *John*, were selected from the stakeholders list, which is shown to the user when a note representing the stakeholder relationship is edited.

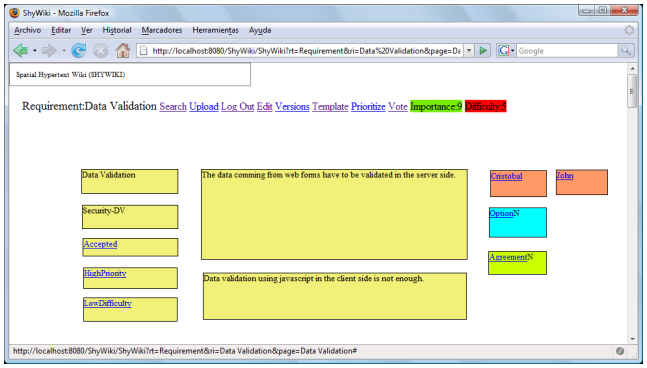

Figure 10. Requirement instance

VI. Related Work

In this section, other tools that have been used for global and distributed requirements elicitation are presented, focusing on wikis.

RequisitePro [41], Doors [42], CaliberRM [43] are elicitation tools that provide support to capturing requirements and traceability, and provide a desktop and a web interface. According to Sinha *et.al.* [44], they lack "deep integration between the requirements and communication environments". EGRET (Eclipsebased global requirements tool) [44] is an eclipse plug-in for distributed requirements management. It allows distributed stakeholders to share a repository of requirements and include contextual communication around the requirements.

Many wikis have been adapted for capturing software engineering requirements. The viability of wikis for requirements elicitation has been demonstrated by their use in industrial and academic projects [8], [26], [45].

WikiWinWin [26] is a wiki that implements the WinWin method. It is based on web forms. Therefore, WikiWinWin does not support the main characteristic of ShyWiki which are facilitates for supporting the emergence of ideas and the collaborative interpretation.

SOP-wiki [8] uses semantic media wiki to capture requirements. In addition, SOP-wiki has an especial extension for presenting links to the wiki pages that represent requirements documents (requirements, use case, actor, etc.). SOP-wiki permits the creation of release versions, and the exportation of the requirements to open office. SOP-wiki has been used in several academic and industrial projects.

SoftWiki Ontology for Requirements Engineering (SWORE) [46] is an ontology for representing requirements defined in the Web Ontology Language (OWL). The instances of the ontology are managed and captured using the semantic wiki OntoWiki [47].

SmartWiki for requirements engineering [45], is based on semantic media wiki with templates for requirements engineering. Their templates are based on the use case templates of Cockburn. In addition, it provides functions for project management. ProjectIT-Enterprise [48] is another semantic wiki with templates, and project management support.

In general, wikis used for requirements engineering provide wiki templates, and typed relationships that permit to define the semantics of objects and relations in the domain of requirements engineering. They provide templates for stakeholders, uses cases, user stories, projects, etc. ShyWiki can define templates and relations, and it is possible to define a conceptual model for requirement engineering in each project. Complex ontologies for requirements elicitation, such as the one provided by SoftWiki, have the problem of cognitive overhead due to the fact that it imposes a structure to stakeholders argumentation.

Although, requirements elicitation uses brainstorming for capturing and prioritizing requirements, all the mentioned wikis do not have adequate support for this activity. Even WikiWinWin, which uses brainstorming as a central element, does not provide support for collaborative interpretation i.e. a virtual board. The main difference between ShyWiki and the other wikis is that ShyWiki satisfies the requirements that a tool has to comply in order to facilitate collaborative interpretation tasks.

VII. Discussion

This section presents how the requirements elicitation process is supported by ShyWiki in comparison to the Easy-

Table I
EVALUATION OF SHYWIKI FOR RE

	EasyWinWin tool	WikiWinWin	*ShyWiki*
Support of WinWin approach	***	***	***
Easy exchange of ideas and knowledge	**	***	***
Easy to update and preserve the revision history	*	***	***
Easy to extend	**	***	***
Easy to incorporate boundary objects	**	***	***
Support synchronous collaboration	***	*	*
Release and baseline requirements	***	***	***
Automation of content management	***	**	**
Exporting the content	***	**	***
Collaborative interpretation	***	*	***
Support to structured information	*	*	***
Hypermedia support		***	***

WinWin tool [18] and the WikiWinWin and also discusses some of ShyWikis properties.

Table I shows the comparison. The evaluated points (with exception of the last three) and rating scale used in Table I were proposed by Yang *et.al* [26] for comparing Wikis, WikiWinWin and the EasyWinWin tool. The rating scale is the following: (.) not supported, (*) marginal, (**) acceptable, and (***) excellent. In addition, the values assigned to EasyWinWin and WikiWinWin are the result of the evaluation performed in [26].

ShyWiki supports the EasyWinWin process well, and provides easy ways to exchange ideas and knowledge among the stakeholders in the project. The version module of ShyWiki preserves the history of the requirements, and brainstorming sessions. ShyWiki can be extended by means of the template mechanism. Users can add new templates and alter the structure of the existent ones. Boundary objects can be added by means of new wiki pages, and hyperlinks. ShyWiki does not support synchronous collaboration. However, we have an alpha version of the synchronous collaborative edition which is based on HTTP Streaming technology.

ShyWiki templates can partially provide consistency to the structure of the wiki. However, ShyWiki does not have a mechanism to enforce the consistency of the whole wiki. To support release and baseline requirements, a datetime parameter is added to the hyperlinks, and it is used to get the version of the target wiki page that was alive in that moment of time. The information stored in the wiki can be read by other applications by using the RDF projections of the wiki pages as described in [39].

ShyWiki can implement lightweight knowledge sharing workspaces for architecture knowledge management [49]. ShyWiki facilitates the annotation of architectures, provides project activity awareness through transclusion and implicit spatial semantics, and gives decisions support using brainstorming. In this way, it is possible to trace the requirements with the architecture, and with other software engineering artifacts using the shared information repository that permits distributed access, and asynchronous collaboration.

The use of cards for user stories, and the definition of tasks is essential in many agile methods. When a project is distributed, teams need tools to support iteration planning and management. Wikis have been used successfully in agile planning. However, it has been observed that they do not provide spatial features which are needed to organize post-it notes in a similar way to a collocated meeting [50]. In ShyWiki, teams can share a single blackboard or each team can have its own one depending on the distribution of the development tasks and the skills needed to solve them.

ShyWiki provides a board that can be used for planning and managing software tasks which is another issue that has to be solved by global software development tools. For example, it can be used in agile development methods such as Scrum [51] or Extreme Programming [52] for performing a distributed *planning game*. Experiences in implementing agile practices in a distributed setting have shown that an essential factor is to have tools that allow a team to share and collaborate using a board among distributed teams [53], [54].

VIII. CONCLUSIONS AND FUTURE WORK

The requirements elicitation process can take advantage of wikis because they provide support for the open collaboration among stakeholders, and low entry barriers. Wikis facilitate the collaborative exchange of ideas, information, and can trace changes by means of versioning.

ShyWiki provides support for distributed and collaborative requirements elicitation process based on the KJ method and EasyWinWin. ShyWiki allows the stakeholders to manipulate spatially the requirements. They can group, relate or merge them easily. In this way, the negotiation and prioritization process can be done through the wiki pages which are virtual blackboards that hold hypertext card notes. ShyWiki also provides stakeholders with a virtual environment that supports the emergence of ideas. This factor is relevant for the definition of innovative requirements, and for facilitating the transfer of tacit knowledge to the requirements definitions.

ShyWiki is an *Integrated Collaborative* tool for global software development. It provides features for viewing

and discussing software artifacts, as well as collaboration through the virtual boards. According to Laredo and Rajan [7]: "tools should fill the vacuum left by traditional conference room and the white board and get the team to collaborate in a given context". ShyWiki can be this kind of tool.

We are currently performing several experiments to validate the usability of ShyWiki in requirements engineering and software architecture design. We are also extending some of ShyWikis groupware support such as providing synchronous communication. In addition, we are working on using ShyWiki in global software development projects.

ACKNOWLEDGMENTS

This work has been funded by Science Foundation Ireland grant 03/CE2/I303_1 to LERO - the Irish Software Engineering Research Centre (http://www.lero.ie).

REFERENCES

[1] J. D. Herbsleb and D. Moitra, "Global software development," *IEEE Software*, vol. 18, no. 2, 2001.

[2] J. D. Herbsleb, "Global software engineering: The future of socio-technical coordination," in *Workshop on the Future of Software Engineering, FOSE 2007*, 2007, pp. 188–198.

[3] H. Hofmann and F. Lehner, "Requirements engineering as a success factor in software projects," *Software, IEEE*, vol. 18, no. 4, pp. 58–66, Jul/Aug 2001.

[4] I. Sommerville and P. Sawyer, *Requirements Engineering: A Good Practice Guide*. John Wiley & Sons, June 2004.

[5] S. Robertson, "Requirements trawling: techniques for discovering requirements," *International Journal of Human-Computer Studies*, vol. 55, no. 4, pp. 405–421, 2001.

[6] A. Davis, O. Dieste, A. Hickey, N. Juristo, and A. Moreno, "Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review," in *, 14th IEEE International Conference of Requirements Engineering*, 2006, pp. 179–188.

[7] J. A. Laredo and R. Ranjan, "Continuous improvement through iterative development in a multi-geography," in *ICGSE '08: Proceedings of the 2008 IEEE International Conference on Global Software Engineering*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 232–236.

[8] B. Decker, E. Ras, J. Rech, P. Jaubert, and M. Rieth, "Wiki-based stakeholder participation in requirements engineering," *IEEE Software*, vol. 24, no. 2, pp. 28–35, 2007.

[9] B. Leuf and W. Cunningham, *The Wiki way: quick collaboration on the Web*. Boston, USA: Addison-Wesley Longman, 2001.

[10] W. Cunningham, "Design principles of wiki: how can so little do so much?" in *WikiSym '06: Proceedings of the 2006 international symposium on Wikis*. New York, NY, USA: ACM, 2006, pp. 13–14.

[11] "Wikipedia, the free encyclopedia," http://www.wikipedia.org.

[12] K. Al-Asmari and L. Yu, "Experiences in distributed software development with wiki," in *Software Engineering Research and Practice*. CSREA Press, 2006, pp. 389–293.

[13] P. Louridas, "Using wikis in software development," *Software, IEEE*, vol. 23, no. 2, pp. 88–91, March-April 2006.

[14] P. Laurent and J. Cleland-Huang, "Lessons learned from open source projects for facilitating online requirements processes," in *REFSQ '09: Proceedings of the 15th International Working Conference on Requirements Engineering: Foundation for Software Quality*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 240–255.

[15] D. Cox and S. Greenberg, "Supporting collaborative interpretation in distributed groupware," in *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*. New York, NY, USA: ACM, 2000, pp. 289–298.

[16] C. C. Marshall and F. M. Shipman, "Spatial hypertext: designing for change," *Communications of ACM*, vol. 38, no. 8, pp. 88–97, 1995.

[17] J. Kawakita, *The original KJ-method*. Kawakita Research Institute, 1982.

[18] P. Gruenbacher, "Collaborative requirements negotiation with easywinwin," in *DEXA '00: Proceedings of the 11th International Workshop on Database and Expert Systems Applications*. Washington, DC, USA: IEEE Computer Society, 2000, p. 954.

[19] B. Boehm and H. In, "Identifying quality-requirement conflicts," *IEEE Software*, vol. 13, no. 2, pp. 25–35, 1996.

[20] D. C. Engelbart, "Authorship provisions in augment (reprint)," *Computer-supported cooperative work: a book of readings*, pp. 107–126, 1988.

[21] M. Bernstein, P. J. Brown, M. Frisse, R. Glushko, P. Zellweger, and G. Landow, "Structure, navigation, and hypertext: the status of the navigation problem," in *HYPERTEXT '91: Proceedings of the third annual ACM conference on Hypertext*. New York, NY, USA: ACM, 1991, pp. 363–366.

[22] H. J. Strauss, "Hypermaps: telling your users where to go," in *SIGUCCS '90: Proceedings of the 18th annual ACM SIGUCCS conference on User services*. New York, NY, USA: ACM, 1990, pp. 377–390.

[23] C. C. Marshall and F. M. Shipman, "Searching for the missing link: Discovering implicit structure in spatial hypertext," in *Hypertext'93: Proceedings of the fifth ACM conference on Hypertext*. ACM, 1993, pp. 217–230.

[24] B. W. Boehm and R. Ross, "Theory-w software project management principles and examples," *IEEE Trans. Softw. Eng.*, vol. 15, no. 7, pp. 902–916, 1989.

[25] R. Briggs and P. Gruenbacher, "Easywinwin: managing complexity in requirements negotiation with gss," in *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, 2002.

[26] D. Yang, D. Wu, S. Koolmanojwong, A. W. Brown, and B. W. Boehm, "Wikiwinwin: A wiki based system for collaborative requirements negotiation," in *Proceedings of the Annual Hawaii International Conference on System Sciences*, 2008.

[27] M. Polanyi, *The Tacit Dimmension*. Anchor Books, 1967.

[28] I. Nonaka and H. Takeuchi, *The Knowledge-Creating Company*. New York: Oxford University Press, 1995.

[29] L. D. Alavi, M., "Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues," *MIS Quarterly: Management Information Systems*, vol. 25, no. 1, pp. 107–136, 2001.

[30] N. Maiden, A. Gizikis, and S. Robertson, "Provoking creativity: Imagine what your requirements could be like," *IEEE Software*, vol. 21, pp. 68–75, 2004.

[31] S. Ya and T. Rui, "The influence of stakeholders on technology innovation: A case study from china," in *Management of Innovation and Technology, 2006 IEEE International Conference on*, vol. 1, June 2006, pp. 295–299.

[32] N. Maiden, S. Manning, S. Robertson, and J. Greenwood, "Integrating creativity workshops into structured requirements processes," in *DIS '04: Proceedings of the 5th conference on Designing interactive systems*. New York, NY, USA: ACM, 2004, pp. 113–122.

[33] L. Mich, C. Anesi, and D. Berry, "Applying a pragmatics-based creativity-fostering technique to requirements elicitation," *Requirements Engineering*, vol. 10, no. 4, pp. 262–75, 2005.

[34] I. Nonaka, K. Umemoto, and D. Senoo, "From information processing to knowledge creation: A paradigm shift in business management," *Technology in Society*, vol. 18, no. 2, pp. 203–218, 1996.

[35] A. Delbecq and A. Van de Ven, "A group process model for problem identification and program planning," *Journal of Applied Behavioral Science*, vol. 7, no. 4, pp. 466–492, 1971.

[36] T. Shigenobu, T. Yoshino, and J. Munemori, "Evaluation and application of creativity collaboration support system gungen dx ii for consensus-building among users," *International Journal of Information Technology and Decision Making*, vol. 6, no. 3, pp. 475–490, 2007.

[37] C. Solis and N. Ali, "ShyWiki-a spatial hypertext wiki," in *WikiSym '08: Proceedings of the 2008 international symposium on Wikis*. New York, NY, USA: ACM, 2008.

[38] C. Solis and N. Ali, "ShyWiki-a spatial hypertext wiki prototype (demo)," in *WikiSym '08: Proceedings of the 2008 international symposium on Wikis*. New York, NY, USA: ACM, 2008.

[39] C. Solis and N. Ali, "A spatial hypertext wiki for knowledge management," in *IEEE 2010 International Symposium on Collaborative Technologies and Systems (CTS 2010)*. IEEE Computer Society, 2010.

[40] A. R. Dennis, J. S. Valacich, T. A. Carte, M. J. Garfield, B. J. Haley, and J. E. Aronson, "Research report: The effectiveness of multiple dialogues in electronic brainstorming." *Information Systems Research*, vol. 8, no. 2, pp. 203 –, 1997.

[41] IBM, "Rational RequisitePro," http://www-01.ibm.com/software/awdtools/reqpro/ last accesed on February 2010.

[42] IBM_, "Rational Doors," http://www-01.ibm.com/software/awdtools/doors/ last accesed on February 2010.

[43] Borland, "CaliberRM," http://www.borland.com/us/products/caliber/index.html last accesed on February 2010.

[44] V. Sinha, B. Sengupta, and S. Chandra, "Enabling collaboration in distributed requirements management," *IEEE Softw.*, vol. 23, no. 5, pp. 52–61, 2006.

[45] E. Knauss, O. Brill, I. Kitzmann, and T. Flohr, "Smartwiki: Support for high-quality requirements engineering in a collaborative setting," in *2009 ICSE Workshop on Wikis for Software Engineering (Wikis4SE 2009)*, 2009.

[46] T. Riechert and T. Berger, "Leveraging semantic data wikis for distributed requirements elicitation," in *2009 ICSE Workshop on Wikis for Software Engineering (Wikis4SE 2009)*, 2009, pp. 7 – 13.

[47] M. Hepp, D. Bachlechner, and K. Siorpaes, "Ontowiki: community-driven ontology engineering and ontology usage based on wikis," in *WikiSym '06: Proceedings of the 2006 international symposium on Wikis*. New York, NY, USA: ACM, 2006, pp. 143–144.

[48] D. Ferreira and A. Silva, "An enhanced wiki for requirements engineering," in *Proceedings of the 2009 35th Euromicro Conference on Software Engineering and Advanced Applications (SEAA 2009)*, Piscataway, NJ, USA, 2009, pp. 87–94.

[49] C. Solis, N. Ali, and A. Babar, "A spatial hypertext wiki for architectural knowledge management," in *International Workshop on Wikis for Software Engineering*. ACM/IEEE, 2009.

[50] M. J. Rees, "A feasible user story tool for agile software development?" *Asia-Pacific Software Engineering Conference*, vol. 0, p. 22, 2002.

[51] K. Schwaber and M. Beedle, *Agile Software Development with Scrum*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[52] K. Beck and C. Andres, *Extreme Programming Explained: Embrace Change (2nd Edition)*. Addison-Wesley Professional, 2004.

[53] H. Smits and G. Pshigoda, "Implementing scrum in a distributed software development organization," Piscataway, NJ, USA, 2007, pp. 349 – 53.

[54] S. Berczuk, "Back to basics: The role of agile principles in success with an distributed scrum team," in *AGILE '07: Proceedings of the AGILE 2007*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 382–388.