CHAPTER 7

Applying web usage mining for the analysis of behavior in web-based learning environments

K. Becker, M. Vanzin, C. Marquardt & D. Ruiz
Faculdade de Informática, Pontificia Universidade Católica do RS, Brazil.

Abstract

The extraction of students' navigation patterns can be an invaluable tool to evaluate the design and effectiveness of web-based learning environments. Web usage mining (WUM) addresses the application of data mining techniques over web data in order to identify navigation patterns. WUM is a complex process composed of three core phases: data pre-processing, data mining, and pattern analysis. In this chapter, we describe the key issues and challenges involved in each of these phases, illustrating them in a case study developed at the distance education department of our university. We then describe two tools we have developed to address key problems faced in this experience. LogPrep is an extensible and customizable pre-processing tool. It provides operators that automate typical tasks performed in this phase, and which are easily combined according to the mining goals using a visual language. O3R provides functionality to support the retrieval and interpretation of navigation patterns, based on the use of a domain ontology. O3R associates semantics to patterns dynamically, as they are analyzed. The combination of these tools with traditional mining algorithms have presented good results, simplifying and speeding up the WUM process, and allowing domain-related people to assume a pro-active role.

1 Introduction

Web-based distance education requires the development of proper learning environments that organize a set of individual and group activities, together with the necessary resources. A web-based Learning Environment (WBLE) is designed as

WIT Transactions on State of the Art in Science and Engineering, Vol 2, © 2006 WIT Press
www.witpress.com, ISSN 1755-8336 (on-line)
doi:10.2495/1-84564-152-3/07

a set of pages, through which the course is delivered and knowledge is shared among students and instructors. Web-based course management infrastructures (e.g. WebCT [1], ATutor [2]) provide a collection of resources to compose a WBLE. Common functionality includes content management, communication (e.g. email, chat), assignment submission, and various accessories (e.g. blackboard, calendar, and quiz). WBLEs establish a distributed and virtual interaction model, which makes difficult the observation and evaluation of how learning resources available in the site are actually explored by students. Typical monitoring functionality includes access statistics, recently accessed pages, and participation in communication tools. However, the evaluation of WBLE adequacy and effectiveness in the learning process is hard and subjective. Thus, monitoring functionality is limited for analyzing students' perception of the WBLE and usage tendency.

The extraction of students' navigation patterns can be an invaluable tool to evaluate the design and effectiveness of a WBLE. Web usage mining (WUM) addresses the application of data mining techniques over web data in order to identify navigation patterns [3]. Large volumes of data are collected from daily operations, and recorded automatically by Web servers. The analysis of this data can reveal how the site is actually being used, providing insights on how to arrange contents and services to better fit its users' needs. WUM is an iterative and complex process, which includes the execution of specific phases, namely data pre-processing (used to select, clean and prepare the log raw data), data mining (application of mining algorithms), and pattern analysis (retrieval and interpretation of yielded patterns to seek for unknown and useful information). WUM has been extensively applied in e-commerce [4–7] and its benefits are being extended to other domains such as distance education [8–10].

WUM allows various types of analysis over the learning process and/or the learning environment, which can be roughly classified into two classes of behavior: usage and navigation. Usage behavior focuses on how resources are used to perform learning activities. It allows the characterization of students' learning processes and models, based on the set of contents they study, tools they use, and how these resources are combined to accomplish goals or acquire competences. The navigation behavior allows investigating (un)frequently used paths, groups of students with similar access characteristics, disorientation, among others. It should be pointed out, however, that applying WUM for the analysis of WBLE effectiveness is even harder than in the e-commerce domain [4]. In the e-learning context, objectives and site effectiveness cannot be easily defined, nor measured. Distinct students may reach a same learning goal by accessing different resources, with distinct frequencies and in a different order. Hence, learning site analysis and evaluation cannot be performed independently of learning process evaluation.

In this chapter, we describe the key issues and challenges involved in each WUM phase, illustrating them in a case study developed at PUCRS-Virtual, the distance education department of our university. We then describe two tools we have developed based on the lessons learned from this practical experience. LogPrep [11] is an extensible and customizable pre-processing tool, which provides operators that automate typical tasks performed in this phase. A visual language allows combining

these operators easily in order to pre-process raw data according to mining goals. O3R [12] provides functionality to support the retrieval and interpretation of navigation patterns, based on the use of a domain ontology. O3R associates semantics to patterns dynamically, as they are analyzed. The combination of these tools with traditional mining algorithms have presented good results, simplifying and speeding up the WUM process, and allowing the active involvement of domain-related people.

The remainder of this chapter is structured as follows. Section 2 discusses each phase of the WUM process, and briefly describes some supporting environments. Section 3 describes the challenges faced in a real case study, and summarizes the lessons learned. The tools LogPrep and O3R are described in Sections 4 and 5, respectively. Section 6 discusses the contributions of these tools. Section 7 presents conclusions and future work.

2 The process of WUM

2.1 Pre-processing phase

Pre-processing involves performing several tasks with the aim of creating a user clickstream (i.e. sequence of page accesses), to be used as input in the data mining phase. The main data sources for that purpose are the Web server logs, which record all page accesses according to some standard format [13]. Typical information includes URL requested, IP that originated the request, request time-stamp, possibly user identification, etc. Pre-processing is the most difficult and laborious phase of the process due to the low quality of the available data, which is a consequence of missing data and disorganization [3, 13].

Moreover, there is a huge semantic gap between the events occurred in the site and how these are translated and recorded in the logs as a set of URLs [14]. When a user requests a page, actually several requests are issued to the server. These requests involve files (e.g. text, pictures, and style sheets) and programs that, together, compose the user's view of the page (i.e. page-view). Proxies, caching, dynamic pages and frame-based systems add additional difficulties [5].

Pre-processing tasks and challenges involved are thoroughly discussed in [13], and summarized below:

- *Data cleaning*: removes from the log the entries that are accessories to compose the page-views (e.g. graphics, style sheets).
- *User identification*: associates a URL request with the corresponding user. Most e-commerce applications are anonymous, thus requiring heuristics for inferring accesses of a same user.
- *Session identification*: groups all page references of a given user and breaks them up into user sessions (clickstream), according to time-oriented or referrer-based heuristics.
- *Path completion*: fills in page references that are missing due to caching.

- *Transaction identification*: breaks down sessions into smaller units, referred to as transactions or episodes. Various criteria are used, such as maximal forward reference, content transactions, auxiliary/content transactions, etc.
- *Data enrichment and integration*: consists in providing meaning to page references contained in the log, possibly by the integration of data from heterogeneous sources (e.g. various types of logs, organizational database, page contents, site topology, user data).

2.2 Data mining phase

Well-known mining techniques have been extensively applied in WUM to extract usage patterns [3, 7]. *Association rules* relate pages that most often are referenced together in a user session. In the WBLE domain, they can reveal which contents students tend to access together, or which combination of tools they explore during their learning processes. *Sequential patterns* describe related accesses in a specific order. It could reveal which content motivated the access to other contents, or how tools and contents are entwined in the learning process. *Clustering* groups together a set of items having similar characteristics. It is suitable for finding pages with similar contents, users with similar navigation behavior, or similar navigation sessions. *Classification* allows characterizing the properties of a group (e.g. user profiles, similar pages, learning sessions).

Most works in WUM do not focus on new mining techniques, but rather on how to efficiently combine existing techniques to explore new applications (e.g. adaptive sites and recommendation systems [7], site evaluation [4]).

2.3 Pattern analysis phase

An interesting pattern is valid, new, useful and simple to understand [15]. Pattern retrieval and pattern interpretation are the key issues in this phase. Mining techniques (e.g. association, sequence) typically yield a huge number of patterns, most of which are incomprehensible or uninteresting to users [16]. Pattern retrieval deals with difficulties involved in managing a large set of patterns. Pattern interpretation deals with pattern interestingness and relevance in regard to the domain. In the context of WUM, pattern interpretation challenges address the semantic gap between URLs and user events [10, 14].

Filtering is a common pattern retrieval approach. A filter defines the properties that patterns must present in order to fit in the analyst's current interest and search space. Filtering mechanisms can be applied in both mining and pattern analysis phases [16]. In the mining phase, the filter is embedded into the mining algorithm, restricting its output. In the analysis phase, it is used to interactively focus the search on potentially (un)interesting patterns, without having to re-mine data.

A filter can express statistical, conceptual and structural properties. Support and confidence are examples of objective statistical filters [17], which aim at reducing the number of rules yielded by mining algorithms. Beliefs expressing domain

knowledge [5, 16] are examples of subjective statistical measures. The use of conceptual and structural properties in filtering is presented in [4, 18].

Conceptual properties in WUM are related to domain events, i.e. contents and services offered by a site, which are represented syntactically by URLs. There is an urge for approaches to provide semantics to these URLs. Semantic is most frequently provided by data enrichment performed in the pre-processing phase (e.g. [6, 19]). This approach is static, in the sense that a new perspective about the patterns may imply re-preprocessing data to enrich it differently. The Semantic Web opens new perspectives for this challenge [14, 20].

2.4 Support environments

Websift (formerly known as Webminer [5, 13]) and the environment named *WUM* [4, 6] are examples of dedicated suites for developing WUM applications. They provide the core pre-processing techniques discussed in Section 2.1. *Websift* offers various mining algorithms, whereas the environment *WUM* is restricted to a specific sequence technique. Both provide functionality for pattern analysis, based on visualization techniques and filtering mechanisms. The environment *WUM* includes MINT, a mining (filtering) language, and visualization functionality for navigation paths. *Websift* supports comparing patterns with domain beliefs about page contents and site structure. WUM can also be developed with the support of generic KDD (Knowledge Discovery on Databases) suites. Commercial and academic suites (e.g. Clementine [21], Intelligent Miner [22], Amadea [23] and Weka [24]) provide generic pre-processing functionality, which have to be complemented or extended by other applications for WUM purposes. They offer various mining algorithms, and filtering and visualization techniques for pattern interpretation.

3 WUM challenges in practice: a case study

This section describes a project developed at PUCRS-Virtual [9]. WebCT is the main infrastructure for the development of learning sites and management of distance courses at PUCRS-Virtual. The project focused on understanding WUM potential for analyzing the effectiveness of WBLEs. The goal was to model this problem as a WUM application, and to explore abstractions and types of patterns that could help in the analysis of site usage. The subject of study was an intensive extracurricular course, which lasted 11 days and involved 15 students, represented by a log containing 15,953 records.

We established a framework for interpreting page accesses in terms of the learning processes that motivate them. The framework helped us to understand the mapping of the learning environment into the technological infrastructure, the specifics of the course at hand and its site, as well as WebCT functionality. Emphasis was settled on how the learning resources were distributed and accessed in the site. This particular course was organized as a set of predefined activities (e.g. learning a concept). For each activity, the use of different resources was recommended or at least expected

(e.g. learning objects, quiz, and communication tools). The web site was designed aiming that all required resources were conveniently accessible for the ensemble of planned activities.

Our framework also defines a paradigm for the search of navigation patterns. It focuses on the actions taken by students over the technological resources available to perform each planned activity. Hence, each planned learning activity is used as a unit for guiding and orienting the WUM process.

This application was exploratory, and the various tasks performed are described in next sections in terms of WUM phases. The lessons learned, which motivated the development of new tools to support WUM, close this section.

3.1 Pre-processing phase

In this phase, we basically applied over the web server log the techniques presented in [13] according to the defined framework. However, specific types of analyses demanded adaptation of these techniques, such as the concept of learning session [8, 9], and different types of transactions. For instance, a *learning session* includes all accesses underlying the execution of one learning activity, representing a time span that ranges from minutes to days, possibly implying that students logged in and out several times. Pre-processing techniques were employed with the support of the Intelligent Miner (IM) suite, in combination with specific purpose applications we developed using embedded SQL and Java.

Enriching data with semantics was the most difficult task of this phase. In the WebCT, most page accesses actually correspond to script executions. URLs are incomprehensible and do not provide any hint about their relationship to learning resources. Classical techniques for capturing site topology and content (e.g. crawler, information retrieval) could not be employed. We manually mapped each URL to the corresponding event. We also developed a taxonomy of events. For example, *send-email* and *read-email* were generalized as *email*, which was generalized as *communication*. This taxonomy was explored in subsequent phases of the process.

Pre-processing techniques were combined differently to produce various datasets, according to the analysis goals established. For instance, if the goal is to find correlation of contents in an activity, we can employ the learning session technique, and divide sessions using the content transaction technique. On the other hand, if the goal is to evaluate the paths between correlated resources, we can use a conventional session technique, without breaking sessions into transactions.

3.2 Data mining phase

Association and sequence mining techniques were employed with the support of the IM suite. According to the activity-oriented approach, we applied filters constraining the presence of specific resources, which were either expected or unexpected. We also sought for generalized patterns [25] (e.g. *Hypertext.pdf* $\rightarrow$ *Communication* is the generalized pattern of *Hypertext.pdf* $\rightarrow$ *Chat*, if the taxonomy defines *Communication* as the generalization of *Chat*).

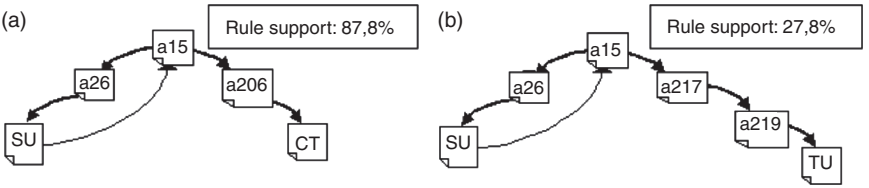

Figure 1: Pattern examples.

Several experiments were run using datasets pre-processed differently, and for different activities. Figure 1 illustrates two sequential patterns extracted for examining site design in terms of available paths among commonly used resources. These patterns represent sequences of accesses that include WebCT submission functionality (SU). This page is reached after accessing two auxiliary pages (pages that provide links for other pages), depicted in the picture using numbered labels (a15, a26). After that, they return to auxiliary page a15.

In Pattern (a) students then proceed to a chat page (CT), whereas in Pattern (b), they head towards the submission tutorial (TU). These patterns possibly reveal a problem related to the submission functionality. Because students have difficulty on its use, very frequently they cancel the submission and seek for help (chat or tutorial). The stronger support of Pattern (a) suggests it represents the most common behavior. Pattern (b) shows a more autonomous behavior with regard to the same problem. However, from a site design perspective, there is no indication of the existence of such a tutorial in the submission page. So perhaps students became aware of its existence by discussing through the chat, or browsing in the site. This is an example of how difficult it is to separate design and learning issues.

3.3 Pattern analysis phase

The goal of this project was to illustrate extractable patterns, and to understand their power if used as an instrument for site usage analysis. Hence, developing this phase to produce real knowledge was beyond the scope of this project. We limited pattern analysis to various discussions with a domain expert. At each interaction, we would show preselected rules yielded by our experiments. Based on her extensive knowledge of the course at hand and WebCT infrastructure, the expert would suggest possible pattern interpretations, such as the ones provided for the patterns in Fig. 1. To aid her comprehension, we would show at first generalized patterns, and when an interesting pattern was identified, we would search for more specific related patterns and deepen the discussion. IM did not provide adequate support for these tasks. Moreover, frequently questions raised by the expert would imply in running new experiments, producing new rules to be validated.

It should be pointed out that the distance education department staff was excited about the results, and willing to participate in a more thorough evaluation. However, there was no available domain expert who could dedicate the time required, particularly given the huge number of patterns.

3.4 Lessons learned

The project yielded very interesting results, but confirmed that in practice the extraction of interesting patterns is hard. Most difficulties were related to pre-processing and pattern analysis. The main lessons learned are described below.

3.4.1 Pre-processing issues

Although most e-commerce techniques could be transposed to the education domain, different approaches were also necessary (e.g. learning session). WUM-dedicated environments provide the core techniques summarized in [13], and implementing variations is not necessarily easy, if possible. KDD suites are difficult to extend to include WUM specific algorithms, and imply a lot of programming.

This application was exploratory: different types of analysis were interactively defined and for each of them, the process was developed according to the goals settled. A key issue was the alignment of mining goals with a set of pre-processing techniques that organize raw data such that intended type of patterns were extractable. Combining all required techniques with the support of different, non-integrated applications was very time-consuming, and resulted in a difficulty for creating and managing the data sets.

The structure and content of a site is a critical input to overcome the semantic gap between site events and URLs. WebCT is mostly based on dynamic pages, and its internal structure makes very difficult to add semantics to URLs. Moreover, the developed framework enabled to considered not only the site itself, but also the characteristics of the overall learning process that guides students' actions. All this knowledge was used to semantically enrich the clickstream, in a laborious, time consuming and error-prone process. However, it was implicitly embedded in the dataset, and could not be explored in the subsequent phases.

3.4.2 Pattern analysis issues

Pattern interpretation can only be performed if the meaning of patterns is intuitive. This problem is typically addressed using static enrichment, performed in the pre-processing phase. In practice, however, very frequently the result of a certain phase suggests new ways of enriching data or new types of analysis. This may imply a return to the pre-processing phase to produce new data sets. The limitations of static semantic enrichment is a major lesson learned.

The overwhelming number of rules yielded by the chosen algorithms is another key issue. This problem was worsened by the use of a taxonomy to produce generalized patterns. Each experiment yielded thousands of rules, most of them redundant or representing domain common sense. Raising the support threshold reduced the size of output, as much as its interest. Explicitly represented domain knowledge is invaluable for filtering rules that are potentially relevant for interpretation.

Filtering reduced the number or rules, by focusing on certain resources (un)-planned for each activity, but with a number of disadvantages. First, this strategy requires knowledge about course planning. Second, there was a lot of redundancy among patterns yielded by different filtering, and the suite adopted provided

no support for consolidating the results. Explicit domain knowledge could support filtering definition and the integration of results.

Providing rules at different abstraction levels revealed itself invaluable for establishing a closer dialog with domain experts. However, the KDD suite lacked support to relate generalized rules with their corresponding specific ones, as well as to establish other types of relationships.

We interacted with the staff of PUCRS-Virtual during the whole project, but carried out most tasks of the process ourselves. After an initial project set-up for understanding the domain and business goals, the only interaction with PUCRS-Virtual staff was during pattern analysis tasks. Each of these interactions would trigger another cycle of the process. The knowledge of the domain-related people is essential for all WUM phases, and they should be given the proper means to be as actively involved as possible.

4 LogPrep: a customizable pre-processing tool

From the lessons learned, we derived a set requirements for designing LogPrep. It should: (1) support the automation of pre-processing tasks; (2) provide alternative operators, corresponding to different techniques used to accomplish a same task; (3) be extensible and customizable in terms of the provided operators; (4) support the easy combination of operators to prepare data according to different mining goals; and (5) support the active involvement of domain-related people by providing appropriate concepts and means to manipulate them. By domain-related people we refer to people with knowledge about the domain (in our case, the learning environment, such as designers, instructors), and a few skills on WUM-related issues.

LogPrep supports the visual definition, reuse and execution of configurations of pre-processing operators. An *operators' configuration* defines a sequence of pre-processing tasks to be applied over the data, where each operator defines a specific technique to execute a task. An operator is an algorithm that implements a technique proposed in the literature for a task (e.g. time-based session) or a new approach (e.g. learning session). The operators' configuration concept allows addressing pre-processing as the activity of combining task-oriented techniques in accordance with mining goals. Consider the examples provided in Fig. 2. Each configuration is suitable for attaining specific mining goals, and the difference between them is simply how the transaction identification task is performed (auxiliary transaction vs. content transaction operators).

A visual language addresses usability and user-friendliness issues related to the definition and execution of operators' configurations. It allows users to intuitively regard the pre-processing phase as a simple visual configuration of task-level operators. Hence, LogPrep does not require advanced skills or extensive training from users. Since each operator automates a task, users can experiment different configurations and observe extracted patterns. If results are not satisfactory, the configuration can be easily altered (add/change/remove operators), resulting in a significant reduction of time and effort. It should be stressed that users are not guided in

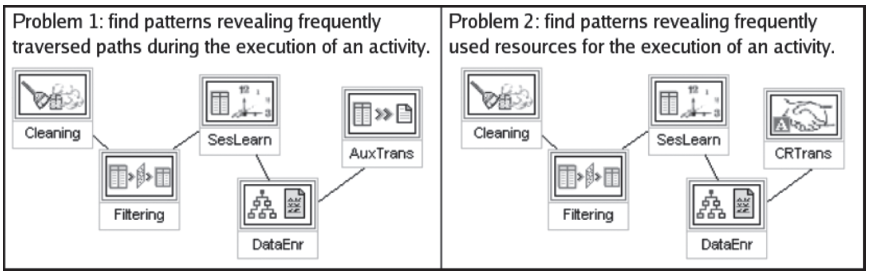

Figure 2: Configuration examples.

the configuration activity, given that expressing knowledge that precisely characterizes a class of problems is not trivial [26]. Nevertheless, LogPrep contributes to this problem in two ways. First, it highlights all tasks that can be applied and combined, with their variance in terms of operators. Second and more important, successful configurations can be documented and made available for later reuse through the concept of *configuration template*. By selecting and refining a configuration template that addresses goals similar to the ones of the problem at hand, users can easily prepare data for the extraction of potentially interesting patterns.

The remainder of this section summarizes the striking features of the current prototype, developed in Java. Further details can be found in [11].

4.1 Configuration language and configuration template

Figure 3 illustrates the user interface of the prototype. The visual configuration language supports all functionality related to the definition and execution of operators' configurations by direct manipulation of operators in the *Configurations Area*. It is a graph-based language, where the nodes represent the operators and the links, the flow of data execution. Hence, each node represents an execution unit that receives a dataset as input, processes data according to its role and parameters, and outputs a transformed dataset. This approach can be found in tools such as Clementine and Amadea. The visual language allows users to: (1) define an operators' configuration, which includes the operators, their parameters and their order; (2) load various types of inputs (e.g. log, enrichment data); and (3) export data. Operators are selected from the ones available in the *Tasks Area*. Two execution modes are provided for a defined configuration: *complete*, where all operators are applied in the defined sequence and a final dataset is produced; or *step-by-step*, where the transformed dataset can be inspected right after each operator is applied.

An operators' configuration can be saved as a template. A configuration template represents successful pre-processing for a well-defined problem class, a procedure for a specific type of analysis, etc. The *Templates Area* allows documenting, searching, inspecting and retrieving templates. The user loads a selected template in the Configurations Area and edit it to create a new configuration.

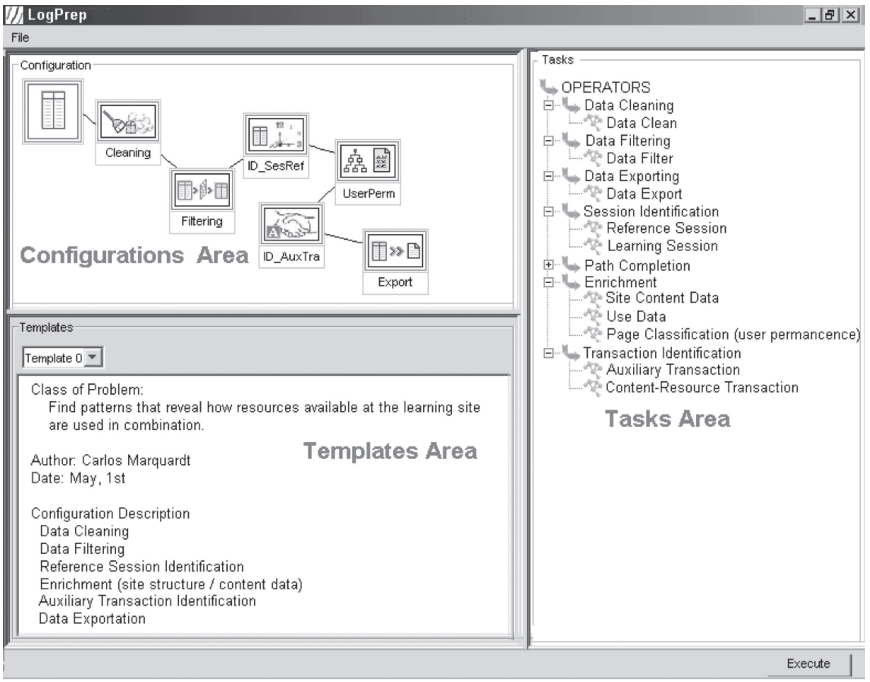

Figure 3: LogPrep user interface.

4.2 Customization features

The extension and customization of LogPrep must be in charge of someone with technical programming skills, referred to as the tool administrator. Operators are executable components that have to be programmed in a compatible language (e.g. Java). The tool administrator customizes LogPrep using a set-up file describing in XML: (1) the tasks, their respective operators, and the components that implement them; (2) rules establishing valid connections and execution flows between operators. When LogPrep is launched, it reads this set-up file and loads the operators, displaying them in the Tasks Area. This mechanism allows the inclusion of new operators for the development of other pre-processing tasks (or the execution of tasks according to different techniques), or even a complete set of operators, targeted at the domain at hand.

5 OR3: ontology-based rule rummaging and retrieval tool

O3R focuses on support for the pattern analysis phase, according to the following requirements: (1) it should support both pattern retrieval and pattern interpretation in an integrated manner; (2) patterns analysis should be based on domain events,

disregarding their syntactical representation in terms of URLs; (3) events should be easily interpreted and manipulated according to various abstraction levels and dimensions of interest, without implying re-processing raw data or re-mining it; (4) it should be possible to establish and maintain various types of relationships between related patterns; (5) domain knowledge should be explicitly represented and exploited by both tool functionality and user; and (6) it must support the active involvement of domain-related people.

O3R functionality encompasses *pattern rummaging*, *pattern filtering* and *pattern clustering*. The former is targeted at pattern interpretation, whereas the later two focus on pattern retrieval. The striking feature of O3R is that all functionality is based on the availability of the domain ontology, composed of concepts describing domain events, into which URLs are mapped. This feature allows the retrieval and interpretation of conceptual patterns, i.e. patterns formed of concepts, in opposition to physical patterns, composed of URLs. Hence, users can interactively explore pattern semantics during analysis activities, according to distinct abstraction levels and dimensions of interest. This approach enables to overcome the limitations of static semantic enrichment. All functionality is based on direct manipulation of visual representations of conceptual patterns and ontology, thus enabling a pro-active involvement of domain users with minimal training and limited technical skills. Since the ontology makes the domain knowledge explicit, users are expected to be familiar with the domain, but not necessarily experts. Users explore the ontology to learn about the domain, and interpret and retrieve patterns more easily, based on domain characteristics.

Current implementation of O3R is limited to sequential patterns extracted according to the sequential algorithm described in [25]. It assumes that these patterns were extracted from a dataset resulting from a typical pre-processing phase. Due to the availability of the domain ontology, no particular data enrichment is assumed for this data set in the pre-processing phase. For the same reason, the mining algorithm should not generate generalized patterns.

5.1 Ontology representation

O3R assumes the representation of domain events in two levels: conceptual and physical. At the physical level, events are represented by URLs. The conceptual level is represented by the domain ontology. The ontology is composed of concepts, representing either a content of a web page, or a service available through a page. Concepts are related to each other through hierarchical or property relationships. A hierarchical relationship connects a descendant concept to an ascendant one. Two types of hierarchical relationships are considered: *generalization*, in which the generalized concept is ascendant of a specialized one; and *aggregation*, in which the ascendant represents the whole assembly and the descendent represents one of its parts. Every concept has at most one ascendant. Property relationships represent arbitrary associations that connect a subject to an object.

URLs are mapped into ontology concepts according to two dimensions: service and content. An URL can be mapped into one service, one content or both, in which

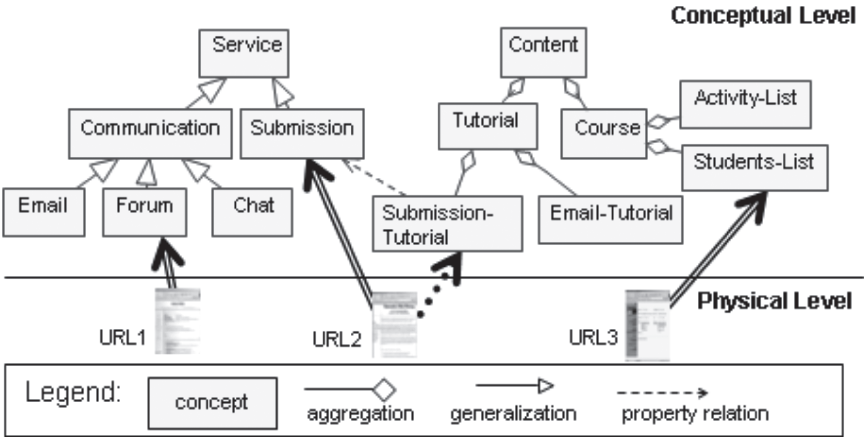

Figure 4: Mapping URLs to semantic concepts.

case the predominant dimension must be defined. A same concept can be used in the mapping of various URLs. Figure 4 illustrates this ontology structure by describing the semantics of a WBLE. Services include chat, email, submission, authentication, etc. Content is related to the material available in the site, or the subject related to some service. In Fig. 4, *URL1* was mapped to the service concept *Forum*; *URL2* was mapped to both service *Submission-Tutorial* and content *Submission* concepts, where service dimension was defined as predominant; and *URL3* was mapped to the content concept *Students-List*.

5.2 Conceptual pattern representation

Since no semantic enrichment is assumed in the pre-processing phase, mined patterns are sequences of URLs. O3R uses the mapping between the physical and conceptual events to present these physical patterns as a sequence of the corresponding concepts, i.e. the *conceptual patterns*. Users manipulate conceptual patterns using the provided functionality. For their analyses, users always have to establish a *dimension of interest*, which can be *content*, *service* or *content/service*. Considering the ontology of Fig. 4, the physical pattern *URL1* → *URL2* corresponds to the conceptual pattern *Forum* → *Submission* according to the both service dimension and content/service dimension (where the predominant dimension is used). The physical pattern *URL2* → *URL3*, according to content dimension, corresponds to *Submission-Tutorial* → *Students-List*.

Conceptual patterns can be interpreted according to different abstraction levels by exploring the hierarchical relationships. For example, the pattern *URL2* → *URL3* can be interpreted as *Submission-Tutorial* → *Students-List*, *Tutorial* → *Students-List*, *Tutorial* → *Course*, *Content* → *Content*, and so on.

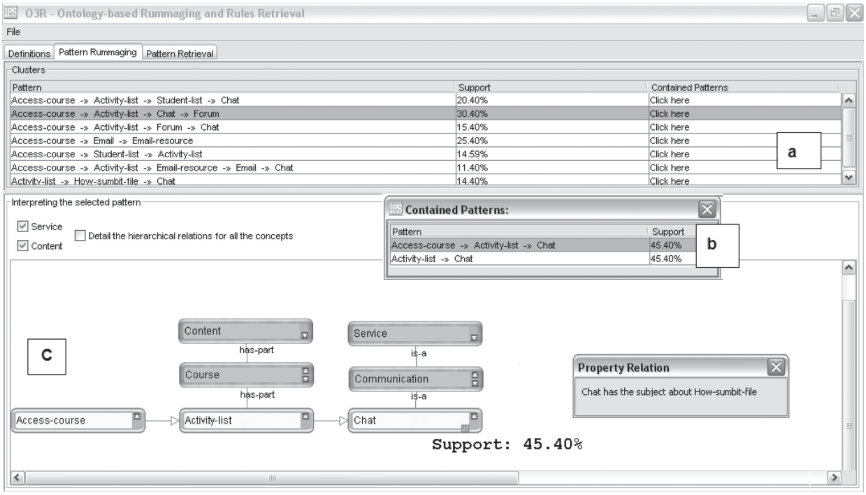

Figure 5: Pattern rummaging interface.

5.3 Pattern rummaging

Interpretation activities are supported through concept-oriented interactive rummaging. Rummaging explores the ontology to: (1) represent patterns in a more intuitive form, thus reducing the gap between URLs and site events; (2) allow pattern analysis according to different dimensions of interest and abstraction levels; (3) establish different relationships between patterns. Figure 5 displays the pattern rummaging interface, in which various representations of conceptual patterns are depicted. This interface is divided into three areas: *Clustering* (Figure 5a), *Contained Patterns* (Figure 5b) and *Rummaging* (Figure 5c). To rummage around a pattern in the Rummaging area, the user chooses a pattern (from the Clustering area, Contained Patterns area, or Filtering Interface), and defines a dimension of interest. In the example, the user has selected the service/content dimension, thus resulting in the conceptual pattern *Access.course* → *Activity-list* → *Chat*. By changing the dimension of interest to service, this same pattern would be visualized as *Access.course* → *Visualize-Information* → *Chat*.

Two groups of operations allow to explore ontology relationships: *detailing* and *drill*. Figure 5c illustrates the resultant pattern after using detailing operations to better understand the events represented by the pattern. The user related pattern concepts *Activity-List* and *Chat* to their respective generalizations, and queried a property relationship of which *Chat* is the subject.

Drill operations are similar to roll-up and drill-down in online analytical processing, and they are a means to establish relationships among patterns in different abstraction levels. Roll-up is used to obtain a generalized pattern, whereas drill-down finds the specific related patterns. Figure 6 illustrates a generalized pattern obtained by rolling up the pattern of Fig. 5c (concept *Chat* was rolled up to

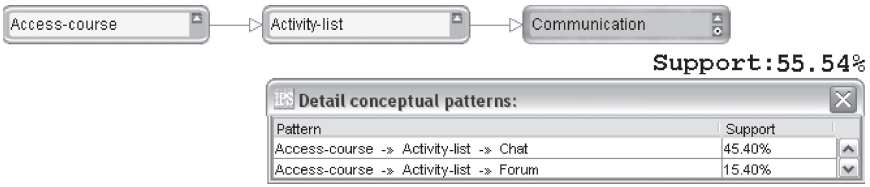

Figure 6: Generalized and specific conceptual patterns.

Communication), with the respective support. Figure 6 also presents a window displaying the patterns found using drill-down. This approach can be contrasted with the generation of generalized rules during the mining phase [25], which results in the generation of a huge set of unrelated rules. In our approach, generalized rules are created on-demand, and it is always possible to relate them to the respective specific rules. Further details can be found in [12].

5.4 Pattern clustering

Retrieval functionality is targeted at managing large volumes of rules. The basic idea is to reduce the interpretation search space by finding sets of related rules. Clustering groups related rules in different sets, such that the analyst can set focus for further inspection on a whole set of rules (or discard them all), as depicted in the Fig. 5. Hence, clustering and rummaging are closely integrated. O3R prototype is currently limited to the maximal sequence criterion to group rules that are subsequences of a maximal sequence [25], but criteria are possible.

5.5 Pattern filtering

Filtering is another mechanism in O3R to manage the elevated number of rules, establishing a relationship among rules that match a same filter. Users have the support of the ontology to understand the domain and establish event-based filters. Filters are quite expressive, in that it is possible to define conceptual, structural and statistical constraints. Users are not required to learn any complicated syntax, because filters are defined visually by direct manipulation of domain concepts and structural operators. Two filtering mechanisms are provided, referred to as *equivalence filtering* and *similarity filtering*. Filtering and rummaging are closely integrated: users choose a filtered pattern to rummage around, which possibly leads to a new filter definition, and so on.

5.5.1 Filter definition

Conceptual constraints define the interest on patterns involving specific domain events, at any abstraction level. Structural constraints establish an order among events (i.e. start, end, and be followed by). Statistical constraints refer the support of sequential rules. Figure 7 shows O3R filtering interface. The domain ontology

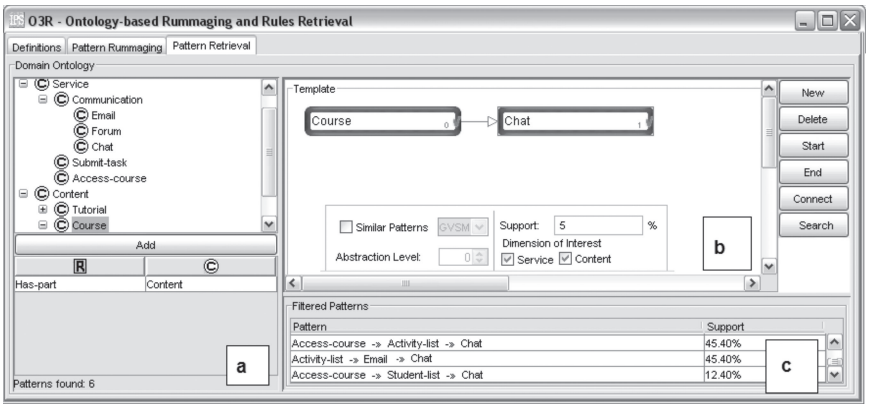

Figure 7: Pattern retrieval interface.

is represented graphically on the left most window (Fig. 7a), which displays all concepts and relationships. To establish conceptual constraints, the user chooses concepts from the ontology and places them at the *Filter Definition* area (Fig. 7b). He then uses the structural operators (buttons at the right of the Filter Definition area) to organize concepts. In the example, the user is interested in sequential rules involving any event classified as *Course*, (immediately or not) followed by the *Chat* event, with at least 5% of support.

5.5.2 Equivalence filtering

The filtering mechanism examines all conceptual patterns, verifying whether each one of them meets the statistical, conceptual and structural constraints of the filter. The statistical constraint is verified by comparing the pattern support with the support threshold. A conceptual constraint states all concepts that must appear in a rule. In the equivalence filtering, a concept is contained in a conceptual pattern either if it explicitly composes the pattern, or one of its descendants does. Structural constraints verify whether these concepts are in the correct order. Figure 7c illustrates possible patterns according to the equivalence filtering mechanism.

5.5.3 Similarity filtering

Similarity filtering extends equivalence filtering. Similarity is defined in terms of the distance between concepts in the ontology, considering the hierarchical relationships [27]. The user must provide the *minimum similarity threshold* and the *ancestor scope level* (ASL), i.e. the farthest common ancestor in the hierarchy to be considered for similarity. In the example of Fig. 7, if ASL is defined as 1, *Communication* is the farthest common ancestor of *Chat*, and therefore, *Forum* is a similar concept. Each filtered pattern has a similarity measure, calculated according to the similarity of individual concepts and structural constraints. Figure 8 exemplifies

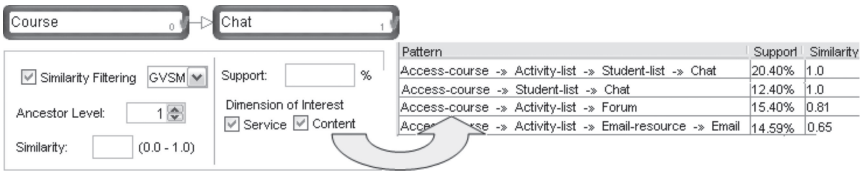

Figure 8: Similarity filtering.

similarity filtering. Patterns with similarity equal to 1 are equivalent to the filter, whereas the others are similar in some degree.

6 Discussions

LogPrep and O3R have been used in combination with traditional mining algorithms for developing WUM applications in our research group. We are also developing experimental evaluations with the staff of PUCRS-Virtual to confirm our claim that these tools are suitable for domain-related people.

LogPrep was tested by seven users with background in computers in education and mathematics. As a preparation, we gave them a 30 min talk about WUM and preprocessing, and developed two training exercises. Then, they were given five mining goals statements, for which they had to develop the corresponding configurations. They were able to establish these five rather complex configurations by themselves, without a single mistake. They all also recognized that two exercises involved slightly different mining goals, and that they could develop a configuration by modification of a previously developed one. They highlighted as advantages: the user friendliness of the visual language, the easiness of structuring the configuration at task level, and the possibility of reusing configuration templates.

LogPrep is suitable for both exploratory and plan-based applications. In exploratory applications, users add, change and remove operators to try out different results, and reapply it over the same data set. Plan-based applications can benefit from configuration templates, which can be applied with the same purpose over different data, possibly with different parameters. Table 1 summarizes the advantages of LogPrep in accordance with the design requirements it meets.

O3R evaluation is currently limited to the demonstration of its functionality to the same expert who participated in the previous experience (Section 3.3). For that purpose, we developed the domain ontology, reproduced one of the experiments yielding a set of sequential patterns, and enacted a typical interaction occurred at that time. We started by showing the clusters, from which the expert selected a rule for rummaging. She detailed the pattern, changed the dimension of interest, rolled the pattern up to generalize it, and then drilled it down to find related patterns, which she selected again for rummaging, and so on. From the insight gained through rummaging, she showed interest on patterns with specific properties, which were filtered with the support of the ontology. She then selected some filtered patterns

Table 1: LogPrep advantages.

Requirement	Advantage
Automation of pre-processing tasks	Reduction of the time and effort spent in pre-processing, as well as errors.
Combination of operators at task level	Easier alignment of mining goals and required data pre-processing.
Alternative operators to accomplish a same task	Flexibility for preparing data as required.
	Easier creation and management of data sets due to the use of a single tool.
Extensibility and customization	Environment can be fully adapted to different contexts and domains.
Active involvement of domain-related people	Direct translation of WBLE evaluation goals into data pre-processing requirements.

Table 2: O3R advantages.

Requirement	Advantage
Close integration of interpretation and retrieval	Support for exploratory and hypothesis-based analysis.
Event-based analysis	Pattern intuitiveness.
	Easiness for identifying interesting patterns.
Interpretation of events according different perspectives	Dynamic enrichment of data.
	No re-execution of previous phases.
Support for various types of relationships among patterns	Reduced number of rules.
	Identification of rules with similar properties.
	Ability to relate generalized and specific patterns.
	Easy identification of redundant patterns.
Explicit domain knowledge representation	Support for developing analysis tasks.
	Deeper insight of the domain.
Active involvement of domain-related people	Direct identification of useful and interesting patterns for the domain.

and rummaged them, leading to the definition of new filters, exploring all the interactivity provided by O3R. In comparison with her previous experience, for which almost no support was provided, the expert highlighted the following advantages: interactivity, intuitive pattern representation, visualization of patterns according to various perspectives, ability to establish various types of relationships, and support provided by domain ontology to perform analysis. An empirical validation with a larger sample of users is under definition. Table 2 summarizes O3R advantages in accordance with its design requirements.

O3R supports both exploratory and hypothesis-based pattern analysis. The former is suitable when the expert does not know what to expect and wishes to explore relationships among concepts and among patterns to identify interesting patterns. In exploratory analysis, filtering is most frequently a consequence of the insight provided by rummaging. Hypothesis-based analysis focuses on filtering for defining hypotheses, and rummaging for interpreting results.

7 Conclusions and future work

In this chapter, we discussed and illustrated in a real case study the challenges of applying WUM in the WBLE domain. We then described two tools developed in response to critical issues faced in that experience. A common characteristic of these tools is that they are aimed at allowing an active involvement of domain-related people in different phases of WUM. Preliminary results display evidences that this requirement was suitably achieved. This is a crucial feature if WUM is to be used in complement with monitoring functionality to understand and evaluate students' behavior.

Our goal is to integrate these tools in a complete framework targeted at identifying learning processes and models, understanding of site usage, and evaluating WBLE effectiveness. For that purpose, the framework must support the application of various mining techniques (with the corresponding pre-processing) and analysis of yielded models. Currently we are extending and evaluating O3R and studying various issues involved in the application of clustering to understand students' behavior. O3R can be easily extended to support other mining techniques (e.g. association), as well as other algorithms for sequential patterns (e.g. [6]). Other limitations of O3R must be addressed, particularly the constraints upon the ontology structure and on the semantic mapping of URLs. Currently, LogPrep is being extended to include pre-processing tasks and operators required for clustering. Domain-ontology is considered for two purposes: defining similarity between learning behaviors and for the easy interpretation of learning clusters.

Future research includes, among others, the definition of the integration architecture for the framework, semantic enrichment through the semantic web as well as additional mining techniques (e.g. classification).

References

[1] WebCT, http://www.webct.com

[2] ATutor, http://www.atutor.ca

[3] Srivastava, J., Cooley, R., Deshpande, M. & Tan, P.N., Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Explorations*, **1(2)**, pp. 12–23, 2000.

[4] Spiliopoulou, M., Web usage mining for web site evaluation. *Communications of the ACM*, **43(8)**, pp. 127–134, 2000.

[5] Cooley, R., The use of web structure and content to identify subjectively interesting web usage patterns. *ACM Transactions on Internet Technology*, **3(2)**, pp. 93–116, 2003.

[6] Berendt, B. & Spiliopoulou, M., Analysis of navigation behaviour in web sites integrating multiple information systems. *VLDB Journal*, **9(1)**, pp. 56–75, 2000.

[7] Mobasher, B., Web usage mining and personalization (chapter 15). *Practical Handbook of Internet Computing*, ed. M.P. Singh, Chapman Hall/CRC/ Press: Boca Raton, FL, 2004.

[8] Zaïane, O.R., Web usage mining for a better web-based learning environment. *CATE: Proc. of the Conf. on Advanced Technology for Education*, Banff, Alberta, pp. 60–64, 2001.

[9] Machado, L. & Becker, K., Distance education: A web usage mining case study for the evaluation of learning sites. *ICALT: Proc. of the Int. Conf. on Advanced Learning Techs*, IEEE Computer Society, pp. 360–361, 2003.

[10] Becker, K. & Vanzin, M., Discovering interesting usage patterns in web-based learning environments. *Proc. of the Int. Workshop on Utility, Usability and Complexity of e-Information Systems*, pp. 57–72, 2003.

[11] Marquardt, C.G., Becker, K. & Ruiz, D.D.A., A pre-processing tool for web usage mining in the distance education domain. *IDEAS: Proc. of the 8th Int. Database Engineering and Applications Symposium*, pp. 78–87, 2004.

[12] Vanzin, M. & Becker, K., Exploiting knowledge representation for pattern interpretation. *Proc. of the Workshop on Knowledge Discovery and Ontologies – KDO*, Pisa, Italy, pp. 61–71, 2004.

[13] Cooley, R., Mobasher, B. & Srivastava, J., Data preparation for mining world wide web browsing patterns. *Knowledge and Information Systems*, **1(1)**, pp. 5–32, 1999.

[14] Berendt, B., Hotho, A. & Stumme, G., Towards semantic web mining. *ISWC: Proc. of the First Int. Semantic Web Conference on The Semantic Web*, Springer-Verlag: London, UK, pp. 264–278, 2002.

[15] Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P., The kdd process for extracting useful knowledge from volumes of data. *Communications of the ACM*, **39(11)**, pp. 27–34, 1996.

[16] Silberschatz, A. & Tuzhilin, A., What makes patterns interesting in knowledge discovery systems. *IEEE Transactions on Knowledge and Data Engineering*, **8(6)**, pp. 970–974, 1996.

[17] Hipp, J. & Güntzer, U., Is pushing constraints deeply into the mining algorithms really what we want?: an alternative approach for association rule mining. *SIGKDD Explorations*, **4(1)**, pp. 50–55, 2002.
[18] Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H. & Verkamo, A.I., Finding interesting rules from large sets of discovered association rules. *CIKM: Proc. of the Third Int. Conf. on Information and Knowledge Management*, ACM Press, pp. 401–407, 1994.
[19] Dai, H. & Mobasher, B., Using ontologies to discover domain-level web usage profiles. *2nd Semantic Web Mining Workshop at ECML/PKDD*, 2002.
[20] Oberle, D., Berendt, B., Hotho, A. & Gonzalez, J., Conceptual user tracking. *AWIC: Proc. of the Web Intelligence, First Int. Atlantic Web Intelligence Conf.*, pp. 155–164, 2003.
[21] Clementine, http://www.spss.com/clementine/
[22] Miner, I., http://www-3.ibm.com/software/data/iminer/fordata/index.html
[23] Amadea, http://alice-soft.com/html/prod_amadea.htm
[24] Weka, http://www.cs.waikato.ac.nz/ml/weka/
[25] Srikant, R. & Agrawal, R., Mining sequential patterns: Generalizations and performance improvements. *EDBT: Proc. of the 5th Int. Conf. on Extending Database Technology*, pp. 3–17.
[26] Bernstein, A. & Provost, F., An intelligent assistant for the knowledge discovery process. *Proc. of the Workshop on Wrappers for Performance Enhancement in KDD*, Seattle, WA, 2001.
[27] Ganesan, P., Garcia-Molina, H. & Widom, J., Exploiting hierarchical domain structure to compute similarity. *ACM Transactions on Information Systems*, **21(1)**, pp. 64–93, 2003.