A Semantic Wiki for Quality Management in Software Development Projects

Roberto García, Rosa Gil, Juan Manuel Gimeno, Toni Granollers,
Juan Miguel López, Marta Oliva, Afra Pascual
Universitat de Lleida
Jaume II, 69
25001 Lleida, Spain

Abstract.
Quality Management has become a strategic issue for organisations and very valuable to produce quality software. However, Quality Management Systems (QMS) are not easy to implement and maintain. Our experience shows the benefits of developing a QMS by first formalising it using Semantic Web ontologies and then putting them into practice through a semantic wiki. The QMS ontology that has been developed captures the core concepts of a traditional QMS and combines them with concepts coming from the MPIu+a development process model, which is geared towards obtaining usable and accessible software products. Then, the ontology semantics is directly put into play by a semantics-aware tool, the Semantic MediaWiki. The developed QMS tool has been running for two years at the GRIHO research group, where it has been used to manage almost 50 software development projects taking into account quality management issues. It has been also externally audited by a quality certification organisation. Its users are very satisfied with their daily work with the tool, which manages all the documents created during project development and also allows them to collaborate thanks to wiki features.

1 Introduction

This paper explores the application of a semantic wiki for the implementation of a Quality Management System (QMS). This system is used to improve the quality of the technology transfer projects developed by the GRIHO research group, mainly projects involving the development of interactive software systems. One of GRIHO's main research lines is Human-Computer Interaction, thus one of the main dimensions for quality measurement are the usability and accessibility of this interactive systems.

Consequently, the quality management system that has been developed implements ISO9001:2008 combined with a software development methodology that focuses on ensuring the usability and accessibility of the resulting product, concretely the MPIu+a methodology detailed in Section 2.2.

The implementation is based on a semantic wiki that supports the software development process but also other organisation processes such as marketing, purchasing, training, user satisfaction, etc. It also manages personnel enrolment, document management (internal reports, requirements documents, minutes...) and captures other kinds of data that can be then used to generate a quality management dashboard [1].

A deployment of Semantic Media Wiki, with some wiki extensions and an ontology that models the quality management domain, is currently being used at the GRIHO research group on a daily basis as the way to

manage its technology transfer projects and, due to its usefulness, also its research projects. It has become the reference tool for GRIHO projects.

Our experience with the semantic wiki QMS is that it provides outstanding capabilities for document management. This is facilitated by the underlying semantic management functionalities that profit from a shared conceptualisation of the domain based on ontologies. This constitutes the main added value in comparison with non-semantics-capable solutions. The ontologies facilitate maintaining the system information architecture and even guiding users while they interact with the QMS. More details about quality management, MPIu+a and related work are available in Section 2. The paper continues presenting the proposed approach in Section 3 together with the main results. Then, there is an evaluation in Section 4 and finally the conclusions and the future work are presented in Section 0.

2 State of the Art

This section contextualises the contribution by introducing the term Quality Management System (QMS) and by describing the development process model to be integrated into the QMS: MPIu+a. Then, existing tools and initiatives related to quality management, software development and the Semantic Web are presented.

2.1 Quality Management System

A Quality Management System (QMS) is built from the set of collective policies, plans, practices, and the supporting infrastructure by which an organization aims to reduce and eventually eliminate non-conformance to specifications, standards, and customer expectations in the most cost effective and efficient manner [2]. Organisation processes are the basis for the definition of a quality system. Correct identification and management of these processes will facilitate the implementation of the quality management system. ISO 9001:2008 defines a process as any activity that uses and manages resources to enable the transformation of inputs into outputs.

2.2 MPIu+a Development Process Model

From the standpoint of developing interactive systems, we can find several approaches that incorporate User-Centred Design (UCD) as a procedural methodology for Usability Engineering (UE). Digital

Equipment Corporation professionals used this term to refer to concepts and techniques to plan, achieve and verify the system's usability objectives.

The main idea lies in defining the "measurable usability objectives" as soon as possible. They will be repeatedly evaluated during the development phase to ensure its achievement [3,4]. Among the various existing UE proposals, e.g. [5,6], we will focus our development on the Usability and Accessibility Engineering Process Model (MPIu+a), which is described in [7] and outlined in Fig. 1.

[Include Fig. 1 here]

Fig. 1: Usability and Accessibility Engineering Process Model (MPIu+a).

Its main features are:

- **Conceptual Organization**: Human-Computer Interaction (HCI) activities are gathered together in a specific part of the process model so development team members can easily recognise what activities are being carried out and which ones are related to them. This conceptual organization was born from the need to organise and coordinate the HCI and Software Engineering parts.
- **Main blocks**: the process model is organised in three main blocs, shown as columns in Fig. 1., which correspond to three different kinds of activities:
    - A "traditional" Software Engineering lifecycle in the left column. Actually, any software engineering model can be used here as long as it is incremental and iterative.
    - Prototyping, in the central column. Production of prototypes is a central activity when developing interactive systems following any UCD methodology. They are intended for evaluation activities and developed during the whole software engineering lifecycle till the implementation is completed.
    - Evaluation, in the right column. Again, UCD design methodologies need a constant use and interaction evaluation. This phase of the model covers all the evaluation techniques that focus on assessing the quality of use of the related system.

- **User**: any UCD process should focus and encourage user's active participation. This aspect is clearly reflected in the scheme: the user is located in the top-central part and networked with all other stages, enabling its constant participation.
- **Iterativity**: repeatable processes are a natural characteristic of any field of engineering. The engineering process model provides a set of arrows with the following goals: (i) providing this constant-iteration and, (ii) the constant and active user's participation in all the development phases (from the requirements analysis stepping to the design and implementation of prototypes and/or further evaluation).
- **Simplicity**: The scheme itself is brief and simple, with few branches and nodes and conditional paths that complicate its comprehensions and implementation.
- **Multidisciplinary**: UCD and UE require multidisciplinary development teams that feature many different mental models interacting altogether. They require specific techniques and tools that facilitate the communication between those involved in the development [8].
- **Flexibility**: The model has not only one single way and neither has restrictive and conditional branching points to lead the development team actions and/or decisions. Then it guides the development providing total flexibility for its implementation.

2.3 Related Work

As software development processes become more complex, it is necessary to provide computer-based tools to support software engineers to perform their tasks. These tools must work together to provide effectiveness in software development process. Integration demands consistent representations of software engineering information, standardized interfaces between tools, homogeneous means of communication between software engineers and tools, and an effective approach that enables software engineering environments to move among various platforms [9].

Software systems relational navigation based on semantic representations goes back to the early 1990s and the LaSSIE knowledge base [10]. In this sense, the emerging field of semantic web technologies provides a new stimulus for Software Engineering research [11]. [12] shows how changes to metadata can be recorded and tracked and propose how they may be used to proactively notify developers of changing requirements and quality measurements that may impact maintenance.

Nevertheless, it must also be taken into account that, in some cases, evolving Semantic Web technologies may not be adequate for all software modelling needs [13]. Besides Semantic Web, ontology languages define a set of representational primitives with which a knowledge domain is modelled. The main purpose of the Semantic Web and ontologies is to integrate heterogeneous data and enable interoperability among disparate systems. Ontologies have been used to model software engineering knowledge by denoting the artefacts that are designed or produced during the engineering process [14]. There is also an ontology that models quality for enterprise modelling [15]. However, this is a very abstract ontology geared towards considering quality while formally modelling an enterprise. It is too abstract for modelling a QMS but might be used as a foundation when modelling more concrete and functional ontologies for QMS.

Apart from their use in software development processes, the evolution of the World Wide Web has made Semantic Web techniques especially useful for meeting the requirements of Web 2.0 web applications, such as social networks. [16] proposes a framework for the association of semantic data to webpage links based on a specific domain ontology, additionally allowing the user to express his opinion regarding his emotions about the content of the link. These data are further exploited to suggest additional links to the user, based on the semantic metadata and the level of user satisfaction with previously viewed content.

The globalisation of technologies has led to a scenario where the possibilities for reuse and transfer of software engineering products are multiplied. In the context of Web 2.0 or social web, the use of Semantic Web represents a revolution in information access and storage that can enable knowledge management and knowledge sharing among developers and organisations.

In this sense, new methodologies and tools have been created to provide mechanisms for development teams to manage knowledge and include semantics in the software development process. Self-organized reuse of artefacts from software and system development using the enhancement of Wiki content with ontologies can solve problems related with the chaotic growth of content in social environments [17].

[18] shows an approach that applies concepts from the areas of social software and Semantic Web to application development, though it is intended just for ontology development. Social Global Repository is a tool that incorporates the established methodological recommendations of Software Engineering in an environment that provides support, flexibility and up-to-date solutions to software development teams [19].

Furthermore, [20] presents an environment that enables semantic and social interaction with the documentation generated during software development. However, all these initiatives do apply semantic technologies to software development but do not take into account quality management.

2.3.1 Quality Management software

As quality management needs data to be gathered from different areas in the organization, it is almost compulsory to use some kind of software to assist in its storage, cataloguing and analysis. That's why ERPs (Enterprise Resource Planners), the software that integrates all business processes into a coherent unit, contain modules that assist in the organisation Quality Management activities. The modules most related to quality management are:

- SCM modules (Supply Chain Management), process requirements for suppliers, tracks supply chain transactions and providers of the organisation allowing the future traceability of the manufactured goods.
- Product Lifecycle Management (PLM) is the process of managing the entire lifecycle of a product from its conception, through design and manufacture, to service and disposal. For software products, as will be explained below, this is where most development tools belong.

However, complexity is the main limitation of QMS solutions based on ERPs, which is derived from the fact that ERPs are intended for much more than quality management. Consequently, this might be a feasible solution for big to midsize organisation but not so much for the small ones.

Finally, we can also consider Computer-Aided Software Engineering (CASE) tools. They definitively facilitate quality management, but just during a small part of the software development organisation operation. A full QMS system is required in order to bring quality management to all processes. In fact, as our approach detailed in the next section shows, QMS and Software Development tools can work in cooperation. While the QMS provides the general framework for quality management the Software Development tools can be used to assist the software development subprocesses.

3 Approach

The proposed approach is based on Semantic Web and wiki technologies. The Semantic Web technologies make it possible to implement the QMS tool on top of the formal conceptualisation provided by an

ontology of the QMS used at GRIHO. The ontology has been developed using Methontology [21]. In this methodology, the ontology is first conceptualised using different tools. In our case UML was used because this was a language the development team was familiarised with. Next, the conceptualisation is implemented by means of Semantic Web technologies. The resulting ontology is available online [22] and it is detailed in Section 3.1.

The Wiki provides a collaboration framework that combined with Semantic Web technologies, as in the case of Semantic Wikis, makes it possible to put into practice the semantics captured by the ontology. This way, the wiki benefits from a more structured information architecture that guides users while they interact with the wiki. More details about how the QMS wiki has been implemented are available from Section 3.2.

3.1 Quality Management Ontology

The ontology models a quality management system equivalent to the ISO 9001 standard, concretely the one used as the foundation for the QM software implemented and used in the GRIHO research group. It is important to note that the QMS used as the source for the ontology is intended to be implemented using paper documents. Consequently, the concepts in the ontology have names such as Document or Form.

We have also kept the identifiers used in the original paper-based QMS and linked them to resources through the Dublin Core [23] "identifier" property. This way, we can keep track of where each concept originates from and facilitate things for external auditors, who in our case are trained in auditing QMSs based on documents that use those identifiers.

We have also kept the original QMS conceptualisation and therefore, as it is shown in Fig. 2, the main classes are: Document, Process and Form. A Form is a kind of Document that is intended to be filled. It also defines entities like Project, Person or Client. These classes have the required properties to keep track about who is working in each project, which clients are financing them and who are the contact persons.

Each filled form constitutes and instance of one of these entities. A Document is another entity, which represents a resource that captures and carries information. A special kind of Form is Indicator. These are values derived from different sources that provide direct clues about how the organization is behaving and can be used to build up a decision support system.

[Include Fig. 2 here]

Fig. 2. Diagram of the core QMS Ontology concepts (subclass of relation denoted by white arrowheads and properties denoted by black arrowhead with domain the arrow source and range the arrow target)

Finally, there are the Processes, which are the main entities in any QMS. They model the dynamics of any enterprise or organisation. They are related to the Documents and Forms that are used in order to capture and share information associated to instances of the process. These Documents and Forms are processed, created or consumed by Processes. The ontology captures, as it is shown in Fig. 3, how all Documents and Forms are related to Processes.

This is modelled using the "part" relation, which associates each Process to all the Documents and Forms that are part of its operation. The ontology also models the specific Indicators being used in the GRIHO QMS, basically year-by-year Indicators, and the source Processes, Documents and Forms from which they derive their values.

[Include Fig. 3 here]

Fig 3. Relations among Indicators (left), Processes (centre) and Documents and Forms (right)

In addition to this conceptualisation based on the original QMS terms, we have taken advantage of the capabilities of ontologies in order to enrich the model. Besides Dublin Core, we have also reused the Semantic Web for Research Communities (SWRC) ontology [24]. This ontology provides many basic concepts that are extended in our ontology. From SWRC we reuse many classes related to project development in the research context like Project Event, Organization; and properties such as "develops", "participant", etc.

We have also detailed further, starting from the QMS Ontology so far, the part dealing with the project development process. As it is detailed in the next section, we have incorporated concepts from the MPIu+a development process model into the development process and the documents and forms that are part of it.

3.1.1 Development Process Model

This is the main Process of GRIHO as it is the one in which software is developed. The QMS specifies that it is related to four Forms:

- Project: represents this entity and provides details about it. It is also used in order to state the staff participating in the development, the client, etc.

- Project Meeting Minutes: the minutes of all meeting with the client and other stakeholders during the development of the project. They include the topics covered, outcomes, participants, etc.
- Project Partial Documentation: these are different forms that document the project at different stages in its development
- Project Closing Approval: this is the document closing the project development as agreed by both parts.

In order to guide developers through a development process following the MPIu+a process model, the steps defined by MPIu+a, detailed in Section 2.2, are considered. They are not explicitly modelled in the ontology but they can be related to different parts of the forms associated to the development process to avoid constraining too much the development process. For instance, to formally model each of the steps to be followed when capturing requirements results in a too constraining and inflexible system. We have adopted a more flexible approach based on guidelines that are provided to developers when performing one of these subprocesses. For instance, a guideline for requirements gathering combined with templates for requirements modelling. These guidelines and templates are implemented using the wiki mechanism, as detailed in the next section.

3.2 Semantic Media Wiki Implementation

The Wikipedia describes a wiki as "a website that allows the creation and collaborative editing of any number of interlinked web pages via a web browser using a simplified mark-up language or a WYSIWYG text editor". The edition process is almost unconstrained by the wiki system so the resulting websites are much more flexible than those based on a CMS (Content Management Systems).

However, this greater flexibility might lead to more chaotic information architectures as different users add content without any kind of underlying guidelines. These guidelines might be informal but, in order to improve usability, it is desirable to have more formal ones that can be implemented into the wiki mechanisms.

This is why we have chosen a semantic wiki [25], i.e. a wiki that has an underlying model of the knowledge described in its pages. Regular, or syntactic, wikis have structured text and untyped hyperlinks. Semantic wikis, on the other hand, provide the ability to capture or identify information about the data within pages,

and the relationships between pages, in ways that can be queried or exported like a database. These models are based on ontologies, in our case the QMS Ontology that captures part of the knowledge related with how the QMS works and that is used by the wiki in order to guide users when working with it.

Semantic MediaWiki [26], an extension of MediaWiki [27], is the best-known semantic wiki software. We have chosen it because it has been extensively tested in many different scenarios and it has a big and very active community around it. This community develops and maintains many plug-ins that extend the functionality of both MediaWiki and Semantic MediaWiki. We have considered so far the following extensions [28]:

- **Semantic MediaWiki**: this is the semantic extension of the base MediaWiki installation. This extension makes it possible to implement semantics-aware mechanisms on top of the MediaWiki. Ontology classes are modelled using wiki categories, class instantiation using page categorisation, subclasses using category subcategorisation, instances as articles, instance datatype properties (attributes) as attribute annotations and instantiated object properties (relations) as typed links. Moreover, it also allows importing ontologies, so all the categories, subcategorisation, attributes, etc. stuff is automatically generated from the ontology. Finally, this extension also allows to pose semantic queries, such as the one marked with the "#ask" command in Table 1. These queries benefit from the semantics captured in the corresponding ontologies, though reasoning is constrained to the previous features (basically subclassing, instantiation and domain and range properties).
- **Semantic Forms**: this extension makes editing forms semantics-aware. This way, the forms used to create or edit an article or category are built taking the underlying ontologies into account and guide users while they fill them. For instance, Fig. 4 shows the semantic form generated for editing the "Project" form. Thanks to the semantics of the underlying ontology, it is possible to assemble the form from the properties associated to the corresponding class. It is also possible to assist the user when filling the form as each property specifies the kind of things it links to. For instance, the "head" property values are just "Person" so the forms assist the user showing instances of "Person" whose label contains the text typed so far.

[Include Fig.4 here]
Fig 4. Example of a Semantic Form providing assistance while filling an input with Person values

- **Loop Functions**: this extension provides limited looping functionality that is necessary to implement some templates and indicators. For instance, the "YearlyTurnover" indicator is implemented using a loop over all the projects closed during a certain year and summing up their individual turnovers.
- **Parser Functions**: this extension provides some logical functions that are used, as in the previous case, to implement the templates that are preloaded into articles depending on their category. For instance, when creating a new project, the Project template is preloaded as the wiki text of the corresponding wiki article. Part of this preload is shown in Table 1. It uses basic wiki text but also parser functions in order to build a starting project page that contains the guidelines that will help developers follow the intended process development tasks. For instance, it includes forms to add project participants or links to the documents to be created during project development, which in turn include the necessary preloads to add the guidelines for filling that particular document.

Table 1. Template based on Semantic Wiki text preloaded into all new Project instances

```
{|
== Participants ==
{{#formlink:ParticipacioTreballa|Afegir
Participant|button|ParticipacioTreballa[projecte]={{PAGENAME}}}}

{{#ask:
  [[Categoria:ParticipacioTreballa]] [[projecte::{{PAGENAME}}]]
  | intro=Participants al projecte
  | ?participant | ?inici | ?fi
  | sort=inici
  | format=timeline|timelinebands=WEEK,MONTH|timelinesize=200px|timelineposition=start
  | default=Sense participants
}}

== Documents ==
* {{#ifexist: {{PAGENAME}}/SituacioPartida|
  [[{{PAGENAME}}/SituacioPartida|Situació de partida]]|
  {{#formlink:Document/{{PAGENAME}}/SituacioPartida|NO EXISTEIX LA Situació de
partida|Document[codiIMP]=IMP028&Document[data]={{CURRENTYEAR}}-
{{CURRENTMONTH}}-
{{CURRENTDAY2}}&preload=Plantilla:Impres/Projecte/SituacioPartida/preload}}
}}
...
```

- **Timelines**: the semantic queries included in Semantic MediaWiki can show results in different formats (table, list, RSS, calendar...). It is also possible to show results into a timeline, when querying for resources that have attributes with time values. For instance, this feature is used when showing the list of tasks identified during project development, so it is easy to generate a graphical view of the project work plan.
- **PermissionACL**: this extension allows defining permissions based on access control lists (ACLs) which have been then combined with user roles: developers, managers, accountants and administrators. This made possible to lock the wiki to a closed group of users and solved the security issues inherent to the openness of wiki engines [29].

Finally, the basic features of MediaWiki have also facilitated the implementation of the QMS. The Document and Record Management processes have been completely automatised thanks to the wiki mechanisms. Everything gets stored in the wiki, both documents and records, which are the result of filling a form. Moreover, it is possible to track every change, who and when it was perform, and even revert or highlight these changes using the version control mechanism that MediaWiki integrates.

4 Evaluation

The QMS tool has been running now for more than two years and it has 15 different registered users, who have used it to manage 47 projects resulting in more than 1700 wiki pages. From them, 530 are wiki documents, i.e. ignoring discussion pages, redirections, etc. The wiki is also used as a repository where different kinds of documents (PDFs, images, etc.) are uploaded and annotated. Right now, 470 files have been uploaded. From the point of view of raw usage, the wiki statistics show that there have been 39472 page views and 11511 page editions. If we go into semantic web features statistics, there are more than 750 form instances currently managed by the QMS wiki. They are semantically annotated with more than 6750 property values using 88 different properties.

Apart from the previous quantitative data, we have conducted interviews with the involved personnel. These were open interviews, based on a basic questionnaire composed by qualitative questions about the main benefits and inconveniences observed. The interviews show that all users perceive the system as a great benefit in comparison with the paper-based QMS that was running for some months before the first

version of the wiki-based QMS was deployed. They perceive benefits beyond the clear one of having a central point where all the documentation is kept. In fact, this one was already available because the paper-based QMS was implemented with a shared folder where digital versions of the documents where stored.

The main benefit that users perceive is that they can work collaboratively in these documents, sharing their changes almost in real-time and with an integrated document control version that keeps track of all changes, with information about who and when performed them. Project managers also find watch-lists very useful. They allow project managers to be notified by e-mail of all editions that other users perform on every watched document.

The guidelines and assistance provided by the system when filling forms and documents are another clearly observed benefit. Form filling is facilitated by the underlying semantic model and Semantic Forms, while document guidelines avoid the burden of facing users with blank documents. New documents are preloaded with guidelines and wiki functions that also encourage using pre-established methodologies and templates, in our case the MPIu+a process model.

Observed drawbacks are mainly related to the limitations of a wiki as a document editor. The quality of these documents is not as good as what can be obtained with non-HTML editors. For instance, diagrams have to be edited with external tools and exported as images that are later embedded into wiki documents. In any case, we are exploring the possibility of integrating extensions that can apply more sophisticated templates and generate, for instance, PDF versions of wiki documents.

Another drawback is derived from the complexity of the wiki text that is directly shown to the user. In our case, it is even higher due to the presence of semantic expressions and wiki parser extensions. Fortunately, developers rarely should change these parts of wiki documents, as they are already preloaded. Finally, it is also important to note that the QMS has been audited and certified by the quality certification company hired by the Catalan technology transfer network GRIHO is a member of.

5 Conclusions and Future Work

Our experience shows the benefits of a novel approach to QMS development based on a Semantic Web ontology that is then put into practice through a semantic wiki. To our knowledge, this is the first initiative in this line, which simplifies the development of a QMS for organisations that do not require a whole ERP.

The contributed QMS ontology captures the core concepts of a traditional QMS and enriches them by reusing other existing ontologies for resource description or project management. Moreover, it is possible to combine the conceptualisation of the organisation processes from the quality management point of view with concepts coming from the development process-modelling domain, in our case with the MPIu+a process model.

It is then possible to put QMS Ontology semantics into practice when combined with semantic-aware tools. In our case, the Semantic MediaWiki is capable of exploiting the ontology in order to generate the whole wiki structure. Moreover, when combined with extensions like SemanticForms, it is also possible to use the ontology in order to guide users while editing wiki articles. These mechanisms are complemented with semantic queries and wiki templates, which define preloads for new documents and provide guidelines for users. These templates and guidelines help developers to follow the MPIu+a process model and avoid them the burden of editing the complex wiki text expressions responsible for computing features such as quality indicators.

The developed QMS has been running for two years at the GRIHO research group, where it has managed almost 50 software development projects. Its users are very satisfied with their daily work with the tool, which manages all the documents created during project development. It also allows them to collaborate thanks to wiki features. It is important to note that external repositories for source code management are used, e.g. Subversion, which are much more capable for this specific task.

Users have also detected some problems, mainly derived from the characteristics of wiki tools. First of all, the tool has shown some limitations when trying to produce high quality documents to be sent to clients, e.g. partial project documentation. We are currently exploring extensions that provide more sophisticated template mechanisms. The other main drawback is that it is not easy to edit complex documents using wiki text. We have found it complicated to integrate WYSIWYG editors because they are not prepared for complex wiki text expressions required for semantic annotations on complex parser functions.

Finally, there are some security issues to be considered, which we have solved by restricting access to the wiki just to the development team. However, in some cases, it is desirable to open certain parts of the system to other stakeholders, especially clients. Consequently, longer-term work focuses on exploring other

alternatives, also based on semantic web technologies that have widely shown their usefulness. Our first candidates are version 7 of the Drupal CMS [30], which incorporates semantic capabilities into its core, and specialised semantic web tools like the Rhizomer platform [31] for semantic data publishing and management.

Acknowledgements

The work described in this paper has been partially supported by Spanish Ministry of Science and Innovation through the Open Platform for Multichannel Content Distribution Management (OMediaDis) research project (TIN2008-06228).

References

[1] Buytendijk, F.: 'Quality Control: How To Respond To 'Just Give Me a Dashboard!"', Business Performance Management, September 2005. http://bpmmag.net/mag/bpm_article_nwarcarticle_14490, accessed June 2010

[2] Foster T.: 'Managing Quality' (Prentice Hall, 2009, 4th edn.)

[3] Rosson, M., and Carroll, J.: 'Usability engineering: scenario-based development of human-computer interaction' (Morgan-Kaufmann, 2002)

[4] Good, M., Spine, T.M., Whiteside, J., and George, P.: 'User-derived impact analysis as a tool for usability engineering', ACM SIGCHI, 1986, 17, (4), pp. 241-246

[5] Nielsen, J.: 'Usability engineering' (Academic Press Professional, 1993)

[6] Brink, T., Gergle, D., and Wood, S.D.: 'Design web sites that work: usability for the web' (Morgan-Kaufmann, 2002)

[7] Granollers, T.: 'User centred design process model, integration of usability engineering and software engineering', Proc. INTERACT 2003, Zurich, Switzerland, 2003, pp. 673-675

[8] Sutcliffe, A.: 'User-Centred Requirements Engineering. Theory and Practice' (Springer, 2001)

[9] Pressman, R.S.: 'Software Engineering: A Practitioner's Approach' (McGraw-Hill, 2004)

[10] Devanbu P., Brachman R.J., Selfridge P.G., and Ballard B.W.: 'LaSSIE: a knowledge-based software information system', Commun. ACM, 1991, 34, (5), pp. 36–49

[11] Happel, H.-J., and Seedorf, S.: 'Applications of Ontologies in Software Engineering', Proc. of Int. Workshop on Semantic Web Enabled Software Engineering, 2006

[12] Hyland-Wood, D., Carrington, D., and Kaplan, S.: 'Towards a software maintenance methodology using Semantic Web techniques and paradigmatic documentation modelling', IET Software, 2008, 2, (4), pp. 337–347

[13] Nadarajan, G., and Chen-Burger, Y.-H.: 'Translating a typical business process modelling language to a Web Services Ontology through lightweight mapping', IET Software, 2007, 1, (1), pp. 1-17

[14] Zhao, Y., Dong, J., and Peng, T.: 'Ontology Classification for Semantic-Web-Based Software Engineering', IEEE Transactions on Services Computing, 2009, 2, (4) pp. 303-317

[15] Kim, H.M., Fox, M.S., and Gruninger, M.: 'An Ontology of Quality for Enterprise Modelling', Proc. 4th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WET-ICE'95, 1995, pp. 105-115

[16] García-Crespo, A., Colomo-Palacios, R., Gómez-Berbís, J.M., and García-Sánchez, F.: 'SOLAR: Social Link Advanced Recommendation System', Future Generation Computer Systems, 2010, 26, (3), pp. 374-380

[17] Decker, B., Ras, E., Rech, J., Klein, B., and Hoecht, C.: 'Self-organized reuse of software engineering knowledge supported by semantic wikis', Proc. Workshop on Semantic Web Enabled Software Engineering (SWESE), Galway, Ireland, 2005

[18] Riechert, T., and Lohmann, S.: 'Mapping Cognitive Models to Social Semantic Spaces – Collaborative Development of Project Ontologies', Proc. 1st Conference on Social Semantic Web, 2007, pp. 91-98

[19] Colomo-Palacios, R., García-Crespo, A., Gómez-Berbís, J.M., Casado-Lumbreras, C., and Soto-Acosta, P.: 'SemCASS: technical competence assessment within software development teams enabled by semantics', International Journal of Social and Humanistic Computing, 2010, 1, (3), pp. 232-245

[20] Gómez-Berbís, J.M., Mencke, M., Chamizo, J., Colomo-Palacios, R., and García-Crespo, A.: 'EsaCake: A Semantic Software Environment for Sharing Software Projects Knowledge Based on the ESA Software Methodology', Proc. Third International Conference on Internet and Web Applications and Services, 2008

[21] Corcho, O., Fernández, M., Gomez-Perez, A., and López-Cima, A.: 'Building Legal Ontologies with METHONTOLOGY and WebODE', in Benjamins, R., Casanovas, P., Breuker, J., and Gangemi, A. (Eds.): 'Law and the Semantic Web' (Springer, 2005), pp. 142-157

[22] http://rhizomik.net/ontologies/2010/04/qmsonto.owl, accessed June 2010

[23] http://dublincore.org/documents/dcmi-terms/, accessed June 2010

[24] Sure, Y., Bloehdorn, S., Haase, P., Hartmann, J., and Oberle, D.: 'The SWRC Ontology - Semantic Web for Research Communities', Proc. 12th Portuguese Conference on Artificial Intelligence, Covilha, Portugal, 2005

[25] http://en.wikipedia.org/wiki/Semantic_wiki, accessed June 2010

[26] Vrandečić, D., and Krötzsch, M.: 'Semantic MediaWiki', in Davies, J., Grobelnik, M., and Mladenic, D. (Eds.): 'Semantic Knowledge Management' (Springer, 2009), pp. 171-179

[27] http://en.wikipedia.org/wiki/MediaWiki, accessed June 2010

[28] http://www.mediawiki.org/wiki/Extension:Semantic_MediaWiki, accessed June 2010

[29] http://www.mediawiki.org/wiki/Security_issues_with_authorization_extensions, accessed June 2010

[30] http://drupal.org/project/drupal, accessed June 2010

[31] García, R., Gimeno, J.M., Perdrix, F., Gil, R., Oliva, M., López, J.M., Pascual, A., and Sendín, M.: 'Building a Usable and Accessible Semantic Web Interaction Platform', World Wide Web, 2010, 13, (1-2), pp. 143-167

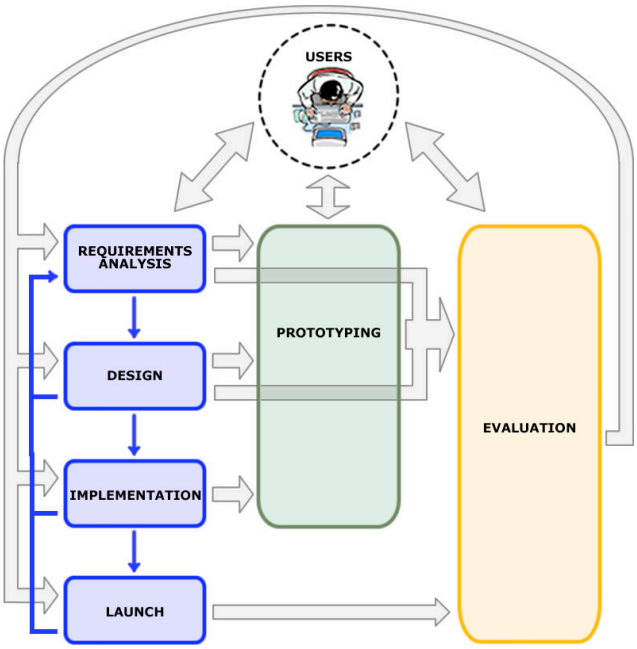

Fig. 1: Usability and Accessibility Engineering Process Model (MPIu+a).

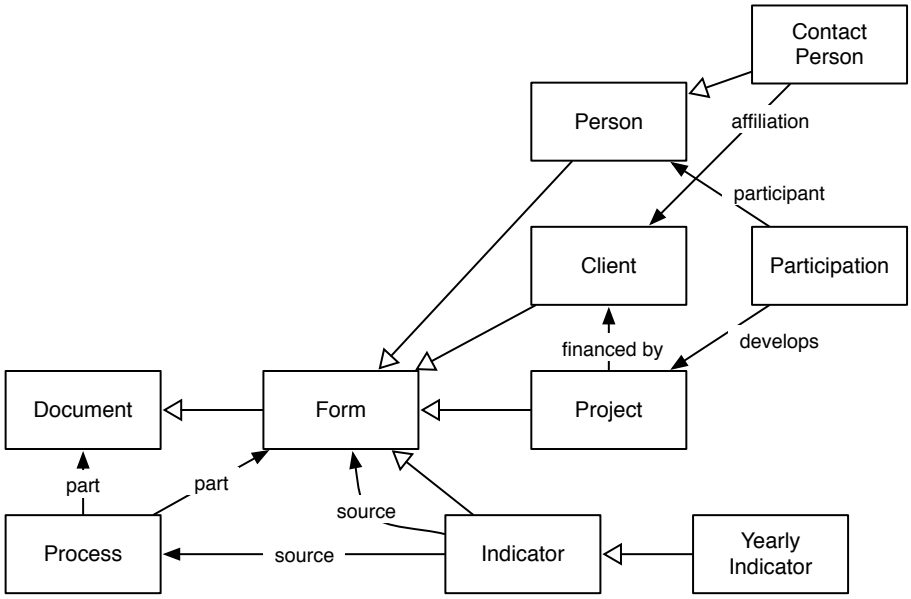

Fig. 2. Diagram of the core QMS Ontology concepts (subclass of relation denoted by white arrowheads and properties denoted by black arrowhead with domain the arrow source and range the arrow target)

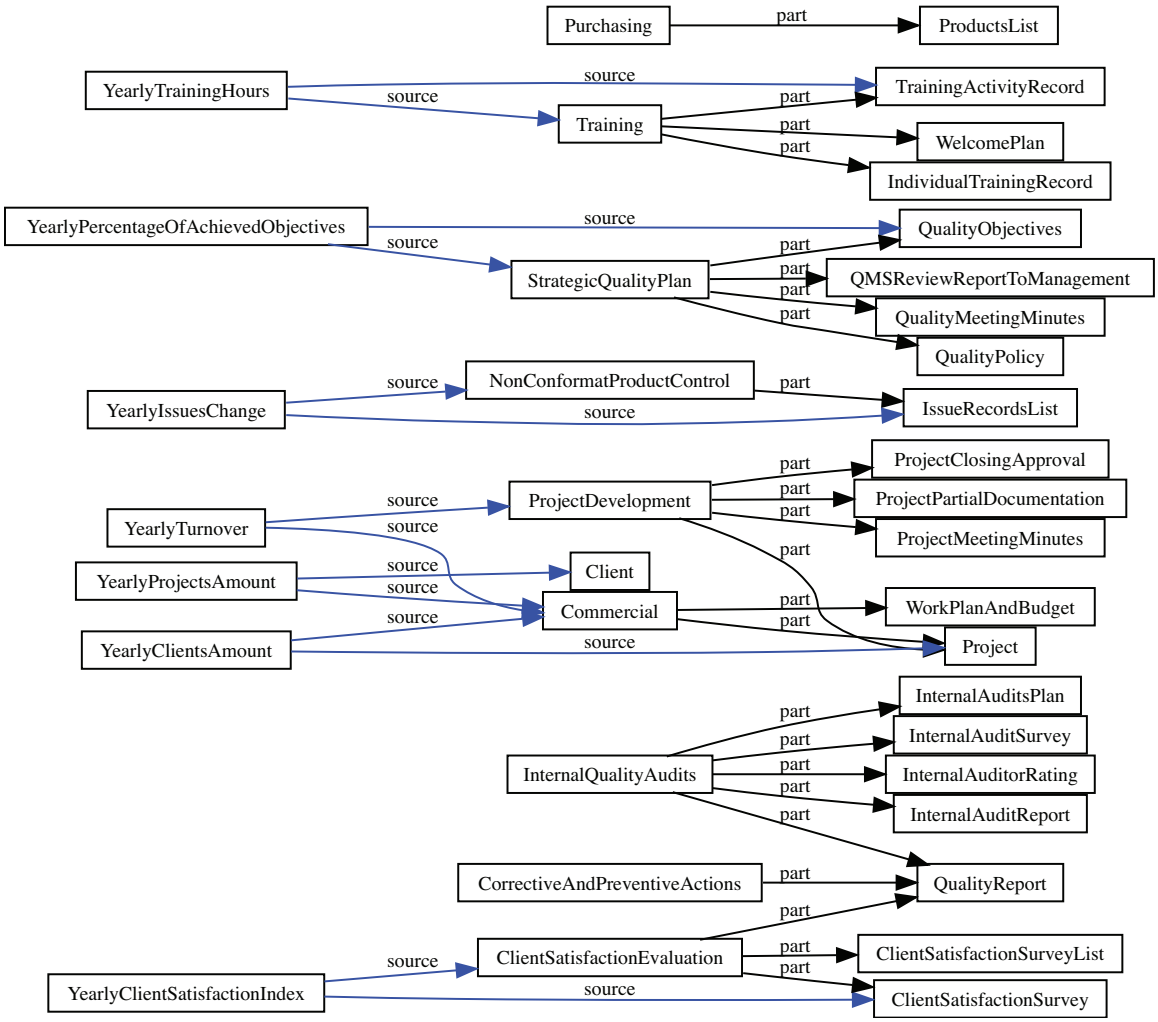

Fig 3. Relations among Indicators (left), Processes (centre) and Documents and Forms (right)

Fig 4. Example of a Semantic Form providing assistance while filling an input with Person values