CLICKSTREAM DATA MINING ASSISTANCE
A Case-Based Reasoning Task Model

Cristina Wanzeller
Departamento de Informática, Escola Superior de Tecnologia, Instituto Politécnico de Viseu, Campus Politécnico, Viseu Portugal

Orlando Belo
Departamento de Informática, Universidade do Minho, Campus de Gualtar, Braga, Portugal

Keywords: Web Usage Mining, Case-Based Reasoning, Processing model, Clickstream Data Mining Assistance.

Abstract: This paper presents a case-based reasoning system to assist users in knowledge discovery from clickstream data. The system is especially oriented to store and make use of the knowledge acquired from the experience in solving specific clickstream data mining problems inside a corporate environment. We describe the main design, implementation and characteristics of this system. It was implemented as a prototype Web-based application, centralizing the past mining processes in a corporative memory. Its main goal is to recommend the most suited mining strategies to address the problem at hand, accepting as inputs the characteristics of the available data and the analysis requirements. The system also takes advantage and integrates corporative related information resources, supporting a semi-automated data gathering approach.

1 INTRODUCTION

Nowadays, establishing a successful presence on the Web is challenging, yet imperative for most the organizations. The Web has matured and users have diverse and rising expectations. Thus, becomes vital to evaluate the effectiveness of Web sites and to find ways to realize opportunities offered by Web, acting more proactively towards reaching the goals of sites. The *Data Mining* (DM) or *Knowledge Discovery* (KD) process applied to data related to user interaction with the Web, known as *Web Usage Mining* (WUM) (Cooley et al, 1997), is a critical tool for both purposes. WUM can help organizations to transform such huge, but very rich, data source into actionable knowledge for improvements that lead to revenue. However, WUM learning curve is a serious obstacle to users without deep knowledge in the domain. Part of the difficulty stems from the subtle nature and intrinsic complexity of clickstream data. There are also a myriad of technical issues, options and particularities under practical WUM problems, in order to get useful results for a specific goal. Most of the success obtained by experts when dealing with WUM problems comes from their acquired know-how, and even they cannot provide general and consistent rules for problem solving.

Our idea to tackle the above obstacle relies on managing the knowledge gained from the experience in solving concrete WUM problems, inside a corporate environment, to build the basis for sharing and reusing such knowledge across the organization. This idea was realised exploring a *Case-Based Reasoning* (CBR) and corporative-wide approach. The CBR paradigm provides a framework capable of meeting our core demands. When exploited at corporative level and integrated with the mainstay organization's information technologies (Kitano and Shimazu, 1996), it may consolidate the past processes into a collective memory and promote the knowledge flow over a larger audience.

This paper describes the main design, implementation and characteristics of a CBR system devoted to assist users along the development and application of WUM processes. We give emphasis to the modelling and implementation of the major tasks that the system has to fulfil. The system aims at proposing the most plausible methods to apply on a concrete WUM problem, described through the data characteristics and analysis requirements. To achieve this aim, the system has to translate

Wanzeller C. and Belo O. (2006).
CLICKSTREAM DATA MINING ASSISTANCE - A Case-Based Reasoning Task Model.
In *Proceedings of the First International Conference on Software and Data Technologies*, pages 61-66
DOI: 10.5220/0001319000610066

imprecise descriptions, from the implicated (human and data) sources, into a proper target problem. Moreover, since it relies on the WUM know-how of the organization, it has to deal with extensive, dispersed and heterogeneous sources of information, ensuring the mechanisms capable of promoting its continuous learning from the corporative experience evolution. Therefore, supporting a semi-automated data gathering approach becomes opportune.

The system was implemented as a prototype application, joining, mostly, Web, XML, database and Java technologies, as well general and CBR specific methodological orientations. These options were made trying to achieve flexibility, according to the established requests. Besides, we are specifically interested in assuring the system extensibility, to simplify its progressive improvement.

2 ASSISTING WUM PROCESSES

The challenge lies in supporting KD, an exploratory and participant driven process, which does not result in exact solutions. It involves several actions and decisions, which comprise (Fayyad, 1996): picking relevant data; identifying proper DM functions; choosing suitable models and setting its parameters; transforming data to improve its quality, to better fit the methods assumptions and to answer a concrete problem. Such activities require a deeper technical understanding of the methods and are influenced by many factors. These factors are often complex and subjective, resulting in uncertain problem descriptions and biased success criteria. Clickstream data subtle nature, intrinsic complexity and massive volume increase even more the general challenge. Further, WUM problem types, the kinds of mining activities, the related practical applications and the key data items are less studied and structured.

Our goal is to promote a more efficient, effective and synergetic use of the corporative resources, decreasing the effort and time required to derive useful knowledge and bringing up together multiple valuable contributions. To achieve this goal, our system has to assist users in two essential ways: collecting, organising and storing the useful examples of WUM processes; proposing the most plausible mining plans to handle one WUM problem, given the target dataset and an informal description of the explicit analysis requirements.

The system has to be proper for users with varying levels of expertise. It should enable novice users to gain insight into the overall WUM development process and its utility, capturing all the core actions that led to the resulting knowledge from the initial data, and the underlying decision-making course. This means that it must kept knowledge about each process, covering dimensions as: (D) characterizations of the target data, at dataset and variables level; (T) categorizations of the WUM problem type, in terms of general and organization's own properties; (A) sequence of activities, including transformation and modelling steps, the involved data, the parameters settings and explanations; (K) prior and derived knowledge, concerning to facts that affected the analysis, the extracted knowledge and its relations to such facts.

Capturing, structuring, storing and sharing the above aspects at corporative level bring up three immediate issues. First, this calls for domain's standards. The *Predictive Model Markup Language* (PMML) (http://www.dmg.org/) is a XML-based norm to define and share statistical and DM models across compliant applications. Since PMML is widely accepted, it provides established vocabulary to adopt, insights to structure KD processes and an opportunity to automate some data gathering. Second, the inter-operation with corporative data management technologies is essential, to tack advantage of the available capacities and to leverage its potential. Such inter-operation can be realized for acquiring data characteristics and to manage the knowledge obtained from WUM processes. Third, the system has to collect a considerable amount of items. Thus, a prerequisite is to automate its gathering, as most as possible, from data sources and KD tools, ensuring an effective and consistent extraction across the heterogeneous types of sources.

Other functional requests rest on the suited support for KD. Greater flexibility is needed to cope with fuzzy problem descriptions, allowing partial specifications, enabling versatile enquiry patterns (e.g. similarity based) and searching for the best or close matches. To tackle KD uncertainly, it is common practice to propose multiple alternatives, meaning that, their presentation should combine indicators to aid decisions and access to successively further detailed information.

KD activities are usually performed by exploring previous experience in the domain, suggesting the CBR paradigm adoption as the framework to sustain the intended assistance. Each useful WUM process may correspond to a case, expressed in terms of the domain problem (comprising the D and T dimensions) and the respective applied solution (covered by the A and K dimensions). CBR is a learning and problem solving approach (Kolodner, 1993; Aamodt and Plaza, 1994) that emphasis the

role of prior experience during problem solving (Mantaras et al, 2005). Multiple strengths of CBR sustain our option. Just to name a few: it simulates (more systematically, Kolodner, 1991) human behaviour in solving real life problems; it provides a flexible similarity-based comparison; it can cope with incomplete and subjective information; it may use specific importance levels to focus more relevant features; it uses cases as a good way to justify decisions (Kolodner, 1991); it is a sustained incremental learning approach; it offers an open environment for integrating different kinds of techniques (Althoff, 2001).

3 PROCESSING MODEL

Many efforts have been elaborated the different aspects of CBR, including the discussion of broader perspectives on CBR as a systematic engineering discipline. Important contributions to these efforts arose from the reported experiences of successful development of CBR applications and from projects where methodology development was explicitly incorporated as a task. Positional work from other areas, as software engineering and knowledge engineering, is also vital (Bergmann et al, 1997) and contemplated on such contributions.

A methodology gives guidelines regarding both the development activities and its product, at different levels of abstraction. At a very high level of abstraction (Bergmann et al, 1997), an incremental prototyping development of a CBR system is considered to be the most effective strategy (Bartsch-Spröl, 1996). The underlying approach is comparable to the spiral model (Boehm, 1988). A sequence of prototype systems is generated and each prototype development can be regarded as one cycle in the spiral model. In the following we consider the product of the development, at the level of its model description, focusing on the manner in which the system might tackles the problem, by defining the system processing model.

Several models have been devised to put CBR in practice. The most influent and widely acknowledge ones comprise the knowledge container model from (Richter, 1995), the process model of the CBR cycle and the task-method-decomposition structure for CBR (Aamodt and Plaza, 1994). The former model focus the different kinds of knowledge found in CBR systems - vocabulary, similarity measures, case base and adaptation knowledge, suggesting a well-established approach to structure the knowledge representation within CBR systems. The remaining models provide means for structuring the CBR system itself, being complementary, since they represent two views of CBR: a general one, identifying the major CBR sub-processes - retrieve, reuse, revise and retain, their interdependencies and outcomes; a task-oriented view, expressing the hierarchical decomposition of general tasks into subtasks and related methods to accomplish them.

We have modelled the CBR process using six central tasks, adapting the typical CBR cycle to our functional requirements. Figure 1 shows these tasks, their interconnections, inputs and outputs, as well the main involved knowledge containers. Five of them are decomposed on Figure 2 into the major subtasks that the system assures. Under the classic CBR cycle the two tasks characterize and construct and the task transform are integrated, respectively, as subtasks of the retrieve and retain steps.

The characterize task has as mission to translate the initial data to be mined into a systematic (meta-) representation for: capturing properties significant to methods selection; replacing the original data within the comparison of distinct datasets. The data characterization includes metadata extracted automatically and properties that cannot be derived, leading to the following subtasks: the dataset source and the supplied metadata are **collected**; some of this information is then used to access the source and to **extract** metadata from raw data; all the obtained metadata is **preserved** on the provisory repository for later use. The collect subtask may occur more than once, for instance, over variables, which are first extracted and then might be enriched.

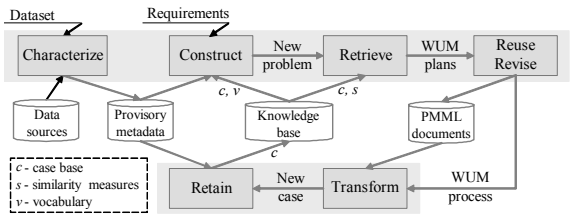

Figure 1: Processing model.

The construct task elaborates a new problem, being activated by a request, expressing the (analysis) requirements, based on the vocabulary and, usually, on a previously characterized dataset, to impose explicit restrictions. It may also rely on an existent problem from the case base. Building a target (problem) consists mainly in getting a set of relevant descriptors and the respective expected values. It starts by **obtaining** the available descriptors, theirs domains, types of applicable exact constraints and default expected values. After

filtering the irrelevant or unknown features, the final descriptor's values, exact constraints and importance levels are used to **define** a new target problem.

The retrieve task basically comes up with a set of alternative WUM plans to address the incoming problem, using mainly the similarity and case base containers. It has split into five subtasks. First, the **search** subtask uses the target problem to find out a set of plausible candidates from the case base. Second, the target and candidate problems are **matched**, calculating similitude values. Third, the most promising candidates are **selected**, given a similarity threshold value. Fourth, the candidates are grouped based on the distinct solutions they hold, which are **evaluated** determining indicators by solution type. Finally, the candidate cases are **organized**, considering the solution type, the degree of similarity and the evaluation indicators. The two last subtasks focus the main parts of the candidate cases that may be transferred to the target problem, preparing the derivational reuse process (i.e. the reuse of the method that constructed the solution).

The reuse and revise steps are done outside the system. In fact, the system does not perform extensive adaptation, in the wide sense, neither contemplates the adaptation container. The user makes part of the reasoning process, choosing among the proposed plans and adapting and revising them to the current needs, counting on explanations to aid these steps. The final WUM process and the PMML document(s) supplied by the KD tool might be entry points to the next phase of learning.

The transform (data) task accepts a heterogeneous description, collected through PMML documents and (or) user interaction, and builds a coherent and correctly sequenced intermediary representation. It aims at supporting the description of a WUM process from the user's convenience point of view, covering two subtasks: **combine** PMML documents and complementary information; (get and) **convert** the relevant PMML elements, to produce a compatibly description.

The retain task essentially augments the case base with a new WUM process and is organized into three subtasks. First, all the available information about a WUM process is **integrated**. Second, such information is **structured**, according to the internal schema, and the WUM process is catalogued, considering the kind of problem and the type of solution, to simplify its future reuse. Third, the case is **stored** in the knowledge base, testing the existence of the different items (e.g. DM functions, models, parameters) and adding the new ones, as well transferring the provisory metadata to the case base. Subsequently, the augmented case may be edited and additional information (e.g. discoveries and even new mining activities) may be integrated.

4 IMPLEMENTATION

The system was realised as a prototype Web-based application developed in Java environment. It follows a client/server typical architecture and is structured into three layers of services: interface, business and data. The options concerning the implementation were made giving priority to free software with open code and multi-platform, and to accepted standards and *Application Program Interfaces* (API). The application's client side uses, mostly, HTML, supported by style sheets formatting and by *Javascript* programming for validation and enhancement of browser behaviour and user interaction. The server side was built on top of the *Java 2 Standard Edition* (http://java.sun.com/javase). The business logic is in charge of Java components and the interface services front-end is based on the *Java Server Pages* (JSP) specification (http://java.sun.com/products/jsp/). The JSP/*Servlets* container *Apache Tomcat* (http://tomcat.apache.org/) assures the publication of such services. The data services explore different API (e.g. JDBC, JAXP), to support and abstract the access to diverse data sources.

Figure 3 shows the main building blocks (packages) of the application over the three layers of services. Within data services, *plain file access*, *sql access* and *xml access* yield generic persistence services for the three types of sources. These functionalities are reused by the remaining services.

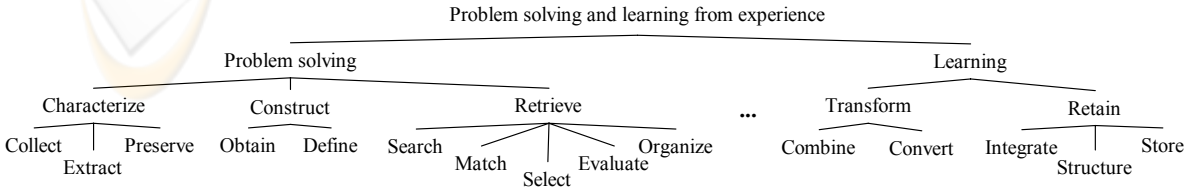

Figure 2: Task decomposition.

pmml access implements essentially the convert subtask previously discussed, supporting multiple PMML versions. *ds source persistence* and *kb persistence* assure, respectively the access to different types of dataset sources and the manipulation of the knowledge base, providing the required functionality encapsulation. For instance, *kb persistence* isolates and hides the case base storage and schema implementation details, realized by a relational database management system.

Figure 3: Application building blocks by layers of services.

The business service layer encapsulates the core functionality of the system, organized under the following main components:

- *domain* stands for and manages the whole collection of WUM process objects, separating the domain model from the methods that undertake the reasoning process;
- *data characterizer* and *construct problem* accomplish the characterize and construct tasks, exploring also interface and data services;
- *configuration* represents the vocabulary and retrieval parameters, so that the task may be generic. For example, it sets attributes used to describe a problem, their domains and weights;
- *similarity* provides a collection of available global and local similarity measures, which can be added independently;
- *retrieve process* implements the retrieve task, being supported, mainly, by *similarity* and *configuration* services;
- *retain process* is responsible for the retain reasoning task, exploiting also other services.

The interface services enable access to the system functionality and support the presentation logic, relying directly on two types of components, Web pages and an *interfaceservices* package.

The class diagram (Figure 4) in *Unified Modelling Language* (http://www.omg.org) simplified notation, shows the major classes involved within retrieve task. The *Retrieve* class controls the task, started after the problem construction, which produces a *CbrTarget* object. *CbrTarget* implements the define subtask and uses methods from *RrProperties* to update some dynamic properties.

CasesSelected embody all the plausible candidate cases. It implements the search subtask, using *CbrTarget* specifications and the *PCasesSelected* subclass services. This subclass knows how to convert tuples and attributes distributed over multiple (case base's) tables into candidate cases. It appeals to its specific methods and also to the common and shared ones provided by the *PGeneralKB* superclass. Conversely, *CbrCase* stands for each individual case and functionality.

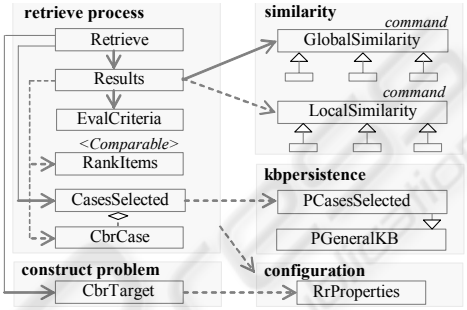

Figure 4: Retrieve process class diagram.

The match and select subtasks are realized through the *Results* class, which is supported by a collection of similarity functions (objects). Here, the command design pattern (Gamma et al, 1995) has an important role (encapsulating commands as objects), so that the retrieve process may stay generic and extensible. The subtasks evaluate and organize are (extensible) ensured by the *EvalCriteria* and *RankItems* classes. *EvalCriteria* determines average values (by solution type) of the evaluation criteria features, obtaining such features from *RrProperties* as a variable length set. *RankItems* implements the *Comparable interface*, providing a dynamic sorting logic, based on: the type of solution; the similitude level; the relative importance given by the user to the evaluation criteria; the variable set of evaluation features. So, this set of features may be changed (modified, increased or decreased) easily, without affecting those two related classes.

5 CONCLUSIONS

We implemented a prototype CBR application to assist users along WUM development and application. It intends to centralize the knowledge about useful WUM processes in a corporative case base and to enable its reuse across the organization. We believe that this application is a useful tool to less skilled users, not only to deal with a WUM problem in cooperation with it, but also to learn or

get new insights about problem solving from the contributions of experienced users. The application may reduce WUM experts' workload, and even they may recall previous effective solutions, instead of solving the problems from scratch.

In this paper, we described the main design, implementation and characteristics of our system, focussing the modelling and implementation of the major tasks that it has to fulfil. The methodological orientations adopted (and reported), were basically a spiral-prototyping incremental development approach and the CBR knowledge-level process model based on the (Aamodt and Plaza, 1994) ideas.

To achieve its aim, the system essentially: translates a raw target dataset into a meta characterization, reflecting inherent restrictions; guides users within the problem description, regarding explicit analysis requirements; produces a set of alternative possible solutions, exploring knowledge from previous WUM experiences; supports a semi-automated data gathering approach to describe new experiences; captures, structures and stores the relevant knowledge from the new experiences into a knowledge base for future sharing and reuse. Under these tasks, the system considers human sources along the organization and integrates other resources, such as corporative data sources and PMML documents, representing the knowledge extracted from data and based on a widely accepted and supported standard in the DM domain.

The system's (current) prototype was implemented combining, mostly, Web, PMML, database and Java technologies. With these options we hope to: win flexibility with respect to the user interaction and application accessibility and use; take advantage from the DM domain's standards; leverage corporative resources; embrace Java environment portability, objected-oriented features, flexibility and Web advantages. Furthermore, the case storage, the domain model and the reasoning steps have been handled as independently as possible, to simplify the application development and to assure its extensibility.

Currently we are working on the construction of a wide set of cases to enlarge the case base and to enable more exhaustive evaluation tests of the system. The obtained results, so far, point to the system effectiveness, but a systematic evaluation becomes necessary. Afterwards, we plan to elaborate further the system, namely, case base maintenance, tacking into account factors such as cases utility and representatively, based on usage statistics and the level of relevance (e.g. distinct solutions they hold).

ACKNOWLEDGEMENTS

The work of Cristina Wanzeller was supported by a grant from PRODEP (Acção 5.3, concurso nº02/2003).

REFERENCES

Aamodt, A. and Plaza, E., 1994. Case–based reasoning: foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications*, IOS Press, 7(1), 39–59.

Althoff, K.-D., 2001. Case-based reasoning. *Handbook on Software Engineering and Knowledge Engineering*. S. K. Chang (Ed.), World Scientific, 549-588.

Bartsch-Spröl, B., 1996. How to make CBR systems work in practice. In *GWCBR'96, 4th German Workshop on Case-Based Reasoning*, Informatik-Bericht, 55, 36-42.

Bergmann, R., Wilke, W., Althoff, K.-D., Breen, S. and Johnston, R., 1997. Ingredients for developing a case-based reasoning methodology. In *GWCBR'97, 5th German Workshop on Case-Based Reasoning*, University of Kaiserslautern, 49-58.

Boehm, B. W., 1988. A spiral model of software development and enhancement. *IEEE Computer*, 21(5), 61-72.

Cooley, R., Mobasher, B. and Srivastava, J., 1997. Web mining: information and pattern discovery on the World Wide Web. In *ICTAI'97, 9th IEEE International Conference on Tools with Artificial Intelligence*, 558-567.

Fayyad, U.M., Piatetsky-Shapiro, G. and Smyth, P., 1996. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11), 27-41.

Gamma, E., Helm, R., Jonhson R. and Vlissides, J., 1995. *Design Patterns: Elements of Reusable Object Oriented Software*. Addison Wesley, Massachusetts.

Kolodner J, 1991. Improving human decision making through case-based decision aiding. *AI Magazine*, 12(2), 52-68.

Kolodner, J., 1993. *Case-based reasoning*. Morgan Kaufman, San Mateo.

Kitano, H. and Shimazu, H., 1996. The experience sharing architecture: A case study in corporate-wide case-based software quality control. *Case-Based Reasoning: Experiences, Lessons, and Future Directions*. D. Leake (ed), AAAI Press, Menlo Park, CA, 235-268.

Mantaras, R., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M., Cox, M., Forbus, K., Keane, M., Aamodt, A. and Watson, I., 2005. Retrieval, reuse, revision, and retention in case-based reasoning. *The Knowledge Engineering Review*, Cambridge University Press DOI.

Richter, M., 1995. The knowledge contained in similarity measures. (Talk) at *ICCBR'95, 1st International Conference on Case-Based Reasoning*, Lecture Notes in Artificial Intelligence 1010, Springer Verlag.