OPEN, DYNAMIC ELECTRONIC EDITIONS OF MULTIDIMENSIONAL DOCUMENTS

Sergio Canazza
University of Udine
Via Diaz 5, 34170 Gorizia – Italy
sergio.canazza@uniud.it

Antonina Dattolo
University of Naples Federico II
Via Cintia, 80126 Napoli – Italy
dattolo@unina.it

ABSTRACT
This paper proposes the application of a new actor-based extension of Nelson's ZigZag model in humanities, in order to create graph-centric browsing tools for Electronic Editions of multidimensional documents.
The combination of the ZigZag model and of cooperation activities of different actor classes allows our model to create innovative, graph-centric browsing perspectives for the user and to offer to him/her authoring tools for the runtime creation of new virtual sources.

KEY WORDS
Multimedia Information Systems, Software Agents, Data Modelling, Education, zzStructures, Electronic Editions.

1. Introduction

This paper proposes the application of a new actor-based extension of Nelson's ZigZag model in humanities, in order to create graph-centric browsing tools for electronic editions of musical works, considered as multidimensional documents, i.e. *open systems* constituted by text, performance, tradition, interpretation, fruition, reception.
In textual philology, an Electronic edition is a well-defined concept and it studies models to represent text and documents, starting from the generative process of a text and its transmission through the years, highlighting the relations among the original source(s), its (their) copies, versions or variants (often *authorial* variants).
Analogous Electronic Edition concept is not formally defined for multidimensional sources.
In the Electronic Edition of a musical work, the situation is more complex than in textual field.
In multimedia artistic works, where the audio recording is included in a complex procedure of audio signal processing and where different *writing* systems flow together, emerges the necessity to define suitable data structures and to convey, in a single (digital) medium, verbal and musical documents, pictures, audio and video signals.

The aim of this paper is to provide an innovative system for the preservation and the access to electro-acoustic music documents, considered as a representative subset of multimedia artistic works.
Usually, the electro-acoustic music scholar must manage a lot of sources stored in different media: notes, sketches, musical scores, photos, articles, audio documents, video recordings.
Therefore, it is necessary to compare various audio documents on different abstraction levels (MPEG7-like) and various media (as text, images, video). In electro-acoustic music field, audio documents cluster an extended set of additional information:

1. primary information: master recordings; n-th generation duplications.
2. meta–data: cover annotations (see Figure 1a); reels annotations; tape annotations (see Figure 1b); annexes; other written as labels, titles, etc.; information relating to the document production system used; information relating to composition and editing techniques.

a)
b)

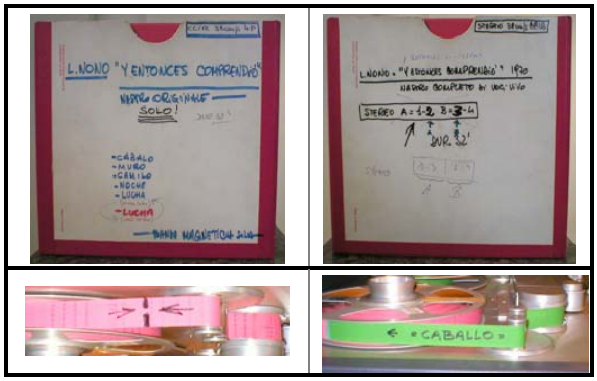

Figure 1. Meta-data examples in audio documents: a) cover annotations, b) tape annotations.

All these data should be preserved and compared among them in an Electronic Edition. It must be highlight that, in the past several years, the lack of musicological rigor and attention to source consistency has already produced relevant mistakes in the analog–digital transfer of electro–acoustic musical archives, generating yet more sources which often feature disfiguring transmission errors.
Some data (for example, textual elements written on media containers such as reel covers, reel flanges, tape marks, etc.) might not be completely trusted (reels are often re–used/misplaced/etc.), but they may provide

essential insight in production schemes and copy generations.
For these and other reasons, the resources needed for the realization of an Electronic Edition are enormous and present complex interconnections, difficult to organize and to visualize.
An additional and important issue is the introduction of authoring tools, that make *open* a musical work, enabling the user to become self-author of new versions of a given work. At the end of the 50's, this concept of *open work* in musical field has been introduced by Pousseur [1]: he thought of a user, capable enough to handle magnetophones for a full usage of the work, who may became self-interpreter in order to elaborate the received message and to create new versions of the same work.
Thus, it is now time to take one step further:

1. proposing innovative philological solutions for electronic editions;
2. using navigation models able to handle complex, multidimensional documents (that is documents analyzed in different domain: symbolic, time, frequency, etc.);
3. generating authoring tools for the runtime creation of virtual sources in order to handle back to the user the option of creative new version of an open work.

These topics are faced using an actor-based extension of the ZigZag model. Conceiving a system as aggregation of autonomous and cooperative agents is one of the most exciting aspects of the challenging arena known as multi-agent system (MAS); this perspective revolutionizes radically the way in which a model may be conceived and work [2]. In addition, the presence of ZigZag model guaranties a graph-centric browsing tool, that supports the representation of persistent context: user sees the node of interest in relation to other associated nodes. But more than this, user sees the node of interest from that node's position relative to multiple perspectives or orientations [3]. As case study, we chose a complex electro-acoustic musical work (*Invenzioni su una voce* by Bruno Maderna), characterized by a variety of sources and of different authorial versions.

2. Existing Models

In this context, the computer science contribution is limited to handle philological approach to *textual* data, that can be considered only a sub-part of our work focus.
The philological approach to textual data has been redefined according to hermeneutic contribution from the computer science. In the field of the technological applications to ecdotic surveying, remarkable progresses have been made in the recent years, but the distrust from philologists and the *naïve* use of computer science (that is, the employment of tools instead of sharing formal models), don't permitted to carry out a real progress in this field.
In past decades, computer science has been used to preserve sources, to data collect and to present Electronic Editions.
Actually, there is a lack of standards in text structuring: in fact, usually, mark-up languages are used to organize the text and to retrieve information. Structure and contents are not separated and this causes the impossibility of considering adequate complex models (often HTML is used) and the use of not general tools to access contents.
The textual philology often used HTML as an encoding model: this choice is justified not by the virtues of the encoding scheme itself but by the fact that HTML is directly interpreted by Web browser. Thus an electronic edition can be disseminated very quickly, and it can easily incorporate links to images, notes, audio, video, and other apparatus.
Unfortunately the real problem with HTML is in actual tags, and in what they indicate. HTML uses a rather peculiar mixture of simple structural tags and some typographical tags. These tags cannot be used for complex searching of the text.
SGML and XML allow authors to define mark-up languages; new mark-up tags, and the relationship among them, are defined in a *document type definition* (DTD), which gives a formal description of the document structure. The DTD's design - to determine what is important and what should be encoded – is a process known as *document analysis*. Fortunately for editors of electronic editions, a great deal of SGML work (and, more recently, XML) has already been done in the humanities. Text Encoding Initiative Guidelines (TEIG) [4] are an international and interdisciplinary standard that enables libraries, museums, publishers, and individual scholars to represent a variety of literary and linguistic texts for online research, teaching, and preservation. TEIG defined four hundred features that might be of interest in electronic text and to specify SGML/XML tags for them [5]. Unfortunately, SGML and XML assume that the text consists of a single hierarchic structure: this produces problems in overlapping structures, a phenomenon that occurs frequently in humanities material and that involves pages and folios [6].
Some other mark-up schemes are capable of handling overlapping structures. Perhaps the best-known of these is the COCOA (word Count and Concordance on Atlas) used by the Oxford concordance Program (OCP) and by the Text-Analysis Computing Tools (TACT) suite. Because it has no end-tag syntax, it offers some flexibility for handling overlapping structures, but it also means that the mark-up cannot be validates (since the structure of the text is not predictable). The Wittgenstein Archives Project at the University of Bergen has defined an encoding scheme called Multi-Element Code System (MECS) that can handle overlapping structures. It contains some of the properties of SGML, but it also contains additional mechanisms for representing structures that are cumbersome in SGML, permitting overlapping structures.
Unfortunately, MECS can be processed only by way of software written at Bergen.

3. Actor-based ZigZag Model

This section introduces respectively in subsections 3.1 and 3.2 the basics of the ZigZag model and of the actor model, in order to present our model in section 4.

3.1 The ZigZag model

ZigZag [7] introduces a new, graph-centric system of conventions for data and computing; it separates the structure of information from its visualisation (i.e. the way the data – text, audio, video - is presented to the user); therefore a ZigZag structure handles all the different visualisations necessary to realize an Electronic Editon of musical work.
A zzStructure can be viewed as a multigraph where edges are colored, with the restriction that every vertex, called *zzcell*, has at most two incident edges of the same color; the subgraphs, each of which contains edges of a unique color, are called *dimensions*. The cells in a same dimension are linked into linear and directed sequences, called *ranks*. Each dimension can contain a number of parallel ranks, which are a series of distinct cells connected sequentially.
Since there is no canonical visualization, the pseudospace generated by zzStructures is called *zzspace* and may be viewed in various ways. A zz*view* is a presentation of some portion of zzspace and is presented by a view program, which visualizes, for example, a region around a particular cursor.
A *2D zzview* can be drawn picking a single cell as a focal point, and drawing the neighbourhood around that cell along two chosen dimensions. By changing the chosen pair of dimensions, we can visually reveal, hide, and rearrange nodes in interesting ways. Considering that a zzStructure may be very large, and that there is usually not enough room in the 2D view for all of the cells, we restrict the dimension of the 2D view.

Some observations are necessary on the zzcells; they are the principal unit of the system and they are conceived not only as passive containers of primitive data (i.e., text, graphics, audio, etc.) but they can have types, based either on their functions or on the types of data they contain. Thus, a zzcell may have a variety of different properties and functions, such as executable program or scripts (this type of cell is called progcell), or represent the package of different cells (this type of cell is called referencial cell).
Analogous observations can be made on the dimensions; in fact, they can be passive and nominal (merely receiving and presenting data) or operational, programmed to monitor changing zzStructures and events, and calculate and present results automatically (for example, the dimensions d.cursor and the dimension d.clone).
From these considerations, it turns out that it is reductive to associate zzcells to passive entities meant simple nodes are in a graph. So, we have considered the opportunity to model a zzcell by means of a specific class of computational agents, the actors.

3.2 Brief description of the actor model

The actor model [8] is a model of concurrent computations in distributed systems; it is organized as a universe of inherently autonomous computational agents, called actors, which interact with each other by sending messages, improving on the sequential limitations of passive objects.
Each actor is defined by three parts: a *passive part*, which is a set of local variables, termed ***acquaintances***, that constitute its internal state; an *active part*, that reacts to the external environments by executing its procedural skills, called ***scripts***. This active part constitutes actor's behaviour; third part is represented by the actor's *mail queue*, that buffers incoming communication (i.e. messages).
Each actor has a unique name (the uniqueness property) and a behaviour, and it communicates with other actors via asynchronous messages. Actors are reactive in nature, i.e., they execute only in response to messages received.
An actor can perform three basic actions on receiving a message: *create* a finite number of actors with universally fresh names, *send* a finite number of messages and assume a new behaviour.
An actor's behaviour is deterministic in that its response to a message is uniquely determined by the message contents and its internal state. Furthermore, all actions performed on receiving a message are concurrent.
In order to describe the actors in our model we adopt the formalism used in [2]:

```
(DefActor ActorName
   [inherits-from Class-Name]
   <acquaintances list>
   {scripts list} )
```

Therefore an actor is described by specifying its superclass, its data part and its script part; the script part represents the set of scripts which can be executed.

4. The Model

The architecture of our model is organized in two layers: a *component layer* contains the zzcells, those are actors specifically designed to model the audio documents domain; a *meta layer* contains the actors classes specialized, for example, to manage connections among zzcells or to generate specific views on them. The interaction between actors is defined using the diagrammatic language AUML (Agent Unified Modelling Language) [9], extension of UML for agents.

4.1 Component layer

The component layer is defined in relation to the magnetic tape structure: each open reel is usually composed by several physical segments, i.e. pieces of magnetic tape connected by means of adhesive tape (called junction).

In each segment, the audio signal is recorded in one, two or more tracks. Following this structure we define the actors Source, PhysicalSegment, and NumericalSignal. Moreover, the actor LogicalSegment is introduced, with the aim to compare the sources on the basis of a segmentation that is different by the physical one.
The actor Source represents the overall characteristics of the document, such as the tape width (typical values are 1/4, 1/2, 1, and 2 inch) and the cataloguing fields.

```
(DefActor Source
    <physicalSegments width archive shelfMark
    inventory conservationCondition ...>
    {calculateDuration ...} )
```

It contains also a list of phisicalSegments, which compound the open reel tape. This actor is able to perform several actions, e.g. the script calculateDuration asks each physical document for his length and rate and it calculates the total duration of the tape.
The LogicalSegment carries out a virtual partition of the Source.

```
(DefActor LogicalSegment
    <start length source rate equalization
    noiseReductionSystem
    tracksLayout numericalSignal audioQuality …>
    {getQuality getSignalProperties …} )
```

Its acquaintances are the start time and the duration of the segment, a pointer to the corresponding source, the recording rate (tipical values are 7.5 or 15 inch/s), the equalization (e.g . IEC1/CCIR), the noiseReductionSystem (e.g. Dolby or dbx), the tracksLayout (i.e. the tracks number and width), a pointer to the digitalSignal representing the audio recorded on each track, and an audioQuality index, that can be subjectively specified by the user.
If the audioQuality field is left blank, the script getQuality asks the numericalSignal to estimate the signal to noise ratio of each track and returns an index of quality.
The script getSignalProperties asks the digitalSignal for the digital audio of each track and estimates if the audio signal is monophonic, stereophonic, or polyphonic. This information doesn't always correspond to the number of tracks, because, following the practice of magnetic recording, a monophonic signal can be recorded on a multi-track format, writing the same signal in all the tracks.
A specialization of this actor class is the PhysicalSegment actor that specifies geometrical features of the segment, such as the angle of his junction.

```
(DefActor PhysicalSegment
    [inherits-from LogicalSegment]
    <junctionAngle …>
    {getFadeDuration …} )
```

The script getFadeDuration, starting from the geometrical properties of the junction, calculates the duration of the fade-in and fade-out of the audio at the segment edges.
The NumericSignal actor represents the audio signal recorded on the tracks.

```
(DefActor NumericSignal
    <size samplingRate resolution data[i,j]>
    {estimateSNR calculateSTFT
    calculateAmplitudeEnvelope} )
```

Its acquaintances are related to the numeric format: number of samples, sampling rate, resolution.
data[i,j] is a matrix, with a row for each track and a column for each sample. This actor can perform several actions that, using signal processing techniques, calculate signal to noise ratio, short-time Fourier transform, amplitude envelope or other kinds of representations.

4.2 Meta Layer

In order to manage the information in a zzStructure context, a meta layer is added. Following the rank definition introduced in ZigZag model, the actor Rank can be described as an ordered list of actors, belonging to the same dimension.
Each specific dimension contains one or more ranks. Examples of dimensions are:

- *source* dimension, that composes given sources as an ordered sequence of segments;
- *equivalent segments* dimension: links segments with a common music content present in different sources.

For each dimension, we define specialized classes of actors, that we use in the case study (such as SourceRank and EquivalentSegmentsRank).

```
(DefActor SourceRank
    [inherits-from Rank]
    <sources ...>
    {swapSegment ...} )

(DefActor EquivalentSegmentsRank
    [inherits-from Rank]
    <sources ...>
    {compareQuality ...} )
```

Another typology of meta-actor is the generativeProcess; this class is devoted to generate dynamic views and new virtual hyperdocuments.

```
(DefActor generativeProcess
    <… >
    {generateView createSource} )
```

5. Case Study

We pass now to the analysis of an interesting case study, the Maderna's electronic work *Invenzione su una voce*. As explained previously, in electro-acoustic music field the original work itself often undergoes:

- a number of duplications;
- many modifications by the author(s) which are often executed on later-generation duplications.

These mutations generate a considerable asset of sources with different kind of noise: broadband noise due to the analogue copy process (with a consequent Signal-Noise Ratio - SNR - decrease) and local noise due to new splice added by the Composer or by the Technician.

In our case study, we have collected 19 different audio documents, related to the Maderna's work and founded in 5 European different archives. The sources have different signal properties (monophonic and stereophonic), different audio qualities, partially different musical content, and a duration ranging from 10'50" to 18'51".
Thanks to the cooperation of different classes of actors, our system allows user-author to surf among the existing recordings and to create a new virtual source, automatically picking up the audio signal with higher SNR or without impulsive noise.
Thanks to the separation of structure (dimensions) from content (zzcell) in zzStructures, different users can customize their workplace and the visualisation and of same contents, creating new dimensions or new virtual sources. This is achieved activating specific actor collaboration schemes and is shown in following sections 5.1 and 5.2.

5.1 Generating dynamic views

Our aim is to define a view that allows user to navigate in an effective and intuitive way into the different sources of *Invenzione*. The first problem to face is the definition of a logical segmentation of the sources, suitable to a comparison among the variants. To this aim, we propose a logical segmentation based on the concept of equivalent segment. We define the *equivalent segment* as the biggest common sections among two or more sources. Let us consider two different sources of this work, named tr530 and fon21. These sources can be compared on the basis of their musical content, in order to find commonalities and differences. Following this criteria, the initial part of tr530 and fon21 can be segmented as showed in Figure 2. Logical segment $tr530_1$ is equivalent to $fon21_1$, $tr530_3$ is equivalent to $fon21_2$, and $tr530_5$ is equivalent to $fon21_3$, whereas $tr530_2$ and 530_4 don't have an equivalent segment in source fon21.

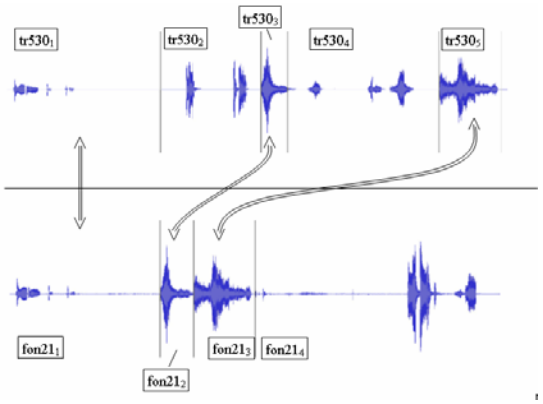

Figure 2. Amplitude envelope of tr530 and fon21 with equivalent segments.

This segmentation process can be iteratively applied to all collected sources, obtaining a set of logical segments linked along two dimensions.
Figure 3 shows a graph-centric view for the initial section of three sources, fon21, tr530, tr532: the first dimension, named *source*, links each segment to the previous and next segments of the same source (e.g. $tr530_3$ is linked to $tr530_2$ and $tr530_4$); the second dimension, named *equivalent segments*, links equivalent segments present in different sources (e.g. $tr530_3$ is linked to $fon21_2$).

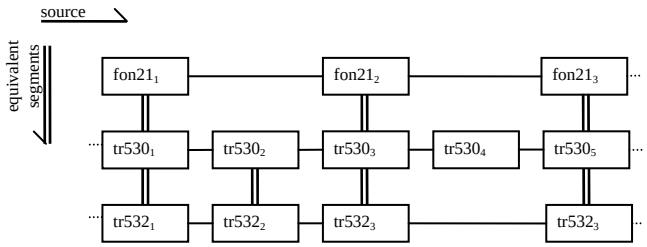

Figure 3. This zzview shows *source* and *equivalent segments* dimensions.

5.2 Creating virtual sources

An another interesting aspect of our model is related to the active role of user, that can generate new virtual sources for a given music work; this may happen by mixing different audio tracks present in different existing sources. To do so, user can apply deterministic laws, stochastic models or self-oriented choices. This allows user to generate new reading sequences of an open work. In this section we describe cooperation schemes get ready among computational agents aimed to automatic generation of a new virtual source, optimized on Signal to Noise Ration (SNR).

Starting from a given *source* vs *equivalent segments* view, we assume that an user is interested in creating the new optimized virtual source from a personalized sequence of logical segments src_1, …, src_n; this request is captured from the meta-actor generativeProcess and it is forwarded from it to the source rank r^{src} (that manages the logical segments src_1, …, src_n of the source src). The collaboration schemes generated among the different classes of actors are shown in the sequence diagram of Figure 4.
r^{src} sends the synchronous multicast message (bestQuality) to all its logical segments. In order to enable the comparison among its quality and that of equivalent segments, each segment (src_j, j=1,...n) assigns this task to the rank $r^{e\text{-}sj}$ (that manages the logical segments equivalent to the src_j). Each $r^{e\text{-}sj}$ contacts (in synchronous multicast way) all its components ($e\text{-}src_{js}$, s = 1, …,m) and, once it has received from them their quality, it choices the component that has best quality. This information is returned to each logical segment, and successively to generativeProcess. This last actor collects the best quality equivalent segments ($e\text{-}src_{k1}$, ..., $e\text{-}src_{nj}$) and creates the new requested virtual source.

An effect of this cooperation activity is the creation of a new virtual source that can be added to zzview proposed in Figure 3.

6. Conclusion

Existing encoding models for Electronic Edition (HTML, XML) fail in representing adequately the complexity of documents transmission phenomenology. At the same time, few projects have a methodological orientation; most are geared to preparation of specific editions, with less interest in the long-term implications. In this paper we introduced a computational actor model, based on ZigZag model, for implementing the functions that, in a critical edition, are carried out by the apparatus - a tool, such as variant readings, textual notes, a concordance, a bibliography, designed for the use of scholars. This model is able to handle complex multidimensional documents and, by generating authoring tools for runtime creation of virtual sources and for analysis in different domains, permits one to propose innovative philological solutions for Electronic Editions in different fields. An interesting application area is the Musical Theatre; in this case it is necessary to manage documents in different domains (video recording, audio signal, text, photos, etc.). Actually these documents are catalogued in heterogeneous repositories navigable in separate ways. Our open methodology could permit to rebuild the whole *opera*.
In order to stress the model, we showed a very complex case study of Electronic Edition: Maderna's *Invenzioni su una voce*. Our work with the Electronic Edition should show how far one might go in integrating a hypermedia approach to audio mark-up and musical analysis. Electronic Edition of electro-acoustic music is so intimately bound up with the audio/video signal processing and computer science that the issues must remain at the forefront of attention of textual scholar, musicologists, art historians, film scholar, engineering and computer scientist. In particular, scholars typically possess a phenomenological understanding of such materials that is obscure to techno-scientific researchers.

Acknowledgements

The authors would like to thank Dr. Pietro Pipi for his careful revision of this paper.

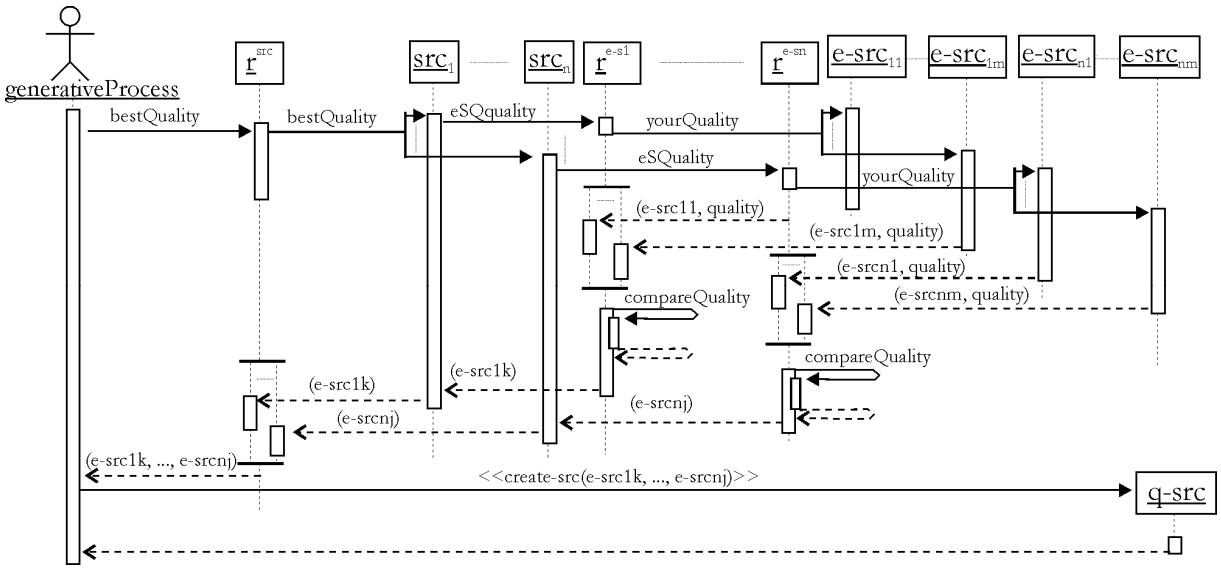

Figure 4. The AUML sequence diagram generating a new virtual source

References

[1] Pousseur, H. Scambi. In *Gravesaner Blätter* IV, 1959, 36-54.
[2] Dattolo A. and Loia, V. Distributed Information and Control in a Concurrent Hypermedia-oriented Architecture. *International Journal of SEKE*, *10*(6), 2000, 345-369.
[3] McGuffin, M. J. and Schraefel, M.C. A Comparison of Hyperstructures: Zzstructures, mSpaces, and Polyarchies. *Proc. 15th ACM Conference on Hypertext and Hypermedia, August 9-13, Santa Cruz, California, USA*, 2004, 153-162.
[4] Sperberg-McQueen, C. M. and Burnard, L. *Guidelines for Text Encoding and Interchange* (University of Oxford, 2002).
[5] McGann, J. Comp[u/e]ting Editorial F[u/ea]tures. In Neil Fraistat, ed., *Re-imagining Textuality*, Milwaukee: Univ. of Wisconsis Press, 2002, 17-27.
[6] Hockey, S. The reality of electronic Edition. In Modiano R., Searle L.F., Shillingsburg P., ed., *Voice, text, hypertext*, University of Washington Press, 2004, 361-377.
[7] Nelson, T. H. Cosmology for a Different Computer Universe: Data Model, Mechanisms, Virtual Machine and Visualization Infrastructure. *Journal of Digital Information*, *5*(1), 2004.
[8] Agha, G. *Actors: A Model of Concurrent Computation in Distributed Systems* (MIT Press, Cambridge, MA, 1986).
[9] Winikoff, M. toward making agent UML practical: a textual notation and a tool. *Proc. first Int. Workshop on Integration of Software Engineering and Agent Technology*, Melbourne, Australia, 2005, 401-412.