IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Hershey PA 17033-1117, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP4261

Modelling Web Application: The Conceptual Page Design

Leonardo Mangia and Roberto Paiano
SET-Lab, University of Lecce, Italy, Tel: (083) 232-0296, {mangia, rpaiano}@ingle01.unile.it

Vito Perrone
Hypermedia Open Center, Politecnico di Milano, Italy, perrone@elet.polimi.it

ABSTRACT

The main goal of this paper is to define a publishing model for Web Applications starting from the analysis of the most well-known modelling methodology, such as HDM, OOHDM, WebML, Conallen's method and others. The analysis has been focused to verify the state of art about the modelling of Web application pages. In particular, the different types of elements that compose the Web page in the above models are taken into consideration. This paper describes the evolution of the HDM methodology starting from the first approach based on the definition of a LP concept up to the more structured and complex Conceptual Page based on the influence of "operations" on the modelling of the dynamics of navigation between pages.

INTRODUCTION AND BACKGROUND

Design and develop of WWW application is quickly evolving to become a more engineered products introducing powerful models of hypermedia applications. The entire lifecycle to obtain affordable outcomes must be considered as a complex process that should be supported by tools in order to help the designer in each phase.

Starting from a conceptual modelling it makes easier to manage the changes but it requires a well-engineered process to correctly drive the entire cycle from the model to the outcomes.

Our research activity is oriented to develop both a Model to suite the complexity and a set of tools to support the designer from the analysis phase to a prototype of the Web application in order to an effective test of model. These tools, based on a relational database, support also the multi-delivery feature to customize the application according to the user role (families of applications). In the 1993 was published HDM (Hypermedia design Model) [1-3], the first modelling approach oriented to the design of multimedia application that was enhanced to support the hypermedia applications [4].

In this environment a relevant aspect is represented by definition of Logic and Presentation Pages. Logic pages have been introduced into the model to better design what the designer consider the unit of fruition of the specific WWW application (example the painter and all his works or the collection of all painters and so on). The Presentation pages are a collection of logic pages that appear to the user into the HTML Page (example the home page and the previous logic page can appear into two frames of a unique HTML page) managing the dynamic behaviour.

We developed the tools, starting from HDM concepts, to support the entire applications lifecycle through the prototyping using an engine that will be briefly described in the section 2.

In the last months the Web environment is being oriented on the Web applications more than Web sites so the HDM model is evolving to its 2000 version (W2000) in order to best capture all the dynamic and navigational behaviour of WWW applications.

Adding the operation to the traditional Web sites means that the model structure and the behaviour may change dynamically and are strictly related to the user profile.

Our research is oriented to define an Information Conceptual Model and a Navigation Conceptual Model by rendering the design concept using customized UML diagrams. UML was extended to define a suitable framework for this task.

The next step is the definition of a Conceptual Publishing Model, which is the core of this paper, which inherits the information and navigation definition and customize the user behaviour.

Several authors are publishing interesting ideas about this problem but they are starting from the page definition while we start from a Conceptual framework.[5-7]

Although for small application into a well-known application domain it's possible to design directly the Web page having in mind, without any formalism, the information and navigation structure, in order to realize a well-structured design it's needed to have a different approach to the design.

JWEB II NAVIGATION ENGINE: THE LOGIC PAGES APPROACH

The main requirements taken into account in the design and in the development of the JWeb II Navigation Engine (NE) are:

- Generation of hypermedia applications described in standard structures (easily maintainable);
- Management of different client devices (multi-delivering) such as Internet, WAP, CD-Rom, etc.;
- More different end-users, maintaining individual history for each session.

According to these requirements, the JWeb II NE design is structured in two different steps. In the first step the part of the application independent from the client device presentation technology (presentation-independent module) is generated, while, in the next step, the other part that dependent from this technology (presentation-dependent module) is built. So the generation of hypermedia applications consist in the creation of the physical pages (presentation-dependent) that will be published directly on the client device and LPs, (presentation-independent) which contain all the necessary information for the creation of the physical page. This distinction allows the complete reuse of the presentation-independent modules that represent the main design and development effort, while the presentation-dependent module must be edited according the particular presentation technology (WAP-WML, HTML, etc.).

In details the logical pages structured as in Fig.1, pick up and join information about

- item (one or more) selected (ex. an artist);
- application link for the selected item (ex. the artist's works);
- active collection (actual showed element position, link to the next and previous elements, etc.);
- centres of other collection (ex. "web sections" collection);
- user navigation (back, user-history, etc.).

The LP contains the information in standard and abstract form, independent from the user device presentation technology. In JWebII NE the LPs are XML files with a specific DTD; these LPs are built according to a specific template and they pick up all the information needed to create the physical page respecting the end-user request.

The figure 2 shows a web page of an e-commerce application that describes the information about a particular model of shirt, its relative

This paper appears in *Issues and Trends of Information Technology Management in Contemporary Organizations,* the proceedings of the Information Resources Management Association International Conference.

Figure 1: Logical page structure

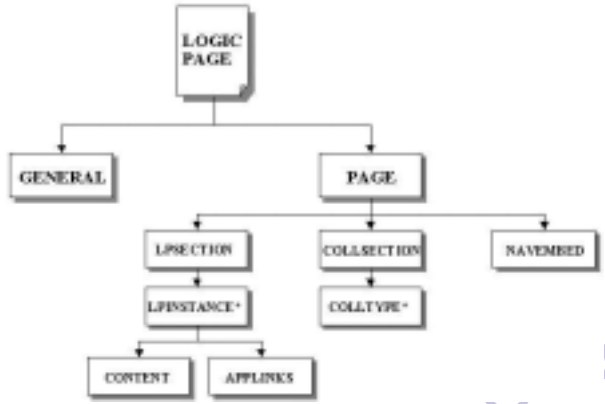

Figure 2: The logic pages

application links, the active collection (all the models with available filters) and a collection (the web application 's sections).

This page has information present in four different LP, respectively:

- LP 1: information about support functionality;
- LP 2: information about active "models" collection;
- LP 3: information about "web sections" collection;
- LP 4: information about selected "models" and its relative application links.

The LPs are composed together in presentation level without a precise design methodology.

This LP approach to model Web Pages was used to realize an e-commerce application in a European project [4]. This experience has underlined that the approach is efficient to model Web application for a run-time navigation engine but some problems was undefined:

- This approach is presentation oriented and not a result of a publishing model;
- The definition of the interaction between logic pages in the same physical page;
- What happen when the "operations" change the contents of a logic page.

W2000: THE CONCEPTUAL PAGES APPROACH

On the basis of the aforementioned open issues on the LPs approach, it's clear that this kind of modelling approach is quite effective for modelling pages to be used by the engine previously described. On the other hand, taking into account our experience in web application modelling, we realise that it is rather weak for modelling pages from the user point of view, capturing the user experience already in the modelling phase. For this reason we introduced the conceptual pages modelling that focuses on modelling web pages just as these will be perceived by users, introducing a user centre semantic in defining the various aspects of the pages, modelling the interactions among pages and within the page. Basing on this definition, the conceptual pages modelling could be located upstream the LPs modelling (opportunely revisited) for that applications that will be executed by the JWeb engine.

Moreover, the model aims to reach the following objectives:

- To define a self-standing model: to allow the designer to directly model the Web pages starting from the requirements;
- To define a model that could become part of a more complex methodology such as HDM and its evolution.

The first objective concerns that, from the analysis of other modelling techniques, don't exit methodologies supporting the directly modelling, in a user centred way, of the Web Pages while this requirement is one of the most important to model small and well-known applications using a structured approach. In this scenario, it's very important to manage both the complexity of the structural composition of the pages (generally the pages join several different elements) and the navigation dynamics between pages. Furthermore, using a structured approach, it's possible to reuse both the model and the multimedia elements in a framework-like environment.

The second objective concerns the need to complete the model W2000 in order to have a complete methodology to design Web Applications.

W2000 model has two distinct levels:

- *Information Design:* where the *information structures* are designed. Information structures represent the contents available to the application user.
- *Navigation Design:* where the *navigation structures* are designed. Navigation structures re-organize contents for a navigational purpose. They define which information elements are accessible by other information elements.

Like the above levels, the conceptual pages model deeply uses the notion of "type" for all its elements, allowing a more flexible page structure to better reuse components and contents. Furthermore, to satisfy the requirement to complete the W2000 model, some guidelines and rules are defined to map the Navigation level to the Page level. We have already said that a relevant aspect of the model is the possibility to model the "user experience" realized by the introduction of a primitive to organize information and navigational features needs to the user to perform a certain task in a specific context. For example some task could be: "browsing items in own shopping bag", "scrolling the pictures of a painter", "exploring all book information", etc.

The Model

The Conceptual Pages Model describes the organization of information content of a hypermedia application into pages. Particularly:

- It identifies the basic elements for presentation, i.e. PUs (PU)
- It organizes them into sections and pages
- It organizes the navigation within and across pages.

It has several primitives defined to describe the properties of the publishing model, the PUs, the connections between these units and the grouping of the publishing elements.

Because the available space, we'll only explain the concepts. A reader can use these concepts with whichever notation, for example it could be used an UML-like notation. [8].

Publishing Unit Type

A PU is the atomic element within the page structure. It's a set of information content shown to the user as presentation unit. All the contents of a PU should be perceived by the user as a "consistent" portion of information (in a multimedia meaning) on the page.

Referring to a navigational model, a PU can be defined using one of the following methods:

1. We derive the PU from a whole node defined in the navigation model (according to the widespread web application models, a node defines the *elementary granules* of information from /to which user can navigate); the content of the node is "presented" to the user into one unit. This unit may inherit links and navigation features from the node from which it is derived.
2. We derive the PU from a part of a node, defined in the navigation model; the content of the node can be "presented" to the user into more units organized in a *Publishing Cluster* (see later in this paragraph). These units may inherit links and navigation features from the node from which they are derived but they also may have *pure links* (see later in this paragraph), which make the navigation across them possible.
3. The PU does not derive from any node. In this case, it represents a special element of the page, such as site "logo" or copyright information.

A *mapping rule* describes the way to derive a PU from a node or picking up the information from a database.

A PU has the following properties (Table 1).

Table 1: PU properties

Property	Description
Name	It univocally identifies the PU
Mapping Rule	It specifies the way through the content of a PU is derived from a node or picked up from a database or another information model.
Comment	It is an informal description of the PU.

Basing on the kind of information that the PU models, we can have:

- **Publishing Unit Type:** it describes the common structure, properties and features of a class of units; it is derived from a node type.
- **Single Publishing Unit:** it describes the structure, the properties and the features of an individual unit. A single PU is derived from a single node or it is not derived from any node; in the last case, it represents special content (logo, copyright information and so forth, index of high level collections).

In the following figure the PUs contained in a page of our example site are shown.

Figure 3: PUs in a Meltin Pot's page. 1, 2 and 5 are single PUs, while 3, 4 and 6 are PU types

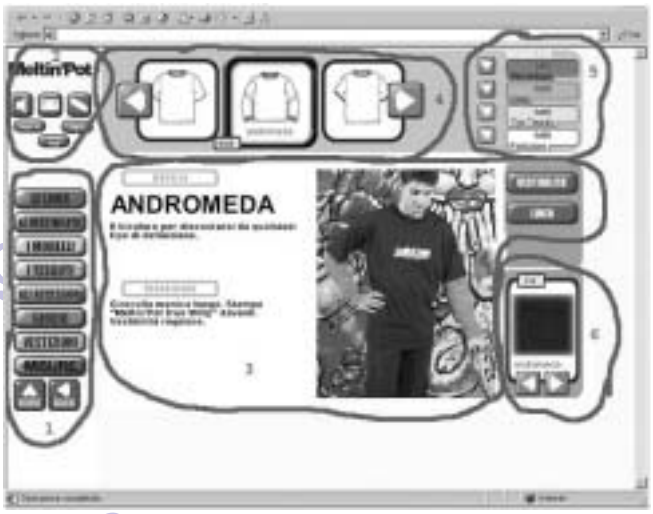

Figure 4: PCs in a Meltin Pot's page

Publishing Cluster Type

The publishing cluster (PC) represents the way to put together the information and the navigational features performed by a user during a task. It groups PUs and links. The relationship between a PC and its PUs is an aggregation. The designer may define navigation across the PUs in the PC and among PUs belonging to different PCs.

For each PC, we define its *default entrance unit*; it is the unit from which the user starts "consuming" the content associated with the cluster.

A PC has the following properties (Table 2).

Basing on the kind of PUs belonging to the PC, we can have:

- **Publishing Cluster Type:** it has got at least a PU type.
- **Single Publishing Cluster:** it has got only single PUs.

In the following figure two PC types are shown. The PC 1 corresponds to the user task "to explore the collection", while the second one corresponds to the user task "to look the details of a particular article"

Page Section Type

A page section is a grouping of content, semantically correlated. It is defined in order to provide the pages of the application with a better organization. A section put together PC semantically correlated. When a user goes from a cluster to another within the same section, he doesn't change his task, but he is changing context performing the same task.

Table 2: PC properties

Property	Description
Name	It univocally identifies the PC.
Comment	It is an informal description of the PC purpose.

For each section, we need to specify its *default entrance*; it is the publishing element (PU or PC) from which the user starts "consuming" the content associated with a section.

A section has the following properties (Table 3).

Basing on the kind of PCs belonging to the page section, we can have:

- **Page Section Type:** it has got at least a PC type.
- **Single Page Section:** it has got only single PCs.

The following figure shows a page of our example with three sections, two of which contain a PC (Section 1 and 2) and one containing two PC (section 3). The section 3 put together two PCs because they are related. In fact when a user selects an article in the upper cluster, its details are shown in the bottom cluster.

Page Type

A page represents what the user see in browser windows. A similar characterization could be done on WAP and PDA. In the model a page

Table 3: Section properties

Property	Description
Name	It univocally identifies the section.
Comment	It is an informal description of the section.

Figure 5: Page sections in a Meltin Pot's page

is a grouping of different sections, which could not be correlated semantically. The same sections may be used in more than one page.

For each page we need to specify its *entrance section*, being the section from which the user starts "consuming" the content associated with the page.

A page has the following properties (Table 4).

Table 4: Page properties

Property	Description
Name	It univocally identifies the page.
Comment	It is an informal description of the page.

Publishing Link

In order to model navigation at publishing level, we redefine the notion of link in more general terms, to denote a general user interaction on a page. Most of interactions are induced and constrained by the navigation design upon which publishing design is built, but we also allow designers to introduce new interactions at this level, for efficiency or usability reasons.

If the designer is using the model below a navigational model, we may distinguish two kinds of links: *derived link* if it derives from a link defined in the navigation model, *pure link* if it is not derived from any link. The pure links are introduced in presentation to add further navigation possibilities, so allowing moving the focus of the page on a particular unit o to navigate towards a particular page.

A publishing link is a connection between two publishing elements, which makes the navigation across them possible. The source of a publishing link is always either a PU type or a single PU; the kind of the target of a publishing link depends on the kind of the link.

The navigation dynamics, lead to the following kinds of link:

1. **Focus Link:** this link allows to move the focus of the page from a PU (o PC) to another one maintaining the current content of the page at the same time. Following this kind of link, we navigate within the same page instance. The target of this kind of link can be one of the following publishing elements: PU (single, type), PC (single, type), page section (single, type).
2. **Intra-Page Link:** this link allows navigating across the instances of the same page type. Following this kind of link results in the change of the page content, but not the change of the page structure. This link could or not involve a focus change. The target of this kind of link can be one of the following publishing elements: PU (single, type), PC (single, type), page section (single, type).
3. **Page Link:** this link allows navigating across the instances of different page types. Following this kind of link results in the change of the page content and structure. The target of this kind of link can be one of the following publishing elements:
   - single page,
   - page type.

   A publishing link has the following properties (Table 5).

Table 5: Link properties

Property	Description
Name	It univocally identifies the link. It is usually the name of the relationship from which the link is derived.
Link Place-Holder	It describes the placeholder of the link, that is, the point on the page where the user should "click" to follow the link.
Population Criteria	It specifies the population rule used to define the content of the target, e.g., the PU type (or single PU) content belonging to the target.
(Optional) Focus	It specifies if the link involves a focus moving.
(Optional) Navigation Pattern	It models the meaning of instantiating a link.
Comment	It is an informal description of the link.

CONCLUSIONS AND FUTURE WORK

The Model will be completed at the end of this year and it will be used to define a design framework oriented to the bank environment in a European project. At the end of this year another European project in the environmental application domain will start allowing us to verify the model in a different situation.

REFERENCES

[1] F. Garzotto, P. Paolini, and D. Schwabe, "HDM - A Model Based Approach to Hypermedia Application Design", ACM Transactions on Information Systems, Vol. 11, N. 1, pp. 1-26, Jan. 1993.

[2] F. Garzotto, L. Mainetti, and P. Paolini, "Information Reuse in Hypermedia Applications", Hypertext 1996, Washington DC, USA, pp. 93-104

[3] F. Garzotto, L. Mainetti, and P. Paolini, "Hypermedia Application Design: a Structured Approach", in J.W. Schuler, N. Hannemann, and N. Streitz Eds., "Designing User Interfaces for Hypermedia", Springer Verlag, 1995.

[4] M. A. Bochicchio,R. Paiano, P. Paolini " JWeb: an HDM Environment for fast development of Web Applications" - ", Proceedings of Multimedia Computing and Systems 1999 (IEEE ICMCS '99), Vol.2 pp.809-813

[5] Jim Conallen : "Modelling Web Application Architectures with UML", Communication of the ACM, Oct. 1999

[6] S. Ceri, P. Fraternali, A. Bongio: "Web Modeling Language (WebML): a modeling language for designing Web sites", to appear on Proc. Int. Conf. WWW9, Amsterdam, May 5 2000

[7] D. Schwabe, G. Rossi, "An Object Oriented Approach to Web-Based Application Design" , Giugno 2000. http://www.telemidia.puc-rio.br/oohdm/oohdm.htm.

[8] V.Perrone, M. Maritati, P. Paolini, L. Baresi, F. Garzotto, L. Mainetti: "Hypermedia and Operation Design: Model, Notation and Tool Architecture". Official Deliverable D7 of the Europen Project UWA IST2000-25131.