A FRAMEWORK FOR CROSS-MEDIA INFORMATION MANAGEMENT

Beat Signer
Institute for Information Systems
ETH Zurich
CH-8092 Zurich, Switzerland
email: signer@inf.ethz.ch

Moira C. Norrie
Institute for Information Systems
ETH Zurich
CH-8092 Zurich, Switzerland
email: norrie@inf.ethz.ch

ABSTRACT

We present an information server architecture, called iServer, that enables cross-media linking based on an object-oriented hypermedia model. The iServer mixed-media platform can easily be extended to support new types of digital or physical multimedia resources. It provides core functionality for link and user management whereas link authoring is handled by plug-ins for specific resource types. Different forms of collaborative cross-media link authoring and link sharing are supported by a distributed iServer implementation based on peer-to-peer technologies.

KEY WORDS

Cross-media information management, multimedia integration, hypermedia model, link server

1 Introduction

The ubiquitous computing community are interested in developing information environments that span digital and physical spaces and offer context-dependent access to information through a range of media and devices. To do this, they are applying hypermedia concepts to structure and link information spaces in a flexible way, while supporting a range of physical and digital media types.

To achieve the goal of ubiquitous and pervasive hypermedia, we believe that it is essential to develop general information management frameworks capable of supporting a range of applications that vary in terms of requirements and environments. Further, to avoid the problem of re-inventing existing models and concepts, it is important that such a framework is capable of exploiting and integrating the wealth of existing cross-media technologies.

Our approach has been to develop iServer, an information server which brings together many existing hypermedia concepts through a generalised link model and a plug-in mechanism to support, not only different media types, but also different means of linking between these types. A user management component has been integrated into the very core of the iServer model to manage link access rights and enable applications to control the visibility of links in both a user- and context-dependent manner.

More recently, a distributed version of the iServer has been implemented based on peer-to-peer (P2P) technologies to support flexible means of sharing dynamically authored links among communities of users. Instead of having a central server, each user becomes an author of his personal link space managed by a local iServer instance and, at the same time, contributes as a publisher to the community's link knowledge.

iServer is based on an object metamodel and object-oriented database technologies. As we will show in this paper, our approach led naturally to a very general and flexible link model that supports various concepts of current hypermedia systems, but also caters for cross-media linking and extensibility for new resource types. The iServer metamodel is a generalisation of an information model that was initially developed within the European research project Paper++ [1] for linking printed and digital media.

We begin in Sect. 2 by motivating the design and implementation of iServer, a general and flexible cross-media information platform. Section 3 then introduces the core concepts of the iServer link model. In Sect. 4, we describe how iServer can be applied to link semantically rich information resources. Collaborative link authoring and link sharing together with several applications that have been implemented on the iServer platform are discussed in Sect. 5. Concluding remarks are given in Sect. 6.

2 Cross-media Information Management

Over the last few years there has been a trend, especially in ubiquitous computing, to not only link digital information, but also augment physical artefacts with digital information. Based on unique object identification, most existing cross-media architectures apply simple mapping approaches to resolve an object identifier to its corresponding digital content. However, to realise more flexible and interactive information environments, more recent mixed-media integration platforms take into account results achieved by the hypertext community.

Various open hypermedia frameworks have been introduced in the past. The sharing of link knowledge within these frameworks, and collaboration between different distributed link services and users, has always been a goal of the open hypermedia community. However, most open hypermedia models and architectures, including more recent ones such as the Fundamental Open Hypertext Model (FOHM) [2], do not handle user management and the issues of *data and link ownership* in the core of their model. In contrast, iServer supports access rights at link level and even on the base of a link's source or target objects. First experiments with our decentralised iServer implementation have shown that the user metadata helps in filtering the links retrieved in an open, distributed environment.

A goal of open hypermedia systems is to manage relationships between documents in external link management components. By not encoding the link information within the documents themselves, users can, not only add link anchors to their own resources, but also annotate other resources without having access rights for document modifications. Well known link servers include Chimera [3] and Microcosm [4]. The XHTML plug-in presented in Sect. 3.2 relates to these projects by defining external links and integrating them transparently into the documents.

We are aware of all the results achieved by the open hypermedia research community on enabling access to link services by third party applications and the integration of different link services. The iServer framework is a novel cross-media information platform that incorporates many features of existing hypermedia systems, while enabling the integration of physical and digital information. New media types are supported by simply implementing specific plug-ins.

3 iServer Model

In this section, we describe the main features and concepts of the iServer model. It is based on the semantic, object-oriented data model OM and we therefore begin with a brief overview of the main features of the OM model.

OM is a two-level model that integrates concepts of entity relationship and object-oriented models [5]. A clear distinction is made between the classification and typing of entities, dealing with these on separate layers. Typing deals with *representation* and entities are represented by objects with attributes, methods and triggers defined by the corresponding object types. *Classification* deals with *semantic roles* and a particular classification is represented by a *collection* of objects. In addition, OM provides a high level *association* construct which enables associations between entities to be classified and manipulated directly. Last but not least, cardinality and other constraints can be defined over collections and associations resulting in enhanced semantic expressiveness, which was a key factor in the development of our iServer framework.

The OM model also differs from commonly used object models such as the object diagrams provided by the Unified Modeling Language (UML) in that it is, not only intended for system design, but also as an operational model for data management. Thus the OM model defines a full operational model over objects, collections and associations as well as constructs for their definition. A number of data management systems have been developed based around the model. These include OMS Java [6] on which the presented iServer framework has been implemented.

3.1 Links

The core link model is shown in Fig. 1. The shaded *rectangular shapes* denote collections of objects (classification) where the name of the collection is given in the unshaded part and the name of the associated type in the shaded part. The type serves both as a constraint on membership in the collection and also as the default view of objects accessed through that collection. Thus, links are represented by objects of type `link` grouped into the `Links` collection. The shaded *oval shapes* represent associations between entities of two collections which can be restricted by cardinality constraints as described later.

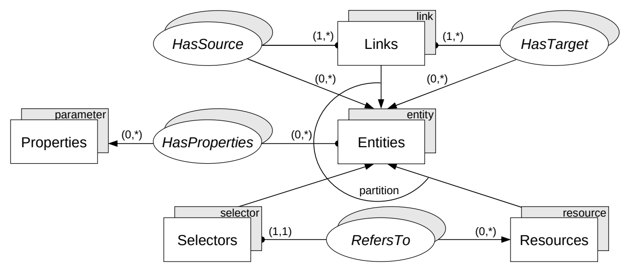

Figure 1. Links

Links within the iServer architecture are always directed and are bound to one or more *sources* and lead to one or more *targets*. A source may be either an entire document or a selected element within a document, commonly referred to an anchor. An information entity can be used equally as a link source or as a link target. This is taken into account in the information model by introducing a generic notion `entity` and making the collection `Entities` the target collection of both the `HasSource` and `HasTarget` associations. The cardinality constraints specified at the source and target points of the associations indicate the possible level of participation of individual objects. Thus `1:*` at the source point of both associations indicates that each link must have at least one and possibly many sources and targets. In this way, we achieve our objective of supporting not only multi-headed links but also links with multiple sources. The `0:*` at the target point of both associations specifies that there is no limit on the number of links for which an entity may be the source or target.

The simplest type of entity is the `resource` type representing an entire information unit. A hypermedia author normally wants to control the link granularity by being able to address specific parts of a document rather than the document in its entirety. Therefore, as a second subtype of the entity type we provide the concept of a `selector`, a construct enabling parts of the related resource to be addressed. This specialisation of `Entities` is reflected in the model by the subcollections `Selectors` and `Resources`. An association `RefersTo` represents the fact that a selector is always associated with exactly one resource, whereas each resource can have more than one referencing selector. The abstract `resource` and `selector` concepts define the interface which has to be implemented by plug-ins to support concrete types of media as described the next section.

By modelling `Links` as a subcollection of `Entities`, we gain the flexibility to create links where the sources or targets are defined by other links. This facility to define *links between links* allows us to annotate any link with supplementary information. Additionally, each entity can be associated with a set of properties as specified by the `Properties` collection and the `HasProperties` association. Those properties, represented by `key` and `value` pairs, are not predefined by the iServer framework. They can be defined individually to customise an entity's behaviour for specific application domains.

Note that since the underlying OM model provides bidirectional associations as a higher level construct, all the associations used within the iServer framework are also bidirectional. This enables us to, not only get all the link targets for a specific link source, but also to find the corresponding link sources given a specific target object.

The OM model presented in Fig. 1, as well as the OM models presented in succeeding sections, could always be transformed to the well-known UML object model. However, as a fully operational model for data management, OM provides specific database functionality not available in UML. Also, multiple instantiation is directly supported by all OM implementations to enable role modelling. For example, an iServer object can be instantiated with type `resource` and type `selector` at the same time. An OM database object can not only have multiple instances of different types, but it can even dynamically gain or lose specific types (object evolution).

3.2 Resources and Selectors

The iServer architecture is designed as a platform which supports the introduction of additional media types. By defining specific `resource` and `selector` subtypes and implementing the corresponding Java plug-ins, new resource types can be added to the system.

A first iServer plug-in was developed as part of the European project Paper++, under the Disappearing Computer Programme (IST-2000-26130), where we investigated the augmentation of physical paper [7]. In the case of the Paper++ plug-in, a printed page is a resource and a selector is an active area defined as a shape on the page as shown in Tab. 1. Special printing and reader technologies are used to enable users to select active areas. The plug-in supports links between printed and digital materials as well as highly-interactive applications where users can easily move back and forth between the printed and digital worlds.

More recently we have developed two additional iServer plug-ins to *fully* support HTML content and movie clips. Of course HTML documents and movies already have basic support in the core iServer implementation as it is possible to link to or from a web page or to a movie clip. By fully supporting these, we mean that it should be possible to link to/from elements within a resource and not just the entire resource by introducing resource-type-specific selectors.

To simplify the definition of an HTML plug-in, at the moment we deal only with HTML documents that conform to the W3C's XHTML standard. The definition of a corresponding selector for XHTML documents is straightforward since we can build on work that has already been accomplished in the context of the XML Linking Language (XLink) [8]. For HTML pages not conforming to the XHTML standard, we would have to supply alternative selectors based on standard HTML parsers.

Medium	**Resource**	**Selector**
paper	document page	shape
web page	XHTML document	XPointer
movie clip	mpeg file, avi file etc.	time span
movie clip	mpeg file, avi file etc.	shape
sound	mp3 file, wav file etc.	time span
image	gif file, jpeg file etc.	shape

Table 1. iServer plug-ins

XLink uses the XML Pointer Language (XPointer) to address a specific part of an XML document. By using XPointers as XHTML selectors within the iServer framework, we can define any part of an XHTML document as a link anchor or link target. In addition to the XLink language, we can define multi-layered links with a well-defined semantics.

Furthermore, the iServer architecture is not limited to linking only within HTML documents. We can freely link between different types of resources and elements within them. For example, a link may be defined between physical paper and parts of an HTML document.

A proxy server approach is applied for integrating the link information managed by the iServer into existing web pages. After a user has sent a request for a specific web page, the proxy server fetches the original Web document. It further connects to iServer to get supplemental information for the requested web page in the form of external links. The original page and the external links are combined by the proxy server (based on a DOM representation of the document) and the resulting HTML page containing additional iServer links is sent back to the browser.

Another iServer plug-in has been implemented for movies, images and sound files based on a QuickTime development kit provided by Apple. Spatial selectors may be defined to address parts of an image as shown in Tab. 1. Portions of a sound file can be selected by temporal selectors. For a movie clip, which is a combination of moving images and a sound clip, we can use either spatial or temporal selectors. In addition, a combination of spatial and temporal selectors may be applied to define link sources or link targets on movie clips.

3.3 Layers

Most existing link services either do not allow nested link anchors at all or, in the case that it is possible to define nested anchors, they lack the possibility of defining the behaviour of the nested links. Here is a brief example showing exactly what we mean by nested links. Assume that we have a link from a paragraph of an HTML document to a target object. In addition, we want to define links from various individual words within the same paragraph to some other link targets. What happens if we select a specific word with an associated link? In some hypermedia systems, we would only get the link associated with the corresponding word, whereas other systems may return the link target of the word as well as the paragraph's link target. To become more flexible in defining the semantics of nested link anchors, we introduce the concept of layers shown in Fig. 2.

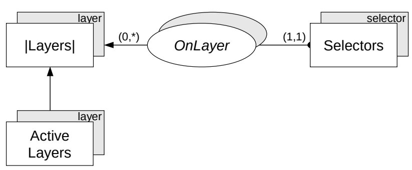

Figure 2. Layers

Each selector is associated with exactly one layer and we do not allow overlapping selectors on the same layer. In the case that a concrete selection would return several links by activating multiple overlapping selectors positioned on different layers, by definition, the link bound to the selector on the uppermost layer will be selected. The notation shown in Fig. 2 indicates that there is an explicit ordering of the layers, i.e. $|$`Layers`$|$ is a totally ordered collection. In addition, specific layers may be activated and deactivated enabling us to generate context-dependent links by resolving a particular selection to different selectors depending on the current set of active layers. Note that the ordering of the layers does not have to be static and may be changed dynamically, if appropriate, by the application. Further details about multi-layered links and, specifically, their use with digitally augmented paper are given in [9].

3.4 User Management

In order to support both personalisation and the sharing of link knowledge, we need a notion of *data ownership* combined with different levels of access rights. Since we consider user management to be a fundamental requirement for ubiquitous hypermedia, we provide the necessary functionality as one of the iServer's fundamental components and implement it in the core layer, as presented in Fig. 3. This contrasts with many systems that instead introduce user management on the application level only.

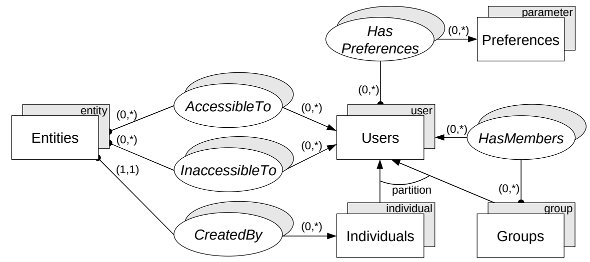

Figure 3. User management

A `user` can either be an `individual` or a `group`. Users can be classified in different groups represented by the collection `Groups`, where a `group` itself can be part of other groups. Each `entity` is created by exactly one individual user (owner). The owner has full control over an entity's content and can define access rights for other groups of users or individuals. The two associations `AccessibleTo` and `InaccessibleTo` are introduced to define access rights in a flexible way. The set of individuals having access to a specific entity is defined by the groups and individuals associated by `AccessibleTo` minus the groups and individuals being part of the `InaccessibleTo` association. In addition, there exists a constraint that access rights defined for an individual always have priority over access rights defined for a group. More formally, a group G is defined by some subgroups $G_1, \ldots G_k$ and some individuals $I_1, \ldots I_m$ it contains, i.e. $G = \{G_1, \ldots G_k, I_1, \ldots I_m\}$. The expanded set of individuals who are members of a group is given by the function *I* where

$$I(G) = \{I | I \in G \text{ or there exists } G_i \in G \text{ s.t. } I \in I(G_i)\}$$

For a specific entity e, let GA_e denote the set of groups explicitly specified as having access to e and GX_e those explicitly denied access. Correspondingly, let IA_e denote the set of individuals explicitly specified as having access to e and IX_e those explicitly denied access. Then $A(e)$, the set of individuals having access to entity e is defined as:

$$A(e) = IA_e \cup (I(GA_e) - I(GX_e) - IX_e)$$

This allows us to define complex access rights for an individual entity of the form "the entity should be visible to everybody except one specific group of users and two particular individuals". The `InaccessibleTo` association helps defining access rights when not all users are known which is often the case in peer-to-peer systems. By defining the access rights at the entity level, we can define individual permissions for links, resources and selectors. Note that we can, not only define different link sources for different users, but also bind different content to the same link source depending on a user's current role.

4 Semantic Resources

In the previous section we outlined how the iServer framework is applied to address parts of a resource and link them to different types of media. HTML documents and movie clips have been introduced as specific instances of resources. However, there is no information about the semantic content of these “simple” resource types.

Therefore, we distinguish between two kinds of resources as shown in Fig. 4. A resource can either be a *content resource* or a *semantic resource*. There already exist a few predefined content resources such as text components, image files and movie clips. Furthermore, it is possible to extend the system with any new type of content resource as described in Sect. 3.2. The content resources can be classified in a hierarchical way by applying the composite design pattern. The semantic expressiveness of such hierarchical structures is quite limited and hence, in addition to the content resources, we support direct links to semantic resources which may represent domain specific concepts or instances.

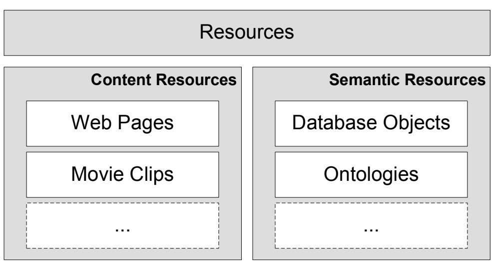

Figure 4. Content and semantic resources

In Fig. 4 database objects and ontologies are introduced as two forms of semantic resources. By linking not only to content resources but also database entries we gain some additional functionality. To give an example let us assume that we have a movie database providing information about movies, directors and moviestars. Instead of linking to a single content resource such as a movie clip represented by a file, the iServer architecture enables us to link to the corresponding movie instance within the application database which might itself contain different content resources (movie clips, textual information etc.). By doing so, we get access to supplementary semantics, such as information about the moviestars performing in the movie.

By having semantic as well as content resources, we support, not only completely heterogeneous and unstructured information spaces, but also well-structured database applications. Various object-oriented or relational database systems can be integrated by implementing specific plug-ins for each type of database. In the case of a database plug-in, the selector can be defined by a database query, which enables parts of a database’s content to be addressed. We have implemented applications where semantically rich resources are directly stored as an extension of the iServer. Other systems, for example the EdFest festivals guide [10] which provides information for tourists based on a paper-driven user interface, store application specific information in a separate database which is then accessed by the iServer framework.

In the case of our OMS databases, a key feature is the uniform handling of data and metadata. Thus all metadata, inclusive both of schema and system data, is represented as objects and can be classified, associated and operated over in the same way as application data. As a result, in the iServer framework, we can support direct links into specific semantic metadata (e.g. concepts of a schema) as well as to application objects. The former corresponds to the use of metadata annotation in the Semantic Web [11] and, specifically, its use to generate links between documents. In this way, we can link into entire application databases containing all sorts of information about the domain of discourse as well as to semantic descriptions of the domain.

5 Information Sharing

Several demonstrator applications have been developed using the iServer framework as part of the Paper++ project. Our initial applications were exclusively based on a publisher/consumer model, where the publisher (e.g. the BBC) does all of the authoring and the consumer reads and browses the preauthored information. The design of the corresponding cross-media information spaces proved to be a time consuming and expensive process. As a result we are investigating new authoring paradigms where much more power is given to the user.

Generally, we can distinguish at least three different types of links: *preauthored*, *personal* and *dynamic* links. Preauthored links are generated by a domain expert and made available to specific groups of users. The authoring of such predefined “tours” by a publisher can be very time consuming and, furthermore, it is often difficult to know different users’ needs in advance. Therefore, in addition to the preauthored links, each user is given the possibility to create his own set of links which can either be kept private or shared with other users. By introducing *virtual users* (software agents) as a special kind of system user, we can provide components that autonomously generate new dynamic links based on gathered user profiling information similar to Bush’s trailblazers in the Memex system [12] or other recommender systems [13].

In addition to being a consumer of preauthored links, a user should become a potential author of new link knowledge. To support this collaborative form of link authoring and sharing, we have developed a distributed version of iServer based on JXTA peer-to-peer protocols [14]. Each user has his personal link information space which can be dynamically enriched with link information from other users. By applying such an adaptive community authoring process, each user can contribute information according to his domain of expertise and, on the other hand, profit by the knowledge of experts in other domains.

After a user has selected a specific resource, the system will query the local iServer instance for links related to the corresponding resource. In the next step, the local iServer instance sends a query over the peer-to-peer network to request link information for the specific resource. The contacted distributed iServers will check if they have links defined on the corresponding resource and, based on the user management component, ensure that the requesting user is authorised to access this information. The link information will then be transmitted in XML format and has to be analysed and filtered by the requesting iServer instance.

Since "everyone" is allowed to participate in the link authoring process, we have to perform a filtering on the returned links to detect the most relevant links and guarantee a certain link quality. The selection and ranking of links can be based on a variety of criteria. A simple filter is based on the frequency of a link: A link delivered by multiple users gets a higher ranking than a link provided by just one individual. Other approaches include filters based on confidence values between different users.

This very open authoring approach can always be combined with the more controlled publisher and consumer model by treating well known publishers as authors with a high confidence value. However, the community authoring process is much more flexible in adapting quickly to evolving or new information and, furthermore, enables parts of the time consuming and expensive authoring process to be sourced out to the community of users.

6 Conclusions

We have presented iServer, an information platform providing the core functionality for flexible cross-media information management. The iServer platform has been implemented on top of the OMS Java data management system and provides a Java API as well as an XML interface for language independence. Our architecture not only allows the linking of various types of digital information, but also enables the digital augmentation of physical objects as well as the physical annotation of digital objects. By developing specific resource and selector implementations based on an object-oriented plug-in mechanism, it becomes easy to integrate new types of media and also associated forms of linking supported in existing hypermedia systems. Associative inter-application and cross-media linking as supported by the iServer architecture enables effective information management across isolated application domains.

The concepts of multi-layered selectors and integrated user management have been introduced into the core of the iServer framework. Together with a very general concept of a link, they build a common base used by all the resource plug-ins. Further, we have shown how the iServer supports collaborative cross-media link authoring and link sharing based on completely decentralised peer-to-peer networking.

References

[1] Paper^{++} Project, http://www.paperplusplus.net.

[2] D. Millard, L. Moreau, H. Davis, and S. Reich. FOHM: A Fundamental Open Hypertext Model for Investigating Interoperability between Hypertext Domains. In *Proc. of ACM Hypertext 2000*, pages 93–102, May 2000.

[3] K. M. Anderson, R. N. Taylor, and Jr. E. J. Whitehead. Chimera: Hypermedia for Heterogeneous Software Development Environments. *ACM Transactions on Information Systems*, 18(3):211–245, 2000.

[4] W. Hall, H. C. Davis, and G. Hutchings. *Rethinking Hypermedia: The Microcosm Approach*. Kluwer Academic Publishers, May 1996.

[5] M. C. Norrie. An Extended Entity-Relationship Approach to Data Management in Object-Oriented Systems. In *Proc. of ER'93*, pages 390–401, Arlington, USA, December 1993.

[6] A. Kobler and M. C. Norrie. OMS Java: A Persistent Object Management Framework. *L'object, Java and Databases*, 6(3), June 2000.

[7] P. Luff, C. Heath, M. C. Norrie, B. Signer, and P. Herdman. Only Touching the Surface: Creating Affinities Between Digital Content and Paper. In *Proc. of CSCW 2004*, pages 523–532, Chicago, USA, November 2004.

[8] S. J. DeRose. XML Linking. *ACM Computing Surveys*, 31(4), December 1999.

[9] B. Signer and M. C. Norrie. Multi-Layered Cross-Media Linking. In *Proc. of ACM Hypertext 2003, 14th Conference on Hypertext and Hypermedia*, pages 106–107, Nottingham, UK, August 2003.

[10] M. C. Norrie and B. Signer. Overlaying Paper Maps with Digital Information Services for Tourists. In *Proc. of ENTER 2005*, Innsbruck, Austria, January 2005.

[11] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, May 2001.

[12] V. Bush. As We May Think. *Atlantic Monthly*, 176(1):101–108, July 1945.

[13] M. Chalmers, K. Rodden, and D. Brodbeck. The Order of Things: Activity-Centred Information Access. In *Proc. of WWW7*, pages 359–367, Brisbane, Australia, April 1998.

[14] B. Traversat, A. Arora, M. Abdelaziz, M. Duigou, C. Haywood, J.-C. Hugly, E. Pouyoul, and B. Yeager. Project JXTA 2.0 Super-Peer Virtual Network. Technical report, Sun Microsystems, March 2003.