1

A Lightweight Approach To Enterprise Architecture Modeling and Documentation

Sabine Buckl, Florian Matthes,
Christian Neubert, and Christian M. Schweda

Technische Universität München, Institute for Informatics,
Boltzmannstr. 3, 85748 Garching, Germany
{sabine.buckl,matthes,neubert,schweda}@in.tum.de
http://www.systemcartography.info

Summary. Not quite a few enterprise architecture (EA) management endeavors start with the design of an information model covering the EA-related interests of the various stakeholders. In the design of this model, the enterprise architects resort to prominent frameworks, but often create what would be called an "ivory tower" model. This model at best case misses if not ignores the knowledge of the people that are responsible for business processes, applications, services etc. In this paper, we describe how the wisdom of the crowds can be used to develop information models. Making use of Web 2.0 techniques, wikis, and an open templating mechanism, our approach ties together the EA relevant information in a way, which is accessible to both humans and applications. We demonstrate how the ivory tower syndrome can be cured, typical pitfalls can be avoided, and employees can be empowered to contribute their expert knowledge to EA modeling and documentation.

Key words: enterprise architecture management, wikis, wisdom of the crowds, information model, collaborative modeling

1.1 Introduction

Trends as globalization, rapid economic change, and the necessity to foster an organization's sustainable competitive advantage have considerably increased the importance of knowledge on the internal structure of an organization. While this knowledge is typically already existing in different data sources, e.g. CMDBs, process descriptions, or as implicit expert knowledge, linking these data sets is a challenge. Enterprise architecture (EA) management is an instrument to address this challenge by providing methods and means to enable a holistic view on the organization to foster the alignment of business and IT (cf. [1, 2]).

Providing a holistic view on the EA yields two main challenges. First, to link the different data sources a common *information model* describing the

EA, i.e. "the fundamental organization of [the enterprise] embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution" [3] has to be developed. This development more often than not suffers from the 'ivory-tower syndrome', i.e. leads to the creation of a wish list in which each stakeholder raises concerns that he is interested in. This results in an unmaintainable giant information model, as documented in the EAM anti-pattern of the same name described by Buckl et al. in [4] . Second, documenting the EA is an advanced topic, as the EA typically includes different organizational units, a few hundreds up to thousands application systems and their interconnections, etc. The documentation process accordingly must be conducted in a way, which is on the one hand feasible for stakeholders with various backgrounds as e.g. process or application owners, and which on the other hand shows the benefits of their time spent on sharing knowledge.

To address challenges of the latter type, a new generation of systems has emerged in the last years facilitating team collaboration and knowledge exchange. These systems of which for example Wikipedia[1] is a prominent example utilize so-called *Web 2.0* technologies (cf. Oreilly in [5]). Due to the success of these tools in the Internet, the techniques are currently adopted and transfered to the enterprise context under the name *Enterprise 2.0* (cf. McAfee in [6, 7]). Today's Enterprise 2.0 tools provide many distinct services, such as collaborative authoring, tagging, bookmarking, awareness, commenting, rating, linking, search, social networking, access control, as well as a multitude of different content types, such as wikis, blogs, or files [8].

In this article, we demonstrate how the "wisdom of the crowd" [9] may be used to enable EA documentation and modeling. By making use of Web 2.0 techniques, wikis, and an open templating mechanism, we show how the ivory tower syndrome can be cured, typical pitfalls can be avoided, and employees are empowered to contribute their expert knowledge. In Section 1.2, we introduce selected Web 2.0 and Enterprise 2.0 techniques. Utilizing these techniques, we present the hybrid wiki approach in Section 1.3. We further illustrate how it contributes to a lightweight approach to EA modeling as well as documentation and discuss use cases in the EA management context in Section 1.4. Complementing the illustration of the approach, Section 1.5 describes a prototypic implementation based on an existing enterprise wiki system. The paper concludes with a critical reflection of the achieved results and sketches areas of future research in Section 1.6.

1.2 Enterprise 2.0 techniques

In [9] Surowiecki writes about collective intelligence and its impact on enterprises. He explains that collective intelligence of a group mostly surpasses

[1] www.wikipedia.org

the knowledge of particular experts and how companies can benefit from this fact, which he refers to as "the wisdom of crowds". Especially in context of managing socio-technical systems, such as enterprises, where a multitude of stakeholders with different backgrounds should be involved, Enterprise 2.0 techniques can be applied beneficially. Considering for instance the design of a suitable information model to model the architecture of an enterprise yields challenges that can be faced by utilizing the wisdom of crowds. More often than not, the ivory tower syndrome can be observed when an information model is developed, where a group of enterprise architects creates an 'invented' information model. Such model does on the one hand not adequately address the concerns of the corresponding stakeholders and on the other hand lacks acceptance as it is based on terms and definitions of a dedicated user group and therefore might be difficult to understand for other stakeholders. Enterprise 2.0 techniques can be used to cure this syndrome by enabling employees of all departments to make explicit and contribute their expert knowledge for parts of the EA they are responsible for (e.g. business processes, applications, IT infrastructure) thereby resulting in a 'vivid' information model.

1.2.1 Collaboration, content sharing, and discretionary access control

Enterprise 2.0 is characterized as the usage of social software within enterprises [6, 7]. A major objective of social software is to empower users to collaborate via the Internet. The users can create different kinds of content objects, such as wikis, blogs, files, which they can share with other users within certain groups. For instance, a group can be a part of the user's personal social network, or a group could have been created for a certain purpose of collaboration (cf. "functional groups" in [8]). Especially in enterprises the sharing of content requires mechanisms to protect sensitive content (objects) from unauthorized access. In [8] this mechanisms are described as "access control" services for integrated Enterprise 2.0 platforms.

Since the architecture of an enterprise can be very complex, consisting e.g. of hundreds or even thousands of business applications, the documentation and modeling of an EA requires the involvement of many different stakeholders with various backgrounds (cf. [10, 11, 12]). Collaboration, synchronization, and content sharing is most likely to be an appropriate approach to gather EA-related information and document constituents of an enterprise, in order to provide a holistic view on the EA. Thereby functional groups can be used to grant access to particular content objects, e.g. customers could be involved in the documentation process for some parts of the EA[2]. Furthermore, social networks may help to find persons which are experts for a certain part of the EA, e.g. a specific business process, to get in contact with for in-depth questions and discussions.

[2] A customer could be interested in providing information on a shared application interface which is critical for his business.

1.2.2 Notifications and recommendations

Todays integrated Enterprise 2.0 platforms provide mechanisms to get users notified in case of certain system activities, which is referred to by Büchner et al. in [8] as "awareness". Users can register to get notifications to a) keep track of other users activities (e.g. a user creating a new wikipage) and b) to follow activities on certain content objects (e.g. a wikipage being modified by a user). These activities are mirrored in a user's stream of notifications made available via a personal dashboard within the platform and typically complemented by an RSS feed.

Since the documentation of an EA has a strongly collaborative character, as discussed before, group-based notifications on modifications and communication of ongoing documentation initiatives is often regarded to be crucial for following reasons:

- A user responsible for certain parts of the EA is getting notified if corresponding parts of the documentation have changed, e.g. to perform quality assurance.
- The head of a group of users employed with documenting parts of the EA can keep track on the progress made in the documentation endeavor.

The notification mechanisms, as implemented in todays Enterprise 2.0 platforms, may be appropriate to address the demands as sketched above. Nevertheless, further demands may exist, e.g. a user might want to get notified on objects, which have not been changed for a certain period of time. These notification might give indications on parts of the EA documentation becoming outdated. Even though this service is not supported by any of the currently available platforms out-of-the-box (cf. Büchner et al. in [8]), such notifications can be realized based on the "last modification date" of a content object.

In *recommender systems* a user profile and user activities are analyzed in order to find information (e.g. content objects, other users) that are most likely to be interesting for the user under consideration. For instance, if a user visited wikipages that are almost all tagged with a certain keyword, it is most probably that he is interested in other wikipages which are tagged just the same. In this case the system could send him a recommendation message to visit these pages. Even though recommender systems are rather rare supported by Enterprise 2.0 platforms, this can be valuable for EA documentation to

- facilitate knowledge exchange, especially in large size companies employees often do not know the competences of their colleagues,
- to identify experts for certain parts of the EA for in-depth questions and discussions, and
- to discover already documented parts of the EA, which can be useful to avoid duplicated documentation efforts.

1.2.3 Structured content in (enterprise) wikis

Wikis are the most widely supported content type of today's integrated Enterprise 2.0 platforms. A wikipage as part of a wiki may be characterized as a website, that can be directly edited in a web browser. Wikipages enable users to publish and share their knowledge in a way, which is easily accessible and thereby provides an intuitive way to collaboratively create and modify content. The most prominent wiki system in the internet is the encyclopedia Wikipedia. In Wikipedia each page represents a particular concept, e.g. a city, which is mainly described in the content of the wikipage. Beside this description a wikipage additionally contains *metadata* of the represented concept, e.g. population and foundation date, which are shown as key-value-pairs in a tabular view. Since it would be laborious to *re-define* these metadata keys for each particular city to be describe in future, Wikipedia provides a reuse mechanism, called *templates*. Templates are prototypes for a certain concept with several attributes. Considering the table as explained above, these attributes are the keys of the metadata-table. Templates as used by the Wikipedia are rather unusual in Enterprise 2.0 platforms, since users have to learn a specific markup-language.

Since wiki templates enable to abstract, reuse and unify information, they might be a suitable instrument for EA documentation activities. An enterprise architect for example could predefine a set of templates in order to give suggestions how to document a certain part of the EA. As already mentioned before, an information model invented by a single person is likely to be unsuitable for EA management, where many different stakeholders should be involved. In Section 1.3 we describe how templates can be used to create an information model in a bottom-up fashion.

1.2.4 Linking and bookmarking

In [13, 14] the importance of relating EA-related information from different layers[3] like business processes, application systems, and infrastructure elements and providing a navigation mechanism between distinct information objects is discussed. This fact is also reflected in almost every Enterprise 2.0 platform. In order to connect any kind of content objects, these platforms supply several services for the management of internal and external links [8]. Considering internal objects, *backlinks* are further supported. Backlinks reference objects the currently considered content object is referenced by. The number of backlinks can be regarded as an indication for the popularity of a content object. In order to save and share links to favorite wikipages, the feature of *social bookmarking* is provided, which further supports tagging of pages.

[3] See Winter and Fischer in [15] for typical layers in the context of EA management.

1.3 Hybrid wikis for EA management

While utilizing the wisdom of the crowds requests for a way to store unstructured information, an EA management initiative builds on a common understanding of the constituents that make up the EA, i.e. structured information. Therefore, we present an approach building on the capability to combine structured and unstructured information in a wiki system. While an open-templating mechanism is used to store the structured parts of a wikipage, the content of the page contains the unstructured information. A wiki supporting to store structured and unstructured information is hence called a *hybrid wiki*. Our hybrid wiki approach is demonstrated alongside an open source web collaboration system called Tricia [16], which can be characterized as an Enterprise 2.0 tool according to Büchner et al. in [8]. Tricia provides several content types, such as wikis, files, and blogs.

A template may be considered a simple table, providing multiple attribute-value-pairs and a global name. The name of the template is used to specify the concept which is described in the wikipage's content, e.g. in the context of EA management this could be the concept of an "Application System". Continuing this example, the wikipage could describe an actual application system "SAP Business Suite" in the page's content. "SAP Business Suite" obviously represents the name of the application system and therefore can be added as an attribute-value-pair (name: SAP Business Suite) to the template. In the content further attributes (and corresponding values) can be identified and also be added to the template, e.g. version, availability, criticality etc. By doing a meta-structure is continuously developed on the wikipages and the corresponding templates. Beside literals hyperlinks can be used as values of template attributes, referencing other wikipages also having a certain template assigned. Thereby, an information model is created bottom-up via the templates – corresponding to concepts in the information model – with the attribute and links – corresponding to the association in the information model.

Templates are connected with a corresponding meta-type, which is initialized when a new template is created on a wikipage "template definition". A template definition has a name and provides all attributes of the corresponding templates. Considering the previous example, the template definition has the name "Application System" and provides the following attributes: name, version, availability, and criticality. That means, all templates are instances of a certain template definition and furthermore, existing template definitions can be assigned to wikipages as templates. To manage the template definitions, e.g. create new definitions, rename attributes, type attributes, a second way of more restrictive modeling is enabled, which we refer to as top-down information modeling.

Figure 1.1 summarizes an application example of out approach. The enterpris hybrid wiki is therein used by several technical departments in the company for documenting relevant information. By using templates, they col-

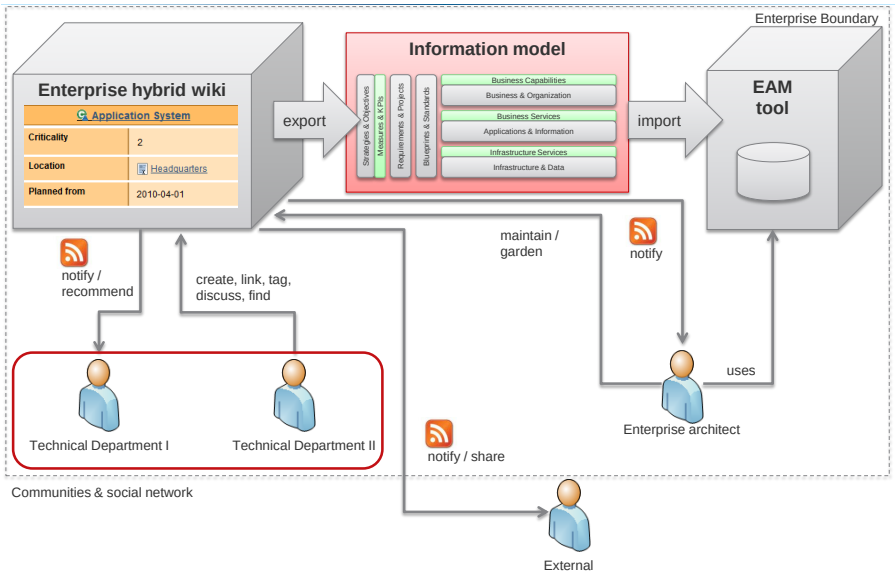

Fig. 1.1. Enterprise Hybrid Wiki for EA management

laboratively elicit an information model which is further "gardened" by an enterprise architect. This means that the enterpris architect uses reporsts on the used templates as well as their associated attributes to decide on the attributes to keep, to set mandatory or to abandon. Further, the architect decides on "refactorings" of the attributes, i.e. on name changes, etc. The employees of the department start creating wikipages for each element to be documented. They assign templates to the wikipages in order define the concept described in the page, structure the content, and link to other pages the modeled concept is related to. Since all employees of the department are in the same functional group, i.e. are members of Technical Department I, all users of this group are notified by any system changes via RSS feeds, e.g. when a template of a wikipage changes or a template attribute is renamed. Since the users are aware of new content and changes, they can use other Enterprise 2.0 services to improve the quality of the content, e.g. by means of discussing, commenting, and rating services.

The system may also recommend wikipages, which might be useful for particular users within the group, by analyzing the content of the pages and user behavior. In the example, an *external user* is allowed to follow the system changes by RSS notifications. For instance, it could be useful to keep customers informed about parts of the EA which are relevant for their business. One could also imagine to grant write access to external users, e.g. allowing customers to comment on wikipages, or take part in discussions.

1.4 Use cases

In the following we describe the use cases a hybrid wiki as discussed in the above section needs to correspond to. These use cases mirror the information model related use cases put forward by Matthes et al. in [14] and form the foundation for the prototypic implementation described in Section 1.5. We distinguish between use cases for the open templates, which can be assigned to wikipages and for the management of their definitions.

Uses cases for the open templating mechanism

New templates can be defined on a wikipage and existing templates can be assigned to a wikipage. Each wikipage can only have one template assigned (UC10). After the assignment of the template all attributes available for this template are shown with empty values (UC20). A new attribute can be added to a template which is assigned to a wikipage, i.e. the new attribute is added to the template definition. In response on all other wikipages having the same template assigned an empty value is shown for the new attribute (UC30). Values can be set for attributes of a wikipage template. A value can either be literal or a link (UC40). An attribute of a wikipage template can be deleted. If another wikipage has the same template assigned and a non-empty value for this attribute, the attribute remains in the template definition. Otherwise the attribute is deleted in the template definition (UC50). Templates can be removed from a wikipage. If the template is not assigned to another wikipage, the corresponding template definition is deleted (UC60). An attribute can have multiple values. In this case a mixture of both text and link values is allowed (UC70). A template has the same access rights as its owning wikipage (UC80).

Use cases for managing templates

A new template definition can be created independent of wikipages. A template definition has a unique name (UC90). A new attribute can be added to an existing template definition. The attribute has a unique name (UC100). For each template definition the following statistics of its templates are provided (UC110):

- A numeric indication is given how many templates belong to a template definition, i.e. total number of instances.
- A numeric indication for each attribute is shown illustrating how many templates have not empty values, i.e. non-empty attributes.

For a template definition the corresponding templates with their attribute values are shown in an tabular view (UC120). A template definition attribute can be renamed, i.e. all corresponding template attributes are renamed too (UC130). A template definition attribute can be deleted, i.e. all corresponding

template attributes are deleted, too (UC140). A template definition can be deleted, i.e. all corresponding templates are deleted, too (UC150). By default all in the system registered users may read a template definition and create a new one. The access rights can be defined more restrictive (UC160).

1.5 Prototype implementation

According the use cases specified in Section 1.4 we introduce a prototypic implementation of our hybrid wiki approach. The implementation is based on the web collaboration and knowledge management tool Tricia [16]. Tricia is an open source web framework which enables a straightforward development of custom applications by providing extension points and a plugin mechanism, following a data model-driven approach. Since Tricia belongs to the category of Enterprise 2.0 tools [8], it provides many of the services as introduced in Section 1.2, e.g. role-based access control, search, notification, etc. and additionally provides a wiki plugin out-of-the-box. Subsequently, we illustrate the concepts of the prototype's data model, present exemplary user interfaces for templates and their definitions, and show how an information model can be extracted from these templates.

Data model

The data model of the hybrid wiki application is shown in Figure 1.2. A `TemplateDefintion` provides a unique name, the name of the concept, e.g. business application. Each `TemplateDefinition` has a rich text field to enable users to add a detailed description of the concept via a hypertext editor and may have multiple features (`TemplateDefinitionFeatures`), also providing a unique name attribute, e.g. availability.

Templates are realized by using the mechanism of OptionalMixins of the Tricia framework [16], which allow dynamically attaching a `Template` to a Tricia wikipage at runtime. Each `Template` belongs to exactly one `TemplateDefinition` and may have multiple features (`TemplateFeature`). The `TemplateFeature`s of a `Template` comply with the `TemplateDefinition-Feature`s of its `TemplateDefinition` (i.e. their names are matching), with the exception, that a `TemplateFeature` is only stored in the database if at least one value (literal or link) is assigned. Each `TemplateFeature` may have multiple values (`TemplateFeatureValue`), latter can either be a literal (`StringValue`) or a hyperlink (`LinkValue`). Since Tricia treats links as first class objects, the link attribute of `LinkValue` only contains the primary key of the linked object.

Management of templates

In the example of Figure 1.3 a `TemplateDefinition` of the concept "Application System" is given. The application system provides a detailed hypertext de-

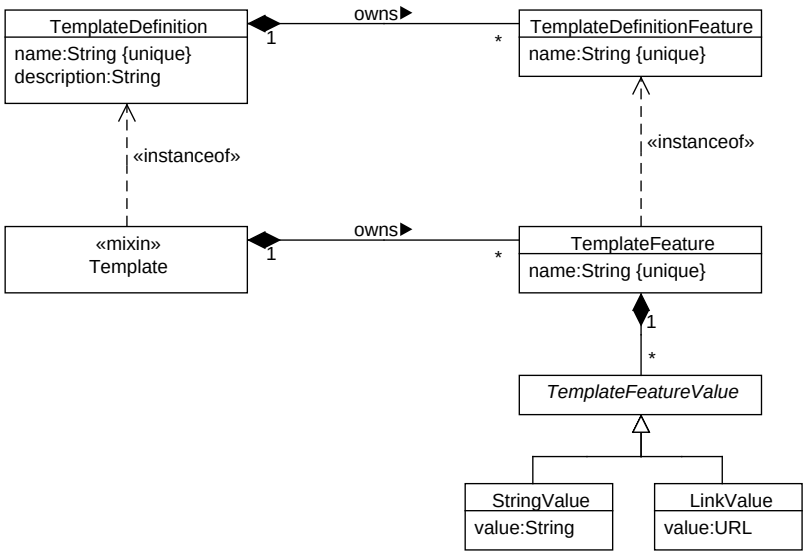

Fig. 1.2. Data model of the hybrid wiki approach

scription of its purpose and several attributes (`TemplateDefinitionFeatures`), e.g. criticality, location, etc. Attributes can be added, edited and removed, and both kind of template statistics are provided (c.f. Section 1.4). Furthermore the corresponding templates with their attribute values are shown in an tabular view. The *New* action shown in the header of the page enables user to create new template definitions independent of wikipages.

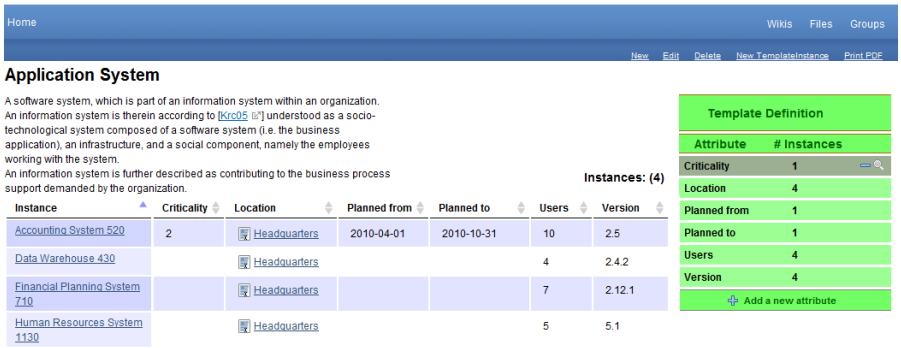

Fig. 1.3. Definition for the template application system

Templates as part of wikipages

Figure 1.5 shows a wikipage having an application system template assigned. Since the site is a wikipage, the application system named *Accounting System 520* can be described in detail assisted by a hypertext editor (WYSIWYG editor). In the example the *Location* attribute contains a hyperlink to an organizational unit *Headquarters*, where the *Application System 520* is located at, i.e. Headquarters is a wikipage, which has an *Organizational Unit* template assigned (cf. Figure 1.4). The remaining attributes of *Accounting System 520* are literal values (`StringValue`). The attribute *Users* is provided by the template definition *Application System*, but not inscribed in the template, i.e. not stored as a `TemplateFeature`.

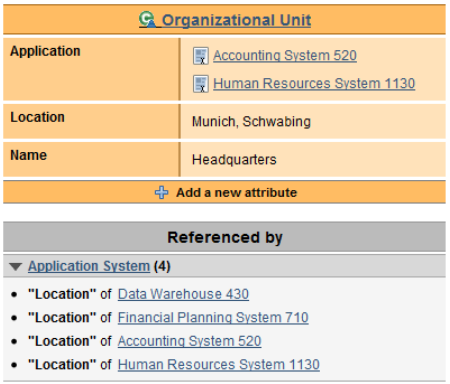

Fig. 1.4. Template *organizational unit* of a wikipage *Headquarters*

The template can be removed from the wikipage, attributes can be added (+), modified (input field), and removed (-). In the division *Referenced by* at the bottom right all templates are listed having an attribute (`TemplateFeature`) with a hyperlink (`LinkValue`) to *Accounting System 520* grouped by template definitions, i.e. in the example *Accounting System 520* is referenced by two *Organizational Unit* wikipages, namely *Headquarters* and *Finance*, from the attribute *Application* (c.f. Figure 1.4). The *Application* attribute from the *Headquarters* template in Figure 1.4 uses multiple hyperlinks (`LinkValue`) to list all deployed applications.

Information model

Considering the sample data from above, an information model, which is shown as an UML class diagram in Figure 1.6, can be extracted from the template definitions and their template instances. Thereby each template definition with its literal attributes (`StringValue`) is mapped to an UML class

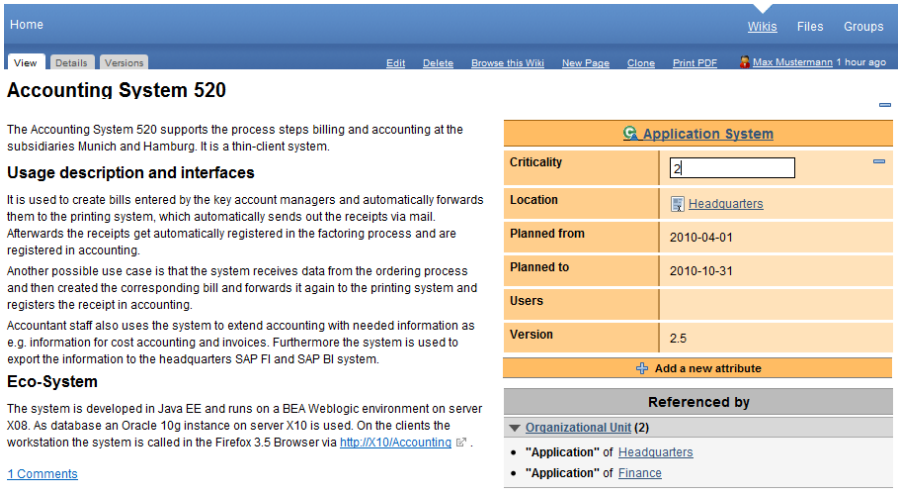

Fig. 1.5. Wiki page with template *Application System*

with corresponding string attributes. Each hyperlink value (`LinkValue`) of a template attribute referencing a wikipage also having a template assigned is mapped to an UML association between the classes representing both, the referenced and the referencing template. Thereby a role name which complies with the name of the template attribute is created for the association. In the example two associations emerge – location and application. The multiplicity of *application* may be assumed as arbitrary (*) since the application attribute of *Headquarters* (c.f. Figure 1.4) has multiple link values to *Application System* templates assigned. The multiplicity of *location* is assumed to be exactly one (1) since all *Application System* templates reference exactly one *Organizational Unit*, namely *Headquarters* (cf. Figure 1.3).

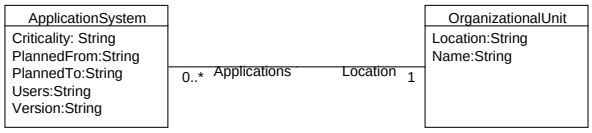

Fig. 1.6. Information model extracted from templates

1.6 Summary and future work

In this paper, we demonstrated how Web 2.0 techniques, wikis, and an open templating mechanism can be used to support EA management. We described

how the wisdom of the crowds can be utilized for the documentation of EA-related concepts and their instances collaboratively in wikipages. Furthermore, we introduced an open templating mechanism for wikipages to collaboratively evolve an information model in the context of EA management. Thereby both ways of lightweight EA modeling can be applied, bottom up by evolving the model by means of the templates and top down by using template definitions. The theoretic discussion was complemented by the description of a prototypic implementation of the hybrid wiki approach based on the open source Enterprise 2.0 tool Tricia (cf. [16]). Other Tricia services may be useful in the context of EA management, e.g. change feeds, to enable users getting notified when the information model or the documentation changes. Therefore, future areas of research encompass the

- export of the information model based on the Eclipse Modeling Framework [17],
- integration of generated visualizations of the information model in wikipages for different stakeholders [18],
- beside literals and links, support of other data types for template attributes, e.g. Date, Integer,
- validation rules and constraints for template attributes, e.g. mandatory attributes,
- semantic annotations within the wiki page's content,
- template-based recommender system with respect to the groups a user belongs to, and
- support of inheritance for template definitions.

Further aspects of interest are e.g. the utilization of rating functions on wiki-pages to allow users to provide feedback in a structured and uniform way. Concerning these functions a multitude of implications, e.g. on user motivation would have to be considered. Additionally, these ratings could provide valuable input to the analysis of documentation templates, although a clear correlation between a low rating and the corresponding template used is not likely to be easily determinable. The Hybrid Wiki approach is currently being evaluated in practice.

References

1. The Open Group: TOGAF "Enterprise Edition" Version 9. `http://www.togaf.org` (cited 2010-02-25) (2009)
2. Zachman Institute for Framework Advancement: The zachman enterprise framework. `http://www.zachmaninternational.com/index.php/the-zachman-framework` (cited 2010-02-25) (2009)
3. International Organization for Standardization: ISO/IEC 42010:2007 Systems and software engineering – Recommended practice for architectural description of software-intensive systems (2007)

4. Buckl, S., Ernst, A.M., Matthes, F., Schweda, C.M.: How to make your enterprise architecture management endeavor fail! In: Pattern Languages of Programs 2009 (PLoP 2009), Chicago. (2009)
5. Oreilly, T.: What is web 2.0: Design patterns and business models for the next generation of software. Social Science Research Network Working Paper Series (August 2007)
6. McAfee, A.: Enterprise 2.0: The dawn of emergent collaboration. IEEE Engineering Management Review **34**(3) (2006) 38–38
7. Bughin, J.: The rise of enterprise 2.0. Journal of Direct, Data and Digital Marketing Practice **9**(3) (2008) 251
8. Büchner, T., Matthes, F., Neubert, C.: A concept and service based analysis of commercial and open source enterprise 2.0 tools. In: International Conference on Knowledge Management and Information Sharing, Madeira, Portugal (2009) 37–45
9. Surowiecki, J.: The Wisdom of Crowds. Random House, New York, USA (2004)
10. Fischer, R., Aier, S., Winter, R.: A federated approach to enterprise architecture model maintenance. In: Enterprise Modelling and Information Systems Architectures – Concepts and Applications, Proceedings of the 2nd International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2007), St. Goar, Germany, October 8-9, 2007, St. Goar, Germany (2007) 9–22
11. Kurpjuweit, S., Winter, R.: Viewpoint-based meta model engineering. In Reichert, M., Strecker, S., Turowski, K., eds.: Enterprise Modelling and Information Systems Architectures – Concepts and Applications , Proceedings of the 2nd International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA'07), St. Goar, Germany, October 8-9, 2007. LNI, Bonn, Germany, Gesellschaft für Informatik (2007) 143–161
12. Moser, C., Junginger, S., Brückmann, M., Schöne, K.M.: Some process patterns for enterprise architecture management. In: Software Engineering 2009 – Workshopband, Bonn, Germany, Lecture Notes in Informatics (LNI) (2009) 19–30
13. Kurpjuweit, S., Aier, S.: Ein allgemeiner Ansatz zur Ableitung von Abhängigkeitsanalysen auf Unternehmensarchitekturmodellen. In: 9. Internationale Tagung Wirtschaftsinformatik (WI 2007), Wien, Austria, Österreichische Computer Gesellschaft (2007) 129–138
14. Matthes, F., Buckl, S., Leitel, J., Schweda, C.M.: Enterprise Architecture Management Tool Survey 2008. Chair for Informatics 19 (sebis), Technische Universität München, Munich, Germany (2008)
15. Winter, R., Fischer, R.: Essential layers, artifacts, and dependencies of enterprise architecture. Journal of Enterprise Architecture **3**(2) (2007) 7–18
16. InfoAsset: Tricia – open source web collaboration and knowledge management software (2010)
17. Moore, B., Dean, D., Gerber, A., Wagenknecht, G., Vanderheyden, P.: Eclipse development using the graphical editing framework and the eclipse modeling framework. `http://www.redbooks.ibm.com/redbooks/pdfs/sg246302.pdf` (cited 2010-02-25) (2004)
18. Buckl, S., Ernst, A.M., Lankes, J., Matthes, F., Schweda, C., Wittenburg, A.: Generating visualizations of enterprise architectures using model transformation (extended version). Enterprise Modelling and Information Systems Architectures – An International Journal **2**(2) (2007) 3–13