A Navigation-oriented Hypertext Model Based on Statecharts

Marcelo Augusto Santos Turine*
Maria Cristina Ferreira de Oliveira**
Paulo Cesar Masiero**

*Instituto de Física de São Carlos
**Instituto de Ciências Matemáticas de São Carlos
Universidade de São Paulo
C.P. 668, 13560-970 — São Carlos, SP, Brazil
E-mail:[mast,ferreira,masiero]@icmsc.sc.usp.br

ABSTRACT
In this paper we present a navigation-oriented model for hyperdocument specification based on statecharts. The HMBS (*Hypertext Model Based on Statecharts*) model uses the structure and execution semantics of statecharts to specify both the structural organization and the browsing semantics of a hyperdocument. The formal definition of the model is presented, as well as its associated browsing semantics. A short discussion on the model's capabilities is also provided. A prototype hypertext system which implements HMBS as its underlying model for hyperdocument authoring and browsing is introduced, and some examples are presented that illustrate the application of the model.

KEYWORDS: Hypertext Document Model, Statecharts, Browsing Semantics, HMBS, Hierarchical Views.

1. INTRODUCTION

Over the last few years there has been a remarkable increase in the use of hypertext technology and the WWW (World Wide Web) for a wide variety of purposes ranging from advertising and marketing to collaborative work and teaching. For a lot of organizations and institutions the WWW provides a new and powerful opportunity to interact with (potential) customers and with users of their services in general. However, groups involved in the development of hypertext applications face a considerable number of problems. The design of such applications differs from traditional software development projects in several critical dimensions. It involves people with very different skills: authors, content designers, artists and programmers. The design requires from the development team the ability to capture and organize the structure of a complex domain making it clear and accessible to users. At the moment, the lack of guidelines and well-established system and design models turns the creation of hypermedia documents into a challenging and difficult process. Moreover, the resulting hyperdocuments tend to be confusing as far as the readers are concerned, and hard to maintain from the authors viewpoint [2]. The need for (semi) formal design approaches, as well as for systematic approaches for defining the structural organization of hyperdocuments has thus become imperative, and there is an ongoing research effort towards the creation of hypertext and hypermedia models. The use of suitable models helps to discipline the authoring activity by encouraging the development in a structured fashion, so that the structure is designed before the actual contents are filled into the document nodes.

Garzotto et al. [6] identify four distinct approaches to hypertext modeling. Namely, there are "*application-oriented*" models which explicitly model the semantics of specific application domains, such as the model employed in the g-IBIS hypertext tool [4]. A second class includes models that are "*system-oriented*", rather than application-oriented. These include, for example, the Dexter Hypertext Reference Model [7]. Such models try to identify the relevant abstractions found in a wide range of existing (and future) systems. A third class includes the "*behavioral methods*", such as the Trellis model [18] and Tompa's model [21]. These are concerned with the modeling of the dynamic behaviour associated to hyperdocument networks and with the browsing semantics allowed for navigating through the net. A fourth class comprises less formal approaches, which emphasize preferred topological structures as building blocks to create the structure of hyperdocument networks. Examples of approaches in this class are the linear structures used in Guide [3] and the hierarchical structures of KMS [1]. This classification may now be updated to include the so-called "*design models*", which provide a consistent set of steps that guide authors in the preparation of hyperdocuments. Examples of this class include HDM - Hypermedia Design Model [6], OOHDM - Object-Oriented HDM [17] and RMM - Relationship Management Methodology [10].

The model proposed in this paper, *HMBS* (*Hypertext Model Based on Statecharts*), uses the structure and execution semantics of statecharts to specify both the structural (hierarchical) organization and the browsing semantics of a hyperdocument. Stotts and Furuta [18] coined this term "browsing semantics" as referring to the dynamic properties of a reader's experience when navigating through a document, i.e., it is the manner in which information is to be visited and presented to the reader. In the context of the aforementioned classification, the HBMS model may be included in the class of behavioral models, due to its ability to model the browsing semantics of hyperdocuments. However, HMBS is also adequate for describing the hierarchical structure common to many hyperdocuments, for the hierarchy levels are directly mapped into the different levels of an underlying statechart model. In addition to the hierarchy mechanism, the model provides parallel and sequential decompositions with associated semantics.

Another major advantage of a statechart-based model is that it provides a mathematical model with associated semantics and algorithms, yet it has an easy and intuitive visual notation associated. It also has the advantage of being a very intuitive model, as statecharts are an extension of finite-state machines, and modeling based on states and events is well-established within the computer science community. Formal models may provide a standpoint for encouraging application portability among systems. Moreover, as statecharts were designed to model concurrent reactive systems, HMBS is suitable for describing concurrency aspects inherently associated to the navigation through a hyperdocument. Another use of statecharts in the context of hypertext modeling is suggested by Zheng and Pong [23], and its relation to our proposal is discussed in Section 5.

This paper is structured as follows. In Section 2 the formal syntax of statecharts is described, as well as the main features of the proposed model with respect to its use for specifying the structural organization and the browsing semantics of hyperdocuments. This section also presents a definition of hierarchical views for hyperdocuments. Section 3 describes simple solutions warranted by the model to some common problems related to hyperdocument visualization, browsing and analysis. Section 4 presents a prototype system for creating and browsing hyperdocuments that uses HMBS as its underlying model. A brief discussion of related work by comparing HMBS to other approaches is given in Section 5. A discussion on

the inclusion of additional statechart features into HMBS is provided in Section 6. Finally, the conclusions are presented in Section 7.

2. A STATECHART-BASED MODEL

Statecharts are a visual formalism for describing states and transitions that extends the classical formalism of finite state machines and state transition diagrams by incorporating the notions of hierarchy, orthogonality (concurrency), a broadcast mechanism for communication between concurrent components, composition, aggregation and refinement of states [8,9]. The statechart formalism provides an effective, concise, and intuitive notation for the specification and design of large and complex reactive systems. They have associated formal syntax and semantics, thus enabling the specification of behavioral aspects of systems in a clear, yet rigorous, manner, and providing means for formal verification and validation of models.

The human-computer interaction management module, or interface, of a hypertext system may be considered as a reactive system, as it must interactively attend to external events given in the form of user requests during browsing. For example, the activation of an anchor may result in a reconfiguration of the display to show new information. The *HMBS* (*Hypertext Model Based on Statecharts*) model uses the structure and execution semantics of statecharts to specify both the contents (including the linked structure) and the browsing semantics of a hyperdocument. This logical structure can be interpreted to generate a final product to be delivered to users. As in other models (e.g., Trellis [18,19]), hyperdocument contents and linked structures, as well as its logical structure and its mapping to specific physical representations, are effectively separated. Therefore, a single model representation may be used to generate different versions, or different presentations, of the same hyperdocument.

We shall briefly describe the formal syntax for statecharts. It should be noted that the syntax described is actually a subset of the one presented by Harel in [9], since not all the aspects of the original definition were included in the model. Shortly, a statechart structure (ST) is a 8-tuple $ST = <S,\rho,\psi,\delta,V,C,E,T>$ in which: S is the *set of states*; ρ: $S \rightarrow 2^S$ is a *hierarchy function* that defines the substates of each state (a state *s* is *basic* if $\rho(s) = \phi$); ψ: $S \rightarrow \{AND,OR\}$ is a function that defines the type of each state; δ: $S \rightarrow 2^S$, is the *default function*, defining the set of initial states which are contained in a state *s*; V is the *set of expressions* containing logical variables names; C is the *set of conditions*, which may be T (true), F (false) or logical expressions; E is the *set of event expressions*, also called *labels*; $T \subset 2^S \times E \times 2^S$ is the *set of transitions*. A *statechart legal configuration* (SC) is defined [8,9] as a maximal orthogonal set of *basic states*, representing a possible set of currently active states of the statechart. Figures 1 and 2 illustrate a statechart model that is going to be used as a running example along this paper.

2.1. Definition of the Model

The HMBS model provides a mechanism for effectively separating the hyperdocument physical and logical structures. The model uses the structure and execution semantics of statecharts to specify both the linked (logical) structure and the browsing semantics of a hyperdocument. The model definition is given below:

Definition: A *Hypertext H* is a 6-tuple $H = <ST,P,R,M,L,V>$ in which

- $ST = <S,\rho,\psi,\delta,V,C,E,T>$ is a statechart structure.
- P is a *set of pages* corresponding to the pieces of information included in the application. These consist of static (passive) media such as formatted data, text, images, graphics, tables, bitmaps, executable code or any other data objects supported by the hypertext system. The set P also includes a special *null page* which has no associated contents. Each element in set P is associated to two pieces of information: *page-title* and *page-contents*.

The first provides a unique identifier to the page, whereas the second describes its content information.

- R is a *set of readers*, or channels of presentation, which are abstractions used to specify the requirements involved in the presentation of information contained in the pages. A reader is an interpreter for a page, operating on the page as a whole and typically displaying the page through some medium (e.g., text formatters, graphic decoders, program interpreters, audio and video players, data manipulation systems, etc.).
- M: $S_S \rightarrow P$ is a *value function* mapping states into data objects (in the form of a page). Possible mappings are defined for states in a set S_S such that

$$S_S: \{x \in S \mid \psi(x) = OR \vee \rho(x) = \phi\}$$

  S_S is the subset of S comprising basic states and OR states. AND states are not mapped into data objects.
- L, the hyperdocument browsing level, is the visibility level ($0 \leq L \leq n$) used for defining the hierarchy depth when displaying pages during navigation (see Section 2.4).
- V: $P \rightarrow R$ is the *vizualization relationship* which associates each hyperdocument page with a single *reader* that is able to interpret it.

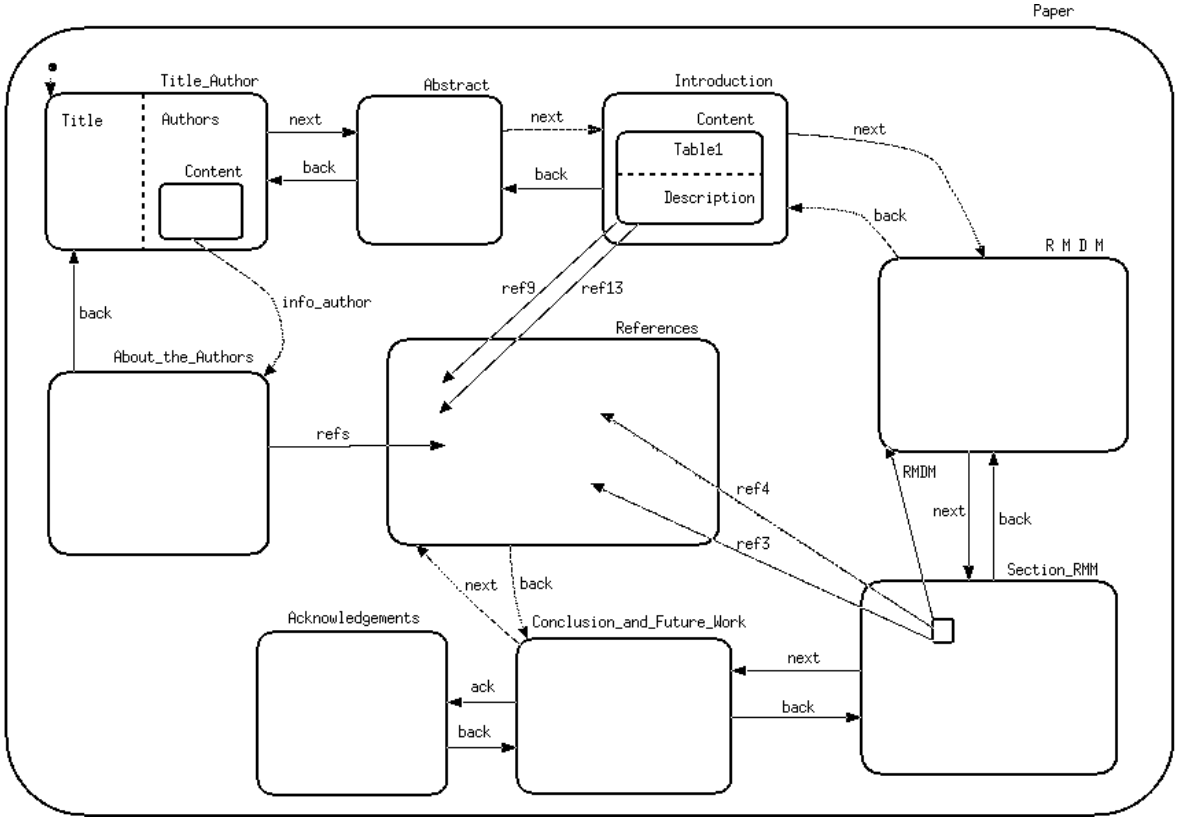


Figure 1. Statechart example: top level.

2.2. Browsing Semantics

In this section some examples are provided to illustrate the power and the flexibility of the proposed model to represent the browsing semantics of hyperdocuments. According to the model, a hyperdocument consists of a statechart structure representing the logical structure of the document in terms of nodes and links, the document's physical structure which are the "human-consumable" components (pages), and the associated display mechanisms. The execution semantics of the underlying statechart provides the model for browsing the hyperdocument.

In order to have an application suitable for use by a set of intended users trying to accomplish a certain set of tasks, it is necessary to map the information represented in the statechart structure into a navigational structure. Thus the nodes of the hyperdocument are mapped into a set of statechart states, and the set of possible link activations are represented by the set of events and transitions allowed in the statechart. Event activation in the statechart is thus used to represent anchor (or link) activation in the hyperdocument. Physically, such anchors may be indicated through buttons, menus or marked keywords within the hyperdocument. Thus clicking a button in the hyperdocument results in the triggering of an event in the underlying statechart model, consequently activating all links (transitions in the statechart) from the source node (i.e., its corresponding state in the statechart) to the destination node (state).

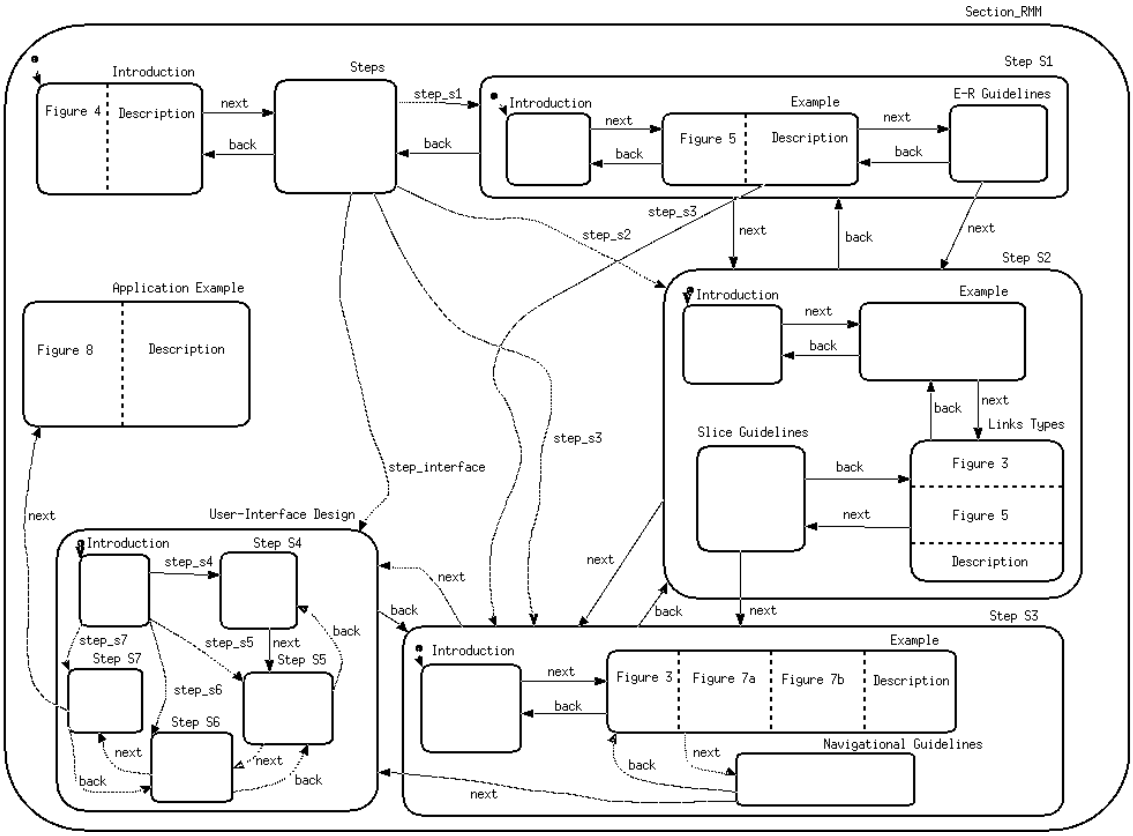


Figure 2. Refinement of state “Section_RMM”.

Within the statechart model, users are positioned at an arbitrary number of states at a time, having concurrent access to an arbitrary number of data objects. The current statechart configuration defines a *user state* in the hypertext system, that gives the set of currently active hyperdocument nodes at a certain stage during browsing, indicating which page contents are displayed for viewing. Nodes, or states of the set S_S, are mapped into pages of the set P according to the given function M. To view a page, it is necessary to invoke an associated reader that interprets and exhibits it through some medium. To visualize the pages associated to the current hyperdocument configuration readers must be invoked for every state in the underlying statecharts current state configuration. Pages associated to ancestral states may also be viewed in conjunction with those associated to currently active states, a facility which is useful for allowing users to simultaneously visualize different levels of the hyperdocument hierarchy as mapped into the statechart. The hierarchy level to be displayed is specified through a visibility level L, described below.

If L = *n then* display all pages p in set D_{all}, given by

$$D_{all} = \bigcup_{i=0,\ldots,n} D_i \text{ where } D_i = \{p \mid x \in S_S \wedge M(x) = p \wedge \rho^i(x) \cap SC \neq \phi\}$$

Under the statechart model, browsing is thus interpreted as follows. When a user selects a link, the hypertext system browser performs the following actions:

- generates an event which causes the triggering of the transition associated to that link. The source set for the transition must be an active set, that is, it must be included in the current state configuration;

- activates all states which are target sets for the transitions fired, thus generating the next state configuration and disabling the previous current state configuration;

- invokes the readers for the pages associated with those nodes corresponding to states in the new configuration and those within the scope of the visibility level L.

2.3. An Example

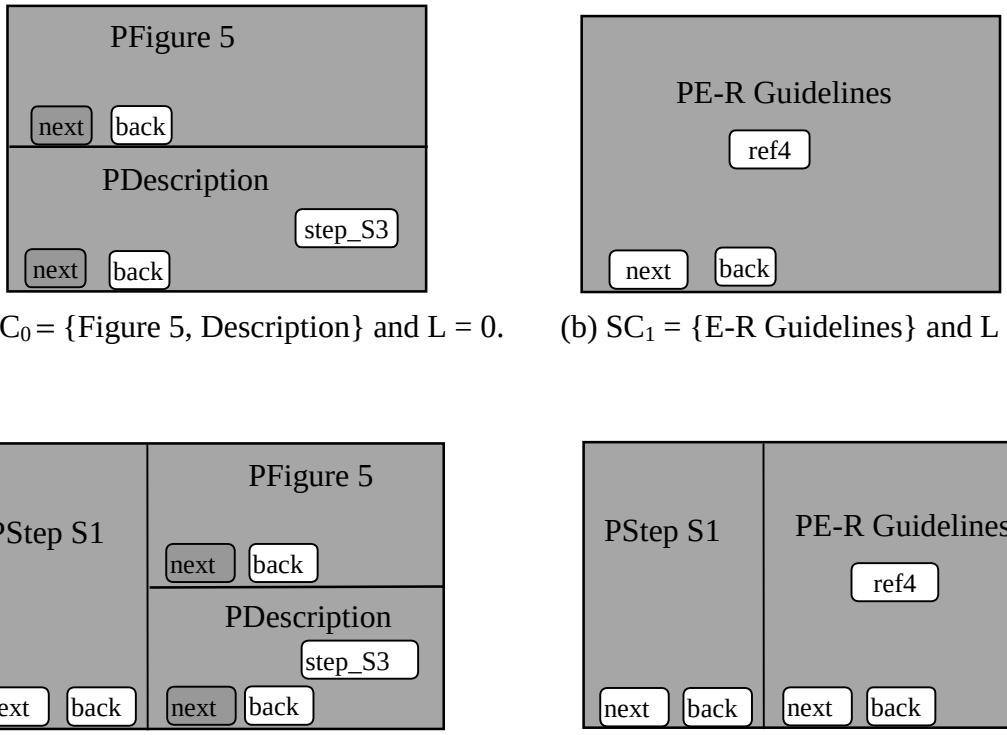
As an example application we are modeling a scientific paper by Isakowitz et al. [10], for which a statechart model is shown in Figure 1. This figure depicts the main states with a one-level decomposition only. Figure 2 shows the expanded statechart model for a specific section of the paper intitled "The Relationship Management Methodology (RMM)", abstracted by the state "Section_RMM" shown in both figures. In Figure 2 the default AND state "Introduction" has two sub-states: one is associated to a page that shows a figure containing a schema of the main steps of the RMM methodology (state "Figure 4"); the other is associated to a page containing a textual overview of the methodology (state "Description"). The state "Steps" is associated to an index page that has links to nodes containing descriptions of the seven steps of the methodology. Each such node is associated to a corresponding state shown in the figure. Note that not all application links are being explicitly represented in the given statechart.

Assuming an initial basic state configuration SC_0 = {Figure 5, Description} for the statechart model shown in Figure 2, let us illustrate possible state configurations to be reached during browsing, and possible layouts for the associated display windows, according to the interpretation given to the statechart semantics [9]. Note that states "Paper", "Section_RMM", "Step S1" and "Example" are also active, and were not explicitly included in the state configuration only because they are not basic states.

If a visibility level L = 0 is specified, a schematic layout for the window displaying the hyperdocument at this stage in browsing is presented in Figure 3(a). The pages associated to the two basic states "Figure 5" and "Description" in the current state configuration are displayed, and the accessible anchors defined for each node being displayed are also shown. For example, clicking at the anchor labeled *next* in page *PDescription* corresponds to activating the event labeled "next" in the underlying statechart, leading to a new state configuration SC_1 = {E-R Guidelines} (thus "Paper", "Section_RMM" and "Step S1" are also active states) and a new layout for the display, which is ilustrated in Figure 3(b). As L = 0, only the pages associated to the one basic state in the current state configuration are displayed, and only the anchors associated to transitions enabled for such state are visible.

If L = 1 and the current state configuration is given by SC_2 = {Figure 5, Description, Step S1}, a possible layout for the display window is shown in Figure 3(c). In this situation, the visibility level is set in order to show not only the pages associated to the basic states in the

current configuration, but also the pages associated to their parent states. Thus, the hyperdocument hierarchical node structure up to a depth of one is displayed, as well as the anchors in each page which correspond to enabled transitions in the underlying statechart. Both anchors *next* shown in the pages *PFigure 5* and *PDescription* are associated to event "next" leaving the state "Example". Selecting this anchor results in leaving this state (and its sub-states), and produces SC_3 = {E-R Guidelines, Step S1} as the new statechart configuration , for which a possible display window is depicted in Figure 3(d).



(a) SC_0 = {Figure 5, Description} and L = 0.

(b) SC_1 = {E-R Guidelines} and L = 0.

(c) SC_2 = {Figure 5, Description, Step S1} and L = 1.

(d) SC_3 = {E-R Guidelines, Step S1} and L = 1.

Figure 3. Possible window layouts displayed during the browsing of the hyperdocument for the given configurations.

2.4. Hierarchical Views

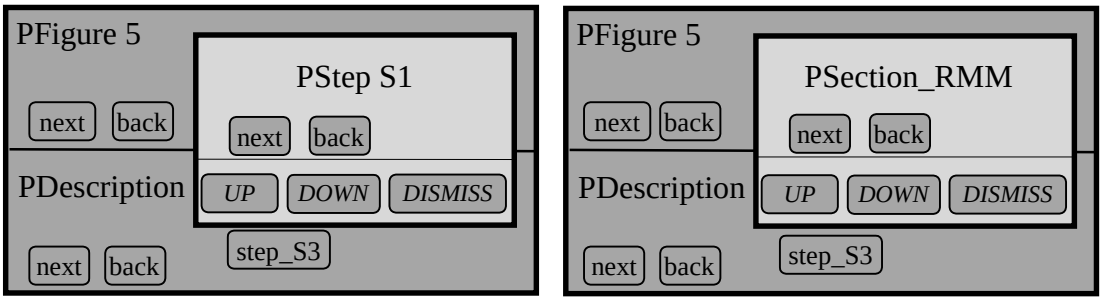
The visual representation of the underlying statechart model clearly describes the hierarchy of the hyperdocument and may be explored in order to minimize disorientation, a potential disadvantage commonly associated to hyperdocument reading (getting "lost in the hyperspace" [4]). The visual representation of the statechart may be presented during navigation as map to orient readers, helping them to find their way within the hyperdocument and also to construct a coherent mental representation of the organization of the hyperdocument. Such a mental representation is deemed essential for reducing the sense of disorientation and for increasing the readers´ability to actually understand the hyperdocument [14,20]. The statechart may also be presented as a clickable map, providing an alternative approach for navigation.

To better explore the hierarchical organization of the statecharts, at the beginning of a browsing session we may set up an auxiliary variable h = L+1, where L is the browsing level.

We may then define an operation *Show-hview* as a hierarchical view function that shows the pages associated to the states immediately above those in level L. Formally, *Show-hview*:

If $h \leq d$ *then*
display all pages *p* in set D_h, $D_h = \{p \mid x \in S_S \wedge M(x) = p \wedge \rho^h(x) \cap SC \neq \phi\}$

Considering the previous example (Figure 3(a)), if the user executes the operation *Show_hview* the hypertext system might produce the screen depicted in Figure 4(a), where the hierarchical view window (*Show_hview*) partially overlaps the main window. The screen generated by *Show_hview* enables three additional operations: UP, DOWN and DISMISS. The only action associated to the latter is the removal of the *Show_hview* window from the display. The *UP* and *DOWN* operations alter the current value of *h*, thus changing the pages displayed as a result of the hierarchical view operation. The execution of *Show_hview* does not modify the current statechart configuration, it only provides access to the nodes which are one level up (or down) in the hierarchy, as long as at least one such a node exists. Figure 4(b) shows the result of activating the UP button in the window of Figure 4(a).



(a) (b)

Figure 4. Activating *Show_hview* during the browsing of the hyperdocument.

3. SOLUTIONS USING THE MODEL

The HMBS model lends itself to providing simple solutions to some common problems related to hyperdocument visualization, browsing and analysis. Alternative solutions based on the statechart model for some of the problems discussed in [18] are shortly presented in the following. A more complete discussion in available in [15].

- *Synchronization of Simultaneous Display*: Statecharts provide a computational model adequate for expressing concurrency. Therefore, they have suitable mechanisms for representing concurrent display of multiple pages and concurrent browsing paths in a hyperdocument, as well as for providing authors the support for specifying synchronization of concurrent activities.

- *Access Control*: The set of Boolean variables associated to an HMBS model may be used to provide an effective mechanism to enforce browsing restrictions on readers of a hyperdocument. The model ensures that access capabilities are an integral part of the document, and allows that different browsing paths share one or more common nodes but still have interspersed mutually exclusive sections, overcoming the limitations of annotated directed graphs.

- *Tailored Versions*: A single statechart model may be used to keep a collection of distinct tailored versions of the same hyperdocument. This may be achieved by delivering the same hyperdocument model with alternative mapping functions M and sets of pages P, thus

allowing the association of different pages to the nodes in set S_S. Such a solution assumes that only the contents of the nodes are altered in the different versions, while the underlying hyperdocument structure is maintained. This is the case, for example, when different versions of the same hypertext in different languages are to be delivered.

- *Direct Access to Pages*: An index providing direct access to pages may be constructed from the available page titles, and readers may invoke this index in order to perform searches (string search, linear search, etc.). Once a page *P* is chosen which corresponds to a node *N*, it is shown in conjunction to the pages corresponding to nodes that with N define a valid set of active states which are chosen by default.

- *Node Reachability and Display Characteristics*: reachability of nodes of a hyperdocument can be verified using the underlying statechart, thus providing authors with facilities for completeness checking. This may be achieved through the construction of a reachability tree of the statechart that generates every valid state configuration [13]. Documents which grow without careful organization, like Web pages, can be analyzed by such a tool and have its anomalies detected [5]. The model also allows the determination of some parameters relevant for the physical presentation of the hyperdocument like the maximum number of windows that will be required simultaneously for any browsing session.

4. PROTOTYPE SYSTEM

We are currently working on the construction of a prototype hypertext system in order to experiment with the ideas presented in this paper. This prototype includes an authoring and a browsing modes, in addition to a statechart simulation environment which integrates a graphical editor for describing and simulating statecharts [11].

Figure 5 illustrates the use of the authoring mode. The author uses the graphical statechart editor to define the underlying hyperdocument statechart structure, whose root state is “Section_RMM”, as shown in the figure. This is the same statechart given in Figure 2. The “Create Page” window allows the creation of a page through the specification of its Contents (either provided in a file, or by entering the *Edit* option), of its Reader type (text in this case) and of its Title. After this information is entered, the system will ask for the association of this page with a statechart node, thus establishing the mapping function (M) for such state. The Figure also illustrates a querying operation that allows the author to see information on a given node, such as the type (AND,OR) and substates of its associated state, the Reader type, Contents, etc.

The browsing mode is illustrated in Figure 6. The figure depicts an implementation of the schematic layout shown in Figure 3(a). The example is using the same statechart, configuration and browsing level of Figure 3(a) (SC_0 = {Figure 5, Description}). The reader sees two separate windows, each corresponding to one of the two active basic states (Figure 5, Description) in the current statechart configuration. Anchors are marked as bold text.

Figure 7 illustrates the same statechart and configuration used in Figure 3(c), with a browsing level L = 1. In addition to the pages associated to basic states in the current state configuration, displayed in two separate windows, the reader also sees the page corresponding to their parent state. Three windows are shown in the overall. Also in that figure, the statechart structure is shown being simulated and controlling the presentation. This corresponds to the authoring mode, which allows the author to “play” the statechart and see the effect on the

presentation, thus being able to verify his/her design and make any desired changes. The active states are shown with dotted lines, and the anchors are shown in bold typeface.

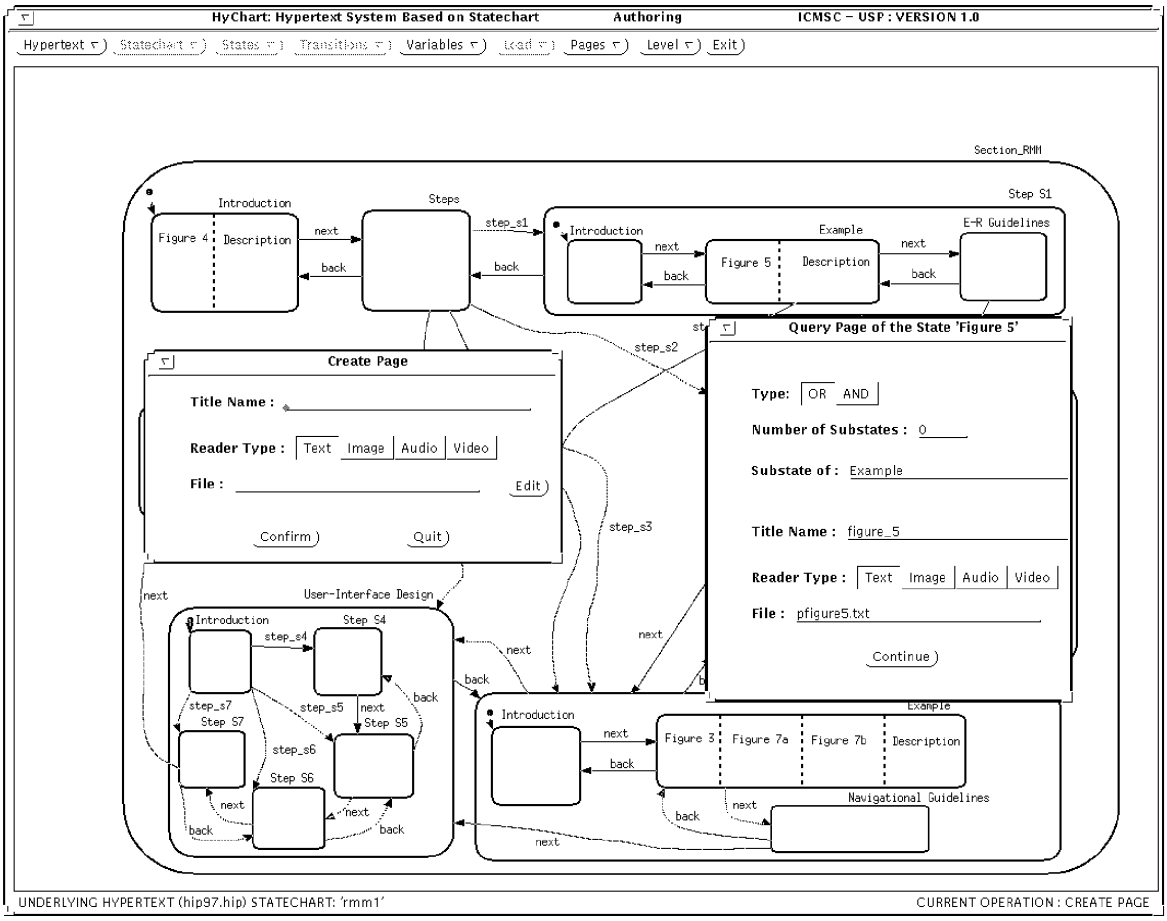


Figure 5. Authoring mode of the prototype hypertext system.

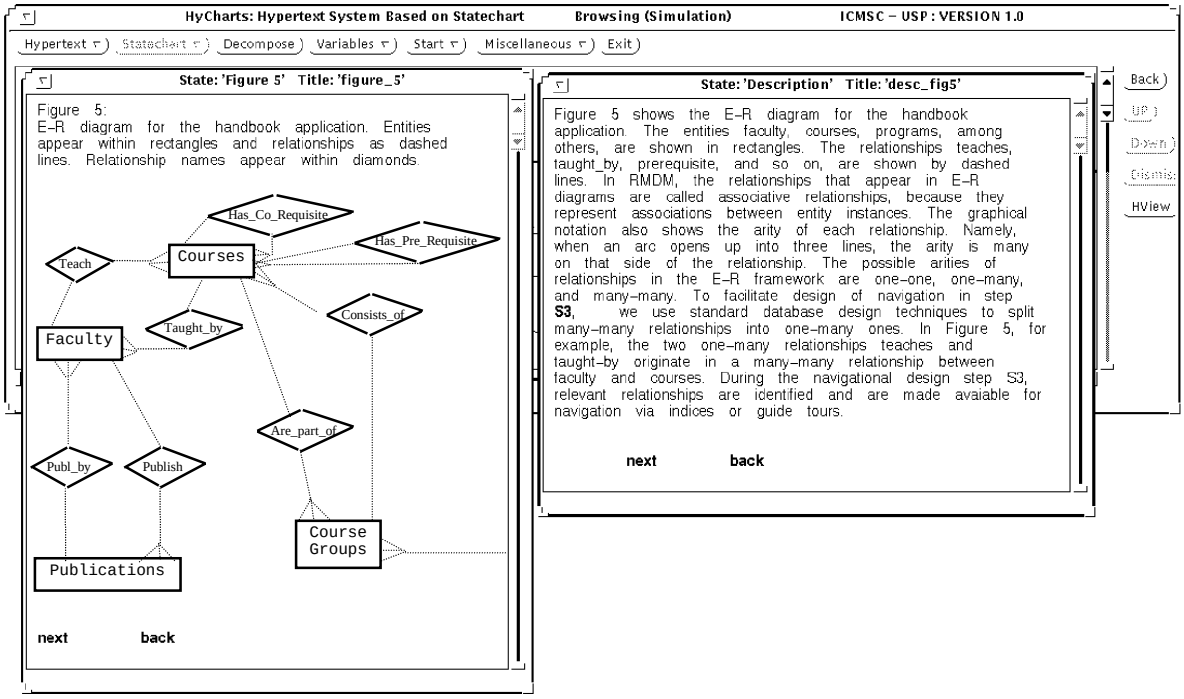


Figure 6. Browsing mode when SC_0 = {Figure 5, Description} and L = 0. Anchors are marked as bold text.

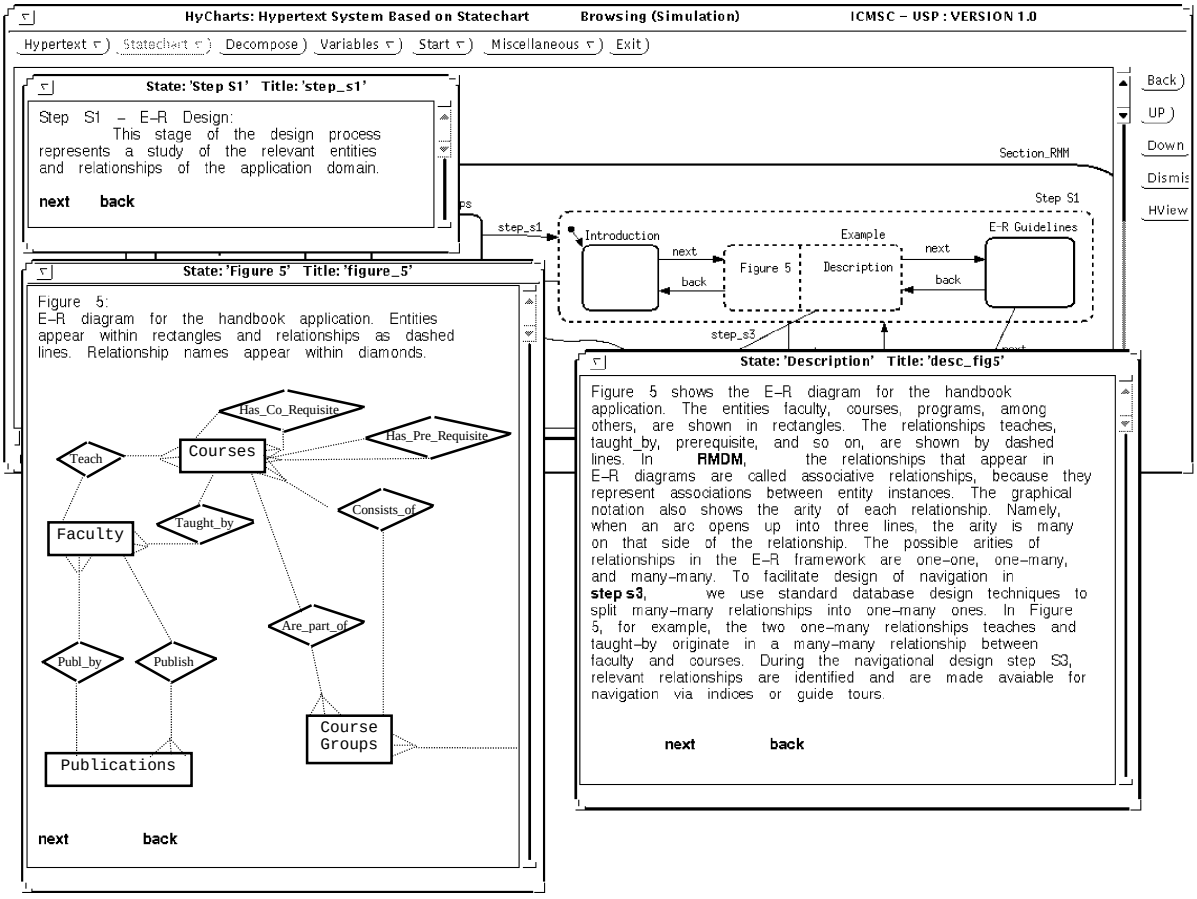


Figure 7. Browsing mode when SC_2 = {Figure 5, Description, Step S1} and L = 1. Anchors are marked as bold text.

5. RELATED WORK

The model implicit in many hypertext systems is the *labeled directed graph*. Although extremely simple, it does not represent the structure of the data and the browsing semantics adequately, and provides no separation between the structural organization of the nodes and their contents. Tompa [21] proposes a *hypergraph* formalism to model generic hyperdocument structures that enables formal identification of commonalities in these structures (nodes, links, labels) and direct reference to "groups of nodes" having a common link semantics (*hyperedges).* This model provides a natural representation for the overall structure of a document and for decomposing it in fragments, but its use of token marking to model the browsing semantics is not satisfactory. This is mainly because the approach of token marking is too general, as there are no restrictions on the marking process and the hypergraphs behave as general finite state machines. Moreover, the graphical representation associated to hypergraphs does not clarify their dynamic behaviour.

Stotts and Furuta [18,19] proposed the Trellis model, based on *Petri nets*. This model uses the Petri net structure and execution semantics not only to represent the structure but also to specify the browsing semantics of a hypertext document. However, a major shortcoming of the Petri net based model is the lack of satisfactory hierarchical structuring facilities, which difficulties the task of specifying synchronization control across different levels of hierarchical structures.

Zheng and Pong [23] also use *statecharts* for hypertext modeling. They claim to "use statecharts ... to specify the structure and browsing semantics of hypertext documents", but our view is that they are actually concerned with the specification of hypertext systems interfaces, rather than with the modeling of the hyperdocument structure, contents and browsing semantics. They model the interface of a specific hypertext system, and then show how a hyperdocument within the system may be presented to the reader. Their approach is a valid one, compatible with the traditional use of statecharts to model reactive systems, such as user interfaces. In this paper we suggest the use of statecharts in a different context, to model the structural organization and the browsing semantics associated to hyperdocuments. HMBS is not influenced by any implementation issues related to a specific hypertext system implementation, either at the interface or any other system level.

Unlike HDM [6], OOHDM [17] and RMM [10], HMBS is not a complete hyperdocument design model in the sense that it does not address the overall design process of a hyperdocument. It rather provides a systematic approach for navigation in a formally structured document such as to explore the hierarchical information and provide structure-oriented information. As such, it is not in competition with any of the current hyperdocument design approaches, but may provide a complement that should allow a more effective use of them. Both OOHDM and RMM propose a four step design approach comprising conceptual, navigational and interface modeling followed by an implementation stage. We are currently working on a complete design model that uses HMBS as its basic model for the navigational modeling stage. In fact, a somewhat related use of statecharts is actually suggested by Masiero et al. [12], who propose a method for analyzing office applications which relies on a model based on statecharts to record the flow of documents within the system. Each type of document is associated with a statechart, and that enables the automatic update of the links maintained in a hypertext database when new documents are added to the database.

6. INCLUDING ADDITIONAL STATECHART FEATURES INTO HMBS

Several extensions of statecharts which were not considered in this article could be further explored in the context of hyperdocument modeling. The *history mechanism* of the

statecharts, for example, ensures that, if an OR state is entered, control is returned to the last active state, rather than to the default state. This facility could be explored for modeling common situations in hyperdocument navigation which require the return to a previously visited node, like following a link to a node with reference information, and then returning to the specific node in which the link was triggered. On-line help manuals for instance could benefit from the history mechanism, which would ensure the user is always returned to the topic previously consulted.

If *integer variables* and *actions* are included in the model, and altering the values of variables during the browsing sessions is allowed, other interesting hypertext applications could be considered, such as programmed learning due to execution of actions associated to the transitions. A system could select the most adequate paths for a given user considering his/her knowledge about the subject, which might be inferred on the basis of a history of user choices made during one or more browsing sessions, or, alternatively, based on a series of grades assigned to the user.

The broadcast mechanism could be explored to provide proper synchronization support for other types of applications, e.g. multimedia systems. This is due to the execution semantic of actions that generate events which may be captured by orthogonal components. As an example, a video that ends before its soundtrack may broadcast a signal to stop the sound and move on to other parts of the presentation [16].

7. CONCLUSIONS

The HMBS model is particularly suitable for modeling hyperdocuments that present a hierarchical structure, such as books, scientific papers, on-line manuals and reports, as the state hierarchy imposed on the underlying statechart model provides adequate mechanisms for modeling such hierarchical organization and the possible browsing actions to be taken [22]. It uses the structure and execution semantics of the statecharts visual formalism to specify both the structural organization and the browsing semantics of a hyperdocument. An overhead in using an HMBS-based system is that the author interacting with such a system must be familiar with the model and its underlying formalism.

Statecharts are a well-known extension to conventional state-transition diagrams based on finite state machines whose outstanding features are hierarchy of states, ability to specify parallelism and a communication mechanism via broadcasting. They also feature a rich set of special notations for augmenting the notational power of state-transition diagrams, thus allowing the specification of complex problems in very concise diagrams. HMBS models are not restrict to hypertext systems in which a single element is visible at any time during browsing (as it is the case in directed graphs); rather, the model allows simultaneous visualization of multiple windows and provides proper support for concurrency and syncronization through the use of AND-states for modeling synchronized browsing actions.

HMBS is an evolving model and will modify according to emerging needs. A greater amount of experimentation with a large number of applications is required. In addition to using HMBS to model real applications and advancing the implementation of the system described in Section 4, we are now looking closely at the problem of extending HMBS to make it a full hypermedia model, thus incorporating the treatment of other media such as animation, soundtracks and videoclips [16].

ACKNOWLEDGMENTS

This work is supported by CNPq and FAPESP, Brazil. The authors are grateful to the anonymous reviewers and also to David Stotts for the many helpful comments.

REFERENCES

1. Akscyn, R.M., McCraken, D.L.; Yoder, E.A. "KMS: a distributed hypermedia system for managing knowledge organizations", *Communications of the ACM* 31, 7, p.820-835, July 1988.
2. Bieber, M.; Isakowitz, T. (Eds.) "Designing hypermedia applications", *Communications of the ACM* 38, 8, p.26-29, August 1995.
3. Brown, P.J. "Turning ideas into products: the Guide system", In*: Proceedings of the ACM Hypertext´87*, Chapel Hill, N.C., USA, p.33-40.
4. Conklin, J. "Hypertext: an introduction and survey", *IEEE Computer* 20, 9, p.17-41, September 1987.
5. Fortes, R.P.M., Garcia Neto, A.; Nicoletti, M.C. "A Formal approach to consistency and reuse of links in hypermedia applications", In: *ACM CHI´96 Workshop on Formal Methods in Human Computer Interaction: Comparison, Benefits, Open Questions*, Vancouver, Canada, p.33-36, April 14-15 1996. (electronic proceedings available at http://www.cenatls.cena.dgac.fr/~palanque/wschi96.html)
6. Garzotto, F., Paolini, P.; Schwabe, D. "HDM — a model-based approach to hypertext application design", *ACM Transactions on Information Systems* 11, 1, p.1-26, January 1993.
7. Halasz, F.; Schwartz, M. "The Dexter hypertext reference model", *Communications of the ACM* 37, 2, p.51-62, February 1994.
8. Harel, D. "Statecharts: a visual formalism for complex systems", *Science of Computer Programming* 8, p.231-274, 1987.
9. Harel, D. "On the formal semantics of statecharts", In: *Proceedings of the 2nd IEEE Symposium on Logic in Computer Science*, Ithaca, New York, p.54-64, 1987.
10. Isakowitz, T., Stohr, E.A.; Balasubramanian, P. "RMM: a methodology for structured hypermedia design", *Communications of the ACM* 38, 8, p.34-44, August 1995.
11. Masiero, P.C., Fortes, R.P. de M.; Batista Neto, J. do E. S. "Editing and simulating the behavioral aspects of real-time systems", In: *Proceedings of the XVIII Integrated Hardware and Software Seminar*, SEMISH, August 5-9, p.45-61, 1991, (in Portuguese).
12. Masiero, P.C., Oliveira, M.C.F., Germano, F.S.R.; Santos, G.P.B. "Authoring and searching in dynamically growing hypertext databases", *Hypermedia Journal*, Taylor Grahan, 6, 2, p.124-148, 1994.
13. Masiero, P.C., Maldonado, J.C.; Boaventura, I.G. "A reachability tree for statecharts and analysis of some properties", *Information and Software Technology* 36, 10, p.615-624, 1994.
14. Nanard, J.; Nanard, M. "Hypertext design environments and the hypertext design process", *Communications of the ACM* 38, 8, p.49-56, August 1995.
15. Oliveira, M.C.F., Turine, M.A.S.; Masiero, P.C. "A statechart-based model for hypertext", ICMSC Technical Report n. 19, 26p., ICMSC-USP, October 1996.
16. Paulo, F. B., Masiero, P.C.; Oliveira, M.C.F. "Extensions to HMBS for the specification of hypermedia presentations", In: *Proceedings of the X SBES - Brazilian Symposium on Software Engineering*, October 14-18, 1996, São Carlos, SP, Brazil, p.241-258, (in Portuguese).
17. Schwabe, D., Rossi, G.; Barbosa, S.D.J. "Systematic hypermedia application design with OOHDM", In: *Proceedings of the ACM Hypertext'96*, Washington, USA, p.116-128, March 1996.

18. Stotts, P.D.; Furuta, R. “Petri-net-based hypertext: document strucuture with browsing semantics”, *ACM Transactions on Information Systems* 7, 1, p.3-29, January 1989.
19. Stotts, P.D. ; Furuta, R. “Dynamic adaptation of hypertext structure”, In: *Proceedings of the ACM Hypertext´91*, New York, p.219-230, December 1991.
20. Thüring, M., Haake, J.M.; Hannemann, J. “What’s ELIZA doing in the Chinese room? Incoherent hyperdocuments - and how to avoid them. In: *Proceedings of the ACM Hypertext´91*, New York, p.161-177, December 1991.
21. Tompa, F.W. “A data model for flexible hypertext database systems”, *ACM Transactions on Information Systems* 7, 1, p.85-100, January 1989.
22. Turine, M.A.S., Oliveira, M.C.F.; Masiero, P.C. “Designing hyperdocuments with HMBS”, ICMSC Technical Report, ICMSC-USP, 1996 (in preparation).
23. Zheng, Y.; Pong, M. “Using statecharts to model hypertext”, In: *Proceedings of the European Conference on Hypertext Technology*, Milano, ACM Press, p.242-50, 1992.