In: Proceedings of the ACIS Fourth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD'03), Dosch, W., Lee, R. Y. (Ed.), ACIS (Pub.), October 16-18, 2003, Lübeck, Germany. ACIS 2003, ISBN 0-9700776-7-X pp 106-113

Using a Semantic Web for Process Information and Quality Management

Ronald Hartwig, Michael Herczeg

Institute for Multimedia and Interactive Systems, University of Luebeck

Willy-Brandt-Allee 31a, D-23554 Luebeck, Germany

{hartwig|herczeg}@informatik.uni-luebeck.de

www.imis.uni-luebeck.de

Abstract

This paper describes an approach (and its implementation) on how to handle the large number of data from user centered engineering processes. It uses object-oriented abstraction methods in combination with a semantic web to cover the development process from the requirements engineering throughout the final product. The objects are made accessible using an easy to use web-interface. The system is based on a simple but flexible XML-database.

1. Introduction

The basic idea of quality management processes is to make all quality requirements and decisions transparent and comprehensible. This targets the developers but also quality managers and end users. Using dynamic, iterative process models like the spiral model or extreme programming (XP) produces a huge amount of such complex semiformal data. This data needs to be interconnected so decion chains can be comprehended and evaluated. The connections represent the dependencies of the process information: Requirements are based on analysis data and the development decisions are based on these requirements. In the end the resulting product should adhere to the requirements. Evaluation data must be included to prove this and to deliver valuable input for the next iteration.

2. Original Scenario

Object-oriented analysis (OOA) and the following *object-oriented design* process (OOD) focus on the description of the interdependencies between system objects. The main idea of the approach described here is to use this powerful information management technique for *all* process data. For example the raw data from use scenarios, user tests or empirical work and the derived requirements may be modeled as objects as well as the OOA data. This broadens the scope of existing CASE tools and takes especially the *user centered process* steps into account.

A very important point while designing a tool support for such a process data repository was the integration of very different domains of knowledge. In typical development processes not only developers but also organizational experts, user interface designers, usability experts, domain experts and end users contribute important information to the development process. In order to support such *participatory design*, the information must be made accessible in a simple and compact way. Readers must be enabled to find valuable data for their area of responsibility without having to browse and comprehend the whole process data model. For this reason small and specialized views on certain aspects are needed. For example users may only want to validate their own scenario. They don't need to see whole information model, maybe dealing with technical requirements, in every detail. Quality managers may want to concentrate on analyzing the decision chains within the process. Developers need context information for their current module but possibly do not have the time to backtrack through the decision chains in detail.

3. Requirements

The model and the tool described within this paper have been developed and used for a development process for multimedia learning modules. It had to focus on different domain experts from areas like pedagogies, usability, web-design and software-design. We believe that the main ideas of this approach may be used for other kinds of (software-) development as well. The following general goals apply:

1. *Consistency* of data during the whole development process
2. *Understandability* of the data to all participants of the development process including the end users.
3. *Flexibility*, scalability and adaptability to different project sizes and quality requirements.

These goals are addressed using two main information management methods:

- A *semantic web*, so that all related information is interconnected. It should help to keep the data consistent (section 3.1-3.3).
- *Inheritance* as known from object-oriented modeling helps to cope with the complexity and amount of data. This addresses the issue of understandability (section 3.4).

The third goal of flexibility is considered in the rule-based implementation concept (see sections 3.5 and 4).

3.1 Classes of Process Data

The central point of this approach is the object model as described in Figure 1. Being based in the quality management the notion of the *"potential deficiency"* is the central object instead of the objects known from OOA or specialized human-factor approaches [4], [5].

Following an ideal process model and the ideas of iterative development one would start with an analysis phase, maybe using use scenarios or other techniques, to asses the context information basis. After this, the intended use and workflow would be described. This is part of the dynamic model. Additionally the intended static model is described based on the context information. This is the place were typical OOA information are held. The (technical, usability or other quality) requirements are then derived from the analysis based on the context of use as well as requirements which are dependent to the chosen dynamic or static concept. The requirements are then documented in guidelines and other process support documents. They are accessed by the developers during their design work. The result is the product or, using versioning or rapid prototyping, a preceding version or prototype. All this is already described as OOD (object-oriented design) and is not original to this model.

From practical use in the German flagship project "Virtual University of Applied Science" (in German "VFH – Virtuelle Fachhochschule"), the project "medin" (Multimedia-Based Distance Education in Medical Computer Science) and the project "WissPro" ("Knowledge-Project: Contextualized Computer Systems") we found that such an ideal process was hard to realize. As known from the description of the software crisis most projects do not start with a proper context analysis. The main focus of our work was to ensure quality even though some initial steps have not made been performed to an optimal extent. We call this approach *"lightweight usability"* [3]. Additionally the ideas of an ISO 9001 [6] compliant test procedure, originally only focusing on usability engineering (see [1], [7], [8]) were adapted to help validating requirements against the context of use.

3.2 A Problem Centered Approach

The objects shown in figure 1 represent the typical engineering phases [9] and ideally would be used clockwise. But the proposed model is flexible enough to allow a reverse approach or starting with the evaluation of early prototypes and going on to the context of use analysis [2] in order to cope with insufficient requirements engineering activities. Practically the use of a product and the evaluation give valuable hints on wether the context of use analysis was precise enough or requirements are not yet valid. Therefore the new object type "potential problem" was added. An identified potential problem results from findings from usage evaluation, e.g. empirical work or user tests which is not yet verified to be a relevant problem. These potential problems then are checked against the already known context information and on its effect on the intended use. The result may be that the problem does not affect the normal use and can therefore be viewed as irrelevant. Another finding may be that not enough analysis data is available to judge the problem. These *potential problems* get the central driving force within the process and allow concentration on user and task relevant problems. It avoids doing quality management on already well done or less important parts of the project.

One application of our approach was the interface design of an e-learning system: most potential problems that were found had to do with navigational issues whereas the readability was almost no problem. With this in mind the quality management process could concentrate on elaborating requirements regarding navigational issues rather than documenting readability criteria that all participants already adhered to. More technical requirements like availability were relevant in this special context too.

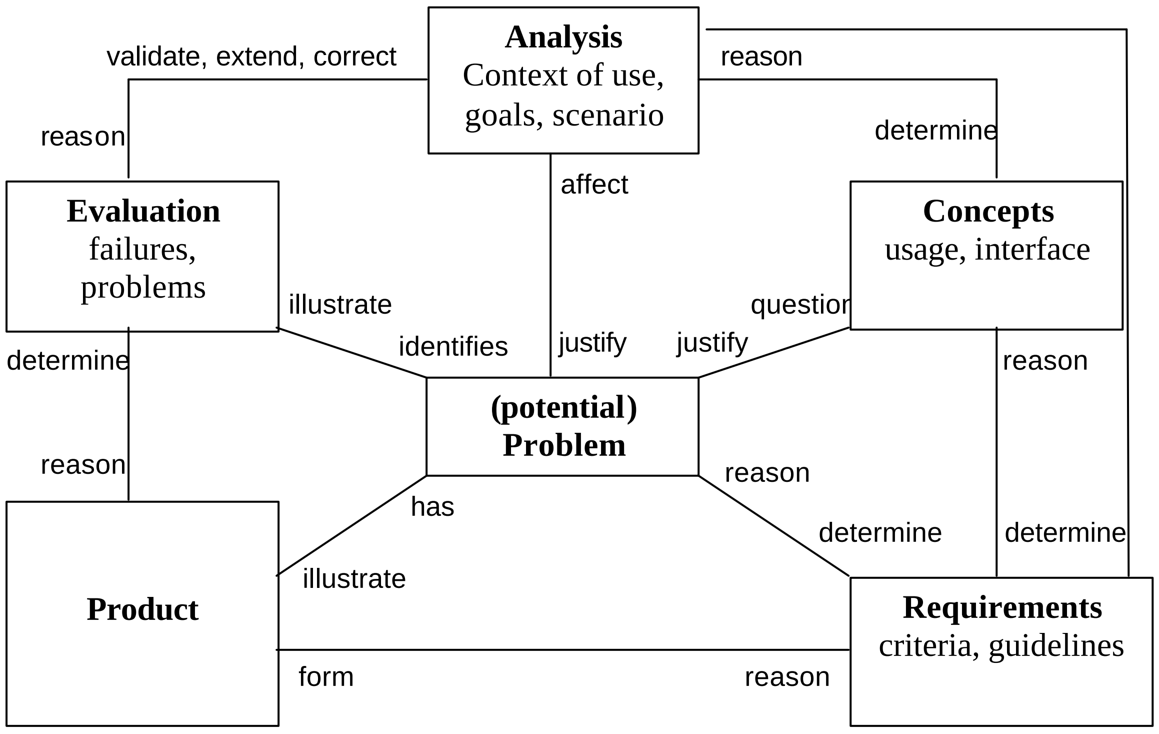
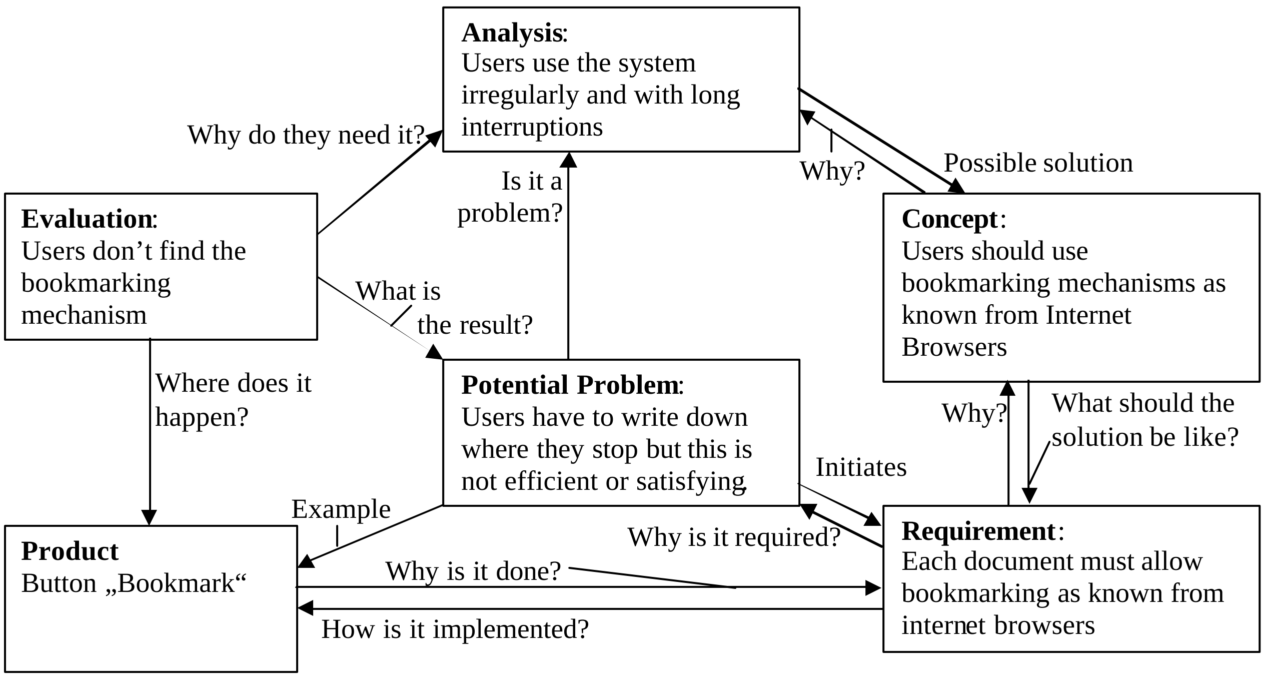


Figure 1: Object model with interdependencies as associations (all associations are “none to many”-associations).



Analysis: Users use the system irregularly and with long interruptions
Evaluation: Users don’t find the bookmarking mechanism
Concept: Users should use bookmarking mechanisms as known from Internet Browsers
Potential Problem: Users have to write down where they stop but this is not efficient or satisfying.
Product Button „Bookmark“
Requirement: Each document must allow bookmarking as known from internet browsers
Why do they need it?
Is it a problem?
Why?
Possible solution
What is the result?
Where does it happen?
Why?
What should the solution be like?
Initiates
Example
Why is it required?
Why is it done?
How is it implemented?

Figure 2: Example with some bi-directional associations

3.3 The Semantic Web

The interconnection of the information classes described above define a semantic web. The annotations in figure 1 show a possible semantic interpretation of the association of different object classes. This semantic web allows backtracking all decisions from the product to the requirements, the concept and its context of use.

The other direction allows propagation of new findings through the development process. To support building up such a bi-directional relational structure the tool automatically generates bidirectional associations between objects.

The example (see figure 2) illustrates this mechanism: While documenting the concept the responsible author should refer to the context of use in order to justify his or her decisions. Then the requirements engineer will refer to the context of use as well as the concept to set up a requirements document and a related development guideline. During development the developers use the development guideline. Within the guideline they will find the links to the reasoning for a guideline rule and may trace back to the context of use information.

Quality managers document their findings as raw, often informal information (like error reports or testing protocols) and derive potential problems. These relate to the context of use in which the problem appears. It is validated as a problem if it obstructs the intended use described in the use scenario. This may lead to a new or more detailed guideline and of course it points to a part of the product where the problem appeared. Therefore the developer may find a link to a potential problem while analyzing a requirement. This potential problem is then illustrated (if needed) by the raw evaluation data to help to understand the problem. This has a valuable impact on the acceptance of such guidelines. Developers tended, at least in our project, to disagree with guidelines which have no articulated reasoning. For a quality engineer it is important to have the reasoning chain available, if developers ask for rationales.

Uni-directional decision chains could be implemented with almost every hypertext system. In our approach the tool support is to automatically add the inverse to all links and inform the target as well as the source object, what kind of object has been associated. In the above example the context of use object literally *knows* which requirements and concepts rely on it. If the context of use has to be updated or corrected, the changes can be propagated through the whole decision chain by looking at the associations related to the changed object. If a requirement is no longer applicable because its foundation in the context of use has changed, it is now possible to track down all dependant decisions. Another example is that if quality engineers find a problem within the product, the product receives a notice on this problem, so while browsing through the product, hints about problems may show up.

The example in figure 2 may illustrate how the tool may help building a semantic web for the process information. Most of the links between different process phases and their related objects can be used both ways even though the documenter may only have had one direction in mind. Making this visible to all participants in the development process and keeping all information in one system is the goal of the proposed tool.

The alternative would be to document the requirements and most of the process data using word-processors and using UML for the system modelling without such connections. CASE tools normally have no space for informal data and no repository for multimedia evaluation data like scenario pictures, plain text usage scenarios or user test videos. This would probably lead to an inconsistent, large and therefore hard to handle mix of documents in different systems .

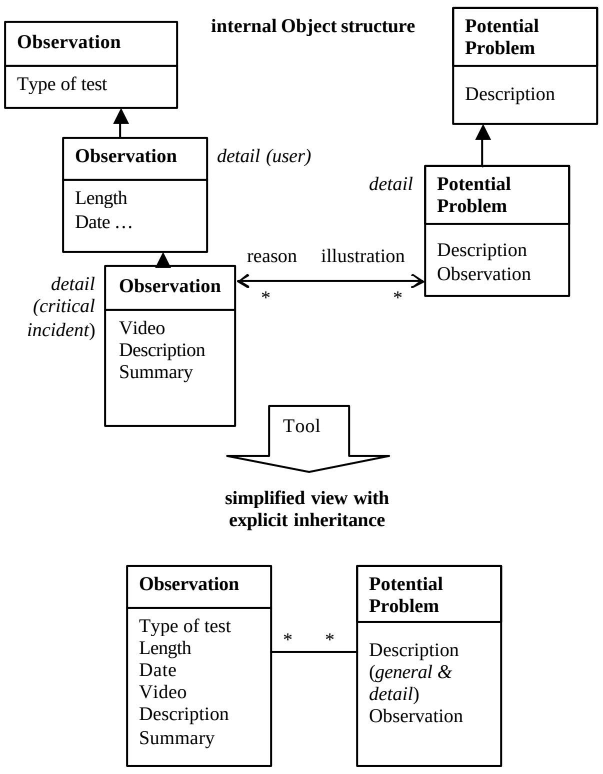


Figure 3: Example usage of explicit inheritance

3.4 Inheritance

The other main idea of the proposed tool is to use inheritance and part-of relations to identify and document even complex context of use and task attributes and all other process data. Instances of the basic object classes described before are structured hiera rchically using object oriented abstraction methods like aggregation and inheritance. The important difference of this framework compared to the existing OOA and OOD techniques is the more holistic approach. It integrates *all* the data from the software lifecycle into one model instead of only the static and dynamic model. But its similarity is the handling of complex data. For example potential problems may be stru ctured in an object hierarchy with different abstraction levels. Higher levels would describe the general impact whereas on a lower, more detailed level a keystroke analysis of the error may be documented.

One practical problem was to use such inheritance mechanisms for users without a background in information mo deling. It was not very realistic to exepct end users or domain experts to get an in-depth understanding of the object oriented information modelling concept. Instead the implicit inheritance of an object hierarchy was made explicit in a simplified view. This means that information from higher levels could be viewed on lower levels without the need of navigating through the object hierarchy. This was crucial in order to integrate the data into the living process instead of a model that would stand apart. Practically special views show certain objects in a self contained form which means that users only have to look at single objects rather than the whole inheritance tree. Figure 3 shows an example on how the different details for a user test observation and a potential problem are specified on diffe rent levels.

Of course this only helps viewing the data and supports minor changes in single objects. For adding objects or constructing a hierarchie the object oriented model must still be understood. But the main use of the tool was to work as an information system, so mainly users *read* what may be relevant for them. But even for the expert users the immediate visibility of inherited information from higher levels helps to avoid constructing contradictions or mu ltiple inconsistent instances of similar objects.

3.5 Flexibility

The last main requirement was the *flexibility* of the approach. In our project the process model was not very elaborated in the beginning. It was not clear which kind of information and to which level of detail would be available. Furthermore the process itself evolved and changed in iterations during the project. Therefore there could not be a conclusive XML-scheme or DTD (Document Type Definition) in the beginning. Object attributes evolved during the process development and therefore the ability to change and maintain the object model itself during its usage becomes a crucial requirement. Changes in the information model should be possible without the need of changing the tool itself. Therefore, authorized users can create new object classes or add attributes at any level during the process and the tool has methods to handle such new objects and attributes based on a flexible rule concept.

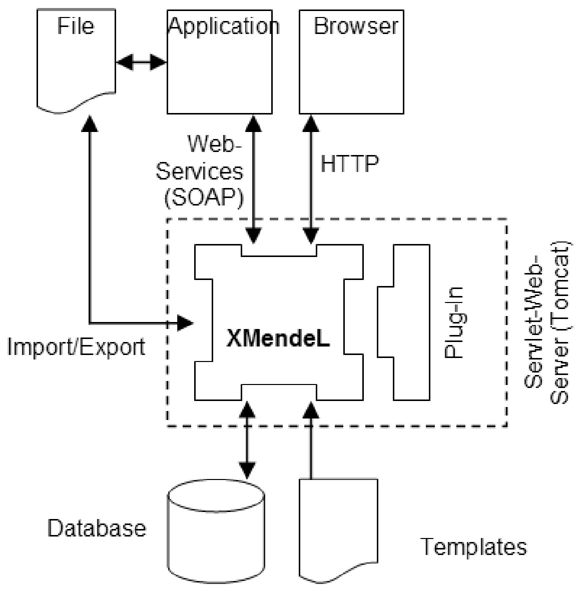


Figure 4: System architecture

4. Implementation

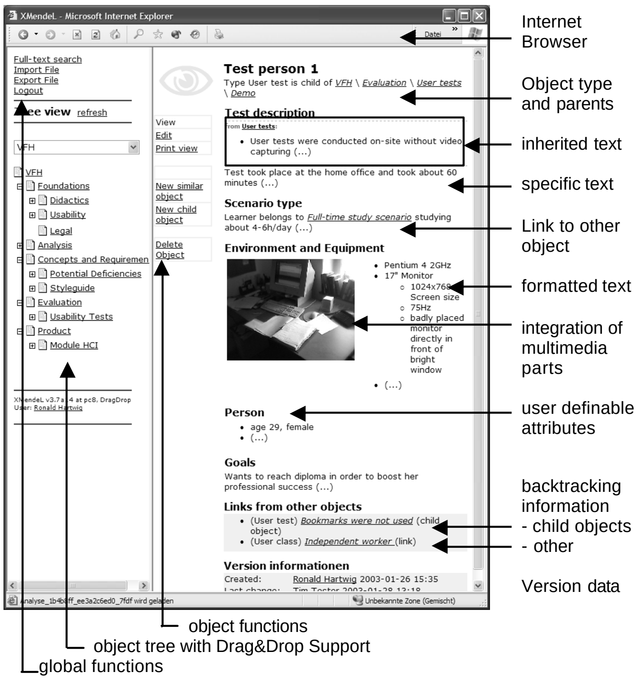
The tool has been implemented using a client-server architecture (see figure 6) in conjunction with simple relational database (mySQL). As a reference to the main concept of inheritance (which has been discovered by Mendel) and the use of XML for a flexible semantic meta-language the tool is called *"XMendeL"*. In order to ease the access to the process data three interfaces are available:

- **Browser**: A browser interface allows entering, viewing and maintaining the data without any additional software installation using an intranet or the internet (see figure 5). Because of the varying technical background of the users a basic toolbar from Microsoft Word™ is part of the interface (see figure 6). It allows to format texts and pictures within the objects. Structural work is done using simple "add child object" or "add attribute" functions. Documents, videos and graphics (e.g. an UML diagram or an evaluation video) are uploaded to the server and embedded into special container objects. Standard objects then may be associated with these non-XML contents. This provides a simple migration option from other (CASE-) tools or word-processors.

- **Export-/Import:** This interface serves the import and export of files which then may be used by external programs, e.g. a LaTeX-typesetting engine. This is useful to make static contents (like a development guideline) available offline (e.g. as a HTML-site) or to export XML-type output for other programs. A flexible, rule-based Import-Interface allows to re-use structured (not necessarily XML formatted) data from external applications.
- **.net/SOAP:** Because of the limited possibilities of the browser interface, especially for inexperienced users, a direct access interface using the SOAP standard is currently under development. Using this applications as a remote access allows users to keep their favorite program, e.g. MS Word, for documentation purposes. The application directly communicates with the XMendeL-System and uses the described inheritance and linking mechanisms. The system restructures contents from the external application and stores them as XMendeL-Objects.

All three interfaces may be used at the same time. Input parsing and output formatting is done using a rule description language (similar to the Cascading Style Sheets CSS from HTML and the XSLT transformation language for XML), which defines rules on how to interpret or to display/write the object contents. There are different views on the same data for different tasks and user groups: e.g. one HTML-view for inexperienced users, another one for usability experts, XML for the file exchange with other programs and finally a LaTeX-view for printing.

Context specific functions to interpret and adapt the data may be added using an open plug-in interface. It allows to add advanced and specialized data processing routines which influence the parsing and formatting of the data within this model without changing the system core.



Full-text search
Import File
Export File
Logout

Tree view refresh

VFH

- VFH
  - Foundations
    - Didactics
    - Usability
    - Legal
  - Analysis
  - Concepts and Requiremen
    - Potential Deficiencies
    - Styleguide
  - Evaluation
    - Usability Tests
  - Product
    - Module HCI

XMendeL v3.7a14 at pc8, DragDrop
User: Ronald Hartwig

View
Edit
Print view

New similar object
New child object

Delete Object

Test person 1
Type User test is child of *VFH* \ *Evaluation* \ *User tests* \ *Demo*

Test description
from **User tests**:
- User tests were conducted on-site without video capturing (...)

Test took place at the home office and took about 60 minutes (...)

Scenario type
Learner belongs to *Full-time study scenario* studying about 4-6h/day (...)

Environment and Equipment
- Pentium 4 2GHz
- 17" Monitor
  - 1024x768 Screen size
  - 75Hz
  - badly placed monitor directly in front of bright window
- (...)

Person
- age 29, female
- (...)

Goals
Wants to reach diploma in order to boost her professional success (...)

Links from other objects
- (User test) *Bookmarks were not used* (child object)
- (User class) *Independent worker* (link)

Version informationen
Created: Ronald Hartwig 2003-01-26 15:35
Last change: Tim Tester 2003-01-28 13:18

Analyse_1b4b8ff_ee3a2c6ed0_7fdf wird geladen
Unbekannte Zone (Gemischt)

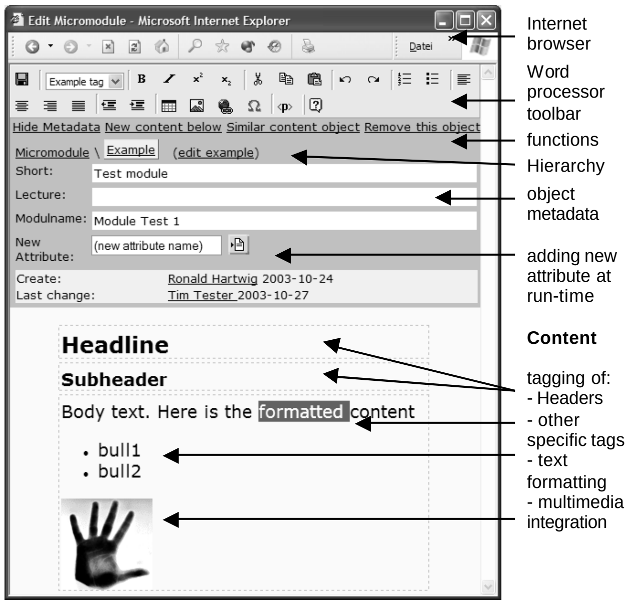
Figure 5: Example screenshot (generic object view)

5. An Integrated Development Environment for Web-based Contents

Because of the flexibility of the rule-based concept other possible applications were found and tested. The system offers flexible objects with a powerful view controller, a simple visual WYSIWYG-type ("What you see is what you get") interface in combination with the object-oriented modelling. This flexible view-controller is a good platform for content management. In combination with the inheritance mechanisms new possibilities on how to store the domain contents arise. In our e-learning projects the desired products were HTML-sites with interactive contents. XMendeL was used as a development and authoring platform in the sense of a content management system (CMS). The final product could be exported from within the system and could then be used without as a stand-alone web site.

Even though the tool was not originally intended to be such a content management system it showed how much such a simple interface (figure 6 shows an example of a task specific edit view) to large scale content databases was needed and appreciated. Even inexperienced users like external physicians which developed parts of the module contents were able to work directly on the product without interfering with the remaining process.

Unlike exitsing authoring software the main advantage of this application is that *all process data* (analysis data, requirements, styleguides, evaluation data) *including* the product itself, could be stored in the same database. Deficient pages of the e-learning modules contained links to the rule they violate. Developers were then able to look up, why this rule was established. Quality engineers could use the database to look up how they decided on special issues earlier and comprehend sometimes long gone (up to 4 years) decisions again.



Short: Test module
Lecture:
Modulname: Module Test 1
New Attribute: (new attribute name)
Create: Ronald Hartwig 2003-10-24
Last change: Tim Tester 2003-10-27

Headline

Subheader

Body text. Here is the formatted content

- bull1
- bull2

Figure 6: Example screenshot (specialized edit view)

6. Conclusion and Outlook

The basic idea, to combine a databased hypertext content management system, inheritance mechanisms from the object-oriented analysis and a server based approach, proved to be unexpectedly applicable and useful in different contexts. The tool is compatible to other processes and does not restrict the user to a fixed model which may be inadequate for his or her special project. The granularity and the amount of data can be scaled with respect to the available resources. The XML-export/import interface allows migration *to* other tools as well *from* other tools. This helps introducing this tool into existing projects and offers a way back if needed. The practical advantages of the tool were the ability to support a simple cooperative content management, the possibilities to structure the heavily interconnected decision making process, espacially the quite informal raw evaluation data like user test protocols.

The system currently holds about 15.000 (!) data objects and is in daily use for the development and (quality) management of online learning material as well as classic class teaching support. It is still under development in order to improve its usability and the visualization of the complex data structures. We hope to be able to offer this system to an interested public in a near future.

7. REFERENCES

[1] W. Dzida, R. Freitag, R.: "Usability Testing - The DATech Standard" In: Wieczorek, Meyerhoff (Editor): Software Quality - State of the Art in Management, Testing And Tools. Springer, New York (USA), 2001

[2] R. Hartwig; M. Herczeg: "A Process Repository for the Development of E-Learning Applications" In Proceedings of the IEEE ICALT 2003, Athens, 2003, pp. 346-347

[3] R. Hartwig; C. Darolti; M. Herczeg: "Lightweight Usability Engineering Scaling Usability-Evaluation to a Minimum?" In Jacko, J., Stephanidis, C. (Publ.): Human Computer Interaction - Theory and Practice (Part I), Lawrence Erlbaum Associates Publishers, London GB, 2003, pp. 474-478.

[4] M. Herczeg: "A Task Analysis Framework for Management Systems and Decision Support Systems" In: Proceeding of AoM/IaoM. 17. International Conference on Computer Science, San Diego, California, August 6-8th, 1999, pp. 29-34.

[5] M. Herczeg: "A Task Analysis and Design Framework for Management Systems and Decision Support Systems" In: ACIS International Journal of Computer & Information Science, Vol. 2, No. 3, September 2001, pp. 127-138.

[6] International Organization for Standardization: "ISO 9001 – Quality management systems - Requirements" International Standard, 2000

[7] International Organization for Standardization: "ISO 9241 - Ergonomic requirements for office work with visual display terminals" Parts 1-17. International Standard, 2000

[8] International Organization for Standardization: "ISO 13407 - Human-centred design processes for interactive systems" International Standard, 1999

[9] B. U. Pagel, H. W. Six: "Software Engineering - Band 1: Die Phasen der Softwareentwicklung" Addison-Wesley, Bonn, 1994