A Framework for Hypermedia Design and Usability Evaluation

F. Garzotto, M. Matera, P. Paolini
HOC-Hypermedia Open Center, Department of Electronics and Information, Politecnico di Milano
Piazza Leonardo da Vinci, 32 - 20133 Milano, Italy
Phone: +39-2-23993520 - Fax: +39-2-23993411
{garzotto, matera, paolini}@elet.polimi.it

Abstract

This paper proposes a *unified* framework for the design and the usability evaluation of hypermedia applications. By providing a *design model*, a set of *design guidelines*, and a set of patterns of evaluation activities called *abstract tasks*, the framework helps a development team to perform both design and usability inspection in a systematic and cost effective way, and supports standardisation of activities and results across different designers and evaluators. The paper presents the framework and examples of its use, also reporting usability weaknesses detected in some commercially available hypermedia CD-ROMs.

Keywords
Hypermedia, Usability Evaluation, Hypermedia Design, HDM.

1 INTRODUCTION

It is generally acknowledged that the quality of a software product is strongly dependent from the quality of its design. In particular, design quality has effects on usability, a fundamental quality factor (Fenton, 1991) which concerns how easy is for users to learn a system, and how efficiently and pleasantly they can use it. We have explored the relationship 'design-usability' in a specific class of software products - *hypermedia*, and we have defined a unified framework that supports both the hypermedia design process and the usability evaluation activity.

The constituents of our framework are a *hypermedia design model* (HDM'98), a set of *design guidelines,* and a set of evaluation patterns for hypermedia usability called *abstract tasks.* The rational of our approach is the following. Design must be

A. Sutcliffe et al. (eds.), *Designing Effective and Usable Multimedia Systems*

supported by an expressive *model* (Garzotto et al., 1993), i.e., a language to describe the application constituents and to specify the design decisions, and by a set of *guidelines* which suggest how to achieve a good design. At the same time, the model identifies the 'subjects of interest' (Fenton, 1991) for evaluation, i.e., the application constituents which the evaluator should focus on; the guidelines suggest some usability properties of these constituents. The set of *abstract tasks* defines which operations must be actually executed on the application constituents to verify their usability.

In our approach, usability evaluation proceeds by *inspection* (Nielsen, 1993), i.e., it does not involve end users, but expert evaluators only. Although it is well known that the most reliable evaluation results can be achieved by combining inspection with user testing (Faraday et al., 1996), inspection techniques have the advantage that "... they save users (Nielsen, 1993)", do not require special equipment or lab facilities, and therefore are cheaper to use.

Finally, our framework distinguishes among different *categories* of design guidelines and evaluation tasks. Each category addresses design and usability of different *dimensions* along which a hypermedia application can be analysed: *content*, i.e., the actual information pieces stored in the application; *structure*, i.e., the organisation of the application content; *navigation*, i.e., the actual links and browsing mechanisms available to explore such structures; *dynamics*, i.e., the run-time behaviour of time-based media and links; *user control*, i.e., the operations available to the user to control the application dynamics; *presentation*, i.e., how all the above features are shown to readers (in other words, the visual properties of lay-out elements - buttons, windows, content fields, menus, etc.). So far, our framework addresses design and evaluation issues related to content, structure, navigation, dynamics, and user control; extensions to address presentation dimensions are subject to our on-going research.

The rest of the paper presents an overview of our framework, focusing on abstract tasks which are the most original aspect of our approach. Section 2 reports a short summary of the HDM'98 model. Design guidelines are briefly described in section 3. Abstract tasks are discussed in section 4, which also reports examples of usability problems detected with our evaluation framework in some commercial hypermedia CD ROMs. Conclusions and directions of our future work are described in section 5.

2 THE HDM'98 DESIGN MODEL

A primary component of our framework is HDM'98 (Garzotto et al., 1998b), the latest version of the Hypermedia Design Model HDM (Garzotto et al., 1993; Garzotto et al., 1994; Garzotto et al., 1995). For lack of space, in this paper we will only provide a short summary of the HDM'98 terminology, to help readers understand some terms frequently used in the following sections. For a discussion on the rationale of the various concepts, the reader is referred to previous publications.

In its current release, HDM'98 focuses on structural, navigational, dynamic, and user control 'dimensions' of hypermedia (as defined in the introduction) abstracting from presentation features.

Primitives for structural modelling distinguish between two sets of structures: *hyperbase* structures - which constitute the so called *hyperbase layer (hyperbase* for short) of the application, and *access* structures - which constitute the so called *access layer.* Hyperbase structures are used to represent domain information, while access structures provide entry points to the hyperbase. The hyperbase consists of *entities* and *semantic connections* among (parts of) them. Entities denote conceptual or physical objects of the application domain and are composite objects; their logical constituents are called *components*, and are organised according to some topological patterns (e.g., sequences, trees, lattices). Components in turn are made of *nodes.* Nodes are the actual containers of the multimedia data describing a component, and aggregate a number of content elements called *slots.* A node may correspond to a page, a page section, a full screen or partial screen window, depending on the adopted lay-out strategy. Their semantics is that different nodes of the same component describe different *perspectives*, i.e., different aspects concerning the component subject. A slot within a node can be *static* or *dynamic,* depending whether it stores time-independent media (such as formatted data, text strings, images and graphics) or time-based media (as video, sound, or animation).

The access layer consists of *collections.* A collection groups a number of *members*, in order to make them accessible. The members of a collection could be either hyperbase elements (entities, components, or nodes) or other collections (nested collections). A collection typically has (although it is not mandatory) a distinguished node called *centre*, which is informative about the collection content and is the starting point of the navigation within the collection. Members are collected according to some semantic criteria (e.g., in a museum application, 'all paintings of a painting school X', or according to an expected user's goal (e.g., 'the top ten paintings' for a quick visit of the museum pieces). 'Tours' or tables of contents are modeled as collections in HDM'98.

Navigation primitives enable the description of browsing paths, i.e., links connecting nodes within the various structures. In HDM'98, links are of different categories: *structural*, *applicative*, or *collection* links. Structural links connect nodes within an entity according to its topology; applicative links connect nodes of different entities related by some semantic connections; collection links connect the constituents of a collection. If a collection has links connecting each member to another one in a given order, it is called *guided tour.* If a collection has links connecting centre with all members, and vice versa, it is called *index.* A *guided tour index* is a collection which includes both sets.

Dynamic primitives describe *behaviour of dynamic slots* and *links.* The behaviour of a dynamic slot concerns how its state evolves along the time by effect of user interaction, discussed below, or in dependency of the state of other slots.

HDM'98 provides a set of *temporal relationships* among slots occurring within the same node* . The behaviour of links refers to the effects of link traversing on the state of slots in the source and destination nodes. When a destination node is left and another is activated as effect of following a link, slots in the source (respectively, in the target) can be *paused* or *stopped* or *kept* playing, depending on the behavioural semantics of the link. The behaviour of links also concerns a mechanism sometime called *automatic navigation.* Automatic navigation means that the transition from a node to another one is performed automatically by the application, either by means of a time-out mechanism, or by synchronising the change of context with the execution of time-based media. For example, the transfer from node A to node B occurs when the audio comment on node A is over.

Finally, HDM'98 primitives for user control refers to the operations available to the user to control the behaviour of links (i.e., the effects of link traversing and automatic navigation - see above) and the behaviour of slots.

3 DESIGN GUIDELINES

The design guidelines proposed in our framework are empirical, in that they are founded on the personal experience of the authors and their group. We have designed, developed, and evaluated hypermedia applications for several years, for different companies and institutions, in a variety of domains; our guidelines try to capture the application properties that we consider useful to get well designed and usable applications.

Our design guidelines are organised in various categories - *structural* guidelines, *navigation* guidelines, *dynamic* guidelines, *user control* guidelines, and *content guidelines*, according to the multiple dimensions of a hypermedia that we have explored so far in our research.

The framework also includes two *meta-guidelines: 'Be consistent'* and *'Match the situation of use'*. Consistency, one of the most general principles of good design, means that conceptually similar elements are treated in a similar fashion, while conceptually different elements are treated differently. The second meta-guideline corresponds to another general principle of good design, known as *task conformance* (Dix et al., 1993; Mayhew, 1992). Task conformance means that any design choice should take into account the physical and temporal context in which an application is used, the reason why users use the system, and their actual mental model. These rules are 'meta' with respect to all other guidelines since they can be

---

* The most important are *exclusiveness, disjointness, concurrency,* and *synchronization.* Two slots are mutually *exclusive*, if they cannot be active simultaneously. Two slots are *disjoint* if they can be active (and controlled) one independently from the other. Two disjoint dynamic slots are *concurrent* when they can be simultaneously active. Two slots are *synchronized* if they satisfy mutual temporal constraints (e.g., one becomes automatically active 'after' the other is de-activated).

applied to each dimension and property of an application, and are implicitly included within each guideline.

The guidelines we have defined so far are reported in the following tables. For lack of space, we will not discuss each guideline in detail, but will include only short comments or examples to clarify their meaning. The reader is referred to (Garzotto et. al., 1998b) for a more complete discussion.

Table 1 Guidelines for Structural Design

S1	***Define appropriate structures for the application content*** The way of organising the hyperbase layer should be adequate to the size, the complexity, the semantics of the actual content. In the hyperbase, for example, if information about some domain objects are scarce, entities with a single component (in turn with a single node) are probably the best solution. On the contrary, a large amount of content should be better represented by entities structured in various components and nodes. Structure design and navigation design are strongly correlated, and this guideline should be considered in conjunction with *N1* (see next table).
S2	***Make access layer organisation 'complete' with respect to the hyperbase organisation*** S2 addresses the access layer coverage issue, prescribing that each instance of each entity type should be a member of at least one collection. The rationale is that if an entity is mentioned nowhere in the access layer, users may never become aware of its existence until they traverse an applicative link (if any) taking to it.

Table 2 Guidelines for Content Design

C1	***Choose appropriate media, with appropriate 'format', to fit the content message of nodes*** The designer of node content must choose the best format to convey the content message, considering the appropriateness of a medium or a combination of media, their physical features (e.g., as resolution, indicative size or duration), as well as rhetorical aspects, such as the literary style of text or the visual style of visual media.
C2	***Make the content appropriate to the chosen delivery medium, its format, and the structure in which it occurs*** This guideline is the dual of the previous one. Once the node structure is defined, a node must be filled in with content which is coherent with such structure and with the physical and rhetorical format of the various slots.
C3	***In collection centres, provide 'correct' information about collection members*** Collection centres must support correct user's understanding of what is in the collection, and how the collection is structured: i) the centre must store descriptors (text labels, icons, miniaturised pictures, or similar) to support the identification of *all* collection members, and only of them*; ii) the visual order of descriptors must corresponds to the navigational order among collection members. For example, if during forward navigation in a linear collection user finds a link 'next' from X to Y, the collection centre should show the titles of these two members, X and Y, one after the other, and not in a different order.
C4	***When reusing a piece of information in a new context, adapt the portion of content which is strictly dependent on the original context*** If a piece of content in a node depends on a given context, it should be removed when reusing the node in another context (and may be replaced with information needed by the new situation). For example, a textual reference in a node to the next node in a given collection must be removed when such a node is placed in a different context, where the following nodes may be different. This guideline is the companion, for content, of guideline N3 for navigation - see table 3.

* It might contain additional content, but member descriptors should be the primary content transmitted to the users

Table 3 Guidelines for Navigation Design

N1	***Define navigational patterns appropriate for the topology of hyperbase and access structures, and for the complexity of the structures content*** Links within and among hyperbase structures and access layer structures should be consistent with the topology of these structures. In linear entities, for example, we expect at least the links 'next' and 'previous' from a component to the following one and vice versa; still, additional links may also be useful for exploring a large component, e.g., 'first' and 'last' links to directly jump to the first or the last component.
N2	***Provide visible and efficient quit mechanisms*** A general usability principle is to allow users to rapidly quit the application at any moment (Nielsen, 1994; Hardman et al., 1989); in hypermedia, this can be achieved by providing each node with an *easy understandable* quit command, or with a direct link to the place where such command is available. Most hypermedia applications provide the quit command only in one node (typically, the home page), but require many steps before reaching this context. In other cases, the quit function is not visible, and it requires to use platform specific shortcuts (e.g., 'Alt+F4' for Windows) not obvious for all users.
N3	***When reusing the same structure in a new context, remove or modify links that are strictly dependent on a different context*** Consider, for example, the reuse of nodes across different linear collections. 'Next' and 'previous' links, from a node to the following and the preceding one, are strongly dependent on the actual collection and its linear order. Reusing the same node in another collection, with different members and a different order, requires to modify the destinations of such 'next' and 'previous' links. N3 is the companion, for navigation, of guideline C4 for navigation (see table 2.)
N4	***Support user perception of his/her current navigation context*** N4 is related to the 'getting lost in the hyperspace' problem - a typical usability issue for large hypermedia. To reduce the disorientation effect, N4 suggests that users should be always aware of the actual status of their navigation session, i.e., they should be able to understand their current position within the current entity or the current collection or the entire application. For this purpose, many hypermedia use active maps and overview diagrams, with indications of the user's current location (and of previous steps), or some perceivable visual cues - for example, different page backgrounds of nodes to distinguish among different types of entities, or textual labels to indicate the title of the current entity.
N5	***Keep backtracking facility distinct from hyperbase and access navigation*** Backtracking allows users to navigate, step by step, back to previous visited nodes. To avoid a potential source of disorientation, N5 prescribes *not* to provide backtracking commands in place of explicit navigation links, even in situations where their effects seem to be equivalent. For example, imagine that a user first navigates the entire structure sequentially, and then finds a way to jump directly to a node X from a node Y different from the one preceding X in the sequence. If 'previous' links are implemented by using backtracking, the use of this link from X returns the user to Y, and not to the node preceding X, as he or she would probably expect.

Table 4 Guidelines for Dynamics Design

D1	***Avoid behaviour interference among concurrent dynamic slots*** Dynamic slots are *concurrent* when they are simultaneously active (see section 2). D1 prescribes that each disjoint slot should exhibit the same behaviour both when it is active individually, and when it is active concurrently with other disjoint slots, avoiding mutual interference and side-effects. The rationale for this guideline is that for building up a predictive model of how dynamic media behave (which is crucial for usability), users first try to understand how each medium behaves and can be controlled individually; then they experiment what happens when several media are simultaneously active. It is easier to recognise the sum of the individual behaviours of the various media, rather than to understand a new different behavioural combination.
D2	***Define link behaviour appropriate for the link semantics and the content of source/target nodes*** This guideline considers the effects of link traversing on the state of source and destination nodes, and on the behaviour of their dynamic slots. Dynamic slots in the source (or the target) might be reset to their initial state, or paused, or kept playing (see discussion in section 2.). The designer should consider a number of factors in order to decide which choice is more appropriate: the nature of dynamic slots, their duration, their content message, the combination of their states when links are traversed. For example, a sound slot in the source should be paused or stopped if link traversing automatically activates another sound slot in the destination, or if its semantic content is totally meaningless in the new context reached by navigation.

Table 5 Guidelines for User Control Design

UC1	***Provide user control on dynamic slots appropriate for the nature of their content, and for their format*** The commands designed for the user to manipulate the state of a dynamic slot depend upon various factors; among them, the nature of the slot (e.g., a picture can be zoomed in or out, but the same commands make no sense for a sound) and its physical properties such as resolution, size, duration-control commands such as 'start', 'stop', 'pause', 're-start', 'forward', 'backward' are meaningful, in principle, for all dynamic slots, but a video or a sound comment might require no interaction if they are very short. Ultimately, the degree of control must be appropriate to the actual need of users, based on their experience with digital multimedia and their goals in using the system.
UC2	***Provide user control on automatic navigation appropriate for the content of structures and their size*** This guideline refers to the user's ability of controlling the execution of automatic navigation (see section 2), e.g. suspending it, or switching from automatic to manual navigation, and vice versa. As for the control of dynamic media, the degree of control on automatic navigation depends upon various factors, such as size, content, and intended use of the navigation structure. Considerations similar to those mentioned for guideline UC1 can also be applied here.

4 ABSTRACT TASKS

Abstract tasks are patterns of operational activities that the evaluators should perform during the inspection in order to detect usability defects (Garzotto et al., 1998a). We use the term 'abstract', since: i) the activity specifications are formulated independently from a particular application, and ii) they refer to categories, or 'types', of application constituents more than to specific constituents.

Like the design guidelines, our abstract tasks are mainly empirical, in that they capture our experience on hypermedia product evaluation: they describe, using the HDM'98 vocabulary, what *we* do when we inspect a product for usability.

An abstract task is composed by five elements: the *Title*; the *Focus of Action*, i.e., a list of application constituents which are the focus of the evaluation activity; the *Activity Description*, i.e., what the evaluators have to do; the *Intent,* which is a short statement explaining what is the rationale of the abstract task, and which guideline(s) it refers to. It is important to note that, beside the evaluation activities explicitly described for the abstract tasks, there is an additional activity which is left implicit in the task formulation, although it is performed during (or after) the execution of each task. It concerns *consistency checking:* each abstract task has to be executed, in principle, on *all* the application objects of the category addressed by the task (mentioned in the 'focus of interest'), in order to verify that conceptually similar elements have been designed and implemented in a consistent fashion across the application, and therefore show the same (good or bad) features. Sometimes consistency checking can not be accomplished exhaustively, especially for large applications. Therefore most times it is executed by induction: during an evaluation session, abstract tasks are applied only to a limited sample of objects, and the results are then generalised. The choice of the sample of objects might be difficult, and there is the risk of considering objects that do not show any problem, omitting other objects that might be more critical. From our experience, evaluators tend to start evaluation without choosing a priori such a sample; they just start executing abstract tasks on some random objects (the number of which depends on the evaluator's personal style, the dimension of the application, and the intended duration of inspection). Then, they are induced to continue executing abstract tasks on additional objects if they find violations, with the intent of determining the severity of the detected problems on a larger set of situations.

Like design guidelines, abstract tasks are organised in various categories, according to the multiple dimensions along which a hypermedia can be analysed: structural tasks, content tasks, navigation tasks, dynamics tasks, user control tasks.

In this section, we will report a sample of abstract tasks, one for each task category. The reader is deferred to (Garzotto et al., 1998b) for a complete list, which currently amounts to thirty five abstract tasks. The tasks reported in this paper are the most representative of our approach, and those which helped us to discover the most frequent problems. For each abstract task, we will describe

examples of usability problems, detected on the seven commercial CD ROMs: *Art Gallery* (by Microsoft, 1993), a hypermedia guide to the National Gallery Museum in London; *Il Seicento* (by Opera Multimedia, 1995), an application about the European History of the XV century, whose content responsible is Umberto Eco; *La Pinacoteca Vaticana* (by E.M.M.E Interactive, 1996), an application about the painting collections of Vaticano, Rome; *Le Louvre* (by Montparnasse Multimedia, and Reunion des Musées Nationaux, 1994), a hypermedia guide to the paintings in the Louvre Museum in Paris that in 1995 won the 'best CD-ROM' award at MILIA'95 - one of the largest exhibition of multimedia titles world wide; *Musée d'Orsay* (by Montparnasse Multimedia, and Reunion des Musées Nationaux, 1996), a hypermedia guide to the paintings in the Musée d'Orsay, Paris; *The Italian Metamorphosis, 1943-1968* (by ENEL Italy, Progetti Museali, and Guggenheim Museum NY, 1994), which derives from an exhibition held at the Solomon R. Guggenheim Museum in New York.

4.1 Abstract task for structures

Title: 'Coverage power of access structures'
Focus of Action: entity types + collections.
Activity Description: consider an entity type.

1. verify if there are collections which allow users to access its instances;
2. verify if there is at least one collection which allows users to access all its instances.

Intent: to verify the completeness of the application entry points, i.e., if the access structure efficiently supports the access to the hyperbase entities (see guideline S2).
Detected Problems: the application ***Musée d'Orsay*** has three hyperbase entities: 'Painting Collections', 'Exhibition Rooms', and 'Painters'. What we noticed is that there are no entry points for the entity 'Painters'. The top level index allows users to access only the entities 'Painting Collections' and 'Exhibition Rooms'. Moreover, there are no collections including the instances of the entity 'Painters'. The only way to access this entity is to navigate in the hyperbase, i.e., to follow applicative links from the instances of the entity 'paintings'.

4.2 Abstract task for content

Title: 'Accurateness of the information content in collection centres'
Focus of Action: collection centres.
Activity Description: verify if the information content of a collection centre accurately describes the content of the collection. For example:

1. verify its *correctness*, i.e., if the supplied descriptions correspond to the actual content of the collection;
2. verify its *completeness*, i.e., if it gives indication about *all* the members in the collection;

3. verify its *ordering*, i.e., if the order in which collection member descriptors are visually listed in the centre corresponds to the navigation order among collection members.

Intent: to verify how well the centre of a collection supports users' understanding of what is and what is not in the collection (see guideline C3).

Detected Problems: in ***La Pinacoteca Vaticana,*** there are several collections (corresponding to different painting taxonomies), that have a centre presenting some thumbnails, one for each painting belonging to the collection. A click on a thumbnail allows users to enter the collection, and to get to the painting node. Starting from there, it is possible to navigate both forward and backward (two buttons are provided). What is surprising is that such navigation follows an order which is exactly the opposite of the one suggested by the collection centre. Therefore, in each collection member, the 'next' (respectively 'previous') button leads to the previous (respectively next) painting displayed in the collection centre.

4.3 Abstract tasks for navigation

Title:' Complexity of applicative navigation patterns'

Focus of Action: applicative links.

Activity Description: in an applicative link:

1. navigate from the source node to one of the target nodes;
2. randomly visit one of the target nodes;
3. systematically visit all the target nodes;
4. every time a target node is reached, try to navigate back to the source node, without using backtracking commands.

Intent: to verify if an applicative link has a navigation patterns which is appropriate for the semantic relationship it represents, and if it includes symmetric links from the target nodes to the source nodes (see guideline N1).

Detected Problems: in ***Art Gallery***, by executing this task on several applicative links, we discovered instances of the same link type that are symmetric, and other instances that can be traversed only in one way. There is a link, for example, from *Tempera* to *The Baptism of Christ* (*Tempera* is the technique used for that painting), but there is no reverse link from *The Baptism of Christ* to its technique.

Title: 'Visibility of navigation status in collection navigation'

Focus of Action: collections.

Activity Description: in a collection, access an arbitrary member, and identify its position in the collection structure.

Intent: to verify if members of a collection contain clear indications about their location in the collection, so that to support users' orientation (see guideline N4).

Detected Problems: in the application ***Il Seicento,*** each covered 'topic' is presented as a 'book chapter', and organised in a sequence of 'pages', with two distinct buttons for going back and forth.

In the pages, there are no presentation elements that help to identify which point of the sequence has been reached, but a little icon, representing a stack of sheets which changes, adding or removing one sheet after the users moves two pages forward or backward. In our opinion this is a poor and not much visible mechanism for representing the navigation status, especially if compared with the mechanisms provided in other applications, where an explicit label indicates, for each navigational step, which member has been reached, how many members have been already visited, how many members are left.

4.4 Abstract task for dynamics (media and links behaviour)

Title: 'Link behaviour & dynamic slots'
Focus of Action: dynamic slots + links
Activity Description: consider a dynamic slot:

1. activate it, and then follow one (or more) link(s) *while the slot is still active*; return to the 'original' node where the slot is placed, and verify the actual slot state;
2. activate the dynamic slot; *suspend it*; follow one (or more) link(s); return to the original node where the slot has been suspended and verify the actual slot state;
3. execute 1 and 2 traversing different types of links (both to leave the node and to return to it);
4. execute 1 and 2 by using only backtracking to return to the original node.

Intent: to verify the cross effects of navigation on the behaviour of dynamic slots, i.e., what happens when the activation of a slot is followed by the execution of navigational links and, eventually, backtracking (see guideline B2).
Detected Problems: in the ***Louvre*** application there are nodes of type 'Painting Presentation', which show a full screen painting image with an audio comment. By applying this abstract task on these nodes, we noticed that the audio is interrupted when the user navigates to another node. In other nodes, those of type 'Loupe', we discovered instead that, if a link is selected while animation and audio are still active, the audio comment continues till the end of the current audio 'slice', although the current node is immediately replaced by the link destination node. Thus users finds themselves on a content which has nothing to do with the comment they are listening to. What is even more surprising is the following: if the selected link takes to a different entity, any further click anywhere on the destination node interrupts the play of the current audio slice, but if the link is structural, i.e., takes to a different component of *the same* entity, any further click does not interrupt the audio slice.

4.5 Abstract task for user control

Title: 'Complexity of control on automatic guided tour navigation'
Focus of Action: collections or entities with automatic navigation.
Activity Description: in an automatic guided tour:

1. verify the complexity of control on the automatic navigation, in terms of number and type of control commands. For example, suspend the automatic navigation and restart it, or suspend the automatic navigation and proceed manually in the collection navigation, etc..
2. verify if the set of the control commands is appropriate, in accordance with the collection structure and organisation.

Intent: to verify the appropriateness of the commands for controlling the automatic guided tour navigation (see guideline UC1). This task is the analogous, for navigation, of the control task defined above for dynamic slots.
Detected Problems: in the application ***Italian Metamorphosis, 1943-1968,*** entities are organised as linear sequences of pages (nodes), each one containing three synchronised media: a scrolling text, a slide show of images, and a audio comment. Navigation along these pages is automatic, and starts as the entity is entered. The transition from one page to another occurs automatically at the end of each sound comment. The only available command to control the automatic navigation is 'STOP', which interrupts the sound command, and abruptly takes the user to the last page. There is no way to restart the activation, unless the user is willing to play the 'usual' trick of navigating somewhere else, and then start the navigation again. This behaviour is consistent across the application. Unfortunately, the lack of control is disturbing, and the effect of the stop command is not self-evident: the users find themselves on a totally new page (the last one), and might get disoriented.

5 CONCLUSIONS

The intended users of our framework are mainly hypermedia design and usability specialists, but they can also be software developers or practitioners. The framework can be used in several stages of the hypermedia development process: model and guidelines are useful during the design phase, abstract tasks during design evaluation, prototype evaluation, and final product evaluation. The output of the design phase is a HDM'98 specification of the application schema, i.e., the types of hyperbase and access structures, their behaviour, and the user operations available on the various types of objects. The output of an evaluation is an organised list of potential usability problems, classified according to the various categories of objects and features of the application.

The use of a framework like the one proposed in this paper has several advantages. It can be used to approach the processes of hypermedia design and evaluation more systematically and efficiently; it can improve the communication among the members of the development team, by providing a common vocabulary

of concepts, terms, and principles; it can support standardisation across different designers and evaluators.

So far, our framework has not addressed design and evaluation of presentation issues, i.e., all features of an applications which concern lay-out objects and their properties. Although most presentation rules can be defined in terms of conventions, standards, generic principles for user interface design, which can be found in the HCI literature (Dix et al., 1993; Mayhew, 1992; Preece, 1994), it is also true that we need presentation models, guidelines, and abstract tasks that address hypermedia specific features (e.g., anchor visualisation). Together with the investigation of additional guidelines and abstract tasks for content design and evaluation, presentation issues are the subjects of our current activity to complete the framework.

A further aspect, not addressed by our framework yet, has to do with rating the severity of the detected usability problems. This is necessary in order to prioritise the activities needed to fix the problems, and to avoid expending disproportionate effort on low-priority problems. Severity ratings are derived from an estimate of the expected user impact of each usability problem, as well as budget issues. Identifying criteria for severity ratings is one of the directions of our future work. A related direction of future research concerns defining a more precise mapping between design guidelines and situations of use. We need to relate hypermedia specific categories of user tasks, application domains, contexts of use, to the applicability of design guidelines.

Finally, we are planning to validate the overall framework, currently based on our long-term personal experience, by performing experiments involving users .

6 ACKNOWLEDGMENTS

This work has been partially founded by the European Commission Project 'SITMOON', and the CNR Project 'Museo Virtuale dell'Informatica'.

We acknowledge Prof. M. F. Costabile, from Department of Computer Science, University of Bari, Italy, for her valuable suggestions.

7 REFERENCES

Dix, A., Finlay, J., Abowd, G., Beale, R. (1993) Human-Computer Interaction. Prentice Hall, 1993.

Faraday, P., Sutcliffe, A. (1996) An Empirical Study of Attending and Comprehending Multimedia Presentations. Proc. of ACM Multimedia'96 Conference (Boston, MA), ACM Press.

Fenton, N.E. (1991) Software Metrics: A Rigorous Approach. Chapman & Hall.

Garzotto, F., Paolini, P., Schwabe, D. (1993) HDM - A Model Based Approach to Hypermedia Application Design. *ACM Trans. Inf. Syst.*, **11** (1), 1-26.

Garzotto, F., Mainetti, L., Paolini, P. (1994) Adding Multimedia Collection to the Dexter Model. Proc. of ECHT'94 - ACM Conference on Hypermedia Technology, Edinburgh, UK.

Garzotto, F., Mainetti, L., Paolini, P. (1995) Hypermedia Design, Analysis, and Evaluation Issues. *Comm. ACM*, **38** (8), 74-86.

Garzotto, F., Mainetti, L., Paolini, P. (1996) Information Reuse in Hypermedia Applications. Proc. of ACM HT'96 Conference (Boston, MA), ACM Press.

Garzotto, F., Matera, M., Paolini, P. (1998a) A Systematic Method for Hypermedia Usability Inspection, *The New Review of Hypermedia and Multimedia,* Taylor Graham - to appear.

Garzotto, F., Matera, M., Paolini, P. (1998b) Design Guidelines and Usability Evaluation Patterns for Hypermedia, Technical Report 98-03, Dipartimento di Elettronica e Informazione, Politecnico di Milano.

Hardman, L. (1989) Evaluating Usability of Glasgow Online Hypertext. *Hypermedia,* **1**(1), 34-63.

Hardman, L., Bulterman, D.C.A., Van Rossum, G. (1994) The Amsterdam Hypermedia Model: Adding Time and Context to the Dexter Model. *Comm. ACM, 37 (2),* 50-62.

Mayhew, D.J. (1992) Principles and Guidelines in Software User Interface Design, Prentice Hall, Englewood Cliffs.

Nielsen, J., (1993) Usability Engineering, Academic Press, New York.

Nielsen, J., Mack, R.L. (1994) Usability Inspection Methods. John Wiley & Sons, New York.

Preece, J. (1994) Human-Computer Interaction. Addison Wesley.

7 BIOGRAPHY

Franca Garzotto is Associate Professor of Fundamentals of Computing at the Department of Electronics and Information, Politecnico di Milano. Her research interests include hypermedia modelling, authoring, evaluation, and multimedia for cultural heritage and education. She has been involved in various European research projects in the above fields.

Maristella Matera is a Ph.D. student at the Department of Electronics and Information, Politecnico di Milano. Her research interests span visual interfaces for databases, intelligent multimedia presentation systems, hypermedia modelling, usability engineering, usability testing.

Paolo Paolini is Full Professor of Multimedia and Computer Graphics at the Department of Electronics and Information, Politecnico di Milano, where he also serves as Scientific Director of HOC (Hypermedia Open Center). His research interests include hypermedia design, hypermedia development systems, WWW interfaces to data bases. He has been scientific responsible of various European research projects in the above fields. He is currently Associated Editor of ACM Transactions on Information Systems (TOIS).