

User Interface Patterns for Hypermedia Applications

Fernando Lyardet *, Gustavo Rossi *, Daniel Schwabe **

*LIFIA Depto. de Informática. UNLP.

La Plata, Argentina

E-mail: {fer,gustavo}@sol.info.unlp.edu.ar

**Departamento de Informática, PUC-Rio, Brazil

E-mail: schwabe@inf.puc-rio.br

Introduction

During the last four years we have been developing hypermedia applications (in CD-ROM and in the Web) using the Object-Oriented Hypermedia Design Method (OOHDM). OOHDM explicitly separates navigation from user interface design; this means that design decisions related with the navigational topology of the application are (in a broader sense) independent respect to those regarding interface issues (See for example [Schwabe96, Schwabe98]). Separating navigational from user interface design allows us to define different interfaces for the same navigation structure and maximize modularity.

We have mined many patterns that show recurrent design problems and their solutions both while developing the navigational architecture of the application and when building its user interface [Rossi96, Rossi97, Rossi99, Lyardet99]. In this paper, we present some patterns related to the design of user interfaces for hypermedia applications.

To put the patterns in context we must briefly explain which is the product of navigational design. During this activity we obtain a set of nodes (that contain multimedia information and anchors) and links that connect them. We also define a set of navigational contexts, i.e. sets in which the user will navigate. The user interface design activity aims at defining how nodes are perceived; this involves defining the interface of their attributes and anchors. One of the forces that constantly appears in all interface patterns is that we may have many kinds of information items in a node, which may have also different purposes: showing some attribute, allowing to trigger a link to other node or even activating not hypermedia functionality (such as initiating a database transaction in an electronic store). We must organize those items in a scarce perceivable space, make them understandable, avoid cognitive overhead, etc.

We next introduce some of our user interface patterns. Although these patterns are not intended to define a pattern language like the one in [Alexander77], they cover most design decisions related with architectural aspects of the interface.

1. Information on Demand

Problem:

How to organize the interface in such a way that we can make perceivable all the information in a node taking into account both aesthetic and cognitive aspects?

Motivation:

We usually find ourselves struggling to decide how to show the attributes and anchors in a node. Unfortunately, the screen is usually smaller than what we need and many times we cannot make use of other media (such as simultaneously playing an audio tape

and showing an image) either for technological or cognitive reasons. Suppose for example an application on Paintings; we may want to show different attributes of a Painting such as Painter's name, year, museum, technical description, etc., but this is difficult to achieve if we want to maximize the space we dedicate to the picture itself.

A common problem appears when a node has an amount of information to be perceived by the reader that does not fit in one screen all together, or may distract the user's attention (for example, an audio recording). Furthermore, scrolling may be often not acceptable because the reader doesn't get an overall view about what he will find in that node; he will have to scroll all the way down to see if there is something that interests him or not.

It may be tempting partitioning the node by using different windows for presenting the information, and defining links among these new nodes. This is also problematic because in our attempt to match design with an implementation issue, we may pollute the overall application's navigational structure. The user may get the impression of dealing with multiple entities, becoming disoriented, while in fact he is accessing another part of the same conceptual entity.

Solution:

Present only a sub-set of the attributes, the most important ones, and let the user control which further information is presented in the screen, by providing him active interface objects (e.g. buttons). The activation of those buttons does not produce navigation; they just cause different attributes of the same node to be shown. This just follows the "What you see is what you need" principle. There are some considerations to be taken into account in this solution: for example we may use the same screen area to show different attributes, we may even select some attributes and allow them to appear together in the screen. When dealing with other kind of media attributes we must analyze the situation carefully: for example an audio recording does not use the screen; however it may also distract the user's attention so it is wise to give the user the chance to activate/deactivate its playing.

Known uses:

Information on Demand is used in Microsoft's Art Gallery, to provide further reading about a Painting; in Against All Odds' "Passage to Vietnam" CD-ROM, it is applied to let the user read further information about a photograph. It is also used widely in RMM's "Le Louvre".

In Figure 1 we show an example of Information on Demand in the context of Microsoft's Frank Lloyd Wright's CD. In Figure 2, we show an example of the same pattern in the WWW.

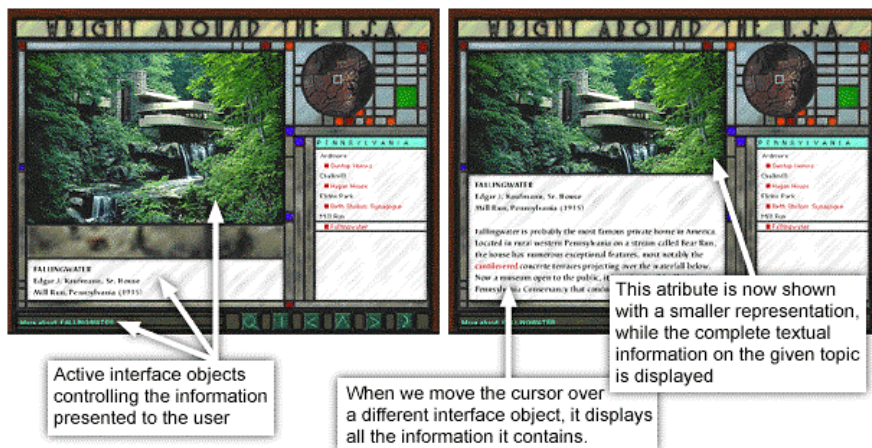


Figure 1: Information on Demand in Frank Lloyd Wright's CD-ROM.

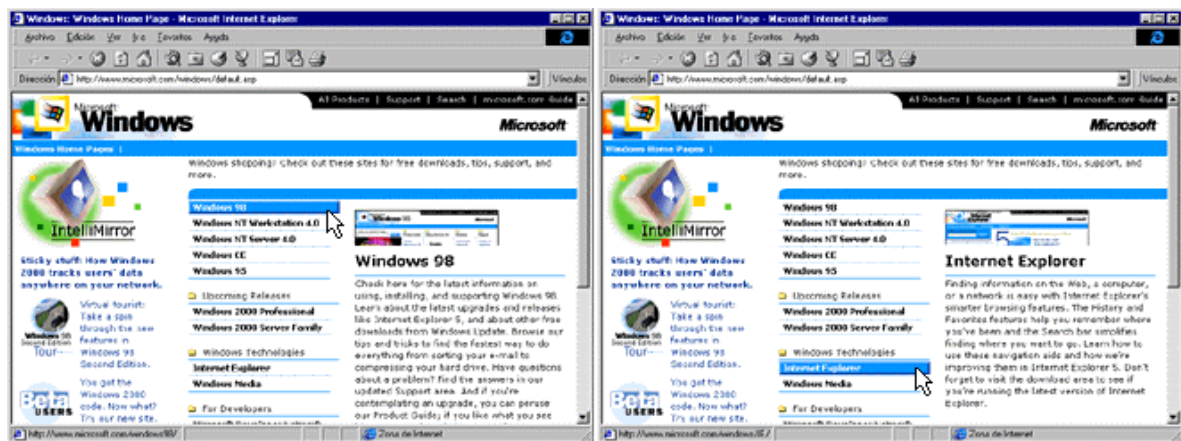


Figure 2: Information on Demand in www.microsoft.com/windows/. Notice that whenever the user selects a different product, the related information is displayed.

Consequences

- Less navigational overhead and context switch: users find all the information as a unit, selecting the information to be read
- Scrolling reduced to a minimum.
- Not suitable for printing. The website must provide a printer-friendly version of the page.
- On web implementations, browsers with JavaScript and CSS support are required.

Implementation

There are different implementation platforms (Multimedia Toolbook, Macromind/Flash and DHTML -HTML+ CSS + JavaScript/VBScript-), nevertheless a general outline of the implementation would be as follows:

1. Provide a list of the node contents.
2. Provide a handler for the OnMouseClicked event for each item in the list.

3. The OnMouseClicked handler should simply hide the current displayed item and show the new item.

Some implementations fail to achieve successful results using frames, since they are only capable of providing either in-page navigation, or force navigation when scrolling is avoided.

2. Behaviour Anticipation.

Problem:

How do you tell the user the effect or consequence of activating an interface object?

Motivation:

Many times, when building an interface, it is necessary to combine different interface elements such as buttons, hotwords, media controls or even custom-designed controls. It is usual to find readers wondering what has happened after activating a control, and the exact consequence of the action performed.

Solution:

Provide feedback about the effect of activating each interface element. Choose the kind of feedback to be non-ambiguous and complete: different cursor shapes, highlighting, small text-based explanations called "tool tips". In addition, these elements can be combined with sound and animations.

If we are using the behavioral Grouping interface pattern we can select different kinds of feed-back according the kind of behaviour provided; when the interface controls refer to a particular media such as animation, we could use a small status field for that family.

Known Uses:

In figure 3, there is an example from the Microsoft Atlas Encarta97. Each time the user positions the cursor over an interface element, a tool tip pops up with an explanation about the effect of activating the control. Similar examples are available on the web, where websites site uses standard GUI ToolTips and JavaScript combination to show information such as <http://www.nervemag.com/> (the ToolTip appears at the bottom of the page) or JavaScript only like www.mercedes.com homepage.

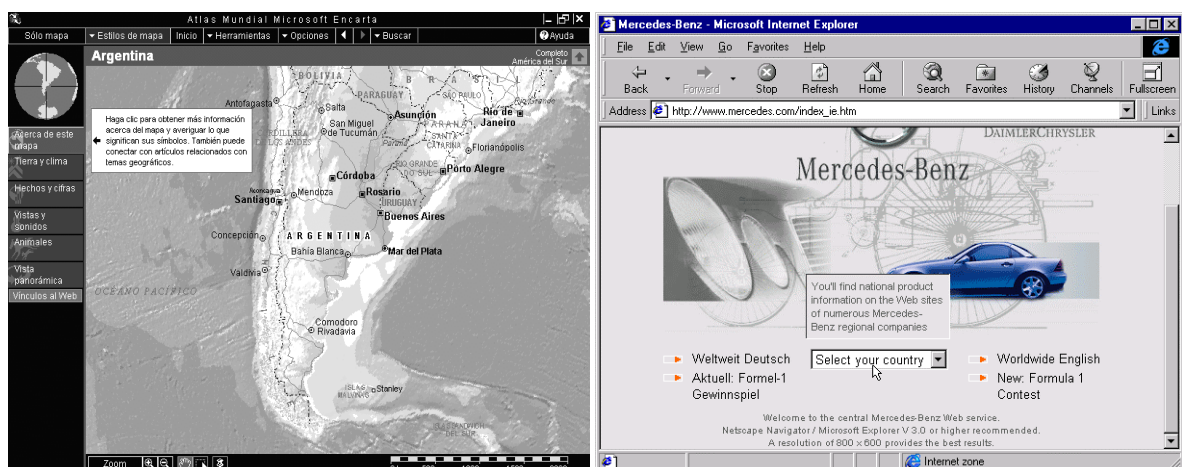


Figure 3 – Examples of “Behaviour Anticipation”. Notice the information displayed as the mouse moves over the different components of the interface.

Consequences:

- Users are informed about the contents of the destination node for a given anchor, thus reducing navigational overhead due to visiting irrelevant nodes.
- In web information systems, helps providing a good mean to handle user expectations about the operations displayed and their actual behavior.

Implementation:

Behaviour anticipation can be implemented in a much similar way like the *Information on Demand* pattern answering to the mouseOver events rather than waiting the user to click on the anchor, which would activate the link. Nevertheless, some implementations provide an audible Behaviour Anticipation instead, and a second click is required to confirm the activation of the link.

On the web, several websites provide feedback to the users by standard GUI hints (ToolTips) using the ALT modifier of the tag. Other implementations more sophisticated has been made using JavaScript to present additional information on the status bar and graphical animations.

3. Information-Interaction Decoupling.**Problem:**

How do you differentiate contents and various types of controls in the interface?

Motivation:

A page of a complex application display different contents, and is related to many other pages, thus providing many anchors. Moreover, if the page supplies means of control activation other than navigation (such as triggering some query), the user may experience cognitive overhead. It is well known that when too many anchors are provided in a text, the reader is distracted and cannot take profit of all of them.

Solution:

Separate the input communication channels from the output channels, by grouping both sets separately. Allow the "input interaction group" to remain fixed while "the output group" reacts dynamically to the control activation. Within the output group, it is also convenient to differentiate the "substantive information" (i.e. content) from the "status information". This solution not only improves the perception of a node's interface, but also the efficiency of the implementation.

Known Uses:

In figure 4, all links to related information on the current topic are displayed on the left. The graphics/video relevant to the current topic is displayed in the middle. Notice there are no links in the text itself.

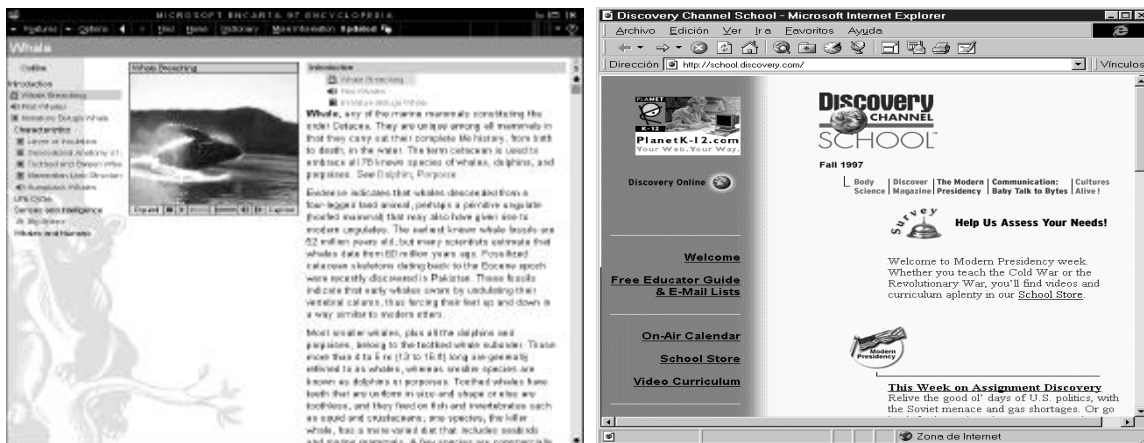


Figure 4 -Two examples of “Information-Interaction Decoupling” from the MS-Encarta97 CD-ROM and the Discovery Channel’s educational web page (<http://school.discovery.com/>)

Consequences.

- Contents are easier to read, while associative navigation among related nodes is also preserved.
- Some websites misuse this organization to overload the page with advertisements, promotions, etc.

Implementation

Even though the use of table has been always somewhat controversial as a layout specification, it is still the preferred implementation approach.

4. Behavioral Grouping

Problem:

How to recognize the different types of controls in the interface so that the user can easily understand them?

Motivation:

A problem we usually face when building the interface of a EIS is how to organize control Objects (such as anchors, buttons, etc.) to produce a meaningful interface. In a typical EIS there are different kinds of active interface objects: those that provide "general" navigation, such as "back" button, or anchors for returning to indexes, objects that provide navigation inside a context; objects that control the interface, etc. Even when applying Information-Interaction decoupling, there may be many different kinds of control objects.

Solution:

Group control interface objects according to their functionality in global, contextual, structural and application objects, and makes each group to enhance comprehension.

Known Uses:

In figure 5, the first picture is taken from MindQ's CD-ROM "An introduction to programming Java Applets" groups the navigation controls at the left and the current topic playing controls at the bottom. Similar examples are broadly available on the web (i.e. www.hotmail.com).

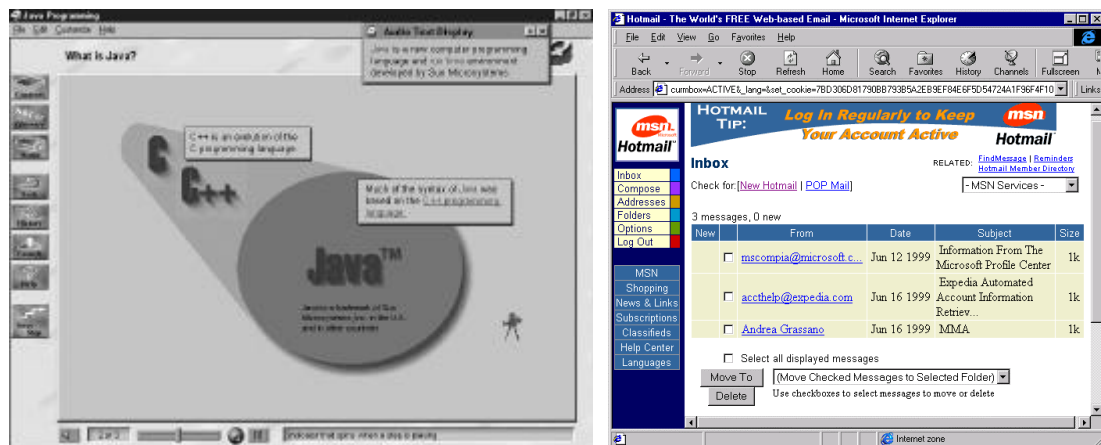


Figure 5 - An example of "Behavioural Grouping" from the MindQ's CD-ROM "An introduction to programming Java Applets" and <http://www.hotmail.com>.

Consequences.

- Easier to read, uncluttered interfaces.
- Helps the user identifying related operations.

Bibliography

- [Alexander 77] Alexander, S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King e S. Angel: "A Pattern Language". Oxford University Press, New York 1977
- [Gamma95] Gamma E.,Helm R., Johnson R., Vlissides J." *Design Patterns. Elements of Reusable Object Oriented Software.* ", Addison Wesley, 1995.
- [Lyardet98] F. Lyardet, G. Rossi and D. Schwabe: "Engineering Multimedia User Interfaces with Objects and Patterns". Second Workshop on Multimedia Software Engineering (MSE-IEEE), ICCSE'98 Conference. Kyoto, Japon.
- [Lyardet99] "Patterns for Adding Search Capabilities to Web Information Systems", F. Lyardet, G. Rossi, D. Schwabe. To be presented at EuroPLOP'99. Bad-Issee, Germany.
- [Rossi96] G. Rossi, A. Garrido, S. Carvalho: "Patterns for object-oriented hypermedia applications". In Pattern Languages of Programs II, Addison Wesley, 1996.
- [Rossi 97] G. Rossi, D. Schwabe and A. Garrido : Design Reuse in Hypermedia Design Applications Development Proceedings of ACM International Conference on Hypertext (Hypertext'97), Southampton, UK, 1997, ACM Press.
- [Rossi99] G. Rossi, D. Schwabe and F. Lyardet: "Patterns for designing navigable information spaces". To appear in Pattern Languages of Programs IV, Addison Wesley, 1999.
- [Schwabe 96] D. Schwabe, G. Rossi and S. D. J. Barbosa . "Systematic Hypermedia Application Design with OOHDM". Proceedings of Hypertext'96 (HT96). Washington, March1996.
- [Schwabe98] D. Schwabe, G. Rossi: "An object-oriented approach to web-based application design". Theory and Practice of object Systems (TAPOS), October 1998.