# MESH Object-Oriented Hypermedia Framework

**Wilfried Lemahieu**
*Katholieke Universiteit Leuven, Belgium*

## INTRODUCTION

Hypermedia systems represent data as a network of nodes, interconnected by links. The information embodied within the nodes can be accessed by means of navigation along the links, whereby a user's current position in the information space determines which information can be accessed in the next navigation step. This property of navigational data access raises hypermedia systems as utterly suitable to support user-driven exploration and learning. The user autonomously determines the way in which he or she will delve into the information, instead of being confined to the rigid "linear" structure of, for example, pages in a book.

Although the World Wide Web contributed tremendously to the popularity of the hypermedia paradigm, it also amply illustrated its two primary weaknesses: the problem of user disorientation and the difficulty of maintaining the hyperbase. Indeed, whereas non-linear navigation is a very powerful concept in allowing the end user to choose his or her own strategy in discovering an information space, the resulting navigational freedom may easily lead to cognitive overhead and disorientation. The latter is called the "lost in hyperspace" phenomenon. The user becomes unable to assess his or her current position and sort out his/her navigational options. Equally stringent is the maintenance problem. Due to the absence of workable abstractions, many hypermedia systems implement links as direct references to the target node's physical location (e.g., the URL in a Web environment). To make things worse, these references are embedded within the content of a link's source node. As a result, hyperbase maintenance becomes a synonym for manually editing the nodes' content. Moreover, the inability to enforce integrity constraints and submit the network structure to consistency and completeness checks results in a hyperbase with plenty of dangling links. Needless to say that the consequences of inferior maintenance will also frustrate the end user and effect additional orientation problems.

## BACKGROUND: A STRUCTURED APPROACH TO HYPERMEDIA AUTHORING AND NAVIGATION

This article introduces the *MESH* hypermedia framework, an acronym for ***M**aintainable*, ***E**nd user friendly*, ***S**tructured*, ***H**ypermedia*. So as to improve maintainability and facilitate end-user orientation, a structured approach is taken towards both authoring and navigation. MESH's fundaments are a solid underlying data model and a context-based navigation paradigm.

The benefits of data modeling abstractions to both orientation and maintainability were already acknowledged in Halasz (1988). They yield richer domain knowledge specifications and more expressive querying. Typed nodes and links offer increased consistency in both node layout and link structure (Knopik & Bapat, 1994). Higher-order information units and perceivable equivalencies (both on a conceptual and a layout level) greatly improve orientation (Thüring et al., 1995). Moreover, the separation of node content from data structure, along with the introduction of a dedicated link storage facility, allows for improved maintainability (Garzotto et al., 1993). Semantic constraints and consistency can be enforced (Ashman et al., 1997), tool-based development is facilitated and reuse is encouraged (Nanard & Nanard, 1995).

Consequently, hypermedia design is to be based on a firm conceptual data model. The pioneering conceptual hypermedia modeling approaches such as HDM (Garzotto et al., 1993) and RMM (Isakowitz et al., 1998) were based on the entity-relationship paradigm. Object-oriented techniques were mainly applied in hypermedia engines to model functional behavior of an application's components, for example Microcosm (Davis et al., 1992) and Hyperform (Wiil & Leggett, 1997). Along with OOHDM (Rossi et al., 2000) and WebML (Ceri et al., 2000), MESH is the first approach where modeling of the application domain is fully accomplished through the object-oriented paradigm.

MESH couples general-purpose object-oriented modeling abstractions to proprietary hypermedia concepts to provide for a formal hypermedia data model. Moreover, its context-based navigation paradigm builds upon the data model to reconcile navigational freedom with nested, dynamically created guided tours, which are to provide a disoriented end user with sequential paths as guidance. This article presents the two elementary concepts behind MESH, its data model and the context-based navigation paradigm, and proposes an implementation framework. As this is only an overview, the discussion is kept very brief. A more elaborate description of the data model and navigation paradigm can be found in Lemahieu (2003) and Lemahieu (2002) respectively.

## THE MESH HYPERMEDIA FRAMEWORK

MESH's data model is looked upon as a hierarchy of node types. Each node is defined as an instance of exactly one "most specific type," which remains unchanged during the whole of the node's lifetime. All node types are assorted in an inheritance tree, each node type being compliant with the definition of its parent. Visual properties for a certain node type are abstracted in layout templates. These can be inherited and refined throughout the node type hierarchy. In this way, node instances representing similar real-world objects bear a similar, recognizable layout. In addition to the primary node classification criterion provided by the static node typing hierarchy, the aspect construct offers a means to attribute specific combinations of link and layout properties to certain sets of nodes. At run-time, a node's features can be altered by adding or removing aspects. As such, aspects provide secondary node classification criteria, which are independent of the node typing hierarchy and which, unlike a node's type, are allowed to be non-disjoint and variable over time.

So as to characterize node interaction, a node type is equipped with a set of link types. These classify the links according to the semantic interpretation of the relations they embody. Link type properties include domain, minimum cardinality, maximum cardinality and inverse link type. These can be seen as constraints imposed upon the respective links, in order to be feasible instances of their type. If a link type is inherited by a child node type, these constraints can be overridden, but may never be weaker than the ones imposed at the parent level. A final notable feature of MESH's data model is the concept of link subtyping. In common data modeling literature, subtyping is invariably applied to objects, never to object interrelations. However, in MESH, link types are full-fledged abstract types, which can be equally subject to subtyping. A child link type's instances are a subset of its parent's instances. It models a relation that is more specific than the one modeled by the parent, possibly reflected in more stringent constraints. Link types are deemed extremely important, as they not only enforce semantic constraints but also provide the key to context-based navigation.

An example is show in Figure 1: the node type artist has two aspects: sculptor and painter, denoted by a special attribute (a so-called aspect descriptor) discipline. The node Michelangelo is an instance of artist. Moreover, it has both the values "sculpting" and "painting" for its discipline aspect descriptor; hence Michelangelo has two aspects: a painter aspect and a sculptor aspect. Each such aspect contains discipline-specific properties of the artist Michelangelo. The artist Rodin only has the value "sculpting" for discipline; hence he only has a sculptor aspect. Van Gogh only has a painter aspect. An artist is linked to his artwork in general through a has-made link type. A sculptor is linked to his sculptures through a has-sculpted link type. A painter is linked to his paintings through a has-painted link type. This is an example of link subtyping: both has-sculpted and has-painted are subtypes of has-made. Rodin and Van Gogh only have has-sculpted and respectively has-painted links. Michelangelo has both: the paintings made by Michelangelo, denoted as Michelangelo.has-painted, will be a subset of all of Michelangelo's artwork, denoted as Michelangelo.has-made. We can say that Michelangelo.has made is the union of Michelangelo.has-sculpted and Michelangelo.has-painted. Please refer to Lemahieu (2003) for more details.

The navigation paradigm as presented in MESH combines set-based navigation principles (Kappe, 1999) with the advantages of typed links and a structured data model. The typed links allow for a generalization of the guided tour construct. The latter is defined as a linear structure that eases the burden placed on the reader, hence reducing cognitive overhead and, consequently, disorientation. As opposed to conventional static guided tour implementations, MESH allows for guided tours to be generated at run-time, depending on the context of a user's navigation. Such context is derived from the type of previously followed links. The context-based navigation paradigm defines navigation in two orthogonal dimensions: on the one hand, navigation within the current tour yields linear access to nodes related to the user's current focus of interest. On the other hand, navigation orthogonal to a current tour, changing the context of the user's information requirements, offers the navigational freedom that is the trademark of hypertext systems. In addition, the typed links sustain the generation of very compact overviews and maps of complete navigation sessions. An example of such overview is provided in Figure 2. The latter represents multiple guided tours, nested in one another. The "outer" tour Paris.Museums consists of all museums in Paris. The "current node" in that tour, that is, the museum currently visited, is the