A State of Art Survey on zz-structures

Antonina Dattolo
Dipartimento di Matematica e Informatica
Università degli Studi di Udine, Italy
antonina.dattolo@uniud.it

Flaminia L. Luccio
Dipartimento di Informatica
Università Ca' Foscari, Venezia, Italy
luccio@unive.it

ABSTRACT

Zz-structures are particular data structures capable of representing both hypertextual information and contextual interconnections among different information.
The focus of this paper is to stimulate new research on this topic, by providing, in a state of the art survey, a short description and comparison of all the material that, to the best of our knowledge, is related to zz-structures: informal and formal descriptions, implementations, languages, demonstrations, projects and applitudes of zz-structures; in fact, despite their large use in different fields, the literature lacks of an exhaustive and up-to-date description of them.

1. INTRODUCTION

Zz-structures are xanalogical structures that were first proposed by Ted Nelson [25, 26, 27, 28, 29, 30]: information is stored inside cells which may also be linked to other cells, forming complex graphs.
Research on this topic has been active and different implementations (see, e.g., [1, 2, 3, 7, 8, 23, 16, 17, 18, 29]), applitudes (see, e.g., [5, 6, 9, 10, 22, 24, 31]) and some formal models ([12, 13, 14, 15, 20, 21]) have been proposed. Implementations are distinct from applitudes; in fact, an "applitude is not merely an application that has been designed to work with ZigZag data, but is rather a part of a zz-structure, utilizing a set of views and dimensions in order to express a specific functionality over a particular part of that space" [24].
Our focus is to stimulate interest on this topic. To this aim we shortly illustrate, examine, and compare, in a state of the art survey, all the material that, to the best of our knowledge, is related to zz-structures.

This paper is organized as follows: in Section 2, we give an informal description of zz-structures, summarizing in Section 3 existing formal formulations; Section 4 is dedicated to known implementations. At end of each section, we summarize and compare respectively formal descriptions, implementations and applitudes of zz-structures. Finally, Section 6 concludes the paper.

2. A GENERAL INTRODUCTION

A zz-structure can be thought of as a space filled with cells each of which may contain data (integers, text, images, audio, etc.), and connections to other cells [30].

Sequences of cells connected through links of the same color define *dimensions*, each of which may be composed of different connected chains of cells called *ranks*. An example of zz-structure is shown in Figure 1 where normal, dotted and thick lines represent three different dimensions. The

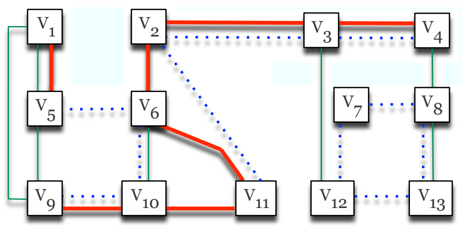


Figure 1: An example of zz-structure.

starting and the ending cells of a rank are called, *headcell* and *tailcell* respectively, and the direction from the starting (ending) to the ending (starting) cell is called *posward* (respectively, *negward*). Each cell has at most one connection in the posward direction, and one in the negward direction of the same color, thus ensuring that all paths are non-branching, and embodying the simplest possible mechanism for traversing links. Dimensions are used to project different structures: ordinary lists are viewed in one dimension; spreadsheets and hierarchical directories in many dimensions.

These structures may be viewed in different ways: e.g., a *raster* is a way of selecting the cells from a structure, while a *view* is a way of placing the cells on a screen. In *two-dimensional rectangular views* cells are placed, using different rasters, on a Cartesian plane where the dimensions increase going down and to the right. The simplest raster is the row and column raster, i.e., two rasters which are the same but rotated of 90 degrees from each other.

The *focus* is a cell that is chosen and placed at the center of the plane (cursor centric view) and may be changed by moving the cursor horizontally and vertically. In a row view I (respectively, in a column view H), a rank is chosen and placed vertically (horizontally), then the related ranks are placed horizontally (vertically). All the cells are denoted by different numbers. An example of H-view, related to zz-structure of Figure 1 and with focus v_2, is shown in Figure 2.
Note that in a view the same cell may appear in different positions as it may represent the intersection of different dimensions.

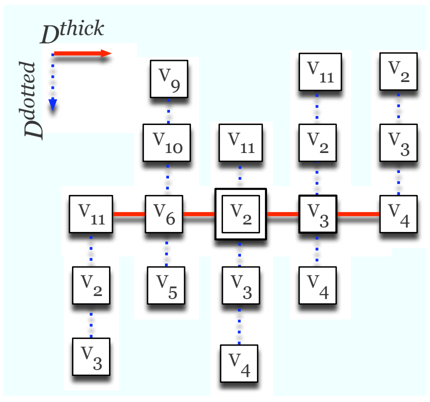


Figure 2: An H-view related to the zz-structure of Figure 1.

3. THE FORMAL MODEL

A detailed formal description of zz-structures is presented in [11]; in this section we summarize the main ideas concerning the development of formal models.

The first formal model was proposed in [20], and later reconsidered and extended in [21]. These works deploy the idea that a zz-structure resembles a graph, where nodes represent cells and sequence of edges represent dimensions. A zz-structure is so defined as a directed multigraph with colored edges (i.e., a graph where pair of nodes may have multiple colored edges connecting them), where each node has at most one outcoming edge and one outgoing edge for each color. Moreover, each of the edge colors corresponds to a different spatial dimension, thus there are paths or cycles of the same color.

These structures can be visualized, via H or I views, as trees. An interesting issue is discussed in [20, 21]: zz-structures are compared with mSpaces and Polyarchies, generating a taxonomy of a graph structure, where zz-structures generalize lists, 2D arrays, trees and also polyarchies; polyarchies generalize mSpace polyarchies. Finally zz-structures and edge-colored multigraph generalize each other. The works [20, 21] have been re-considered, and extended in [13, 14, 15] where the authors propose a better and deeply formalized model, together with some novel concepts. In particular, in [13] the authors re-propose the notion of local orientation used in the field of distributed computing, in order to formally define the concept of posward and negward directions.

Reference	Characteristics
[20, 21]	zz-structure as a directed multigraph with colored edges, edge colors are dimensions, H/I views are trees, taxonomy of graph structure
[13]	deeply formalized model, notion of local ori− entation dynamical addition of connections
[14]	extend notion of view to to n-dimensions H/I views, 3-dimensions extended H/I views, etc.
[15]	displaying of neighbouring views

Table 1: Formal definitions.

The resulting refined model is also the base for the definition of a dynamic actor-based model where actors can add new connections between cells of the zz-structure, i.e., can modify its structure. In [14] the authors also extend the standard notion of view to higher dimensional views, e.g., to n-dimensions H and I views, 3-dimensions extended H and I views, etc.. Finally, in [15] they propose techniques that allow users to display neighbouring views, i.e., views centered in a cell at distance one from the previous one.

Table 1 summarizes all the known formal models.

4. IMPLEMENTATIONS

This section presents the ZigZag$^{\text{TM}}$[1] implementations, dividing them into three categories: ZigZag Virtual Interactive Machines, languages and demonstrations.

4.1 Zzvims

A *zzvim* (term used by Nelson in [30]) is a ZigZag Virtual Interactive Machine, implemented to view and manipulate the universe of zzcells. The literature proposes a list of different zzvims, each of which presents some new features with respect to the previous ones. A list of zzvims that have already been programmed (Azz, Ezz, Gzz and Zzz, and Lzz) is cited in [30].
The first implementation of ZigZag was realized by Andrews Pam in 1997 with *Azz*, a full open-source implementation of zzvim, that offers only row and column views. Initially proposed only for Linux, in 1998 Azz was also able to run under Windows.
From 2000 to 2003, the Hyperstructure Group of the University of Jyväskylä (Finland), directed by Tuomas J. Lukka, implemented in Java a collaborative, open-source version *GZigZag* [17], successively called Gzz and finally replaced by the Fenfire project [16]. Gzz extends the features of Azz, by offering several views (not only row and column views), moreover, it accepts zz-structures via XML, and represents them using RDF (Resource Description Framework) graphs.
In the same years, Les Carr of the University of Southampton implemented, in different platforms, i.e., HTML/XML/XSLT, *Lzz* [8], a prototype that works client-side with various results in different browsers and implements much of the ZigZag infrastructure directly as JavaScript variables rather than ZigZag cells.
Successively, in 2003, Mikhail Seliverstov generated a re-implementation of Azz, called *Ezz*. Implemented in Java, Ezz runs on Mac and Windows and puts out and accepts XML as Gzz.
A successive version by Jeremy Alan Smith is *Zzz*: Developed in C with Python and running on many different platforms, i.e., Windows, Linux and Mac, it supports the routine-by-routine conversion of Azz and it is extensible in Python. It also offers some 3D views in OpenGL.
A research project on bio-informatics [23] (by Adam Moore, Tim Brailsford, and Helen Ashman of the University of Nottigham) highlights some limitations of the Gzz platform for real applications: Gzz is not designed to handle large volumes of data; it is dependent upon an obsolete version of Java; and it is no longer maintained by the original developers. On this basis, [23] develops a new server-based imple-

[1]"ZigZag" is a registered trademark in the U.S.A. for the zzstructure-based software of Project Xanadu.

Name	Chracteristics	Platforms
Azz	Only row/ column views	Linux/Windows
Gzz [17]	several views, accepts zz-structure via XML, Java	Unix and MS-Windows
Lzz [8]	applied client side	HTML/XML/XSLT
Ezz	reimplementation of Azz, accepts zz-structure via XML	Java, on Mac/Windows
Zzz	C with Python, Windows/Linux/Mac OS	Routine-by-routine conversion of Azz, extensible in Python some 3D views in OpenGL
Mantra server [23]	C++, client-server model, new language Mantra	Mac OSX/Ubuntu Linux
Diablo [1]	Python, replacement back-end for Mantra	Mac OSX/ Ubuntu Linux
BigBag [4]	implementation in XML of an EAD finding aid	Windows, Linux and Mac
Rzz [2]	industrial version of ZigZag	Unix and MS-Windows

Table 2: Zzvim implementations.

mentation of ZigZag, called the *Mantra server* (2002-2006) that uses a client-server model, and introduces a new language, called Mantra. The Mantra server is written in C++ and runs on Mac OSX and Ubuntu Linux.
In order to demonstrate the ZigZag navigation in the mantraf client (Flash) without the need to patch and build from source, in 2008 it has been developed *Diablo* [1], a simple ad-hoc replacement back-end for Mantra with a single-user server and partial query processor implemented in Python. Diablo is not a database and it stores zz-structures using JSON (JavaScript Object Notation).
A new prototype, called *BigBag*, has been created by a project [4] directed by Ian G. Anderson of the University of Glasgow; BigBag is a ZigZag implementation in XML of an EAD finding aid (Gateway to Archives of Scottish Higher Education - GASHE) to support a flexible visualization created in Flash.
Currently on the pages [2] of Xanadu project it is announced an industrial version of ZigZag, called *Rzz*. Table 2 summarizes the zzvim implementations.

4.2 Languages

Different zz-structured languages [18] have been proposed by a research group at the University of Jyväskylä (Finland). One of the first ZigZag-based languages is *Thales Clang* whose main feature is that everything is an expression (ranks, dimensions, etc.) thus can be evaluated to a value. This language has however never been fully implemented. *Flowing Clang*, is the first ZigZag scripting language in Gzz and the first ZigZag language implementation that features a debugger showing the entire operation of the program inside the zz-structure. The third implementation of GZigZag is *Clasm* and is used to implement some essential features of the GZigZag client.

Finally, *Nuzzl* (NU ZigZag Language), is a spatial programming language in ZigZag space implemented by Jeremy Smith (see [29]). Table 3 summarizes ZigZag languages.

4.3 Demos for Web browsers

Different demonstrations, ranging in a big variety of fields, have been developed by Les Carr of the University of Southampton [7]; they use an XML/XSL/JavaScript implementation of Zigzag in a Web browser, achieving interaction by dynamic HTML and storing the data either as XML or JavaScript declarations embedded in an HTML framework:

Name	Characteristic
Thales Clang [18]	everything is an expression, never fully implemented
Flowing Clang [18]	ZigZag scripting language in Gzz, debugs showing operation of the program inside the zz-structure
Clasm [18]	implements essential features of the GZigZag client
Nuzzl [29]	spatial programming language in ZigZag space

Table 3: Zigzag languages.

- the *Schedule* Demo highlights how new dimensions can make personal record-keeping easier;
- the *Holm Family* Demo models a complicated structure such as the family tree using GZigZag; in GZigZag, the user only needs to create new cells and connect them along different dimensions, in usual computer systems instead this requires the generation of a specific dedicated program;
- the *PicZag* Demo is a test which just demonstrates that in ZigZag it is possible use any kind of data: Pictures, sound or videos.
- the *Function* Demo shows how a function can be evaluated as part of the automatic rendering of a cell, or when requested by a user.
- the *London Underground* Demo contains the basic infrastructure of the Central London underground routes in terms of the major lines (cells are train stations, dimensions are train lines). It also links underground stations to attractions above, and different touristic attractions above ground.

Table 4 summarizes the demos.

5. APPLITUDES

An interesting characteristic of zz-structures is their flexibility: data may be stored and efficiently retrieved by following the ”meaning” of each dimension. E.g., in an e-learning context, a dimension may represent a studying topic, let us say ancient history, thus following this particular dimension the

Name	Characteristic
Schedule	Personal record-keeping
Holm Family	A family tree
PicZag	Test proving the use of any kind of data
Function	Function evaluation
London Underground	London underground routes

Table 4: Zigzag demos.

user may navigate and extract all the information related to ancient Greeks. Another nice feature of zz-structures is that they allow to display cells, i.e., data, on the space, providing a nice overview and an easy reading of its contents. Given its powerful characteristics, these structures have been exploited to solve many different real-world problems. In this section, we give a very short description of the ones that, to the best of our knowledge, have been proposed in the literature. As the reader will notice, these problems range in very different and not closely related fields. As we have mentioned in the introduction, we will use the term applitude to refer to "a part of a zz-structure, utilizing a set of views and dimensions in order to express a specific functionality over a particular part of that space" [24].

Bionformatics Workspace. Bioinformatics is a field in which computer science techniques are applied to biology. In biology information is wide, complex, interrelated, thus there is a very strong need for efficient storage and query mechanisms, and zz-structures easily provide all these features. In particular, bioinformatics aims at properly reorganizing this information in order to, e.g., devise treatments for diseases. Two different applications of zz-structures to bioinformatics have been presented in [22, 24].
In [22], the authors propose the use zz-structures for the structure/binding prediction of new molecules that are necessary for the design of new drugs. To this aim, it is important to define efficient mechanisms for the choice of a molecule that will mimic an interaction at a specific place (called "active site") with a particular protein.
In [24] the authors propose a Bioinformatics Workbench, i.e., an information manager for bioinformatics. They store different information in a zz-structure and organize it in what they call *zzLists.*

Grid models. The massive use of the Web's functionality and the need to gather enough computational resources for running different applications at different heterogeneous locations are some of the aspects that have led to the birth of the grid infrastructure.
In [13], the authors concentrate on AZ, a grid model based on an extension <u>A</u>ctor-based of <u>Z</u>z-structures. They provide a formal description of virtual organizations (VOs), meaningful part of a particular grid infrastructure, the data grid. The choice of modeling with zz-structures some fundamental parts of a grid, such as the organizations, responds to key aspects of grids, that are, e.g., composition of resources, closure and fractal properties [19]; also, zz-structures are minimalist and may be defined in a recursive way, by composition, generating local and global grids, or a hierarchy of grids, and larger grids can be constructed by composing smaller (perhaps local) grids.

Cellular phones. Typically, cellular phones contain several items of information, almost all of which are related to each other: contacts are associated to individual phone numbers; phone numbers are related to both individual phone conversations and SMS texts; contacts also have addresses; and so on. Although the intertwined relationships of these items are complex, in the current generations of phones, this information is stored using a simplistic model, and often the underlying connectivity is severely constrained.
zzPhone [24] is a ZigZag phone applitude which exploits the powerful viewing features of zz-structures; by selecting an item of information and the connected dimensions of interest, it is possible to see the item in the desired context and manage a wide set of connections, not directly accessible in traditional systems.

Web-based education. Web-based education has become a very important area of educational technology and a challenge for semantic Web techniques. Web-based education enables *learners* and *authors* (teachers) to access a wide quantity of continuously updated educational sources. In order to simplify the learning process of learners and the course organization process of authors, it is important to offer them tools to: 1) Identify the collection of "interesting" documents; 2) store the found collection of documents in adequate structures; create personalized adaptive paths and views for learners.
In this area, we discuss of two works [14] and [5], that exploit the power of zz-structures for the organization and retrieval of information.
In [14] the authors concentrate their attention on point 2); in particular, they assume that an author has a collection of available documents on a given topic that have to be organized in concept maps, suitable for different learners. Authors need adequate tools to organize documents in a concept space, and to create semantic interconnections and personalized maps. The proposed solution is based on the use of zz-structures, and extends the concepts of H and I views from a number 2 towards a number $n > 2$ of dimensions, in order to present a new concept map model for e-learning environments. [5] provides a tool for supporting the authors in their tasks of selecting and grouping the learning material. The 'à la' (Associative Linking of Attributes) in Education enhances the search engine results by extracting the attributes (keywords and document formats) from the text. The relationships between the attributes are established and visualized using the ZigZag principles.

Virtual museum tours. Guided tours inside virtual museums have strong similarities to standard information systems searches. In virtual learning museums, users explore a structured hyperspace with context-adapted narration, interacting with a system that recreates a real life museum tour guided by a real museum guide. Generally, museums have sites that assume standard figures of users and do not allow personalized visits, based on different interests, backgrounds, etc.
In [15] the authors present an application to the user tours of virtual museums, that exploits the retrieval power of zz-structures. In particular, the authors consider the formal model presented in [14] for the visualization of personalized views, recreate it in the context of virtual museum tours, and extend it in order to allow the users to interact with

the system and (partially) choose and personalize the path to follow during their navigation. That is, they show how users may first create and display personalized H-views, and then personalize their paths, by deciding to which neighboring view they will move, what dimension they would like to add/remove, and so on. During this navigation process, the users can also store the information they find interesting in an album, in order to create a personal, re-usable workspace.

Archival finding aids. As more archival finding aids, of increasing complexity, become available on-line the difficulty of browsing and navigating the results increases. This is particularly the case when the finding aids are implemented in EAD (Encoded Archival Description) a hierarchical organization of archival collections, typically implemented in XML. In part, navigational difficulties are inherent in any hierarchical structure, but also they are a symptom of the lack of innovation in visualizing archival information.
In [4], the author develops a novel approach for structuring and visualizing archival information by applying a flexible visualization interface to an EAD finding aid that has been transformed into Ted Nelson's zz-structure. BigBag uses a development of the XML zz-structure produced for London Underground demo (see Section 4.3).

Electronic Editions of musical works. Archimedes [6] is a project devoted to the preservation and the access to electro-acoustic music documents. These documents cluster an extended set of data that provide essential insights in production schemes and copy generations, and need to be preserved and compared.
Archimedes proposes the application of a new actor-based extension of zz-structures. The cooperation activity of different actor classes allows to create innovative, graph-centric browsing perspectives for the users and to offer to them authoring tools for the runtime creation of new virtual sources.

Personal information spaces.

In [9] the authors present an innovative architecture, conceived in terms of a multi-agent systems and aimed at creating, managing and sharing personal information spaces. Data and knowledge may be directly added by users, but also collected and structured with the support of content retrieval, filtering and automatic tagging techniques. Conceptual spaces organize personal information spaces using zz-structures, and propose, by means graph-centric views, contextual interconnections among heterogeneous information. The structure of each conceptual space, constituted by a set of links to items (cells and edges) included into the Knowledge Base and a set of private items, is stored into the User Profile (UP). A UP is assigned to each registered user; it is used to store, in addition to data representing user's conceptual spaces, user information collected both implicitly and explicitly.

Sentiment Classification. In [10], the authors consider the problem of tracking the opinion polarity, in terms of positive or negative orientation, expressed in documents written in natural language and extracted from a heterogeneous set of Web sources. More specifically, they focus their attention on the movie reviews domain and are interested in evaluating the performance obtained by a set of high performance opinion polarity classifiers for the Italian language. Classification of polarity expressed by the input documents is achieved by means of several sets of specialized autonomous or interacting agents, devoted, respectively, to document gathering, classification and visualization. In particular the results of opinion analysis are represented by means of a graphical interface, where a multi agent based implementation of zz-structures is exploited to offer graph-centric views and navigation of results. The specific experimental evaluation performed so far shows an accuracy level, which is higher than previous results reported in the literature.

Associative writing tool. AWT (Associative writing tool) [31] is a tool for supporting the writing process. Textual artifacts are organized in different layers, have explicit links connecting them, and implicit Xanadu links between their contents. The tool supports spatial positioning of textual artifacts that are stored in zz-structure cells, but depending on the type of connections they can be atomic elements or parts of a set. Table 4 summarizes all the known applitudes.

Applitude	Chracteristics
Bioinformatics Workspace [22, 24]	structure binding prediction of new molecules, a Bio-informatics Workbench
Grid models [13]	Actor-based grid model
Cellular phones [24]	ZigZag phone
Web-based education [14, 5]	concept map model e-learning
Virtual museum tours [15]	visualization/navigation in virtual museums
Archival finding aids [4]	structuring/ visualizing archival data
Electroninc Editions of musical works [6]	preservation/access to electro-acoustic music documents
Personal information spaces [9]	creating/managing/ sharing personal
Sentiment Classification [10]	opinion polarity
Associative writing tool [31]	writing process support

Table 5: Applitudes.

6. CONCLUSION

In this paper we have presented a state of art survey of formal descriptions, implementations and applications of zz-structures. As we have shown these structures are widely accepted and adopted. Implementations have been developed for different platforms (Windows, Linux, Mac), applitudes have been designed in different fields, from bioinformatics, to e-learning, etc.. In our opinion, the state of art is very promising and implementations and applitudes of the future promise to be very intuitive and accessible in different fields. We hope this survey will stimulate further discussion on the topic.

7. REFERENCES

[1] Diablo. https://code.launchpad.net/python-diablo.
[2] Rzz. http://www.xanadu.com/zigzag/.
[3] I. Anderson. Bigbag. http://www.hatii.arts.gla.ac.uk/research/visual/visual.htm.

[4] I. Anderson. From ZigZag^{TM} to BigBag: Seeing the wood and the trees in online archive finding aids. In *Proceedings of Workshop on New Forms of Xanalogical Storage and Function*, June 29 2009.

[5] M. Andric, V. Devedzic, W. Hall, and L. Carr. Keywords linking method for selecting educational web resources à la zigzag. *International Journal of Knowledge and Learning*, 3(1):30–45, 2007.

[6] S. Canazza and A. Dattolo. Open, dynamic electronic editions of multidimensional documents. In *IASTED Proceedings of European Conference on Internet and Multimedia Systems and Applications*, pages 230–235. Chamonix (France), March 14-16 2007.

[7] L. Carr. http://users.ecs.soton.ac.uk/lac/zigzag/.

[8] L. Carr. Zigzag for web browsers, 2001. http://www.ecs.soton.ac.uk/∼lac/zigzag.

[9] P. Casoto, A. Dattolo, F. Ferrara, N. Pudota, P. Omero, and C. Tasso. Toward making agent uml practical: a textual notation and a tool. In *Proceedings of the Workshop on Adaptation for the Social Web, 5th ACM Int. Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 14–23. Germany, 29 July - 1 August 2008.

[10] P. Casoto, A. Dattolo, and C. Tasso. Sentiment classification for the italian language: A case study on movie reviews. *Journal of Internet Technology, Special issue on Intelligent Agent and Knowledge Mining*, 9(4):365–373, 2008.

[11] A. Dattolo and F. Luccio. A formal description of zigzag-structures. In *Proceedings of Workshop on New Forms of Xanalogical Storage and Function*, June 29 2009.

[12] A. Dattolo and F. L. Luccio. A New Concept Map Model for E-learning Environments. Lecture Notes in Business Information Processing, ISSN 1865-1348, LNBIP 18, 2009.

[13] A. Dattolo and F. L. Luccio. A new actor-based structure for distributed systems. In *International Conference on Hypermedia and Grid Systems (HGS07)*, pages 195–201. Opatija, Adriatic Coast (Croatia), May 21-25 2007.

[14] A. Dattolo and F. L. Luccio. Formalizing a model to represent and visualize concept spaces in e-learning environments. In *Proceedings of the 4th Webist International Conference (WEBIST08)*, pages 339–346. Funchal, Madeira, Portugal, 4-7 May 2008.

[15] A. Dattolo and F. L. Luccio. Visualizing personalized views in virtual museum tours. In *International Conference on Human System Interaction (HSI08)*, pages 109–114. Krakow, Poland, 25-27 May 2008.

[16] Fenfire. http://fenfire.org/.

[17] GZigZag. Home page. http://gzigzag.sourceforge.net.

[18] A. J. Kaijanaho and B. Fallenstein. Totally different structural programming programming languages in zigzag. In *Proceedings of the First International ZigZag Conference, part of ACM Hypertext Conference 2001.* Aarhus, Denmark, August 2001. http://www.mit.jyu.fi/antkaij/plinzz.html.

[19] F. Manola and C. Thompson. *Characterizing Computer-Related Grid Concepts.* Object Services and Consulting, Inc., 1999.

[20] M. McGuffin. A graph-theoretic introduction to ted nelson's zzstructures. January 2004. http://www.dgp.toronto.edu/∼mjmcguff /research/zigzag/.

[21] M. McGuffin and m. c. schraefel. A comparison of hyperstructures: Zzstructures, mspaces, and polyarchies. In *Proceedings of the 15th ACM Conference on Hypertext and Hypermedia (HT'04)*, pages 153–162. Santa Cruz, California, USA, August 9-13 2004.

[22] A. Moore and T. Brailsford. Unified hyperstructures for bioinformatics: escaping the application prison. *Journal of Digital Information*, 5(1):Article No.254, 2004.

[23] A. Moore, T. Brailsford, and H. Ashman. Zigzag for bioinformatics, 2002-2006. http://www.cs.nott.ac.uk/Research/webtech/zzbio/.

[24] A. Moore, J. Goulding, T. Brailsford, and H. Ashman. Practical applitudes: Case studies of applications. In *Proceedings of the 15th ACM Conference on Hypertext and Hypermedia (HT'04)*, pages 143–152. Santa Cruz, California, USA, August 9-13 2004.

[25] T. H. Nelson. What's on my mind. In *Invited talk at the first Wearable Computer Conference.* Fairfax VA, May 12-13 1998. http://www.xanadu.com.au/ted/zigzag/xybrap.html.

[26] T. H. Nelson. Welcome to zigzag. 1999. http://xanadu.com/zigzag/tutorial/ZZwelcome.html.

[27] T. H. Nelson. Xanalogical structure, needed now more than ever: parallel documents, deep links to content, deep versioning, and deep re-use. *ACM Computing Surveys*, 31(4:33), 1999.

[28] T. H. Nelson. Zigzag (tech briefing): Deeper cosmology, deeper documents. In *Proceedings of the 12-th ACM conference on Hypertext and Hypermedia (HT'01)*, pages 261–262. University of Aarhus, Aarhus, Denmark, August 14-18 2001.

[29] T. H. Nelson. Structure, tradition and possibility. In *Proceedings of 14th ACM Conference on Hypertext and Hypermedia (HT'03).* Nottingham, United Kingdom, August 26-30 2003. http://www.ht03.org/keynote-ted.html.

[30] T. H. Nelson. A cosmology for a different computer universe: data model mechanism, virtual machine and visualization infrastructure. *Journal of Digital Information: Special Issue on Future Visions of Common-Use Hypertext*, 5(1):298, 2004.

[31] K. Wideroos. Awt (associative writing tool): supporting writing process with a zigzag based writing tool - work in progress. In *Proceedings of the 12-th ACM conference on Hypertext and Hypermedia (HT'01)*, pages 35–36. University of Aarhus, Aarhus, Denmark, August 14-18 2001.

A formal description of zz-structures

Antonina Dattolo
Dipartimento di Matematica e Informatica
Università degli Studi di Udine, Italy
antonina.dattolo@uniud.it

Flaminia L. Luccio
Dipartimento di Informatica
Università Ca' Foscari, Venezia, Italy
luccio@unive.it

ABSTRACT

The focus of this paper is on particular and innovative structures for storing, linking and manipulating information: the zz-structures.

In the last years, we worked at the formalization of these structures, retaining that the description of the formal aspects can provide a better understanding of them, and can also stimulate new ideas, projects and research.
This work presents our contribution for a deeper discussion on zz-structures.

1. INTRODUCTION

At the first Wearable Computer Conference [18], Ted Nelson proposed "a prototype that implements some interesting ideas, intended to lead to such a new kind of simple and unified world, possibly to permit the unification of everything that non-computer people want to do with computers". The software was ZigZag$^{\text{TM}}$[1], a "principled system of interconnections" [19] and a new, graph-centric system of conventions for data and computing, based on the so-called zz-structures.

Nelson writes [19]: "The ZigZag system is very hard to explain, especially since it resembles nothing else in the computer field that we know of, except perhaps a spreadsheet cut into strips and glued into loops". Zz-structures are hyper-orthogonal, non-hierarchical structures for storing, linking and manipulating data. Intuitively, data are contained in cells that are connected by crossing dimensions forming a structure that resembles a spreadsheet but contains intricate connections [18, 19, 20, 21, 22, 23].

Zz-structures have been seen from many different perspectives: merely a large variety of implementations and *applitudes*, and some formal models have been proposed. We specify that we use the term "applitude" instead of "application". Nelson states [23]: "Instead of "applications", separated zones of function and usage connected by the narrow channels of clipboard and file export/import, we have "applitudes" which are deeply interconnected to the whole, amongst themselves, and amongst their parts". The main difference is that, unlike applications, applitudes exploit the dense and intricate connections among the information contained in a zz-structure. Applitudes can also be combined with each other and are not "walled off" from the rest of the system. Table 1 collects a list of research contributions (applitudes and implementations), developed using zz-structures.

Year	Domain	Reference
1984-2004	zz-vims (Azz, Ezz, Gzz and Zzz, and Lzz)	[23]
2001	Associative writing	[24]
2001	Various Demos	[4]
2002-2006	Bionformatics	[16, 17]
2004	Cellular phones	[17]
2006-2007	Archival finding aids	[1]
2007	Grid Systems	[9]
2007-2009	Web-based Education	[2, 10, 12]
2007-2009	Audio Archives	[3, 7]
2008	Personal Information Space	[5]
2008	Sentiment Classification	[6]
2008	Virtual Museums	[11]
2009	Publication Sharing Systems	[8]

Table 1: Models and applitudes

Zz-structures are "a generalized representation for all data and a new set of mechanisms for all computing" [23]: innovative structures for storing, linking and manipulating information.

The intention of this work is to present our contribution to the formalization of zz-structures and to encourage further discussion. In our opinion, the description of a formal model can provide a deeper understanding of the model and can stimulate new ideas, projects and research.

The paper is organized as follows: in Section 2, we give an informal description of zz-structures, that prepares the reader to the formal model presented in Section 3. We conclude in Section 4 with a brief discussion and future look.

2. THE ZZ-STRUCTURES

Zz-structures introduce a new, graph-centric system of conventions for data and computing [19]. A zz-structure can be thought of as a space filled with cells. Each cell may have a content (such as integers, text, images, audio, etc.), and it is called *atomic* if it contains only one unit of data of one type, or it is called *referential* if it represents a package of different cells [23].

Cells are connected together with links of the same color into linear sequences called *dimensions*. A single series of cells connected in the same dimension is called *rank*, i.e., a rank is in a particular dimension and a dimension may contain many different ranks. The starting and the ending

[1]"ZigZag" is a registered trademark in the U.S.A. for the zzstructure-based software of Project Xanadu.

cells of a rank are called, *headcell* and *tailcell*, respectively, and the direction from the starting (ending) to the ending (starting) cell is called *posward* (respectively, *negward*). For any dimension, a cell can only have one connection in the posward direction, and one in the negward direction. This ensures that all paths are non-branching, and thus embodies the simplest possible mechanism for traversing links. Dimensions are used to project different structures: Ordinary lists are viewed in one dimension; spreadsheets and hierarchical directories in many dimensions.

There are many different ways to view these structures: A *raster* is a way of selecting the cells from a structure, while a *view* is a way of placing the cells on a screen. *Generic views* are designed to be used in a big variety of cases and usually show only few dimensions or few steps in each dimension. Among them the most common views are the *two-dimensional rectangular views*: The cells are placed, using different rasters, on a Cartesian plane where the dimensions increase going down and to the right. The simplest raster is the row and column raster, i.e., two rasters which are the same but rotated of 90 degrees from each other. A cell is chosen and placed at the center of the plane (cursor centric view). The chosen cell, called *focus*, may be changed by moving the cursor horizontally and vertically. In a row view I, a rank is chosen and placed vertically. Then the ranks related to the cells in the vertical rank are placed horizontally. Vice versa, in the column view H, a rank is chosen and placed horizontally and the related ranks are placed vertically. All the cells are denoted by different numbers. Note that in a view the same cell may appear in different positions as it may represent the intersection of different dimensions.

3. FORMALIZING ZZ-STRUCTURES

As we have mentioned in the introduction, zz-structures have been studied from many perspectives, and many implementations and applitudes have been proposed in different fields. Some research has also been provided towards a formal definition of these structures. In the seminal work of [14], we find the first proposal of formalizing these structures in terms of graphs; this work has been extended and motivated in [15], where the authors use this formalization in order to compare these structures with mSpaces and Polyarchies. This comparison is done building a taxonomy as a subsumption diagram, a subsumption being a generalization of something. The general result is that zz-structures subsume lists, 2D arrays, trees and also polyarchies; polyarchies subsume mSpace polyarchies. Finally zz-structures and edge-colored multigraph subsume each other.

Later, in [9, 10, 11, 12] we have revisited and redefined into more precise mathematical terms, the definitions of zz-structures provided in [14, 15], and have also introduced new notions such as the one of local orientation, borrowed from the field of distributed computing, and required to provide a formal definition of local posward and negward directions. The formal model is a requirement in [9] for the construction of an actor-based model where actors can add new connections between cells of the zz-structure, i.e., dynamically modifying its structure. Moreover, this formalization has provided interesting tools for the introduction of new formal concepts, such as the extension of the standard notion of view to higher dimensional views (e.g., n-dimensions H and I views, 3-dimensions extended H and I views, etc.) [10, 12], and for the definition of techniques that allow users

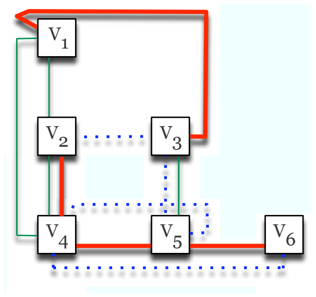


Figure 1: An example of zz-structure.

to display neighbouring views (i.e., views centered in a cell at distance one from the previous one) [11].

The aim of this section is to summarize, analyze, and illustrate with new examples, discuss and relate all the above proposals in order to provide a general overview of the formal model, at least for the original concepts defined by Nelson in [18].

Zz-structure.

In [14, 15] the authors define a zz-structure as a directed multigraph with colored edges (i.e., a graph where pair of nodes may have multiple colored edges connecting them), where each node has at most one outcoming edge and one outgoing edge for each color. In [9] the authors further formalize these concepts (choosing however bidirectional links) as follows.

Consider an *edge-colored multigraph* $ECMG = (MG, C, c)$ where: $MG = (V, E, f)$ is a multigraph composed of a set of *vertices* V, a set of *edges* E and a surjective function $f : E \to \{\{u, v\} \mid u, v \in V, u \neq v\}$. C is a set of colors, and $c : E \to C$ is an assignment of colors to edges of the multigraph. Finally, $deg(x)$ (respectively, $deg_k(x)$) denotes the number of edges incident to x, (respectively, of color c_k).

Definition 1. **: Zz-structure** - A *zz-structure* is an edge-colored multigraph $S = (MG, C, c)$, where $MG = (V, E, f)$, and $\forall x \in V$, $\forall k = 1, 2, ..., |C|$, $deg_k(x) = 0, 1, 2$. Each vertex of a zz-structure is called *zz-cell* and each edge a *zz-link*. The set of isolated vertices is $V_0 = \{x \in V \ : deg(x) = 0\}$.

An example of a zz-structure is shown in Figure 1. Normal, dotted and thick lines represent different colors.

Dimension.

In [15] the authors state that "each of the edge colors correspond to a different spatial dimension". This concept, together with all the following definitions, is further formalized in [9] where the authors state that an alternative way of viewing a zz-structure is a union of subgraphs, each of them containing edges of a unique color.

Proposition 1. Consider a set of colors $C = \{c_1, c_2, ..., c_{|C|}\}$ and a family of indirect edge-colored graphs $\{D^1, D^2, ..., D^{|C|}\}$, where $D^k = (V, E^k, f, \{c_k\}, c)$, with $k = 1, ..., |C|$, is a graph such that: 1) $E^k \neq \emptyset$; 2) $\forall x \in V$, $deg_k(x) = 0, 1, 2$. Then, $S = \bigcup_{k=1}^{|C|} D^k$ is a zz-structure.

Definition 2. **: Dimension** - Given a zz-structure $S = \bigcup_{k=1}^{|C|} D^k$, then each graph D^k, $k = 1, \dots, |C|$, is a distinct *dimension* of S.

The zz-structure of Figure 1 contains three dimensions D^{thick}, D^{normal} and D^{dotted}, respectively represented by thick, normal, and dotted lines and shown in Figure 2. In turn, each

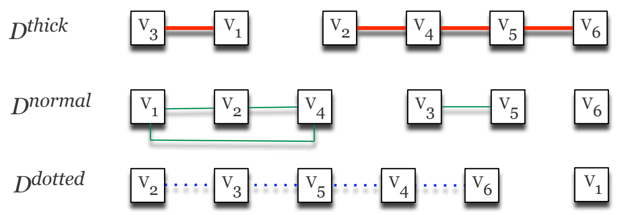


Figure 2: The dimensions D^{thick}, D^{normal} and D^{dotted}.

dimension is composed by a set of connected components and a set (eventually empty) of isolated vertices. As an example, D^{normal} is composed of a cycle $\{v_1, v_2, v_4, v_1\}$, a path $\{v_3, v_5\}$, and one isolated vertex v_6, while D^{thick} is composed of two distinct paths $\{v_3, v_1\}$, $\{v_2, v_4, v_5, v_6\}$, and no isolated vertex.

Rank.

Each "series of cells connected sequentially in any dimension" identifies a *rank* [23].

Definition 3. **: Rank** - Consider a dimension $D^k = (V, E^k, f, \{c_k\}, c)$, $k = 1, \dots, |C|$ of a zz-structure $S = \cup_{k=1}^{|C|} D^k$. Then, each of the l_k ($l_k \geq 1$) connected components of D^k is called a *rank*.

Thus a rank is an indirect graph $R_i^k = (V_i^k, E_i^k, f, \{c_k\}, c)$ ($i = 1, 2, \dots, l_k$) such that 1) $E_i^k \in E^k$ and $E_i^k \neq \emptyset$; 2) $\forall x \in V_i^k$, $V_i^k \in V$, $deg_k(x) = 1, 2$.

Definition 4. **: Ringrank** - A *ringrank* is a rank R_i^k, where $\forall x \in V_i^k$, $deg_k(x) = 2$.

In Figure 2, the dimension D^{thick} has two ranks: $\{v_3, v_1\}$ and $\{v_2, v_4, v_5, v_6\}$; the dimension D^{normal} has one rank $\{v_3, v_5\}$, and one ringrank $\{v_1, v_2, v_4, v_1\}$.

Cells and their orientation.

A vertex [9] has local orientation on a rank if each of its (1 or 2) incident edges has assigned a distinct label (1 or -1). More formally (see also [13]):

Definition 5. **: Local orientation** - Consider a rank $R_i^k = (V_i^k, E_i^k, f, \{c_k\}, c)$ of a zz-structure $S = \cup_{k=1}^{|C|} D^k$. Then, $\exists$ a function $g_x^i : E_i^k \to \{-1, 1\}$, such that, $\forall x \in V_i^k$, if $\exists y, z \in V_i^k : \{x, y\}, \{x, z\} \in E_i^k$, then $g_x^i(\{x, y\}) \neq g_x^i(\{x, z\})$. Thus, we say that each vertex $x \in V_i^k$ has a *local orientation* in R_i^k.

Definition 6. **: Posward and negward directions** - Given an edge $\{a, b\} \in E_i^k$, we say that $\{a, b\}$ is in a *posward* direction from a in R_i^k, and that b is its *posward cell* iff $g_a^i(\{a, b\}) = 1$, else $\{a, b\}$ is in a *negward* direction and a is its *negward cell.* Moreover, a path in rank R_i^k follows a posward (negward) direction if it is composed of a sequence of edges of value 1 (respectively, -1).

Head and tail cells.

If we focus on a vertex x, $R_i^k = \dots x^{-2} x^{-1} x x^{+1} x^{+2} \dots$ is expressed in terms of negward and posward cells of x: x^{-1} is the negward cell of x and x^{+1} the posward cell. We also assume $x^0 = x$. In general x^{-i} (x^{+i}) is a cell at distance i in the negward (posward) direction.

Definition 7. **: Headcell and tailcell** - Given a rank $R_i^k = (V_i^k, E_i^k, f, \{c_k\}, c)$, a cell x is the *headcell* of R_i^k iff $\exists$ its posward cell x^{+1} and $\nexists$ its negward cell x^{-1}. Analogously, a cell x is the *tailcell* of R_i^k iff $\exists$ its negward cell x^{-1} and $\nexists$ its posward cell x^{+1}.

Views.

In the following, we denote with $x \in R_{(x)}^a$ the rank $R_{(x)}^a$ related to vertex x, of color c_a.

Definition 8. **: H-view** - Given a zz-structure $S = \cup_{k=1}^{|C|} D^k$, where $D^k = \cup_{i=1}^{l_k} (R_i^k \cup V_0^k)$, and where $R_i^k = (V_i^k, E_i^k, f, \{c_k\}, c)$, the H-view of size $l = 2m + 1$ and of *focus* $x \in V = \cup_{i=0}^{l_k} V_i^k$, on main vertical dimension D^a and secondary horizontal dimension D^b ($a, b \in \{1, ..., l_k\}$), is defined as a tree whose embedding in the plane is a partially connected colored $l \times l$ mesh in which:

- the central node, in position $((m+1), (m+1))$, is the focus x;
- the horizontal central path (the $m + 1$-*th* row) from left to right, focused in vertex $x \in R_{(x)}^b$ is: $x^{-g} \dots x^{-1} x x^{+1} \dots\ x^{+p}$ where $x^s \in R_{(x)}^b$, for $s = -g, \dots, +p$ ($g, p \leq m$).
- for each cell x^s, $s = -g, \dots, +p$, the related vertical path, from top to bottom, is: $(x^s)^{-g_s} \dots (x^s)^{-1} x^s (x^s)^{+1}\ \dots (x^s)^{+p_s}$, where $(x^s)^t \in R_{(x^s)}^a$, for $t = -g_s, \dots, +p_s$ ($g_s, p_s \leq m$).

Intuitively, the H-view extracts ranks along the two chosen dimensions. Note that, the name H-view comes from the fact that the columns remind the vertical bars in a capital letter H. Observe also that the cell x^{-g} (in the $m + 1$-*th* row) is the *headcell* of $R_{(x)}^b$ if $g < m$ and the cell x^{+p} (in the same row) is the *tailcell* of $R_{(x)}^b$ if $p < m$. Analogously, the cell x^{-g_s} is the headcell of $R_{(x^s)}^a$ if $g_s < m$ and the cell x^{+p_s} is the tailcell of $R_{(x^s)}^a$ if $p_s < m$. Intuitively, the view is composed of $l \times l$ cells unless some of the displayed ranks have their headcell or tailcell very close (less than m steps) to the chosen focus.

As an example consider Figure 3 left that refers to the zz-structure of Figure 1. The main vertical dimension is D^{dotted} and the secondary horizontal dimension is D^{thick}. The view has size $l = 2m + 1 = 5$, the focus is the node v_5, the horizontal central path is $\{v_2, v_4, v_5, v_6\}$. The vertical path related to v_4 is $\{v_3, v_5, v_4, v_6\}$, that is v_6 is the tailcell of the rank as $p_s = 1 < m = 2$.

The I-view can be defined analogously to the H-view [9]. An example of I-view with main horizontal dimension D^{dotted}, secondary vertical dimension D^{thick}, size $l = 5$ and focus v_5 is shown in Figure 3 right.

We can now extend the known definition of H and I views to a number $n > 2$ of dimensions [10]. Intuitively, we will

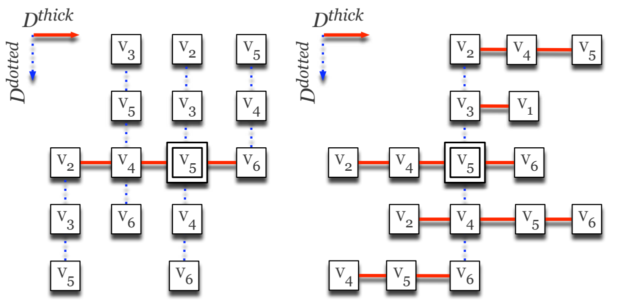


Figure 3: An example of H-view and I-view.

build $n-1$ different H-views (respectively, I-views), centered in the same focus, with a fixed main dimension and a secondary dimension chosen among the other $n-1$ dimensions. Formally:

Definition 9. **n-dimensions H-view** - Given a zz-structure $S = \cup_{k=1}^{|C|} D^k$, where $D^k = \cup_{i=1}^{l_k}(R_i^k \cup V_0^k)$, and where $R_i^k = (V_i^k, E_i^k, f, \{c_k\}, c)$, the *n-dimensions H-view* of size $l = 2m+1$ and of *focus* x, $x \in V = \cup_{i=0}^{l_k} V_i^k$, on dimensions $D^1, D^2, \ldots, D^n$ is composed of $n-1$ rectangular H-views, of main dimension D^1 and secondary dimensions D^i, $i = 2, \ldots, n$, all centered in the same focus x.

Analogously, we can define an n-dimensions I-view. An example of a 3-dimensions H-view is provided in Figure 4.

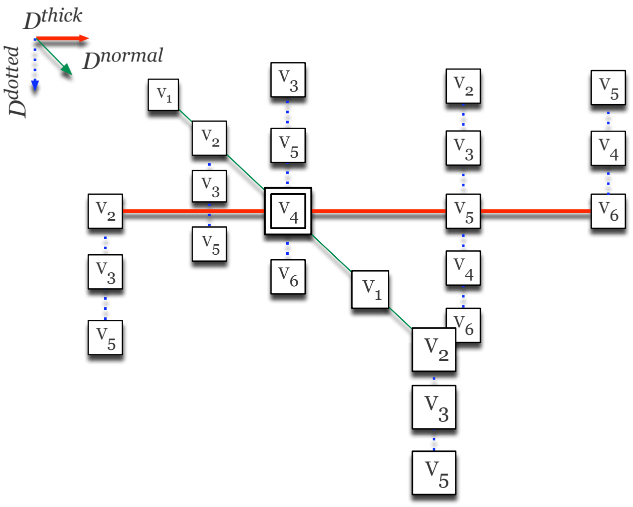


Figure 4: An example of a 3-dimensions H-view.

This view has focus on v_4, size $l = 5$, main dimension D_{dotted}, and secondary dimensions D_{thick} and D_{normal}.

A star view [12] visualizes information related to a focus vertex and a set of n chosen dimensions. There are two typologies of star views: the *star view* and the *m-extended star view.*

Definition 10. **Star view** - Given a zz-structure $S = \bigcup_{k=1}^{|C|} D^k$, where $D^k = \bigcup_{i=1}^{l_k} R_i^k \cup V_0^k$, and where $R_i^k = (V_i^k, E_i^k, f, \{c_k\}, c)$, the *star view* of *focus* $x \in V = \bigcup_{i=0}^{l_k} V_i^k$ and dimensions $D^1, D^2, \ldots, D^n$ is a star graph ${}_{n+1}$-*star* on central vertex x and neighborhood $N(x) = \{y \in V : y = x^{+1}, x^{+1} \in R^i_{(x)}, i \in \{1, \ldots, n\}\}$.

The *m-extended star view* extends the number of cells directly accessible from a view; it is based on a star view, but, for each vertex y in the neighborhood $N(x)$, adds the set of the p ($p \le m$) posward cells related to the given dimensions.

Definition 11. **m-extended star view** - Given a zz-structure $S = \bigcup_{k=1}^{|C|} D^k$, where $D^k = \bigcup_{i=1}^{l_k} R_i^k \cup V_0^k$, and where $R_i^k = (V_i^k, E_i^k, f, \{c_k\}, c)$, the *m-extended star view* is a *star view* of focus $x \in V = \bigcup_{i=0}^{l_k} V_i^k$, dimensions $D^1, D^2, \ldots, D^n$, and each extension constituted, $\forall y \in N(x)$ and $\forall i \in \{1, \ldots, n\}$, by the paths $(y^{+1}, \ldots, y^{+p}) \subseteq R^i_{(x)}$ $(p \le m)$.

Fig. 5 shows an schematic example of 5-extended star view.

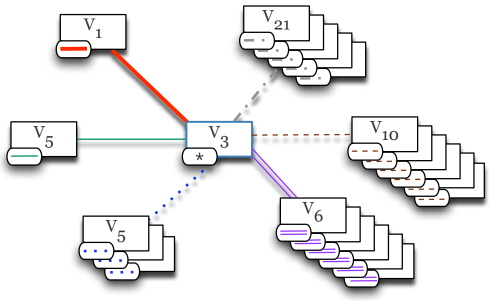


Figure 5: A 5-extended star view.

The view has focus v_3 and shows the connections along six dimensions. Note that cells v_1 and v_5 have, e.g., extension $p = 0$, while the maximum extension $(p = m = 5)$ is reached only by the connection along the dashed and double lines.

For lack of space, we cannot provide all the formal extensions to multidimensional views, the algorithms for the additions of new connections and the displaying of neighbouring views and we refer interested readers to [9, 10, 11, 12]. Just note that the above formal models have found interesting real world applications. Very briefly, in [9] the authors present an actor-based model, capable of representing both hypermedia distribution and collaborative schemes among different and heterogeneous entities which are part of a particular grid infrastructure and cooperate in order to achieve common goals and solve problems. In [10, 12], the authors propose the use zz-structures in order to help an author of an e-learning environment, to organize documents on a given topic in a concept space, and to create semantic interconnections and personalized maps. Finally, in [11] the authors propose a multi-agent adaptive system to support tours of virtual museums. In particular, the agents collaborate in order to help users visualizing their personalized views and choosing their navigational path inside the virtual museum.

4. DISCUSSION AND CONCLUSIONS

In this paper we have concentrated our attention on the formal models for representing zz-structures. Besides the motivations, above provided, in our opinion a formal model can help the navigation of a user by providing extra information, such as, e.g., the distances between the cell where (s)he is located and one where (s)he wants to move. Defining zz-structures as graphs allows, e.g., the application of known algorithms for the (dynamical) computation of shortest paths, or of paths with small stretch factor (i.e., ratio between the best path connecting two nodes and the shortest path). A user may thus compute a general shortest path, or, e.g., a shortest path (given that the two nodes are connected) in the subgraph induced by a particular color, meaning that (s)he wants to move following a unique dimension, i.e., concept.

Currently we are extending the formal model and preparing a survey of current literature on zz-structures, in order to analyze and synthesize it, and to stimulate new reflections and studies on this innovative way of conceiving the organization of information and knowledge.

5. REFERENCES

[1] I. Anderson. From ZigZag^{TM} to BigBag: Seeing the wood and the trees in online archive finding aids. In *Proceedings of the Workshop on New Forms of Xanalogical Storage and Function*. Turin, Italy, 29 June 2009.

[2] M. Andric, V. Devedzic, W. Hall, and L. Carr. Keywords linking method for selecting educational web resources à la ZigZag. *International Journal of Knowledge and Learning*, 3(1):30–45, 2007.

[3] S. Canazza and A. Dattolo. Open, dynamic electronic editions of multidimensional documents. In *IASTED Proceedings of European Conference on Internet and Multimedia Systems and Applications*, pages 230–235. Chamonix (France), 14-16 March 2007.

[4] L. Carr. ZigZag for web browsers, 2001. http://www.ecs.soton.ac.uk/∼lac/zigzag.

[5] P. Casoto, A. Dattolo, F. Ferrara, N. Pudota, P. Omero, and C. Tasso. Toward making agent uml practical: a textual notation and a tool. In *Proceedings of the Workshop on Adaptation for the Social Web, 5th ACM Int. Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 14–23. Germany, 29 July - 1 August 2008.

[6] P. Casoto, A. Dattolo, and C. Tasso. Sentiment classification for the italian language: A case study on movie reviews. *Journal of Internet Technology, Special issue on Intelligent Agent and Knowledge Mining*, 9(4):365–373, 2008.

[7] A. Dattolo. Authoring and navigating ethnic music audio archives. *Signal Processing*, 2009 (*to appear*).

[8] A. Dattolo, F. Ferrara, and C. Tasso. Supporting personalized user concept spaces and recommendations for a publication sharing system. In *UMAP '09: Proceedings of the 1th and 17th international conference on User Modeling, Adaptation, and Personalization*. Trento, Italy, 22-29 June 2009.

[9] A. Dattolo and F. L. Luccio. A new actor-based structure for distributed systems. In *International Conference on Hypermedia and Grid Systems (HGS07)*, pages 195–201. Opatija, Adriatic Coast (Croatia), 21-25 May 2007.

[10] A. Dattolo and F. L. Luccio. Formalizing a model to represent and visualize concept spaces in e-learning environments. In *Proceedings of the 4th Webist International Conference (WEBIST08)*, pages 339–346. Funchal, Madeira, Portugal, 4-7 May 2008.

[11] A. Dattolo and F. L. Luccio. Visualizing personalized views in virtual museum tours. In *International Conference on Human System Interaction (HSI08)*, pages 109–114. Krakow, Poland, 25-27 May 2008.

[12] A. Dattolo and F. L. Luccio. A new concept map model for e-learning environment. In *Lecture Notes in Business Information Processing 18*, pages 404–417, April 2009.

[13] P. Flocchini, B. Mans, and N. Santoro. Sense of direction: Definitions, properties and classes. *Networks*, 32(3):165–180, 1998.

[14] M. McGuffin. A graph-theoretic introduction to Ted Nelson's Zzstructures. January 2004. http://www.dgp.toronto.edu/∼mjmcguff/research/zigzag/.

[15] M. McGuffin and m. c. schraefel. A comparison of Hyperstructures: Zzstructures, mSpaces, and Polyarchies. In *Proceedings of the 15th ACM Conference on Hypertext and Hypermedia (HT'04)*, pages 153–162. Santa Cruz, California, USA, 9-13 August 2004.

[16] A. Moore and T. Brailsford. Unified hyperstructures for bioinformatics: escaping the application prison. *Journal of Digital Information*, 5(1):Article No.254, 2004.

[17] A. Moore, J. Goulding, T. Brailsford, and H. Ashman. Practical applitudes: Case studies of applications. In *Proceedings of the 15th ACM Conference on Hypertext and Hypermedia (HT'04)*, pages 143–152. Santa Cruz, California, USA, 9-13 August 2004.

[18] T. H. Nelson. What's on my mind. In *Invited talk at the first Wearable Computer Conference*. Fairfax VA, 12-13 May 1998. http://www.xanadu.com.au/ted/zigzag/xybrap.html.

[19] T. H. Nelson. Welcome to ZigZag. 1999. http://xanadu.com/zigzag/tutorial/ZZwelcome.html.

[20] T. H. Nelson. Xanalogical structure, needed now more than ever: parallel documents, deep links to content, deep versioning, and deep re-use. *ACM Computing Surveys*, 31(4:33), 1999.

[21] T. H. Nelson. ZigZag (tech briefing): Deeper cosmology, deeper documents. In *Proceedings of the 12-th ACM conference on Hypertext and Hypermedia (HT'01)*, pages 261–262. University of Aarhus, Aarhus, Denmark, 14-18 August 2001.

[22] T. H. Nelson. Structure, tradition and possibility. In *Proceedings of 14th ACM Conference on Hypertext and Hypermedia (HT'03)*. Nottingham, United Kingdom, 26-30 August 2003. http://www.ht03.org/keynote-ted.html.

[23] T. H. Nelson. A cosmology for a different computer universe: data model mechanism, virtual machine and visualization infrastructure. *Journal of Digital Information: Special Issue on Future Visions of Common-Use Hypertext*, 5(1):298, 2004.

[24] K. Wideroos. Awt (associative writing tool): supporting writing process with a ZigZag based writing tool - work in progress. In *Proceedings of the 12-th ACM conference on Hypertext and Hypermedia (HT'01)*, pages 35–36. University of Aarhus, Aarhus, Denmark, 14-18 August 2001.

From ZigZag™ to BigBag: Seeing the wood and the trees in online archive finding aids

Ian G. Anderson
University of Glasgow
11 University Gardens
Glasgow G12 8QH
+44 (0)141 330 3843

I.Anderson@hatii.arts.gla.ac.uk

ABSTRACT

This paper reports on a one year speculative research project that sought to test the technical feasibility, practical implications and usability of transforming an XML Encoded Archival Description (EAD) finding aid into an XML ZigZag™ structure and applying a relational browser interface.

Categories and Subject Descriptors

H.5.4 [**Hypetext/Hypermedia**]: Navigation; User Issues

General Terms

Design, Experimentation, Human Factors.

Keywords

EAD, ZigZag™, Ted Nelson, XML, Browsing, Visualisation.

1. INTRODUCTION

On the whole the archive profession is a conservative and traditional one. Since its inception the principles of provenance, or *Respect des Fond*, and adherence to original order have been dominant characteristics in most archive communities. As a result the practice of describing archive collections in hierarchical arrangements is firmly embedded. Compared to other information services, however, standardisation, both in terms of descriptive standards and arrangement have been relatively late developments, as has the provision of online finding aids.

However, as more archival finding aids, of increasing complexity, become available online the difficulty of seeing the 'wood from the trees' increases. This is particularly the case when these are implemented in Encoded Archival Description (EAD) [1]. EAD is an XML DTD for the creation of machine readable, cross searchable archival finding aids and its creators consciously based its structure on hierarchical analogue finding aids. Whilst this provided an important comfort zone for archivists to migrate to encoded finding aids, it is also meant EAD inherited the innate difficulty of navigating hierarchical structures.

Whilst an archive's physical space, catalogue arrangement and archivist's assistance all help to guide users' navigation in the analogue world, this paradigm does not easily translate to the electronic. Nor has there been a significant body of research established on archive user's information seeking behaviour. Indeed there is little evidence that traditional archival arrangement adequately served the needs of users in the analogue world. It is unlikely, therefore, that replicating such arrangements in the digital world would prove any more successful.

Where research on archive user needs has been undertaken a range of characteristics have been discovered that suggest a more flexible approach to archival access is required. The very earliest studies in the late 1990s indicated that time, training and access to information about information were crucial barriers to electronic access, even though this access had become a critical component of historians' research methods [2]. Later studies have revealed the plurality of historians' information seeking behavior but also the need for both research and archival context that was common amongst the most popular methods [3] and the importance of intermediaries in the use of online material [4]. Academic historians require multiple pathways to access primary research materials and the need for user education on electronic searches suggests that current provision hinders access [5]. Moreover, the need for orientation in even the most experienced user has been emphasized [6].

Archive portal sites such as the Archives Hub, A2A, AIM25, ANW and SCAN are evidence of the desire to search across collections and repositories but typical means of browsing or displaying search results, such as lists and directories, severely restrict users' ability to see where they are, how they got there and where they can go next [7, 8, 9, 10, 11]. Providing linked 'cross-walks' such as subject keywords, functional descriptions, person, place and corporate names can only go so far in addressing this problem. Points at which these cross-walks intersect can not easily be displayed and users wishing to move from one to another need to repeat searches or navigate up and down the hierarchy.

This problem increases exponentially where related material is held in different series, collections or repositories. In these circumstances trying to follow a particular person, function or responsibility is extremely difficult. In following one path, users lose sight of others, where they cross and what their relationships are. In essence the multidimensional relationships that exist within the finding aid are subordinate to its hierarchical structure.

The threads of this research all combine to suggest that a means by which archive users can quickly and intuitively orientate themselves within collections and identify the relationships and context of the resources they are viewing would be immensely beneficial. This project, funded by the UK's Arts and Humanities Research Council (AHRC) through a one year speculative research grant, sought to test a novel approach to structuring and visualising archival information by applying a relational browsing interface to EAD finding aids that have been transformed into a multidimensional structure.

2. A MULTIDIMENSIONAL SOLUTION

One potential solution to this problem is to structure and visualise this information multidimensionally. For example, repository, collection, date and function could each be a separate dimension, rather like lines on a London Underground map. Therefore, a user viewing the person name dimension (or line) would see each individual represented in a finding aid as a cell. This person may appear in different parts of a collection, separate collections at the same repository and at other repositories, quite possibility related to different organisations, functions or roles. Whilst well developed finding aids can make these links, it is very difficult for users to see and navigate them.

One such means of organising information multidimensionally is the ZigZag™ concept developed by Ted Nelson [12]. In other words, a piece of information can exist in different places at the same time and have many connections to other information that may also exist in more than one place. The beauty of the ZigZag™ system is that the user can bring multiple instances of the same information into one view and by changing the dimensions can instantaneously see how the related bits of information are connected. Thus the user is always presented with a locally relevant view of the information, irrespective of how complex the structure is, and without losing the ability to navigate and view all the interconnections. The possibility to represent archival information in this way may provide both functionality and usability that reflects the deep interlinked structures of today's online finding aids. These additional dimensions could be used to provide a whole range of context specific information, such as related bibliographies, digital surrogates, user comments and help files. This would allow online finding aids to move from an access tool to an expert system.

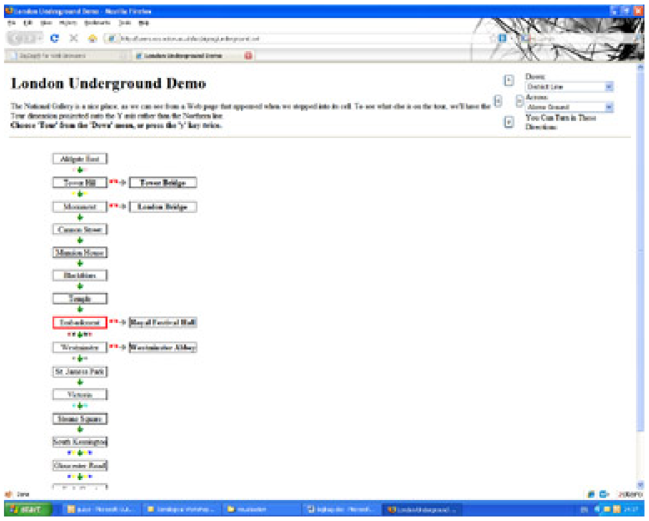
The advent of XML encoded finding aids, particularly EAD, and the wide scale implementation of descriptive standards made this an ideal time to test the viability of a ZigZag™ structure and visualisation.

The number and extent of dimensions it is possible to represent, does of course, depend upon the quality and extent of the underlying data. For this project two finding aids, Gateway to Archives of Scottish Higher Education (GASHE) and Navigational Aids for the History of Science and Technology (NAHSTE), provided by the University of Glasgow Archive Services were selected [13, 14]. These finding aids provided the project with the opportunity to test the concept against EAD, the descriptive standards General International Standard Archival Description (ISAD(G)2) and International Standard Archival Authority Record for Corporate Bodies, Persons, and Families (ISAAR(CPF)). The GASHE finding aid also including function and activity 'cross walks' within it. Both finding aids cross multiple collections and repositories.

Overall, the project aimed to achieve 'proof of concept' status - that it was technically feasible to map between EAD and a ZigZag™ structure; that the transformation between the two could be automated; that a web based interface could represent the multidimensions and; that it supported more intuitive browsing for users.

3. DEVELOPMENT

Several working examples of ZigZag™ structures have already been created in other projects using Perl, C, Python and Java to run on Windows, Linux and Mac. Initially the most promising of these for this project was the combination of XML, XSL and JavaScript, successfully demonstrated by Les Carr at the IAM Research Group, University of Southampton on a map of the London Underground, see Figure 1 below [15].



London Underground Demo

The National Gallery is a nice place, as we can see from a Web page that appeared when we stepped into its cell. To see what else is on the tour, we'll have the Tour dimension projected onto the Y axis rather than the Northern line.
Choose 'Tour' from the 'Down' menu, or press the 'y' key twice.

Figure 1. ZigZag™ for Web Browsers London Underground Demo

Taking this project as an inspiration and re-using the ZigZag™ for Web Browsers XML dialect kindly provided by Les Carr the project's functional and technical specification was defined using the Unified Modeling Language (UML), in particular use-case and activity diagrams. During this process it was decided that static, rather than 'on the fly', transformation of the finding aids were most appropriate in this context, given that the source data was static and the additional computational demands that online dynamic transformations would entail.

Mapping the EAD finding aids to the ZigZag™ structure through XSL was the first major project milestone. The finding aids actually comprise hundreds of files, in the case of GASHE, an XML EAD file using the ISAD(G)2 descriptive standard, XML ISAAR authority file, XML FANDA file (for functional and activity descriptions) for each individual collection, in each repository. The relationship between these files needed to be understood before the EAD file could be mapped to the ZigZag™ structure. At this point the application was named BigBag, partly as it was a nice alliteration of ZigZag™ but also because the dimension lines between cells in the mapping diagrams resembled the large string shopping bags used to carry groceries, see Figure 2 below.

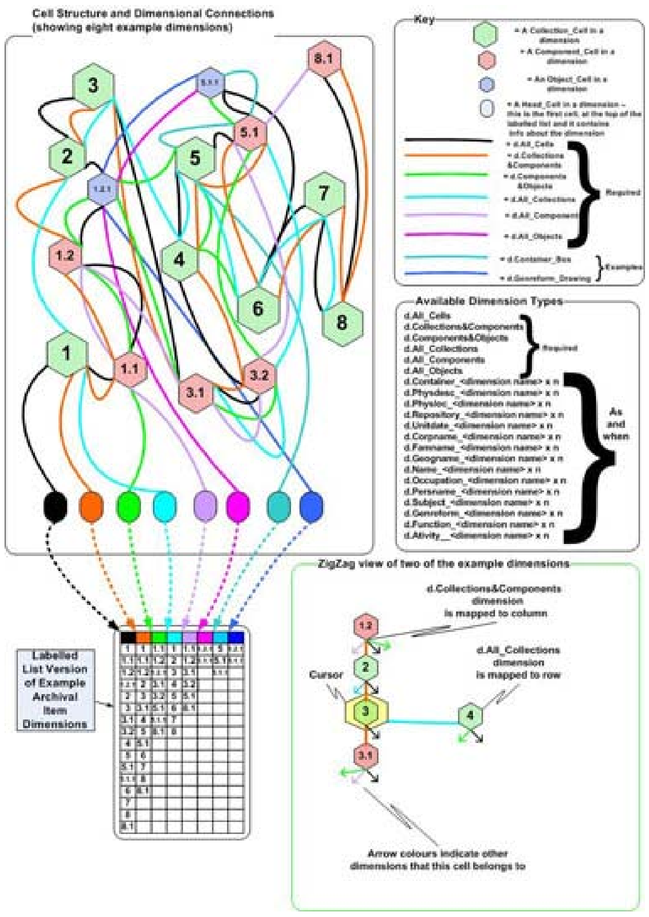

Cell Structure and Dimensional Connections
(showing eight example dimensions)
3
5.1.1
8.1
2
5
5.1
7
1.2.1
1.2
4
6
8
1
1.1
3.1
3.2
Key
= A Collection_Cell in a dimension
= A Component_Cell in a dimension
= An Object_Cell in a dimension
= A Head_Cell in a dimension – this is the first cell, at the top of the labelled list and it contains info about the dimension
= d.All_Cells
= d.Collections &Components
= d.Components &Objects
= d.All_Collections
= d.All_Components
= d.All_Objects
Required
= d.Container_Box
= d.Genreform_Drawing
Examples
Available Dimension Types
d.All_Cells
d.Collections&Components
d.Components&Objects
d.All_Collections
d.All_Components
d.All_Objects
Required
d.Container_<dimension name> x n
d.Physdesc_<dimension name> x n
d.Physloc_<dimension name> x n
d.Repository_<dimension name> x n
d.Unitdate_<dimension name> x n
d.Corpname_<dimension name> x n
d.Famname_<dimension name> x n
d.Geogname_<dimension name> x n
d.Name_<dimension name> x n
d.Occupation_<dimension name> x n
d.Persname_<dimension name> x n
d.Subject_<dimension name> x n
d.Genreform_<dimension name> x n
d.Function_<dimension name> x n
d.Ativity__<dimension name> x n
As and when
Labelled List Version of Example Archival Item Dimensions
ZigZag view of two of the example dimensions
d.Collections&Components dimension is mapped to column
d.All_Collections dimension is mapped to row
Cursor
Arrow colours indicate other dimensions that this cell belongs to

Figure 2. Archive Structure to ZigZag™ Structure Mapping

Thirteen EAD features were mapped to the ZigZag™ output tree by the projects 'Transform Finding Aid to ZigZag' stylesheet so each of the many ways to categorise an archival component became a dimension: subject, repository, personal name, location (shelf number etc.), container (box, folder, album etc.), format (book, film, letter etc.). function, date, century, business or corporate name, a daisy chain crossover linking all archive components, a crossover linking collection and series, and a crossover linking series and objects. As the underlying structure is one of linked circular lists the same cells may appear simultaneously in different orders in several linked lists.

The stylesheet had to manage some peculiarities of EAD and handle function, subject and place dimensions differently from other elements as multiple elements were possible, nested within a <p> tag. For example the stylesheet had to avoid adding the current place if it was the same as a place that had already been added, unless the current place was a sibling of the place that had already been added.

The stylesheet expected six required EAD elements, sixteen optional EAD elements and six optional multiple and recursive EAD elements. Seven escaped character codes were also stripped from the input tree as well as 13 characters that were illegal in JavaScript.

The cells of the output tree could be one of three types: collection (this included collection, fonds, class and record group descriptions), series (this included series, subfonds, subgroups or subseries descriptions) or object (this included item or file descriptions). Fragments from the XML ZigZag™ structure are provided in Figure 3 below.

```
<dimensions>
<dimension name="AllComponentsandObjects"
description="All Components and Objects" />
<dimension name="12thCentury"
description="12th Century" />
<dimension name="Subject:Accounting"
description="Subject: Accounting" />
<dimension name="Date:1971/1993"
description="Date: 1971/1993" />
<dimension name="Repository:Glasgow
Caledonian University Archives"
description="Repository: Glasgow Caledonian
University Archives" />
<dimension name="Format:file"
description="Format: file" />



<cells>
<cell n="1">
<url>http://www.gashe.ac.uk:443/cgi-
bin/view_isad.pl?id=GB-1847-
GP&amp;view=basic</url>
<title>Records of Glasgow Polytechnic
formerly Glasgow Collegeformerly Glasgow
College of Technology</title>
<content>fonds</content>
<link direction="AllArchiveComponents"
posward="2" />
<link
direction="AllCollectionsandComponents"
posward="2" />
<link direction="20thCentury"
posward="2"></link>
<link direction="Place:Cowcaddens Road
Glasgow" posward="2"></link>
<link direction="CorporateName:Glasgow
Polytechnic" posward="4"></link>
<link direction="Date:1971/1993"
posward="2"></link>
<link direction="Repository:Glasgow
Caledonian University Archives"
posward="2"></link>
</cell>
```

Figure 3. ZigZag XML Code

The transformation from EAD to ZigZag™ used Microsoft's Command Line Transformation Utility (MSXSL). This process was a two step transformation of archival finding aid data, from EAD XML into ZigZag™ XML and then into ZigZag™ HTML. Les Carr's ZigZag™ for Web Browsers is limited to 40 cells so a test file was selected that outputted 27 cells.

However, initial tests of a sample of data from the GASHE finding aid using Les Carr's XML dialect and JavaScript interface proved problematic. The transformation produced a functionally correct interface, but one that had limited usability, comprising hundreds of small black arrows dispersed across several screen widths, see Figure 4 below. Furthermore, even with the small sample data set, well specified PCs (dual core Pentium processors, 2GB RAM and 256MB dedicated graphics memory) were returning warnings that the JavaScript was causing the computer to run slowly. Although the number of cells was small, the number of dimensions associated with each cell in GASHE was far greater than in the original London Underground demo. These factors suggested that the JavaScript development path was unlikely to scale well enough for the amount of data and number of relationships required or provide sufficient complexity for the visualisation.

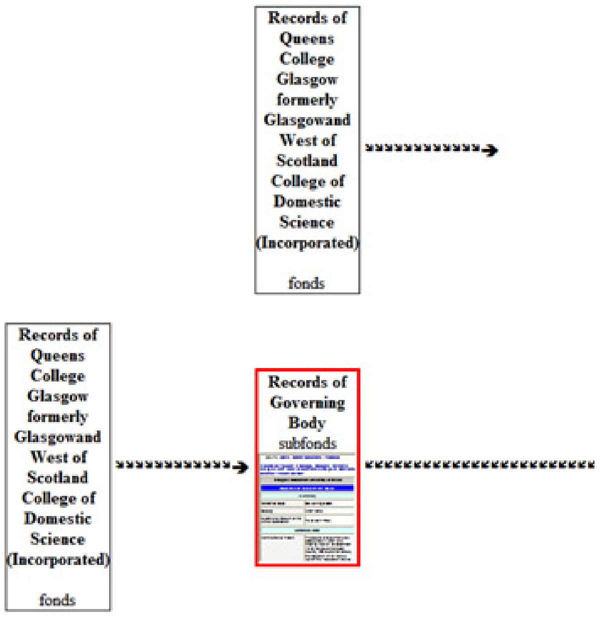

Records of Queens College Glasgow formerly Glasgowand West of Scotland College of Domestic Science (Incorporated)
fonds
Records of Queens College Glasgow formerly Glasgowand West of Scotland College of Domestic Science (Incorporated)
fonds
Records of Governing Body
subfonds

Figure 4. Section of BigBag JavaScript Demo

Whilst the appearance of the interface and the efficiency of the data handling could undoubtedly have been improved a decision was taken to seek an alternative means of visualisation. Initially an SVG interface was an attractive solution. It would keep the data within the XML family and the Parip Explorer project had successfully demonstrated a visualisation style that could suit the data [16]. However, the lack of project experience with SVG and the limitations of browser support led the project, after further research, to develop its interface using Macromedia/Adobe Flash based on an original idea by Moritz Stefaner [17]. Stefaner's relational browser for the CIA World Fact Book provided the underlying physics for an interface that positioned the selected 'cell' in the centre of the screen with lines spanning out to related cells of information. Selecting an outlying cell brought this to the centre of the screen and redrew the relationships. In other words it provided users with locally relevant view of their selected information without losing sight of the immediately bigger picture. An initial trial with a simple greyscale version of the relational browser interface demonstrated that it was capable of being modified to reflect, in part at least, the underlying ZigZag™ structure.

The second version of the interface, and the first to be tested with users, added a colour keyed sliding selector for the various dimensions as well as drop down menus for selecting instances of dimensions and archive components. A breakout box that linked to the original finding aid for each selected cell was also added as well as history and home buttons, see Figure 5 below [18].

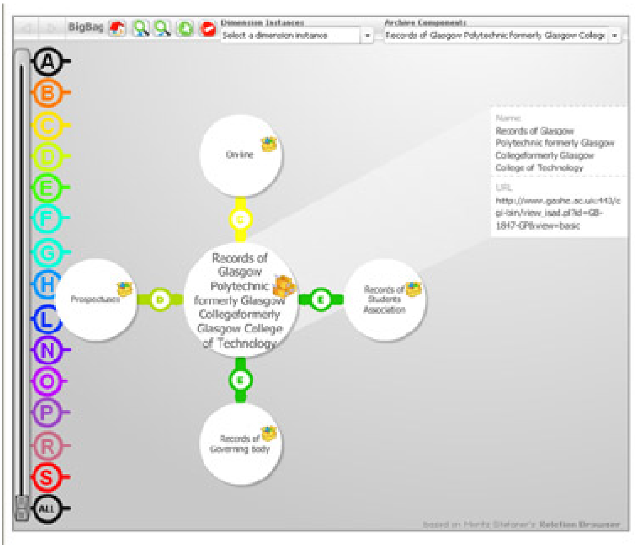

BigBag
Dimension Instances
Select a dimension instance
Archive Components
Records of Glasgow Polytechnic formerly Glasgow College
A
B
C
D
E
F
G
H
L
N
O
P
R
S
ALL
Online
Prospectuses
Records of Glasgow Polytechnic formerly Glasgow Collegeformerly Glasgow College of Technology
Records of Students Association
Records of Governing body
Name
Records of Glasgow Polytechnic formerly Glasgow Collegeformerly Glasgow College of Technology
URL
http://www.gashe.ac.uk/143/cgi-bin/view_isad.pl?id=GB-1847-GP&view=basic
based on Moritz Stefaner's Relation Browser

Figure 5. BigBag Flash Demo Version 2

A small, targeted sample of six people, two archivists, two historians and two students were selected to test this first version. Although the feedback was positive on the whole, with participants finding the interface clear, intuitive and supporting their browsing behaviour it was also evident that the multidimensionality of the underlying ZigZag™ structure was not being adequately expressed. Stefaner's relational browser only had to express one type of 'part of' relationship between two cells at a time and employed a single line to do so. However, with the finding aid ZigZag™ data there are potentially many different relationships between each cell that a single line cannot adequately convey. The sliding dimension selector was an attempt to overcome this problem but users did not like having to scroll through each dimension on the slider to see if it applied to their selected cells. It was evident that a means of immediately representing the number and type of relationships between cells was needed.

The next version of the interface, version three, tested the technical possibility of having multiple lines, each representing a different dimension, connect each cell and for the width of these lines to reflect the number of instances within that relationship. Once the project had established that this was technically and aesthetically possible version four of the interface was released. This removed the sliding dimension selector and replaced it with a simple key to the coloured lines. The format of the breakout box to link to the original finding aid was simplified and the screen split to show the original finding aid to the right [19]. In this version of the visualisation the colour of the line again indicates the dimension type with line widths indicating the number of instances for each type, the thicker the line, the greater the number of instances. Icons indicated whether the cell data existed at the collection, series or object level in the original finding aid, see Figure 6 below.

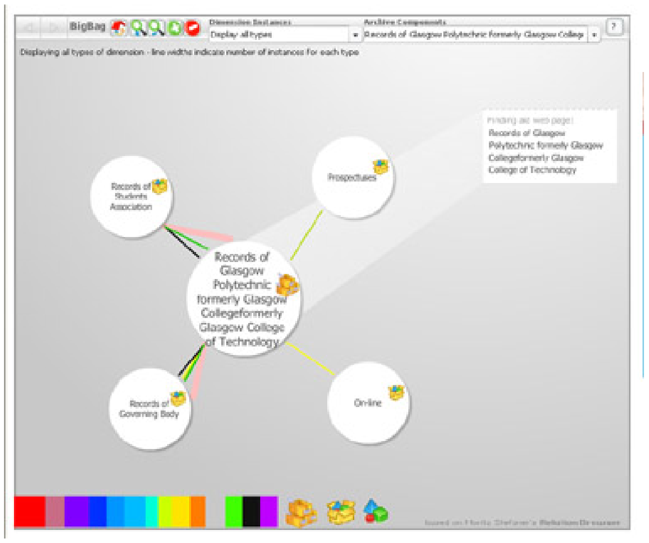

BigBag
Dimension Instances
Display all types
Archive Components
Records of Glasgow Polytechnic formerly Glasgow College
Displaying all types of dimension - line widths indicate number of instances for each type
Finding aid web page:
Records of Glasgow Polytechnic formerly Glasgow Collegeformerly Glasgow College of Technology
Records of Students Association
Prospectuses
Records of Glasgow Polytechnic formerly Glasgow Collegeformerly Glasgow College of Technology
Records of Governing Body
On-line

Figure 6. BigBag Flash Demo Version 4

The same six users who previously evaluated the project were shown the final version of the interface. In this case the underlying relationships within the data were agreed to be more explicit and enhanced the browsing of the finding aid data. However, users now instinctively wanted to click on the connecting lines to isolate a particular dimension, a functionality that was not possible, rather than use the dimension instance drop down menu. Furthermore, bugs and inconsistencies, particularly in the way dimension instances were selected significantly hampered users. Selecting a subject dimension form the drop down menu did not alter the cell display, an error that was not present in the previous version of the interface, and an additional erroneous subject instance also appeared on the menu.

By this late stage in the project time was a major constraint and it was not possible to either complete the scheduled interface testing or implement the zoom in and out function, or the add and subtract cells feature. Indeed it was a struggle to get the final version of the visualisation working in time at all.

4. STRENGTHS & WEAKNESSES

In part the project has fulfilled its main objectives. It established that it was conceptually possible to map from EAD to ZigZag™ and that a stylesheet could be developed that automated this process. However, it was not possible to establish that this transformation could be undertaken on all instances of EAD finding aids. Even working within the GASHE collection, variations in EAD encoding practices posed a challenge to efficient transformations. In part this is an inherent weakness of EAD in that its minimal compliance requirement amounts to little more than a collection description, akin to a minimally compliant TEI header. In the projects test data the lack of entity declarations for special characters also interfered with attempts to create suitable visualisations. In retrospect, editing the GASHE EAD prior to transformation would have created a far more efficient process. However, in trying to create a transformation that would be applicable to real life situations it would be unrealistic to expect archivists to amend their EAD files in order to accommodate our visualization. One useful spin-off from this, however, was the development of a set of EAD templates for the NoteTab text editor that placed greater constraints on coding choices. The objective of this exercise was that archivists might adopt them when creating new EAD finding aids and so avoid many of the common problems found in the EAD finding aids that hindered this project.

Although the project was able demonstrate the technical viability of an XML ZigZag™ for web browsers on larger and more complex data than the London Underground demo, this was not significantly so and time did not allow for the transformation and visualisation of the entire GASHE finding aid let alone test the stylesheet against NAHSTE.

Throughout the project a difficult balance had to be struck between refining and testing the stylesheet against larger and more varied sets of source data and developing a meaningful visualisation to test with users. In the end neither component was as fully developed as it could have been, but the project would have failed in an important respect if it had successfully transformed a large amount of data without any means of displaying the results. In retrospect the project may simply have been too ambitious in its scope.

After a few false starts the project did create a visualisation that reflected the underlying multidimensionality of the ZigZag™ data, albeit imperfectly. Although the fourth and final interface is the closest conceptually to the goals the project set itself its limited development time, even compared to the second version, proved a hindrance to establishing with certainty that this provided a significantly more beneficial interface to online archive users. It was never the projects intention to undertake extensive user evaluation or usability testing but within the constraints of what was possible the generally positive feedback is sufficient to suggest that the approach adopted does bring benefits for browsing archive finding aids online. How great those benefits are, for what type of information seeking behaviour and in what circumstances are questions that this project is unable to answer.

5. CONCLUSION

Perhaps inevitably for speculative research this project ultimately raises more questions than it answers, but has at least demonstrated sufficient merit to warrant those questions being investigated further. In particular the relative importance of the underlying EAD finding aid, ZigZag™ structure and visualisation on the end user's understanding of the data needs to be examined.

It is the intention to continue this research by creating a set of alternative structures and visualizations based on the same underlying archive data – a relational visualisation directly on an EAD fining aid; archive data that has been directly inputted to a ZigZag™ structure rather transformed; an EAD to ZigZag™ transformed visualisation (essentially an updated version of the current visualization) and; the archive data as displayed in its native state.

These alternative representations will provide a test bed through which end users understanding of the archive data will be examined using reception theory. Reception theory, sometimes called audience response theory, is a version of reader response theory that first developed in literary studies and was subsequently extended to include performance works. Reception theory proposes that a text does not have an inherent meaning, but meaning is created within the relationship between the text and the reader, shaped by the reader's background, influences and biases. By applying this theory to archival data it is hoped to explore the extent to which meaning is created by the user, is inherent to some extent in the data itself, and/or meaning is shaped by the way in which the data is structured or visualized.

There is also the potential for the approach tested here to be applied to information domains other than archives. Since this research was completed a brief market analysis was conducted to try and identify other areas that might benefit from this approach. Although this survey was by no means comprehensive, and there are a range of commercial data visualization products already available, the areas of social networks, personal or business contact lists, customer relationship management and enterprise relationship management are potentially new areas that future research could address.

6. ACKNOWLEDGMENTS

My thanks to the AHRC for funding, the Humanities Advanced Technology and Information Institute and the Faculty of Arts at the University of Glasgow for additional support, Lesley Richmond and Victoria Peters of University of Glasgow Archive Services for the supply of data and advice, Les Carr at the University of Southampton for permission to re-use his XML ZigZag™ for Web Browsers dialect, Moritz Stefaner for permission to re-use the underlying design of his relational browser and Steve North, Research Assistant, for his creativity, dedication and hard work.

7. REFERENCES

[1] EAD http://www.loc.gov/ead/

[2] Andersen, D.L. "Academic Historians, Electronic Information Access Technologies, and the World Wide Web: A Longitudinal Study of Factors Affecting Use and Barriers to that Use", The Journal of the American Association for History and Computing 1, June 1998. Available at: http://mcel.pacificu.edu/jahc/1998/issue1/articles/andersen/

[3] Anderson, I. "Are you being served? Historians and the Search for Primary Sources", Archivaria, 58, Fall 2004.

[4] Duff, W. Craig, B. Cherry, J. "Historians Use of Archival Sources: Promises and Pitfalls of the Digital Age," The Public Historian 26, no 2, Spring 2004.

[5] Tibbo, H. "Primarily History: How US Historians Search for Primary Sources at the Dawn of the Digital Age," American Archivist 66, no 1. Spring/Summer 2003.

[6] Duff, W. and Johnson, C. "Accidentally Found on Purpose: Information Seeking Behaviour of Historians in Archives," Library Quarterly 72, no. 4, 2002.

[7] Archives Hub http://www.archiveshub.ac.uk/

[8] Access to Archives (A2A) http://www.nationalarchives.gov.uk/a2a/

[9] AIM25 http://www.aim25.ac.uk/

[10] Archives Network Wales (ANW) http://www.archivesnetworkwales.info/

[11] Scottish archives Network http://www.scan.org.uk/

[12] ZigZag http://www.xanadu.com/zigzag/

[13] GASHE http://www.gashe.ac.uk/

[14] NAHSTE http://www.nahste.ac.uk/

[15] Les Carr's Zigzag for Web Browsers London Underground demo http://users.ecs.soton.ac.uk/lac/zigzag/

[16] Parip Explorer Project http://parip.ilrt.org/

[17] Moritz Stefaner's Web Site http://der-mo.net/

[18] BigBag Interfacev2 http://www.hatii.arts.gla.ac.uk/research/visual/demov2/index.html

[19] BigBag Interfacev4 http://www.hatii.arts.gla.ac.uk/research/visual/demov4/index.html

Creating Dynamic Wiki Pages with Section-Tagging

Denis Helic
IWM, TU-Graz
Inffeldgasse 16c
8010 Graz, Austria
dhelic@iicm.edu

Anwar Us Saeed
IWM, TU-Graz
Inffeldgasse 21A
8010 Graz, Austria
anwar.ussaeed@student.tugraz.at

Christoph Trattner
IICM, TU-Graz
Inffeldgasse 16c
8010 Graz, Austria
ctrattner@iicm.edu

ABSTRACT

Authoring and editing processes in wiki systems are often tedious. Sheer amount of information makes it difficult for authors to organize the related information in a way that is easily accessible and retrievable for future reference. Social bookmarking systems provide possibilities to tag and organize related resources that can be later retrieved by navigating in so-called tag clouds. Usually, tagging systems do not offer a possibility to tag sections of resources but only a resource as a whole. However, authors of new wiki pages are typically interested only in certain parts of other wiki pages that are related to their current editing process. This paper describes a new approach applied in a wiki-based online encyclopedia that allows authors to tag interesting wiki pages sections. The tags are then used to dynamically create new wiki pages out of tagged sections for further editing.

Categories and Subject Descriptors

I.7.1 [**Document and Text Processing**]: Document and Text Editing – *Document management, Version control.*

General Terms

Design, Experimentation

Keywords

Section Tagging, Wiki systems, Austria-Forum

1. INTRODUCTION

The popularity of social software has brought up new user generated content and metadata resources in the form of wikis, blogs, social tagging and bookmarking applications. These new systems have emerged as a major force driving the reshaping of information spaces on the World Wide Web to better serve both collaborative and personalized information needs of users. In social software applications Web has drifted towards users' content creation instead of the commercial content as a major contributing factor to Web resources.

For instance, wikis are used for sharing, management, and organization of knowledge. Wikipedia is a user-created encyclopedia and a well known example of a wiki system. Wiki systems are asynchronous, collaborative authoring and content versioning systems where any user can add and edit content. A new version of the page is stored in the system after each editing operation [2].

In wiki systems, user's content-creation/authoring processes involve laborious tasks like information selection from diverse resources, restructuring, modification, and adaptation of information object according to the perceived context [6]. The reuse of existing content in the form of copy-paste mechanisms in order to restructure and create new documents is applied by authors frequently. For example, a typical editing workflow in wiki systems involves investigating volumes of information where in fact only small part of that information is relevant to the current user need. Thus, the user has to browse all the resources again and again to review the related pieces of information from their relevant or selected resources. This typically requires a lot of effort and time.

On the other hand resource organization with tagging and bookmarking services like Delicious, Citeulike or bibsonomy have received community focus due to ease of use and information discovery mechanisms. In social tagging and bookmarking applications users assign free form keywords and annotations to the addresses (URLs) of an information resource (e.g., a web page) [8]. These keywords relate the current user context to the content of a tagged resource. The weighted set of keywords (tags) assigned to a resource by all users within a system is called the tag cloud. Tag cloud is a visual representation of tag terms in which their font is scaled according to their frequency weights.

As [3] suggests the user motivation to tag a resource might be organizational or communicational on one hand, and on the other hand the users tag resources for their personal use and/or to share them with others. For example, users who tag resources for their personal use in an organizational sense use social tagging applications to organize interesting, important, and related resources according to their current needs. The tags are applied as a support for later search and retrieval of tagged resources via search or navigating the tag cloud. Typically, the tag cloud provides an overview of defined tags showing only the tags themselves but not the actual content of the tagged resources. The resources are represented via navigable links. Another motivation of using tags is to share them with other users and in such a scenario tags are typically used in a communicational sense to send signals to other users about resources that might be of interest in a more general case.

Regardless of users tagging scope- personal resource organization or sharing it with others- they have to tag the whole resource. This, however, does not always fulfill the users need. For example, users are often viewing content and are interested only in one part of the whole content. For future use users tag and bookmark it with a keyword that would be helpful later to retrieve the content. In this case users tag the whole content with a navigational keyword useless to represent the context of resource but a useful one for them to reach the content section of their interest. This unrelated navigational tag in tag cloud will create noise. But users have no option to tag a particular interesting section within the whole resource. Such an option of tagging a part of resource may increase the user efficiency for later content retrieving, as well as help reducing noise from document tag cloud and providing a separate content-focused section tag cloud.

To overcome above mentioned problems we present a novel modified social tagging approach. The benefit of such an approach has been illustrated in a wiki system on the example of simplifying the editing process. We call this new approach section-tagging as it supports users to assign keywords and annotate sections of a wiki page.

To practically implement and test the idea we extended the functionality of an online encyclopedia called Austria-Forum with section tagging along with the conventional social tagging. The rest of the paper is organized as follows. The next section describes in more details the Austria-Forum system. The third section discusses the idea of section tagging in Austria-Forum and how it may be used to support content retrieval and to simplify a typical editing workflow. The fourth section describes the implementation of section-tagging idea within Austria-Forum. The last section provides conclusions and an overview of the future work.

2. AUSTRIA-FORUM

Austria-Forum [4] is a networked information system that manages a very large repository of information items, where new information items are easily published, edited, checked, assessed, and certified, and where the correctness and a high quality of each of these items is backed by a person that is accepted as an expert in a particular field. Consequently, each of the information items is citable as any other editorially checked content and might be used in education, scientific research, or journalism. The content of Austria-Forum is always related to Austria – as such Austria-Forum might be seen as an Austrian online encyclopedia.

In the first experimental phase of Austria-Forum the system had an editorial board of more than 20 editors and a growing community of users. The number of users who contributed with the content was more than 100. The number of unique users who have visited the site is around 4000 each month.

The current number of contributions is around 80000 (including pictures and videos as well as the content converted from the well-known Austrian cultural information system AEIOU [1]), out of which around 6000 are user-generated contributions – approximately 8% of all contributions. Most of these user contributions are pictures and photos, with a small number of blogs, discussion forum posts, and comments. Although these numbers are quite substantial for a site that has been online experimentally a more active community involvement is desired. Community tools and facilities are already present in the system. However, as a number of users suggested, usability and a better integration of different community tools with the main system needs to be improved.

Therefore, the original system that was technically based on an in-house developed content-management system has been replaced by an open-source wiki software called JSPWiki [5]. The idea here is that more users will be attracted to a well-known collaborative authoring tool such as wiki. Moreover, the intention is to offer a number of community tools that will support users in retrieving information quickly and reduce the complexity of editing workflow. Among such tools is also the above presented section-tagging tool.

Even if the Austria-Forum wiki is still under development, it nearly offers ideal environment to test the concept because a huge amount of test data is available.

3. SECTION TAGGING AND PERSONALIZATION

Section tagging is a novel social tagging approach which allows users to annotate the content of interest within a resource using free form keywords.

The implemented approach differs from existing tagging and bookmarking services in the following way. First, it allows the tagging of subdocument level content. Second, tag retrieves not merely the set of links annotated by tag keyword but also the actual content of the tagged sections. Thus, when the user clicks on a tag all sections from wiki pages that have been tagged with the particular term by the specific user are dynamically loaded and presented to the user in the form of a standard wiki page.

The section of a wiki page is a self explaining piece of information about some topic of interest. Tagged content snippets in the case of section tagging have conceptual relationship to perceived structure of an information object that the user relates to the tag terms. Hence, the context of information snippet of user's interest is more relevant to the user perception of an information object in relation to the tag terms. The underlying idea of such an approach is based on personalized content aggregation from different wiki pages because the wiki system may not hold the required information in one page but typically in various pages.

Personalization in Austria-Forum refers to the content annotation and aggregation from different wiki pages according to users' intent. A typical personalization scenario involves users collecting, customizing, and modifying diverse text snippets from different wiki pages within an informational focus being described by the given tag keyword.

System offers two levels of personalization:

- Users can tag and annotate sections of wiki pages as well as full pages and hence personalize the content of interest.
- A dynamic personalized wiki page content view is created for a user by aggregating all sections tagged by him with a particular keyword. The aggregated sections are retrieved from the same versions of wiki pages which were used while tagging. The rank of a particular section within this aggregated set is determined by the frequency of same tag assigned by other users to this section.

The resulting dynamic personalized wiki page can further be collaboratively edited to create a logically complete information object reflecting the particular user context. The system facilitates further the personal/collaborative knowledge creation and management. Dynamic wiki pages created by collecting snippets of information from diverse wiki pages allow users to restructure and organize information on multiple axes of personalization.

Currently, the section tagging is primarily used for supporting editing workflow in the system. For example, suppose that an author is writing a new contribution on the Mozart's birth house. Before writing about the birth house the author wants to have an introductory section about Mozart that includes the basic biographical information, the list of Mozart symphonies and a picture of the Mozart monument in Vienna. The basic biographical information is included in the first section of the page on Mozart biography, the list of symphonies is described in the page on Mozart's work and the Mozart monument is depicted in the page that talks about monuments in Vienna. Thus, the author tags all the appropriate section in pages in question with a tag "Mozart". In the personal section-tag cloud the tag "Mozart" is now visible. When the author clicks on that tag a new dynamic wiki page including three tagged sections from three different wiki pages is created on the fly. The author chooses to save the dynamically created page in the system. Now, the author can access the new page as any other wiki page and edit it by restructuring sections and adding new sections about Mozart's birth house.

4. IMPLEMENTATION ASPECTS

As described before, the core of the section-tagging mechanism is to allow users to tag not only a whole wiki page, but also to tag a particular section (identified with a heading). In this way users add semantic information to arbitrary sections of different wiki pages. In the next step, it is possible to extract sections referred by a particular tag and to create a new personalized wiki page out of tagged content snippets.

The implementation of the section concept is comprised of two functional modules, called Section-Tagging (ST) and Personalized-Content-Creation (PCC) module.

The JSPWiki system is based on a clean and extensible plug-in and filter architecture that allows easy addition and configuration of new modules.

The filter mechanism allows on the fly parsing and modifying of wiki pages before they are rendered.

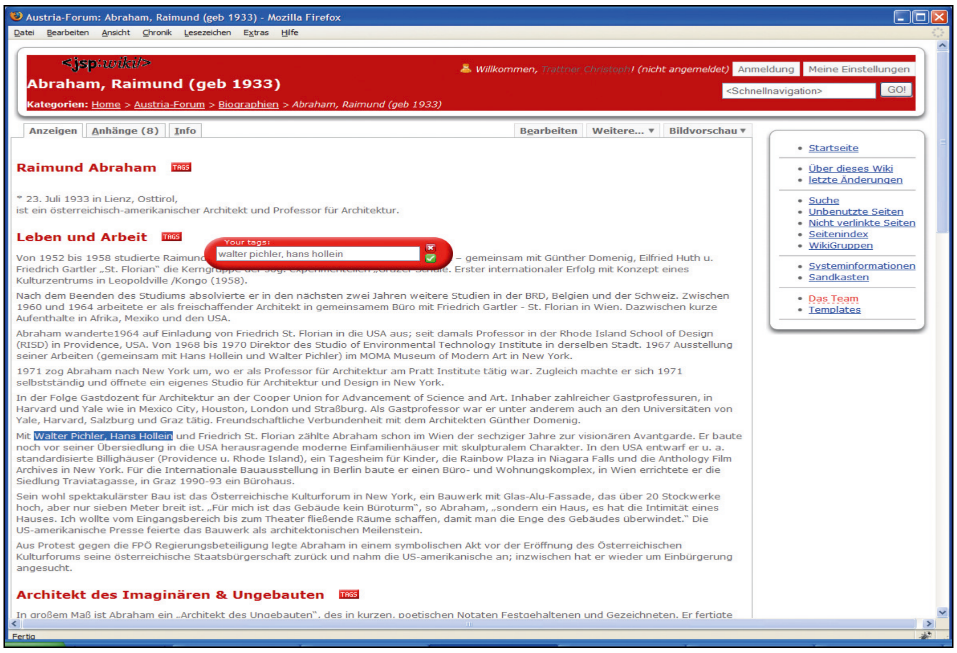
On the other hand, the plug-in mechanism allows server-side code to be referenced from within a wiki page. This code dynamically produces wiki content that can be included in the wiki page that refers to the plug-in.

Thus, technically the ST module is a filter module as it inserts section-tagging functionality into already existing wiki pages by pre-processing them; the PCC module is a plug-in module that dynamically creates a new wiki page according to the selected tag and the tagged sections from various wiki pages.

4.1 Section-Tagging Module

ST module is a filter for pre-processing of rendered wiki pages. This unit is responsible for extending document object model (DOM) of a rendered wiki page via a JavaScript module called ST form module. As shown in Figure 1, this module supplies a simple to use pop-up form (red colored box in front of section) that visualizes particular semantic section information by an onmouseover effect and letting the user tag a section using the onclick event. Moreover the ST form module also supplies the database connector module with information about the currently tagged section number and page version.

The actual centerpiece of the ST module is a unit called ST plug-in. It loads and manipulates the data from the ST data storage backend module, extracts user data from the ST security module and handles data sent by the ST form module via XMLHTTPRequest (see Figure 2).



Abraham, Raimund (geb 1933)

Kategorien: Home > Austria-Forum > Biographien > Abraham, Raimund (geb 1933)

Raimund Abraham

* 23. Juli 1933 in Lienz, Osttirol,
ist ein österreichisch-amerikanischer Architekt und Professor für Architektur.

Leben und Arbeit

Von 1952 bis 1958 studierte Raimund [illegible] – gemeinsam mit Günther Domenig, Eilfried Huth u. Friedrich Gartler „St. Florian" die Kerngruppe der sog. experimentellen „Grazer Schule. Erster internationaler Erfolg mit Konzept eines Kulturzentrums in Leopoldville /Kongo (1958).

Nach dem Beenden des Studiums absolvierte er in den nächsten zwei Jahren weitere Studien in der BRD, Belgien und der Schweiz. Zwischen 1960 und 1964 arbeitete er als freischaffender Architekt in gemeinsamem Büro mit Friedrich Gartler - St. Florian in Wien. Dazwischen kurze Aufenthalte in Afrika, Mexiko und den USA.

Abraham wanderte1964 auf Einladung von Friedrich St. Florian in die USA aus; seit damals Professor in der Rhode Island School of Design (RISD) in Providence, USA. Von 1968 bis 1970 Direktor des Studio of Environmental Technology Institute in derselben Stadt. 1967 Ausstellung seiner Arbeiten (gemeinsam mit Hans Hollein und Walter Pichler) im MOMA Museum of Modern Art in New York.

1971 zog Abraham nach New York um, wo er als Professor für Architektur am Pratt Institute tätig war. Zugleich machte er sich 1971 selbstständig und öffnete ein eigenes Studio für Architektur und Design in New York.

In der Folge Gastdozent für Architektur an der Cooper Union for Advancement of Science and Art. Inhaber zahlreicher Gastprofessuren, in Harvard und Yale wie in Mexico City, Houston, London und Straßburg. Als Gastprofessor war er unter anderem auch an den Universitäten von Yale, Harvard, Salzburg und Graz tätig. Freundschaftliche Verbundenheit mit dem Architekten Günther Domenig.

Mit Walter Pichler, Hans Hollein und Friedrich St. Florian zählte Abraham schon im Wien der sechziger Jahre zur visionären Avantgarde. Er baute noch vor seiner Übersiedlung in die USA herausragende moderne Einfamilienhäuser mit skulpturalem Charakter. In den USA entwarf er u. a. standardisierte Billighäuser (Providence u. Rhode Island), ein Tagesheim für Kinder, die Rainbow Plaza in Niagara Falls und die Anthology Film Archives in New York. Für die Internationale Bauausstellung in Berlin baute er einen Büro- und Wohnungskomplex, in Wien errichtete er die Siedlung Traviatagasse, in Graz 1990-93 ein Bürohaus.

Sein wohl spektakulärster Bau ist das Österreichische Kulturforum in New York, ein Bauwerk mit Glas-Alu-Fassade, das über 20 Stockwerke hoch, aber nur sieben Meter breit ist. „Für mich ist das Gebäude kein Büroturm", so Abraham, „sondern ein Haus, es hat die Intimität eines Hauses. Ich wollte vom Eingangsbereich bis zum Theater fließende Räume schaffen, damit man die Enge des Gebäudes überwindet." Die US-amerikanische Presse feierte das Bauwerk als architektonischen Meilenstein.

Aus Protest gegen die FPÖ Regierungsbeteiligung legte Abraham in einem symbolischen Akt vor der Eröffnung des Österreichischen Kulturforums seine österreichische Staatsbürgerschaft zurück und nahm die US-amerikanische an; inzwischen hat er wieder um Einbürgerung angesucht.

Architekt des Imaginären & Ungebauten

In großem Maß ist Abraham ein „Architekt des Ungebauten", des in kurzen, poetischen Notaten Festgehaltenen und Gezeichneten. Er fertigte

Figure 1: ST form module

As a data storage module the open-source content-management system Scuttle [7] is deployed. The database itself is not accessed by the API which the systems offers but by the database connector module which extracts user data such as username and IP address directly from the JSPWiki user session module. This user data record is stored together with a special section URI to the Scuttle database by the plug-in module every time a section is tagged by the user, in order to guarantee an unambiguous relationship between user and tagged sections.

In order to have a clear relationship between page sections, page versions and corresponding tags and still offer a readable URI without changing the database structure itself, the well known (X)HTML method of creating links within a hypertext document was adopted in the following form:

http://<URI>#<section ID>_<version>

Thus a section of a wiki page can be easily addressed to a tag and vice versa by adding a fraction identifier holding information about the section ID (<section ID>) and page version (<version>).

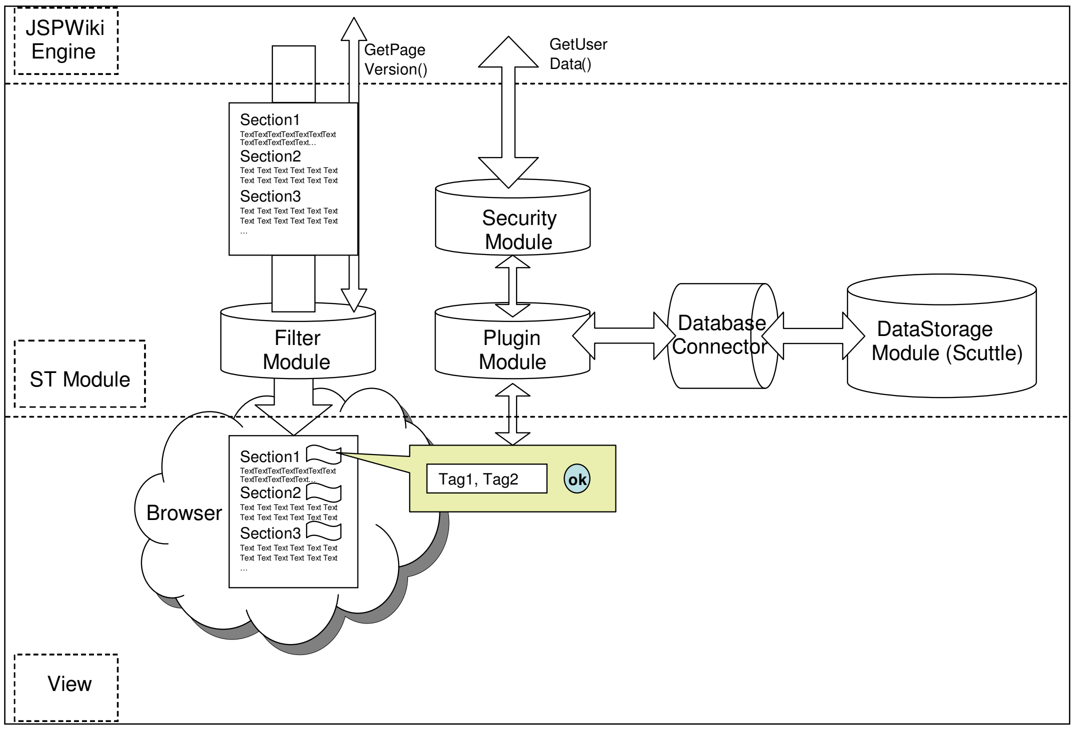


Figure 2: Architectural diagram of the ST module

4.2 Personalized-Content-Creation Module

The PCC module is implemented as a plug-in that can be included in any wiki page. Currently, this module is included in a personalized wiki page that is shown on the right-side of the user screen. It shows a standard tag cloud with tags assigned by a particular user to wiki page sections of interest. When a user clicks on a tag the PCC module retrieves all tagged sections using the appropriate wiki page versions. The sections are then dynamically combined into a wiki page that is shown to the user. The user has then the possibility to edit and modify this new wiki page using the standard wiki editor and to save the editing operations in a completely new wiki page for later retrieval. Moreover, the dynamic page can be still retrieved at all times by simply clicking on the appropriate tag. Note that the dynamic page is always created on the fly, thus whenever the user adds tags to sections of some other wiki pages this will be reflected in the dynamic page as the page will include the new sections.

5. CONCLUSIONS AND FUTURE WORK

A novel approach for tagging sections of wiki pages has been presented. This approach is able to personalize the users' content in an efficient way. This approach has reduced the manual effort required to author a wiki-page about a topic. Often, the wiki system may not have the required info in one page but typically in various pages. Therefore, a combination of the social tagging approach with the wiki concept in an innovative manner facilitates an easy retrieval of the relevant content in the form of a new dynamically created wiki page. Such dynamic wiki pages created by collecting snippets of information from diverse wiki pages allow users to restructure and organize information on multiple axes that best fit their current needs.

The future work includes:

- Testing and evaluating the section-tagging approach with a number of users during the experimental phase of Austria-Forum.

- Sharing of section tags between users, i.e. not only a personalized section-tag cloud should be generated but also a global one with tags from all users.
- Interesting aspects of global section-tag clouds will be the tag and section selection strategy in the case that there are numerous sections tagged by a particular tag. A collaborative filtering approach taking into the account the user profiles might be needed to limit the sections only to those that are most relevant.
- Extending the section-tagging approach to arbitrary Web resources. This can be implemented as browser plug-in in future which will gather the tagged content in a dynamic wiki system as a Web based service.

6. ACKNOWLEDGMENTS

We would like to thank Prof. Hermann Maurer and Mr. Muhammad Tanvir Afzal for providing supporting discussions, valued inputs, and comments. The research of this contribution is funded by Higher Education Commission of Pakistan and TUGraz.

7. REFERENCES

[1] AEIOU - Annotierbare Elektronische Interaktive Oesterreichische Universal-Informationssystem, http://aeiou.iicm.tugraz.at. (Last visited: March 30, 2009).

[2] Alain Désilets , Sébastien Paquet , Norman G. Vinson, Are wikis usable?, Proceedings of the 2005 international symposium on wikis, p.3-15, October 16-18, 2005, San Diego, California.
DOI= http://doi.acm.org/10.1145/1104973.1104974.

[3] Ames, M. and Naaman, M. 2007. Why we tag: motivations for annotation in mobile and online media. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (San Jose, California, USA, April 28 - May 03, 2007). CHI '07. ACM, New York, NY, 971-980. DOI= http://doi.acm.org/10.1145/1240624.1240772.

[4] Austria-Forum, http://www.austria-forum.org. (Last visited: March 30, 2009).

[5] JSPWiki, http://www.jspwiki.org. (Last visited: March 30, 2009).

[6] Nelson, L., Smetters, D., and Churchill, E. F. 2008. Keyholes: selective sharing in close collaboration. In CHI '08 Extended Abstracts on Human Factors in Computing Systems (Florence, Italy, April 05 - 10, 2008). CHI '08. ACM, New York, NY, 2443-2452. DOI= http://doi.acm.org/10.1145/1358628.1358701.

[7] Scuttle, http://sourceforge.net/projects/scuttle. (Last visited: March 30, 2009).

[8] Tony Hammond, Timo Hannay, Ben Lund, and Joanna Scott. Social Bookmarking Tools (I): A General Review. D-Lib Magazine, 11(4), April 2005.

Towards XML Transclusions

Angelo Di Iorio, Silvio Peroni, Fabio Vitali
Dept. of Computer Science
University of Bologna
Mura Anteo Zamboni, 7
40127, Bologna, Italy
{diiorio | speroni | fabio}@cs.unibo.it

John Lumley, Tony Wiley
HPLabs Bristol
Filton Road, Stoke Gifford
Bristol BS34 8QZ
United Kingdom
{john.lumley | anthony.wiley}@hp.com

ABSTRACT

The idea of *transclusion* has been at the same time the strength and weakness of Xanadu: some people considered it as an extremely powerful mechanism to get any version of any fragment of any document in a global shared document space, others as a very complex solution too difficult to be actually implemented and delivered. We believe transclusions are still worth implementing and would allow designers to build very sophisticated hypermedia applications. On the other hand, we are aware that the original design of Xanadu cannot be implemented without uprooting current systems, protocols and technologies - *in primis* the World Wide Web and XML. In fact, there is a great distance between the original data model of transclusions - strongly based on external referencing mechanisms - and the XML data model - strongly based on hierarchical structures and embedded markup.
This paper investigates to what extent the concept of transclusion can be shaped for the world of XML, and studies simplified models for building functionalities inspired by Xanadu. Particular attention is given to the support for tracing fragments provenance in multi-source documents and for synchronizing distributed content through transclusions. The paper also traces a roadmap to actually implement transclusions for XML - identifying three incremental steps - and briefly describes some experimental prototypes.

Categories and Subject Descriptors

H.5.4 [**INFORMATION INTERFACES AND PRESENTATION (I.7)**]: Hypertext/Hypermedia

1. INTRODUCTION

The concept of *transclusion* is rooted in the early days of hypertext[13]. A transclusion is a very advanced inclusion, whose content is not actually copied but stored as a virtual reference to the original source. There is only one copy of each fragment in the whole document space and transcluded data is permanently connected to the original.

Transclusions were the core idea of Xanadu. Xanadu documents were built on-the-fly from fine-grained references so that users could access, modify and reuse any fragment from any document in a safe and controlled way. The implementation of transclusions relies on external referencing to locations in a text data stream, through complex addressing mechanisms. Mark-up information, links and metadata are expressly distinct from the flow of text in order to guarantee flexibility and expressiveness.

The world of XML (and SGML) relies on a completely different strategy: mark-up is embedded, documents are strictly hierarchical and assertions about text fragments are made by wrapping them with elements, enriched by attributes. Ted Nelson himself pointed out disadvantages of such an approach in relation with transclusions[12]. The three objections he raised can be summarized as: (1) SGML approach interposes a 'forced' structure between users and actual content while editing, (2) SGML approach only supports well-formed inclusions and does not allow users to change included content, (3) SGML approach does not support overlap and non-hierarchical relationships.

Thus, these two positions seem to be irreconcilable. The goal of this paper is to investigate whether and how transclusions can be implemented for XML documents and trace a possible course towards that goal.

The preliminary step is to understand what we mean by 'XML transclusions'. Our goal is not to re-implement a revised version of Xanadu based on XML technologies, rather to support users in creating and editing composite XML documents that make some Xanalogical functionalities possible. In particular, we are interested in: *tracing fragment provenance* and *remote manipulation*. First of all, composite XML documents would benefit from rich information about the origin of each fragment. That makes it possible to identify single contributions in collaboratively edited documents, to display multiple changes in a single document, to go back to original resources and navigate documents in a free and powerful way. The permanent connection between transcluded content and original sources would also make possible sophisticated forms of editing. Changes to remote documents could be propagated through transclusions or - the other way round - local modifications could actually update remote resources. Yet, such scenarios also require other tricky issues to be addressed such as content merging, synchronization, access permissions, reliability and so on.

The core of this paper discusses three approaches for implementing XML transclusions, in section 4. These solutions use different syntaxes and are progressively expressive and powerful. Before that, we briefly review the recent literature about transclusions and XML. The paper also mentions some prototypes we are developing.

2. TRANSCLUSIONS AND XML

The two most recent research efforts for implementing transclusions are outside the XML universe. Nelson and his team proposed Transliterature[11] a revision of the original Xanadu project built on newer technologies. Transliteral documents are dynamically built on top of transcluded fragments, so that rich and fine-grained connections between documents are permanently available. Two prototypes are worth mentioning: *Transquoter*, that allows users to hide, highlight and surf multi-source quotations and *XanaduSpace*, that provides users a 3D view of the overall document space and shows some of the advanced surfing and displaying functionalities envisioned for such inter-connected documents.

Kolbitsch at al.[8] investigated transclusions for HTML, presenting a prototype that allows users to select content from web pages and transclude them into new documents. Transclusions directives are stored as in-line elements and very complex URLs. Interesting issues are still open about content addressing and merging, especially considering that web pages may change often and may have complex (and badly-formed) internal structures.

The closest solution to transclusions in the XML universe is XInclude[10]. XInclude is a W3C standard for merging XML documents, by writing inclusion directives and retrieving other (parts of) documents. The focus of XInclude is on well-formed XML fragments. Although even text fragments can be included (`parse="text"`), XInclude does not allow users to include bad-formed fragments or ranges. That makes it impossible to implement fully-fledged transclusions. Simplified forms of transclusions are possible through XInclude anyway. XIPr[17] is an implementation of XInclude in XSLT 2.0, very efficient and simple to be integrated in other XML applications. XSLT technologies could be also used straightforwardly for simplified transclusions: a general approach is discussed in [5].

XLink[3] could also be cited as a way to transclude pieces of content. The `@show` and `@actuate` attributes allow users to define *hot-links*, that are similar to transclusions apart from implementation details. On the other hand, such a solution is only partial and XLink does not seem to succeed as expected.

XLink is a relative recent effort, rooted in a very long experience in the hypermedia research. Since the early 90s researchers of the Open Hypermedia community have been designing systems that let users to add sophisticated links to a wide range of documents, merging original content and external interventions. Chimera[1], Microcosm[6] and DLS[2] are only few representatives of these systems. More recently Tzagarakis et al.[16] presented CB-OHS (Component-Based Open Hypermedia Systems), an extensible infrastructure for managing complex and distributed hypermedia elements.

The design and implementation of CB-OHS follows the ‘philosophy of the primacy of structure over data’[16]. This is a key feature of the Open Hypermedia approach, strongly related with the idea of transclusions: the separation between the relationships and the information they relate to. It makes possible to handle data separately, to create links pointing to read-only reasorces, to create multiple and overlapped links on the same content and so on.

3. A CASE-STUDY: COLLABORATIVE REVIEWS

The idea of transcluding fragments from and to XML documents is still under-developed. Hereinafter we discuss a road-map to make that development possible, by using a case study throughout the paper.

Let us suppose we are building a system for supporting multiple users to write collaborative reviews about movies. It would be useful to let reviewers quote fragments from other reviews, keep trace of their source and maintain a ‘live’ channel between connected fragments. In fact, a review process can be improved by quoting opposite criticism, by letting reviewers access related reviews, by automatically updating distributed reviews with new material and by fostering discussion among reviewers. It is worth noting that such an idea is both rooted in the early days of hypermedia (the essence itself of hypermedia is the inter-connection between documents) and central in the recent trend of the World Wide Web (one of the milestones of the so-called Web 2.0 is just collaboration).

Consider now that reviewers are writing their comments about ‘Australia’, starring Nicole Kidman and Hugh Jackman[1]. The film is one of the most controversial of the early 2009. David wrote a quite positive review: “Baz Luhrmann’s Australia is good, but not a masterpiece”.[2] Brad has an opposite opinion: “A major miscalculation if there ever was one”[3]. Assume that Brad replied to David quoting his original note. In fact, a third reviewer - say Mike - might be interested in quoting both these opinions, even by citing a different comment by a fourth reviewer (for instance, Rebecca saying “I actually regret having seen the film through to the end.”[4]). Figure 1 shows a possible view of such a composite review, highlighting multiple contributions[5].

The multi-contribution view is only one of the applications of such an advanced quoting. Users might also be given the possibility to surf to the original review, in order to collect more information about the movie. Moreover, permanent connections between fragments could be enriched with metadata (stating, for instance, that the remote review is ‘positive’ or ’negative’), so that advanced search could be performed over the network of documents. Note also that

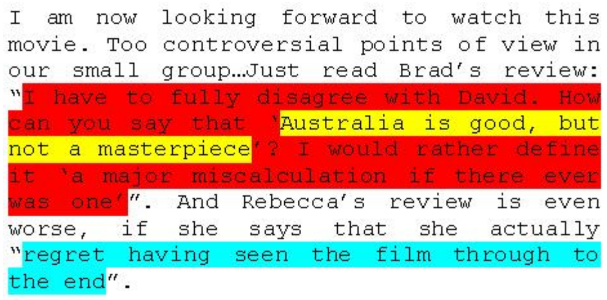
[1]http://www.australiamovie.com/

[2]http://www.theaustralian.news.com.au/story/0,25197,24670334-601,00.html

[3]http://www.ropeofsilicon.com/article/movie-review-australia-2008

[4]http://movies.about.com/od/australia/fr/austral-review.htm

[5]The picture, shown in the following page, contains the final rendering of the document rather than its internal (XML) markup



I am now looking forward to watch this movie. Too controversial points of view in our small group…Just read Brad's review: "I have to fully disagree with David. How can you say that 'Australia is good, but not a masterpiece'? I would rather define it 'a major miscalculation if there ever was one'". And Rebecca's review is even worse, if she says that she actually "regret having seen the film through to the end".

Figure 1: A possible view of Mike's review, highlighting multiple contributions

such a composite view makes it possible to rebuild links between all reviews (even the first two) from the document itself, without requiring remote documents to be available.

Last, but not least, connections across fragments can be used as 'channels to make documents communicate'. Such an approach can be considered as a *simplified form of transclusions*: transclusion as a 'live and bidirectional channel' between pieces of content. The term 'channel' stresses the fact that such a mechanism would allow automatic updates of content, in both directions: from a remote review to all reviews including that fragment and from a transcluded fragment to the original review it belongs to. Yet, the automatic update of content opens tricky issues of synchronization, conflict resolution, priority, content merging and so on. On the other hand, it opens fascinating perspectives for collaborative documents accessing and editing.

The natural candidate to mark-up these composite documents is XML. It is standard, universally supported and very powerful. Surprisingly enough, none of the XML applications we are aware of provides users all these Xanalogical functionalities together. The problem is paradoxically the same hierarchical structure of XML, that makes impossible to address and overlap fine-grained transclusions[12]. The question is now at a different level: to what extent XML transclusions can be implemented? Is it possible to create a fully-fledged Xanalogical environment even for XML documents and use it for 'collaborative reviews'?[6]

4. CHARTING A COURSE FOR XML TRANSCLUSIONS

Intermediate results are also interesting, leading to a full implementation of XML transclusions. This section first discusses some partial objectives - that can be achieved today - and then focuses on long-term developments.

4.1 Step 1: Embedding simplified forms of XML transclusions

An extension of XInclude can support simplified forms of XML transclusions. Let us encode the use-case reviews as DocBook documents, whose quotations are actually XInclude inclusions. Figure 2 and 3 show simplified source codes for Mike's review (including Brad's and Rebecca's reviews) and Brad's one (including David's).

[6]Note that many other scenarios would benefit such collaborative reviews approach. Just think about the creation of multi-source reports in a company that collect information from existing resources, internal documents and external publications.

```
<para>Just read Brad's review: "<xi:include
href="brad.xml" xpointer="..." parse="..."/>".
And Rebecca's review is even worse: she actually
"<xi:include href="rebecca.xml" xpointer="..."
parse="..."/>".</para>
```

Figure 2: XInclude instructions in *mike.xml*

```
<para>I have to fully disagree with David. How
can you say that '<xi:include href="david.xml"
xpointer="..."  parse="..."/>'? I would rather
define it  'a major miscalculation if there
ever was one'</para>
```

Figure 3: XInclude instructions in *brad.xml*

An XInclude processor transforms *mike.xml* into a final document where all inclusions are resolved and content fragments are merged together. The (partial) result we want to achieve enriches that document with detailed information about the origin of each fragment. It is crucial that *all* inclusions traversed to reach the content are stored in the document.

XInclude supports two types of inclusions: (i) text fragments (`parse="text"` and `xpointer` addresses a sequence of characters) or (ii) XML elements (`parse="xml"` and `xpointer` addresses a well-formed XML fragment). The second case obviously requires that the source documents already contain the XML elements to be included. In the example, all quotations are encoded through the DocBook `quote` element in *brad.xml*, *rebecca.xml* and *david.xml*.

A first solution consists of introducing in the output new elements wrapping the included content. For instance, we could use a new element `<xi:included>` decorated with metadata about the inclusion. With that information, we can build applications to let users access metadata, surf to original content, identify contributions, update content, etc.

There are two main drawbacks to such an approach. Firstly, any tool processing the final document has to know about the element `xi:included`. It is then impossible to directly reuse legacy tools such as DocBook converters or renderers. Tools that directly process children of unknown elements (basically 'ignoring' the presence of `xi:included`) could be used as well, but we cannot take such a 'transparent' behavior for granted. A second issue is related to the way users define inclusions. Consider a fifth reviewer quoting fragments from (the expanded view of) *mike.xml*: how can she/he set the `xpointer` attribute of inclusion instructions? She/he has to also consider the `xi:included` elements and write very complex expressions, unless XPointer addresses are filtered to 'ignore' those elements. In any case, a tangled and error-prone management of internal addresses is required. A detailed discussion about these issues can be found in[7], along with a possible solution based on Architectural Forms[15]. A running prototype is also presented in the same paper.

The basic idea of Architectural Forms is to extend the attribute set for an element to express semantic information. Since attributes (that belong to a different namespace) can be added to SGML and XML elements without impacting the document's integrity and without interfering with other applications, Architectural Forms can be exploited to mix information in a transparent manner.

The same approach can be applied to inclusions. The idea presented in [7] is to process all inclusions and 'embed' information in qualified attributes (belonging to a reserved namespace), so that the main structure of the XML document is not altered. The presence of these attributes does not impact the basic processing, parsing and integrity of the document. On the other hand, full expressive information about inclusions is available. Figure 4 shows such a solution applied to the use case.

```
<para>Just read Brad's review: "<quote
xi:inclusion_history="brad.xml#xpointer(...)">I have
to fully disagree with  David. How can you say that
'<quote xi:inclusion_history="brad.xml#xpointer(  ),
david.xml#xpointer(...)"> Australia is good,but not
a  masterpiece </quote>'? I would rather define it
'a major miscalculation if there ever was one'
</quote>". And Rebecca's review is even worse: she
actually "<quote xi:inclusion_history="rebecca.xml
#xpointer(...)">regret having seen the film through
to the end</quote>".</para>
```

Figure 4: Resolving and embedding inclusions in *mike.xml*

The attribute `xi:inclusion_history` is an extension property of the XInclude standard to record data about inclusions. The example uses that attribute to store information about nested inclusions and spread that information all over the XML tree. Consider, for instance, the quotation 'Australia is good, but not a masterpiece': the document knows that it has been included from *brad.xml* but it was actually originated in *david.xml*. Thus, all copies of a given fragment are very well connected in a complex network of inclusions, available to both users and applications.

In practice, `xi:inclusion_history` was never given great importance and it was never really supported by the XInclude implementations. In fact, the prototype presented in [7] uses a different syntax. Syntactic details are not relevant here: what is really important is that record of inclusions are hidden within documents and can be activated on-demand. That information can be exploited to build the aforementioned simplified form of XML transclusions, allowing users to navigate transclusion metadata, to retrieve original content, to highlight multiple contributions and to automatically update content.

There many reasons why such a solution is only a partial step towards our goal. First of all, it only applies to the inclusion of well-formed XML, since qualified attributes - meant to be added to the included `infoset` - can be only added on elements. One of the consequences is that no information about 'dangling' inclusions is eventually stored in the document. The term 'dangling inclusion' indicates an inclusion that failed because content was not available. It may happen for different reasons: errors in the document address or in the internal location of a fragment, temporary or permanent disconnectivity, permission issues and so on. Solutions exploiting XML comments or decoration of preceding/following siblings might be investigated but seem to be tortuous and awkward.

The second - and more important - issue is that such a model does not allow a lot of very common inclusions. Consider, for instance, a reviewer interested in quoting only few words of an existing quotation, or a sentence that partially spans over two paragraphs, or an interval that spans over a bad-formed XML fragment. Such selections can be supported only by exploiting external referencing mechanisms, that go beyond the scope of the strictly hierarchical organization of XML.

4.2 Step 2: Externalizing XML transclusions

A further step to overcome the aforementioned limitations consists of storing data about transcluded content in external and ad-hoc data structures, instead of embedding that information in the document itself. The prototype presented in [7] exports a module handling such externalization, although only some transclusions are actually supported. Figure 5 shows the example document *mike.xml* represented through externalized transclusions.

```
<ted:out-of-band>
  <ted:transclusion
       transclusion-id="d6e7"
       source="brad.xml#xpointer(...)"/>
  <ted:transclusion
       transclusion-id="d9e11"
       source="david.xml#xpointer(...)"/>
</ted:out-of-band>

<para>Just read Brad's review: "I have  to fully
disagree with  David. How can you say that
'Australia is good, but not a  masterpiece'?I would
rather define it 'a major miscalculation if there
ever was one'". And Rebecca's review is even worse:
she actually "regret having seen the film through
to the end".</para>
```

Figure 5: Resolving and externalizing inclusions in *brad.xml*

The content - even the transcluded one - is stored as a plain stream of text while the `ted:out-of-band` data-structure contains pointers to text fragments and metadata about each fragment. Several advantages of such a representation can be outlined. First of all, it contains rich information about the nesting of transclusions as well as rich metadata about original sources, without interfering with users and applications that interact with the document. In fact, all transclusions data are stored externally and can be ignored by transclusion-unaware processors. The position of the `ted:out-of-band` data-structure is also worth discussing: it can be placed within the XML document itself (for instance, as a first child of the root) or even in a completely external file. The first approach produces a single self-contained resource that can be moved from one system to another without requiring any management of relative links; on the other hand, it adds extra structures to the original XML tree. The second approach is completely transparent and decoupled from the document tree-structure but it requires applications to process links and set of resources. Both these approaches are valid, according to users needs and preferences.

However, the main strength of such an externalized solution is its capability of defining transclusions that span over text fragments, ranges or even overlapped transcluded content. While Architectural Forms only work for transcluding well-formed XML fragments - as information is embedded into attributes of the transcluded elements - external pointers may refer to any location in the document, through proper XPointer expressions. For instance, it is possible to identify a transcluded fragment by simply indicating a pointer to the *start*- and *end*- offset of that fragment, and without altering the document itself. Note also that this approach allows users to store information about dangling transclusions (whose content is not available because of network errors or simply because of different regimes in accessing content) by simply adding a reference to the point where the transcluded content was supposed to be. There is no 'intruder' structure in the XML tree but very rich information is available about nested or empty transclusions.

On the other hand, such an approach opens very tricky issues of content merging and manipulation of references. The main problem is related to the 'bad-formedness' of included fragments. External references - and in particular XPointer expressions - allow users to also include *any* piece of an XML tree. Consider, for instance, a user selecting the last sentence of a paragraph and the first one of the following paragraph (a possible HTML source code: `'concluding paragraph</p><p>And starting a new one'`). Several questions arise from that simple selection: which is the best way to include that fragment into a new document? Is it a plain string (without the `p` elements), or does it generate two paragraphs or is it a completely different structure? The adoption of an externalized approach require us to address these issues and to build a general and flexible document architecture for manipulating fine-grained fragments. The consistence of external data structures is another very difficult issue to face. Consider as example the implementation of an editor for such documents: it requires any edit operation to be propagated to the external data structure in a consistent way. Even small changes require a complex network of statements and relations to be updated and manipulated.

4.3 Step 3: Further generalization and externalization via RDF and EARMARK

The previous approach can be improved, although it is enough to provide users with a fully-fledged Xanalogical environment. The idea is to exploit existing XML standards: instead of using ad-hoc data structures and operations, XML transclusions could be described through RDF (and OWL) statements. RDF is a language for representing information about resources in the World Wide Web[9], that allows users to make any assertion on any identifiable resource: statements on any fragment of text, multiple statements on the same resource and overlapped statements are all possible in RDF. The additional benefit of RDF lies on its standardization: expressing transclusions in RDF would allow us to exploit legacy tools of the Semantic Web community and to create sophisticated services of searching and reasoning over transcluded content. Nevertheless, the adoption of RDF leaves still open all the very complex issues related to consistent update and manipulation of externalized transclusions.

We finally propose a further generalization of RDF to integrate in a single approach advantages of both embedded markup (*a la* XML, presented as a first step of our roadmap) and external annotations. A detailed discussion of that proposal - called EARMARK (Extreme Annotational RDF Markup) - can be found in [14]. EARMARK derived from the analysis of some limitations of RDF. The main problem is that RDF assertions need URIs to address resources: statements only apply to whole documents, fragments provided with an identifier, or fragments addressable by some URI schema. The XPointer standard (actually, the full XPointer schema of XPointer [4]) makes it possible to refer to arbitrary pieces of text and XML documents. The adoption of that standard in conjunction with RDF would solve all the URI-related issues. The problem is that the XPointer full schema has never been confirmed by the W3C and its approval seems remote.

The second interesting aspect of RDF is the fact that all URIs are considered 'opaque' strings regardless of the type of resource they refer to. It is very difficult to differentiate an assertion about a text fragment from an assertion about an element, to differentiate classes of assertions and relations among assertions. The solution we propose, EARMARK, defines an ontologically sound model that formalizes the concepts of the XPointer schema, and a natural way to express *externally* both RDF assertions and embedded markup constructs in the same framework. We also implemented a first prototype translating documents from and to EARMARK format, RDF and embedded XML. Although the implementation of EARMARK is at a very early stage, such a 'semantic layer on top of transclusion' is very flexible and promising.

5. CONCLUSIONS

Transclusions have contributed to shape the most successful hypermedia projects, although they were never fully implemented. After several years from their invention - dated back to the early 60s - they still keep the original attractiveness and potentialities. The challenge for the community is now to adapt the original Xanalogical design to the current technologies and systems, in particular to the World Wide Web and XML.

The contribution of this paper is manifold: (i) a preliminary discussion of simplified forms of transclusions, viable for the world of XML, (ii) a roadmap to actually implement XML transclusions and (iii) a brief presentation of early prototypes we are developing in that direction.

There are actually other crucial issues not discussed in this work: content merging, synchronization and access rights. They all require a much deeper analysis and experientation we haven't done yet. Our impression, however, is that externalizing transclusions pave the way towards these goals. In fact, a flexible and fine-grained model makes it possible to integrate multi-source and multi-author fragments. An orthogonal step will be designing a distributed architecture that propagates changes - in both directions - and that handles access permissions.

In conclusion, a lot fascinating challenges await us on the horizon. It is our intention to foster the discussion within the community and to actually integrate Xanalogical functionalities in a renewed generation of Xanadu-like systems.

6. REFERENCES

[1] K. M. Anderson, R. N. Taylor, and E. J. Whitehead, Jr. Chimera: hypermedia for heterogeneous software development enviroments. *ACM Trans. Inf. Syst.*, 18(3):211–245, 2000.

[2] L. Carr, D. De Roure, W. Hall, and G. Hill. The Distributed Link Service: A Tool for Publishers, Authors and Readers. *The Web Journal*, (1), 1995.

[3] S. DeRose, E. Maler, and D. Orchard. XML Linking Language (XLink) Version 1.0. http://www.w3.org/TR/xlink/, 2001. W3C Recommendation.

[4] S. J. DeRose, E. Maler, and R. Daniel. XPointer xpointer() Scheme. World Wide Web Consortium, Working Draft WD-xptr-xpointer-20021219, December 2002.

[5] B. DuCharme. Transclusion with XSLT 2.0. http://www.xml.com/pub/a/2003/07/09/xslt.html, 2003.

[6] W. Hall, H. Davis, and G. Hutchings. *Rethinking Hypermedia: The Microcosm Approach*. Kluwer Academic Publishers, Norwell, MA, USA, 1996.

[7] A. D. Iorio and J. Lumley. From XML Inclusions to XML Transclusions. To appear in the Proceedings of ACM Hypertext 09.

[8] J. Kolbitsch and H. Maurer. Transclusions in an HTML-based environment. *Journal of Computing and Information Technology*, 14(2):161–174, 2006.

[9] F. Manola and E. Miller. RDF Primer. http://www.w3.org/TR/rdf-primer/, 2004. W3C Recommendation.

[10] J. Marsh, D. Orchard, and D. Veillard. XML Inclusions (XInclude) Version 1.0. http://www.w3.org/TR/xinclude/, 2006. W3C Recommendation.

[11] T. H. Nelson. Transliterature: A Humanist Format for Re-Usable Documents and Media. http://translit.org/.

[12] T. H. Nelson. Embedded markup considered harmful. *World Wide Web J.*, 2(4):129–134, 1997.

[13] T. H. Nelson. Xanalogical structure, needed now more than ever: parallel documents, deep links to content, deep versioning, and deep re-use. *ACM Comput. Surv.*, page 33, 1999.

[14] S. Peroni and F. Vitali. EARMARKing documents for arbitrary, overlapping and out-of-order annotations. To appear in the Proceedings of ACM DocEng 09.

[15] J. K. Truss. International Organization for Standardization. A.3 Architectural Form Definition Requirements (AFDR). In *In ISO/IEC 10744:1997, Annex A, SGML Extended Facilities*, pages 1601–1608, 1997.

[16] M. Tzagarakis, D. Avramidis, M. Kyriakopoulou, m. c. schraefel, M. Vaitis, and D. Christodoulakis. Structuring primitives in the callimachus component-based open hypermedia system. *J. Netw. Comput. Appl.*, 26(1):139–162, 2003.

[17] E. Wilde. XIPr: XInclude Processor. http://dret.net/projects/xipr/, 2007.