Discussions at the Data Border : From Generalised Hypertext to Structural Computing

David E. Millard

Intelligence, Agents, Multimedia, Dept. of Electronics and Computer Science, University of Southampton, U. K. dem@ecs.soton.ac.uk

Abstract

Structural Computing grew from the trend in hypertext research towards generalised systems, it asserts the primacy of structure over data. As a philosophy it has been compared to Structuralism in anthropology and linguistics and has given birth to a new trend in systems design known as Multiple Open Services (MOS). The Fundamental Open Hypermedia Model (FOHM) is an alternative approach to Generalised Hypertext that views the various Hypertext domains as continuous rather than discrete. Its relationships to Structural Computing, Structuralism and MOS have never been fully explored.

This paper examines these relationships. We explore how FOHM might be implemented in MOS environments and describe the Data Border, the point where Structure meets Data. We then use this to explore how FOHM and Generalised Hypermedia are related to Structuralism and Structural Computing.

Key words: Structural Computing, Generalised Hypertext, Multiple Open Services, FOHM

1 Introduction

Structural Computing is an approach to service provision that grew out of the interoperability work of the Open Hypermedia Systems community. It first appeared as a concept in 1997 [1] and has since become a research area in its own right, with a series of workshops and several prototypical systems such as Construct [2] and Callimachus [3].

Despite its appeal as a methodology there remains a great deal of confusion between Structural Computing and the systems (and architectures) that implement it. In turn this has lead to ambiguity concerning whether work that does not rely on such architectures should be termed Structural Computing or not.

In particular the Fundamental Open Hypermedia Model (FOHM) [4] is another approach to hypertext interoperability that takes a different stance to systems such as Construct. Both support the notion of hypertext across multiple domains, which I would term Generalised Hypertext, but FOHM takes the approach that the distinctions between domains is artificial and that hypermedia is better seen as a continuous spectrum of different structures and behaviours. FOHM's single generalised model makes it easy to blur the boundaries between domains and experiment with new structures.

This paper examines the relationships between FOHM's brand of Generalised Hypertext, Multiple Open Service Architectures and Structuralist philosophy, in an attempt to more precisely position FOHM in relation to Structural Computing.

2 Background

Open Hypermedia has been an active research field for more than a decade. Hypertext systems such as Chimera [5], DHM [6], Microcosm [7], Multicard [8] and the HB/SP series [9] clearly separated hypertext structure (links) from content (documents). The First Workshop on Open Hypermedia was held in Edinburgh in conjunction with ECHT'94. The subsequent series of workshops lead to the formation of the Open Hypermedia Systems Working Group (OHSWG). One of the objectives of this group was to promote Open Hypermedia technology and work towards interoperability between Open Hypermedia Systems.

An initial proposal from Antoine Rizk was that the group could work towards producing a lightweight message based protocol that could be used to communicate about simple link service functions. The idea lead to the production of the first draft of the Open Hypermedia Protocol (OHP) [10].

The subsequent development of OHP is well-documented elsewhere [11–13]. It is enough to note here that the scope of the project changed from an attempt to provide a lightweight communication mechanism for shared clients and heterogeneous link servers, to an attempt to create a reference model and architecture for Open Hypermedia Systems.

A standardised data model and basic set of operations were agreed and two reference implementations of OHP were built and demonstrated at the ACM conferences on Hypertext in 1998 and 1999. During the development of these systems several problems with the scope of the original draft proposal became evident. In particular the understanding that the typical type of navigational hypermedia OHP was addressing was only one of many hypertext domains. As a result the protocol was split into several domain protocols, each domain dealing with a particular type of hypermedia. The original OHP was therefore renamed OHP-Navigational (OHP- Nav) and reduced in scope to deal exclusively with navigational (node/link) hypermedia. Other domains were envisaged such as Spatial Hypermedia [14] (OHP-Space) and Taxonomic Hypermedia [15] (OHP-Tax).

During the OHP development period a new approach to hypermedia research started to emerge. Structural Computing 'asserts the primacy of structure over data' [1]. As such it is concerned with looking at how structure can be discussed and managed at all levels of computing. In this world view, OHP-Nav is simply one protocol of many that facilitates different processes operating across structure, in this case the processing of navigational structure.

In OHP terms, different Structure Servers would serve the different structures from the various hypermedia domains [16]. Thus a Navigational Structure Server would be developed that provided an OHP-Nav interface to clients. Other OHSWG developed protocols, such as OHP-Space, would be supported via separate Structure Servers.

In a way early Structural Computing was a reaction to the need to support an extensible set of hypermedia structures, but in the last couple of years it has moved in scope beyond its hypertext roots and been promoted as a general approach to computing [17].

In the Hypertext field there remains a need to deal with structure and the research boundaries have naturally moved beyond the original domains tackled by OHP. An alternative approach to the deconstruction of hypermedia (as represented by Structural Computing) was presented in the Fundamental Open Hypermedia Model (FOHM) [4]. This took the data structures developed in OHP, generalised them, and separated them from a protocol implementation. This move was informed by separate agent and infrastructure research of the time, which demanded a vocabulary for discourse without any associated protocol or architectural assumptions.

3 FOHM

FOHM was developed because of three observations:

(1) The definition of a communications protocol or infrastructure is a separate research question to the definition of a vocabulary for hypertext.
(2) The OHP-Nav model is almost, although not quite, expressive enough to model other domains of hypertext, particularly Spatial and Taxonomic Hypertext.
(3) There are interesting structures that lie in the space between domains. These could be exploited by a general model.

By extending the model it was hoped to create a basic vocabulary with which components (computational agents) could discuss hypertext structure and with which researchers could explore new structures.

We know that there are structures that OHP-Nav does not support that seem intuitively Navigational, such as trails, concepts and tours. In addition, systems such as StorySpace [18] often make use of layout, map views and hierarchical organisation that seem inherently Spatial, yet they combine these with Navigational ideas of linking.

Although there have never been any claims that FOHM is capable of modelling all possible domains, it is certainly capable of expressing more than the traditional three described by the OHSWG. By developing tools that use FOHM it becomes easy to experiment with new hypermedia structures as well as explore variants on the ‘core’ structures (such as real-world Links [19]).

The philosophy behind FOHM questions the move to view the domains of hypertext as separate and exclusive. Not just from a systems design point of view, but from a hypertext theory perspective. After all, what is a domain of hypermedia? A particular set of structures, a particular set of behaviours or a new paradigm of authorship, readership and use?

FOHM allows us to easily define new types of structure and explore the boundaries of the domains, facilitating functional re-use (such as storage, scoping and context) across a consistent representation of hyperstructure.

3.1 Generalising Hypertext

It has already been mentioned that the OHP-Nav model was almost capable of expressing all three domains of hypermedia and that it was this observation that led to thinking about FOHM. It should therefore come as no surprise that FOHM looks very similar to the model that lay at the heart of the OHP-Nav protocol.

The basic FOHM model is constructed from four core objects (which were given different names to distinguish them from their OHP origins):

- *Associations*: represent relationships between Data objects and Associations.
- *Data* objects: wrappers for any piece of data held outside of the model.
- *References*: which are used to point at (or into) Data objects and Associations.
- *Bindings*: which are used to attach the References to the Association structure.

Figure 1 shows two basic navigational links, which share a Reference, expressed in FOHM :

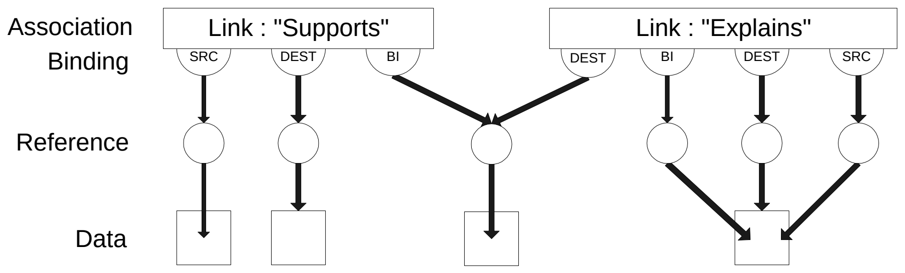

Fig. 1. Two Navigational link in FOHM.

There are two important differences between FOHM and the last version of the OHP-Nav model (OHPNAV-1.3-Darmstadt [20])

(1) In OHP-Nav members of a Link bind to it with very specific semantics, they are either sources or destinations of the link (or both). In FOHM the Link structure has been generalised into a typed Association (a Link is therefore just one type of Association). The ‘direction’ attribute has gone, replaced by a feature-space (a list of attributes to which each Binding must give a value). Each Binding then defines a feature-vector, effectively describing its position in that space and therefore its reason for being bound to this Association.

 For example, in an Association with the type ‘Link’ the feature space has only one feature, ‘direction’, to which each Binding must give a value (as either ‘source’, ‘dest’ or ‘bi’).

(2) In OHP-Nav a Context object was expressed as a collection of object id’s. Although these objects could share a set of characteristics, descriptions and presentation specifiers, they effectively represented a linkbase, which is a very static way of scoping a set of structures.

 In FOHM a Context object is a descriptive device that defines in which context a given part of a FOHM structure is visible. Thus a Link Structure might have many Context objects attached to it at different points, each one describing in which context the link, one of its members, or the membership of one of those members, is visible.

 Since parts of a FOHM structure may ‘disappear’ depending on the context of the viewer (the stated context of the querying component), FOHM also contains rules which describe how hyperstructures collapse gracefully to compensate for missing objects.

 In this way FOHM supports a more sophisticated partitioning of the hyperstructure than mere linkbases, where different views of a single contextual linkbase are generated dynamically according to the context of the viewer.

These differences allow FOHM to model the various domains. The extended binding semantics allow the model to express different kinds of Links, Spaces and Taxonomic Categories in a consistent manner. The contextual views of the hyperstructure allow the model to express Taxonomies that branch according to the perspectives of their authors.

The most immediate effect of cross-domain fertilisation is that the contextual mechanism becomes integral to all three domains. This creates sophisticated contextual Open Hypermedia that can be used to formally express many of the techniques of existing Adaptive Hypermedia Systems [21].

3.2 Implementations

Because FOHM does not define any communication mechanism it must be bound to one before it can be used. It is important here to note that by communication mechanism we actually mean not only the means of communication (for example TCP/IP sockets or Java RMI) but also the *method* of communication. This might be as simple as a set of defined operations or as complex as an agent 'language'. To date there have been three different implementations of FOHM that have explored this in different ways. We present them here as exemplars of why a hypermedia model unbound to a specific protocol and set of operations is desirable.

3.2.1 SoFAR FOHMServer

The Southampton Framework for Agent Research (SoFAR) [22] is a multi-agent framework designed for Distributed Information Management (DIM) tasks. It is a communication infrastructure based around dynamically discovered components or agents. Communication in SoFAR is based on collections of related propositions known as ontologies. Each proposition in the ontology asserts a particular fact, and therefore is known as a predicate.

Communication is achieved by passing predicates between agents along with one of a set of performatives that show the sender's intention. These performatives allow agents to assert predicates as true or false, subscribe to such assertions, ask each other questions on the validity of predicate statements and to ask each other to return sets of statements where certain criteria are true.

In order for agents to discuss FOHM structures a FOHM ontology was defined, this included a predicate for each named FOHM object (Associations, References and Data items). Asserting a predicate asserts the existence of that structure.

A FOHMServer agent was written that stored a set of FOHM structures and responded to these performatives [23]. This was then used to provide the structures to Navigational and Spatial application agents (via the SoFAR communication mechanism).

3.2.2 RTSP Server

The FOHMServer SoFAR implementation used FOHM to describe both the Spatial and Navigational domains of hypermedia. The second implementation of FOHM concentrates on the Navigational side, this time binding the model to the Real Time Streaming Protocol (RTSP) to allow streaming of linkbases [24].

This system handles a request for a FOHM 'linkbase' at the beginning of a stream. This is then parsed at the client and the links presented as they occur in the streamed media.

At any point the client can attempt to create a link, this is handled as a command to the server via the RTSP stream, the server then creates an appropriate FOHM Association in the 'linkbase' at the server.

3.2.3 Auld Linky

The most recent implementation of FOHM is the Auld Linky structure server [25]. This is similar to the SoFAR FOHMServer but runs independently of any communication infrastructure as a single process that uses HTTP and an XML pattern matching language for communication.

Both of the previous implementations merely served the FOHM structure, but Auld Linky also implements the contextual culling process. Queries to the server can be made in context, the server will then filter the returned structures according to that context (checking the query context with the context objects attached to the structure and collapsing structural branches where appropriate).

In addition Auld Linky extends FOHM to include Behaviour objects. These are opaque to the Auld Linky implementation and are interpreted by clients. They allow hypermedia authors to specify how certain events, such as a link traversal, alter the users context.

Contextual filtering make it possible to perform *adaptable* hypermedia with FOHM, where the visible hyperstructure alters according to a users context. With Behaviour it becomes possible to implement *adaptive* hypermedia as well, in this case the user's context constantly evolves and the visible hyperstructure changes according to the user's actions.

Recently this has been used to explore narrative support in hypertexts [26] in a way that has been described by Bernstein as 'Sculptural Hypertext' [27], where links are selectively removed from view as opposed to selectively added.

4 FOHM and Structural Computing

A great deal of the work undertaken so far in Structural Computing is based around a software system known as Construct [2]. Construct is the code-base successor of HOSS [28], DHM [6] and HyperBase [29]. It is a CB-OHS [1] that attempts to provide an extensible open hypermedia platform based on the latest OHSWG standards.

In Construct the middleware layer is opened up into an extensible set of structure servers, an approach described by Wiil as Multiple Open Services (MOS) [30]. Wiil questions the relationship between MOS and Structural Computing, suggesting that MOS is in some way a generalisation of Structural Computing as MOS deals with arbitrary services, while Structural Computing could be seen to deal with structural (or even just hypermedia) services.

Nürnberg et al turn this around, arguing that Structural Computing views structural services not as a subset of services but as a conceptualisation of services in general [31]. Thus relegating MOS from a generalisation of Structural Computing to an infrastructure approach that can support Structural Computing.

Where then does FOHM fit into this picture? Firstly we shall look at FOHM as an Open Service and how it relates to Construct and the MOS infrastructure approach. We will then look at its relationship with Structural Computing philosophy.

4.1 FOHM as an Open Service

FOHM requires a binding to a communication model before it can be used. As FOHM places no restrictions on architecture it should be entirely possible to use FOHM within the Construct environment. There are several places where it could arguably be incorporated (see Figure 2).

(1) *As a Back-End API*: A FOHM communication protocol could be used by the Structure Servers and their common store to communicate.
(2) *As a Structure Server API*: A FOHM communication protocol could be supported by a dedicated Structure Server and provided on a par with more specific domain protocols.
(3) *As a Middleware Server API*: A FOHM Structure Server could act as a middleware service. Domain Structure Servers could translate the FOHM structures for specific applications and enforce any domain rules.

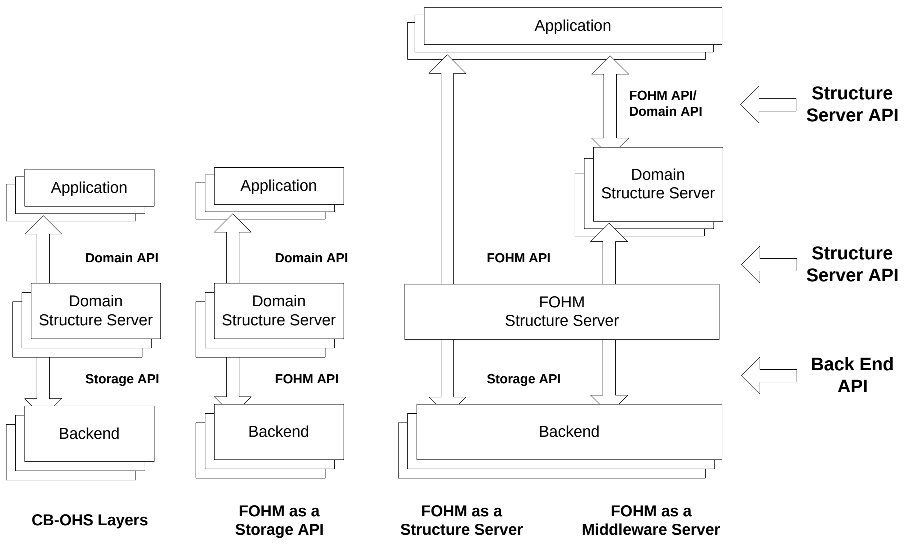

Fig. 2. Integrating FOHM with Construct

4.1.1 FOHM as a Storage API

Firstly we consider the idea of using FOHM as a storage model (back-end API). It is understandable to view FOHM in this way, as the philosophical successor to early HyperBase work [29], and to a certain extent FOHM stands in indirect support of that approach. However, in terms of Construct, the problem is that FOHM is specialised to deal with associations found in hypermedia structures (of all domains), while structure servers are envisaged to deal with arbitrary structure.

It has been claimed that typed associations (such as those in FOHM) are as expressive, and in some ways more expressive, then meta-data [32]. From this it could be argued that FOHM is capable of expressing arbitrary structure. However, FOHM was based on the premise that to be *practical* a model would have to represent the highest common structure across the domains considered. Only by representing the highest common structure can the performance penalties associated with over-abstraction be avoided. So although FOHM is capable of being used as a structure storage API it would not make a very practical choice unless the environment was exclusively a hypertext one.

4.1.2 FOHM as a Structure Server API

Secondly FOHM could replace specific domain protocols. Thus rather than a Navigational, Spatial and Taxonomic Structure Server we provide a single FOHM Structure Server.

This decision has both its advantages and disadvantages. The most obvious benefit is that it allows structures from the three domains to be mixed together. This kind of cross-domain browsing has been discussed in the literature before [33,17] and would allow different domains of hypertext to be used where a user felt most appropriate (for example using a Spatial metaphor for information triage and dynamic authoring and a Navigational one for browsing).

The disadvantage is that it requires all the clients to be aware of the possible domains. This does not actually mean that they need to understand the union of all the possible structures and values, but that they need to check that the structures that they are given lie within a subset of these that the client understands.

In fact some of the communication mechanisms that FOHM has been used with make this trivial. Both the SoFAR and Auld Linky implementations allow the software that uses them to specify which structures they are interested in. For example, in the SoFAR framework an agent can register that it understands a subset of structures expressed in FOHM. When another agent queries the system for agents that understand FOHM structures they can specify which type of structures they are interested in and will only receive a reference to the first agent if the structure types match.

4.1.3 FOHM as a Middleware Server API

An alternative to a straight Structure Server is to treat the FOHM Structure Server as a Middleware Server to be utilised by other Structure Servers. Domain servers could then perform the domain specific operations across the FOHM structures and ensure that single domain clients did not become confused by cross-domain structures.

FOHM fits well into this layered middleware design, where it forms an appropriate layer to perform operations that apply to many domains (such as the context culling process). In addition more sophisticated applications could access the FOHM Structure Server directly, meaning that easy experimentation with new structures is still possible.

Viewing FOHM as an alternative to several hypermedia domain servers is not a rebuke of MOS. The specific domain server approach is still valid, even in parallel with a FOHM structure server.

It is perhaps unfortunate that MOS, represented by Construct, emerged from the field of hypertext and therefore emphasised the multiple domains of hypertext as an example of the need for structural abstraction. FOHM represents an alternative to that example but not an argument against MOS itself.

5 Structuralism and Generalised Hypertext

Several comparisons have been made between Structural Computing and the philosophical approach known as 'Structuralism' [17,31]. This comparison is understandable, Structural Computing argues that computational services are better understood as structural operations, while Structuralism maintains that the world is better understood in terms of structural relations.

Structuralism emerged in the thinking of early twentieth century European linguists, most noticeably Saussure [34] who considered language in terms of two fundamental dimensions, *Langue* and *Parole*. Where Parole represents the instantiations of language we are familiar with (speech, writing etc.) while Langue is concerned with the 'hidden' relationships and rules that applies across all Parole.

Saussure is perhaps best known for his work on signs. A linguistic sign can be understood in terms of the relationship between signifier and signified, for example the word 'dog' is the signifier for the signified physical class 'dog'. The interesting aspect here is the arbitrary nature of this relationship (there is no logical reason we call a dog a 'dog'). Because of this it is possible to view language as a self contained system of structure, that is, as a system that defines itself. There is no meta-language that explains language, instead we are faced with a self-referential system where words explain other words.

Levi-Strauss found that this structure-first approach also had application in the field of anthropology [35]. The Parole in this case manifests itself in the variety of human cultures and the Langue in the hidden structures that unite them. This was effectively an attack on other types of philosophy that believe that there is a 'core' of truth that represents 'reality'. Instead Levi-Strauss promoted the notion that phenomena can only be understood when considered in relation to other phenomena, in other words there is no 'core' of truth, only the physical Parole to the cultural Langue.

Structuralism has broadened into the field of Semiology, the study of signs. Early questions concerning the relationship of Semiology to Hypertext concentrated on the notion of a source anchor as a signifier for a signified destination anchor [36]. It is Structural Computing and its insistence on viewing systems in terms of structural abstractions that really brings to the fore the question of how best to understand not only hyperstructure but also the structure of the *actual items being anchored*, the data of traditional systems.

Structuralist philosophy leads us to view data not as atomistic, but as sophisticated structure. For any given system, data is the name we give to structure that the system does not understand, either due to computational restrictions or by design choice.

In the same way that structuralism in anthropology opposes the view that there is a core of truth in the world, structuralism in computing would take the view that there is no core of data in computing. Instead it presents the view that there is only structure referencing other structure.

This is not to say that data is not a useful abstraction, it is fundamental for lots of very pragmatic reasons (storage, efficiency, etc.), but that we should be aware that the line where known structure ends and data begins, the Data Border, is an engineering decision and not an absolute one.

Consider as an example a hypertext system that attempts to store the structure of a typical novel. On one level there is the structure of the medium itself, the sequence of words, sentences, paragraphs and chapters. On another level there is the structure of the story as an abstract entity, a sequence of scenes, characters, dialog, etc. Then there is the structure of the subjects depicted in the novel, the relationships between characters, the unfolding of a plot, etc. Perhaps most complex of all there is the relationship between the author of the novel with the reader(s).

These multi layered structures are extremely complex and may be quite subtle and open to interpretation, hence the large body of literary criticism. Language, either written or recorded, is a powerful mechanism that is capable of embodying this structure. In other words the novel stored in the hypertext system is already richly structured, the real problem is not the lack of structure but the inability of modern computing systems to analyse and process that structure.

This has led to research into simple ‘knowledge structures’ that are easy for machines to interpret. The ontologies described for the SoFAR system above are a good example of these, where predicates are unambiguous statements of fact and performatives are unambiguous speech acts about predicates.

The same problem on the Web has resulted in Tim Berners-Lee’s Semantic Web initiative [37], a drive to add to the traditional content of the web with simply structured, machine parsable knowledge.

Hypertext as a technology is not designed to reduce human knowledge to a machine level for machine reasoning, but to augment that knowledge for human reasoning. This is evidenced by literary theorists embracing hypertext as an extension of the structures already found in language (for example [38]).

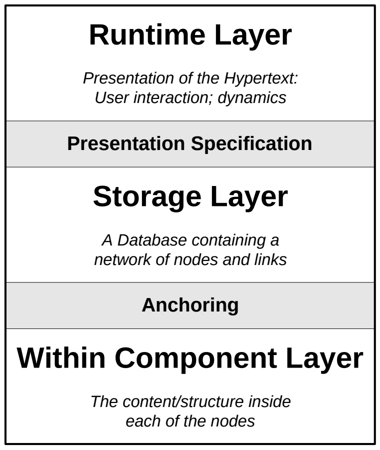
Runtime Layer
Presentation of the Hypertext: User interaction; dynamics
Presentation Specification
Storage Layer
A Database containing a network of nodes and links
Anchoring
Within Component Layer
The content/structure inside each of the nodes

Fig. 3. Dexter Hypertext Model

5.2 The Dexter Model

In December 1990 a workshop was held on hypertext standardisation at which various models of hypertext were discussed. One of the results of this workshop was the ‘Dexter Reference Model’ [39] the goal of which was to allow easy comparison between systems and thus work towards interoperability standards.

We can use this model to describe where Hypertext systems draw the Data Border and how they reference across it. Figure 3 shows how the model divides systems into three layers. The Runtime Layer contains all the facilities a user requires for constructing and browsing a hypertext. The Storage Layer represents the actual structures of the hypertext (e.g. links and nodes). The Within-Component Layer represents the content of the particular node or document.

There are also two interfaces within the model. The Presentation Specifications Interface lies between the Runtime and Storage layers. It represents information on how the runtime layer is to process (represent) the objects in the storage layer. The Anchoring Interface is a mechanism for addressing locations or items within the content of an individual component. This maintains a clean separation between the Storage and Within Component layers.

Although the Dexter model has been criticised for failing to address some of the requirements of large-scale distributed hypermedia [40], and also embedded links [41], this fundamental distinction of three layers remains useful.

When constructing a hypermedia system the decision of where to draw the Data Border relates to the Anchoring Interface in Dexter. The Data which lies on the far side of the Border belongs to the Within-Component Layer, while the structure that is understood belongs to the Storage Layer. The way in which the structure references the ‘opaque’ data is thus defined by the Anchoring Interface.

6 FOHM and the Data Border

FOHM acknowledges structure in three ways [33]:

- *Explicit External*, the relationships between objects, embodied by the FOHM notion of an Association (for example a Spatial List).
- *Implicit External*, the relationships between objects in a given Association (for example relative positions in a list).
- *Internal*, the relationships between implicit objects inside Data objects (for example paragraphs of a text).

FOHM uses a traditional Navigational Anchoring mechanism, which in FOHM is called a Reference, to allow *External* structure to refer to *Internal* structure. In effect the Reference embodies FOHM's Anchoring Interface, allowing portions of Data objects (the Within-Component Layer) to be referenced by the hyper-structure (the Storage Layer).

FOHM also has a notion of *Implicit External* structure. This stems from the fact that FOHM needs to represent Spatial Hypertext Spaces, which not only have structure themselves but also may have fuzzy members (i.e. some items are more a member of the Association then others).

Now let us consider our previous observation that while a given component views the content of a Data object as opaque that object is actually richly structured, if only the component contained the understanding to parse that structure. In FOHM opaque data 'blobs' are wrapped in Data objects, these are structural elements that are understood by FOHM components.

There is an interesting analogy between the Reference object, which effectively ties itself to a part of the opaque structure of the Data object, and the Binding object, which ties itself to a part of the visible structure in the Association object.

In effect we have two structural objects, Associations which we expect generic structure servers to understand and Data objects which we don't, and we attach these structural objects together with a Binding-Reference pair. Thus we end up with the Structuralist notion of structure referencing structure as a self-contained whole.

So we have come full circle. Hypertext is concerned with augmenting information for human beings. It does this by creating structure external to the data that people work with (hyper-structure), it is possible to view this as simple structure (parsable by generic components) and the original data as complex structure (parsable by specific components and in some cases only by human beings). The Data Border is simply the line we draw between these which effectively determines which part of the structural whole given parts of the system will understand, and which they will treat as opaque.

6.1 Placing the Data Border

Most hypermedia systems place the Data Border at a file level, such that a Data object would wrap an individual file or data stream. This is a convenient location as files tend to represent complex conceptual ‘chunks’ of information. Anchors allow a finer granularity of reference but do not replace the structures that lie within the data itself.

If we take the perspective that everything is structure, but temper that with the pragmatic view that not all structure should be (or can be) explicit, we are left with an interesting possibility. We could move the Data Border further towards the complex hidden structures in the data, thus revealing the simpler structures for our structural systems to manipulate.

In fact one of the first conceptual hypermedia systems did exactly that. Xanadu conceived by Ted Nelson in the 1960’s (described in ‘Literary Machines’ [42]) used the concept of transclusions. Documents were not stored as binary ‘blobs’, instead text was stored on a universal ‘permascroll’ and documents were simply collections of references to that permanent record. In this case the Data Border has moved to reveal the sequence of atomic fragments (and their sources) that make up a document.

Sequence is one of the most obvious structures that is normally internal to Data that it can be useful to reveal. One of our experiments with Auld Linky has taken advantage of this.

Documents are broken down into a set of ordered fragments. These fragments are placed into a FOHM Association that represents a transclusive tour (analogous to a Xanadu document made up of references). A servlet retrieves the tours on command and parses the FOHM structure building the (HTML) document dynamically.

Because FOHM supports context, and Auld Linky the filtering of structure via context, we can support what is effectively conditional transclusions [43]. Figure 4 shows the data structure we used. In this case the members of the transclusive tour are not data fragments but sets of data fragments. We can either use a concept (the semantics of transclusion dictate that the servlet picks one of the members) or we can use a Level of Detail (LoD) structure (this is similar to a concept except that the members are ordered according to their level of detail, the servlet can therefore make a choice about how much detail it wants to transclude).

When a user visits the web page they select the context in which they want to view the document (actually a list of weighted preferences describing their interests),

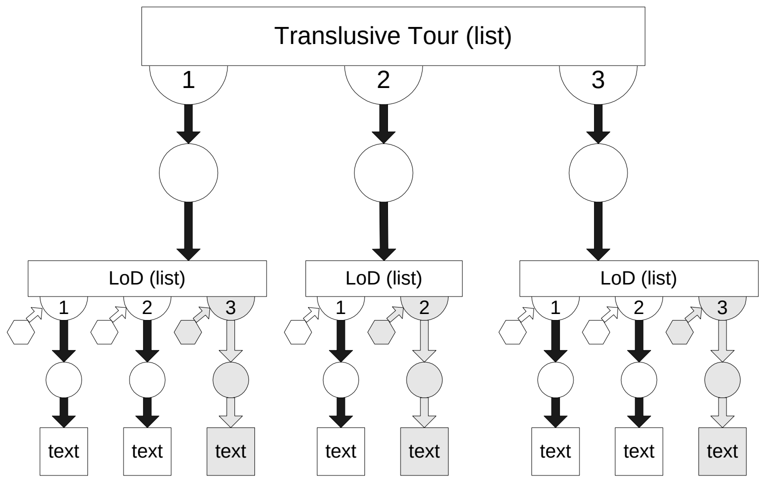

Fig. 4. A Transclusive Tour of LoD Structures

this is then used to query for the structure, some parts of which are subsequently culled, providing a dynamic page that is tailored to the users interests.

Since in different contexts an id can be resolved into either a Data object or an Association, FOHM actually allows *dynamic* placing of the Data Border according to context. I.e. in some contexts the first segment of the transclusive tour might be a Data object while in others it might resolve to be a second transclusive sub-tour.

6.2 Narrative Sequencing

Moving the Data Border and exposing the structure of documents in this way appears to be a very desirable *modus operandi* for modern hypermedia systems. Such systems, often grouped under the name Adaptive Hypermedia [44,45], attempt to modify the hyperstructure and contents that users see according to a user model and the users previous navigational choices. Although the application domain is maturing there are, as of yet, only a few attempts at formalising the models behind such systems [46,47], it seems that the OHS idiom, generalised in the spirit of Structural Computing, might fill that gap [21].

Generalised Hypertext also suggests that, beyond basic sequencing and preconditions, it might prove valuable to expose the more complex relationships that lie hidden within documents. This would allow more sophisticated *Narrative Sequencing* which takes into account features of media such as Narrator, Voice, Focalisation and other traditional tools of literary study.

7 Conclusions

As we move into the ‘post-Web Revolution’ era of Hypertext, research efforts seem to be concentrated around three broad topics. Firstly we have a growing body of Critical Theory work, secondly there is the (web-driven) development of Adaptive Hypermedia systems, and thirdly there is the OHSWG line of work concerned with modeling Hypertext for interoperability.

This last effort began with the development of OHP, grew into the realisation of multiple domains of hypertext and has resulted in the development of FOHM, MOS and Structural Computing.

It has taken several years for the definition of Structural Computing to become clear, in the past it has been confused with the Construct System, MOS in general and the philosophy of Structuralism itself. It is now clear that Structural Computing does not make the grand claims of early Structuralists (of discovering the ‘laws’ of structure) but is instead a method of approaching service provision that states that any service is better seen as a structural service.

Thus MOS is an infrastructure approach to support Structural Computing but it is not synonymous with Structural Computing.

It is also important to make clear the relationship between FOHM and MOS, which initially seem to take quite different approaches to the problems of Generalised Hypertext. FOHM argues against treating the domains of hypertext as exclusively separate but not against MOS itself, you could even implement FOHM in a MOS environment such as Construct.

FOHM could also benefit from some of the MOS formalisms, for example in our exploration of what is possible with FOHM, Context and Behaviour, domain development is occurring in an often ad-hoc fashion. There is a need to formalise the existing domains and devise a way of keeping track of the new ones (and their rules), structural schemas similar to the ones used by Callimachus [48] might be a solution.

In this paper we have also examined FOHM in terms of Structural Computing and Structuralist philosophy.

In particular we have identified the fact that FOHM takes a view that is at least as structural as CB-OHS’s implemented in MOS environments. Not only via the support of multiple domains of hypertext but also in the way that the Data Border, the point beyond which structure becomes opaque, can be pushed back, revealing structure that would otherwise remain ‘hidden’ to the hypermedia system (although it remains referencable via an Anchoring Interface).

This allows us to use our structural tools on structure that was previously unavailable. For example, we can use context filtering to support dynamic sequencing and transclusion for Adaptive Hypermedia.

Does this support for structure and the generalisation away from traditional node/link hypertext mean that FOHM is Structural Computing?

FOHM certainly promotes the development of tools that deal in general structure. For example Auld Linky is clearly a structure server that deals in first-class structure (as opposed to a link server). However, FOHM is just a model for expressing structure and its use does not necessarily endorse the view that explicit structure is preferable to the ‘hidden structure’ of data. In pushing back the Data Border our work with FOHM acknowledges the *existence* of that border.

One question that might be asked is whether or not a Structural Computing environment can have a Data Border at all? Generalised Hypertext plays to the Structuralist view and encourages us to push back the Data Border and allow our systems to manipulate structure that was previously ‘hidden’ inside data, but surely Structural Computing Environments should go much further and push the border back to the level of Structural Atoms [49]?

It would seem that Generalised Hypermedia is very much in the spirit of Structural Computing. FOHM could certainly be supported naturally within Structural Computing environments, including MOS environments, but this does not mean that a system that implements FOHM is automatically a Structural Computing environment.

Setting aside any debate about whether the domains of hypermedia should be considered as separate or continuous, we can still say that Generalised Hypermedia takes the view that *hypermedia* is better seen as a structural service and that this is evidence towards the broader claim that all services are better seen as structural.

8 Acknowledgments

This research is funded in part by EPSRC IRC project “EQUATOR” GR/N15986/01. Several people have also been involved in the development of the systems and models described in this paper. In particular I would like to thank my colleagues at the University of Southampton: Mark Weal, Danius Michaelides and David De Roure. I would also like to thank the reviewers for their helpful and constructive comments.

References

[1] P. J. Nürnberg, J. J. Leggett, E. R. Schneider, As We Should Have Thought, in: Proceedings of the ’97 ACM Conference on Hypertext, April 6-11, 1997, Southampton, UK, 1997, pp. 96–101.

[2] U. K. Wiil, P. J. Nürnberg, Evolving Hypermedia Middleware Services: Lessons and Observations, in: Proceedings of of the 1999 ACM Symposium on Applied Computing (SAC ’99), San Antonio, TX, 1999, pp. 427–436.

[3] D. Christodoulakis, M. Vaitis, A. Papadopoulas, M. Tzagarakis, The callimachus approach to distributed hypermedia, in: Proceedings of the ’99 ACM Conference on Hypertext, February 21-25, 1999, Darmstadt, Germany, 1999, pp. 47–48.

[4] D. E. Millard, L. Moreau, H. C. Davis, S. Reich, FOHM: A Fundamental Open Hypertext Model for Investigating Interoperability Between Hypertext Domains, in: Proceedings of the ’00 ACM Conference on Hypertext, May 30 - June 3, San Antonio, TX, 2000, pp. 93–102.

[5] K. M. Anderson, R. N. Taylor, E. J. Whitehead, Chimera: Hypertext for Heterogeneous Software Environments, in: ECHT ’94. Proceedings of the ACM European conference on Hypermedia technology, Sept. 18-23, 1994, Edinburgh, Scotland, UK, 1994, pp. 94–197.

[6] K. Grønbæk, R. H. Trigg, Design issues for a Dexter-based hypermedia system, Communications of the ACM 37 (3) (1994) 40–49.

[7] A. M. Fountain, W. Hall, I. Heath, H. C. Davis, MICROCOSM: An Open Model for Hypermedia With Dynamic Linking, in: A. Rizk, N. Streitz, J. André (Eds.), Hypertext: Concepts, Systems and Applications (Proceedings of ECHT’90), Cambridge University Press, 1990, pp. 298–311.

[8] A. Rizk, L. Sauter, Multicard: An open hypermedia system, in: ECHT ’92. Proceedings of the ACM conference on Hypertext, November 30-December 4, 1992, Milan, Italy, 1992, pp. 4–10.

[9] J. L. Schnase, J. L. Leggett, D. L. Hicks, P. J. Nürnberg, J. A. Sánchez, Design and implementation of the HB1 hyperbase management system, Electronic Publishing—Origination Dissemination and Design 6 (1) (1993) 35–63.

[10] H. C. Davis, A. Rizk, A. J. Lewis, OHP: A Draft Proposal for a Standard Open Hypermedia Protocol, in: U. K. Wiil, S. Demeyer (Eds.), Proceedings of the 2nd Workshop on Open Hypermedia Systems, ACM Hypertext ’96, Washington, D.C., March 16-20. Available as Report No. ICS-TR-96-10 from the Dept. of Information and Computer Science, University of California, Irvine, 1996, pp. 27–53.

[11] N. O. Bouvin, Experiences with ohp and issues for the future, in: S. Reich, K. M. Anderson (Eds.), OHS6 and SC2, Proceedings of the ..., Published in Lecture Notes in Computer Science, (LNCS 1903), Springer Verlag, Heidelberg (ISSN 0302-9743), 2000, pp. 13–22.

[12] O. H. S. W. G. (OHSWG), The open hypermedia protocol (OHP), Tech. rep., Open Hypermedia Systems Working Group (OHSWG), Available as http://www.ohswg.org/ (1999).

[13] S. Reich, U. K. Wiil, P. J. Nürnberg, H. C. Davis, K. Grønbæk, K. M. Anderson, D. E. Millard, J. M. Haake, Addressing Interoperability in Open Hypermedia: The Design of the Open Hypermedia Protocol, New Review of Hypermedia and Multimedia (2000) 207–243.

[14] C. C. Marshall, F. Shipman, J. H. Coombs, VIKI: Spatial hypertext supporting emergent structure, in: ECHT '94. Proceedings of the ACM European conference on Hypermedia technology, Sept. 18-23, 1994, Edinburgh, Scotland, UK, 1994, pp. 13–23.

[15] P. H. V. Dyke, Don't Link Me In: Set-Based Hypermedia for Taxonomic Reasoning, in: Proceedings of the '91 ACM Conference on Hypertext, Dec. 15-18, 1991, San Antonio, TX, 1991, pp. 233–242.

[16] P. J. Nürnberg, J. J. Leggett, U. K. Wiil, An Agenda for Open Hypermedia Research, in: Proceedings of the '98 ACM Conference on Hypertext, June 20-24, 1998, Pittsburgh, PA, 1998, pp. 198–206.

[17] P. J. Nürnberg, Repositioning structural computing, in: S. Reich, K. M. Anderson (Eds.), OHS6 and SC2, Proceedings of the ..., Published in Lecture Notes in Computer Science, (LNCS 1903), Springer Verlag, Heidelberg (ISSN 0302-9743), 2000, pp. 179–183.

[18] M. Bernstein, Storyspace 1, in: Proceedings of the '02 ACM conference on Hypertext, Maryland, U.S.A., 2002, pp. 172–181.

[19] D. E. Millard, D. T. Michaelides, D. D. Roure, M. J. Weal, Beyond the Traditional Domains of Hypermedia, in: Proceedings of the 2002 Workshop on Open Hypermedia Systems, ACM Hypertext '02, Maryland, U.S.A., June 11-15. To be published as a Technical Report from FernUniversitaet Hagen, Germany, 2002.

[20] H. C. Davis, D. E. Millard, S. Reich, N. O. Bouvin, K. Grønbæk, P. J. Nürnberg, L. Sloth, U. K. Wiil, K. M. Anderson, Interoperability between hypermedia systems: The standardisation work of the OHSWG (technical briefing), in: Proceedings of the '99 ACM Conference on Hypertext, February 21-25, 1999, Darmstadt, Germany, 1999, pp. 201–202.

[21] C. Bailey, W. Hall, D. E. Millard, M. J. Weal, Towards Open Adaptive Hypermedia, in: Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web-bases Systems, Malega, Spain, May 2002, 2002, pp. 36–46.

[22] L. Moreau, N. Gibbins, D. DeRoure, S. El-Beltagy, W. Hall, G. Hughes, D. Joyce, S. Kim, D. Michaelides, D. Millard, S. Reich, R. Tansley, M. Weal, SoFAR with DIM Agents. An Agent Framework for Distributed Information Management, in: The Fifth International Conference and Exhibition on The Practical Application of Intelligent Agents and Multi-Agents, Monday April 10 - Wednesday April 12, 2000, Manchester, UK, 2000.

[23] D. E. Millard, Hypermedia Interoperability: Navigating the Information Continuum, Ph.D. thesis, Department of Electronics and Computer Science, University of Southampton, UK (2000).

[24] N. Ridgway, D. D. Roure, FOHM+RTSP: Applying open hypermedia and temporal linking to audio streams, in: S. Reich, K. M. Anderson (Eds.), OHS7, SC3 and AH3, Proceedings of the ..., Published in Lecture Notes in Computer Science, (LNCS 2266), Springer Verlag, Heidelberg (ISSN 0302-9743), 2000, pp. 71–81.

[25] D. T. Michaelides, D. E. Millard, M. J. Weal, D. C. D. Roure, Auld leaky: A contextual open hypermedia link server, in: S. Reich, K. M. Anderson (Eds.), OHS7, SC3 and AH3, Proceedings of the ..., Published in Lecture Notes in Computer Science, (LNCS 2266), Springer Verlag, Heidelberg (ISSN 0302-9743), 2001, pp. 59–70.

[26] M. J. Weal, D. E. Millard, D. T. Michaelides, D. C. D. Roure, Building Narrative Structures Using Context Based Linking, in: Proceedings of the ’01 ACM conference on Hypertext, Aarhus, Denmark, 2001, pp. 37–38.

[27] M. Bernstein, Card Shark and Thespis: exotic tools for hypertext narrative, in: Proceedings of the ’01 ACM conference on Hypertext, Aarhus, Denmark, 2001, pp. 41–50.

[28] P. J. Nürnberg, HOSS: An Environment to Support Structural Computing, Ph.D. thesis, Department of Computer Science, Texas A&M University, College Station, TX (1997).

[29] H. A. Schutt, N. A. Streitz, HyperBase: A Hypermedia Engine Based on a Relational Data Base Management System, in: European Conference on Hypertext Technology (ECHT) 1990, 1990.

[30] U. K. Wiil, D. L. Hicks, P. J. Nürnberg, Multiple Open Services: A New Approach to Service Provision in Open Hypermedia Systems, in: Proceedings of the ’01 ACM conference on Hypertext, Aarhus, Denmark, 2001, pp. 83–92.

[31] P. J. Nürnberg, monica m. c. schraefel, Structural Computing and its Relationship to Other Fields, in: S. Reich, M. M. Tzagarakis (Eds.), OHS7, SC3 and AH3, Proceedings of the ..., Published in Lecture Notes in Computer Science, (LNCS 2266), Springer Verlag, Heidelberg (ISSN 0302-9743), 2001, pp. 183–193.

[32] G. Moore, L. Moreau, From Metadata to Links, in: S. Reich, K. M. Anderson (Eds.), OHS6 and SC2, Proceedings of the ..., Published in Lecture Notes in Computer Science, (LNCS 1903), Springer Verlag, Heidelberg (ISSN 0302-9743), 2000, pp. 77–86.

[33] D. Millard, H. Davis, Navigating Spaces: The Semantics of Cross Domain Interoperability, in: S. Reich, K. M. Anderson (Eds.), OHS6 and SC2, Proceedings of the ..., Published in Lecture Notes in Computer Science, (LNCS 1903), Springer Verlag, Heidelberg (ISSN 0302-9743), 2000, pp. 129–139.

[34] F. de Saussure, Course in General Linguistics, Fontana, 1974, translated from original Cours de Linguistique Generale, 1915 (Charles Bally, Albert Sechehaye and Albert Riedlinger editors).

[35] C. Levi-Strauss, Structural Anthropology, Penguin Books, 1972, translated from original Anthropologie Structurale, 1958.

[36] M. Neumüller, Applying Computer Semiotics to Hypertext Theory and the World Wide Web, in: S. Reich, K. M. Anderson (Eds.), OHS6 and SC2, Proceedings of the ..., Published in Lecture Notes in Computer Science, (LNCS 1903), Springer Verlag, Heidelberg (ISSN 0302-9743), 2000, pp. 58–65.

[37] T. Berners-Lee, J. Hendler, O. Lassila, The Semantic Web, Scientific American Available from http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html.

[38] A. Miles, Hypertext Structure as the Event of Connection, in: Proceedings of the ’01 ACM conference on Hypertext, Aarhus, Denmark, 2001, pp. 61–68.

[39] F. Halasz, M. Schwartz, The Dexter Hypertext Reference Model, Communications of the ACM 37 (2) (1994) 30–39.

[40] K. C. Malcolm, S. E. Poltrock, D. Schuler, Industrial Strength Hypermedia: Requirements for a Large Engineering Enterprise, in: Proceedings of the ’91 ACM Conference on Hypertext, Dec. 15-18, 1991, San Antonio, TX, 1991, pp. 13–24.

[41] K. Grønbæk, R. Trigg, Toward a Dexter-based Model for Open Hypermedia: Unifying Embedded References and Link Objects, in: Proceedings of the ’96 ACM Conference on Hypertext, March 16-20, 1996, Washington, D.C., 1996, pp. 149–160.

[42] T. H. Nelson, Literary Machines, Published by the author. Mindful Press, 1987.

[43] A. Moore, T. J. Brailsford, C. D. Stewart, Personally tailored teaching in WHURLE using conditional transclusion, in: Proceedings of the ’01 ACM conference on Hypertext, Aarhus, Denmark, 2001, pp. 163–164.

[44] P. Brusilovsky, J. Eklund, E. Schwarz, Web-based education for all: A tool for developing adaptive courseware, in: Computer Networks and ISDN Systems. Proceedings of Seventh International World Wide Web Conference, 14-18 April 1998, 1998, pp. 291–300.

[45] P. De Bra, L. Calvi, AHA: a Generic Adaptive Hypermedia System, in: Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia, ACM Hypertext’98, Pittsburgh, USA, June 20-24, 1998., 1998.

[46] P. D. Bra, G.-J. Houben, H. Wu, AHAM: A Dexter-based Reference Model for Adaptive Hypermedia, in: Proceedings of the ’99 ACM Conference on Hypertext, February 21-25, 1999, Darmstadt, Germany, 1999, pp. 147–156.

[47] P. Brusilovsky, Methods and Techniques of Adaptive Hypermedia, Journal of User Modelling and User-Adaptive Interaction, UMUAI6 .

[48] M. Kyriakopoulou, D. Avramidis, M. Vaitis, M. Tzagarakis, D. Christodoulakis, Broadening Structural Computing towards Hypermedia Development, in: S. Reich, K. M. Anderson (Eds.), OHS7, SC3 and AH3, Proceedings of the ..., Published in Lecture Notes in Computer Science, (LNCS 2266), Springer Verlag, Heidelberg (ISSN 0302-9743), 2001, pp. 131–140.

[49] D. L. Hicks, Structural Computing: Evolutionary or Revolutionary?, in: S. Reich, K. M. Anderson (Eds.), OHS6 and SC2, Proceedings of the ..., Published in Lecture Notes in Computer Science, (LNCS 1903), Springer Verlag, Heidelberg (ISSN 0302-9743), 2000, pp. 170–178.