Learning Object Repositories with Dynamically Reconfigurable Metadata Schemata

Joaquín Gayoso-Cabada, Daniel Rodríguez-Cerezo, José-Luis Sierra
Fac. Informática
Universidad Complutense de Madrid
Spain
{jgayoso,drcerezo,jlsierra}@fdi.ucm.es

***Abstract*—In this paper we describe a model of learning object repository in which users have full control on the metadata schemata. Thus, they can define new schemata and they can reconfigure existing ones in a collaborative fashion. As consequence, the repository must react to changes in schemata in a dynamic and responsive way. Since schemata enable operations like navigation and search, dynamic reconfigurability requires clever indexing strategies, resilient to changes in these schemata. For this purpose, we have used conventional inverted indexing approaches and we have also devised a hierarchical clustering-based indexing model. By using *Clavy*, a system for managing learning object repositories in the field of the Humanities, we provide some experimental results that show how the hierarchical clustering-based model can outperform the more conventional inverted indexes-based solutions.**

***Keywords*—learning object repository, metadata schemata, dynamic reconfigurability, learning object indexing, browsing**

I. INTRODUCTION

The dominant trend in the production of Learning Object (LO) repositories [15] follows a *top-down* approach, based on the heavy use of standards and recommendations (e.g., metadata standards like LOM [10], packaging proposals like IMS CP [21], SCORM [5] or IMS Common Cartridge [6], and interoperability proposals like IMS DRI[1] or OAI-PMH[2]).These standardization efforts make it possible, for instance, the federation and interoperability of LO repositories in distributed networks (being AGREGA [17] a well-known example in the context of Spain).

However, the top-down approach is not particularly oriented to facilitate the inductive creation of domain-specific metadata schemata (i.e., the schemata that norm how LOs are described). It is a critical aspect in learning settings like the Humanities, in which metadata schemata must be frequently created, revised and modified in parallel to the creation of the repositories [20].

In order to facilitate the inductive construction and refinement of metadata schemata, in this paper we describe how to support a more *bottom-up* approach, according to which communities of users (e.g., instructors, researchers and students) collaborate in the construction of these schemata in addition to use these to describe learning materials. This collaboration supposes not only to define new schemata and/or use existing ones, but also to reconfigure these schemata. As consequence, the repository must react to the changes in schemata accordingly. In addition, since typically schemata are reconfigured with experimental and/or exploratory purposes in mind, it is necessary to ensure that users don't need to wait for long periods until the schemata reconfigurations are reflected in the repository; on the contrary, ideally they should be able to realize the reconfiguration's effects immediately after changing the schemata. From a system architecture perspective, this is a particularly demanding requirement, since reconfigurations in schemata can affect to the way in which the repository is browsed and / or searched. Thus, in this paper we introduce indexing strategies able to face with these strong requirements posed by dynamic reconfigurability.

The rest of the paper is organized as follows. Section II introduces our model of repository with dynamically reconfigurable metadata schemata. Section III analyzes dynamic reconfigurability in these repositories. Section IV proposes some indexing approaches to enable dynamic reconfigurability and provides some comparative results. Section V analyzes some related works. Finally, section VI outlines the final conclusions and some lines of future work.

II. THE REPOSITORY MODEL

This section introduces our model of repository with dynamically reconfigurable metadata schemata. Subsection II.A describes the repository's structure, and subsections II.B, II.C, II.D and II.E their different parts (resources, metadata schemata, LOs, and navigation maps).

A. Structure of the repository

According to our model, repositories comprise the following parts:

- A set of *resources*. These resources are the atomic digital assets that integrate the LOs.
- A set of *metadata schemata*. These schemata characterize how to describe the types of objects that can integrate the repository.
- A set of LOs. These LOs aggregate resources and simpler LOs in educationally-meaningful clusters.
- A *navigation map*. This map makes it possible to navigate the repository using the structures imposed on LOs by metadata schemata.

Fig. 1 sketches an example of repository structured according to our model (it is a repository concerning artistic

[1] www.imsglobal.org/digitalrepositories

[2] www.openarchives.org/OAI/openarchivesprotocol.html

objects from the Prehistoric and Protohistoric artistic periods in Spain).

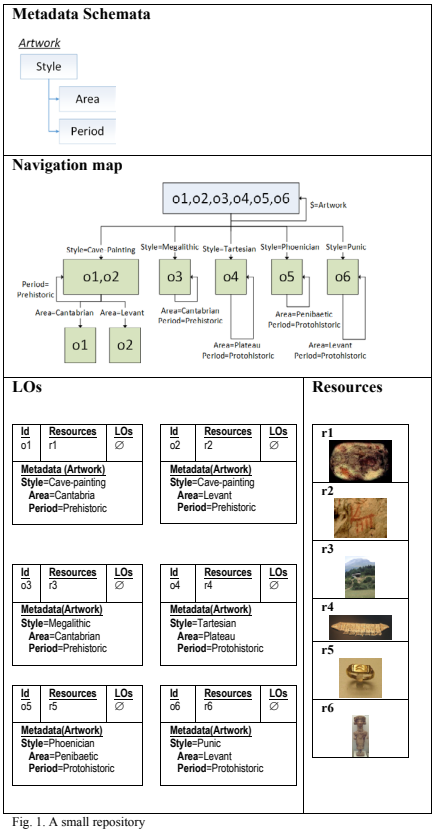


Fig. 1. A small repository

B. Resources

Resources in our model can be any digital entity with educational value. Therefore, resources can be archives of different types (images, sound or video archives, electronic documents, e-books, etc.), external resources identified by a URL, or even entities of more abstract nature (tuples of a table in a relational database, records in a bibliographical catalog, elements in an XML document, rows in a spreadsheet, etc). Each resource has associated a unique identifier, which is useful to refer the resource from LOs.

For instance, the repository of Fig. 1 includes six image archives as resources, corresponding to photographs of different artistic objects (Fig. 1 actually shows thumbnails of these images).

C. Metadata Schemata

Metadata schemata are a cornerstone aspect of the repositories. In our proposal, users can freely create new schemata and editing existing ones[3]. In this way, it is necessary to adopt a schemata model general and agnostic-enough to accommodate a great variety of users' expressive needs. For this purpose, our model is inspired by generalized markup languages (e.g., SGML or XML) [2]. In this way, each schema, in addition to have a unique name, is a hierarchical arrangement of *elements*. Each element is characterized by a descriptive name, and it can be of one of the following two types:

– *Description element*. These elements introduce descriptive values.
– *Structural element*. These elements do not introduce values, but they are useful to create intermediate structures.

Thus, by providing suitable hierarchies of structural and description elements, it is possible to mimic the description capabilities of common metadata schemata (e.g., LOM).

For instance, the repository of Fig. 1 includes one single schema, named *artwork*, oriented to provide a simplified description of an artistic object in terms of its artistic *style*, and, within this cultural style, in terms of the geographical *area* and the cultural *period*.

D. Learning Objects

Concerning LOs, they comprise the following parts:

– A (possible empty) set of references to resources (references are made by id).
– A (possible empty) set of references to other LOs.
– A *metadata document*. It is a tree-like structure conforming one metadata schema. For this purpose, suitable values are assigned to the description elements (this assignment does not need to be complete: by default, values will be initialized to $\perp$).

The repository of Fig. 1 includes one LO for each resource included in the repository (notice, however, that this one-to-one correspondence between resources and LOs cannot be necessarily extrapolated to other repositories). For each LO there is a metadata document indicating the artistic style, geographical area and cultural period associated to the LO.

[3] In concrete implementations it is possible to restrict editions to privileged users (e.g., instructors), as well as to introduce a more complex permission system.

E. Navigation map

Finally, the navigation map is a directed graph in which:

- Nodes represent sets of LOs, and arcs are labelled with *element-value* pairs used to narrow down the LOs: an arc's target node will contain only those LOs exhibiting the *element – value* pair in the source node.
- The structure of the map is constrained by the schemata hierarchies. In this way, nodes only can be narrowed down with *element – value* pairs comprising child elements of elements present in incoming arcs.
- There is also a root node, which represents the overall set of LOs. It can be narrowed down by a special element $, whose values are the different schemata names, and whose child elements are the schemata root elements.

Fig. 1 also shows a navigation map for the repository. Notice how each path in this map is constrained by the schemata structure (in this way browsing starts by selecting a value for the artistic style, and then continues by selecting a value either for the geographical area or for the artistic period).

III. RECONFIGURABILITY

In this section we address the concern of dynamically reconfiguring the metadata schemata of a repository. Subsection III.A analyzes how this reconfiguration is carried out and the effects in the different parts of the repository. Subsection III.B describes how avoid such effects in LOs representation. Subsection III.C describes, in its turn, how to deal with navigation.

A. Reconfigurable Metadata Schemata

Our model lets users reconfigure metadata schemata by rearranging the hierarchical organization of elements. For instance, Fig. 2a shows an example concerning the reposiory in Fig. 1, which primes the artistic period as primary classification focus instead the cultural style (as in the example of Fig. 1).

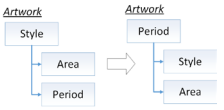
Since the organization of a repository ultimately relies on its schemata, by reconfiguring these schemata the overall repository's structure is also reconfigured. More precisely:

- The metadata documents of each LO must be changed to reflect the new hierarchical organization of elements. As an example, this effect is made apparent in Fig. 2b.
- The navigation map is also deeply affected by the reconfiguration. For instance, Fig. 2c shows how, after reconfiguring the schema of the repository of Fig. 1, the navigation map is also altered to reflect the change in focus represented by the reconfiguration (entering by *period* and refining by *style* or by *area* instead of entering by *style* and refining by *period* or by *area*).

B. Reconfigurable metadata documents

In order to address the effect of schemata' reconfigurations in metadata documents, it is needed to find document representations resilient to reorganizations of the element hierarchies. Fortunately, since all the metadata documents conforming a particular schema share a common structure (indeed, that represent by the schema), the solution in this case is easy: documents can be represented as tables assigning values to elements in the schemata instead of the whole hierarchical structure. Fig. 3a exemplifies this representation for the repository in Fig. 1. Notice that these tables remain invariant whatever the reorganizations carried out in the element hierarchies. In addition, the additional cost incurred by the representation is negligible: one indirection level. Indeed, structure recovering is a simple matter of traversing the corresponding metadata schema and of querying the table for each traversed element.

(a)



(b)

o1 **Metadata (Artwork)** **Period**=Prehistoric **Style**=Cave-painting **Area**=Cantabrian	**o2** **Metadata (Artwork)** **Period**=Prehistoric **Style**=Cave-painting **Area**=Levant	**o3** **Metadata (Artwork)** **Period**=Prehistoric **Style**=Megalithic **Area**=Cantabrian
o4 **Metadata (Artwork)** **Period**=Protohistoric **Style**=Tartesian **Area**=Plateau	**o5** **Metadata (Artwork)** **Period**=Protohistoric **Style**=Phoenician **Area**=Penibaetic	**o6** **Metadata (Artwork)** **Period**=Protohistoric **Style**=Punic **Area**=Levant

(c)

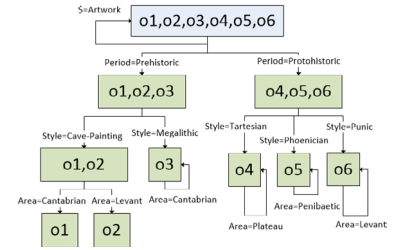


Fig. 2. (a) A reconfiguration of the schema of the repository in Fig. 1; (b) Effect of the reconfiguration in the metadata documents; (c) Effect in the navigation map

C. Reconfigurable navigation maps

The reconfiguration of the navigation map is a substantially more convolved matter. Indeed, as Fig. 2 makes apparent, a simple reconfiguration in a metadata schemata can involve a complete reconfiguration of the underlying navigation map. Therefore, it is needed to look for alternatives to the explicit

representation of such a map.
Subsection III.B describes how avoid such effects in LOs representation. Subsection III.C describes, in its turn, how to deal with navigation.

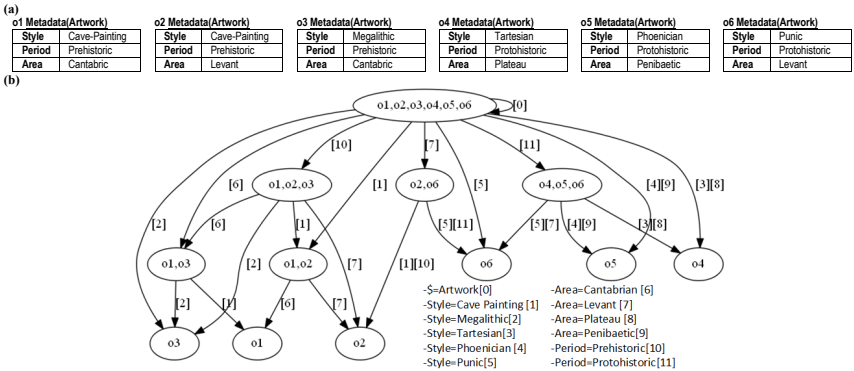


Fig. 3. (a) Tabular representations of the metadata documents in the repository of Fig. 1; (b) navigation automaton for the repository of Fig. 1

Ideally, it would be convenient to provide a structure able to represent *all* the possible navigations induced by *all* the possible reconfigurations of the schemata in a compact and unified way. For this purpose, it is needed to free *element-value* pairs from the hierarchical organizations induced by these schemata. Therefore, a plain set of *element-value* pairs must be considered and, in each interaction state of the navigation process, the applicability of all the meaningful selections must be envisioned. The result can be represented as a finite state machine, which we will call a *navigation automaton*. This automaton will consist of *states* labelled by sets of LOs, and *transitions* labelled by element-value pairs.

More precisely:

– There will be an initial state labelled by all the LOs in the repository.
– Given a state *S* labelled by a set of LOs ***O***, for each element-value pair *e*=*v* in the metadata document of some LO in ***O*** there will be a state *S*' labelled by *all* the LOs in ***O*** with *e*=*v* in their metadata documents, as well as a transition from *S* to *S*' labelled by *e*=*v*.[1]

Fig. 3b shows the navigation automaton for the repository in Fig. 1. Notice that the navigation automaton does not depend on the hierarchical organization of elements in the schemata, but only on the element-value pairs in the metadata documents. Therefore, it is not affected by reconfigurations in the schemata.

Unfortunately, and although the explicit availability of the navigation automaton provides an efficient and elegant solution to navigation in the presence of reconfigurable schemata, in some cases the number of states in this automaton can grow very fast (in the worst case, exponentially with respect to the repository's size). This fact can be realized by identifying states in navigation automata with *formal concepts* in *concept lattices* (such as these are understood in *formal concept analysis* [18])[2]. The most extreme case, in which the number of states is 2^n-1 (with *n* the number of LOs), arises, for instance, by distinguishing each pair of metadata documents in a single *element-value* pair[3].

This worst-case exponential grow ratio conforms a theoretical barrier that can hinder the explicit representation of the navigation automaton, especially in live and open scenarios as those faced by a general-purpose LO repository. Therefore, it can be recommendable to look for alternative indexing approaches.

IV. INDEXING APPROACHES

This section introduces two indexing approach to enable the dynamic recreation of navigation automata: *inverted indexes* (subsection IV.A) and *navigation dendrograms* (subsection

[1] Notice that *S* and *S*' can be the same -when all the LOs in ***O*** have *e*=*v* in their metadata documents.
[2] Indeed, navigation automata can be actually though as an explicit representations of concept lattices. As indicated in [12], the problem of determining the size of concept lattices is proved to be a #P-complete one (i.e., harder than NP-complete). Thus, the exponential factor underlying the intrinsic complexity of the problem can hinder the direct applicability of the technique on repositories of moderate or large sizes.
[3] This construction is actually suggested by the proof of theorem 1 in [12]

IV.B). Subsection IV.C provides some experimental results comparing both approaches.

A. Inverted indexes

Inverted indexes are standard artifacts used for information retrieval [24]. Basically, for each element-value pair, an inverted index associates the set of LOs including such a pair in its metadata document. Fig. 4a shows an example of inverted index for the repository in Fig. 1.

Notice that this kind of inverted index can be used to determine the set of selected objects in each navigation path by intersecting the sets associated with the element-value pairs traversed. The cost of evaluating the cited intersection operations constitutes the main shortcoming of the approach. While there has been extensive research in performing these intersection operations efficiently [3], the cost is not negligible. On the positive side is the availability of many mature implementations and frameworks that can be used in a straightforward way to support the technique. For instance, in our experiences, we used Lucene [14] for such a purpose.

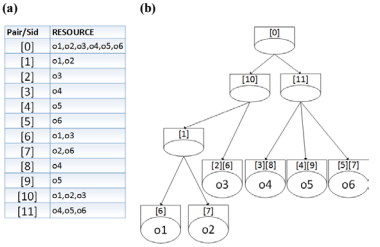


Fig. 4. (a) An inverted index for the repository of Fig. 1; (b) A navigation dendrogram (References `[0]`,`[1]`,etc. are defined in Fig. 3b)

B. Navigation dendrograms

In order to avoid the proliferation of intersection operations, which is characteristic of inverted indexes representations, we have designed a tree-shaped indexing scheme inspired by *dendrograms* in hierarchical clustering [11]. The resulting structures are called *navigation dendrograms*.

Nodes in a navigation dendrogram represent subsets of the overall LO set. The LO set associated to a node is not explicitly stored in this node. Instead, each LO is only hosted in one node (the LO's *host node*). LOs placed in a node are called the mentioned node's *own* LOs. The overall LO set of a node is given by its own LOs and by all the own LOs of its descendants. Finally, in order to partition the LO space, each node has a set of *filtering* element-value pairs associated, so that all the own LOs in the node and in all their descendants' must include these filtering pairs in their metadata documents.

Navigation dendrograms can be built to contain as most $2K$ nodes (K being the number of LOs in the repository). In addition, navigation can be articulated by maintaining a set of dendrogram's nodes. Then, when an element-value pair is selected, this set is refined as follows:

- Nodes containing the selected pair in their filtering sets or having an ancestor accomplishing such a condition are preserved.
- Nodes having any descendant containing the selected pair in its filtering set are replaced by all the descendant accomplishing such a condition.
- Any other node is discarded.

By maintaining all the required information to carry out this refinement in the nodes (i.e., filtering pairs of node's ancestors, and references to descendants per filtering pairs) this process can be carried out very efficiently. Indeed, the resulting structure is a non-deterministic version of the navigation automaton that explicitly avoids the aforementioned potential exponential factor.

Fig. 4b shows an example of navigation dendrogram for the repository in Fig. 1.

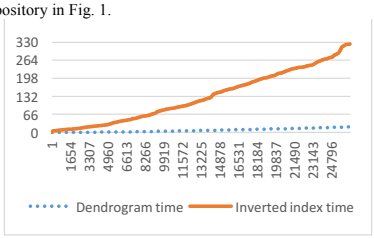


Fig. 5. Accumulated time of inverted indexes vs. dendrograms

C. Experimental evaluation

In order to compare the two approaches described, we implemented both on *Clavy*, an experimental system for managing LO repositories with reconfigurable metadata schemata[4].

We also set up an experiment consisting of adding the LOs in *Chasqui* [20], a repository of 6283 LOs on Precolombian American archeology, to *Clavy* and to simulate runs concerning navigation and schemata reconfiguration operations. Each run interleaved 100 LO insertion with $0.1n$ navigation operations randomly interleaved with $0.01n$ reconfigurations (n being the number of LOs inserted so far). Each navigation operation consisted, in turn, of selecting a feasible element-value pair, computing the next interaction state, and visiting all the LOs filtered. Reconfiguration operations, then, consisted of feasible interchanges of two randomly selected elements,[5] followed by a navigation step. Inverted indexes were managed using Lucene, while navigation dendrograms were managed using our

[4] http://clavy.fdi.ucm.es

[5] By *feasible* we mean avoiding cycles in the resulting schema.

own implementation. In both cases, in-memory indexes were used to avoid side effects of persistence, disturbing the experiment.

Fig. 5 shows the results obtained from the two runs (experiment run on a PC with Windows 10, with a 3.4GHz Intel microprocessor, and with 8Gb of DDR3 RAM). The vertical axis corresponds to the number of operations carried out so far. The horizontal axis corresponds to accumulated time (in seconds). As is made apparent, the dendrogram-based approach clearly outperforms the inverted indexes (even though we are using a highly optimized framework, like Lucene, for inverted indexing vs. our own in-house experimental implementation for dendrograms).

V. RELATED WORK

Our proposal is similar to browsing systems for browsing information spaces that, like ours, envision the possibility that the user reconfigures the underlying metadata schemata (e.g., [8][19]). However, these systems are typically supported by general-purpose semantic web or relational database solutions instead of by model-specific indexing approaches.

A seminal work on using concept lattices to organize and navigate information spaces is [4]. Some recent systems using concept lattices as their underlying indexing structure are [7][22]. However, all these approaches face the theoretical limit imposed by the intrinsic complexity of formal concept analysis. It is why we proposed a simpler but still practical approximation based on navigation dendrograms.

Inverted indexes have been extensively used to support hierarchical navigation (e.g., guided by faceted thesauri). Works like [23] describe efficient approaches to enable this navigation. However, all these approaches are based on the assumption of pre-established and immutable schemata. As noticed in [1], if this assumption is left out, inverted indexes can become costly due to the set operations involved.

Finally, it is worthwhile to notice that clustering techniques has been extensively used in open metadata schemata (i.e., folksonomy-like systems) to enable the discovering of useful semantic relationships among terms in order to provide better guidance to users (e.g., [9][13][16]). Thus, clustering in these approaches is oriented to enhance users' navigation efficiency, while our navigation dendrograms are oriented to enhance the internal efficiency of the supporting software.

VI. CONCLUSSIONS AND FUTURE WORK

In this paper we have addressed the problem of dynamic reconfigurability in LO repositories. Since metadata schemata can be rearranged in unexpected ways, it is necessary to use internal representation mechanisms resilient to these changes. In the case of metadata documents we have shown how a tabular representation of the assignment of values to elements in the schemata suffices. However, dealing with the navigation system is substantially more cumbersome. We have shown how a concept lattice-like representation (which we have called a *navigation automaton*) can elegantly address this concern. However, this representation exhibits a potential exponential factor that, at least in theory, hinders its applicability (especially in live and open settings, in which schemata evolution cannot be envisioned to priori). For this purpose, we have proposed alternative indexing approaches (one based on inverted indexes, and another one based on dendrograms). We have also provided some evidence of how dendrograms can outperform inverted indexes.

We are currently working on optimizing and persisting our representations. In addition, we want to further study the practical grow ratio of the navigation automaton in real-world scenarios, to support arbitrary Boolean queries, and to run more empirical evaluations.

ACKNOWLEDGEMENTS

This work has been supported by the BBVA Foundation (grant HUM14_251) and Spanish Ministry of Economy and Competitiveness (grant TIN2014-52010-R)

REFERENCES

[1] Berchtold, S., Böhm, C., Keim, D-A., Kriegel, H-P., Xiaowei, X.: Optimal Multidimensional Query Processing Using Tree Striping. DaWaK'00, 244-257. 2000

[2] Coombs, J. H., Renear, A. H., DeRose, S. J. Markup Systems and the Future of Scholarly Text Processing. Communications of the ACM, 30 (11), 933-947. 1987

[3] Culpepper, J-S., Moffat, A.: Efficient Set Intersection for Inverted Indexing. ACM Transactions on Inf. Systems 29(1), article 1 (2010)

[4] Godin, R., Saunders, G. Lattice Model of Browsable Data Space. Information Sciences 40(2), 89-116. 1986

[5] González-Barbone, V., Anido-Rifón, L.E. Creating the first SCORM object. Computers & Education 51(4): 1634-1647. 2008

[6] Gonzalez-Barbone, V., Anido-Rifón, L.E. From SCORM to Common Cartridge: A step forward. Computers & Education 54(1): 88-102. 2010

[7] Greene, G-J. A Generic Framework for Concept-Based Exploration of Semi-Structured Software Engineering Data. ASE'15, 894-897. 2015

[8] Hildebrand, M., Ossenbruggen, J-v., Hardman, L.: /facet: A Browser for Heterogeneous Semantic Web Repositories. WWW'06, 272-285. 2006

[9] Huang, J-W., Chen, K-Y., Chen, Y-C., Yang, K-N., Hwang, S., Huang, W-C. A Novel Spatial Tag Cloud Using Multi-Level Clustering. Journal of Information Science and Engineering 30, 687-700. 2014

[10] IEEE Standard 1484.12.1-2002. 2002. IEEE Standard for Learning Object Metadata

[11] Jain, A-K., Murty, M-N., Flynn, P-J.: Data Clustering: a Review. ACM Computing Surveys 31(3), 264-323. 1999

[12] Kuznetsov, S. On computing the size of a lattice and related decision problems. Order 18(4), 313–321. 2001

[13] Li, R., Shenghua, B., Fei, B., Su, Z., Yu, Y. Towards Effective Browsing of Large Scale Social Annotations. WWW'07, pp. 943-952. 2007

[14] McCandless, M., Hatcher, E., Gospodnetic, O.: Lucene in Action, 2nd Edition. Manning Publications. 2010

[15] Polsani, P. Use and Abuse of Reusable Learning Objects. JODI 3(4).2003

[16] Radelaar, J., Boor, A-J., Vandic, D., van Dam, J-W., Fasinca, F. Improving search and exploration in tag spaces using automated tag clustering. Journal of Web Engineering 13(3-4), 277-301. 2014

[17] Sarasa-Cabezuelo, A., Canabal-Barreiro, J-M., Sacristán-Heras, J-C. Agrega - Spanish Education Community Federation of Repositories Of Learning Objects. eLearning 2008: 47-50

[18] Sarmah, A-K., Hazarika, S-M., Sinha, S-K.: Formal Concept Analysis: Current Trends and Directions. Art. Int. Review 44(1), 47-86. 2015

[19] Schraefel, M-C., Wilson, M., Russell, A., Smith, D-A.: MSPACE: Improving Information Access to Multimedia Domains with Multimodal Exploratory Search. Communications of the ACM 49(4), 47-49. 2006

[20] Sierra, J.L., Fernández-Valmayor, A., Guinea, M., Hernanz, G. From Research Resources to Learning Objects: Process Model and Virtualizacion Experiences. Ed. Tech. & Society 9(3), 56-68. 2006

[21] Sierra, J.L., Moreno-Ger, P., Martínez-Ortiz, I., Fernández-Manjón, B. A highly modular and extensible architecture for an integrated IMS-based authoring system: the <e-Aula> experience. Software Practice and Experience 37(4): 441-461. 2007

[22] Way, T.; Eklund, P. Social Tagging for Digital Libraries using Formal Concept Analysis. CLA'10. 2010

[23] Yitzhak, O-B., Golbandj, N., Har'El N. et al. Beyond Basic Faceted Search. WSDM'08, 33-44. 2008

[24] Zobel, J., Moffat, A.: Inverted Files for Text Search Engines. ACM Computing Surveys 33(2), article 6. 2006