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

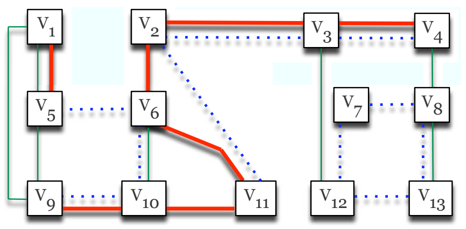

Figure 1: An example of zz-structure.

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.

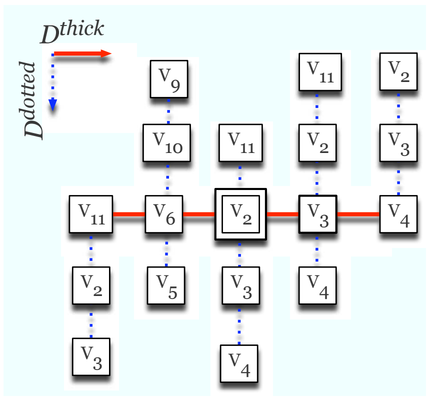

Figure 2: An H-view related to the zz-structure of Figure 1.

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^{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, exten sible 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.

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.

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:

- 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 Actor-based of Zz-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 electroacoustic 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 Bioinformatics 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.