Śivadharma Database CMS.
HTML and graph as a starting point for digital editions

Martina Dello Buono[1], Francesca Tomasi[2]
[1] University of Bologna, Italy – martina.dellobuono2@unibo.it
[2] University of Bologna, Italy – francesca.tomasi@unibo.it

ABSTRACT
Śivadharma Database is a Content Management System (CMS) for creating, publishing, and updating Digital Scholarly Editions. By entering the *textus constitutus* first, scholars can annotate it via user-friendly tools to provide their editions with additional components, such as apparatus, notes, translation, parallels, and citations.
The annotation system is based on HTML and graph data structure. The *textus* corresponds to an HTML file ready to be displayed. Textual fragments annotated in HTML are the atomic unit of multiple relationships organized in a graph database, allowing the formal representation of complex and overlapped textual phenomena.
These methodological and practical approaches result in good annotation expressiveness and a minimum effort to learn technologies during the editing workflow by scholars.

KEYWORDS
Digital Scholarly Editing; Text Encoding; HTML; Graph Database; CMS

1. INTRODUCTION

One of the most frequent issues in the Digital Scholarly Editing field is the best encoding for digital editions. XML/TEI[1] is the traditional solution [7]. Another proposal is the still-emerging RDF[2], applied in the context of the Semantic Web. They both have pros and cons. XML/TEI is popular and easy to learn. However, the extent of its vocabulary undermines the choice of descriptors by editors and the interoperability of editions [16]. Furthermore, its limitations in representing complex textual phenomena, e.g., overlapping hierarchies, are known in the literature [10, 12, 15, 17].
On the other hand, RDF can solve the inherent problems of an XML-based tree structure, but it is a more expensive solution regarding learnability. In addition, visualization is not in the scope of RDF [1].
Another critical issue related to text encoding goes beyond its methodological implications. The encoding workflow often requires high technical skills and tasks, e.g., learning a specific language [11, 13, 14]. Scholars, instead, should have *user-friendly* software, tools, and services to streamline work.
Therefore, technological choices have an impact not only on the expressiveness of the annotations but also on the scholars' work itself. On the one hand, they must be able to manage textual complexity; on the other, they must facilitate the annotation workflow. Therefore, finding a solution that meets both needs is necessary.
The *Śivadharma Database* project proposes a solution to the highlighted issues by implementing a CMS based on HTML and graph data structure to create editions from scratch via an interface.
The present paper is structured as follows. Section 2 inspects the pros and cons of the currently available tools for text encoding. The following Section 3 reports the *Śivadharma* use case. Then, Section 4 focuses on the methodology to design the *Śivadharma Database* CMS. In conclusion, Section 5 analyzes the *Śivadharma Database* annotation system.

2. STATE OF THE ART

Scholars have few user-friendly tools for text encoding at their disposal. Some of these tools align their annotation systems to traditional XML/TEI for the sake of technology popularity and reusability, while others experiment with emerging technologies, e.g., RDF.
EVT[3] is undoubtedly the most widespread tool in the Digital Scholarly Editing field. Based on XML/TEI, it is not precisely an annotator but a viewer of digital editions. It inputs an XML/TEI file and returns its visualization via a specific interface.

[1] Text Encoding Initiative: https://tei-c.org/
[2] RDF: https://www.w3.org/RDF/
[3] EVT: http://evt.labcd.unipi.it/

Basic released functionalities can be checked in the pilot EVT project, *The Digital Vercelli Book*[4], a diplomatic and diplomatic-interpretative digital edition. More advanced functionalities, such as the critical apparatus and parallel texts, are available in *Edizione Logica Avicennae* [experimental encoding][5]. The main drawback of such a system is the required scholars' prior knowledge of the specific XML/TEI schema to visualize their editions.

In other cases, RDF is the framework on which annotation systems rely. For instance, KWICKWOCKWAC (KeyWords In Context, KeyWord Out of Context, KeyWord Alongside Context)[6] exploits the potentialities of RDFa[7]. The *Edizione Nazionale delle Opere di Aldo Moro* project [8] shows that KWICKWOCKWAC allows annotating entities, such as persons and places, in full-text documents. The results of the annotation workflow are RDFa documents.[8] It implies embedding the annotations into texts, thus undermining their readability.

The SPEEDy editor [9] allows the same annotations, e.g., persons and places. However, it differs from KWICKWOCKWAC in the technologies implied. Since it records the annotations as stand-off properties in Neo4j[9], a graph database, it can elegantly handle complex and overlapped structures.

As KWICKWOCKWAC and SPEEDy, *Pundit*[10] provides a user-friendly interface to highlight and comment on web pages. An example of its use is available in the *Burckhardt Source*[11] project, a collection of letters from and to Jacob Burckhardt where annotations concern the persons, places, artworks, and bibliographies mentioned [3]. In addition, one specific version of *Pundit*, i.e., *Pundit Annotator Pro*[12], exploits semantic technologies to create semantic annotations and link them to external resources, e.g., Wikipedia entities.

Such projects, i.e., KWICKWOCKWAC, SPEEDy, and *Pundit*, do not require coding skills since a graphical interface allows user-side annotation control. On the other hand, their annotation systems do not allow the creation of sophisticated edition components, such as the apparatus.

This task, instead, can be accomplished in the *Classical Text Editor*[13] environment. It supports advanced features such as the apparatus, notes, and parallel text creation. It also allows the XML/TEI export for interoperability [5]. However, its license, as in the case of *Pundit*, does not allow widespread use in the community. In addition, it is more focused on printed editions.

3. USE CASE

Translocal Identities: The Śivadharma and the Making of Regional Religious Traditions in Premodern South Asia, better known as the *Śivadharma Project*, is an ongoing ERC project. One of its objectives is to produce Digital Scholarly Editions of the Sanskrit texts of the so-called *Śivadharma corpus*. Written around the 6th and 7th centuries, they describe the duties a Śaiva devotee must observe [2]. They are very complex, starting from their rhythmic structure. They are in stanzas, each of which includes four quarters, Skr. *pādas.* They are conventionally called *pāda* a, *pāda* b, *pāda* c, *and pāda* d, and do not match with lines [6]. Moreover, specific textual features often break the boundaries of *pādas* and overlap. For instance, Fig. 1 shows that the string identified as the *lemma* in an apparatus entry does not fit in a single *pāda* but spans over two ones, thus undermining the traditional XML-based tree structure commonly used in text encoding.

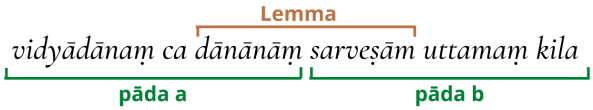

Figure 1. Lemma breaking the boundaries of *pādas*.

Other phenomena, such as omissions, lacunae, transpositions, and many translations and parallels, underline the same issue. It follows that it is necessary to design a specific data structure for text encoding to handle this complexity.

Leveraging specific technologies such as HTML and a graph database, we outline methodological and practical approaches to overcome the limitations in the annotation workflow.

[4] The Digital Vercelli Book: http://vbd.humnet.unipi.it/beta2/
[5] Edizione Logica Avicennae [experimental encoding]: http://evt.labcd.unipi.it/demo/evt2-beta2/avicenna/index.html#/readingTxt?d=doc_1&p=C-112v&s=text-body-div&e=critical
[6] KWICKWOCKWAC: https://aldomorodigitale.unibo.it/about/docs/processing#structure-section
[7] RDFa: https://rdfa.info/
[8] *Edizione Nazionale delle Opere di Aldo Moro* technical specifications: https://doi.org/10.5281/zenodo.5144961
[9] Neo4j: https://neo4j.com/
[10] Pundit: https://www.netseven.it/pundit/
[11] Burckhardt Source: https://burckhardtsource.org/
[12] Pundit Annotator Pro: https://thepund.it/annotator-pro-web-annotation/
[13] Classical Text Editor: https://cte.oeaw.ac.at/

4. METHODOLOGY

The present research proposes HTML and graph data structure as a starting point for encoding digital editions. It is a practical strategy for several reasons. First, HTML is perfect for visualization since it does not need specific conversions. Especially if a digital edition is a short-term project, an HTML encoding can satisfy the basic request to display the edition. In addition, it can be the matrix of conversion in other languages.

The graph data structure allows the formal representation of complex and overlapping textual phenomena identified by editors. HTML can be the starting point for modelling the data of editions as a graph. Specifically, a textual fragment identified in HTML can work as a graph external database's atomic unit for multiple relations, even with external resources. So, the interpretation by editors becomes a stand-off layer, thus increasing the readability of texts and improving data interoperability [16].

Finally, such technologies can be solid pillars for building a user-friendly stand-off annotation CMS [16] based on *forms* to encode editions via an interface. A solution of this kind can allow editors to focus only on scholarly questions and decrease the potential occurrence of errors. Furthermore, it can allow complete annotation control if the text changes. The atomic unit of annotation is the fragment, which is directly selected from the text by the editor, uniquely identified, and sent to the database via a form. If the editor modifies an already identified fragment, the system detects the change by verifying its ID (Identifier) and automatically sends it back to the database. At the same time, its assigned annotation remains linked to the fragment and does not change.

A system of this kind can support the creation of even complex components of editions, e.g., apparatus, notes, parallels, citations, and translation. Since the fragment is the basic unit and it is possible to associate it with *n* relations, building and visualizing the apparatus, for instance, is accessible. As shown in Fig. 2, the selected fragment `:selectedFragment` is the subject of the relation `:HAS_LEMMA`. The object `:lemma` is, in turn, the subject of the relation `:HAS_VARIANT`. Then both `:lemma` and `:variant` are subjects of the relation `:ATTESTED_IN` to link them to their witnesses.[14]

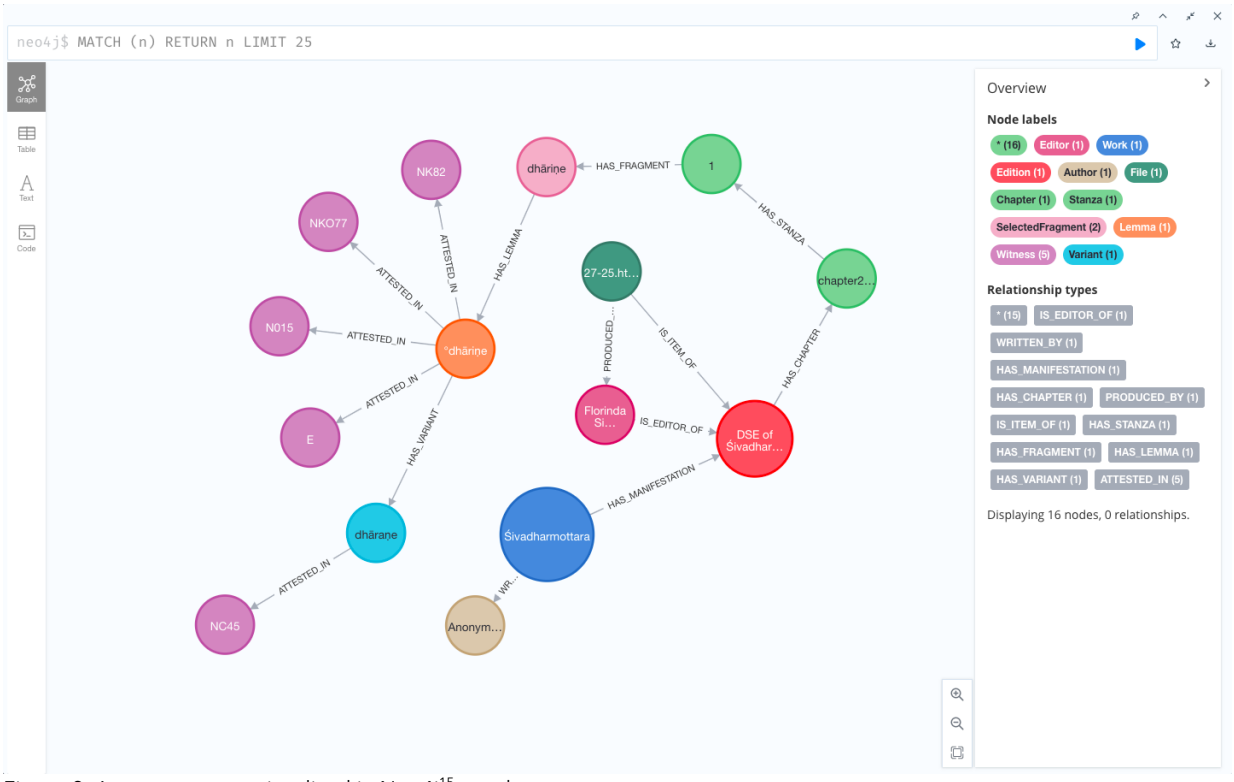

Figure 2. Apparatus entry visualized in Neo4j[15] graph.

Thus, we can easily visualize the apparatus entries and connect them to the text by extracting these data from the database.

[14] The reference language is Cypher, Neo4j's graph query language: https://neo4j.com/developer/cypher/

[15] As described in Section 5, Neo4j is the native graph database implemented in *Śivadharma Database*.

Finally, a system based on *commits* will be implemented to allow editors to track the earlier versions of their editions at each update.[16] Each commit will report (1) a title; (2) an agent corresponding to the actual author of the update; (3) a date; (4) an optional description of the operations performed. In this way, it is possible to trace every modification to editions and their provenance and sort them in order of time.

5. RESULTS

Śivadharma Database is a CMS to create, publish, and update Digital Scholarly Editions over time via an interface. It has been developed from scratch in Node.js[17], Express[18], Vanilla JS[19], EJS[20], SCSS[21], and Neo4j[22]. It provides a user-friendly environment that supports the production of (1) the *textus constitutus*; (2) apparatus; (3) notes to the text; (4) translation and notes to the translation; (5) parallels, notes to the parallels, translation of the parallels, notes to the translation of the parallels; (6) citations. The data creation and management system exploits the potentiality of the *form* interface component. It is possible to send data to the graph Neo4j database in the backend by filling out specific forms for each edition component. As Fig. 3 shows, the workflow to annotate the text is basic.

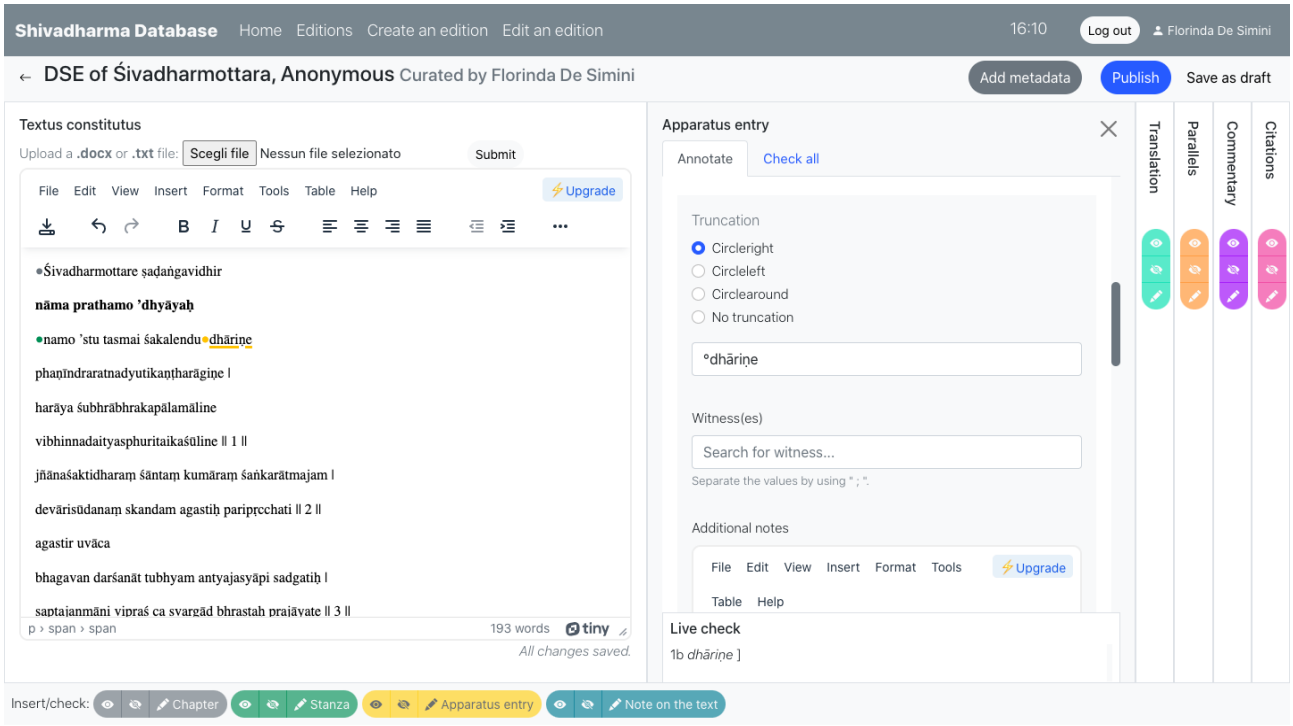

Figure 3. Forms to build an apparatus entry in *Śivadharma Database*.

The required steps are the following. First, scholars should provide the *textus constitutus* by uploading a file or typing it in the main text area. Its conversion into an HTML file and storage in a specific folder are automatic. Then, each fragment of the text can become an object of annotation. Buttons in different colors show the available types of annotations, e.g., yellow for the apparatus entries. A line in the specific color highlights the selected fragment in the text once clicked on one of them. In HTML, the selected fragment corresponds to a string surrounded by uniquely identified empty spans, which act like milestones to handle overlapping issues. Simultaneously, a module opens up on the right of the *textus*. It contains forms in which to declare specific information about the selected fragment. For instance, it is possible to create an apparatus entry by declaring the location of the selected fragment, lemma, variant readings, witnesses, additional notes, and omissions. Finally, the *Save* operation allows the submission of data to the database. Such data correspond to specific nodes

[16] The reference system is the GitHub system to track changes to the uploaded files: https://docs.github.com/en/desktop/contributing-and-collaborating-using-github-desktop/making-changes-in-a-branch/committing-and-reviewing-changes-to-your-project
[17] Node.js: https://nodejs.org/en
[18] Express: https://expressjs.com/
[19] Vanilla JS: http://vanilla-js.com/
[20] EJS: https://ejs.co/
[21] SCSS: https://sass-lang.com/
[22] Cfr. Note 9.

and relations automatically linked to their reference fragment, in turn, identified as a node in the database [Fig. 2]. The same system is applied for updating any data.
Regarding the visualization of editions, it exploits two different mechanisms. The *textus* corresponds to an HTML file. Thus, it automatically results in a visualization. The other components, instead, e.g., the apparatus, need intermediate data processing. First, they are extracted from the database by specific queries in the backend, sorted according to preset criteria, and finally displayed. Along with the visualization, the CMS guarantees the conversion of editions into XML/TEI to ensure their reuse and preservation.[23]
The *Shivadharma Database* source code and documentation on how to run it are available on GitHub[24] to guarantee CMS replicability. Since it is built in modules[25], the CMS customization according to the project specifics is easy. The help of a web developer is recommended to accomplish this task.

6. CONCLUSION

The technological choices for text encoding are pivotal since they influence the expressiveness of annotations and the effort required to learn and apply these technologies. The literature suggests two trends in Digital Scholarly Editing. On the one hand, the encoding process in the traditional XML/TEI is straightforward from the purely practical point of view of writing code. However, choosing the descriptors and handling complex textual structures are demanding tasks. On the other hand, the less consolidated RDF proposes a data structure capable of handling innate textual complexity but implies extended practical learning. In addition, the visualization does not fall within its objectives.
The present research participates in the debate proposing a solution based on HTML and graph data structure. This solution shows both methodological and practical advantages. First, HTML encoding immediately satisfies the essential requirement of Digital Scholarly Editing, i.e., the visualization of editions. In addition, it lends itself to multiple conversions into other languages, e.g., XML/TEI, for reuse and preservation purposes. On the other hand, the graph data structure guarantees a good level of expressiveness of the annotations, where the textual fragment is the pivot.
Śivadharma Database implements this solution by providing scholars with user-friendly tools to create, publish, and update Scholarly Digital Editions from scratch via an annotation interface. Providing an interface reduces the scholars' required effort on purely technological issues and guarantees complete control over the annotations, even when updating the text.
Future work includes the release of additional functionalities, e.g., the automatic reconstruction of witnesses and tools for data visualization and edition filtering, and the extension of its documentation to meet reusability needs.
Since the tools available are not strictly related to the Sanskrit field, this application is valuable regardless of the application domain[26]. Even the replicability of the CSM itself is ensured by the online availability of its open-source code and instructions for its running.

REFERENCES

[1] Daquino, Marilena, Francesca Giovannetti, and Francesca Tomasi. «Linked Data per le edizioni scientifiche digitali. Il workflow di pubblicazione dell'edizione semantica del quaderno di appunti di Paolo Bufalini». Umanistica Digitale, no. 7 (18 December 2019). https://doi.org/10.6092/issn.2532-8816/9091

[2] De Simini, Florinda. «Śivadharma Manuscripts from Nepal and the Making of a Śaiva Corpus». In One-Volume Libraries: Composite and Multiple-Text Manuscripts, 233–86. De Gruyter, 2016. https://doi.org/10.1515/9783110496956-009

[3] Di Donato, Francesca, and Susanne Müller. «Burckhardtsource.Org: The Unpublished Correspondence to Jacob Burckhardt». Lexicon Philosophicum: International Journal for the History of Texts and Ideas, no. 2 (14 March 2014). https://doi.org/10.19283/lph-20142.409.

[4] Griffiths, Arlo, and Axelle Janiak. «Encoding Guide for Critical Editions. Draft Version 0.1». 2020-2023.

[5] Hagel, Stefan. «The Classical Text Editor. An attempt to provide for both printed and digital editions». Digital philology and medieval texts (2007): 77–84.

[6] Hahn, Michael. «A brief introduction into the Indian metrical system (for the use of students)». https://www.academia.edu/6353023/Michael_Hahn_A_brief_introduction_into_the_Indian_metrical_system_for_the_use_of_students_.

[23] The development of the conversion into TEI functionality is ongoing. The reference guidelines are developed within the DHARMA Project [4]: https://dharma.hypotheses.org/a-propos
[24] *Śivadharma Database* GitHub repository: https://github.com/martinadellobuono/shivadharma-database
[25] Specifically, a module for each kind of Scholarly Digital Edition component, e.g., one for covering the apparatus, one for translations, etc.
[26] The flexibility and reusability of *Shivadharma Database* CMS will soon be tested for creating the FICLIT Digital Library collection of Italian, Latin, and Greek Scholarly Digital Editions: https://dl.ficlit.unibo.it/s/lib/page/home

[7] Mancinelli, Tiziana and Elena Pierazzo. Che cos'è un'edizione scientifica digitale. Roma: Carocci editore, 2020.

[8] Moro, Aldo.« Edizione Nazionale delle Opere di Aldo Moro». Voll. Università di Bologna, 2021. https://doi.org/10.6092/unibo/aldomoro

[9] Neill, Iian, and Desmond Schmidt. «SPEEDy. A Practical Editor for Texts Annotated with Standoff Properties». Graph Data-Models and Semantic Web Technologies in Scholarly Digital Editing 15 (2021): 45.

[10] Peroni, Silvio, and Fabio Vitali. «Annotations with EARMARK for Arbitrary, Overlapping and out-of Order Markup». In Proceedings of the 9th ACM Symposium on Document Engineering, 171–80. DocEng '09. New York, NY, USA: Association for Computing Machinery, 2009. https://doi.org/10.1145/1600193.1600232

[11] Pierazzo, Elena. «What Future for Digital Scholarly Editions? From Haute Couture to Prêt-à-Porter». International Journal of Digital Humanities 1, no. 2 (1 July 2019): 209–20. https://doi.org/10.1007/s42803-019-00019-3

[12] Renear, Allen, David Dubin, and C. M. Sperberg-McQueen. «Towards a Semantics for XML Markup». In Proceedings of the 2002 ACM Symposium on Document Engineering, 119–26. DocEng '02. New York, NY, USA: Association for Computing Machinery, 2002. https://doi.org/10.1145/585058.585081

[13] Robinson, Peter. «Current Issues in Making Digital Editions of Medieval Texts—or, Do Electronic Scholarly Editions Have a Future?». Digital Medievalist 1, no. 0 (20 April 2005). https://doi.org/10.16995/dm.8

[14] Schmidt, Desmond. «The Current State of the Digital Scholarly Edition and Three Challenges». Accessed 21 February 2023. https://www.academia.edu/37585331/The_Current_State_of_the_Digital_Scholarly_Edition_and_Three_Challenges.

[15] Schmidt, Desmond. «The Inadequacy of Embedded Markup for Cultural Heritage Texts». Digital Scholarship in the Humanities 25, no. 3 (2010): 337–56.

[16] Schmidt, Desmond. «Towards an Interoperable Digital Scholarly Edition». Journal of the Text Encoding Initiative, no. Issue 7 (12 November 2014). https://doi.org/10.4000/jtei.979

[17] Sperberg-McQueen, C. M., and Claus Huitfeldt. «GODDAG: A Data Structure for Overlapping Hierarchies». In Digital Documents: Systems and Principles, edited by Peter King and Ethan V. Munson, 139–60. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2004. https://doi.org/10.1007/978-3-540-39916-2_12