Reconciling Top-Down and Bottom-Up Design Approaches in RMM

Tomás Isakowitz
Wharton School of Business
University of Pennsylvania
Philadelphia, PA
Tel: (215) 573-9646
Fax:(215) 898-3664
E-mail: tomas@wharton.upenn.edu

Arnold Kamis
Stern School of Business
New York University
44 W. 4th St.
New York, NY 10012
Tel: (212) 998-0845
Fax: (212) 995-4228
E-mail: akamis@stern.nyu.edu

Marios Koufaris
Stern School of Business
New York University
44 W. 4th St.
New York, NY 10012
Tel: (212) 998-0390
Fax: (212) 995-4228
E-mail: mkoufari@stern.nyu.edu

ABSTRACT

The Relationship Management Methodology (RMM) is a well-known hypermedia design methodology. In this paper we provide an extension to it that enhances the design process. We present an iterative process of application design that incorporates the design of the entire application as well as its building blocks called m-slices*.*

1. Introduction

The proliferation of Intranets and Extranets as well as the vast expansion of the World Wide Web (WWW) and electronic commerce [1] indicate the need for a structured hypermedia design methodology that will guide the design, development and maintenance of large multimedia and hypermedia information systems [2] and collaborative systems [3]. The Relationship Management Methodology (RMM) proposed initially in [4] and extended in [5] is a well-known hypermedia design methodology based on the E-R model. RMM has been used to design and develop several applications, including the Hyper-Proceedings of the WWW-6 conference. In this paper we provide an additional extension to RMM which enhances the design process by handling both the global structure of the site and the details of its screens.

There have been other attempts at a hypermedia design methodology. The HDM data model [6] and its successor HDM2 [7] describe the structure of a database application domain adequate to support hypermedia access, but provide little support for building user views. In other words, while they describe an application domain, they do not facilitate the design and development of applications. RMM builds on HDM and HDM2 to provide the first full methodology. For database domains, RMM has the advantage of using tools such as E-R diagrams, with which designers are already familiar. OOHDM [8] [9] is an object-oriented hypermedia design methodology that is geared toward multimedia applications. OOHDM incorporates some of the same functionality as RMM within an Object Oriented framework. The OOHDM concept of a navigational class schema, presented in detail in [8], is similar to the m-slices described in [5] and here. Although OOHDM has a programming-like language to describe navigational class schemas, it lacks a graphical notation.

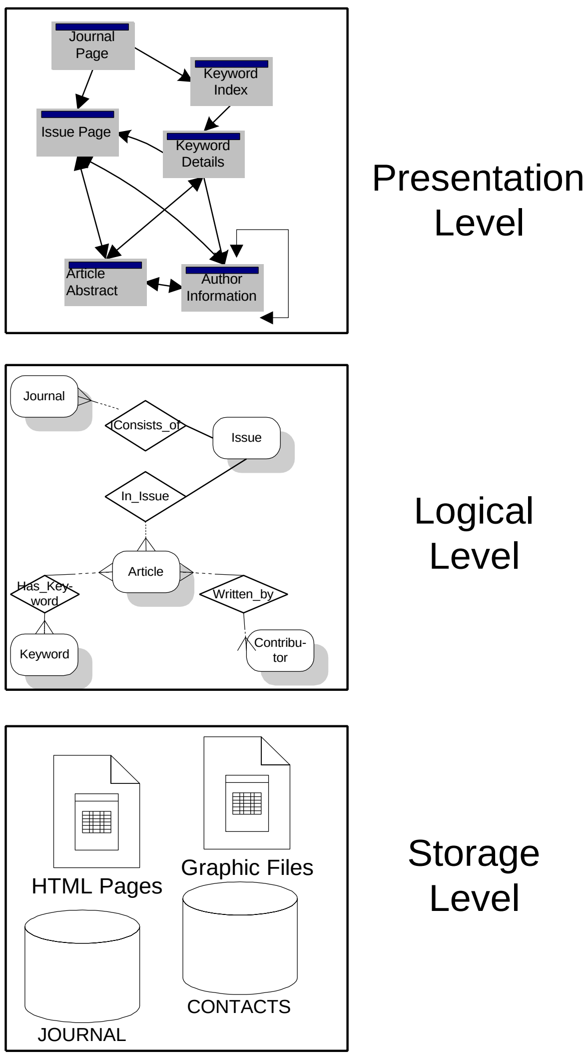

Figure 1: Three levels of information modeling, from least to most abstract. All three represent, in this case, elements of the JMIS Website.

The modeling process of hypermedia applications can be divided into three levels: *storage*, *logical* and *presentation* as shown in Figure 1. The storage level describes how information is stored, in terms of what software applications (e.g., databases, graphic editors) are needed, what files are used, the directory structure, etc. The logical level describes the structure of information that the hypermedia application manipulates. Elements of the logical level are, for example, *tables* in a database; *entities* and *relationships;* and *objects* and *methods.* The presentation level deals with information presented to users. At this level, one considers *what* information to group in screens (or more generally, presentation units) and *how* to navigate among screens.

In many cases, the distinctions between the levels are blurred, resulting in software development and maintenance difficulties. Between these levels are mappings, which determine how stored information is organized at the logical level and portrayed at the presentation level. In most cases, these mappings exist only at a cognitive level in developers minds. When these mappings can be formalized, as in the case of RMM, they can also be supported with software tools. This is one of the goals of RMM: to provide the foundations for a useful design process that can be supported with software tools.

In this paper we extend RMM's modeling capabilities at the presentation level. We present an iterative process of application design which incorporates the design of the entire application as well as its building blocks called *m-slices*. First, we give a brief description of the Journal of Management Information Systems website, which will serve as the example throughout the paper. We then provide a summary description of m-slices as described in [5]. We continue with the description of the iterative process of design and refinement, including the introduction of a new concept called the *application diagram*. We conclude the paper with a summary of our findings.

2. The JMIS Website

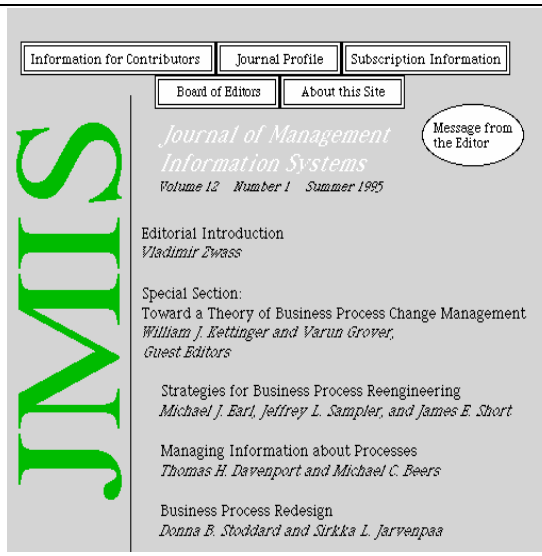

Figure 2: The issue table of contents page.

To illustrate the concepts in this paper, we use as an example the website for the Journal of Management Information Systems (http://www.stern.nyu.edu/jmis). As shown in Figure 2, each issue has its own page with a table of contents of all the articles and authors featured in that issue as well as a number of buttons providing information about the journal in general. We also constructed a top-level page, called the journal toppage, which lists all the issues of the journal available on the website.

There is a separate page for each article in which the abstract, its keywords, and the authors' names are provided. Each author's name is linked to a page containing information about her or him. There is also a keyword index that lists the available articles classified by each keyword. Throughout the site, references such as article titles, author names, and issue volume-number-season, are linked to the appropriate pages.

3. The M-Slice Solution

RMM prescribes a series of steps which are described in [4] [5]. In this paper we concentrate on the "presentation layer" aspects not covered in previous publications. M-slices are constructs used to

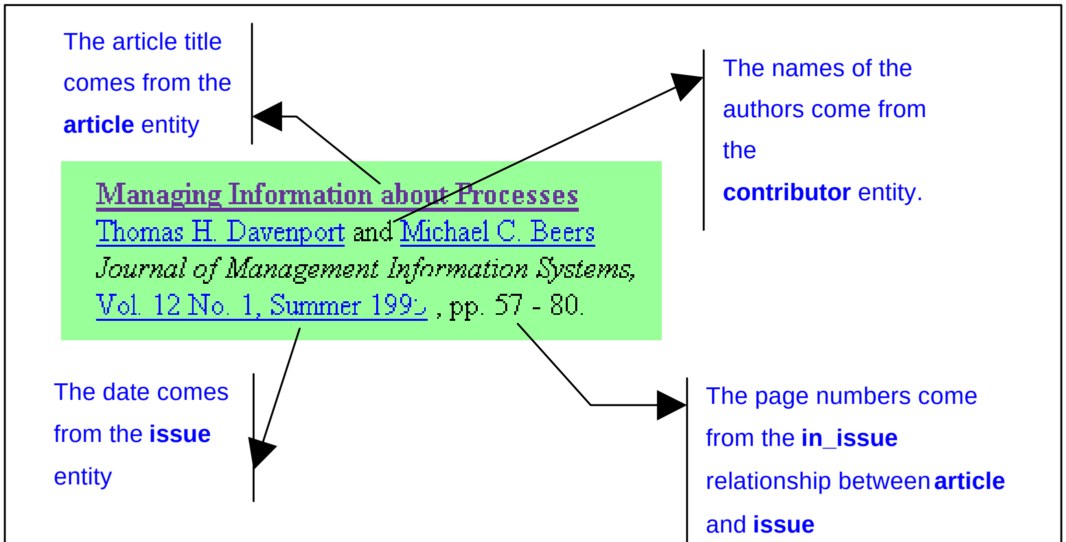

Figure 3: Rendering of the **artice.bib_citation** m-slice

group information into meaningful information units and they can be aggregated and nested to form higher level m-slices (the "m" in "m-slice" derives from the nested nature of Russian Matrjeska dolls.) Ultimately, higher level m-slices correspond to presentation units, e.g., HTML pages on the WWW. Besides fostering design reuse, this approach fosters a bottom-up design process.

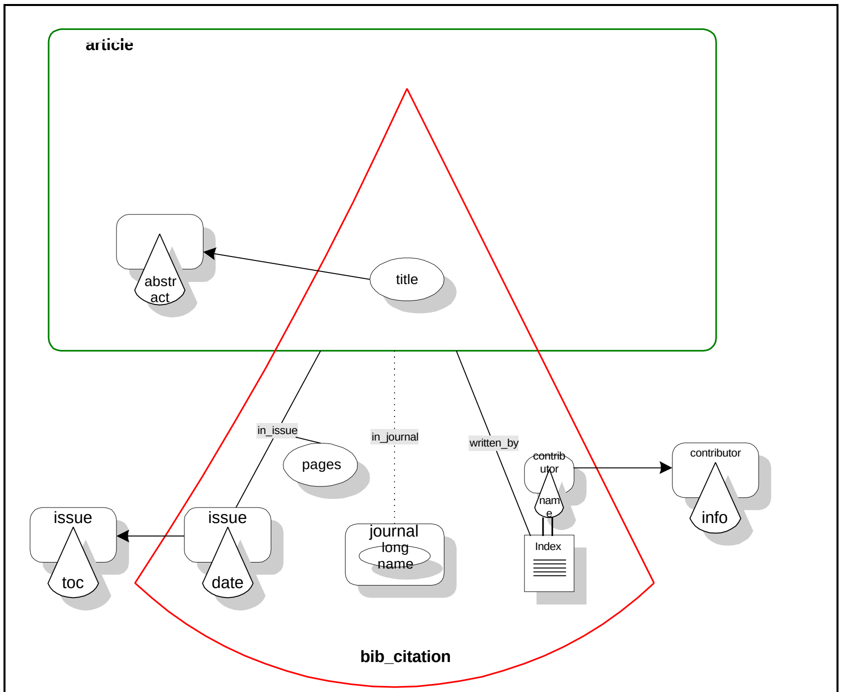

Figure 4: The graphical notation of the **article.bib_citation** m-slice.

Consider, for example, the **article.bib_citation** m-slice, which contains bibliographic information about an article and links to closely related information. The rendering of this m-slice is depicted in Figure

3 while its graphical notation is depicted in Figure 4 (See [5] for an explanation of the symbols). Every m-slice is owned by a specific entity in order to be considered an element of that entity[1]. In this way, an m-slice can be reused as many times as needed, by itself or as part of another m-slice, without the need to redefine it. In the case of **article.bib_citation**, the owner is the **article** entity. To stress the role of owner entities, m-slices are denoted as <owner entity>.<slice name>. Note that in addition to containing elements from its owner entity (the article's title, in this case), the **article.bib_citation** m-slice also contains elements from other sources. For example, the names of the authors come from the contributor entity.

Thus m-slices encapsulate information from various sources: attributes of the owner entity, attributes of related entities, and access structures such as indices. They can also be nested. For example, the **issue.date** m-slice, shown in Figure 5 is included in the **article.bib_citation** m-slice.

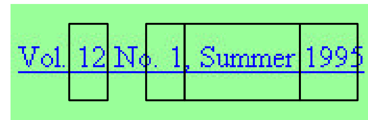

Figure 5: The issue.date m-slice

A thorough description of the graphical and program specification languages used to represent m-slices can be found in [5]. It is important to stress that m-slices describe only what information is to be part of a construct and where to obtain it. M-slices do not dictate how this information is to be shown. That is left to the user-interface design stage of RMM. M-slices provide the power needed for RMM to represent arbitrarily complex information units while supporting a structured, re-usable, manageable and programmable approach to hypermedia design and development.

4. The Application Diagram

We now introduce a new component of RMM, the *application diagram*, which provides a global view of a hypermedia application. The application diagram can be used in negotiating the application design with the users and other interested parties. It can also be used in an iterative fashion to find bugs and omissions and, generally, to refine the application.

After consulting with all the interested parties, the designer can construct an initial version of the RMM application diagram. Figure 6 shows the initial application diagram for the JMIS website. There are six presentation units which will correspond to web pages: *journal page*, *issue page, keywords page, keyword detail page*, *article abstract page* and *author page.* Consider, for example, the *article abstract page* appearing at the bottom left of Figure 6. There is one *article abstract page* per article, each containing information about that article (title, abstract), as well as links to its authors, to the volume the article appeared in, and to the *keyword detail page* corresponding to the keywords that classify the

---

[1] M-slices can also have no owner entity.

article. The star that appears in the *journal page* means that all other presentation units point to its presentation unit.

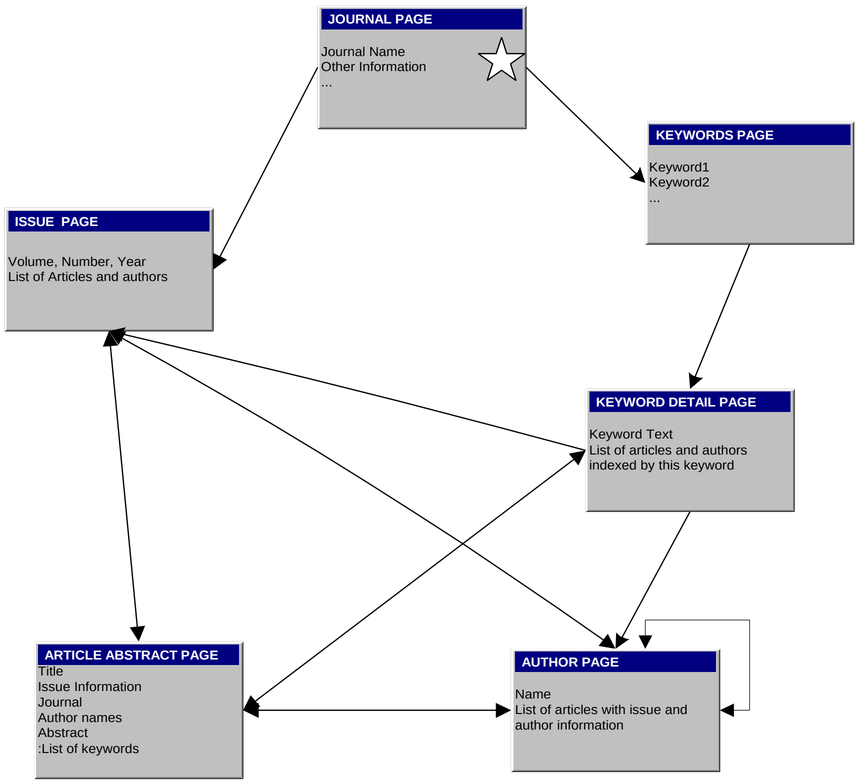

Figure 6: The initial application diagram for the JMIS Website. All Web pages and links among them are shown.

Next, each presentation unit is designed in a top-down fashion by decomposing it into its components which will be lower-level m-slices. After designing some m-slices, an application diagram can be automatically generated in a bottom-up fashion as described in the next section. By comparing such a diagram with the initial application diagram, the designer can keep track of what parts of the application need changes, updates, or additions. This iterative process is depicted in Figure 7.

5. Building the Application Diagram Bottom-Up

An RMM application diagram can be automatically generated from the collection of M-slices using the following two-step algorithm:

1) Examine all the m-slices used in the application and note all the hyperlinks that are anchored in those m-slices. The targets of those hyperlinks are m-slices that represent presentation units in the application, e.g., pages of a website. Place all of these m-slices in the application diagram.

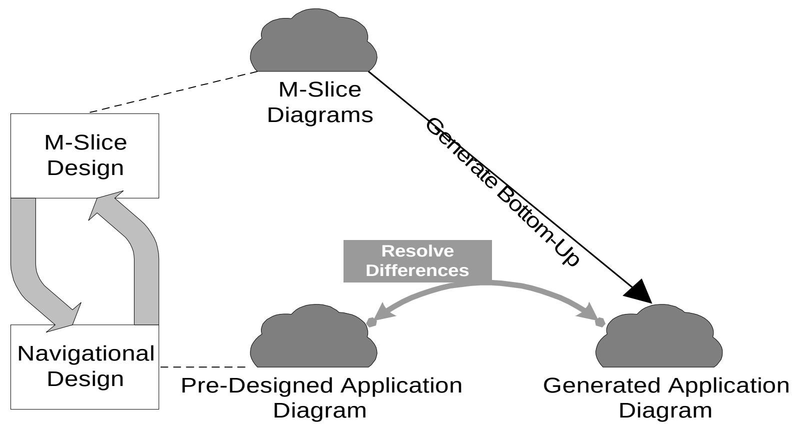

Figure 7: At each iteration of slice design, an application diagram is generated in a bottom-up fashion and compared to the application diagram designed *ex ante*. Unimplemented M-slices are highlighted and differences are resolved, leading to further iterations.

2) Identify all the interactions between the presentation units. (Only the m-slices that have been placed in the application diagram are used in this step.) Examine each one for links. Place those links in the application diagram as arrows from the anchor m-slice to the target m-slice. Name them after the relationship from the E-R diagram that they are based on. Examine the m-slices for any lower-level m-slices that are nested inside. Place in the application diagram any links found in those nested m-slices. Name them after the relationship they are based on, concatenated with the relationships to the nested m-slices.

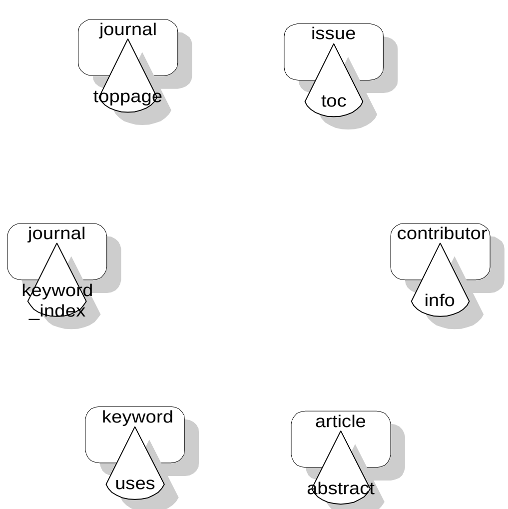

Figure 8: The first step is determining the presentation units.

Consider, for example, the generation of the application diagram for the entire JMIS website. The first step is to select the presentation units, i.e., the pages, of the site. Looking at the m-slice **article.bib_citation** (Figure 4), we can immediately see that there are three links to m-slices outside this m-slice. The three hyperlinks in this example have as their targets the m-slices **article.abstract, issue.toc,** and

contributor.info. These three m-slices are placed in the application diagram as presentation units. After examining all the m-slices of the site, we obtain the diagram shown in Figure 8.

The next step is to place in the application diagram the hyperlinks that exist between these pages. To do this, we consider only the m-slices that are now in the application diagram. For each m-slice we find all the hyperlinks to other m-slices and we place them in the application diagram. Looking at the **article.bib_citation** m-slice we find three hyperlinks to **article.abstract, issue.toc**, and **contributor.info** (a recursive link). We name these hyperlinks after the concatenation of the relationship of the hyperlink with the relationship to each nested m-slice. For example, we name the hyperlink to **issue.toc** as **is_author_of + in_issue**. After going through all the pages and finding all the hyperlinks, we have

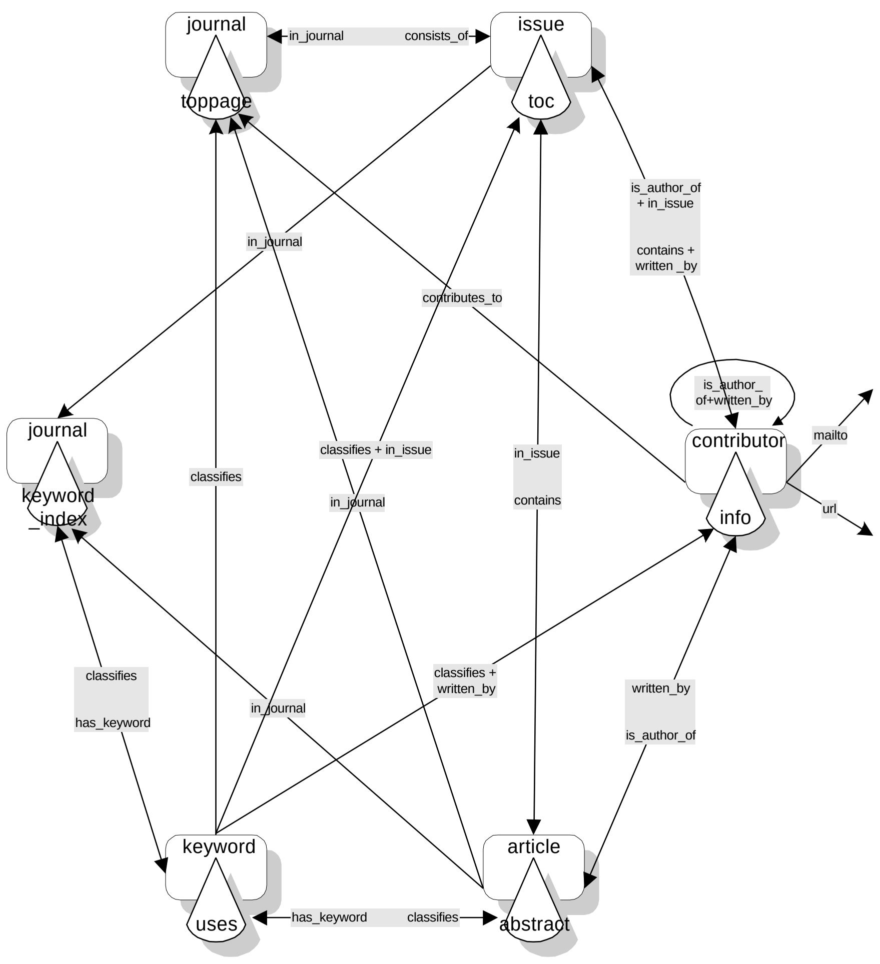

Figure 9: The complete application diagram for the JMIS website.

obtained a complete application diagram, which can be seen in Figure 9. Note that for simplicity, we use two-way arrows to denote two hyperlinks between the same m-slices. In this case, the name of each link is placed closer to the target of the arrow.

The similarity between the application diagrams shown in Figure 6 and Figure 9 ought to be clear. It is important to stress that while the first is created *ex ante* and thus represents a top-down approach to design, the latter is produced *ex post* from the design elements in a bottom-up fashion. Discrepancies between the two would indicate m-slices that have not yet been implemented (those that appear in the top-down approach but not in the bottom-up), or m-slices that ought to be either removed or hidden within higher level m-slices (those that appear in the bottom-up approach but not in the top-down). In either case, developers can use these two diagrams to re-shape the application by going back and forth.

6. Summary

In this paper we have extended the Relationship Management Methodology for hypermedia application design. We have introduced the RMM application diagram, which can be used in an iterative fashion to refine an application design. The designer can start with an initial conception of the hypermedia application and use it to construct the necessary m-slices. Using those m-slices, one can automatically generate the actual application diagram. Comparing the two diagrams allows the designer to modify, correct and refine the application he or she is designing.

These extensions have been developed to be consistent with the RMM process and notation so that they can be seamlessly integrated with existing hypermedia design software tools, such as RM-CASE [10]. The m-slice enhancement to RMM combined with the application diagram can provide the necessary power and flexibility to RM-CASE necessary for the design of complex hypermedia applications. At the same time, application diagrams ensure that applications designed with RMM are well structured, maintainable, and extensible.

References

[1] R. Kalakota, A. Whinston, *Frontiers of Electronic Commerce*, Addison-Wesley, USA, 1996.

[2] K. Takahashi, E. Liang: "Analysis and Design of Web-based information systems". Electronic Proceedings of The Sixth International WWW Conference, Santa Clara, USA, 1997.

[3] O. Günther, R. Müller, P. Schmidt, H. Bhargava, R. Krishnan, “MMM: A Web-Based System for Sharing Statistical Computing Modules”. IEEE Internet Computing, May – June 1997, pp. 59-68.

[4] Isakowitz, T., Stohr, E., & Balasubramanian, P. “RMM: A Methodology for the Design of Structured Hypermedia Applications”. *Communications of the ACM*, 38(8), 1995, pp. 34-44.

[5] Isakowitz, T., Kamis, A., Koufaris, M. "Extending the Capabilities of RMM: Russian Dolls and Hypertext". Proceedings of the Thirtieth Annual Hawaii International Conference on System Sciences, 1996.

[6] Garzotto, F., Paolini, P., & Schwabe, D. (1993) "HDM - A Model-based Approach to Hypermedia Application Design" *ACM Transactions on Information Systems*, 11(1), 1-26.

[7] Garzotto, F., Mainetti, L., & Paolini, P. (1996) "Navigation in Hypermedia Applications: Modeling and Semantics." *Journal of organizational Computing and Electronic Commerce*, Vol. 6, No. 3, 1996, pp. 211-238.

[8] D. Schwabe, G. Rossi and S. Barbosa: "Systematic Hypermedia Design with OOHDM". Proceedings of the ACM International Conference on Hypertext (Hypertext'96), Wash., March, 1996.

[9] G. Rossi, D. Schwabe and A. Garrido: "Design Reuse in Hypermedia Applications Development" Proceedings of ACM International Conference on Hypertext (Hypertext'97), Southampton, April 7-11, 1997, ACM Press.

[10] Diaz, A., Isakowitz, T., Maiorana V. and Gilabert G. RMCase: "A Tool To Design WWW Applications". *World Wide Web Journal*, Vol. 1, No. 1, 1995, pp. 559-566.