HySCharts: A Statechart-Based Environment for Hyperdocument Authoring and Browsing

MARCELO AUGUSTO SANTOS TURINE mast@icmc.sc.usp.br
Instituto de Física de São Carlos, University of São Paulo, CP 369, 13560-970, São Carlos, SP, Brazil

MARIA CRISTINA FERREIRA DE OLIVEIRA cristina@icmc.sc.usp.br
PAULO CESAR MASIERO masiero@icmc.sc.usp.br
Instituto de Ciências Matemáticas e de Computação, University of São Paulo, CP 668, 13560-970, São Carlos, SP, Brazil

Abstract. The *HySCharts* environment (**Hy**perdocument **S**ystem Based on State**Charts**) supports the formal specification of hyperdocuments using a novel formalism called *HMBS* (**H**yperdocument **M**odel **B**ased on **S**tatecharts). This paper presents the HySCharts system architecture, with emphasis on its underlying model and on the functionality of its authoring and browsing modules. HMBS is a statechart-based, navigation-oriented model for hyperdocument specification that 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 HMBS and its associated browsing semantics are introduced. A discussion of the system and its capabilities, as supported by the model, is also provided.

Keywords: hypermedia system, hypermedia model, statecharts, HMBS, hierarchical views, HySCharts.

1. Introduction

As software products, hypermedia applications and hyperdocuments can greatly benefit from models and methods capable of organizing their overall design and construction process. Conventional software development methods and techniques are not directly applicable to hypermedia application design, which differs from traditional software development processes in several features. One can mention, for instance, the needs of managing large and complex pieces of information and of combining user controlled browsing with the intrinsic nature of multimedia information [4, 2]. Moreover, the hypermedia design process requires from a development team the ability to capture and organize the structure of a complex domain and make it clear and accessible to users. Hence, systematic approaches must be established for defining the organizational structure and browsing semantics of the application. Also, it is not uncommon that a single specification must be used to generate hyperdocument versions for multiple platforms.

Such problems strongly encourage the search for models and methods capable of providing guidelines for structured hyperdocument design and introducing some level of discipline in the authoring activity. The use of suitable models not only can improve the hyperdocuments' development and design process [1, 9] but also simplify their analysis and evaluation [3].

In this paper we adopt the Statecharts formalism as a supporting technique for hyperdocument specification. A model called *HMBS – Hyperdocument Model Based on Statecharts* [8, 13] is used to minimize some of the problems associated to hyperdocument development. Similarly to the Trellis model [11], HMBS allows an effective separation between the description of the hyperdocuments' organizational and navigational structure and its mapping into specific physical representations. Therefore, a single specification may be used to generate different presentations of the same hyperdocument. The model provides solutions to problems related to hyperdocument specification by exploiting useful statechart features. For example, orthogonal states are used to model concurrent presentation of information, and conditional expressions may be employed to control navigation. HySCharts – the ***Hy****perdocument* ***S****ystem based on State****Charts*** - has been developed as a system that adopts HMBS as its underlying model for hyperdocument authoring and browsing, thus supporting the automated creation and validation of HMBS specifications.

This paper describes the HySCharts environment, and it is organized as follows. In Section 2 the syntax of HMBS is described, as well as the main features of the model regarding its use for specifying the structural organization and the browsing semantics of hyperdocuments. Section 3 presents the HySCHarts system architecture, emphasizing the authoring and the browsing modules and their functionality. In Section 4 the hierarchical view operation supported by HMBS and implemented in HySCharts is introduced and its use is illustrated. Section 5 contains a discussion on the generation of Web applications from HySCharts specifications. Some final remarks and conclusions are presented in Section 6.

2. A Statechart-Based Model

2.1. Definition

HMBS uses the statechart structure and operational semantics to specify both the organizational structure and the browsing semantics of large and complex hyperdocuments [8, 13]. Statecharts extend 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 [5,6]. HMBS describes a hyperdocument in terms of three classes of objects, namely structural, navigational and presentation objects. The organizational structure is defined by an underlying statechart comprised of states, transitions and events, which are the structural objects of the model. The hyperdocument navigational structure is given by pages, links (although these are not explicitly represented in the model) and anchors, which are the model's navigational objects. The presentation objects consist of channels invoked during browsing to interpret and exhibit information contained in pages. Thus, a set of statechart states is mapped into pages that correspond the hyperdocument nodes containing chunks of information. Statechart events are associated to anchors representing link sources. Anchor activation

results in the triggering of an associated event in the underlying statechart, thus firing the corresponding transition from a source state (mapped to the link's source page) to a destination state (mapped to the link's target page).

According to HMBS, a hyperdocument *H* is a 7-tuple H = <*ST*, *P*, *m*, L, *pl*, *ae*, N>, in which:

a) ST represents the hyperdocument underlying statechart structure. The statechart definition adopted is actually a simplification of the one presented by Harel [6], being defined by ST = <S, ρ, ψ, γ, δ, V, C, E, A, R, T> denoting, respectively: the set of states, the hierarchy function, the decomposition type function, the history function, the default function, the set of expressions, the set of conditions, the set of event expressions, the set of actions, the set of labels and, finally, the set of transitions [13, 14]. Additional information regarding the simplifications introduced in the statechart definition adopted in HMBS may be obtained in [13, 14];

b) P is a finite set of information pages that define the contents of the hyperdocument. Each page $p \in P$ is conceptually defined by the triple <c, t, Anc_p>, such that "c" is the information content expressed in any static (such as text, graphics or image) or dynamic (video, audio or animation) media; "t" represents a title that uniquely identifies the page; and "Anc_p" defines a collection of anchors contained in the page. The set P may also include a special *null page* with no associated contents, titles or anchors, which may be associated to states that do not specify the presentation of information;

c) m: $S_S \rightarrow P$ denotes a value function mapping states of set S_S ($S_S \subset S$) into hyperdocument pages $p \in P$. The subset "S_S" is defined by S_S: $\{x \in S \mid \psi(x)=OR \vee \rho(x) = \phi\}$, i.e., S_S is the subset of S comprising basic states and OR states. AND states are not mapped into pages, being used solely to specify concurrent presentation of information;

d) L defines the set of presentation channels, which are abstract devices used to interpret and exhibit the information contained in pages. Channels allow authors to specify spatial coordination parameters and control global presentation attributes. The use of channels in HMBS has been inspired in their use by the *Amsterdam Hypermedia Model* [4] and the *Nested Context Model* [10];

e) pl: $P \rightarrow L$ defines a visualization relationship that associates each hyperdocument page $p \in P$ with a single channel $l \in L$ that is able to interpret it;

f) ae: $Anc_p \rightarrow E$ defines a function that associates anchor elements "a_p" ($a_p \in Anc_p$) to statechart events that control the firing of transitions. A restriction imposed by HMBS is that an event in a transition label must be unique in order to have an associated anchor. Moreover, the page containing the anchor must be mapped to a state that is in the source set of the transition or, alternatively, it may be mapped to a sub-state of a state in this source set. Transitions and their corresponding event labels may be reused to define anchors in different navigational contexts; and

g) N ($N \geq 0$) defines the hyperdocument visibility level. The author may use the value of N to define the hierarchy depth when displaying pages during navigation, as described in the following section.

2.2. Browsing Semantics

The browsing semantics adopted in HMBS defines the pages to be presented during browsing, which anchors are enabled and which navigational transformations occur during reader interaction. In Section 4 it is pointed out how the navigational semantics defined provides mechanisms that highlight the hierarchical structure typical of a number of applications. The hyperdocument dynamic behavior is based on the statechart operational semantics from a given initial state configuration [5]. The basic navigational concept in HMBS is the visit to a page, and presentation of information contained in pages is determined solely by the reader's decision of following a link by activating an anchor.

During browsing channels are invoked to exhibit the hyperdocument contents by interpreting an arbitrary number of pages. Multiple pages may be concurrently exhibited, as determined by the current statechart configuration and the visibility level "N". For a hyperdocument with a visibility level "N" set to zero (N = 0), at any stage during a browsing section all the pages associated to the atomic states within the current active state configuration will be exhibited. If "N" were set to one (N = 1), the pages exhibited include the same ones defined for N = 0, plus those associated to the immediate OR ancestor states of the states in the current state configuration, and so on for greater values of N (N > 1). Thus, the set of pages presented defines a "hyperdocument context configuration" which comprise the pages associated to the basic states in the current statechart state configuration, in addition to those pages associated to non-basic states that satisfy the visibility level as specified by the author. The initial context configuration (CC_0) is defined by the statechart default state configuration. Formally, a context configuration is defined as follows:

If $N = 0$ *then* display all pages p in set CC_0, in which

$$CC_0 = \{p \mid x \in S_S \wedge M(x) = p \wedge \rho^0(x) \cap SC \neq \phi\}$$

If $N = 1$ *then* display all pages p in set $CC_0 \cup CC_1$, in which

$$CC_1 = \{p \mid x \in S_S \wedge m(x) = p \wedge \rho^1(x) \cap SC \neq \phi\}$$

Generalizing,

If $N = n$ *then* display all pages p in set CC_n, given by

$$CC_n = \bigcup_{i=0\ldots n} CC_i \text{, in which}$$

$$CC_i = \{p \mid x \in S_S \wedge M(x) = p \wedge \rho^i(x) \cap SC \neq \phi\}$$

Under the operational semantics of statecharts, the browsing semantics defined by HMBS is interpreted as follows. When a user activates an anchor within a page, the following actions must be taken by the system:

a) The event associated to the anchor (through mapping "ae") is generated, thus causing the triggering of the transition associated to that link;

b) All states in the target sets of the fired transitions are activated, thus generating the next state configuration and disabling the previous one. In sequence, the logical windows associated to

pages corresponding to states in the previous configuration and also to their ancestral states (according to the given level N) are removed from the display; and

c) The hyperdocument's new context configuration is generated, being composed by the pages associated to the states included in the new current state configuration and to those that satisfy the scope defined by the visibility level N. The presentation channels (as given by mapping function "pl") for each page in the new context configuration are invoked and these pages are exhibited.

3. The *HySCharts* Environment

The *HySCharts* environment has been designed to support the creation and interpreted execution of HMBS specifications. It has been developed as an extension to *StatSim* (**STA**Techart **SIM**ulator) [7], a system with a graphical interface that allows the automated definition and validation of conventional statechart software specifications. The HySCharts architecture consists of three main layers, namely the application, structuring and storage layers (Figure 1). The application layer comprises an authoring and a browsing modules, in addition to a module that implements statechart edition and simulation tools. The structuring layer is in the core of the system, as it defines the hyperdocuments' internal structure in terms of the HMBS +structural, navigation and presentation objects. These objects are stored in databases managed through functions defined in the storage layer.

3.1. The Authoring Module

As part of the authoring activity the author visually specifies the structural, navigational and presentation objects, and analyzes the navigational structure of the proposed hyperdocument. A graphical editor (GSE – Graphical Statechart Editor) is used to create the underlying statechart that defines the structure of the hyperdocument, as illustrated in Figure 2. The organizational structure presented is for a hyperdocument that introduces the Ecological Park of São Carlos (EPSC). This hyperdocument contains general information about the park and its animals, a description of activities maintained within the park, a historical background, a word from the director, and hints for the visitors. The major focus is on the animals, as the park keeps nearly 600 of them distributed amongst 73 different species. The hyperdocument provides na introduction to each specie, including a photo and a description of some of its popular and scientific characteristics [14]. The window entitled "*QUERY STATE: Animal*" superimposed on the main window provides structural information about the state "*Animal*". This is an OR state which is further decomposed into three sub-states at a different hierarchical level (shown in Figure 8).

Figure 3 illustrates the graphical editor for HMBS objects (GEHMBS), which provides tools for managing pages, anchors and channels. Three windows related to page objects are depicted, named "*Create Page*", "*Query Page of the State 'Identification_Text'*" and "*Pages Created*". The "*Create Page*" window allows the specification of a page by defining its attributes *title*, *presentation channel*, *media type* and *content*. The page content is provided by a file ("*File*") identified by its name. Each page must be associated to a single (author specified) presentation channel that can interpret it and exhibit its content. Basically, the channel defines the media type (text, image, audio or video) associated to the page and some visualization properties. Once such information is provided, the

author may associate the page to its corresponding structural object (state), thus creating an instance of the "m" mapping between states and pages.

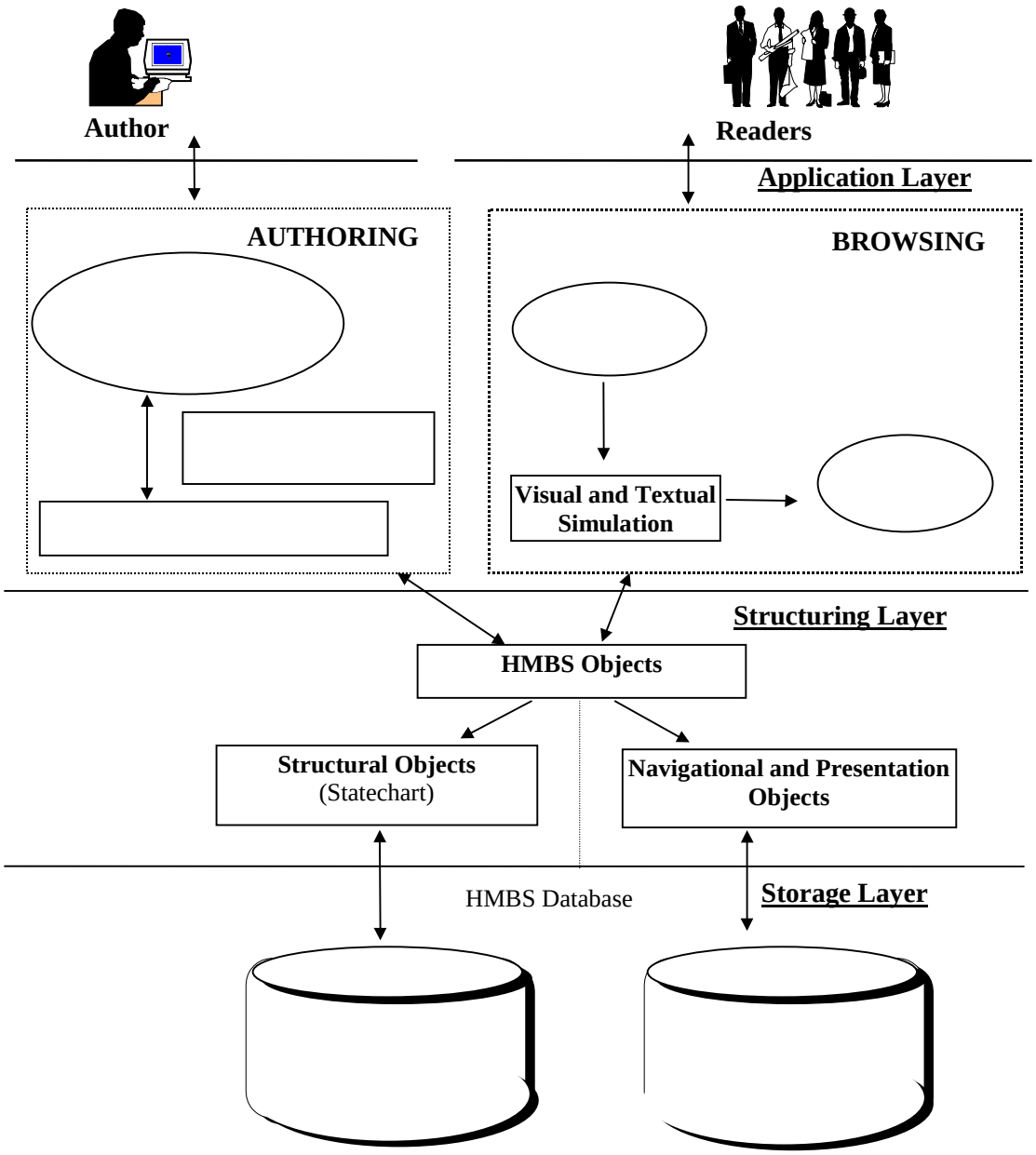

Figure 1. *HySCharts* architecture.

Figure 3 also illustrates the operation for querying pages. By selecting a state with an associated page, information related to both the page and the state is presented to the user. For instance, in the window "*Query Page of the State 'Identification_Tex't*" information relative to state "*Identication_Text*" (which is a sub-state of "*HomePage*") and its associated page is shown. The window "*Pages Created*" contains information (title, media type and name of the associated state) concerning all the pages created for the instantiated hyperdocument.

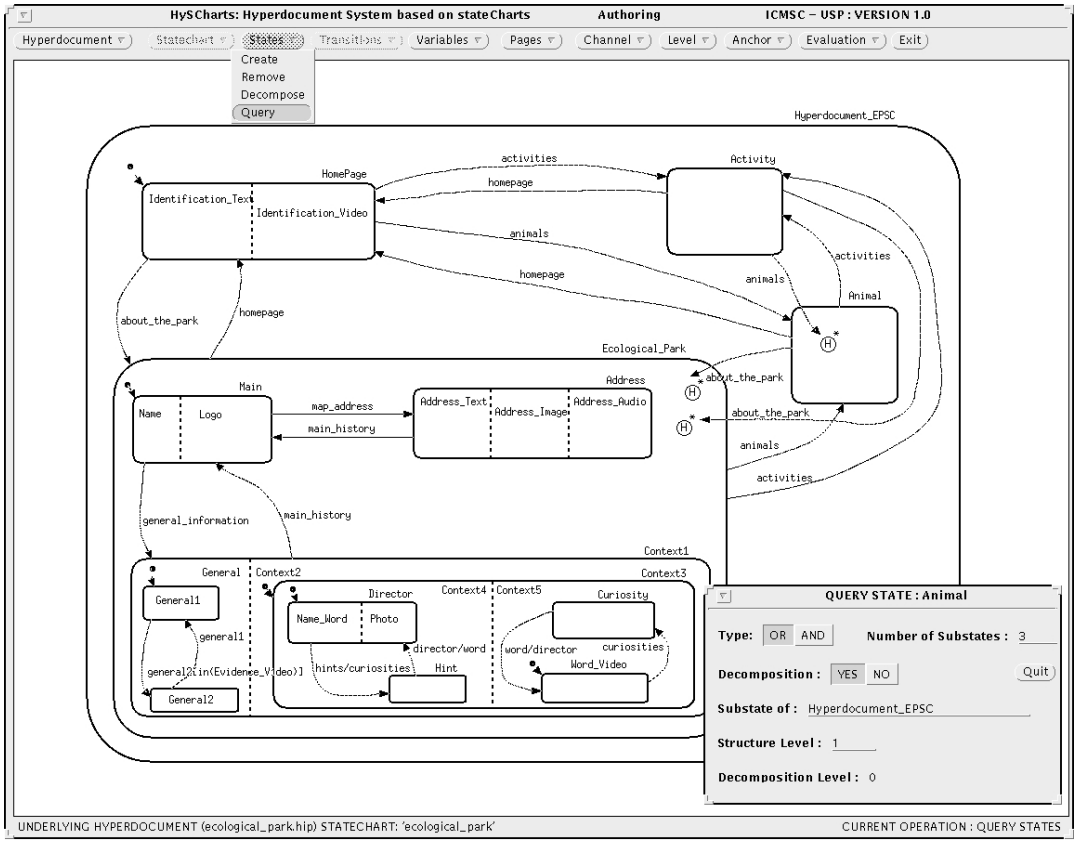

Figure 2. GSE of the authoring module of HySCharts.

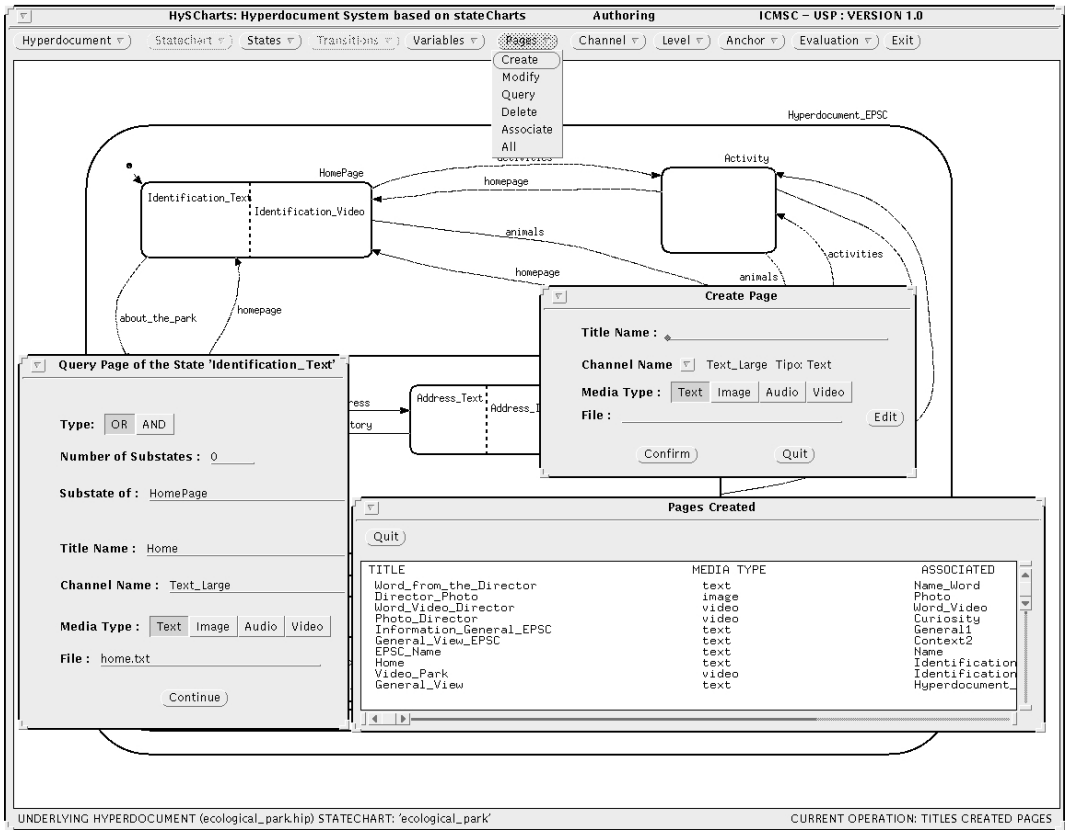

Figure 3. Windows in the module responsible for managing page objects.

Having created the pages and associated them to their corresponding states, the author may specify the anchors responsible for enabling the hyperdocument links. Anchors belonging to pages of type text are visually presented during navigation as text strings with underlined and bold typeface. In the remaining page types (consisting of image, video or audio medias) anchors are presented as buttons. In order to define an anchor object within a page the author chooses its identifying keyword and then selects its associated event, which must belong to a transition label in the statechart. In this way an instance of the "ae" relationship is created, and the navigational objects anchor and link are defined.

In Figure 4 the content of the text page "$P_{Identification\_Text}$", associated to state "*Identification_Text*", is being shown in the window named "*Viewer*". This figure also illustrates the window "*Query Anchor-> Event*", that depicts the result of a query operation on the anchor "ABOUT_THE_PARK" in page "$P_{Identification\_Text}$". Such anchor is associated to the event labeled "about_the_park", which appears in a transition originating from the state "HomePage" and going into the state "Ecological_Park".

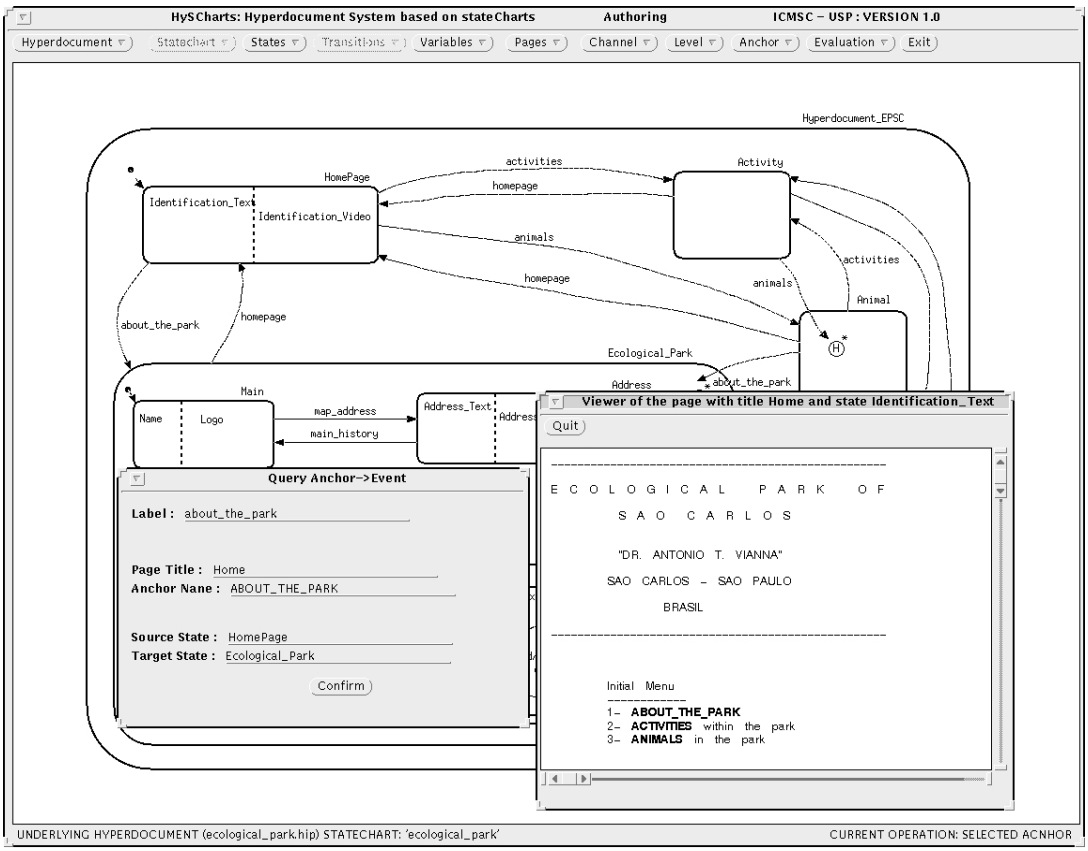

Figure 4. Windows for managing anchor objects.

The structuring layer also stores the channel objects. They support the definition of high-level presentation attributes for pages, such as font style and size, plus window background color in the case of text media channels, and spatial coordination attributes such as window position, height and width. HySCharts adopts an object-oriented approach for dealing with such presentation objects. Default channel classes are defined which may be instantiated, according to the authors' needs, using the mechanisms of inheritance and specialization/generalization. For instance, one may employ two

different channels for presenting text, one using a font "*Times New Roman*" and the other using a font "*Arial*". Analogously to the CMIfed system [15], the use of channels allows a single document to be presented in different manners by redefining its presentation attributes rather than its navigational structure.

The authoring module also provides tools for analysis and verification of the HMBS hyperdocument specifications (module "Properties and HMBS Objects Checker" in Figure 1). HMBS supports the analysis of some properties and allows the identification of structural inconsistencies and problems related to presentation and navigation. Property analysis is based on the generation of a reachability tree that provides subsidies for detecting specification anomalies. Some dynamic properties of statecharts have been defined and introduced in the context of hyperdocuments: page reachability from any given context configuration, reinitiability, deadlock during navigation, vivacity of navigational links and valid sequences of anchors [14].

3.2. Browsing Module

In order to make a hyperdocument available for readers, HMBS specifications must be interpreted and "executed" in the browsing module, according to the navigational semantics defined by the model. The *Browser* module of HySCharts (Figure 1) has a statechart machine that receives external events (generated by user actions) and interprets them according to the statechart semantics, thus generating a new statechart active configuration and a corresponding presentation.

Navigation may be initiated from an initial default configuration context "CC_0"; alternatively the reader may choose the title of a specific page of interest from an index that provides direct access to pages from its titles [8]. Considering the first approach, Figure 5 depicts the layout corresponding to a context configuration $CC_0=\{P_{Identification\_Text}, P_{Identification\_Video}\}$ of the hyperdocument specified in Figure 2, assuming N = 0, which corresponds to its default configuration. The reader sees two logical windows: a text one (intitled "*State: 'Identification_Text' Title: 'Home'*") and a video one (intitled "*State: 'Identification_Video'*") with its associated control window (which is always exhibited close to video windows). The "*Show_Hview*" window is also always presented during navigation and will be explained in Section 4. The text window associated to page "$P_{Identification\_Text}$" contains the anchors "ABOUT_THE_PARK", "ACTIVITIES" and "ANIMALS", identifiable by their underlined and bold font style.

If the anchor "ABOUT_THE_PARK" is activated, the link associated to the transition labeled by the event "about_the_park" in Figure 2 is fired, and the browser generates a new presentation layout associated to a new context configuration $CC_1 = \{P_{Name}, P_{Logo}\}$ (not shown). Activating the anchor "INFORMATION" in page "P_{Name}" (associated to event "general_information" in Figure 2), and assuming N = 1 (rather than N = 0), results in the context configuration $CC_2 = \{P_{General1}, P_{Name\_Word}, P_{Photo}, P_{Word\_Video}, P_{Context4}\}$, shown in Figure 6. This particular presentation of the hyperdocument includes the contents of five pages, four of which are directly associated to states of the statechart configuration given by SC_2={General1, Name_Word, Photo, Word_Video}. The page "$P_{Context4}$", represented by the logical window named "General_View_EPSC" at the top of Figure 6, corresponds to the immediate ancestor state of those states in "SC_2". Figure 6 also shows the graphical

representation of the navigation structure of the hyperdocument under simulation. In this way, the author can “execute” the hyperdocument specification and observe the effect in its graphical structure representation. This ability of interactively simulating a navigation session for a hyperdocument is useful to support an analysis of its behavior during navigation and, consequently, the identification of changes required in the specification. The availability of a graphical representation during browsing may also help to improve reader comprehension of the hyperdocument and to reduce disorientation.

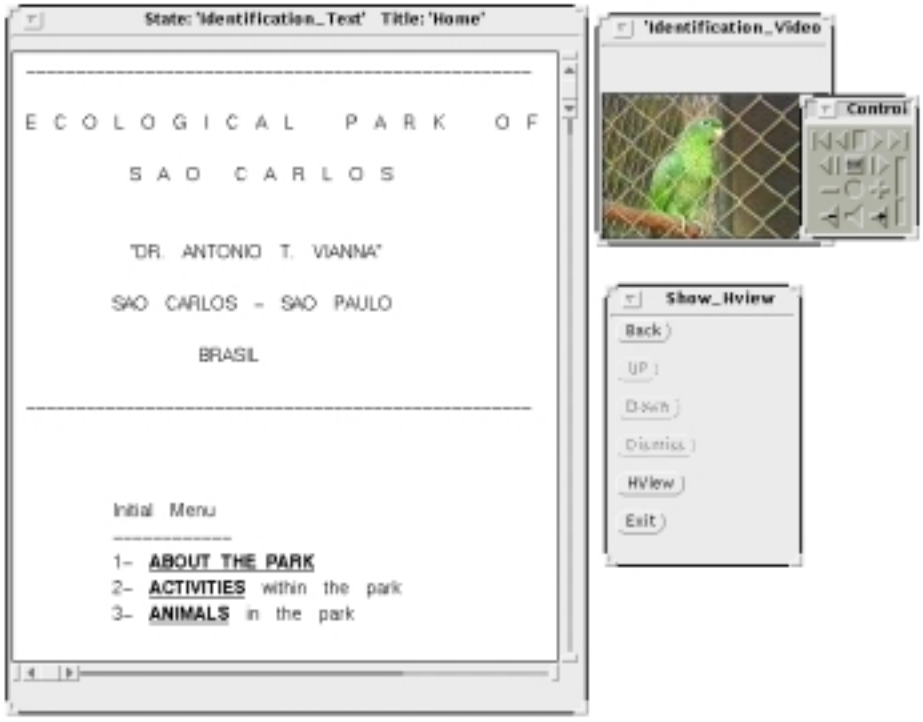

Figure 5. Browsing window of the hyperdocument described by the statechart in Figure 2, assuming N=0 and $CC_0 = \{P_{Identification\_Text}, P_{Identification\_Video}\}$.

An additional capability provided by HMBS is navigation control. Conditional expressions may be employed to ensure restricted access to parts of the hyperdocument; to present different classes of readers with alternative navigation paths; and also to manage different navigational contexts [9] in a straightforward manner. For example, suppose that in the EPSC application the author wants to provide alternative guided tours for the readers to browse information about the animals. One tour allows navigation through an arbitrary subset of the species available, whereas another one provides a tour through the animals that are in danger of extinction. This situation is depicted in Figure 7, which shows the modeling of the two guided tours using two logical control variables *cn1* and *cn2*, respectively, which are initially set to false. It may be noticed from observation of this figure that some states belong to both tours, implying that their associated pages may appear in both. This is the case of states named “Ema” and “Mico_Leao_Dourado”. Thus, if the reader chooses to follow the second guided tour the control variable *cn2* will be reset to true, and state “Ema” will be activated by the firing of transition enabled by the event “extinction_tour”. Consequently, those pages associated to states “Mico_Leao_Dourado” and “Onca_Pintada” are reachable, but not the one associated to state “Tucano”, for example, as this latter one does not belong to this tour. This behavior is ensured by the fact the transition “next” going into state “Tucano” is not enabled as long as its conditional expression

cn1 is false.

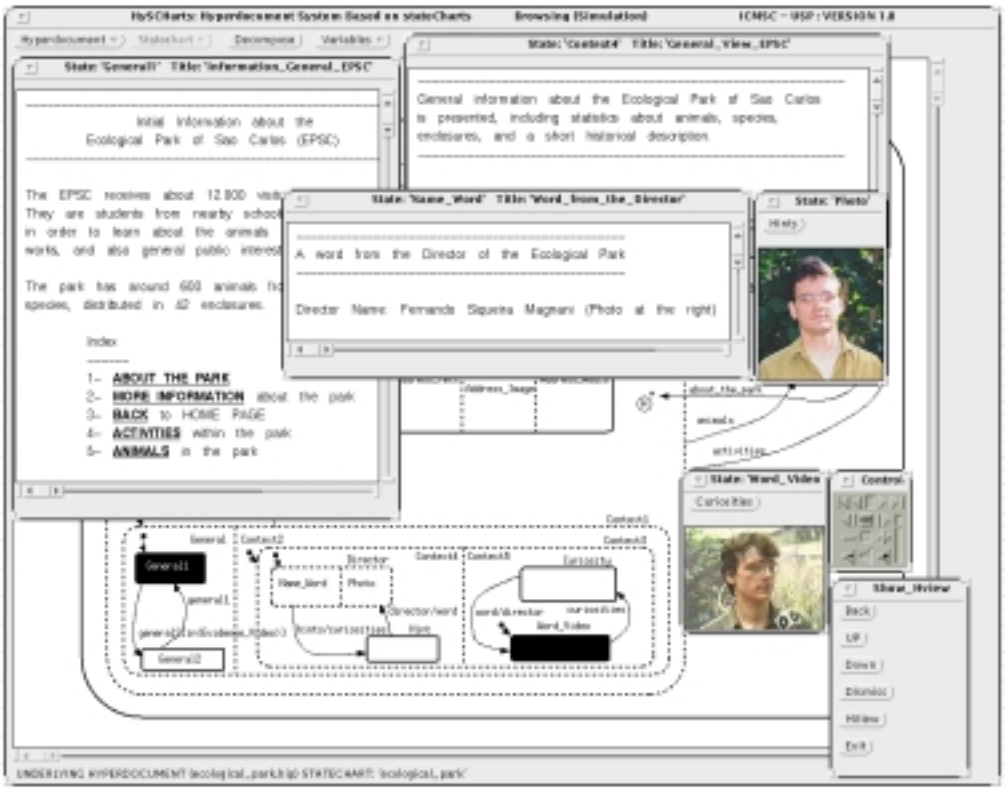

Figure 6. Hyperdocument presentation referring to $CC_2 = \{P_{General1}, P_{Name\_Word}, P_{Photo}, P_{Word\_Video}, P_{Context4}\}$ for the hyperdocument depicted in Figure 2; N = 1.

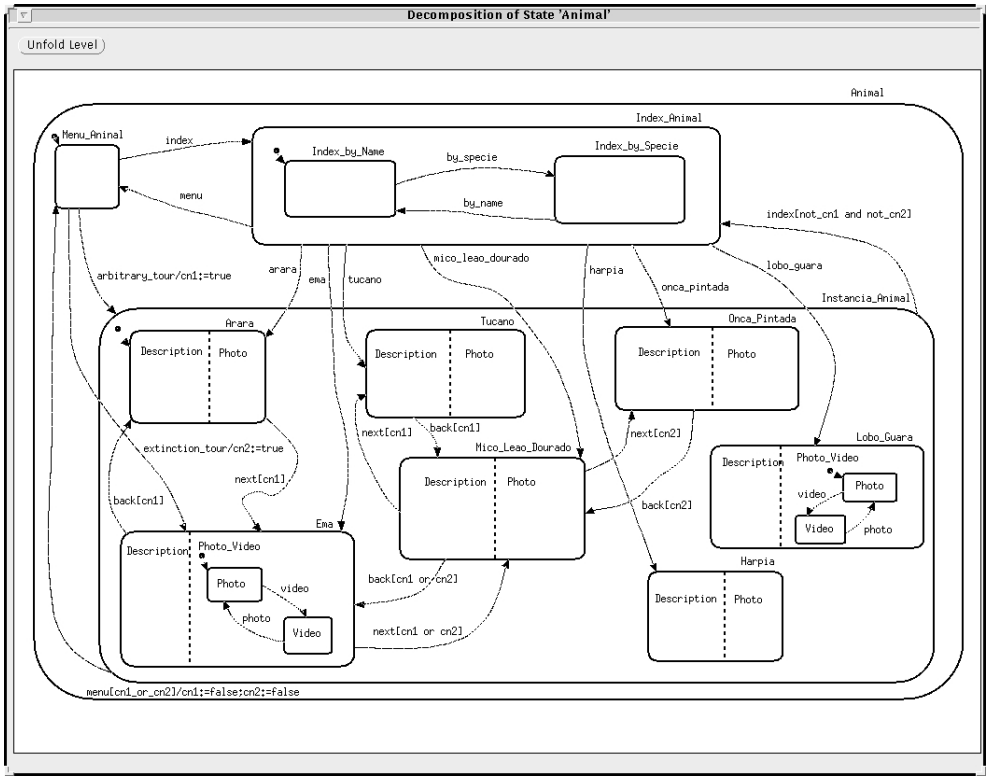

Figure 7. Modeling of navigational contexts using control variables (names of the species are given in Portuguese).

4. Hierarchical Views

According to Thüring [12] a visual representation of the hyperdocument structure may help the reader to identify topics of interest and to keep a track of his/her orientation. Furthermore, mechanisms to display and browse the document hierarchy in a structured manner are highly desirable. This type of "structured browsing" is strongly supported by HMBS and the HySCharts environment. In this section we present an operation for displaying hierarchical views of a document, called "Show-Hview" (from "show hierarchical view"), which provides a natural mechanism for exploiting a hyperdocument's hierarchy.

The HySCharts interface for the "Show-Hview" operation is provided by a special window which is permanently visible during navigation. At the beginning of a browsing session an auxiliary variable "h" is defined, with h = N + 1, in which N is the visibility level. If h ≤ d, where d is the depth of the and/or tree describing the statechart state hierarchy, then the system must show all pages *p* of the set CC_h such that $CC_h=\{p \mid x \in S_S \wedge m(x) = p \wedge \rho^h(x) \cap SC \neq \phi\}$.

Considering Figure 5 with N = 0, three new structural links "*Up*", "*Down*" e "*Dismiss*" are enabled when the reader activates the "HView" option in the "Show_Hview" window. The result is shown in Figure 8, which illustrates a presentation of the context configuration $CC = \{P_{Identification\_Text}, P_{Identification\_Video}, P_{Hyperdocument\_EPSC}\}$. The page "$P_{Hyperdocument\_EPSC}$" is associated to state "Hyperdocument_EPSC", which is the immediate ancestor of the states in the current configuration SC_0={Identification_Text, Identification_Video} (as shown in Figure 2). The "Dismiss" operation closes the "Show_Hview" window. Operations "Up" and "Down" change the value of "h" by going one level up and down, respectively, in the hierarchy levels:

"*Up*": If h < d, then h = h+1; activate "Show_Hview".

"*Down*": If h > N, then h = h-1; activate "Show_Hview".

The execution of the "*Show_Hview*" operation does not change the statecharts' current state configuration. It is just an alternative dynamic feature that allows access to information in pages associated to states hierarchically above the ones being presented, as fixed by the visibility level N.

5. HySCharts and the Web

A strong feature of HySCharts is that it may be used as a front-end to generate web applications. After specifying and interactively simulating an application, a designer may automatically generate an HTML version to be read by a standard web browser. The generator implemented supports two kinds of code transformation: one that preserves the browsing semantics of the hyperdocument as defined in HMBS and supported in HySCharts; and one that does not preserve the HMBS semantics. In this latter case the statechart is flattened and each page is shown in isolation with proper navigation links attached: up and down for hierarchical navigation; left and right for horizontal navigation as defined in the original specification of the application. Hence, statechart configurations are not used to derive presentation context configurations, as described in Section 4.

The first solution provides more interesting possibilities. Frames are employed to display the multiple pages within a context configuration derived from the current statechart configuration. Pages

associated to non-basic states within the current configuration may also be shown, according to the visibility level defined. Thus, before the HTML code is generated the designer must define the visibility level, which will be fixed for the whole presentation. Different presentations may be generated using different visibility levels.

Figure 8. Result of executing the "Show_*Hview*" operation, considering the presentation layout depicted in Figure 5.

6. Conclusions

This paper presented HySCharts, an environment for authoring and browsing hyperdocuments specified using the HMBS formalism. HMBS allows authors to specify the organizational and navigational structure of hyperdocuments, as well as their dynamic behavior, in a precise and semantically rigorous way, thus encouraging the development of structured hyperdocuments. HySCharts provides facilities for rapid prototyping and interactive simulation of a hyperdocument's underlying structure, as well as for some property analysis and verification, based on the underlying statechart structure.

HySCharts was developed mainly as a proof of concept for HMBS, in order to validate the model usage in a practical, real-world context. It can be used in a closed-environment mode and also as a front-end to generate web-based applications. Among its interesting features we emphasize the abilities to handle hierarchy and controlling navigation, which are of particular importance for a wide range of applications, including on-line help documents, books, articles, kiosks, and academic catalogues. Additionally, hyperdocuments may be analyzed for node reachability, deadlocks, and valid sequences of events.

The environment usage may be integrated into a general hyperdocument design method that adopts HMBS as its underlying navigational model. The combined use of HySCharts, HMBS and a design method is beneficial for both users and designers, as it may help reducing development times and improving application quality.

Acknowledgments

The authors wish to acknowledge the financial support of CNPq - The Brazilian National Research Funding Agency - and of FAPESP - The State of São Paulo Research Funding Agency.

References

1. U. Cavallaro, F. Garzotto, P. Paolini, and D. Totaro, “HIFI: hypertext interface for information systems”, IEEE Software, Vol. 10, No. 6, pp. 48-51, November 1993.
2. F. Garzotto, L. Mainetti, and P. Paolini, “Adding multimedia collections to the Dexter model”, in Proceedings of ACM ECHT’94, European Conference on Hypermedia Technologies, Edinburgh, Scotland, September 1994, pp.70-80.
3. F. Garzotto, L. Mainetti, and P. Paolini, “Hypermedia design, analysis, and evaluation issues”, Communications of the Association of Computer Machinery, Vol. 38, No. 8, pp. 74-86, August 1995.
4. L. Hardman, D.C.A Bulterman, and G. Van Rossum, “Links in hypermedia: the requirements for context”, in Proceedings of ACM Hypertext’93, Seattle, Washington USA, November 1993, pp.183-191.
5. D. Harel, “Statecharts: a visual formalism for complex systems”, Science of Computer Programming, Vol. 8, No. 3, pp. 231-274, July 1987.
6. D. Harel, A. Pnueli, J.P. Schmidt, and R. Sherman, “On the formal semantics of statecharts”, in Proceedings of II IEEE Symposium on Logic in Computer Science, 1987, pp.54-64.
7. P.C. Masiero, R.P.M., Fortes, and J.E.S. Batista Neto, “Editing and simulating behavioral aspects of real-time systems”, in Proceedings of XVIII Integrated Hardware and Software Seminar, SEMISH, Santos, Brazil, August 5-9 1991, pp. 45-61 (in Portuguese).
8. M.C.F. Oliveira, M.A.S. Turine, and P.C. Masiero, “An overview of HMBS: a statechart-based model for hypertext”, in Proceedings of. II Workshop on Hypermedia Systems, WOSH, Fortaleza, CE, Brazil, May 18-19 1996, pp. 11-20.
9. D. Schwabe, G. Rossi, and S.D.J. Barbosa, “Systematic hypermedia application design with OOHDM”, in Proceedings of ACM Hypertext’96, Washington DC, USA, March 16-20 1996, pp.116-28.
10. G.L. Souza Fillho and L.F.G. Soares, “Edition and execution of hypermedia document presentations in HyperProp”, in Proceedings of II Workshop on Hypermedia Systems, WOSH, Fortaleza, CE, Brazil, May 18-19 1996, pp. 61-70 (in Portuguese).
11. P.D. Stotts and R. Furuta, “Petri Net based hypertext: document structure with browsing semantics”, ACM Transactions on Information Systems, Vol. 7, No. 1, pp. 3-29, January 1989.
12. M. Thüring, J. Hannemann, and J.M. Haake, “Hypermedia and cognition: designing for comprehension”, Communications of the Association of Computer Machinery, Vol. 38, No. 8, pp. 57-66, August 1995.

13. M.A.S. Turine, M.C.F Oliveira and P.C. Masiero, “Designing structured hypertext with HMBS”, in Proceedings ACM Hypertext’97, VIII Int. ACM Hypertext Conference, Southampton, UK, April 6-11 1997, pp. 241-56.
14. M.A.S. Turine, “HMBS: A statechart-based model for the formal specification of hyperdocuments”, Doctoral Thesis, IFSC, USP, São Carlos, Brazil, June 1998 (in Portuguese).
15. G. Van Rosum, J. Jansen, J., K.S. Mullender, and D.C.A. Bulterman, “CMIFed: A presentation environment for portable hypermedia documents”, in Proceedings of I ACM Int. Conference on Multimedia, Anaheim, CA, USA, August 1993, pp.183-188.