Multimedia Tools and Applications, 27, 5–21, 2005

CRITON: A Hypermedia Design Tool

PARIS AVGERIOU* pavger@softlab.ntua.gr
National Technical University of Athens, Department of Electrical and Computer Engineering, Software Engineering Laboratory, 15780 Zografou, Athens, Greece

SYMEON RETALIS retal@softlab.ntua.gr
Department of Computer Science, University of Cyprus, 75 Kallipoleos Str., P.O. Box 20537, CY-1678 Nicosia, Cyprus

Abstract. The WWW has turned into a development and run-time environment for large-scale and complex applications. Such sophisticated applications are being deployed in increasing numbers without having been developed according to appropriate methodologies, tools and quality standards. The reason is not only that the hypermedia industry resists to utilize formal methods, but also that these methods and corresponding tools are very few and of dubious standards. The consequence is that the hypermedia applications being developed are of poor functionality and lack qualities such as modifiability, usability and maintainability. Especially the design phase is one of the phases that lack sufficient support from methods and CASE tools. This paper presents CRITON, a cross platform tool, built to support a hypermedia design method within an integrated environment. CRITON manages all three aspects of hypermedia design: conceptual design, navigational design and graphical user interface design, utilizing well-established theories and practices from software as well as hypermedia engineering. It employs these designs to generate a preliminary, exemplary form of the hypermedia application for the purpose of assessing the designs before the implementation phase.

Keywords: hypermedia design, web site design, CASE tool, design tool, navigation, page metaphor

1. Introduction—Motivation

The ubiquity of the Internet and the World Wide Web in all the fields of the new knowledge-based economy is setting new demands and requirements for the engineering of hypermedia applications. The evolution from hand-crafted personal web-pages to the multi-billion market of complex e-commerce, e-learning and e-government applications has happened too fast and is not to be taken lightly while developing such applications. Regretfully, ad hoc methodologies that were originally used for web site development, still express, to a large extent, the current state of application development in the Web environment [13, 17, 18, 31]. It is more than evident that web sites cannot be designed and implemented like they used to be. Instead these trial-and-error approaches must relinquish to methodical and systematic engineering approaches for hypermedia development [17–19, 23]. Therefore the construction of high quality hypermedia applications within specific time and fund limits demands effective development methodologies.

In accordance with the definition from software engineering practices, a *methodology* for engineering hypermedia applications is a set of process models, methods, tools,

*To whom all correspondence should be addressed.

documentation aids and guidelines that help the developers in building quality hypermedia applications, respecting the constraints imposed in time and resources. Such a methodology, of course, is not a mere collection of elements but advocates specific development philosophy and offers specific benefits, such as risk mitigation, quality assurance, the ability to manage change, etc. [21]. Of all the different constituents of methodologies, this paper focuses on CASE (Computer Aided Software Engineering) tools, which can significantly promote the efficiency of development work, as they provide automated or semi-automated support for processes and methods [21].

Unfortunately software engineering techniques can't be applied in hypermedia engineering, due to the vast differences between hypermedia and conventional software systems [9, 20]. To make matters worse, in contrast to generic software engineering, where significant progress has been made the past twenty years, there is still a great deal of work to be done on formalizing process models, and defining methodologies or design methods for hypermedia applications [18]. The same argument stands for the corresponding tools, as hypermedia application processes lack CASE tools that could support the analysis, design or evaluation phases [17, 25, 26], and only provide low-level implementation tools, such as web page and web site editors.

In this paper we do not intend to tackle the hypermedia development process as a whole but rather to focus on the design phase by presenting a CASE tool, named CRITON that supports a simple design method. Good design is crucial, as it can provide a blueprint for the communication between all the stakeholders, i.e., the development team, the clients, managers, etc. Moreover, it can offer guidelines to the implementers and accelerate the implementation process. It also can provide an analytical guide for the maintenance of the hypermedia application, and is the best way to ensure scalability of the application by providing a transferable abstraction of the system. An efficient design tool is required in order for the design process to be supported in a customized, uniform, integrated environment. The design method supported by CRITON follows the object oriented hypermedia design principles and is a stepwise method, where the interim products are the conceptual, navigational and interface design. It is noted that even though hypermedia applications include many kinds of applications such as web sites, CD-ROMs and information kiosks, we basically refer to the category of web sites in this paper.

The rest of the paper is structured as following: In Section 2 the relevant research work in the area of hypermedia design tools is given, including both research approaches, as well as commercial products. In Section 3, an overview of the three steps of the design process are described, followed by an analysis of the CRITON CASE tool in Section 4. The tool is presented through a case study dealing with the design of web-based courseware, a rather popular hypermedia application nowadays. In Section 5 the evaluation of the tool from its up to date use in hypermedia development projects is presented and some thoughts about its future expansion are shared. Finally, Section 6 contains some concluding remarks.

2. Related work

CRITON is a hypermedia *design* tool that supports the whole of the design process, incorporating conceptual, navigational and interface design. It adopts the data models that this

design method specifies and follows a step-by-step design process. It provides an integrated environment with a uniform interface and embraces all three steps of the design method in a whole. Relevant research and development work includes three categories of tools: hypermedia CASE tools that support design methods or methodologies; commercial web site development tools that offer design facilities; other generic tools that can be used for designing hypermedia. This section presents these three categories of tools and states the differences between them and the proposed design tool.

2.1. Hypermedia CASE tools

At present there are few hypermedia design methods and methodologies that have been originated in academic research centers and aim to provide a systematic approach in hypermedia development. These methods build on the initial hypertext and hypermedia models, like the ones described in [2, 8, 10, 11]. In some of these approaches, customized tools have been especially constructed so as to provide a solid development framework and ease the work of the development team [4, 5, 26, 30]. The most important of these approaches OOHDM-Web, RMCase, and WebRatio are briefly described hereafter.

A successful and well-established method for hypermedia design is the Object-Oriented Hypermedia Design Model (OOHDM) [27–29], which uses object-oriented techniques to produce a hypermedia design model and leverages the engineering of complex well-structured hypermedia applications. A tool that was built to support this design model is OOHDM-Web [25, 26], which provides support for navigational design, abstract interface design, as well as automatic generation of web pages. CRITON offers roughly the same features with OOHDM-Web, though in CRITON the design takes place visually through a graphical user interface, while OOHDM-Web uses text configuration files and command-line 'make' programs. Also CRITON supports the conceptual design phase, which is something that OOHDM-Web lacks.

A methodology that provides step-by-step hypermedia development is the Relationship Management Methodology (RMM) [13–15]. RMM offers complete representation of the semantic schema and the navigational schema, and follows the traditional Entity-Relationship model to standardize the conceptual and navigational design but gives limited support to the interface design. The Relationship Management Methodology is aided by the RMCase [5], a graphical CASE tool that implements the design steps of the methodology. In specific it supports two design steps: conceptual design through Entity-Relationship design and 'Slice' design, which is the design of the entity details and navigational design. Compared to CRITON, it can be claimed that RMCASE provides a similar conceptual and navigational design, and they both offer a preview of the hypermedia application for inspection and evaluation. On the other hand CRITON offers a much more advanced data model, based on the Object-Oriented paradigm, uses the well-established Unified Modeling Language [1, 24] and also provides a complete interface design phase, which is absent in RMCase.

The Web Modeling Language (WebML, http://www.webml.org/) [3] is a modeling language especially designed to support abstract modeling of web sites and includes conceptual, navigational and interface design, as well as a type of user modeling. The corresponding tool that supports WebML is called Webratio [http://www.webratio.com/]. Webratio is also

quite similar to CRITON, their differences being that Webratio performs interface design through textual coding with XML, while CRITON achieves this graphically.

2.2. Web site implementation tools

Concerning the web site implementation tools, most of them put main emphasis on web site implementation and support design only in a minimal and simplified way, e.g. by providing elementary navigational diagrams. Having excluded the simple web page editors that provides HTML coding, the rest of the web site implementation tools can be classified into two categories [25]:

1. *Web site editors*, which except for supporting web page editing, also provide a mechanism for the easy creation of a navigational design schema as well as for the management of a set of pages like a file management system does. Examples of such tools are Microsoft FrontPage [http://www.microsoft.com/], Macromedia DreamWeaver [http://www.macromedia.com], NetObjects Fusion [www.netobjects.com] and CyberStudio [http://www.golive.com]. Most of these tools also provide a more structured development by defining a common look in the form of templates. In these tools though there is a complete absence of a conceptual design phase, as well as abstract interface design, since the focus is on the implementation.
2. *Web site building environments*, which create hypertext documents at run-time by instantiating templates. Examples of such tools are Vignette Story Server [www.vignette.com], and Allaire Cold Fusion [www.allaire.com]. These tools are also effective in automatic generation of web sites, using various template mechanisms but are completely deficient in providing higher-level design mechanisms.

In conclusion, web-site implementation tools are very useful, if the development team has *already* made the conceptual, navigational and abstract interface blueprint of the hypermedia application, as the whole design phase is very poorly supported by them, if supported at all.

2.3. Generic tools that support hypermedia design

This category is comprised of tools like modeling tools that serve general-purpose software development but can be customized for use in hypermedia design as well. For example, the Unified Modeling Language, which is becoming a de facto standard for modeling languages in the software industry is supported by a number of tools, and can effectively be used for designing hypermedia application at an abstract level. Examples of such tools are both commercial tools such as TogetherSoft Control Center http://www.togethersoft.com/products/controlcenter/index.jsp, and open source tools such as ArgoUML [http://argouml.tigris.org/].

A tool that goes one step further from the above is Rational Rose [www.rational.com/rose], a visual modeling tool for software development that supports the UML, and can be extended to perform other activities such as business modeling, data modeling and web modeling [22]. This tool incorporates the Web Modeler, which is a set of functions, templates and pre-defined UML concepts that help a development team to model a web application. Conceptual design can be performed quite effectively, as well as navigational design with the

constructs available by the Web modeler, while interface design is not supported. It is obvious that such tools can aid in hypermedia design but only partially, since they fall short compared to specialized hypermedia design tools like CRITON that offer more specialized design features like abstract interface design or preview generation.

3. A hypermedia design method overview

CRITON advocates an object-oriented design method specifically created for the needs of hypermedia design. It proposes a stepwise design process, as shown in figure 1: Conceptual Design, Navigational Design and Interface Design. The intermediate products of each step are validated according to guidelines for hypermedia design (checking structural, navigational, aesthetics and functional issues). The whole design process is considered to be iterative, where in each iteration loop the designs are evaluated and the feedback from the evaluation is used for their improvement, until they reach the desirable level. The evaluation is based on the tool's ability to generate the preview of the hypermedia application, which is a semi-functional prototype of the application under development.

In order to explicate the design steps we present some examples of an exemplar hypermedia courseware application that is comprised of an electronic book and other resources.

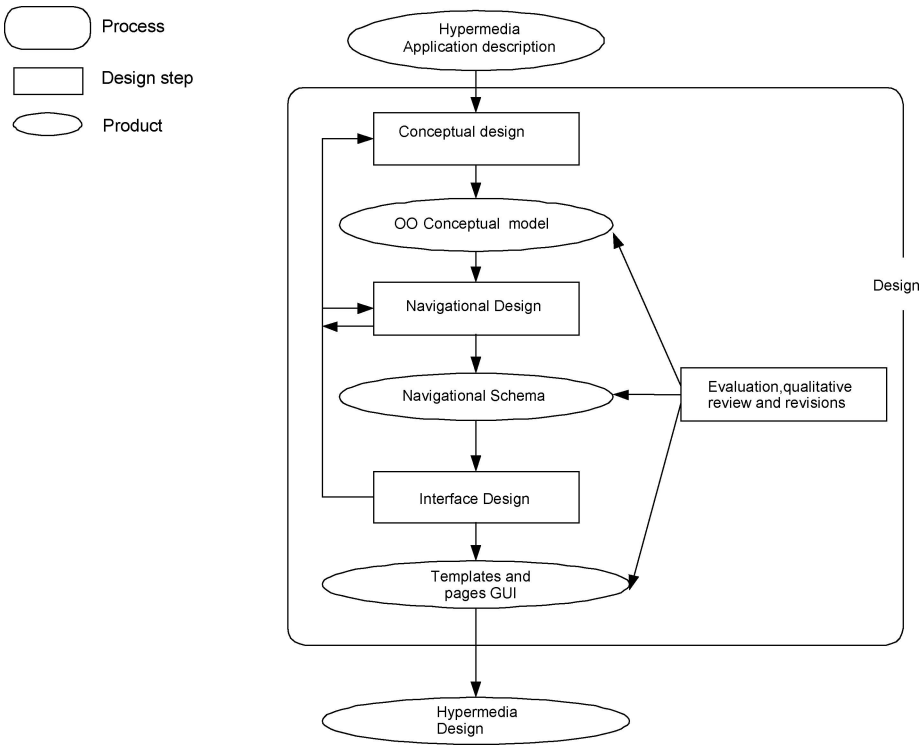

Figure 1. The three design steps.

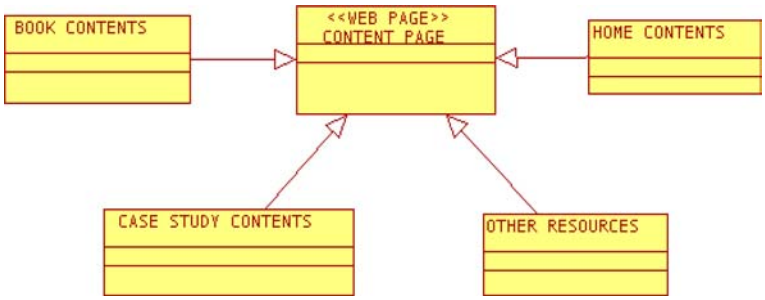

Figure 2. Class diagram of the courseware content pages.

3.1. Conceptual design

In this step, the description of the hypermedia application is transformed into a conceptual design following an object-oriented data model and a conceptual design framework. According to this framework, a hypermedia application is a mosaic of resources such as hierarchically arranged sets of pages, dynamic pages created on-the-fly, site maps, search engines, communication tools, etc. For each type of web page, the designer must specify the elements that comprise it, such as media elements, active behavior, etc. To serve the needs of the object-oriented data model, the design method has adopted the Unified Modeling Language [http://www.rational.com/uml], a graphical modeling language widely adopted by the software industry and strongly supported by the Object Management Group [http://www.omg.org]. In particular, conceptual design adopts *use case diagrams* and *class diagrams* from the UML repository. Use case diagrams depict the ways that the hypermedia application is used by external *actors*, or in other words use cases specify the system requirements. Class diagrams contain object-oriented constructs such as classes, interfaces, packages (grouping mechanisms), and various relationships between them and aim at specifying in detail the hypermedia application structure at an abstract level. Figure 2 depicts a class diagram using the UML notation that shows some classes of specific pages of contents and how they all inherit from the class "CONTENT PAGE" of the stereotype "WEB PAGE".

3.2. Navigational design

In this step the navigational schema of the hypermedia application is analytically designed, so that it is clearly specified how web pages are inter-connected with hyperlinks. The data model of the navigational design contains web pages, single and bi-directional hyperlinks. The navigational design provides a way of checking the implementation of all the hyperlinks in the final product. More importantly it facilitates the maintenance of the web site, especially when web pages are added or deleted and hyperlinks to and from them have to be updated. In this way, the well-known problem of 'dangling' links can be avoided. The navigational structures proposed for this kind of design, are well accepted by many hypermedia design approaches, such as HDM [7, 8], RMM [13–15] and OOHDM [27–29]. More specifically they are: (a) *indices* that provide direct access to every indexed node,

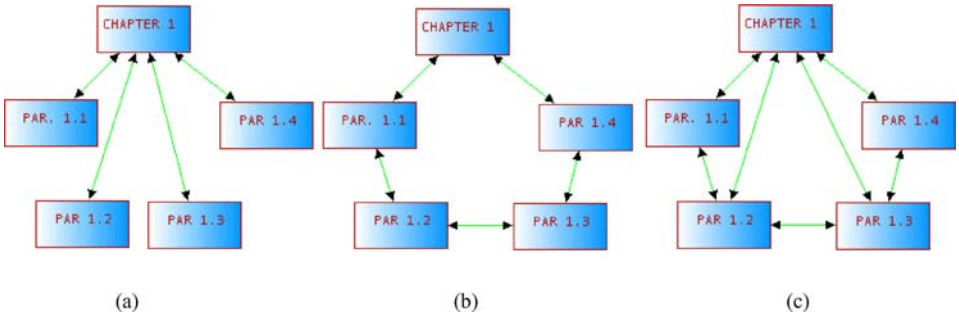

Figure 3. Navigation through the paragraphs of chapter 1 with (a) indices, (b) a guided tour, (c) an indexed guided tour.

(b) *guided tours* which are linear paths across a number of nodes and (c) *indexed guided tours* which combine the two previous structures. The navigation through the paragraphs of a chapter of an on-line book is shown in figure 3 using indices, a guided tour and an indexed guided tour.

3.3. Interface design

In this step, the Graphical User Interface (GUI) of the hypermedia application is designed, that is the content, layout and 'look and feel' of the web pages. Hypermedia application interface design is ruled by the principles of the *page metaphor*, a practice taken from multimedia engineering where it has been extensively adopted and used. Page metaphor is used to specify the page components with graphic symbols and deploy them on the screen showing their layout. Therefore, with the use of graphical semantics, the design depicts the page form just as it will be implemented. The data model for the interface design contains six kinds of page components: plain text, multimedia elements, active elements, hyperlinks, frames and forms. The designs made are actually re-usable page templates. For instance, if we design the page template of one paragraph of an on-line book in a hypermedia application, then all the other paragraphs of the book might have the same look, using the same components with the same layout, have the same frames, etc. A page template of a paragraph of the electronic book in our example is shown in figure 4. During the interface design, except for designing page components and their layout, we define certain metadata on them. All page components can have metadata that describe various aspects of them, like author details, type or format, etc. The definition of metadata during the design phase is of paramount importance as it facilitates the management of the page resources and their accessibility and reusability.

4. The CRITON CASE tool

CRITON has been implemented in Java and can be considered as a 100% pure Java application, and thus to a large extent cross-platform. In order for the design steps to take place

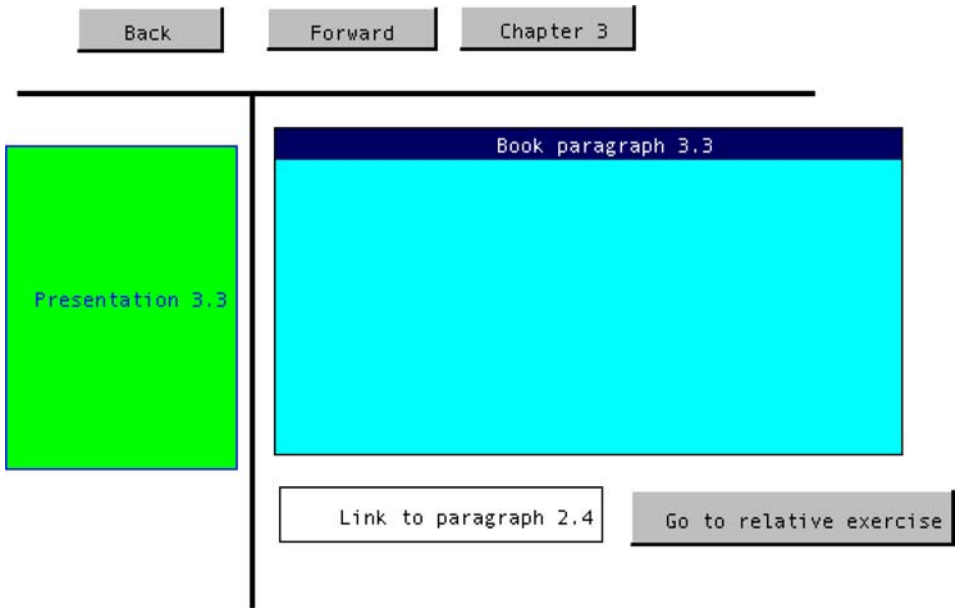

Figure 4. Page template for the paragraph 3.3 of the on-line instructional book.

concurrently and in parallel, the tool adopts the model of Multiple Document Interface [32]. This model is instantiated by having multiple internal windows, each of which represents a different design process. CRITON's technical infrastructure comprises many of the latest Java APIs, like Swing, 2D Graphics, and the JPEG encoder/decoder.

CRITON generally follows a design philosophy which is often met in relevant environments with standard GUI components like menus, menu bars, project trees, design toolbars, and design frames bearing design windows, as seen in figure 5. Except for the standard 'File' and 'Edit' menus, there is also a 'Look & Feel' menu where the user can alter the general appearance of the application by selecting one of the options that Java provides. The 'Build' menu refers to the tool's ability to generate a preview of the hypermedia application.

The project tree is a tree structure that represents the hypermedia application design. The nodes of the first level represent the three design steps, whereas the nodes of the following levels represent the products of each step. In particular the second level nodes are the different designs of each step and all other nodes have a special meaning according to which of the three designs they belong to. For instance the children of a node that represents a navigational design are its web pages. Finally there are certain operations that can be performed on each node, e.g. it is possible to export the designs represented by the second level nodes into the JPEG graphical format.

The design toolbars contain all the elements used for the implementation of each design. There are three different toolbars corresponding to each design category and they are activated according to which design is currently being edited. The elements in each toolbar belong to the respective data model; e.g. the conceptual design toolbar contains

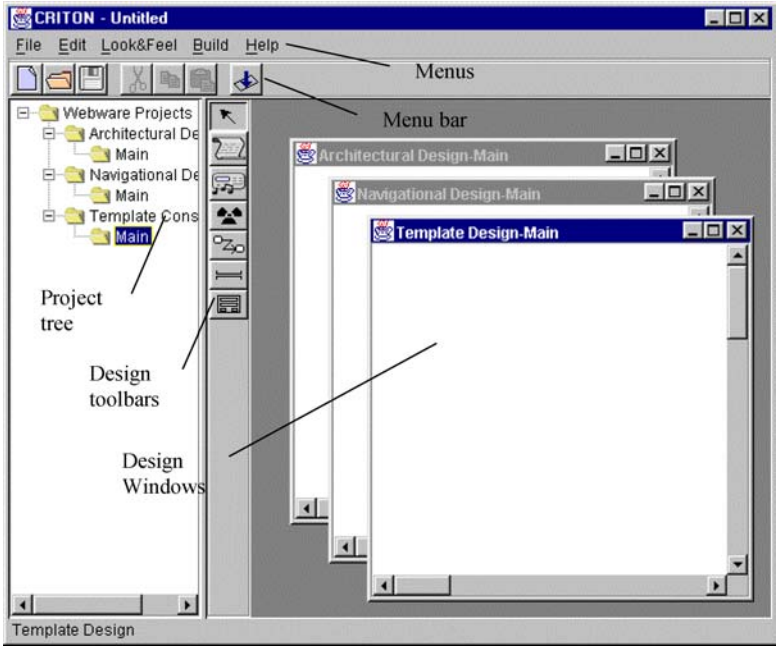

Figure 5. The initial window of CRITON.

the UML graphical symbols. The design process takes place with the designer selecting an element from the toolbar, placing it onto the design windows and then specifying some of its characteristics, such as name, documentation or other specification data. All the elements specified in the designs can of course be dragged and dropped, deleted, cut, copied, pasted, renamed and have certain specifications viewed and edited. These specifications depend on the semantics of each element, in accordance to its data model.

All the designs produced during the design steps described earlier can be exported from the tool by generating JPEG graphical format images. This means that the development team can insert these images into the project deliverables and reports, which are usually Word or PDF documents, HTML files, etc. Therefore the designs can be easily embedded in project documents and communicated to the various stakeholders, acting as a transferable abstraction of the hypermedia application under development.

4.1. Conceptual design (or architectural design)

It takes place with the use of UML and the object-oriented design framework. The toolbar of the conceptual design involves the following elements:

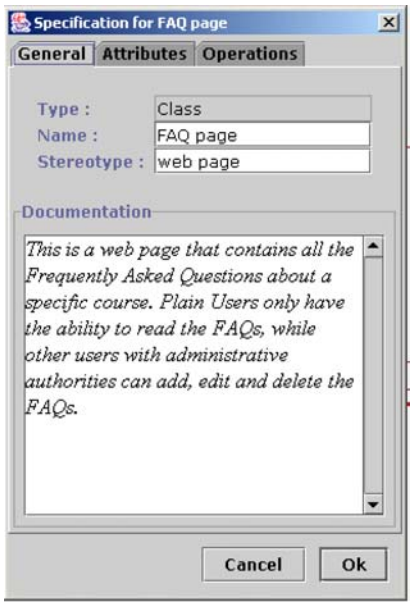

Figure 6. Specification of the FAQ web page.

class, package, interface, actor, use case, note, association, dependency, link to note, aggregation, generalization, uni-directional association and refinement/realization. The first six elements are object-oriented entities, that have specific semantics as defined by UML and the rest are relationships between them. The specification of these elements, according to their semantics, may include name, stereotype, documentation, attributes, operations, cardinalities, and roles. Figure 6 depicts the specification of a class that represents a web page containing "Frequently Asked Questions" about a course. CRITON provides strict type validation concerning the relationships that are allowed between elements according to the specification of UML [24]. For example a designer cannot relate a class and a package with an aggregation relationship, as the modeling language forbids it.

During this design process, the designer attempts to describe the hypermedia application in a conceptual level and make an abstract but complete representation of it. The designer needs to follow the object-oriented design framework and analyze the hypermedia application into components, and the components into web pages. The web pages that are specified in this step are automatically created as modeling elements of the next design step, the navigational design. What remains to be done in the navigational design is to position these web pages in diagrams and connect them with hyperlinks.

4.2. Navigational design

It uses the aforementioned navigational structures, i.e., indices, guided tours and indexed guided tours to describe the way, web pages are connected with hyperlinks. The toolbar of the navigational design involves the following elements:

web pages, single hyperlinks and bi-directional hyperlinks. Bi-directional hyperlinks between two pages are equivalent to two single ones of opposite direction between those pages. The specifications of the web pages include name, URL, documentation and page template, as illustrated in figure 7. The page name needs to be unique in each design so that no two different pages have the same name in the web site, even though the same page may appear in several designs. This means that two page objects that have the same name actually refer to the same page. The URL is the Internet address of the page and can be either a complete URL or a relative address, for example regarding the starting point of the site. Finally the page template defines the name of a GUI template that this page is related to. This is important in order for the page to have its interface designed according to that template. For example all the pages that are paragraphs of the on-line book will be related to the "book paragraph" template. The designer can either specify a new page template or select one from a scroll-down list. The page template field in the page specification is necessary for the next design step, so every page must be related to a template. In the interface design, initially each template is designed and after that the design continues for every page that refers to that template. In the project tree, under each navigational design, we can see all the web pages of that design as leaves. These leaves can be used to create the navigational designs for the corresponding web pages. The design process follows a top-down tree-like flow: first the designer makes the navigational design for the starting

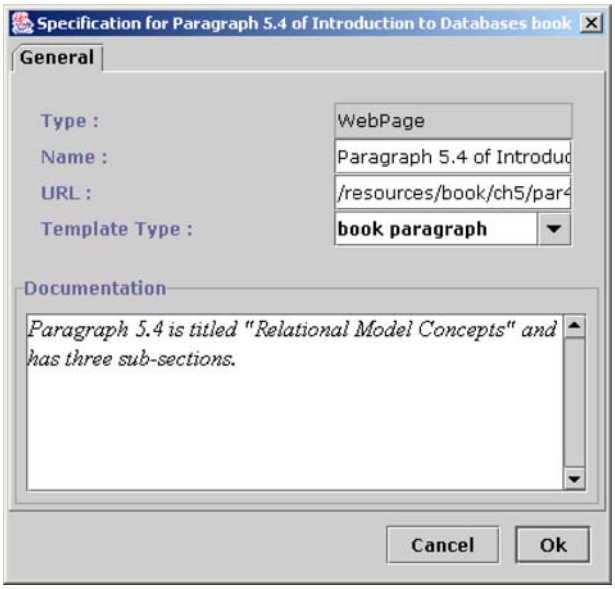

Figure 7. Specification for a web page representing a paragraph of the on-line book.

node (probably the home page of the web site) and then recursively the designs for all the pages that the starting node is connected to.

4.3. Interface design

After the navigation of the whole hypermedia structure has been specified, the development team needs to design the generic interface of the page templates, as well as the specific interfaces of the pages themselves. The interface design in general takes place with the use of the page metaphor. The toolbar of the interface design involves the following elements:

plain text, multimedia elements, active elements, hyperlinks, frames and forms. All these elements can also be considered as part of the UML semantics. In particular these elements are defined as UML class stereotypes according to the UML extension mechanism, i.e., they are classes with some extra semantics. The specification of these elements includes name, documentation and various metadata. The metadata are different for each modeling element and can suggestively be format, author, size, file name, run time environment, etc. Embedding the metadata into the designs is a very useful feature, as it allows fast and easy access and modification and a single place of storage for the metadata. Figure 8 depicts the specification of metadata for an active object. The interface design commences from the design of the page templates, as these have been specified during the navigational design and carries on with the interface design of all the pages. When a page's template is specified in the navigational design, a new blank interface design for the template is automatically created and inserted in the project tree. Also another blank interface design is created for the page itself and inserted in the project tree as a child of the former. If the template already exists then only the second design is generated. In this sense the tool offers full consistency between the navigational design and interface design compelling, that each web page in

Figure 8. Specifying metadata for an active object.

the navigational design must specify its template interface design for the next design step. When a template design has finished, then all of its children can import that design and start customizing it.

The generation of the hypermedia application preview. The three design steps of the design method that CRITON supports, generate three different products: the conceptual design, the navigational schema and the interface design. Based on these products alone, the development team can proceed with the implementation of the hypermedia application. In order to close the gap between the design phase and the implementation phase, CRITON generates the preview of the hypermedia application, which is a set of first-cut web pages inter-connected with hyperlinks. In detail the tool creates a web page for every page designed during the navigational design and connects it with hyperlinks to the appropriate pages. The contents of such a page is the name given by the designer, which is also the page's title, the hyperlinks to other pages and a JPEG image, which corresponds to the interface design of this particular page. For reasons of clarity, it is noted that this JPEG image merely depicts the graphical user interface as a static image, in order to give a first-cut idea of how the page will look like, when it is implemented. For instance the preview of paragraph 3.3 of the on-line book is shown in figure 9. The preview conceptualizes an initial form of the hypermedia structure and interface of the hypermedia application, which is very close to the final implementation and many useful conclusions can be derived from it and fed into the design

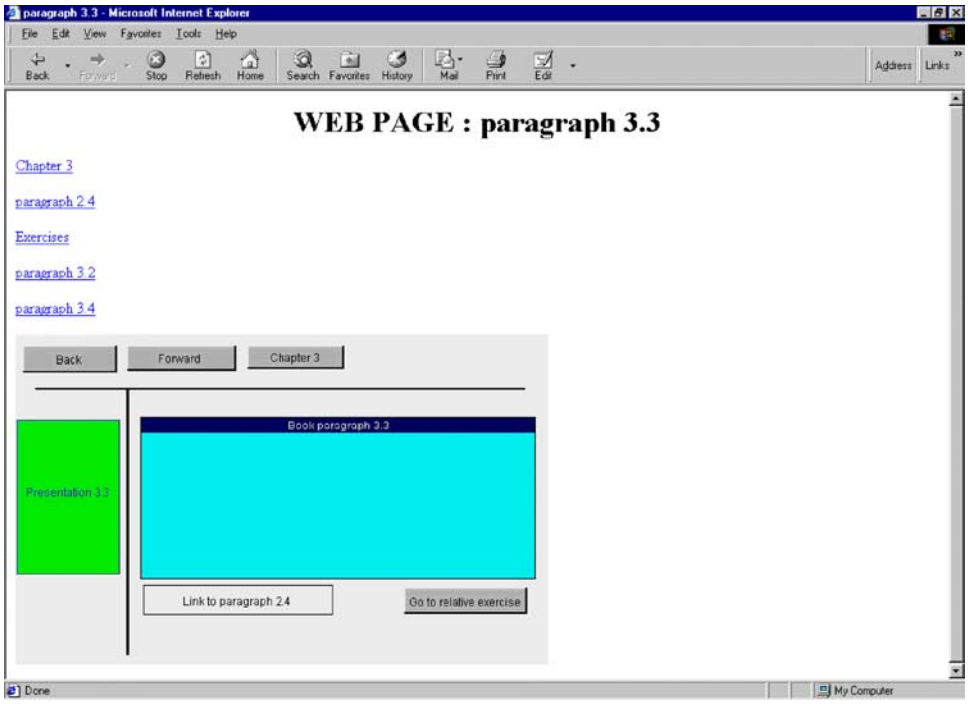

Figure 9. HTML page created by the production of the preview for the paragraph 3.3 of the on-line instructional book.

process. When the evaluation of the preview concludes that the hypermedia application satisfies the goals specified during the analysis phase of the engineering methodology, the implementation may begin. All that is left to be done for the implementation is to build the single components of the page and insert them exactly as they have been specified.

5. Evaluation

CRITON has been used in small-scale web-based hypermedia development projects during the Fall Semester of 1999, 2000 and 2001 for the needs of the undergraduate course "Software Engineering" and the postgraduate course "Web Engineering" at the National Technical University of Athens. The students that took part in the hypermedia application development had a solid computer science background and some of them had already been involved in hypermedia development projects in the past.

The design tool has been evaluated in this context. The evaluation method used was qualitative and was based on focus group interviews, which were applied on project teams of 3 students, after they had completed their course projects. The content of the interviews concerned fundamental usability criteria such as: how quick and easy is it for a user to learn to use the system's interface (*learnability*)? Does the system respond with appropriate speed to a user's request (*efficiency*)? Can the user remember how to do system operations between uses of the system (*memorability*)? Does the system anticipate and prevent common user errors (*error avoidance*)? Does the system help the user recover from errors (*error handling*)? Does the system make the user's job easy (*satisfaction*)? Other issues covered in the interview included the level of adequate documentation material and level of support to the design method and its three steps. The students were also asked whether the tool's capability of exporting the designs in the JPEG format and generating the hypermedia application preview were helpful in the design process. Finally the students were invited to propose enhancements to the tool's features and capabilities.

The evaluation of the tool under this test bed has given some encouraging qualitative results and some valuable remarks for the future evolution of the tool. The development teams have noted that it is quite important to use a uniform environment that embraces the whole of the design process, which they need to follow. The data models chosen were also appraised positively: most of the students were already familiar with UML as it has become part of several courses and the navigational and user interface data models were considered straightforward. Also the platform independence has enabled the tool to run in different hardware and operating systems, thus alleviating the limitations of hardware or software dependant applications. Moreover there were some positive comments about the tool's ability to export all designs in the JPEG format and therefore making it easy to embed designs in web pages or other document formats as project deliverables. The trade-off for platform independence is reduced run-time performance, especially when there are computationally intensive tasks like for example when the JPEG encoding is taking place. That can be diminished by compiling the code in a specific platform, although this would contradict our cross-platform philosophy.

Although in this first version of CRITON we attempted to cover as much of the hypermedia application design process as possible, there are a few additions and improvements

that can be made. The next versions will provide enhancements in matters like the design of active elements (CGI Scripts, Javascripts, Java applets) with UML activity diagrams, the full automation of report-generation, and the optimization of the interface data model. There will also be an enhancement to the hypermedia application preview, so that the implementation can actually take place by editing the automatically generated web pages of the preview. Finally, the tool will be supported by on-line help and the metadata of the interface elements will be improved so that they comply with well-accepted or official standards. These standards can either be general such as the Dublin Core Metadata Element Set [6] or specific such as the MPEG-7 for multimedia, or the IEEE LTSC Learning Object Metadata [9], for e-learning applications. The design tool CRITON will continue to evolve in order to improve the engineering approach in designing hypermedia applications. The tool is currently in the process of being integrated into an open source project, so that its source code can be shared, and it can be used, evaluated and extended by interested parties. Also, in later stages a more thorough evaluation will be conducted in order to generate more substantial and objective feedback on all aspects of the tool.

6. Concluding remarks

The software industry and research institutions are spending a lot of resources in research and development, in order to create hypermedia applications of high quality. However in most cases the development teams use an ad-hoc approach rather than a well-formed methodology of hypermedia engineering, which would guarantee end-quality to some extent. The development of hypermedia applications is crucial, as all of these efforts must be based on sound methodologies, which will guarantee that the final product meets certain quality criteria. Since the risk of failure must be minimized at all circumstances, the use of methodologies is not only useful but also rather imperative. Moreover, tools that support methodologies can be extremely important because they ease and speed up the development process, and assure the correct use of it.

The CASE tool CRITON has been constructed to support a specific three-step design method for hypermedia applications. It combines three design steps in an integrated environment and it relates these steps making the hypermedia application design, a consecutive and iterative process. For the conceptual design, it uses a standardized object-oriented modeling language, the UML, whereas for the navigational design it uses a widely adopted data model. The interface design is based on the page metaphor, which is also acknowledged between the hypermedia developers and introduces the specification of metadata during the design. Finally the hypermedia application preview is a function that allows the development team to examine a depiction of the final product and receive valuable feedback from it.

Acknowledgment

The authors wish to thank the reviewers of this journal for their valuable and insightful comments.

References

1. G. Booch, J. Rumbaugh, and I. Jacobson, The UML User Guide, Addison-Wesley, 1999.
2. M.A. Casanova, L. Tucherman, M.J. Lima, J.L.R. Netto, N. Rodriguez, and L.F.G. Soares, "The nested context model for hyperdocuments," in Proceedings of Hypertext'91, San Antonio, 1993.
3. S. Ceri, P. Fraternali, and A. Bongio, "Web modelling language: A modelling language for designing Web sites," in Proceedings of WWW9 Conference, Amsterdam, May 2000.
4. J. Conklin and M.L. Begeman, "gIBIS: A hypertext tool for exploratory policy discussion," ACM Transactions on Office Information Systems, Vol. 6, No. 3, pp. 303–331, 1988.
5. A. Diaz, T. Isakowitz, V. Maiorana, and G. Gilabert, "RMC: A tool to design www applications," The World Wide Web Journal, Issue 1, Dec. 1995.
6. The Dublin Core, "Dublin Core Metadata Element Set Reference Description, v 1.1," The Dublin Core 1999 (http://purl.org/dc)
7. F. Garzotto, L. Mainetti, and P. Paolini, "Navigation in hypermedia applications: Modeling and semantics," Journal of Organizational Computing and Electronic Commerce, Vol. 6, No. 3, pp. 211–238.T., 1996.
8. F. Garzotto, D. Schwabe, and P. Paolini, "HDM- A model based approach to hypermedia application design," ACM Transactions on Information Systems, Vol. 11, No. 1, pp. 1–26, 1993.
9. F.G. Halasz, "Reflections on notecards: Seven issues for the next generation of hypermedia systems," Communications of the ACM, Vol. 31, No. 7, pp. 836–852, 1988.
10. F.G. Halasz and M. Schwartz, "The dexter hypertext reference model," in Proc. of the NIST Hypertext Standardization Workshop, J. Moline, D. Benigni, and J. Baronas (Eds.), Feb. 1990, pp. 95–133.
11. L. Hardman, Dick C A. Bulterman, and Guido van Rossum, "The Amsterdam hypermedia model: Extending hypertext to support real multimedia," Hypermedia, Vol. 5, No. 1, pp. 47–69.
12. IEEE Learning Technology Standards Committee, "Draft standard for learning object metadata," [http://ltsc.ieee.org/doc/wg12/].
13. T. Isakowitz, "Hypermedia, information systems and organizations: A research agenda," in Proceedings of the Twenty-Sixth Annual Hawaii International Conference on System Sciences, IEEE Computer Society Press, January 1993, pp. 361–369.
14. T. Isakowitz, A. Kamis, and M. Koufaris, "Extending the capabilities of RMM: Russian dolls and hypertext," in Proceedings of HICSS-30, 1997.
15. T. Isakowitz, A. Kamis, and M. Koufaris, "The extended rmm methodology for web publishing," Working Paper IS-98-18, Center for Research on Information Systems, 1998.
16. T. Isakowitz, E. Stohr, and P. Balasubramaniam, "RMM, A methodology for structured hypermedia design," Communications of the ACM, pp. 34–48, August 1995.
17. M. Lang, "Hypermedia systems development: Do we really need new methods?," in Proceedings of IS2002 Informing Science + IT Education Conference, 2002, Cork, Ireland, June 19–21 pp. 0883–0891.
18. D. Lowe and W. Hall, Hypermedia & the Web, an Engineering Approach, John Wiley & Sons, 1999.
19. S. Murugesan, "Editorial," ACM SIGWEB Newsletter, Vol. 3, No. 3, October 1999.
20. J. Nanard and M. Nanard, "Hypertext design environments and the hypertext design process," Communications of the ACM, Vol. 38, No. 8, pp. 49–56, 1995.
21. R.S. Pressman, Software Engineering: A Practitioner's Approach, McGraw Hill, 5th Edition, 2000.
22. T. Quatrani, Visual Modeling with Rational Rose and UML, Addison-Wesley, 1998.
23. S. Retalis, Y. Psaromiligkos, and P. Avgeriou, "Web engineering: New discipline, new educational challenges," Information Services & Use, ISSN 0167-5265, IOS Press, Vol. 20, Nos. 2/3, pp. 95–108, 2000.
24. J. Rumbaugh, I. Jacobson, and G. Booch, The UML Reference Manual, Addison-Wesley, 1999.
25. D. Schwabe and R. de Almeida Pontes, "OOHDM-WEB: Rapid prototyping of hypermedia applications in the WWW," Tech. Report MCC 08/98, Dept. of Informatics, PUC-Rio, 1998.
26. D. Schwabe, R.A. Pontes, and I. Moura, "OOHDM-Web: An environment for implementation of hypermedia applications in the WWW," SigWEB Newsletter, Vol. 8, No. 2, June 1999.
27. D. Scwhabe and G. Rossi, "The object-oriented hypermedia design model (OOHDM)," Communications of the ACM, Vol. 35, No. 8, August 1995.
28. D. Schwabe and G. Rossi, "An object oriented approach to Web-based application design," Theory and Practice of Object Systems, Wiley and Sons: New York, ISSN 1074–3224, 1998, Vol. 4, No. 4.

29. D. Schwabe and G. Rossi, "Developing hypermedia applications using OOHDM," in Workshop on Hypermedia Development Processes, Methods and Models, Hypertext'98, Pittsburgh, USA.
30. I. Sommerville, N. Haddley, J.A. Mariani, and R. Thomson, The Designer's Notepad—A Hypertext System Tailored for Design, Hypertext: State of the Art, Intellext Ltd., 1990, pp. 260–266.
31. D. Thomas, "Web time software development," Software Development Magazine, October 1998, pp. 78–80.
32. D. Wall and A. Griffith, Graphics Programming With JFC, John Wiley & Sons, 1999.

Paris Avgeriou is an ERCIM (www.ercim.org) Research Fellow at the Software Engineering Competence Center, University of Luxembourg and the Fraunhofer Institute, in Darmstadt, Germany. He received a diploma (M.Sc.) in Electrical and Computer Engineering (1999), as well as a Ph.D. in Software Engineering (2003) from the National Technical University of Athens (NTUA), Greece. He has worked as a Visiting Lecturer at the Department of Computer Science (Jan. 03–Jan. 04), University of Cyprus, and as a research and teaching assistant at NTUA (Sept. 99–Dec. 02). He has participated in a number of European Union R. & D. projects and has published several papers in international journals and conferences. He acts as a reviewer in journals, standards and conferences and is a founding member of the World-Wide Institute of Software Architects and the president of its Greek chapter. His research interests concern the area of model-driven engineering and particularly software and business modeling, with emphasis on software architecture. He also conducts research in patterns, quality assurance, web engineering and mobile clients. His research is mainly applied in the domains of e-learning, CSCL and CSCW.

Symeon Retalis is Assistant Professor at the Department of Technology Education & Digital Systems, University of Piraeus. He holds an M.Sc. degree in Information Technology—Knowledge Based Systems from the Department of Artificial Intelligence, University of Edinburgh, Scotland, and a Ph.D. diploma from the Department of Electrical and Computer Engineering, National Technical University of Athens, Greece. His research interests lie on the development of web-based learning systems, design of adaptive hypermedia systems, web engineering, and human computer interaction. He has participated in various European R&D projects such as ELEN, MENU, UNIVERSAL, etc. He serves in the editorial board of international journals such as Computers in Human Behavior, IEEE Journal of Educational Technology and Society, ACM Computing Reviews, Journal of Information Technology Education. He participates to the ACM Web Engineering special interest group, to the CEN/ISSS learning technologies workshop. His publication list contains more than 70 items.