Use of Contextualized Attention Metadata for Ranking and Recommending Learning Objects

Xavier Ochoa
Escuela Superior Politécnica del Litoral
Campus "Gustavo Galindo"
Guayaquil, Ecuador
+59342269773
xavier@cti.espol.edu.ec

Erik Duval
Computerwetenschappen Dept., KULeuven,
Celestijnenlaan 200 A,
B-3001 Leuven, Belgium
+3216327066
Erik.Duval@cs.kuleuven.be

ABSTRACT
The tools used to search and find Learning Objects in different systems do not provide a meaningful and scalable way to rank or recommend learning material. This work propose and detail the use of Contextual Attention Metadata, gathered from the different tools used in the lifecycle of the Learning Object, to create ranking and recommending metrics to improve the user experience. Four types of metrics are detailed: Link Analysis Ranking, Similarity Recommendation, Personalized Ranking and Contextual Recommendation. While designed for Learning Objects, it is shown that these metrics could also be applied to rank and recommend other types of reusable components like software libraries.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *information filtering, relevance feedback.*

General Terms
Algorithms, Measurement, Design, Human Factors, Standardization

Keywords
Learning Objects, Ranking, Recommending, Attention Metadata

1. INTRODUCTION
One of the main reasons to capture and analyze the information about the interaction between a user and a tool is to improve the user experience. For example, a online library system could record the subject of the books that a client has bought before in order to recommend him new books about a similar subject the next time he/she logs in, saving him/her the hassle to search for them [1]. A news web site could record the topic of the news articles that a user normally read in order to filter out news that do not interest such user [2]. A collaborative browser could use the information recollected from the browsing patterns of a given community to improve the rank of different pages on the searches of an individual user, member of that community [3]. The generic name of Attention Metadata[4] has been applied to describe the information about these interactions.

When the information stored does not only contain the reference to the user and the action that it performs, but also register when the action took place, through which tool the action was performed, what others thing was doing the user at the same time, what is the profile of the user performing the action, to what community he/she belongs, etc, it leads to an improved and more useful form of record, called Contextualized Attention Metadata [5] (CAM). AttentionXML [6] and its extensions [5] are an effort to standardize the way in which CAM is stored. This standardization will lead to the opportunity to share attention records between different applications. For example, a second generation of an Attention-Sharing online library could know which news topics the user is interested in and it could recommend him/her books related to those topics.

The authors believe that one group of applications that could greatly benefit from CAM information is the search and find of Learning Objects. These applications have suffered from an under-par performance compared to similar applications in other fields [7] [8]. The main reason for this is the lack of a meaningful and scalable way to rank or recommend the objects to the users. Currently, two main methods are used to rank (not even recommend) Learning Objects: Manual Rating or Metadata Content Rating. In the first approach, Manual Rating, each Learning Objects should be rated by a group of experts and/or the user community. For each search, the returned objects are ranked based on their average rate. While this is bound to provide meaningful ordering, it does not scale. For example MERLOT use this approach, but only 10% of the total content of the database has ever be rated [9]. The other approach, use only the information contained in the metadata record to perform ranking based on the similarity with the query terms. The most common method used for this is the TFIDF metric[10] that measure in a Vector Space the distance between the query vector and the vector composed from the text contained in the metadata record. Given than TFIDF was designed to work over full text documents and that metadata records contain very few textual descriptions [11], normally the ordering is not meaningful for the user. SILO (Search and Indexing Learning Objects) tools from the ARIADNE [12] repository use this approach. CAM could be used to generate a third approach, one in which the human attention (meaningful) is processed to construct an automated (scalable) rating and recommending procedure.

The following sections of this work describe in detail what information should be stored in the CAM record of Learning Object Applications (Section 2) and the mechanisms by which such information could be used to generate rating and recommending metrics (Section 3). It is also discussed how these mechanisms and metrics could be applied to related contexts (Section 4) and which research questions need to be addressed in further work (Section 5). The work finalize with an overview of related research (Section 6)

2. CAM FOR LEARNING OBJECTS APPLICATIONS

Users interact with a Learning Object through the object's whole lifecycle. CAM recorders capture and timestamp all those interactions in order to provide the information needed to calculate useful metrics to be used in a next generation breed of learning object management tools. According to the AttentionXML extension proposed by Najjar et al at [5], these interactions are stored inside an Action record. This work will briefly list the different Actions that should be recorded through the Learning Object lifecycle. The lifecycle phases are taken from the enumeration done by Collins and Strijker in [13]. Also, it is suggested which applications should generate the attention records.

Creation: In this phase the author creates or assembles the learning object in its digital form using some sort of authoring tool. The Creating Action should be captured and it must include the identity of the created object, its author(s), the authoring tool used and the list of component-objects [14] reused through the creation process. This record should be created by the authoring tool, for example Microsoft Power Point.

Labeling: At this stage the author, an indexer or even an automated system could add a metadata record that describes the Learning Object. The Labeling Action must include information that identify the object, the labeler, the origin of the metadata (Automated, Semi-Automated, Manual), the metadata format used, the level of confidence of the information (how sure the autor is that metadata values are correct) and a unique identifier for the metadata record. Normally this record should be also created by the authoring tool at the end of the creation of the objects, but could also be created by metadata editors as [15] or automated metadata generators as [16].

Offering: At this stage the author or indexer inserts the object in a repository or other system that allow the object to be shared with others. The Inserting Action must include the following information: Object Unique Identifier, Inserter, Tool Used and Learning Object Unique Identifier inside the sharing tool. This record should be created by the sharing tool, being it a Learning Object Repository or a Peer to Peer sharing application.

Selecting: In this stage the user search, find and select Learning Objects in the Sharing System. Several Actions should be captured during this phase. A Searching Action when a query is performed to find relevant objects. It must include information that describe the query performed and the objects returned. A Recommending Action when the system suggests relevant objects without the user performing a query. It must contain information a list of the object(s) recommended, the user action that trigger the recommendation and the tool used to perform the recommendation. A Browsing Action when the user reviews the metadata or description of an object. It must store information that identifies the metadata record browsed and the time expend in the review. Finally, A Selecting Action when the user chooses an object by downloading it or accepting the recommendation. It must contain information that identifies the selected object. All this actions should also contain information about the user that performs the action. These records should be generated by the sharing or recommending tool.

Using: This stage comprehends all the actions that the final user (instructor or learner) performs with the learning object during its normal utilization in a learning environment. There are several actions to be registered. A Publicating Action when the instructor inserts the object into a Course belonging to some kind of Learning Management System. It must contain information that identifies the published object and the context (course, lesson) where it was published. A Sequencing Action when one or more objects are included in an instructional design or sequenced package. It must contain information about the identification (in an ordered form) of the integrated objects. A Viewing Action, the object is read or viewed by learners. It must contain information about the time spent reviewing the material. An Annotating Action when the instructor or the learner add a comment or rate the learning object. It must include information about the comment or the rate given and the identifier of the object. All these action should also store information about the user that performs them. Different tools should be in charge of the generation of the attention records, a LMS for the Publishing Action, a Learning Activity Management System [17] or SCORM [18] Packager for the Sequencing Action, a Web browser or document reader for the Viewing Action and a Rating or Review system for the Annotating Action.

Lifecycle	Actions	Main Information	Source
Creation	*Creating*	author, components	Authoring tool, Components
Labeling	*Labeling*	metadata format, origin, confidence	Authoring tool or Metadata generator
Offering	*Inserting*	inserter	LOR or Sharing app.
Selecting	*Searching*	query, results	LOR's search tool
	Recommending	objects recommended	Recommender
	Browsing	Time	LOR or Recommender
	Selecting	object identifier	LOR or Recommender
Using	*Publicating*	LMS context	LMS
	Sequencing	list of sequenced objects	ID tool or Packager
	Viewing	Time, tool used	Browser or Reading app.
	Annotating	rate or review	LMS
Retaining	*Retaining*	decision to keep or delete	LMS

Table 1. Proposed CAM information to be stored for Learning Object Applications

Retaining: In this phase, the instructor check for the validity of the learning object and decides if it is still useful or if it should be replaced / updated. The Retaining Action should contain information that identifies the object and the decision taken (keep, update, delete). This attention record will normally be generated by the LMS where the object has been published.

A summary of the Actions that CAM should record is presented in Table 1. Some of these CAM Actions (Creating, Inserting, Selecting, Viewing) are already produced and stored in different tools [5]. The others are easy to implement in existing tools taking in account that most of them (LMS, Metadata Generators, etc) already produce a log with the user's interactions. In the next section, metrics to exploit this Action records to improve tools to search and find Learning Objects are proposed.

3. RANKING AND RECOMMENDING METRICS USING CAM

Several ranking and recommending metrics will be proposed. These metrics will use only two sources of information to be calculated: the first one is the Learning Object Metadata (LOM) [19] record that describe each Learning Object; the second one is the CAM Actions described in the previous section.

3.1 Link Analysis Based Ranking

One of the most famous and successful ranking algorithms at the present is PageRank [20]. PageRank use the information contained in the network of links between web pages to calculate the relative "importance" of a page. It could be summarized as: a page is important if it is linked by a high number of pages. Also, the importance increases if the pages linking to it have also a high importance rank. Unfortunately, this algorithm could not be applied directly to Learning Objects. While LOM records have a linking field, it is rarely populated [11]. Also, LOM linking reflect just a semantic relationship; it does not imply a "vote" for that object as it is assumed for Web pages.

As an alternative to the explicit linking structure that the web posses, CAM allow us to create an implicit linking between Learning Objects and other entities related to them: Authors, Users, Courses, Learners, etc. For example: Creating Actions can be converted into a link between an author and an object, Selecting Actions can be converted into a link between a user and an object, Publicating Actions can be converted into a link between a course and an object and also between a user and the same object. Viewing Actions can be converter into a link between a learner and an object. As result of this conversion of CAM to links between different entities, a K-partite graph is created (a graph with different partitions, where there are not links between nodes of the same partition). In this graph each type of entity (Learning Object, User, Course, and Learner) is considered a partition Figure 1 present diagram of an example of such a graph.

Once CAM information is represented as a graph, it is easy to use basic graph algorithms to calculate ranking metrics. Following there are some metrics that could be developed this way:

- **Popularity Rank (PR):** Using the information contained in the Selecting Action (converted already in a 2-partite graph), it is easy to obtain the number of times that an object has been downloaded. To calculate just count the number of incident links that each Learning Object node has from nodes in the User Partition. This metric is a just a basic way to put most downloaded objects first in the result list.

$$PR(object) = inDegree(object)$$

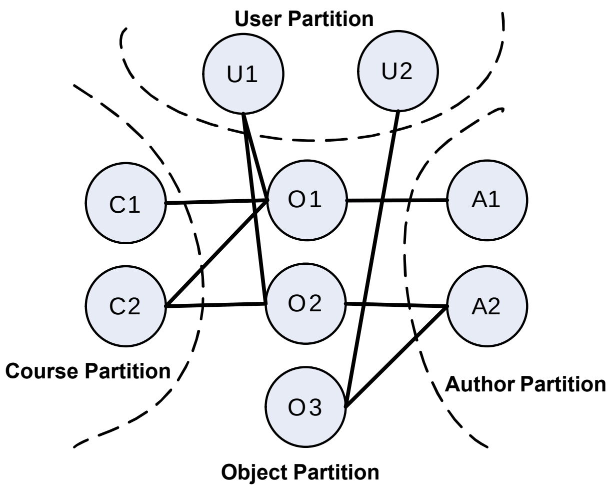

Figure 1. K-Partite Graph representation of CAM

- **Author-Corrected Popularity Rank (ACP):** Combining the Creating and Selecting Actions, it could be calculated how popular an object is based on the number of downloads and the popularity of the Author. The first step is to create a 3-partite graph with Users, Objects and Author partitions. Then the PopularityRank (PC) is calculated for all the objects. Next, the Author Popularity (AP) is calculated adding the PC of all the Learning Objects nodes that are linked to the author node. Finally, the AP is multiplied by a weighting factor and added to the also weighted PC. This metrics enable new objects (that do not have any downloads) from a well downloaded author, appear higher in the list.

$$AP(author) = \sum_i PR(object_i);\ \text{if object}_i \text{ is linked to author}$$

$$ACP(object) = \alpha \cdot PR(object) + \beta \cdot AP(author)$$

- **Weighted Popularity (WP):** Selecting, Publicating and Retaining Actions can be combined to generate a 2-partite graph between Users and Learning Objects. The links of this graph will be weighted: if the link is made using the Retaining information ($inDegree_R$) it will have more weight as if the link was made using the Publication information ($inDegree_P$). In the same way, Publication links will weight more than Selection links ($inDegrees_S$). The rationale behind this metric is that different actions mean different level of "preference" for an object. If the instructor has use the object and she is happy with it to keep it for the next semester is a stronger vote of support than just using it for the first time or that just downloading it. That difference of importance is represented in the weight given to each kind of link.

$$WP(object) = \alpha \bullet inDegree_S(object) + \\ + \beta \bullet inDegree_P(object) + \gamma \bullet inDegree_R(object)$$

$$\text{with } \gamma > \beta > \alpha$$

- **Rate of Reuse Rank (RRR):** Using the Selecting Action (or also the Publicating and Retaining Actions as in the previous metric), the number of times that an object has been downloaded during a given period of time P (last week, month, year) can be calculated. The 2-partite graph (Users and Objects partitions) can be constructed but only taking in account the Actions occurred in a given period of time. For example: if the last week is selected as P, This rank will calculate how often the object has been downloaded (inserted or retained) on the last 7 days. This value could be normalized using the age of the object, obtained from the related Creation action. This metric will help to rank higher object that have been reused frequently and are relatively new.

$$RRR(object) = \frac{inDegree(object)}{age(object)} \text{; for links inside period P}$$

- **Manual Rank (MR):** Using the information that is stored in the Annotation Action, the number of times that an object has been positively (or negatively) rated or reviewed could be considered to calculate a metric. A 2-partite graph (Users and Objects partitions) is created. The procedure will weight the link as 1 if it is a positive rate or review, -1 if it is a negative rate or review. The actual value of the rate is only used to evaluate if the rate is a positive or negative "vote", because different users and system have different scales to grade. The reviews can only be considered if their positiveness or negativeness value is included in the Annotation Action or could be automatically inferred from the text.

$$MR(object) = inDegree_{Positive}(object) - inDegree_{Negative}(object)$$

These metrics can be calculated off-line because they are not user or query specific. They calculate an average importance or relevance of the learning objects based in the agglutination of attention information. These metrics, and others that can be developed afterwards, could be integrated in a final ranking metric. This Compound Popularity Metric (CP) can be calculated as the weighted sum of the values of the individual metrics. For example, Google integrates more than 100 of different simple metrics in order to provide its results [21].

$$CP = \alpha PR + \beta ACP + \gamma WP + \delta RRR + \varepsilon MR$$

The weighted coefficients (α, β, etc) should be estimated (not trivial procedure) to provide an optimal result ordering. Methods to make these estimates are described in [22] and [23]. Also, manual rates should be used carefully because the Annotation Information is optional and could not exist for all the objects involved in the calculation.

3.2 Similarity Metrics for Recommendation

One property of a 2-partite graph is that it can be folded over one of its partitions, generating a normal graph with just one entity and links between its nodes. For example if we have a 2-partite graph of Users that have download Learning Objects, we can fold over the Learning Object partition and we will end up with a graph where the Users are linked between them. Each link mean that those two users have download the same object at least once. This new graph could be used to calculate similarity between the users based on the download patterns. In Figure 2 we can see a representation of the folding result. The first part of the figure represents a 2-partite graph with the User and Objects partitions. The graph shows that, for example, that User 5 had downloaded Object 2 and Object 3 and User 1 had only downloaded Object 1. The second part of the figure illustrates the folded version of the graph. In this new graph, the users have a link between them if they linked to the same object in the unfolded graph. The more objects the users have in common, the thicker the line. For example User 1 and User 4 are linked because they both have downloaded Object 1. User 2 and User 5 have a stronger links because they both have downloaded Object 2 and Object 3. This technique is similar to the one applied in scientometrics to obtain relations between different authors, based on the papers that have co-authored[24].

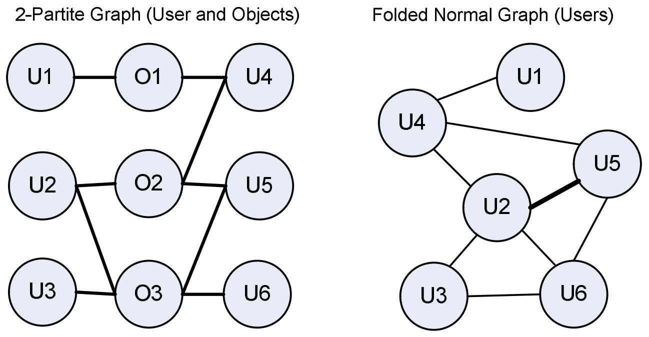
2-Partite Graph (User and Objects)
Folded Normal Graph (Users)
U1
O1
U4
U2
O2
U5
U3
O3
U6
U1
U4
U5
U2
U3
U6

Figure 2. Unfolded and Folded 2-Partite Graph

We present several similarity metrics that can be calculated using the information contained in CAM Actions detailed in the previous section.

- **Object Similarity based on Number of Downloads:** Create a 2-partite graph with the information of the Select Actions (when a User download a Learning Object), and fold over the User Partition. A link between two Objects in the final graph means that those objects have been downloaded by the same user. The strength of the similarity is number of users that have downloaded both objects.
- **Object Similarity based on Re-Use:** Create a 2-partite graph with the information from the Publish Actions (when a Learning Object is inserted into a Course), and fold over the Course Partition. A link between two Objects in the final graph means that two objects have been inserted in the same course. The strength of the similarity is number of courses that include both objects.
- **Users similarity based on Downloads:** Create a 2-partite graph with the information from the Select Actions, and fold over the Object Partition. A link between two Users means that they have downloaded the same object. The strength of the similarity is the number objects that the users have in common.
- **Author similarity based on Re**-Use of Components: The Creation Action information could be use to identify re-use of learning object components. For example several authors could use the same picture or diagram inside they presentations. As the Creation Action store information about which existing components have been reused (see Section 2), a 2-partite graph between Authors and Components can be created and then folded over the

Components partition. The new graph will represent relationship between different authors. More components those authors have used in common, the stronger their similarity.

The similarity metric obtained from the graph could be then applied in recommendation tools. For example: If a user finds an object useful, a link to similar objects could be provided (similar to what Amazon does with books [1]). Also, the similarity between users can be exploited to recommend Learning Objects to a user, based on what other users that are in the same community have recently download (similar to collaborative browsing applications [3]). To automatically extract the communities from the graph, an algorithm like EdgeBetweeness [25] can be applied. The same procedure could be applied to the Author Similarity graph. The communities of authorship can be automatically extracted from the graph. The author then can be recommended with components that have been created by other authors in the same authorship community.

Beside recommendation systems for Learning Objects, these similarity metrics could be considered as distance metrics. The distance metric can be used inside clustering algorithms to automatically find groups of similar objects. These clusters could be used to improve the presentation of results of a search, much as Vivisimo [26] does for Web Pages.

3.3 Personalized Ranking

To be able to personalize the search result order for a given user, the application should have a representation of such user in a profile. While this profile could be created explicitly by the user, CAM information could help the application to learn it form the user interaction with the tool. For example, the information about stored in the Select, Publish and Retain from a user could help us to determine in which objects is he/she interested, and rank higher objects that are similar to those.

This work proposes the creation of a fuzzy profile that could easily account for the evolving and not fixed behavior of an instructor downloading learning objects. Instead of having a crisp preference for one type of object, this profile will provide different grades of likeness for several characteristics of the learning object. This profile is constructed with several Fields. The Fields could be a subset of the fields considered in the LOM standard, especially the ones that use a vocabulary or represent a classification. Each field will contain 2 or more fuzzy sets that represent the values that the field could have (from the vocabulary or the classification values). A user could "prefer" in different degrees 1 or more of the values of a Field. The preference of the user for each one of the values is calculated based on the number of objects that the user have download before that contained that value in the corresponding LOM field. This fuzzy profile has been derived from research done to produce automatic TV recordings for PVRs [27] like TIVO.

The fuzzy profile could be easily operationalized to provide a personalized rank for Learning Objects. First, each field should have a weighting value (that express how important that field is). That value could be assigned by an expert or could be calculated automatically for entropy of the distribution of the field values for that user. For example if a user downloads objects from a wide variety of topics, the weight of topic as a good ranking measurement is low. Instead, if the user only downloads objects in one language, the weight of that field should be high. Second, each LOM record from the result list is converted to a similar representation, using the same fields and a preference value of 1.0 for the value found in the metadata. Finally, the object representation is operated with the profile in order to obtain how well the object fits the preference of the user. This operation is described in the following equation:

$$PersonalRank(object, user) = \sum_i \alpha_i \bullet \sum_j Field_i(user).value_j \bullet Field_i(object).value_j$$

For example, lets consider a user that have download 20 objects, 16 with topic Computer Sciences and 4 with topic Physics. Of those 20, also, 12 are in English, 4 are in French, 4 are in Spanish. A fuzzy profile that represents that user could be expressed as:

U1 = {(0.8/ComputerScience + 0.2/Physics),
(0.6/English + 0.2/Spanish + 0.2/French)}

The fields weighting terms are 0.9 for Topic and 0.6 for Language. Lets now considered 2 objects represented also as fuzzy sets:

O1 = {(1.0/ComputerScience), (1.0/Spanish)}

O2 = {(1.0/Physics, 1.0/English)}

The calculated rank for both objects is:

O1 = 0.9*0.8 + 0.6*0.2 = 0.84

O2 = 0.9*0.2 + 0.6*0.6 = 0.54

O1 will be ranked higher than O2 as it is more similar to the user profile.

The personalized calculation could be combined with the popularity ranking described before to create a better ranking algorithm, the same way as Google personalized Search mix the standard Popularity measure with information from the user profile to order the results.

3.4 Contextual Recommending

If the CAM is considered not only as a source for historic data, but also as a continuous stream of contextualized attention information, we can use very recent CAM (in the order of seconds or minutes) to generate recommendations based on what the user is focusing his/her attention at the moment. For example, the recommender system could use the information stored in the if the user has inserted an object inside a Course in a Learning Management System (LMS), the LMS will generate a CAM record with contextual information about which object was inserted and in which lesson of the course. The recommending system could use that information to present the user with similar objects to the one inserted or others that have been used in similar courses, based on the topic of the course or in similarity metrics as the one explained in the Section 3.2.

The recommending system could also present objects that suit the application that the user is using at a given moment, based on the information about the object (LOM record). For example, if the user is working Microsoft Power Point authoring tool, presentations, slides, small texts, images and diagrams will be recommended. If he/she is working with a SCORM Packager, complete learning objects will be presented instead.

Contextual recommending techniques have been tried before in several fields [28] [29]. Blinkx [30] is an example of this kind of applications. It recommends web pages, videos and news based on the present content of the screen of the user. A similar application could be developed in a LMS for example, where the system could recommend to the instructor materials to add to each lesson, or could recommend the learner with similar or complementary materials to the one that the instructor has added to the course.

4. APPLICATION IN OTHER CONTEXTS

While the CAM based metrics proposed in this work were designed for Learning Objects, they could be easily extended or adapted to work for other kind of reusable components where CAM could be collected. For example, given the exponential grow of open source software libraries that could be reused inside software projects, programmers are sometimes overwhelmed with the amount of available choices. It makes sense to develop some kind of ranking or recommending system that could help the developers to select the right tools.

To construct the ranking application we can use the same methods proposed for learning objects. The k-partite graph used to calculate the popularity metric could be constructed using the metadata information about the library (who is the author of a software library) and contextual attention information about how and when the programmers interact with the library (which programmers have download it, in which software project they have been used). Most of this information could be obtained from open source project repositories like SourceFourge [31]. The rationale behind the ranking would be: A library that have been downloaded more often / at a higher rate is more useful. A library produced by authors with highly useful libraries could also be useful. A library re-used in many projects is probable highly useful. This metrics are parallel to the ones described for learning objects.

Recommending systems for software libraries could also be constructed in a similar way to the ones proposed for learning objects. For example, we can fold the Libraries-Programmers 2-partite graph over the Libraries Partition, creating a graph that relate Programmers between them based on the Libraries that they have downloaded/used. Communities could be extracted form the resulting graph and could be used for recommending a programmer with new libraries that other members of his/her community have used in their projects.

The precaution to have when applying this metrics to other domains is the semantics of the relations that are created with the graphs. For example, if two learning objects are used in the same course, those two learning objects must have something in common (same topic for example), while if two libraries are used inside the same project, that does not mean that the libraries are related (you could use a database access library and graphical interface library inside the same project).

Other contexts where CAM information could be exploited to rank and recommend elements with a similar strategy as the one presented in this work are music mixes (component songs or loops) and news aggregators.

5. FURTHER WORK

This work is just an introduction to how CAM information could be used to rank and recommend Learning Objects. Several topics should be solved before a big scale application that use the metrics presented could be built:

- **Collection and Integration of the different CAM sources:** While today exist several applications that generate CAM, there is not an established multi-application CAM repository that could be used to collect and integrate attention information.
- **Combination of different ranking strategies:** When different ranking strategies are combined, some weighting coefficient must be applied. The calculation of those coefficients is not trivial and should be made using extensive user feedback.
- **Critical mass vs. Closed Community:** To be useful, the metrics should be calculated over a significant amount of CAM data. But if we integrate data from different communities to obtain a bigger amount of CAM (for example attention from different LORs), there will probably not exist common objects, users or courses that could be used to generate relations between the communities.

6. RELATED WORK

Broisin et al in [32] propose a framework to capture usage information about Learning Object from different Learning Management Systems and Repositories in order to analyze the usage patterns of the users through a Management Application. The approach of this paper goes a step further, using the attention information to calculate metrics that could be used to improve existing tools. Broisin's work also uses a simplified format of attention (basically usage information) in a non-standard format, limiting the possible use of the information by other systems, because existing applications should be reprogrammed to produce that format. This work proposes the use of an extension of AttentionXML standard to be able to capture the CAM from a variety of systems that already produce it.

In a related area, digital libraries, Nicholson in [33] propose the fusion of bibliometrics analysis with user-related data mining to generate a new field of study, bibliomining. His proposal could be compared with the one presented in this work: Using the information about the book and the usage information generated by the interaction of the users with the digital library system to improve the user experience. While Nicholson mentions several ways in which the attention metadata could be used, he does not detail any specific metric to improve digital library systems.

7. CONCLUSIONS

The current immaturity of the tools to search and find Learning Objects could be overcome if CAM information is store through the lifecycle of the Learning Object and used to compute metrics for ranking and recommendation. These metrics should generate a meaningful and automated way in which Learning Object could be ranked. This work presented detailed methods to calculate various metrics and propose several uses for those metrics. The proposed calculations could also be applied to rank and recommend other reusable components from which CAM could be gathered, as it was shown for the case of open source software libraries example.

While the metrics are easy to calculate, and some initial data is also present, more research is needed to be able to assemble a large scale system that could gather the necessary amount of CAM in order to render the calculations meaningful.

8. REFERENCES

[1] Linden, G.; Smith, B. and York, J. *Amazon.com Recommendations: Item-to-Item Collaborative Filtering*. IEEE Internet Computing, 7, 1 (2003), 76-80.

[2] Shepherd, M.; Watters, C. and Marath, A. *Adaptive User Modeling for Filtering Electronic News*. In Proceedings of the 35th Annual Hawaii International Conference on System Sciences, 2002. HICSS. (2002), 1180- 1188.

[3] James, S. Outfoxed Collaborative Browsing, http://www.getoutfoxed.com. Retrieved on May, 2006.

[4] Najjar, J., Meire, M. and Duval, E. *Attention Metadata Management: Tracking the use of Learning Objects through Attention.XML.* In Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications. (2005). 1157-1161.

[5] Najjar, J., Wolpers, M. and Duval, E., Attention Metadata:Collection and Management", WWW2006 workshop on Logging Traces of Web Activity, Edinburgh, Scotland, (2006).

[6] AttentionXML, AttentionXML specifications, http://developers.technorati.com/wiki/attentionxml. Retrieved on June, 2006

[7] Duval, E. and Hodgins, W., *A LOM research agenda*. In Proceedings of WWW2003: Twelfth International World Wide Web Conference, (2003), 659-667.

[8] Ochoa, X. *Learning Object Repositories are Useful, but are they Usable?* In Proceedings of IADIS International Conference Applied Computing. (2005), 138-144

[9] Duval, E. *LearnRank: the Real Quality Measure for Learning Materials.* Policy and Innovation in Education - Quality Criteria, (2005)

[10] Aizawa, A. *An information-theoretic perspective of tf–idf measures*. Information Processing and Management, 39, (2003), 45-65.

[11] ISO/IEC JTC1 SC36. International LOM Survey: Report. http://mdlet.jtc1sc36.org/doc/SC36_WG4_N0109.pdf (2004).

[12] Ariadne Foundation. Ariadne Foundation. http://www.ariadne-eu.org (2005).

[13] Collis, B. and Strijker, A. *Technology and Human Issues in Reusing Learning Objects*. Journal of Interactive Media in Education, 4, (2004).

[14] Verbert, K. Jovanovic, J. Gaševic, D. and Duval, E. *Repurposing Learning Object Components.* OTM 2005 Workshop on Ontologies, Semantics and E-Learning, (2005).

[15] IEEE-LOM Editor, http://www-i5.informatik.rwth-aachen.de/i5new/staff/chatti/LOMEditor/index.html. Retrieved June 2006.

[16] Cardinels, K., Meire, M., and Duval, E. *Automating metadata generation: the simple indexing interface*. In Proceedings of the 14th WWW conference, (2005), 548-556

[17] Dalziel, J. *Implementing Learning Design: The Learning Activity Management System (LAMS)*, ASCILITE (2003)

[18] ADL, SCORM Standard, http://www.adlnet.gov/index.cfm, Retrieved March, 2006

[19] IEEE. IEEE Standard for Learning Object Metadata. http://ltsc.ieee.org/doc/wg12/ (2002).

[20] Page, L., Brin, S., Motwani, R. and Winograd, T. *The PageRank Citation Ranking: Bringing order to the Web*. Technical Report, Computer Science Department, Stanford University (1998)

[21] Google Technology, http://www.google.com/technology/. Retrieved, August 2006.

[22] Radlinski, F. and Joachims, T. Q*uery Chains: Learning to Rank from Implicit Feedback*, Proceedings of the ACM Conference on Knowledge Discovery and Data Mining. (2005).

[23] Fan, W., Gordon, M. and Pathak, P. *A generic ranking function discovery framework by genetic programming for information retrieval*, Information Processing and Management. 40 (2004) 587–602

[24] Nascimento, M., Sander, J. and Pound, J. *Analysis of SIGMOD's co-authorship graph*. ACM SIGMOD Record, 32, 3. (2003). 8-10

[25] Girvan, M. and Newman, M. *Community structure in social and biological networks*. Proc. Natl. Acad. Sci. 11. (2002).

[26] Vivisimo Clustering Engine. http://www.vivisimo.com. Retrieved August 2006.

[27] Pigeau, A., Raschia, G., Gelgon, M., Mouaddib, N. and Saint-Paul, R. A fuzzy linguistic summarization technique for TV recommender systems. The 12th IEEE International Conference on Fuzzy Systems, 2003. FUZZ '03. 1 (2003) 743-748.

[28] Google AdSense, https://www.google.com/adsense/. Retrieved August 2006.

[29] Fan, W., Gordon, M. and Pathak, P. Incorporating contextual information in recommender systems using a multidimensional approach. Information Processing and Management. 40, 4. (2004). 587-602.

[30] Blinkx Contextual Search. http://www.blinkx.com. Retrieved August 2006.

[31] Sourceforge, Open Software Repository. http://www.sourceforge.net. Retrieved August 2006.

[32] Broisin, J., Vidal, P. and Sibilla, M. A Management Framework Based On A Model Driven Approach For Tracking User Activities In A Web-Based Learning Environment. EDMEDIA, (2006) 896-903

[33] Nicholson, S. The basis for bibliomining: Frameworks for bringing together sage-based data mining and bibliometrics through data arehousing in digital library services. Information Processing and Management. 42, 3 (2006). 785-804.