A Case Study in Hypermedia Design: CD-ROM Encyclopedias

Ravi Arunkundram
Leonard N. Stern School of Business
New York University

Tomas Isakowitz
The Wharton School
University of Pennsylvania

Edward A. Stohr
Leonard N. Stern School of Business
New York University

March 1998

Working Paper Series
Stern #IS-98-5

Center for Digital Economy Research
Stern School of Business
Working Paper IS-98-05

A Case Study in Hypermedia Design: CD-ROM Encyclopedias

by

Ravi Arunkundram
Tommy Isakowitz
and
Edward A. Stohr

Information Systems Department
Leonard N. Stern School of Business
New York University

Abstract

The function of a design model in a hypermedia project is to provide a formal method for specifying the multimedia data objects that are to be stored and the screens and navigational paths that are to be provided to users. This formal expression provides a way to communicate design decisions and to automatically generate and maintain hypermedia applications. The RMM approach to hypermedia and WWW design [Isakowitz et al 95] has been successfully applied in a number of real world applications over the last few years. In this paper, we extend the RMM model in two ways. First, we develop an approach to handle the "unstructured" components of the hypermedia applications such as those on the World Wide Web (WWW). Second, we show how dynamic (program) elements can be represented in the model thus facilitating its application to a wider range of multimedia and especially, to Java applications. These extensions were developed as a result of an attempt to simulate the design of a commercial multimedia encyclopedia. The resulting design is used to illustrate the new HM design concepts.

1. Introduction

As anyone who has tried to develop a hypermedia (HM) application, whether on CD-ROM or the World Wide Web (WWW), can attest, the development process is error-prone, time-consuming and fraught with unexpected difficulties. Moreover, after the initial HM application has been developed, there are difficult technical and managerial problems involved in keeping it up-to-date. These experiences point to the need for better hypermedia design, development and maintenance technologies. In particular, we need to develop formal models to represent the unique design elements in hypermedia, application generators to translate the design to a running application, and software environments to help users maintain and revise the HM application over time.

Until quite recently, there was relatively little research on design models for hypermedia. Perhaps the earliest example of a HM design model is HDM [Garzotto et al 93], which was followed by its successor, HDM2 [Garzotto et al 95]. HDM proposes a structured, database-oriented approach for representing and supporting hypermedia design and development. Building on HDM, the RMM (Relationship Management Model) also takes a database-oriented approach to the development of structured hypermedia applications [Isakowitz et al 95]. More recently, Lange [Lange 96] and Schwabe and Rossi [Schwabe & Rossi 95] have proposed object-oriented approaches to HM development.

While there are advantages and disadvantages to each of the approaches that have been proposed, there can be little doubt about the usefulness of a design model and a more formal approach to developing HM applications. These benefits include:

- Improved communications amongst the designers and maintainers of the application.
- Machine representation of designs and subsequent automatic generation of HM applications.
- Reduced design and implementation errors.
- Consistency and uniformity of style in the developed HM application
- Easier reuse of multimedia structures, components and processes.

The work in this paper is based on the RMM model developed by [Isakowitz et al 95] and a subsequent extension [Isakowitz et al 97, 98]. RMM has been used by a number of developers including commercial organizations such as M. E. Sharpe, Merrill Lynch and Bellcore and academic institutions such as New York University, Pace University and Staffordshire University. An example of a web site that has been completely designed and maintained by the RMM methodology is the Journal of Management Information Systems (JMIS) site [Isakowitz et al 97]. A CASE tool, RM-Case, which supports the RMM methodology by providing graphics tools and an application generator for WWW development, is under construction [Diaz et al 95].

Despite the initial successes of RMM, there remain several gaps in the methodology and a number of questions to be answered. Among the gaps is RMM's inability to represent the non-database (unstructured) portion of an HM application and also, the dynamic, interactive aspects of HM applications. Among the questions are RMM's ability to represent the broad range of structures and navigation mechanisms that are found in commercial hypermedia applications. While WWW sites and other HM applications can be successfully designed using RMM, there is a distinct possibility that designers are limited by the structure of the design model and by the limited number of design primitives it provides. They might therefore be constrained to design more pedestrian and less powerful applications than they would without the use of the methodology. This is the problem addressed by the stream of research underlying this paper. Specifically, the objectives of this paper are to:

- Validate the RMM data and navigation models by seeing if they can adequately represent a commercial multimedia system (encyclopedia.)
- Develop any extensions that might be needed.
- Refine and formalize the RMM representation scheme and design methodology.

The paper is organized as follows. Section 2 provides some background and a brief description of the research methodology employed in the paper. Section 3 provides an introduction to Microsoft's Encarta'94 CD-ROM multimedia encyclopedia [Microsoft 94] that is the main subject of our investigations in the remainder of the paper. Section 4 contains a comprehensive overview of the RMM model, develops several extensions, and provides an extensive example of RMM design using Encarta'94 as an example. Section 5 provides a brief RMM description of the 1997 version of Encarta [Microsoft 97], which we use to discuss the evolution of the design of the encyclopedia. We conclude in Section 6 with some suggestions for future research.

2. Background and Research Approach

The overall RMM systems life cycle is depicted in Figure 1, which is adapted from [Isakowitz et al 95]. Note that the development process will not follow a strictly linear order - there will normally be many returns to previous stages and some steps (in particular, steps 5, 6 and 7 in the Figure) can be performed simultaneously.

(Figure 1 - Design Steps and Outputs)

The RMM design consists of seven steps concerned, respectively, with database structure and content, groupings of related information into "contexts", navigational paths, screen content, interface design, conversion of the RMM constructs to their equivalents in the software system chosen for the implementation, and navigational dynamics. The original version of RMM primarily addressed the first three design steps. A subsequent development has addressed the fourth task of rigorously specifying the content of each screen or display window [Isakowitz et al 97]. The fifth step, interface design, is primarily artistic in nature and is therefore outside the RMM methodology per se. To aid in the sixth step, the RMCase tool mentioned above is developing the conversion protocols needed for automatic development of WWW applications. The seventh step, that of dynamic design, is currently left up to the developers. We intend to extend RMM to handle the dynamics of HM applications in the future. In the meantime, the present paper takes a step in this direction by modifying steps 1 through 4 of the methodology to include programmable software objects.

The underlying philosophy of RMM is to provide a rigorous machine readable and executable definition of a hypermedia application in terms of four components: information content, contexts, navigation paths and presentation units. Having an automatable representation should make it easier to develop an automated design aid and application

generator, to maintain the system and to facilitate reusability of components. This automated representation will also make it easier to perform steps 6 and 7 - the conversion to a particular hypermedia engine and the specification of the dynamic, interactive elements of the design. We now provide a brief overview of the status of our efforts with regard to each of the four HM design components mentioned above.

Information Content: RMM was initially designed for what we call "structured" applications - that is, applications where there are (1) a number of repetitive elements (entities) that need to be represented in essentially the same way and (2) well-defined relationships between the different classes of entity. Examples of repetitive classes are personnel records in a Personnel HM application, products in a HM catalog or faculty in an academic department's home page. Borrowing from database theory, RMM begins with a definition of the data content of the HM application expressed in a traditional entity-relationship (E-R) format. Ideally, all the structured content of the application is stored and maintained in a database and retrieved automatically on demand. In this paper, we extend the original RMM data model in two ways. First, we accommodate what we have hitherto called the "unstructured" component of a hypermedia application - that is, non-repetitive information elements that are linked together such as a collection (or "web") of individual HTML pages in a WWW site. Second, we introduce an object-oriented notation to allow the specification of interactive components.

Contexts: A context is a meaningful (to the end-user) grouping of information. The idea is to give users a sense of locality to help avoid information overload and the sense of being "lost in hyperspace." Contexts also simplify the design process by allowing the designer to aggregate details that might otherwise be overwhelming. In this paper, we formalize the original RMM concept of a context (which was captured by "entity diagrams" in [Isakowitz et al 95]) and generalize it to include "webs" of related information elements as mentioned above.

Navigation Paths: The most significant contribution of RMM (and similar HM design models) is to help specify the navigation paths that users may take through the content of the application. Database design does not have this need because navigation is essentially handled by the system. While a variety of methods for navigating through the information in an HM application have been used in practice, RMM predefines a limited but useful subset of basic types. These basic types can then be defined in more detail and specialized for the chosen hypermedia software implementation during the conversion step (step 6.) In this paper, as a result of our exploration of some existing HM applications, we develop a number of new (to RMM)

navigation mechanisms and, especially, the Program Control construct, which represents a complex computer program.

Presentation Units: Another significant contribution of RMM is that it supports a rigorous definition of the units of information that the designer wishes to display in screens, pop-up windows, etc. These "m-slices" are essentially recursively defined database views. They can therefore be described in terms of a query language and retrieved and presented on demand [Isakowitz et al 97].

In this research, we investigate the "expressiveness" of the RMM design model by which we mean its ability to rigorously specify an HM application. The practical experiences cited above support the idea that the original RMM mechanisms were expressive enough to describe a range of different HM applications. However, these applications were designed with the philosophy and using the mechanisms of the original RMM model. We want to find out the extent to which the RMM constructs constrain designers. To do this, we have embarked on a research program in which we use RMM to describe existing commercial HM systems designed by a variety of designers for a variety of applications. The objective of this research program is to extend and formalize the RMM representation model and methodology. As we encounter problems, we modify and improve the RMM representation scheme. Through the process of formally representing a large number of existing HM applications, we also expect to understand the nature of successful and unsuccessful HM designs and even to develop an approach to HM "literary criticism" [Garzotto et al 95].

In this paper, we use the RMM model to describe a commercial CD-ROM encyclopedia - the 1994 version of Microsoft's Encarta [Microsoft 94]. It should be pointed-out that we are not interested in how Encarta was actually designed or implemented. Rather, we are interested in investigating whether or not it is possible to describe a rich application such as Encarta using RMM. In the process of performing this "reverse design'" we discovered a number of gaps in what could be specified in RMM and were able to develop several significant extensions.

3. A Brief Overview of Encarta'94

Millions of copies of popular CD-ROM encyclopedia titles such as Grolier, Compton's New Media and Microsoft's Encarta, are sold annually. While each encyclopedia has its own distinct style and feel, most share a set of common features - some of which are direct translations from the book form, while others are unique to the medium. Common features

include text articles, cross-references, tables of contents, word and subject indexes, video, sounds, animations, video clips, a timeline and an interactive atlas.

For background, we provide a very brief introduction to some of the display screens in Encarta'94. When Encarta'94 is executed, it first displays the opening or "Title Screen" for the encyclopedia as shown in Figure 2. At the same time, a background audio of historic "sound bytes" and other recognizable sounds is played. The Title Screen serves as an entry point to the rest of the application. Clicking on "Enter Encarta" on the lower right of the screen brings one to the last topic accessed (in a prior session.) As an example, the Topic Screen for Australia is shown in Figure 3. This is the main information display for the encyclopedia. The scrollable text on the right of the screen is the electronic version of a paper-based encyclopedia article. Note that a large number of menu items and buttons are available that allow access to other parts of the encyclopedia from the Topic Screen. For example, clicking on the Timeline button in Figure 3 brings one to the Timeline screen shown in Figure 4, which displays historic events on a timeline stretching from pre-history to the present day. Essentially, one can scroll through history clicking on the various icons to learn more about important events and topics and to view historic maps. Returning to the Title Screen, it can be seen that the user may access the encyclopedia in a number of different ways. For example, clicking on the "Category Browser" icon in Figure 2 takes one to the screen shown in Figure 5, which provides a mechanism for browsing through the subject matter of the encyclopedia in a hierarchical fashion.

(Figures 2 through 5 Encarta'94)

We will refer to these figures in the remainder of the paper in which we simulate their design using the RMM methodology.

4. The RMM Model and Extensions: Illustration

In this section, we describe each of the first four design steps in Figure 1 and illustrate the associated RMM concepts using the Encarta'94 multimedia encyclopedia [Microsoft 94]. Where the concepts we wish to illustrate are not present in Encarta, we use examples from an academic department Internet site. The section concludes with a very brief overview of the remaining three design steps (steps 5, 6 and 7 in Figure 1.)

Step 1: Entity Relationship Diagram (Object Class Diagram)

The objective in this step is to specify the information that is to be maintained by the application and presented to the user using one or more E-R diagrams [Chen 76] or object class diagrams [Blaha et al 88].

The advantage of a database-oriented approach is that it encourages a formal representation of the information which, in turn, improves communication about the design and facilitates implementation and maintenance of the application. The standard E-R approach is familiar to most software developers and translates easily into a relational database design. The idea is to store the data for the HM application in a database. A number of different strategies can then be used to present the information to the user. For example, each item of information that appears in the final application can be retrieved dynamically from the database and presented to the user on demand. Alternatively, the web-site or HM application can be updated periodically from the database using a batch process as with the JMIS site mentioned in section 1 [Isakowitz et al 97, 98].

In this paper, we use an adaptation of object-class diagrams rather than E-R diagrams to represent the information. This allows specification, at a high level, of the dynamic components of the design. These are represented in the diagram as traditional objects that have encapsulated data and methods or processes (see [Gorman & Choobineh 91] for a similar treatment of object class and E-R diagrams.) Figure 6 shows the graphical conventions that are used in the construction of an RMM object class diagram, while Figure 7, shows the Object Class Diagram for Encarta'94. Figure 7 was constructed by the authors after extensive interaction with the application.

(Figure 6: E-R Diagram (Object Class Diagram) Conventions)
(Figure 7: Encarta'94 Object Class Diagram)

While all of the items of interest in a HM Application might be designed formally as objects in the object-oriented programming sense, we have chosen to separate the static elements from the dynamic and to show both in the one diagram together with their relationships or associations and (optionally) their attributes. As shown in Figure 6, an entity class (static element) is represented by a rectangle. Following [Blaha et al 88] an object class (dynamic element) is specified by a rectangle with three divisions representing, respectively, the name of the object class, its associated methods, and its associated data attributes. The three relationship types (associative links, "is-a" relations and "aggregations") are depicted by arcs or arrows following common conventions in object class hierarchy design [Blaha et al 88]. Note that we have attached minimum and maximum cardinalities to each end of the association links for additional precision and to help with the database design step. The Web construct shown in Figure 6 will be explained later.

Entities are static, data-oriented items of interest in the HM application and are familiar from database theory and from previous descriptions of RMM. Entities are described by their

attributes and participate in relationships with each other and with the objects in the system. Depending on the application, they may be stored in a relational database and retrieved on demand.

Objects allow the specification of the dynamic aspects of the HM application. An Object in an RMM model is a collection of information and associated procedures for retrieval and display of that information. For instance, when the Timeline object is accessed in Encarta'94 (see Figure 4), elements of data (Timeline Events, Topics and Maps) are automatically accessed and displayed. Methods and procedures associated with the Timeline Object then control the user interaction. As another example, procedures associated with a video object may range from simple data access protocols (MPEG, etc.,) to modifying other elements in the users' context (stopping the display of visual and audio objects or exiting all other contexts to remain in the current one exclusively.)

The objects and associated methods and procedures are described only at a high level in the initial stages of an RMM specification - detailed program specifications can be developed using any object-oriented software engineering technologies available to the developers (during step 7 of the process.)

The information content design for Encarta'94 in Figure 7 shows three independent object class diagrams. Encarta Highlights in the lower right of the Figure is a relatively simple object class consisting of an aggregation of specific highlights or demonstrations of the capabilities built into the encyclopedia. Its methods implement a simple guided tour (see below) of these highlights. The second object class diagram represents the Dictionary (lower left), which can be accessed directly from the main menu or indirectly by "striping" a word in any other part of the encyclopedia and pressing "enter."

The main object class diagram represents the encyclopedia itself. The objects Gallery, Category Browser, Timeline and Atlas, provide dynamic interactive access to the information. For example, the Category Browser (Figure 5) consists of an aggregation of Class/Category titles that are arranged hierarchically to represent an overview of the information in the encyclopedia. Each Class is associated with a Class-Graphic, which appears in the top left of the topic screen (see Figure 3) when a topic within that class is displayed. The Gallery Browser object provides interactive access to Gallery items, that may be Pictures, Animations, etc., as shown by the "is-a" relationships on the left of the diagram. Similarly, the Atlas consists of an aggregation of maps arranged in a hierarchy through which the user may "zoom" in and out. The Gallery items/maps associated with a topic are displayed in the bottom left corner of the Topic screen (see Figure 3). Finally, as mentioned earlier, the

Timeline object (Figure 4) consists of an aggregation of specialized Topic, Event and Map entities that are accessed under program control.

The Topic entity class at the center of Figure 7, is the heart of the encyclopedia from a content point-of-view. Each topic entity consists primarily of text as in a conventional encyclopedia. As shown in the figure, Location is a specialized topic that, among other things, is associated with a Map entity. Finally, Country Topic and State Topic are further specializations of Location, each with their own unique properties. For example, each State Topic is associated with a "State Fact Box" listing the state's flower, crest, motto, and so on.

To illustrate the concept of a web, we turn to the second example application - that of an academic department's homepage as shown in the E-R diagram in Figure 8(a). In the diagram, there are a number of entity classes as described above - those for faculty, Course, Working Papers and Events - and two "webs" - one for "Research" and one for "Student Clubs." The components of a web are, in general, "one-off" individual "pages" rather than instances of entity classes; they are usually linked together in a primarily hierarchical manner using hypertext links. A Web is a particular form of "Context" as defined below.

Figure 8 (a) and (b) - Academic Department Application

Step 2: Design of Information/Presentation Contexts

In this step, the information content of the application is designed in more detail by specifying the content of its "contexts." A context is a designated aggregation of multimedia entities that represents a particular subject area instance. While a context could be arbitrarily complex, we simplify the concept in RMM by recognizing only two types of context. The first type is the "Web" context introduced above. The second form, first introduced in [Isakowitz et al 95], allows the designer to designate a "focal" entity/object class and to specify the associated information elements (including other entity/object classes) that should always occur with it - in the same "conceptual space" from the user's point-of-view.

There are two reasons for considering contexts in HM design. The first reason is to simplify the E-R design in step 1 by omitting unnecessary (and possibly ad hoc) details. The second reason for specifying contexts is to give users a clear sense of context or location - an important design goal in HM applications [Thuring 95]. This can be done, for example, by using a common set of visual clues to display all of the information elements in a context.

Figure 8 illustrates the usefulness of aggregation in the design process. It is only necessary in the initial E-R diagram to show the association relationships that each web has with the structured Entity/Object Classes in the application. The design of the detailed

structure of the web is deferred until later. As an example of such a design, the structure of the pages for the Research Center web is shown in Figure 8(b) using E-R diagram conventions. A "head" menu or "homepage" at the top of the web hierarchy is usually designated as the access point into the web. (However, there may be more than one access point.)

An Entity (Object) Context is limited to a single focal entity (object) class and its associated entities and/or slices. The context is depicted by an "entity diagram" using E-R diagramming conventions to designate the contextual information associated with each instance of the entity class. Figure 9 shows the conventions used to draw an entity diagram, while Figure 10 shows the Entity Diagram for the "Topic" entity class in Encarta'94 (the corresponding screen was shown in Figure 3.) An Entity Diagram anticipates the screen content design phase by using a rounded rectangle to depict the information that will be shown on the screen for each entity instance. Information elements outside this area are associated with other entity instances or slices and can be accessed without changing the context from the user's point-of-view. Just how this last objective is achieved is specified during the interface design in step 5. (For example, one might display the associated information elements in pop-up windows overlaying the information for the underlying focal entity instance.)

Figure 9: Entity Diagram Conventions
Figure 10: Encarta'94 Entity Diagram for a Topic

Figure 10 specifies that topics can cross-reference each other and that the screen for the Topic Entity will contain:

(1) The text for the topic.

(2) An associated Category Graphic that contains the title of the category in the Category tree to which the Topic instance belongs.

(3) Zero, one or more "Gallery Items" associated with the topic. Gallery Items for a given topic are accessible via a "Guided Tour" (see below.)

Figure 10 also specifies that the user should be able to access the contributor information and bibliography for the article. When these are accessed, the user should be aware that they are associated with the current Topic instance.

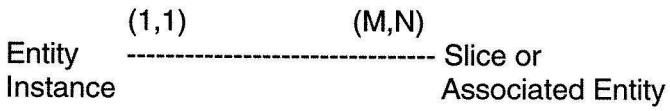
We now provide a formal explanation of entity diagrams by defining their constituents in more detail.

Associated Entities: If other entities depend for their existence on the focal entity instance (so-called "weak entities"), then they will normally become part of the context for that entity. For example, in a Personnel HM application, "Spouse" and "Pension-Plan" might be weak entities

associated with the Employee entity. In the database design, these would normally be stored in separate database tables. On the Employee screen in the HM application they might appear in pop-up windows when the appropriate hypermedia button is clicked - thus preserving the user's sense of location or context. Other entities (that are not weak in the preceding sense) can also appear in the context of an entity instance. In this case, the entities are associated to the context of the focal entity by the designer on the basis of the relevance of the information rather than by any structural properties from the E-R diagram. For example, in the Topic entity diagram in Figure 10, Contributors and Bibliography are associated entities containing, respectively, the name(s) of the author(s) of the topic and one or more lists of references. They are shown outside the rounded rectangle representing the screen in Figure 10 because they are not normally visible but appear in a pop-up window if clicked from buttons within the topic text. Category Graphic and Gallery Item are other entities associated with the Topic Entity in Figure 10. They are shown within the rounded screen rectangle because they are normally present on the left side of the Topic screen (see Figure 3). The Category Graphic represents the information category to which the entity instance belongs. Zero, one, or more Gallery Items (usually pictures) are specified by the HM author to illustrate the text of the topic.

Slices: An entity in the E-R diagram may be broken down for presentation to the user into a collection of subsets of attributes with each subset regarded as a "slice" or "view." In a Personnel HM Application, for example, financial attributes might constitute a slice of Employee that is accessible only to authorized users. Continuing this example, additional slices of the context may be instantiated in different media. For example, a photograph of the employee might represent a slice, which could be displayed in a part of the screen or accessed by clicking a button from the main Employee screen. In Encarta'94, Topic Text is a slice of the entity Topic.

To summarize, an associated entity or a slice is a distinct grouping of multimedia information that is related to its focal entity instance in the following fashion:



Here, $N \geq 1$ and $M \geq 0$. If M=0, the associated entity or slice is optional (may not occur in some context instances.) The links between an entity instance and its associated entities or slices are called "Structural Links" in RMM because they show the structure of a particular entity instance

and its associated context. They are shown by dashed lines to differentiate them from the association or "Application Links" in an E-R or Object Class Diagram.

The Entity Diagram does not formally specify the contents of the screen. Each entity context with its related entities and slices is formally defined in the fourth step (screen content design) using the concept of "m-slices." Nor does an entity diagram specify the dynamics of access to the information. For example, in Encarta'94, the Contributed-By and Bibliography entities will be generated on the screen if the user clicks on the associated links in the text, while the Topic outline can be obtained in a pop-up window by clicking a menu bar button. These design details are deferred to a later stages (steps 5 through 7.)

As the information content aspects of the application have been laid-out after this step in the design process, it is now feasible to design the database. Later steps in the design process, in particular, the m-slice design in step 4, are dependent on the database design. The database design for Encarta'94 is developed directly from the Object Class and Entity Diagrams and is shown in outline in Figure 11 below.

(Figure 11 - Encarta'94: Partial Database Design)

Step 3: Navigational Paths and Access Mechanisms

In this step, the navigational paths through the information are specified using an RM Diagram [Isakowitz et al 95]. As described above, an E-R diagram (or object-class diagram) specifies the associations between entities and the integrity relationships that must be maintained by the database. Some or all of these relationships may be the basis for navigational paths traversed by users in the HM application. However, most HM applications require many more access paths. In addition, these access paths can take different physical forms. For example, the user may be constrained to follow a more-or-less fixed path from entity to entity through the application as in a "guided tour" (see below.) Alternatively, in moving from one entity class to another, the user may first be presented with an "index" of possible paths (target entity instances from which to choose.) At a more sophisticated level, the user may move from one entity class to another under program control - as in the "Category Tree" in Encarta'94 or a Microsoft "wizard" which essentially presents a series of dialog boxes to the user. Many other such navigation mechanisms are possible. The point is that a navigation path not only specifies routes through the information but also how to get there. In this sense, the navigation paths have a major impact on the experience of the user with the HM system.

At any point in time, the system and user will be in a given "information state" consisting of a set of currently accessed and displayed information elements. The information state can be

quite complex consisting of diverse elements of information from many different entities and slices. In RMM, we therefore break down the specification of the possible information states and the transitions between them into two steps - navigation design, which we explain here, and M-Slice design, which is explained in the next step.

The purpose of the RM Diagram, which is the main output of the navigation design step, is to specify all the logical paths from one *individual information element* in the application to another. By information element, we mean an entity instance (or entity context instance), a web context, an object, or a Grouping (menu) as defined below. Note that that an entity may have many instances but there is only one instance of each type of RMM object, Web or Grouping. The RM Diagram specifies how the system can use the associative relationships between information elements to move from one "focal" information element to another. However, this logical path is not to be confused with the series of information states (screens) that are presented to the user. The RM Diagram logical path can contain a number of information elements that are not displayed to the user; or conversely, as we will see below, the information state displayed to the user can contain a number of information elements that are displayed simultaneously on one screen.

The conventions used in RM Diagrams are shown in Figure 12. The RM Diagrams for Encarta'94 are shown in Figures 13(a) and 13(b). Note that the Objects, Entities and Webs that were shown in the E-R diagrams appear again here. Note also that the navigation paths within the contexts (entity diagrams and webs) must also be designed (see below.)

Figure 12: Conventions Used in RM Diagrams
Figure 13 (a): Encarta'94 RM Diagram for Title Screen
and *(b): Encarta'94 Main RM Diagram*

The navigation paths between the various information elements in RMM are represented by directed arcs with a different notation for each class of navigation. Most of these constructs were introduced in an earlier paper [Isakowitz et al 95]. Additional navigation constructs introduced in this paper are the "Cross-reference", "Heterogeneous Guided Tour", "User Search", and access under "Program Control." For completeness, we briefly overview and illustrate each of the conventions shown in Figure 12.

A "Grouping" represents a starting point from which the user can take different paths to access the information in the application. The Title (or entry screen) for Encarta'94 shown in Figure 2 is an example of a grouping as are the menu bars for the "Topic Screen" shown in Figure 3.

The remaining constructs in Figure 12 represent RMM's classification of the navigation paths that are possible in a HM application. Each navigation path, N, is an automated 1:1 mapping from a particular set of information elements, E, *into* a new set of information elements E':

$$N: E \xrightarrow{m}{}^{P(E')} E'$$

Here, E and E' are sets of information entities from the ER/RM Diagrams. A movement from E to E' is related to, but is not isomorphic to, changes on the user's screen as explained above.

E = a Grouping, Web, entity or object class indicated at the tail of the arrow.

E' = another Grouping, Web, object class, a single entity instance or a subset of entity instances from the same or different classes as indicated at the head of the arrow.

P = a predicate that specifies a subset of the range, E', of the navigation mapping. If P is omitted, then the entire range is assumed.

m = the access mechanism to be used (this is why we called N an "automated" mapping.) An access mechanism specifies the nature of the interaction with the user in moving from E to E' and, in some cases, the order in which the information in E' is accessed.

The access mechanisms, m, provide the designer with a rich set of choices. The simplest access mechanisms are unidirectional and bi-directional *direct access* which are depicted by arrows. These correspond to hyperlinks in the application. A predicate, P, can be placed on a direct access arrow as in the RM Diagram for Encarta's Title Screen (Figure 13(a)) where the designer has specified that the Topic to be accessed is the last one from the previous Encarta session.

A *Cross Reference* from one entity to another (or to itself) indicates the existence of one or more hyperlinks between the source and target entity classes. Cross-reference links arise from the desire of the HM author to make associations of content between entity instances. Unlike other hyper links in RMM, they do not follow a fixed pattern and are usually hand-crafted rather than automatically generated. For example, Encarta contains a rich set of content-based cross-references between Topic instances as indicated by the cross-reference pointer and the notation "Related Topic" in Figure 13(b).

The remaining access mechanisms involve more user interaction. The *Index* navigation mechanism specifies that an index pointing to instances of E', the target entity set, is to be presented to the user. The user than specifies one or more specific instances of E' that

he/she wishes to visit. As can be seen from Figures 13(a) and 13(b), index access is very common in Encarta. Among other examples, index access is provided from the main screen to all topics and via the Gallery object to the various classes of Gallery Item or to various sub-classes, P="x", where x is a logical constraint or predicate specified by the user.

A *Homogeneous Guided Tour* specifies that a subset (often specified by the predicate, P) of the target entity instances from a single entity class is to be traversed. The user is restricted to movement along the specified path until he/she reaches the end. Different kinds of Guided Tour can be specified (in step 7 below.) For example, the user is usually allowed to move backwards and forwards along the path and is provided with an "escape" to the starting point. Thus, the Encarta Highlights Guided Tour in Encarta (see Figure 13(a)) leads the user through a fixed sequence of Highlight entity instances with "Next", "Prior" and "Close" buttons. A *Heterogeneous Guided Tour* provides a similar mechanism involving a tour of entity instances from different entity classes. For example, in Encarta, a non-homogenous guided tour providing a lecture on "Art History" might lead the user through a sequence of entity instances chosen from Topics, Gallery Items, Timeline Events, etc. Heterogeneous Guided Tours are usually hand-crafted. For instance, some HM systems allow users to compose and store their own guided tours by simply recording the sequence of their visits to information items. Heterogeneous Guided Tours usually start from a menu or Grouping point. To save clutter on the RM Diagram, the target, E', is simply specified as a list of the entity classes involved in the tour (see Figure 12.) An *Indexed Guided Tour* (of either type) specifies that an index of target entity instances is to be presented to the user so that he/she can choose the starting point for the tour. Again, the sophistication of the index can vary from a simple list to a list that highlights points already visited by the user. This kind of detail is specified later in the design process.

The *User Search* access mechanism allows the user of the HM application to input a search string and returns the entity instance(s) satisfying the request. In most cases, the starting point for a User Search will be a grouping. Depending on the application, the target for the search might be a single entity set as in the Keyword Access provided by the Find Wizard in Encarta to the Topic Class (Figure 13(b)). In other cases, the target might be a set of heterogeneous entity classes - in which case, the target entities are simply listed at the end of the arrow as for a heterogeneous guided tour. User Search comes in two flavors depending on how the results are to be presented to the user - as an Index or as a Guided Tour. To indicate this, we add the symbol for an index or guided tour to the outgoing arc. For

example, in Encarta'94, the identifiers of the Topic instances that satisfy a query entered using the Find Wizard are presented to the user in an index (see Figure 13(b).)

Program Control is the most general navigation mechanism. In this case, the user's access to information is through interaction with a special computer program. In Encarta, the Category Tree, Atlas, and Art Gallery object classes are examples of Program Control mechanisms. The launching point, E, for the program (usually a Grouping) together with the entity classes, E', that are accessed are shown in the RM Diagram. Details of the program design are left to step 7 of the methodology. Again, the Index or Guided tour symbols can be attached to the outgoing arc to indicate the results of the interaction. An example is the Gallery Wizard object in Figure 13(b), which actually produces either an index of Topic entities satisfying a particular user query, or one of several pre-coded guided tours.

The navigation paths involving the Topic Entity context (see Figures 3 and 10) are shown in detail in Figure 14. Most of this information could have been shown in Figure 13(b) but was omitted to avoid clutter. Note that Figure 14(b) shows the information items that can be accessed from the *current instance* of Topic - its own Contributor(s), Bibliography, associated Gallery Items, and so on - and not those of other Topic instances.

Figure 14 - Encarta'94 : RM Diagram for Topic Context

Most of the constructs in Figure 14 have been explained already. For example, the Category Graphic supports access to other Topics in the same category either via an index (this is the meaning of the "List" button in Figure 3) or via a Guided Tour (the "<" and ">" buttons.) One can also go directly to the Category Browser (via the middle button under the Category Graphic.) Figure 14 specifies that the user can take a Guided Tour of all the Gallery Items related to the current Topic (via the "<" and ">" buttons below the Gallery Item graphic in Figure 3.) Finally, additional aids are to be provided to help the user traverse long scrolling texts. The first aid is a pop-up index (outline) into the text (obtained by clicking on the "Outline" button shown at the bottom of Figure 3.) The second aid is the specification of "internal links" or hot buttons within the text that allow one to jump directly to different parts of the text (this is indicated by the loop within the text slice icon in Figure 14.)

Finally, we return to the Academic Department Homepage example to illustrate the use of RM diagram conventions in the design of webs (Figure 15). The Department Home Page is specified as a Group (menu) allowing access to Faculty entities (via an Index), to course entities (via either a Guided Tour or an Index), and to the webs for the Research Center and

Student Clubs (Figure 15(a)). The RM diagram for the Research Center web consists of a home page containing the content of the "Research Center Mission" entity from Figure 8(b). It is depicted functionally in Figure 15(a) as a Group with direct links to pages describing the history of the center, research grants, and so on. Note that the designer has attached a predicate to the index of events specifying that the only research seminar events are to be accessible from the "Activities Summary" page.

Figure 15(a): Academic Application: main RM Diagram
and *(b): Academic Application: RM Diagram for Research Web*

Step 4: Information Content of Screens

A recent extension to the original RMM model developed by [Isakowitz et al 97] provides a rigorous specification of the content of the screens in the application. The objective is to provide a machine-readable definition of the presentation units (screens, windows, hypertext anchors, text, images, and so on) that will be seen (and/or heard) by the user. The presentation units are called m-slices in RMM and are a generalization and formalization of the concept of a slice introduced earlier. They are defined as nested hierarchies with the highest m-slice representing a complete screen, which is comprised of lower level m-slices. Each m-slice can contain other m-slices until the total information content of the screen has been specified. (The "m" in "m-slice" comes from the nested structure of Russian Matrjeska dolls.) Note that database attributes (such as "Contrib-Name, etc., in Figure 11) are elementary m-slices whenever they appear on the screen. The distinction between attribute values as they appear in the database and attributes as m-slices is an important one since the latter may be "decorated" and formatted for visual display.

It must be emphasized that the purpose of this design step is to specify the content and information retrieval characteristics of the m-slices - not their physical appearance. In their original form [Isakowitz et al 97, 98], the m-slices were essentially database "views" that were entirely expressible through a database retrieval language such as SQL. In this paper, we extend the concept of m-slices to include calls to executable components (the objects specified in the earlier steps.) The m-slice specifications are intended to be executable during the later conversion step (Step 7.) For example, they might be translated into HTML and Java or to any other hypermedia language such as Macromedia Director [Macromedia 97]. The Entity Diagrams of Step 2 and the navigation structures in the RM Diagrams of Step 3 are the starting points for the definition of the m-slices.

Figure 16 provides a specification of some of the important m-slices in Encarta'94[1]. We will start our description with the top-most screen (the Title screen in Encarta) and proceed down the hierarchy of screens and m-slices. (However, it is also possible, and sometimes easier, to design in a bottom-up fashion.)

Figure 16: Partial Specification of M-Slices in Encarta

Naming Conventions: In keeping with the Entity Context concept introduced above, an m-slice is *owned* by one specific "focal" entity, or, in the case of the top-most m-slice by the application itself. M-slice names therefore take the form: *<owner entity>:<slice name>*. Examples of m-slices in Figure 16 are Encarta:Title-Screen and Topic:Topic-Text where Encarta is the application name and the Topic entity is the owner of Topic-Text. The latter consists primarily of the Text attribute from Topic in the database design (Figure 11). As mentioned above, database attributes are also m-slices but, for clarity, we will indicate database attributes using the convention: *<table name>.<attribute name>* where a period is used as the separator rather than a colon. An example from Figure 16 is Contibutor.Contrib-Name which appears in theTopic:Contributor m-slice. Literals are enclosed in double quotes and can appear in a number of places in the specification.

The definition of the Encarta:Title-Screen m-slice in Figure 16 specifies the contents of the Title Screen (see Figure 3) in a straight forward manner. First, the designer specifies that the literal, "Encarta, 1994 Edition" is to appear on the screen and that an Audio file is to be played. The remainder of the opening screen is an RMM Grouping - in other words, the opening screen is to be a launching pad for the application. As shown in the RM Diagram, Figure 13(a), the user can move to an Index of all the topics in the application, to the Category Browser, Gallery-Wizard or Find-Wizard objects, to a guided tour of Encarta highlights or directly into the main topic screen for Encarta. In essence, the opening page consists of a series of *hyperlinks* that are specified using the syntax:

Notation: *[relation] * <anchor> => <destination>*

Examples: [Topic-Contributor] * Contributor.Contrib-Name => Contributor.Credentials;
* ("Category Browser", Icon(Cat-Browser)) => exec Category-Browser Top;

The optional [relation] component in the specification for a hyperlink specifies a database relationship or an access structure from the RM Diagram that is to be used to access the destination associated with the current m-slice's owner entity instance. Thus, in the first of the

[1] A set of graphic conventions for m-slices has also been developed (see [Isakowitz et al 97].)

two examples given above (from the Contributor m-slice definition in Figure 16), [Topic-Contributor] refers to the many:many Topic-Contributor relationship that was specified as a database relation consisting of two foreign keys in Figure 11. If the destination information is associated with the current entity instance, the notation [this] where "this" represents the current instance can be used or the relation part of the specification can simply be omitted. The relation part can also be omitted when, as in the second example above (from the Title Screen m-slice), the hyperlink activates an object such as the Category-Browser.

An asterisk separates the relation part from the anchor part of the hyperlink specification. The anchor specifies the visible text or image on which the user clicks to activate the hyperlink. In the Topic-Contributor example above, this consists of the value of the Contrib-Name attribute from the Contributor database table. In the second example, the visible anchor consists of the literal "Category Browser" accompanied by the "Cat-Browser" icon, which is stored in the Icon relation in the database. (Note that the parentheses enclose the identifier for the icon element to be retrieved.)

The arrow, "=>", separates the anchor from the destination part of the hyperlink specification. In the first example, the destination is the m-slice for the Credentials attribute from the Contributor table in Figure 11. The second example above shows how dynamic elements such as objects or Java scripts can be specified using the "exec" keyword followed by (pseudocode for) the message that will invoke the object. Thus, control is to pass to the Category Browser object with the "Top" message specifying that the top-most category view is to be shown to the user. In Figure 16, a method, Encarta-Topic-Screen, is introduced in the specification of the Title-Screen m-slice to handle the dynamics of the user interaction for the Topic-Screen m-slice. For example, as noted on the Entity Diagram in Figure 14, the designer wants the Gallery Items associated with an entity instance to change as the topic text for the entity is scrolled[2].

If the user clicks "Enter Encarta" from the main screen, the Encarta:Topic-Screen m-slice will appear (see Figures 3 and 13(b).) The definition of the Encarta:Topic-Screen m-slice in Figure 16 specifies that the Topic-Screen m-slice is composed of a menu-bar and three m-slices - Topic:Topic-Text, Category:Graphic and Gallery-Item:Window. The notation to specify an m-slice (within a parent m-slice) is as follows:

Notation: [relation] <m-slice>

[2] It would have been possible to show Topic in the original object-class diagram of Figure 7 as an object. However, we chose to emphasize the database view in Figure 7 as our strategy is to store the Topic data in a relational database.

Examples: [this] Topic:Topic-Text;
[Topic-Gallery] Gallery-Item:Window;

(These two examples come from the definition of the Topic-Screen m-slice.) Note that the database relation or RMD component that is to be used to access the m-slice is specified followed by the name of the m-slice. If there is no m-slice for a particular entity instance (as occurs often for the Gallery-Item), the relation will give a null result and the corresponding portion of the screen will be empty. Recall also from our earlier discussion, that a single attribute is an m-slice by default.

As indicated in the Topic:Screen m-slice, a menu-bar is specified using the following syntax:

Notation: <menu-bar name>: **menu-bar**
begin
<button>
<button>
.....
<button>
end menu-bar

Example: Topic-Screen-Top-Menu: **menu-bar**
begin
* "Menu" => **exec** Main-Menu;
* "Contents" => **index** Contents;
.....
* "Help" => **exec** Help
end menu-bar

Here, **begin**, **end** and **menu-bar** are keywords and a button is specified using the same syntax as for a hyperlink. Note that only the functional characteristics of the menu bar are specified in this design step - its physical appearance and placement are specified in step 5 below.

Finally, the Contributor and Bibliography m-slices illustrate the *index* construct:

Notation: <index name>: **index**
begin
[relation] <index content>
end index

Example: <Bib-Index>: **index**
begin
[Topic-Bibliography] * Bibliography.Bib-ID => Bibliography.Bib-List
end index

An index specification causes an iteration of <index content> through all instances of the relation that are related to the current m-slice owner instance. Note from Figure 3 that there

can be more than one contributor and more than one bibliographical list for an article and that these appear at the end of the topic text as lists of buttons. In the Bib-Index example, the relation Topic-Bibliography is accessed to obtain all the bibliographic lists associated with the current Topic instance. The <index content> consists of pointers to the Bib-List attribute of the Bibliography database table. When the user clicks on one of these references, the associated bibliography appears in a pop-up window.

The specifications in Figure 16 for the remaining m-slices (Category:Graphic, Gallery-Item:Window and Gallery-Item:Gallery-Title) use no new elements of the m-slice specification language and should be readily understandable when viewed in conjunction with the Topic Screen in Figure 3.

Step 5: Visual aspects of screens, sound, video, etc.

Once the content of each screen has been designed, it is up to the graphics designers to compose an aesthetically pleasing and compelling presentation. This involves the assignment of visual attributes (size, font, color, image maps, sound accompaniment, and so on) to each of the screen's m-slices. While advanced Editors such as Shockwave and Adobe Photoshop can assist in the artistic process, this step of the design can not easily be formalized and is not addressed by the RMM methodology.

Step 6: Conversion protocol design

The preceding steps are largely independent of the system chosen for implementation. In this step, the structural and navigational features that have been specified in the E-R and RM diagrams are mapped to the implementation system. For example, an RMM index might be implemented as an unordered or numbered list of "anchors" in HTML. Hypermedia building kits such as Macromedia Director [Macromedia 97], provide methods and tools that implement the various grouping, indexing and guided tour features shown on the RM Diagrams. Such tools are now also becoming available for Web development [Scharl 98]. Most implementations of RMM have performed these conversions manually. However, we are now in the process of developing an RMM to HTML compiler, which will take an m-slice specification (see Figure 16) and a database design (see Figure 11) as input and produce a hypermedia application template as output [Diaz et al 95].

Step 7: Dynamic aspects of the HM application

The final design step involves developing the dynamic requirements of the application. These are determined by the required functionality and can include such requirements as the synchronization of audio and video elements and designing the interaction of users as they fill-

in forms or otherwise interact with the HM application. For instance in a "Slide Show" (guided tour) of Art in Encarta'94, certain pictures (JPEG) can be accessed only if others (one or more predecessors) have been accessed. In general, such restrictions can range from simple data retrieval procedures to restrictive rule-based retrieval of data where the rules may have a three pronged structure consisting of When, If, and Then clauses. As an example, consider the rule:

When - a user attempts to visit Visual element N (say a painting) -
If - User belongs to the context of Guided Tour G (say a tour of renaissance paintings)
AND - **if** User has visited Painting N-1,
Then - the element may be accessed using data retrieval mechanism - X (use a JPEG Viewer or TIFF viewer etc.)

Complex requirements such as these would be handled by the object classes specified in earlier steps in the design and would be designed and developed using normal software engineering methods. In other cases, the dynamic aspects of the application are quite routine and can be handled by the hypermedia software engine itself. This is the case in most web applications such as the JMIS site mentioned in section 1. If this is the case, no objects need be specified and the development process can be simplified accordingly.

Summary: The seven steps of the RMM Design methodology that we have described in this section represent a distinct way of decomposing the overall problem of hypermedia design. They can be characterized as follows: (1) specification of the information content; (2) grouping of the content into distinct contexts; (3) specification of the logical paths through the content; (4) spatial organization of the content by the m-slice design, which defines the content of the screens and/or windows that are viewed by users; (5) specification of the artistic and aesthetic design elements of the interface; (6) conversion of the design for implementation on the chosen software engine; and, (7) specification of the temporal aspects of the media such as their synchronization and the specification of the user interaction. For clarity, we have shown each of the first four steps in detail. In practice, some of the diagrams that we have shown might be omitted - for example, HM designers might omit context entity diagrams such as 8(b) and 10 and proceed immediately to the corresponding RM diagrams (Figures 14 and 15(b)).

5. Encarta'97: The Evolution of a Genre

As part of our investigations of the adequacy of RMM, we have also analyzed the design of a number of other encyclopedias. We have found the exercise of developing E-R and RM diagrams useful in comparing multimedia designs. This corroborates the findings of [Garzotto et al 95] using the HDM design model. For example, the RM Diagrams for the 1997

version of Encarta are shown in Figures 17(a) and (b). Comparing these with the equivalent diagrams, Figures 13(a) and (b) for Encarta'94, we notice a number of structural changes. At a superficial level, a number of features have new names: "Topic" becomes "Article", "Find Wizard" becomes "Pin-Pointer", and "Gallery Wizard" becomes "Media Features". More fundamentally, one can immediately notice a number of new features in Encarta'97. The most important of these is the attempt to integrate the encyclopedia with the World Wide Web (the "web" construct accessed by the "Online Features", "Library", Links" and Year Book" objects.) A related feature is the "Subscriptions" object in Figure 17(a), which allows one to subscribe automatically to Microsoft Network and other services such as monthly updates to the yearbook. It is also interesting to note the features that have been dropped from Encarta'94. The most conspicuous of these are the Category Graphic and the Category Manager that were shown in Figures 3 and 5, respectively. Evidently, the designers felt that positioning articles in a hierarchy of subject categories was not useful to many readers, or at least not as useful as other, newer features that were introduced in Encarta'97.

Figure 17(a): Encarta'97 RM Diagram for Title Screen
and *(b): Encarta'97 Main RM Diagram*

Of course, the structural changes revealed by E-R and RM diagrams are only part of the story. Over time, all of the encyclopedias have improved the quality of their content and squeezed more video and interactive features onto their CD-ROMS. However, a careful examination of the RM diagrams for various encyclopedias provides many insights into alternative designs and can facilitate a quantitative evaluation of the efficiency of the designs relative to different user tasks such as searching for a topic, backtracking to a previous topic, and so on. We will report on the use of the RMM design model for "multimedia criticism" in a future paper.

6. Conclusions and Future Research

The research described in this paper is part of an ongoing program aimed at developing approaches and tools for hypermedia design and implementation. Here, we investigated the adequacy of the RMM model as a vehicle for hypermedia design. By reverse engineering an existing multimedia product, Encarta'94, we discovered several new requirements and developed appropriate extensions to the RMM model. With these extensions, we demonstrated that RMM is capable of representing a quite complex commercial product.

Future research on RMM will evolve in three directions. First, we will continue to investigate extensions and refinements of the RMM methodology. In this regard, a future paper will develop the detail of the methodology for the later design stages (steps 5, 6 and 7)

that were not described in detail in this paper. Second, we continue to work on the development of the RMCase [Diaz et al 95] tool for automatic generation of WWW applications. Finally, as mentioned above, we will investigate the use of RMM and other design models for critical analysis of alternative hypermedia designs.

Although this paper has focused on a single HM design approach, the issues that we have discussed are general in nature. These issues include the integration of the static and dynamic elements of hypermedia into a single design representation, the representation of a broad range of navigation mechanisms, and techniques for the formal specification of the content of screens. The research approach in this paper is essentially experimental in nature. Briefly, we decided to test the RMM representation scheme on existing commercial hypermedia and multimedia and products. This has the advantage of removing "developer bias" in assessing the adequacy of the design tool and, as we have shown, can reveal gaps in capabilities of the representation that might not otherwise have been obvious. While more research must be performed, the RMM design representation that we have developed seems quite promising. Nevertheless, there is much to learn about hypermedia design and a number of different approaches have been proposed in the literature. Experience using these various design approaches in industrial strength applications will represent the ultimate test of their relative worth.

References

1. Blaha, Michael R., William J. Premerlani and James E. Rumbaugh, "Relational Database Design Using an Object-oriented Methodology," *Communications of the ACM*, Vol. 31, No. 4, August 1988, pp. 414-427.
2. Chen, P.P.S, "The Entity Relationship Model: Toward a Unified View of Data," *ACM Transactions on Database Systems*, Vol. 1, No. 1, March 1976.
3. Diaz, A., T. Isakowitz, V. Maiorana and G. Gilabert, "RMCase: A Tool to Design WWW Applications," *World Wide Web Journal*, Vol. 1, No. 1, 1995, pp. 559-566.
4. Garzotto, Franca, Paolo Paolini and D. Schwabe, "HDM: A Model-based Approach to Hypertext Application Design," *ACM Transactions on Information Systems*, Vol. 11, No. 1, January 1993, pp. 1-26.
5. Garzotto, Franca, Luca Mainetti and Paolo Paolini, "Hypermedia Design, Analysis and Evaluation Issues," *Communications of the ACM*, Vol. 38, No. 8, August 1995, pp. 74-83.
6. Gorman, Kevin and Joobin Choobineh, "The Object-Oriented Entity Relationship Model (Ooerm)," *Journal of Management Information Systems*, Vol. 7, No. 3, pp. 41-65.
7. Isakowitz, Tomas, Edward A. Stohr and P. Balasubramanian, "RMM: A Methodology for the Design of Structured Hypermedia Applications," *Communications of the ACM*, Vol. 38, No. 8, August 1995, pp. 34-44.

8. Isakowitz, Tomas, Arnold Kamis and Marios Koufaris, "Extending the Capabilities of RMM: Russian Dolls and Hypertext", in R. Sprague (ed), *Proceedings of the Hawaii International Conference on Information Systems (HICSS 30)*, Volume VI, January 1997, pp. 177-186.

9. Isakowitz, Tomas, Arnold Kamis and Marios Koufaris, "Reconciling Top-down and Bottom-up Design Approaches in RMM", in R. Sprague (ed*), Proceedings of the Hawaii International Conference on Systems Sciences (HICSS 31)*, Volume VI, January 1998, pp. 201-210.

10. Lange, D. B., "An Object-Oriented Design Approach for Developing Hypermedia Information Systems", *Journal of Organizational Computing and Electronic Commerce*, Vol. 6, No. 8, 1996, pp. 269-294.

11. Macromedia Inc. *Macromedia Director*, San Francisco, CA, 1997.

12. Microsoft Corporation, *Encarta: The Complete Multimedia Encyclopedia, 1994 Edition*, Redmond WA, 1994.

13. Microsoft Corporation, *Encarta 97 Encyclopedia, 1994 Edition*, Redmond WA, 1997.

14. Nielson, Jakob, "The Matters that Really Matter for Hypertext Usability", *Hypertext '89 Proceedings*, 1989.

15. Scharl, Arno, "Reference Model of Commercial Web Information Systems Using the Extended World Wide Web Design Technique (eW3DT)," in R. W. Blanning and David R. King (eds), *Proceedings of the Hawaii International Conference on Systems Sciences (HICSS 31)*, Volume IV, January 1998, pp. 476-484.

16. Schwabe, D., and G. Rossi, "The Object-Oriented Hypermedia Design Model," *Communications of the ACM*, Aug. 1995, pp. 45-46.

17. Thuring, Manfred, Jorg Hanneman and Jorge M. Haak, "Designing for Comprehension: A Cognitive Approach to Hypermedia Development", *Communications of the ACM*, Vol. 38, No. 8, August 1995, pp. 57-66.

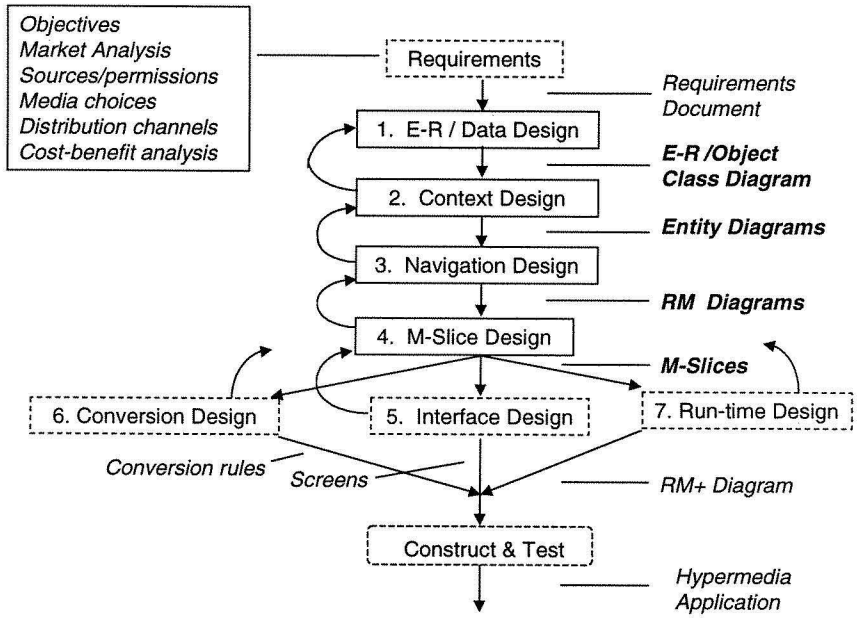


Figure 1 - Design Steps and Outputs

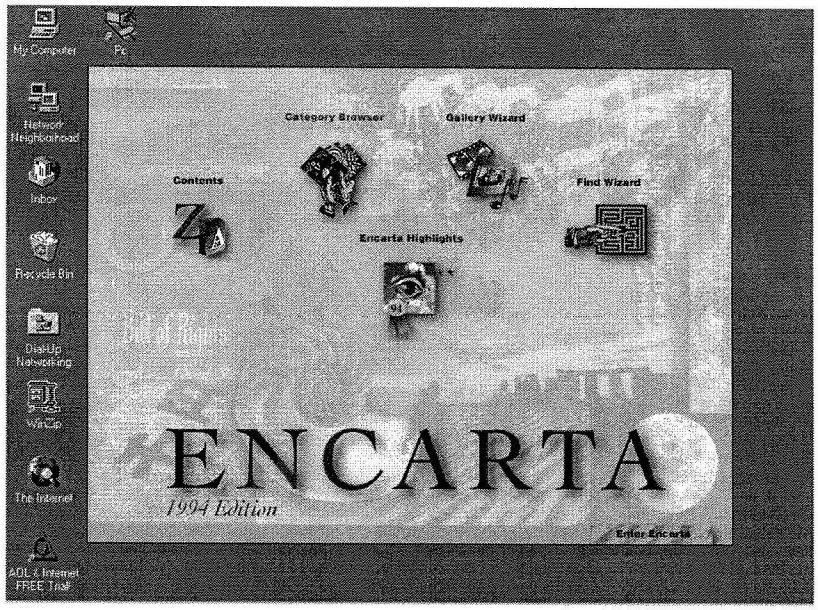


Figure 2 - Encarta'94 Title Screen

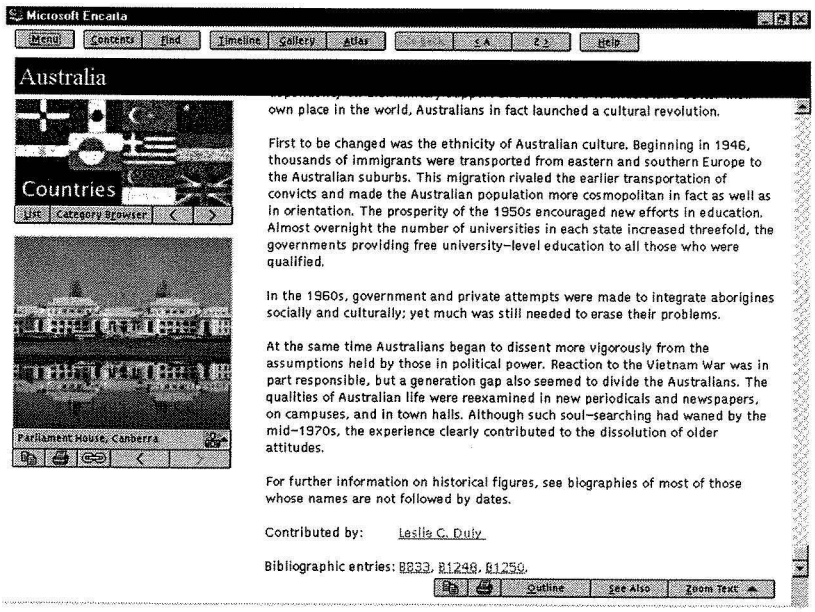


Figure 3 - Encarta'94 Topic Screen

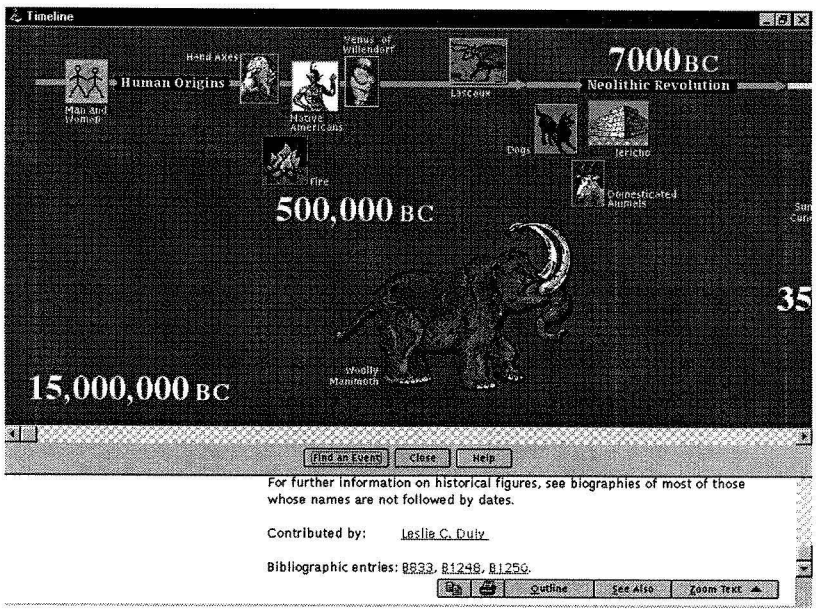


Figure 4 - Encarta'94 Timeline Screen

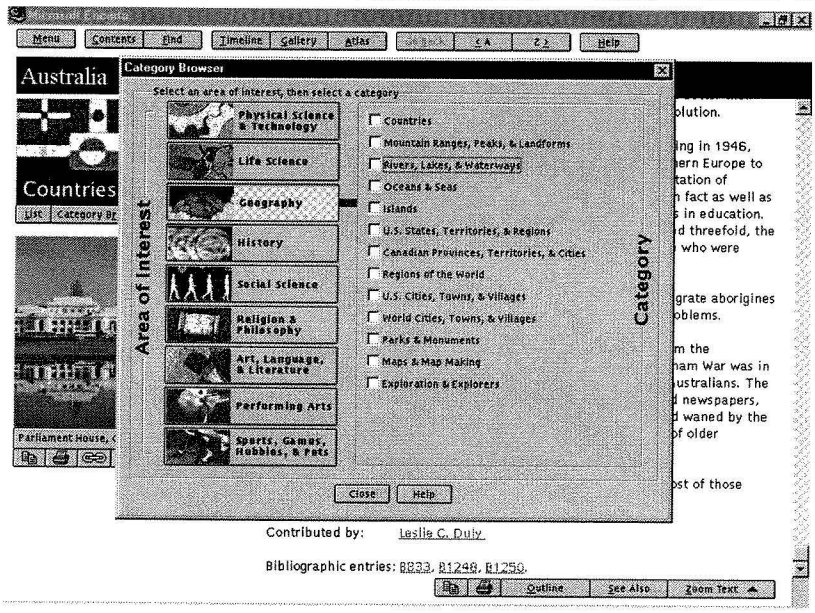


Figure 5 - Encarta'94 Category Browser

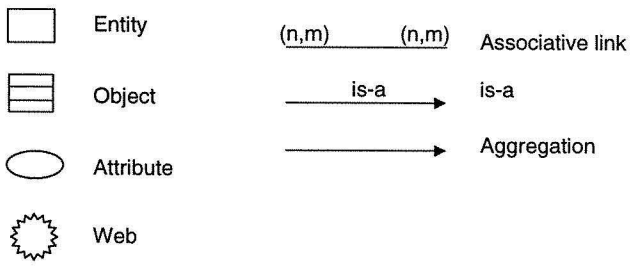


Figure 6 - E-R Diagram (Object Class Diagram) Conventions

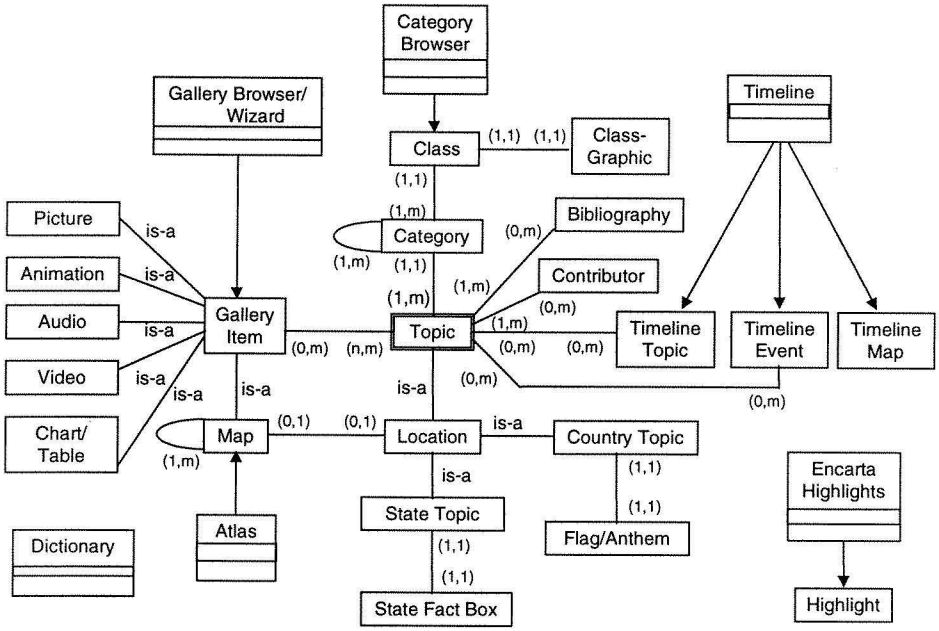


Figure 7 - Encarta'94: Object Class Diagram

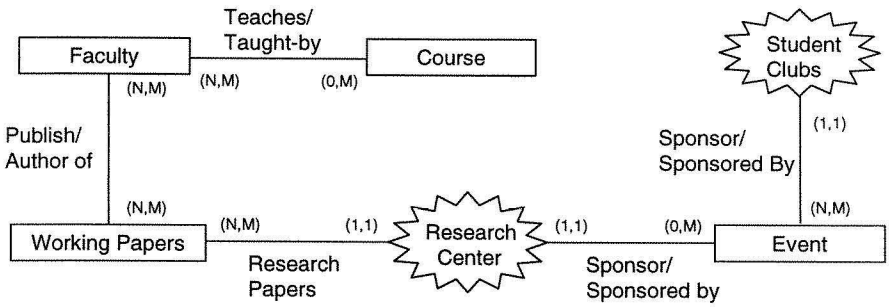


Figure 8(a) - E-R Diagram for Department Home Page

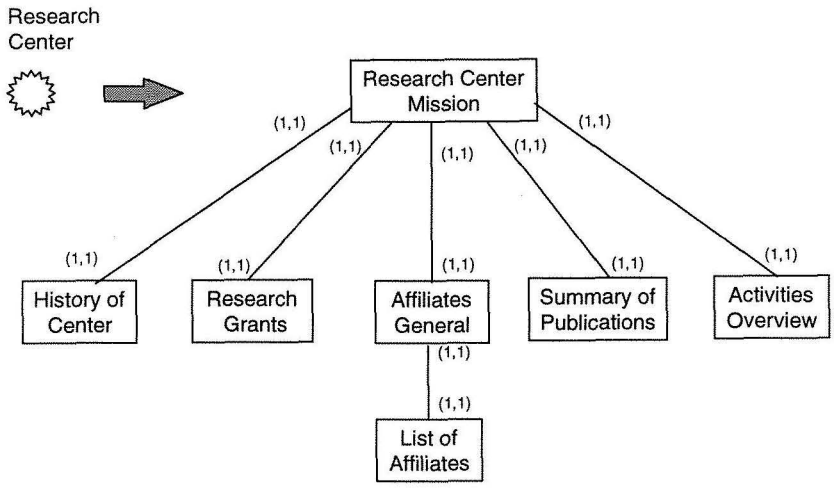
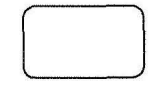
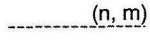


Figure 8(b) - E-R Diagram for Research Center Web



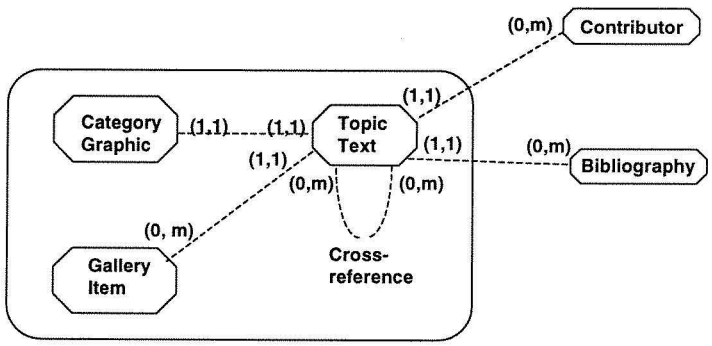
Screen outline - elements within the outline appear on the screen by default

Slice - set of attributes of an entity that constitutes a distinct unit of (multimedia) information (e.g., a video). Head slice - the default access point to an entity.



Structural link - access path between slices

Figure 9 - Entity Diagram Conventions



Notes:
As the text is scrolled, the associated gallery item changes dynamically.
Topics can cross-reference each other.
Bibliography and Contributor are pop-up windows in this context.

Figure 10 - Encarta'94: Entity Diagram for Topic

Class(Class-Name, Class-title, Class-Graphic)

Category(Cat-Name, *Cat-Parent-Name*, *Class-Name*, ... other...)

Topic(Topic-ID, *Cat-Name*, Topic-Title, Topic-Text, ... other ...)

Topic-Bibliography(*Topic-ID*, *Bib-ID*)

Bibliography(Bib-ID, Bib-List)

Topic-Contributor(*Topic-ID*, *Contrib-Name*)

Contributor(Contrib-Name, Credentials)

Location(*Topic-ID*, Location-ID, Loc-Name, Loc-Text, ...other ...)

Country-Topic(*Location-ID*, C-Name, C-Flag, C-Anthem, ... other...)

State-Topic(*Location-ID*, S-Name, State-Fact-Details, ... other..)

Topic-Gallery(*Topic-ID*, *Gallery-ID*)

Gallery-Item(Gallery-ID, G-Name, G-Text, ...other...)

Picture(*Gallery-ID*, Pic-ID, Pic-File, Pic-Size, ... other...)

Animation(*Gallery-ID*, Ani-ID, Ani-File, A-Size, ... other...)

Map(*Gallery-ID*, Map-ID, Map-File, Map-Size, ... other...)

Map-Location(*Map-ID*, *Location-ID*, ...other...)

Icon(Icon-ID, Icon-Body)

" " "

Legend: Primary keys are underlined, foreign keys are in italics.

Figure 11 - Encarta'94: Partial Database Design

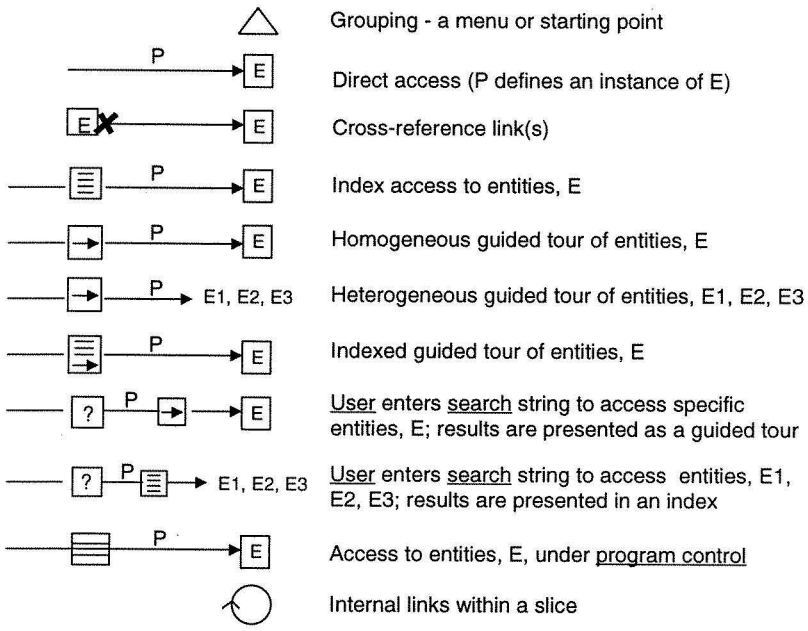


Figure 12 - RM Diagram Navigation Mechanisms

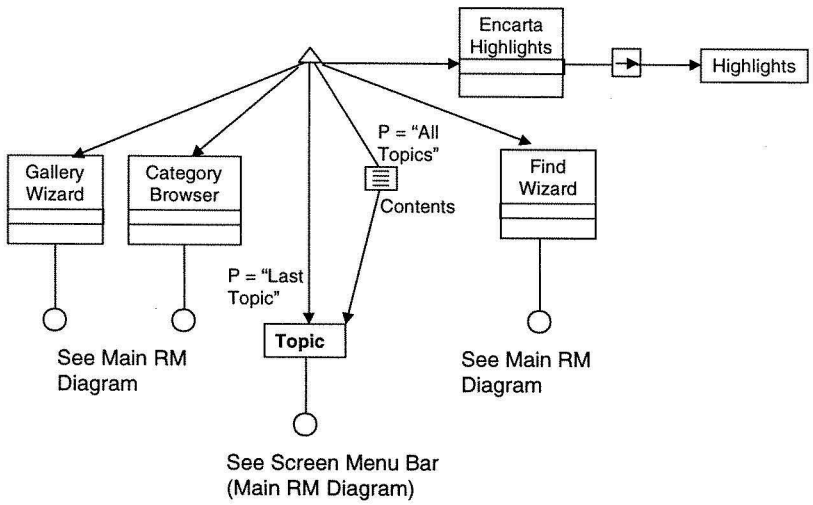


Figure 13(a) - Encarta'94: RM Diagram for Title Screen

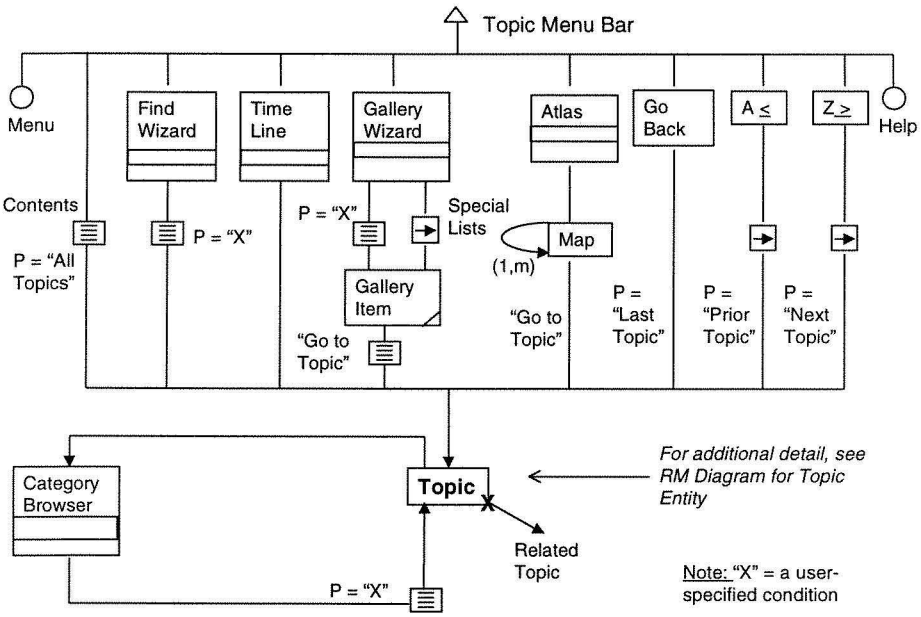
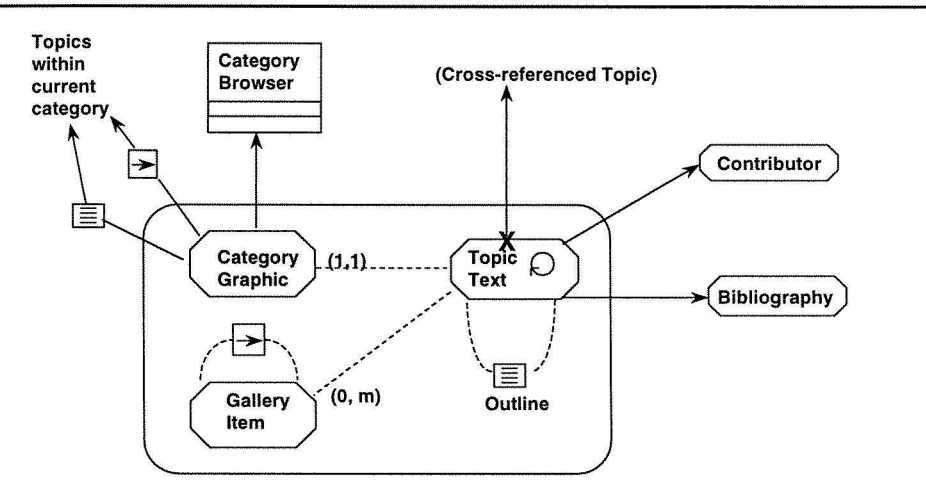


Figure 13(b) - Encarta'94: Main RM Diagram



Notes:
As the text is scrolled, the associated gallery item changes dynamically.
Outline, Bibliography and Contributor are pop-up windows in this context.

Figure 14 - Encarta'94: RM Diagram for Topic Context

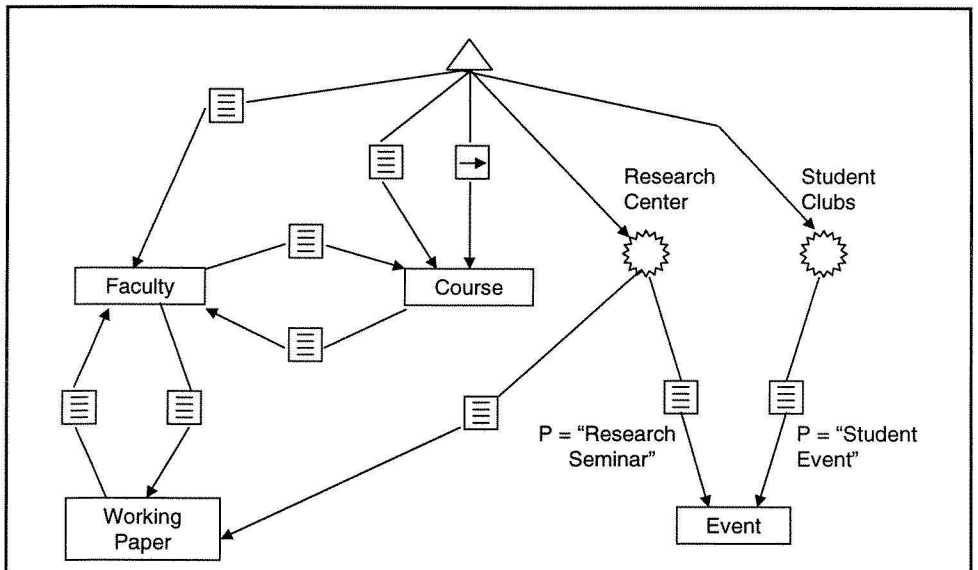


Figure 15(a) - RM Diagram: Department Home Page

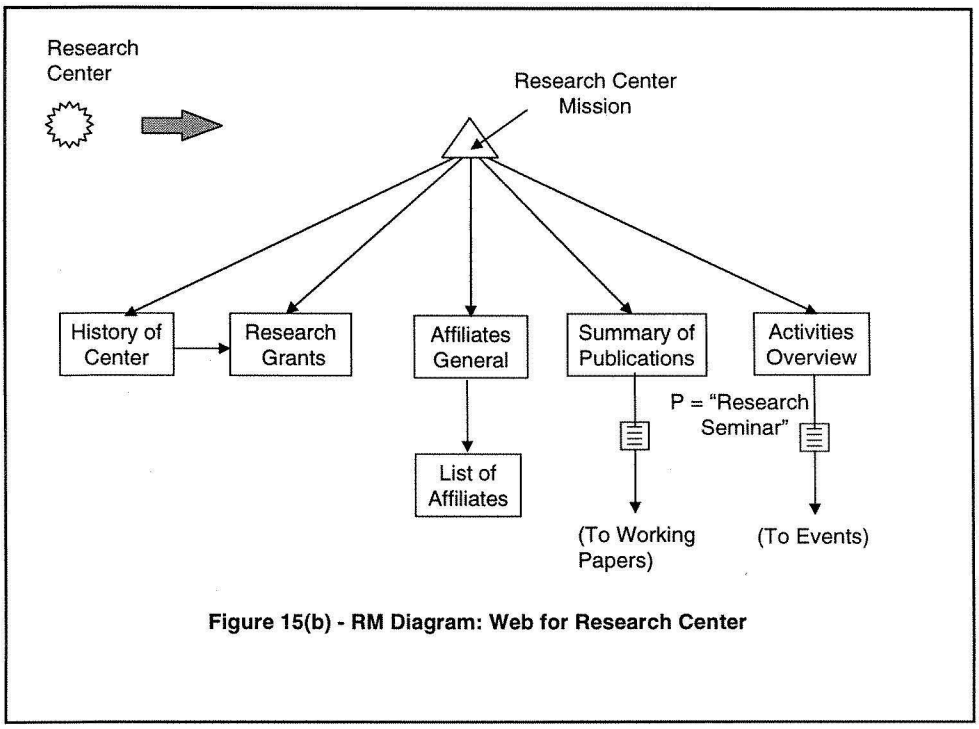


Figure 15(b) - RM Diagram: Web for Research Center

```
Encarta: Title-Screen: m-slice
begin
"Encarta, 1994 Edition";
Audio: encarta-main.au;
   begin grouping
        * ("Contents", icon(Contents)) => index Contents;
        * ("Category Browser", icon(Cat-Browser)) => exec Category-Browser Top;
        * ("Gallery Wizard", icon(Gallery-Wiz) => exec Gallery-Wizard Top;
        * ("Find Wizard", icon(Find-Wiz) => exec Find-Wizard Top;
        * ("Encarta Highlights", icon(Highlights)) => exec Highlights-Guided-Tour;
        * "Enter Encarta" => exec Encarta-Topic-Screen Last
   end grouping
end

Topic.Topic-Screen: m-slice
begin
   [this]Topic.Article-Text;
   [Topic-Category] Category.Graphic;
   [Topic-Gallery] Gallery-Item.Window;
   Topic-Screen-Top-Menu: menu-bar
   begin
        * "Menu" => exec Main-Menu;
        * "Contents" => index Contents;
                .....
        * "Help" => exec Help
    end menu-bar
end

Topic.Article-Text: m-slice
begin
   [this] Topic.Text;
   "Contributed by: " [Topic-Contributor] Topic:Contributor;
   "Bibliographic entries: " [Topic-Bibliography] Topic:Bibliography;
   Article-Text-Menu: menu-bar
   begin
        * icon(Clipboard) => exec Clipboard Topic.Text;
        * icon(Printer) => exec Print Topic.Text;
        * icon(Outline) => exec Outline Topic.Text;
        *  "See Also" => exec SeeAlso Topic.Topic-ID;
        * ("Zoom-Text", icon(Red-up-arrow)) => exec Zoom Topic.Text;
    end menu-bar
end
```

Figure 16: Partial M-Slice Specification for Encarta'94 (continued next page)

```
Topic:Contributor: m-slice
begin
        begin index
        [Topic-Contributor] * Contributor.Contrib-Name => Contributor.Credentials;
        end index
end

Topic:Bibliography: m-slice
begin
        begin index
        [Topic-Bibliography] * Bibliography.Bib-ID => Bibliography.Bib-List;
        end index
end

Category:Graphic: m-slice
begin
    [Topic-Category-Class] Class.Class-Graphic;
    Category-Menu: menu-bar
    begin
        * "List" => exec Category-Browser Index Cat-Name;
        * "Category-Browser" => exec Category-Browser Open Cat-Name;
        * icon(Left-Arrow) => exec Category-Browser Guided-Tour Backward Cat-Name;
        * icon(Right-Arrow) => exec Category-Browser Guided-Tour Forward Cat-Name;
    end menu-bar
end

Gallery-Item:Window: m-slice
begin
    [Topic-Gallery] Gallery-Item.Gallery-Id;
    [Topic-Gallery] Gallery-Item:Gallery-Title-Bar;
    Gallery-Menu: menu-bar
    begin
        * icon(Clipboard) => exec Clipboard;
        * icon(Printer) => exec Print Current;
        * icon(Link) => exec Link;
        * icon(Left-arrow) => exec Topic-Browser Guided-Tour Backward;
        * icon(Right-arrow) => exec Topic-Browser Guided-Tour Forward;
    end menu-bar
end

Gallery-Item:Gallery-Title-Bar: m-slice
begin
    [Topic-Gallery] Gallery-Item.G-Name;
     Icon(Gallery-item)
end
```

Figure 16: Partial M-Slice Specification for Encarta'94 (continued)

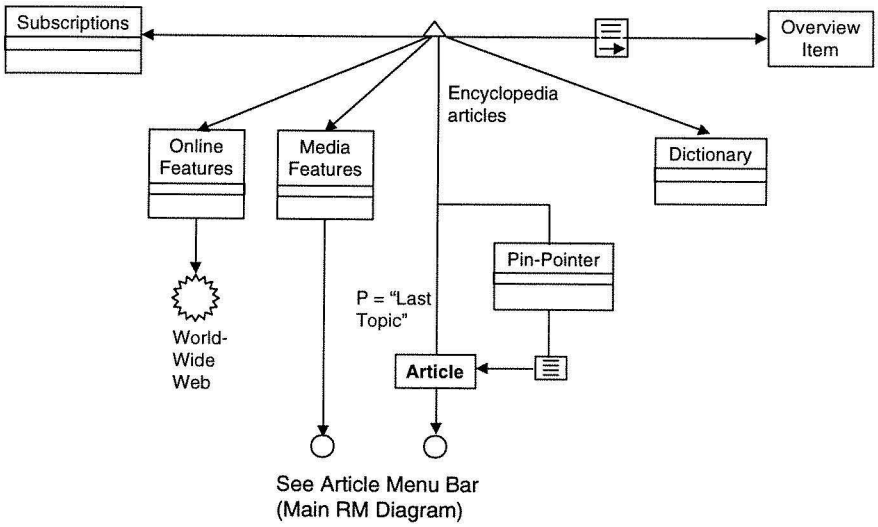


Figure 17(a) - Encarta'97: RM Diagram for Title Screen

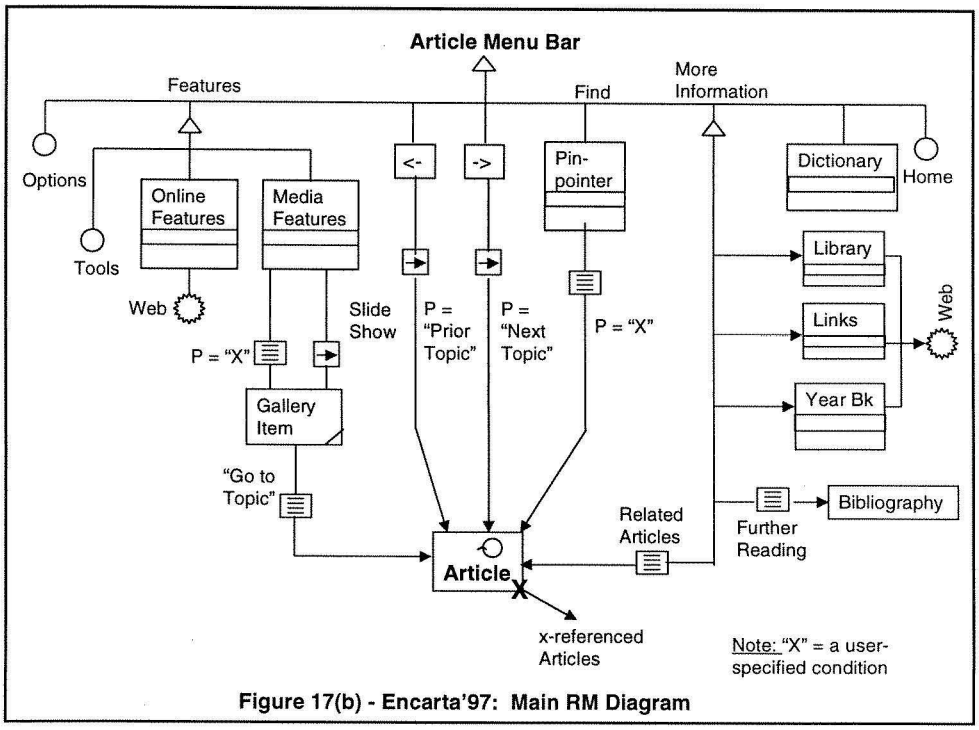


Figure 17(b) - Encarta'97: Main RM Diagram