# An Architecture for Open Cross-Media Annotation Services

Signer, Beat; Norrie, Moira C.

*Published in:*
Proceedings of the 10th International Conference on Web Information Systems Engineering (WISE 2009)

*Publication date:*
2009

*License:*
Other

*Document Version:*
Accepted author manuscript

# An Architecture for Open Cross-Media Annotation Services

Beat Signer[1] and Moira C. Norrie[2]

[1] Vrije Universiteit Brussel
Pleinlaan 2
1050 Brussels, Belgium
bsigner@vub.ac.be
[2] Institute for Information Systems, ETH Zurich
CH-8092 Zurich, Switzerland
norrie@inf.ethz.ch

**Abstract** The emergence of new media technologies in combination with enhanced information sharing functionality offered by the Web provides new possibilities for cross-media annotations. This in turn raises new challenges in terms of how a true integration across different types of media can be achieved and how we can develop annotation services that are sufficiently flexible and extensible to cater for new document formats as they emerge. We present a general model for cross-media annotation services and describe how it was used to define an architecture that supports extensibility at the data level as well as within authoring and visualisation tools.

## 1 Introduction

With the rapid growth of Web 2.0 communities, many users are no longer simply passive readers of information published on the Web and have become actively involved in the information management process by creating new content or annotating existing resources. While web technologies have enabled the large-scale and low-cost sharing of information, annotation services allow users to integrate and augment that information in an ad-hoc manner without any pre-defined integration schemas or the need to have a local copy of that information or even update access. This allows user communities to build a knowledge layer on top of the Web through various forms of annotation services. As a result, the idea of external link metadata as introduced by the hypertext community in the form of dedicated link servers, has nowadays found its manifestation in more widely used applications in the context of Web 2.0.

The opening of information resources to third-party contributors has also been recognised by the digital library community as a way of enriching existing content with community-based annotations and associations to supplementary external resources. By bridging the gap between content managed within a digital library and digital information available outside of the library, as well as enabling annotations across digital library systems, external annotation and

link services may contribute to the integration of content managed by different digital libraries.

The potential of knowledge sharing through collaborative annotations can only be fully exploited, if a general and sustainable *annotation fabric* can be established to ensure that annotations persist over time and can be reused and extended by future applications. Therefore, some common annotation standards and guidelines are required to make different solutions interoperable rather than producing isolated and proprietary annotation services. In the context of the Web, we have already seen first efforts to establish specific annotation standards such as the one defined by the Annotea[3] framework. The digital library community has also tried to establish annotation standards by defining digital library reference models which include annotations as information objects.

However, in addition to specifying common annotation models and standards, it is necessary to define a flexible and extensible reference architecture capable of supporting any form of cross-media annotation. It is no longer sufficient to support only textual or a fixed set of multimedia annotations. The Web is a platform with a rich and continuously evolving set of multimedia types and it is important to ensure that link and annotation services can be easily extended to cater for new media types at the data level as well as by integrating them into authoring and visualisation tools. In this paper, we present such an architecture along with the general cross-media annotation model on which it is based.

We begin in Sect. 2 by providing an overview of existing annotation systems. In Sect. 3 we introduce the concept of open cross-media annotation systems and discuss some of their requirements in terms of extensibility on both the model and architecture level. We then introduce our cross-media annotation model in Sect. 4, discussing how it supports extensibility and comparing its main features with existing annotation proposals. Details of how to realise an annotation service based on the proposed model and architecture are provided in Sect. 5. Concluding remarks are given in Sect. 6.

## 2 Existing Annotation Systems

Before discussing different solutions for content annotation, we consider the question of what the difference is between an annotation and a link or association with supplemental information. In our opinion, the annotation process mainly "differs" from regular linking as known from a variety of hypermedia systems through the fact that the creation of a new annotation often includes the content authoring of the annotation object itself. In contrast, link authoring usually creates associations between existing resources. We would therefore see annotation services as a specialised application of more general link services. This implies that we do not treat annotations as metadata but deal with them on the same level as any other information object. In this section, we therefore cover more general hypermedia solutions as well as specialised annotation services.

[3] http://www.w3.org/2001/Annotea/

The Annotea [1] project developed by the World Wide Web Consortium (W3C) provides a framework for collaborative semantic annotations and bookmarks as well as topics. Annotea makes use of the Extensible Markup Language (XML) in combination with the Resource Description Framework (RDF) to store annotation metadata about XML documents on separate servers. The W3C's Amaya[4] browser and editor uses Annotea to annotate arbitrary web pages. The Amaya editor enables parts of an HTML document to be addressed based on XPointer expressions which can then be annotated by textual information. Some of the ideas introduced by Annotea are nowadays used in social bookmarking and tagging systems. The linking (and annotating) of XML resources is also supported by the XML Linking Language (XLink) [2].

While Annotea and XLink make explicit assumptions about the type of document to be annotated, the Flexible Annotation Service Tool (FAST) [3] claims to be more flexible by providing a core annotation service with different gateways for specific information management systems. The gateway approach is a good mechanism to integrate the annotation service with different information management systems. However, FAST does not explicitly deal with extensibility issues in terms of different media types on the annotation tool and application level. For a cross-media annotation service, it is essential that new media types can be introduced without having to change already existing applications as we show in the next section.

The interoperability of link services has also been discussed by the open hypermedia community and different proposals such as the Open Hypermedia Reference Architecture (OHRA) [4] have been made. The same comments that have been given for FAST in terms of a simple extension with new media types are also valid for the OHRA architecture.

An annotation service addressing parts of documents managed by a digital library system through the concept of *marks* is presented by Archer et al. [5]. Annotations can be stored either together with the document or in an external repository. While the system provides a flexible means of addressing specific document parts, it currently supports only textual annotations.

A fixed set of multimedia annotations is supported by the web-based MADCOW [6] multimedia digital annotation system which uses a client-server architecture in combination with a browser plug-in. A good overview of MADCOW and other annotation solutions is provided in [7]. While these systems can be extended on the model level to support new types of media, we will show that there is a lack of simple extensibility on the application level. In an optimal case, there should be a clear separation of concerns not only between the media-specific annotation details on the model level but also between a general annotation authoring and management tool and its components dealing with various types of annotation resources. As a contribution of this paper, we therefore discuss some limitations of existing annotation tools and introduce an architecture for extensible cross-media annotation services.

[4] http://www.w3.org/Amaya/

## 3 Open Cross-Media Annotation

In this section, we discuss the limitations of existing digital annotation tools with respect to support of cross-media annotations and introduce the requirements for true cross-media annotation tools. Existing annotation architectures and services can be classified based on the types of resources that can be annotated as well as the potential media types that can be used in annotations. To illustrate the different types of systems, we define the *annotation matrix* shown in Fig. 1. On the horizontal axis, we mark the number of different resource types that can be annotated whereas on the vertical axis we record the number of different media types that can be used in annotating a given resource.

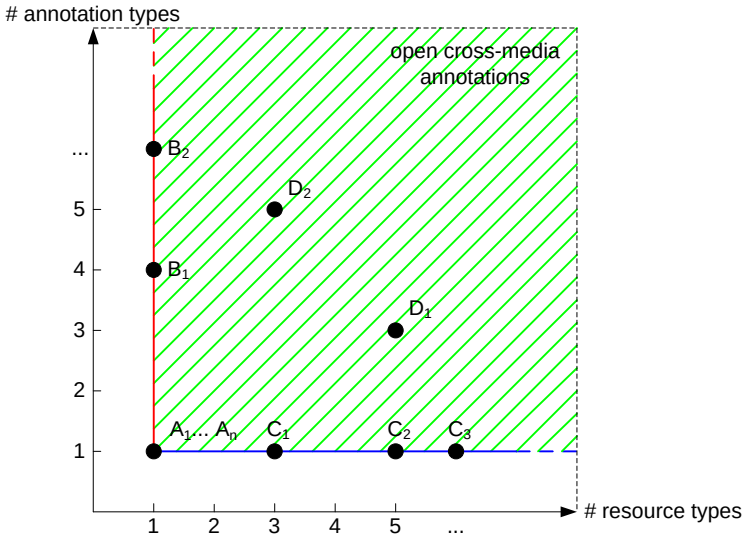


**Figure 1.** Annotation matrix

The simplest type of annotation service, represented by $\mathscr{A} = \{A_1, \ldots, A_n\}$ in Fig. 1, only provides functionality for one type of resource to be annotated (e.g. text) and the annotations themselves can also be of a single type only (e.g. sound). An example of such a system is Annotea where XML documents are annotated with textual content. Some more flexibility is provided by systems where a single type of resource can be associated with annotations of different media types. For example, textual content is annotated with text notes, sounds and movies. These types of annotation services $\mathscr{B} = \{B_1, \ldots, B_n\}$ are located on the vertical line going through 1. The Stickis[5] browser toolbar is such a solution where regular webpages can be annotated with a set of rich media content. A third class of systems $\mathscr{C} = \{C_1, \ldots, C_n\}$ enables the annotation of different types of resources, but with a single annotation media type only. Those solutions can

[5] http://stickis.com

be found on the horizontal line going through 1. Last but not least, we have true cross-media annotation services $\mathscr{D} = \{D_1, \ldots, D_n\}$, where a set of different resource types can be linked to annotations of different media types. An example of such an annotation service is MADCOW, where a fixed set of digital resource types (i.e. text, images and videos) can be annotated with text, images, sound or videos.

Even if we have a true cross-media annotation service, there is often a limitation in terms of there being a fixed set of media types that can be annotated and used in annotations. We aim for an extensible solution where any new type of resource or annotation can be added at a later stage. We name these types of extensible solutions *open cross-media annotation systems*. Open cross-media annotation systems are not represented by a single point in our annotation matrix, but rather cover the entire shaded area. While some existing solutions such as the FAST model support this kind of extensibility on the model level—at least for digital media types—we show that there is a lack of extensibility when it comes to the architecture and application level.

To illustrate what we mean by a lack of extensibility on the annotation architecture and application level, let us have a closer look at the MADCOW [6] multimedia digital annotation system. As mentioned earlier, the authoring tool for creating new multimedia annotations has been realised as a browser plug-in. The tool currently deals with text, image and video annotations which is also reflected through different visual elements such as media-specific buttons in the MADCOW user interface. Let us consider what happens if it is decided that a new media type, for example sound, should be supported by the MADCOW annotation system. Since the authoring tool has been implemented as a single monolithic component, the user interface would have to be extended to deal with the new type of resource. This implies that for each newly introduced media type, a new version of the user interface would have to be deployed. Furthermore, since there is no flexible mechanism to dynamically extend the set of supported media types on demand, each instance of the annotation tool always has to support all existing types of resources even if a user works only with a limited subset of these media types. Last but not least, often there is not a single annotation tool but different versions (e.g. browser plug-in and standalone component) making use of the same underlying annotation model. Therefore, we have to ensure that the user interfaces of all existing annotation tools are extended individually in order to support a single new media type. This problem of extensibility on the annotation tool or application level is not something that is present in MADCOW only, but rather is common to most existing annotation solutions when faced with requirements to introduce new media types. Our solution to deal with this extensibility problem is to make sure that the visual definition of annotation anchors (selectors) for a specific resource type is no longer part of the general annotation tool but realised in separate visual plug-in components that can be automatically installed on demand.

We propose an architecture for an open cross-media annotation system based on a cross-media annotation model that supports this form of extensibility.

The basic idea is that we have one or more annotation services that offer their functionality to different client applications as shown in Fig. 2. A first important thing to point out is that we make a clear distinction between the core annotation and link service and any media-specific implementation. The annotation service knows how to deal with the underlying annotation model presented in the next section but any media-specific functionality is introduced via specific *data plug-ins*. To extend the annotation service with a new media type, a data plug-in has to be provided. An annotation service might be installed with an existing set of data plug-ins, but plug-ins can also be downloaded and installed on demand from different resource plug-in repositories (see dashed arrows in Fig. 2). Since we aim for extensibility not only on the model and data layer but also on the application level, a *visual plug-in* has to be developed in addition to the data plug-in. While it seems to be obvious to separate the media-specific creation and visualisation of annotation or link anchors from the general annotation tools, it is exactly the current lack of this separation of concerns that makes it difficult to flexibly extend existing annotation solutions with new media types.

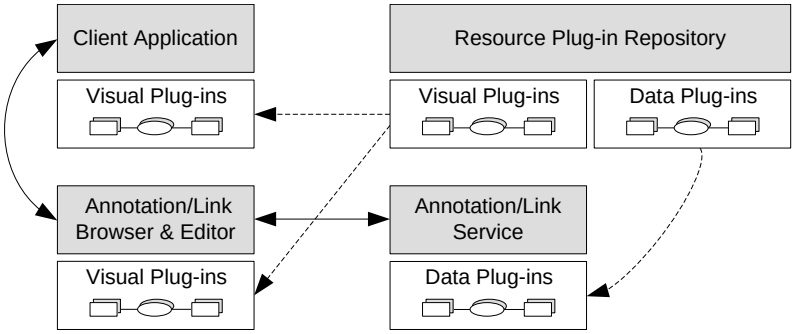


**Figure 2.** Open cross-media annotation architecture

Since we do not want to force application developers to rewrite and change their entire application to make use of our annotation service, we propose a standalone annotation/link browser component that runs on the client platform. The only required communication between a client application and the annotation browser deals with information about the resource that is currently accessed within the client application. Based on a unique resource identifier, the annotation browser contacts the annotation service to get information about any additional external annotation and link data that has been defined for the given resource. The annotation browser also has to ensure that a visual plug-in for the given resource type is installed. Each visual plug-in has basically two purposes. First, it has to be able to render a specific resource type and visualise any annotation anchors (defined by selectors) that have been defined within that resource. Secondly, the visual plug-in has to provide some functionality to create and delete resources as well as selectors. After the information about

the annotations has been retrieved from the annotation service, the annotation anchors will be highlighted by the visual plug-in. In the case that an annotation is selected within the annotation browser, a request is sent to the annotation server to get supplemental information for the selected annotation. As soon as another resource is accessed in the client application, the information shown in the annotation browser is automatically updated. While the communication and integration of existing applications with an annotation tool is not novel and has already been used in related approaches, again the extensibility of these annotation tools is often limited due to the fact that the application logic of the tool deals with media-specific details.

Having presented the general idea of open cross-media annotation systems along with the requirements for extensibility on both the data and application levels, we will provide some details of how extensibility is achieved on each of these levels in the next two sections.

## 4 Annotation Model

In this section, we start by looking at one of the proposed reference models for annotation services before going on to present our general cross-media annotation model that could be used as a basis for the implementation of such services.

Within the DELOS Network of Excellence on Digital Libraries[6], a reference model (DLRM) was defined to support more systematic research on digital libraries and serve as a foundation for comparing the functionality of different digital library implementations. We briefly outline the parts of the DELOS reference model dealing with annotations. This enables us to position our model in relation to the existing reference model as well as highlighting some of the major differences arising from the goal and intended use of the model.

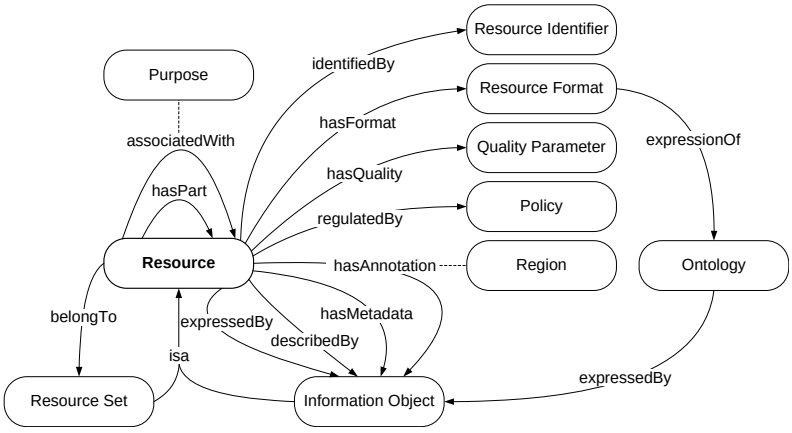


**Figure 3.** Digital library resource domain concept map

[6] http://www.delos.info

Figure 3 shows parts of the digital library resource concept map as introduced in the DLRM document [8]. The most general concept in the reference model is the `Resource` which is used to represent any digital library entity. Particular instances of digital library resources (e.g. text, videos and annotations) are represented by the `Information Object` concept. A `Resource` defines some characteristics which are shared by all the different types of resources. These characteristics include a unique resource identifier, information about the resource format and quality as well as specific resource policy information.

The definition of composite resources is supported through the `hasPart` relation whereas the linking of different resources is enabled by the `associatedWith` relation. The annotation of arbitrary resources (or particular regions) with other information objects is represented by the `hasAnnotation` relationship between the `Resource` and `Information Object` concepts. Since we will pay special attention to the annotation mechanism while comparing our model with the reference model, we would also like to give the exact definition of an annotation as provided in the DLRM document:

> An *Annotation* is any kind of super-structural *Information Object* including notes, structured comments, or links, that an *Actor* may associate with a *Region* of a *Resource* via the <*hasAnnotation*> relation, in order to add an interpretative value. An annotation must be identified by a *Resource Identifier*, be authored by an *Actor*, and may be shared with *Groups* according to *Policies* regulating it (*Resource* is <*regulatedBy*> *Policy*). An *Annotation* may relate a *Resource* to one or more other *Resources* via the appropriate <*hasAnnotation*> relationship.
>
> Candela et al. [8]

After this very brief overview of the concepts for annotating and linking resources in the DELOS digital library reference model, we now introduce our model. The first thing to note is the fact that our model is defined using the OM data model [9] that integrates concepts from both entity relationship (ER) and object-oriented data models and is intended to bridge the gap between conceptual and implementation models. This means that our model can be mapped directly to database structures and is therefore a step closer to the realisation of annotation services than the typical reference models while still being at the conceptual level.

As explained in Sect. 2, we treat an annotation as a special type of link between two or more resources. Our annotation model is actually an application and extension of our more general resource-selector-link (RSL) model [10] for cross-media linking. The extended RSL model is shown in Fig. 4.

The OM model supports information modelling through a separation of classification and typing. While *typing* deals with entities represented by objects with attributes, methods and triggers, the *classification* through named collections deals with the semantic roles of specific object instances. In Fig. 4, collections are represented by the rectangular shapes with the membertype specified in the shaded upper right part. The OM model provides a high-level

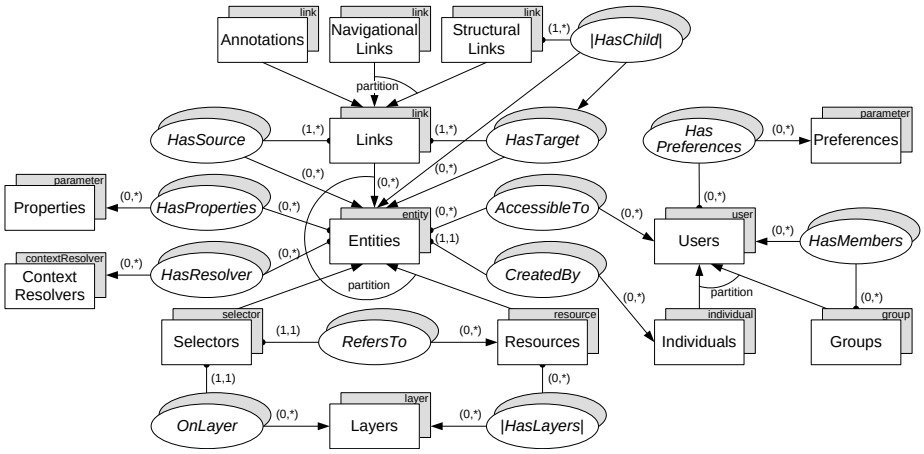


**Figure 4.** RSL-based annotation model

*association* construct, represented by an oval shape, which enables associations between entities to be classified and manipulated directly. A ranking over an association is indicated by placing the association's name between two vertical lines (e.g. |`HasLayers`|). It is important to emphasise that OM also serves as a modelling language for a set of object-oriented data management systems and has been used to implement our link and annotation server (iServer) [11].

Similar to the `Resource` concept in the DLRM model, our annotation model introduces the generic notion of an `entity` type and all entity instances are classified and grouped by the collection `Entities`. As in the DELOS reference model, an entity has different characteristics which are shared among all specialisations of the `entity` type. Each entity is created by exactly one `individual` which is represented in the model by the `CreatedBy` association. Furthermore, access rights can be defined at entity level by the `AccessibleTo` association. Note that these access rights can be granted on the group level or to individuals as well as to combinations of groups and individuals. A set of `contextResolver` instances can be associated with each entity which defines if an instance is available within a specific context. Last but not least, arbitrary properties (parameters) in the form of key/value pairs can be associated with an entity by using the `HasProperties` association. This enables the extension of entities with any additional metadata required by third party applications without having to extend the core data model. To deal with complex metadata, an entity can also be associated with other entities by using the concept of a link introduced in the following paragraphs. The RSL model offers three specialisations of the abstract entity concept represented by the `resource`, `selector` and `link` subtypes.

The `resource` type represents any particular digital or physical resource that has to be managed by the annotation and link model. It is similar to the `Information Object` concept in the DLRM model. For each specific resource

type to be supported, a new resource subtype with media-specific characteristics has to be defined via a resource plug-in mechanism.

The definition of links between different entities is supported by the `link` type. A link can have one or multiple source entities and point to one or more target entities which is reflected by the cardinality constraints on the `HasSource` and `HasTarget` associations. As mentioned earlier, we treat annotations as a special classification of links which is represented by the collection named `Annotations` in our model. Note that by treating links and annotations as first-class objects and at the same time modelling them as specialisations of the `entity` type, we gain some flexibility compared to the DLRM model where links are represented by the `associatedWith` relation. We can not only define links between resources but also create links that have other links as source or target objects. This enables us, for example, to easily add an annotation to a link; something which is not possible in the DLRM model since the `hasAnnotation` relationship cannot be defined over the `associatedWith` relation.

Often we want to link or annotate specific parts of a resource rather than entire resources. In our model, we therefore introduce the `selector` type as a third specialisation of the `entity` type. A selector is tightly coupled to a specific resource type (over the `RefersTo` association) and enables the selection of a specific part of a given type of resource. For example, a selector for sounds might be time-based (i.e. from time $t_i$ to time $t_j$) whereas a selector for text documents could be based on character positions (i.e. from character $c_i$ to character $c_j$). It is up to the developer of a new resource plug-in to not only provide an implementation for the specific `resource` type but also the corresponding `selector`. Each selector is further associated with a `layer` which, in the case of overlapping selectors, defines their precedence order.

How does our selector concept compare to the resource addressing functionality offered by the DLRM model? In the DLRM model, specific regions of a resource can be annotated by using the `Region` concept. However, the mechanism for selecting a specific region of a resource is only available for the information object to be annotated but not for the annotation itself. This means that, in DLRM, only entire information objects can be used as annotations whereas, in our RSL-based model, also parts of resources can be used to annotate other entities. Another benefit of the selector concept and the modelling of links and annotations as first-class objects becomes evident, if we revisit the concept of links provided by the `associatedWith` relation in the DLRM model. There, links can only be defined between entire resources whereas in our model we can use the selector concept to create links between specific parts of different resources.

As described earlier, our RSL-based annotation model defines any access rights at entity level. This has the advantage that we can not only specify if a resource is available as supported in the DLRM model by the `regulatedBy` and `Policy` concepts but also define access rights on the selector and link level. This means that we can, for example, define that a selector which is used to annotate a resource is only available for specific users whereas the resource itself may be

available for everybody. We can therefore specify access rights on a very fine level of granularity and not just define if an entire resource is accessible or not.

The same flexibility that has just been described for accessing annotations, links, resources and selectors based on user profiles is also applicable to the context-specific information delivery based on the `contextResolver` concept introduced earlier in this section. This implies that an annotation or any other entity might only be accessible in a specific context. For example, some annotations might only become available if the user has already accessed specific resources beforehand.

A final remark has to be made about the representation of different types of annotations in our annotation model. In Fig. 4, only a single type of annotation, described by the `Annotations` collection, is shown. Of course it is easily possible to distinguish different types of annotation by introducing further subcollections. We can, for example, distinguish between formal and informal annotations as well as comments, explanations and other types of annotations. Since the OM model offers the possibility that an object can be a member of different collections, it is even possible that an annotation has multiple classifications at the same time as described in [12]. Annotea also offers a flexible classification of annotations via the annotation subtype concept. A slightly different approach has been chosen in FAST [7], where parts of an annotation can be classified via a specific meaning mechanism.

Our annotation model introduces some flexibility in terms of the granularity and the types of objects that can be annotated as well as used in annotations. While the model has many similarities to existing solutions, for example the DLRM model, it also shows that through generalisation and the treatment of annotations and links as first-class objects, we become more flexible in cross-annotating digital as well as physical content. While the presented model can be extended to deal with new types of media by providing specific resource and selector implementations, the management of cross-media annotation and link information is only part of the problem to be addressed. Whereas other annotation models such as the FAST model also deal with media extensions on the model level, in the next section we investigate some of the problems arising when this extensibility should be supported at the application and annotation tool level. Based on our experience in implementing solutions for different types of cross-media annotations, we propose an extensible and scalable architecture for open cross-media annotation services.

## 5 Extensible Annotation Tools

After highlighting the requirements for extensible cross-media annotation services and presenting our solution on the model layer, we now show how the extensibility can be dealt with on the authoring tool and visualisation level. As introduced earlier, the data plug-ins are responsible for persistently storing any additional data that is required to support a new media type. In particular, a specific implementation of the resource and selector concepts have to be provided

for each new data plug-in and the interface methods to create, read, update and delete (CRUD) media-specific data have to be implemented.

The functionality of a visual plug-in is defined by an interface that has to be implemented by concrete visual plug-in instances. Each visual plug-in has to provide some functionality to define new resources as well as selectors which can then be used as annotation sources or targets by the general annotation tool. Furthermore, the interface defines a number of methods that are used by the general annotation tool to get access to the selector or resource that is currently selected within the visual plug-in. This is the only direct connection from the annotation browser introduced earlier in Fig. 2 to arbitrary visual plug-ins.

The annotation browser can not only be used to browse existing annotations but also as an authoring tool to define new cross-media annotations. In the default setting, the annotation browser shows two main windows next to each other as indicated in Fig. 5. The window on the left-hand side represents the source document whereas the one on the right-hand side is for the target document. The tool further provides functionality to create and delete annotations (CRUD) as well as to deal with more general functionality of the link model (RSL). To define an annotation for a given source document, the user first selects the specific part of the resource to be annotated in the left window and then annotates it with parts of the resource shown in the right window. Note that as part of the annotation process, the user can not only select existing resources but also create new annotation resource instances based on the editing functionality offered by the visual plug-in. After selecting the 'create annotation' command, the authoring tool gets access to the required selected entities via the visual plug-in interface. Note that since this single dependency between the authoring tool and any existing plug-ins is defined on the entity level (resources or selectors), the authoring tool does not deal with any media-specific implementation and therefore does not have to be changed at all to support a new resource type via the visual plug-in mechanism.

The default setup with two adjacent windows for the source document and its annotation is very similar to the configuration of the Memex described by Bush, where also a source and target screen are available [13]. The major difference is that in Bush's vision there is only a single resource type (microfilm) available, whereas in our case we have a potentially unlimited number of resource types represented by the set of available data and visual plug-ins. Of course the type of resources visualised in the two windows can be changed independently since each window is managed by a separate instance of a visual resource plug-in. Furthermore, different configurations of the annotation authoring tool with more than two resource windows are also imaginable.

While the use of the annotation browser and authoring tool provides access to external annotation services without any GUI changes to an existing client application, it is also possible to integrate the visualisation functionality for specific media types directly within the client application. A client application can either make use of existing visual plug-ins or the functionality defined by the visual plug-in interface can be implemented in an application-specific manner.

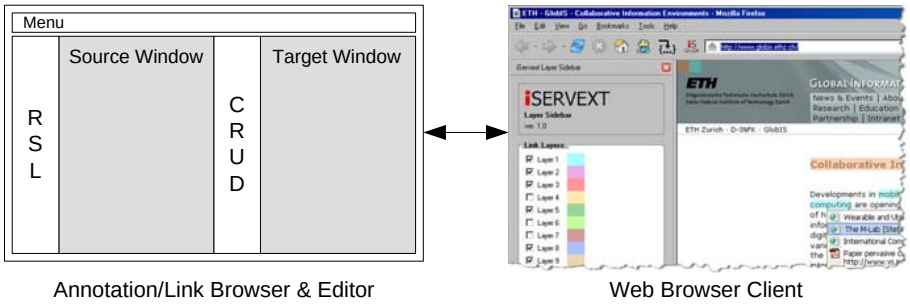


**Figure 5.** Annotation browser and editor interfacing with external clients

For example, the right-hand side of Fig. 5 shows a web browser client with a visual plug-in which we developed for the XHTML resource type. The web browser client communicates with the annotation editor and can either act as a substitute for the source or target window. The important thing to note is that each resource type is treated separately through a specific plug-in. If a user selects a highlighted annotation selector within the client, it will be checked whether a visual plug-in for the linked annotation is available and, if so, the annotation is visualised. In the case that there is no client-specific visualisation available, the annotation browser will be used as a mediator to visualise the corresponding annotation. This has the major advantage that we can add new types of resources to our annotation service without the client application having to know about them. Of course, if desired, the client application can then always be extended to "natively" support the new media type as shown for the web browser extension.

In the annotation authoring process described earlier, we can not only define the selectors within the authoring tool but also directly access information from the visual plug-ins installed in external client applications. In this case, the client application informs the annotation tool about the currently active selector which has to be used as an annotation source or target. This has the advantage that, for annotation-aware client applications (with the corresponding visual plug-ins), any selections can be done directly within the application and only the command to create the annotation has to be issued by using the annotation authoring tool.

Various applications have been realised based on the presented cross-media annotation and link model. For that purpose, different plug-ins for digital resources (e.g. web pages or movies) as well as physical resources (e.g. interactive paper or RFID-tagged objects) have been implemented [11]. While our earlier applications were based on a simpler client-server architecture, we are currently implementing the described architecture which should finally result in the desired open cross-media annotation and link service.

## 6 Conclusions

We have presented an architecture for an open cross-media annotation system that can be dynamically extended with new media types. Through generalisation

and the treatment of annotations and links as first-class objects, the presented RSL-based annotation model introduces some flexibility in comparison to existing annotation models. While a number of existing annotation models deal with extensibility on the model level, the corresponding extensibility is missing on the authoring and visualisation level. We have presented an integrated open cross-media annotation solution providing a sustainable annotation fabric in terms of an extensible cross-media annotation model together with an architecture that guarantees future extensibility and ensures that annotations persist and can be reused over time.

## References


1. Koivunen, M.R.: Semantic Authoring by Tagging with Annotea Social Bookmarks and Topics. In: Proc. of SAAW2006, 1st Semantic Authoring and Annotation Workshop, Athens, Greece (November 2006)
2. Christensen, B.G., Hansen, F.A., Bouvin, N.O.: Xspect: Bridging Open Hypermedia and XLink. In: Proc. of WWW 2003, 12th Intl. World Wide Web Conference, Budapest, Hungary (May 2003)
3. Agosti, M., Ferro, N.: A System Architecture as a Support to a Flexible Annotation Service. In: Proc. of the 6th Thematic Workshop of the EU Network of Excellence DELOS, Cagliari, Italy (June 2004)
4. Goose, S., Lewis, A., Davis, H.: OHRA: Towards an Open Hypermedia Reference Architecture and a Migration Path for Existing Systems. Journal of Digital Information **1**(2) (December 1997)
5. Archer, D.W., Delcambre, L.M.L., Corubolo, F., Cassel, L., Price, S., Murthy, U., Maier, D., Fox, E.A., Murthy, S., McCall, J., Kuchibhotla, K., Suryavanshi, R.: Superimposed Information Architecture for Digital Libraries. In: Proc. of ECDL 2008, 12th European Conference on Research and Advanced Technology for Digital Libraries, Åarhus, Denmark (September 2008)
6. Bottoni, P., Civica, R., Levialdi, S., Orso, L., Panizzi, E., Trinchese, R.: MADCOW: A Multimedia Digital Annotation System. In: Proc. of AVI 2004, Intl. Working Conference on Advanced Visual Interfaces, Gallipoli, Italy (May 2004)
7. Agosti, M., Ferro, N.: A Formal Model of Annotations of Digital Content. ACM Transactions on Information Systems (TOIS) **26**(1) (November 2007)
8. Candela, L., Castelli, D., Ferro, N., Ioannidis, Y., Koutrika, G., Meghini, C., Pagano, P., Ross, S., Soergel, D., Agosti, M., Dobreva, M., Katifori, V., Schuldt, H.: The DELOS Digital Library Reference Model - Foundations for Digital Libraries (December 2007)
9. Norrie, M.C.: An Extended Entity-Relationship Approach to Data Management in Object-Oriented Systems. In: Proc. of ER '93, 12th Intl. Conference on the Entity-Relationship Approach, Arlington, USA (December 1993)
10. Signer, B., Norrie, M.C.: As We May Link: A General Metamodel for Hypermedia Systems. In: Proc. of ER 2007, 26th Intl. Conference on Conceptual Modeling, Auckland, New Zealand (November 2007)
11. Signer, B.: Fundamental Concepts for Interactive Paper and Cross-Media Information Spaces. PhD thesis, ETH Zurich (2006) Dissertation ETH No. 16218.
12. Decurtins, C., Norrie, M.C., Signer, B.: Putting the Gloss on Paper: A Framework for Cross-Media Annotation. New Review of Hypermedia and Multimedia **9** (2003)
13. Bush, V.: As We May Think. Atlantic Monthly **176**(1) (July 1945)