Constructing and Transforming CBR Implementations: Techniques for Corporate Memory Management[*]

Arijit Sengupta and David C. Wilson and David B. Leake
Computer Science Department
Lindley Hall, Indiana University
150 S. Woodlawn Ave
Bloomington, IN 47405
{asengupt,davwils,leake}@cs.indiana.edu

Keywords: Case-Based Reasoning, Database, XML, Corporate Memory, CBR Implementation

Abstract

Achieving widespread case-based reasoning support for corporate memories will require the flexibility to integrate implementations with existing organizational resources and infrastructure. Case-based reasoning implementations as currently constructed tend to fall into three categories, characterized by implementation constraints: *task-based* (task constraints alone), *enterprise* (integrating databases), and *web-based* (integrating web representations). These implementation types represent the possible targets in constructing corporate memory systems, and it is important to understand the strengths of each, how they are built, and how one may be constructed by transforming another. This paper describes a framework that relates the three types of CBR implementation, discusses their typical strengths and weaknesses, and describes practical strategies for building corporate CBR memories to meet new requirements by transforming and synthesizing existing resources.

1 Introduction

Constructing corporate memories that identify, acquire, and share relevant experiential knowledge across an organization is an important and challenging task. Knowledge that is stored and processed in ways optimized for one task (e.g., queries on a centralized database) may need to be used in a substantially different manner (e.g. for transfer and use over the web). Though the knowledge itself may not change, how it is collected, stored, distributed, and used may vary throughout the organization and as organizational goals change. Thus it is important to examine support mechanisms for varying implementational needs in building and maintaining corporate memories.

[*] This paper is a revised version of "On Constructing the Right Sort of CBR Implementation," to appear in *Proceedings of the IJCAI-99 Workshop on Automating the Construction of Case Based Reasoners*, Stockholm, Sweden, 1999. The authors' research is supported in part by NASA under award No NCC 2-1035.

Achieving widespread case-based reasoning support for corporate memories will require flexibility in integrating implementations with existing organizational infrastructure and resources, as well as with new vehicles for sharing knowledge. We need to investigate not only that particular implementations address some pieces of the puzzle, but also general implementation infrastructure on which all systems can build. CBR systems as currently constructed tend to fit three general implementation models, defined by broad implementation constraints on representation and process, which reflect evolutionary developments in CBR practice. We call these models task-based, enterprise, and web-based.

Traditionally, *task-based* implementations have addressed system goals based only on the constraints imposed by the reasoning task itself. Most research systems, for example, focus on particular (often idiosyncratic) methods and representations optimized to address a specific reasoning task, either to demonstrate the effectiveness of the method or to meet specific task goals.

Recently, there has been an increasing and successful trend of incorporating CBR into enterprise systems (e.g., (Watson 1997; Stolpmann and Wess 1998)) to leverage corporate knowledge assets by knowledge management (e.g., (Becerra-Fernandez and Aha 1999)). *Enterprise* implementations reflect the additional implementation constraints imposed on CBR systems as part of an overall enterprise architecture (see (Kitano and Shimazu 1996)). In our view, the most important implementational constraint in this context is that typically such CBR integrations must operate in conjunction with database systems, the mainstay of corporate knowledge activity. This will usually mean inter-operation with the more prevalent relational database systems (e.g., (Gardingen and Watson 1998; Kitano and Shimazu 1996; Allen *et al.* 1995)), but may also include object database systems (e.g., (Ellman 1995)). Thus enterprise CBR implementations provide for and make use of database functionality. Note that not all "enterprise CBR systems" will have an enterprise implementation in this sense.

Currently, CBR systems are emerging that take advantage of recent developments in knowledge representation and sharing on the world-wide web (e.g., (Shimazu 1998; Gardingen and Watson 1998; Doyle *et al.* 1998)). *Web-based* implementations reflect additional constraints imposed on CBR systems by conforming to structured document representation standards for web/network communication, in particular XML—Extensible Markup Language (Bray *et al.* 1998). Note that our type distinction here is based on the construction of the reasoning system itself, not on how it presents information. Thus a task-based implementation might have a web interface, and a web-based implementation might not.

Other constraints, for example proprietary software inter-operability or Standard Generalized Markup Language (SGML) compliance, could be used to characterize enterprise or web-based implementations (and certainly more than one factor may apply). We have chosen constraints that are timely, are representative of the type of implementation concern, and that have broad applicability. As such, these implementation characterizations are intended to be useful, not perfect. They represent implementation targets in constructing corporate memories, and varying task aspects and contexts may prefer one to another. Thus it is important to understand (1) how the models compare, (2) their individual construction, (3) their combination, and especially (4) how one may be constructed by transforming another. Transformations are useful when new task criteria suggest a model that differs from current implementation (conversion), and when differing models are used in different aspects of a combined system (combination—e.g., database storage, web communication, task-based front end). This paper outlines a framework of practical constructions and transformations, represented in Figure 1 (dashed lines represent additional information requirements), that we expect will play an important role in building and maintaining case-based corporate memories.

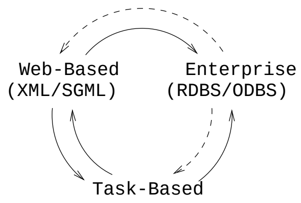

Figure 1: Relating CBR implementation types

2 Implementation Models

Our implementation characterizations can be applied at many levels of typical CBR systems, and here we find it useful to differentiate CBR process (retrieval, adaptation, evaluation) and (case) representation. Although we recognize the importance of complex object and object-oriented database models, as well as SGML ((ISO86 1986)), here we restrict our discussion to relational database models and XML.

Task-Based: Task-Based implementations account for the bulk of current CBR practice. These systems allow for highly tuned and efficient metrics and representations, but it may prove difficult to reuse them outside of the system context. Some efforts have used standardized representations to ameliorate these difficulties (e.g., (Manago *et al.* 1994)), but this is not widespread.

Enterprise: Integrating CBR implementations with enterprise database systems imposes standardization constraints that are almost universal in the enterprise community. Representations must accord with the table model of relational database systems (RDBS), while process must adopt Structured Query Language (SQL) conventions. CBR systems gain the strengths of the underlying RDBS, such as security, concurrency control, backup/recovery, and scalability. Moreover, integration allows the use of enterprise data both for normal corporate tasks (e.g., reporting), as well as for reasoning. However, because SQL has been designed to provide certain performance guarantees, it is limited in power, so refined metrics may be difficult to construct. Also, while complex cases are representable, they may be difficult to model in manual construction.

Web-based: XML is emerging as the vehicle for knowledge representation on the web. It provides a medium that allows (1) definition of customized representational markup languages and (2) application independent exchange of these complex hierarchical representations over existing web/network channels. XML also allows for customizable display of information using the associated Extensible Style Language[1]. While XML is currently viable for use (e.g., for transfer and parsing), it is a fairly new standard, so support (e.g., for browsing) is limited though growing. Its usability for applications such as CBR is also still evolving relatively rapidly (Hayes and Cunningham 1999). Thus some benefits are immediately available for individual systems, but developing standard representations for community knowledge sharing will be a crucial task for widespread use in the field. Since XML is primarily a representation standard, it is not as tightly coupled with process as are databases, so task-based applications are generally required for process. However, direct structured query mechanisms, analogous to SQL, are under development (Sengupta 1998; W3C 1998).

[1] http://www.w3.org/TR/1998/WD-xsl-19981216

3 Realizing Implementations

The realization of a framework for automatic implementation transformation involves outlining process and representation for each model, as well as defining and exemplifying the inter-model transformations. This section outlines the enterprise and web-based models (we omit the wide-ranging task-based model), and section 4 describes the transformations.

3.1 Enterprise/RDBS

Constructing an enterprise implementation involves associating a case structure with a corresponding relational database schema. Figure 2 shows an Entity-Relationship (ER) model for typical CBR systems, where stored data represents cases (problems) which result in proposed decisions (solutions), and their outcomes (evaluations). This ER model can be fully implemented in a RDBS. The construction is straightforward for feature-vector case structures, where a single table row corresponds to a case. For more complex case structures, relational normalization techniques are used to model the data.

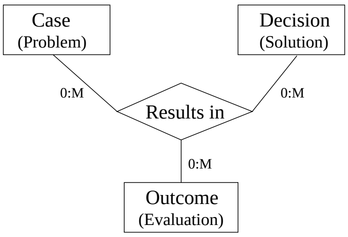

Figure 2: Entity Relationship diagram for a typical case-based reasoning process

Database systems can also be used for CBR process, for example by implementing k-nearest neighbor (k-nn) retrievals. A number of novel data structures have been proposed in the database literature for efficient implementation of k-nn algorithms (e.g., (Berchtold *et al.* 1997)), but standard database systems do not currently offer such support. However, if the similarity metric can be expressed as a numeric-valued function, database cases can be retrieved as ordered by the similarity results. Thus, in our view, the database/CBR processes may take place on at least three levels:

1. *Simple Storage*: The database is used only as a storage medium. All cases are retrieved and processed by an external system. This combines the storage benefits of the database systems with task-based processing power, but requires a full task-based implementation. The basic query to the database in this case is:

```
SELECT * FROM case_table
```

2. *Simple Retrieval*: A simple selection is performed based on conditions applied from the target, and the resulting subset is processed externally. This shifts part of the processing task to the database system, but may require considerable modeling effort to pre-compute similarity as in (Shimazu 1998), or to relax query specifications as in (Gardingen and Watson 1998; Daengdej and Lukose 1997). The basic query here is:

```
SELECT * FROM case_table WHERE conditions
```

3. *Metric Retrieval*: A metric function is used to order the rows based on a similarity value, `metric(c)`—a function of the target case c. This uses only the database system itself to perform a full k-nn analysis. This method is inefficient, since it must both compute and sort with every record and it loses the efficiency of optimized database indexing. Thus metric retrieval has been rejected previously in principle (Shimazu *et al.* 1993), but could prove useful (given available computing power) for some (smaller-scale) implementations, since it does not require additional/external case processing for retrieval. Determining the utility of this method for a particular application requires testing in context. We have used metric retrieval with good response time in a prototype application containing 4709 cases with 24 numeric-valued features. The basic query is:

   ```
   SELECT * FROM case_table ORDER BY metric(k)
   ```

To take full advantage of database capabilities, a pre-selection of the cases in the case-base could be performed using simple retrieval before evaluating metric retrieval, to reduce (if possible by exact/ranged matching) the number of retrieved cases.

3.2 Web-based/XML

Based on the entity-relationship model of CBR in Figure 2, we can also describe the structure of a full CBR system using an XML document type definition (DTD). Selected lines from the DTD are shown below:

```
<!ELEMENT CBR   (DATA, PROCESS?)>
...
<!ELEMENT DATA (PROBLEM, SOLUTION, EVAL?, RESULT?)>
<!ELEMENT PROBLEM (ATTRIBSET)>
<!ELEMENT ATTRIBSET (ATTRIB | ATTRIBSET)+>
...
<!ELEMENT PROCESS (METRIC+, ADAPT*)>
...
```

XML documents conforming to this CBR DTD describe the structure (i.e. meta-data) of particular CBR systems. Components of the case base are expressed as relations (attribute sets) and their constituent attributes. Complex hierarchies are supported by allowing sub-relations inside a relation (i.e., an ATTRIBSET inside another ATTRIBSET in the DTD). In contrast to other DTDs for CBR (Shimazu 1998; Hayes *et al.* 1998), we allow representation of both process (similarity, adaptation, evaluation) and (case) representation, either together or individually. For example, we are currently working on an implementation that incorporates MathML[2], an XML DTD for describing mathematical structure and content, to represent similarity metrics.

Using the XML model: An instance of the above DTD describes the actual case structure, which is used by a separate XML application to generate the proper structural definition (a separate DTD) of the case data. The actual case data can then be defined as conforming instances of the generated DTD. This two-step process has the following advantages:

[2] http://www.w3.org/TR/REC-MathML/

1. *Consistency*: By generating the case data DTD from the CBR system markup, we ensure that no separate check is necessary to assert that the structure of the case data (i.e., the separate DTD defining how the data should be structured and validated) is consistent with the data format required by the reasoning system.

2. *Validation*: Document type definitions in which the system attributes are represented as generic identifiers (tags) instead of XML attributes allow the case data to be validated against its DTD (as generated for consistency with the system) to ensure its integrity (i.e., the case data is in the form expected by the system). Given the DTD for the case data and the data itself, the validation can be performed by standard XML tools.

While XML has no particular associated process for retrieval, evolving query language implementations such as DSQL in DocBase (Sengupta 1998) and XML-QL (W3C 1998) will enhance the applicability of XML as a web-based CBR implementation model.

4 Transforming Implementations

Perhaps as important as the implementations themselves is the transformation of one implementation to another. This is useful in two situations: When new task criteria prefer a model that differs from current implementation, and when differing implementation models are used in different aspects of a combined system (e.g., database storage, web communication, task-based front end). Here we outline the transformations in the framework.

4.1 Web-Based → Enterprise

An XML representation of case structure can be converted to a database system using an XML application that processes XML markup tags/content and generates appropriate Data Definition Language (DDL) statements to create tables in a relational database. Consider the following fragment of a CBR system description, relating a people to their automobiles:

```
<ATTRIBSET NAME="Person">
  <ATTRIB ID="ID" REQD="REQD" TYPE="longint">SSN
  </ATTRIB>
  <ATTRIB TYPE="char" SIZE="20" REQD="REQD">Name
  </ATTRIB>
  <ATTRIBSET NAME="Auto">
    <ATTRIB TYPE="char">Make</ATTRIB>
    <ATTRIB TYPE="int">Year</ATTRIB>
  </ATTRIBSET>
</ATTRIBSET>
```

By parsing this XML fragment and mapping the XML structure to relational table structure, the patterns can be translated into the following relational DDL statements:

```
create table Person (SSN longint not null,
                     Name char(20) not null);
create table Auto (Person_SSN longint not null,
                   Make char(50), Year int);
```

For complex case structures, the application can adopt a simple foreign key strategy by augmenting a substructure with the key of the parent structure. In order to facilitate a possible future back-translation, this application should also update a database catalog (organized list) with the role of each created tables in the CBR model. A similar transformation application can be used to transform XML case data to fill the generated tables.

4.2 Web-Based → Task-Based

The main task in transforming an XML implementation to a task-based implementation is to identify a mapping between XML and task-based structures. We assume that the user or developer of the task-based systems will have the necessary tools and information to create case data in the task-based model. Taking CASUEL (Manago *et al.* 1994) as an example, an application like the one described in section 4.1 can generate appropriate CASUEL declarations from the XML structure. This process is similar to the Web-Based→Enterprise generation process, except that the generated statements are in CASUEL instead of SQL.

4.3 Enterprise → Web-Based/Task-Based

Transforming an existing database model into a conforming XML model or task-based model is more involved. Because the database lacks explicit case structure (when using more than a single table), transformation applications need to understand the role of various database objects in the CBR representation. Maintaining a catalog of the database objects and their roles, as suggested in section 4.2, should significantly reduce the amount of reasoning required prior to transformation. This process of role determination can be performed in several ways:

1. *Manual interaction:* The system may ask a user to assist in the process of determination of the roles of each of the objects,

2. *Catalog information:* The system may use a catalog that includes the roles of each of the objects,

3. *Mining:* The system may use data mining techniques to determine appropriate database objects and their roles.

The dashed lines in Figure 1 represent the extra information requirements for these transformations.

4.4 Task-Based → Web-Based/Enterprise

Converting from task-based to an XML or database format also depends on the actual task-based model, and the availability of tools that can assist in such transformation. For example, cases represented using CASUEL can be mapped into the corresponding XML schema or a database format using an application built on top of a CASUEL parser.

5 Conclusion

We have presented a categorization of current CBR implementation models into three classes, and shown how this view leads to practical support for building and maintaining case-based corporate memories. The general transformations from one implementation model to another allow for the conversion of existing implementations and facilitate the combination of implementation types to meet new and changing task requirements. We also view these methods as a natural extension and generalization of mining databases to aid in system construction.

Based on this framework, we present three challenges to the community: (1) to create community standard XML representation specifications for CBR, (2) to build a set of standard methods/libraries for translating between these XML representations and standard database representations, and (3) to develop standard CBR functionality within database systems. This requires shifting some attention to building infrastructure for the field in areas that are constrained enough to be feasible and consequential enough to be worthwhile. Community standards for representation have long been sought in many areas of AI. The structural foundations provided by web-based and enterprise media, coupled with the impetus for developing successful case-based corporate memories provides an environment ripe for achieving this goal for CBR. By providing general representational frameworks that already have ties to the world of practical application, as well as the tools to integrate them with one another and with traditional practice, the CBR community can shape the building blocks for constructing the next generation of successful research and industrial CBR systems. As CBR practice evolves, we expect the different implementation types to become increasingly integrated, and we hope to facilitate that transformation.

References

Jonathan RC Allen, David WR Patterson, Maurice D. Mulvenna, and John G. Hughes. Integration of case based retrieval with a relational database system in aircraft technical support. In *Proceedings of ICCBR-95*. Springer, 1995.

Irma Becerra-Fernandez and David W. Aha. Case-based problem solving for knowledge management systems. In *Proceedings of FLAIRS-99*. AAAI Press, 1999. To Appear.

Stefan Berchtold, Christian Bohm, Daniel Keim, and Hans-Peter Kriegel. A cost model for nearest neighbor search in high-dimensional data space. In *Proceedings of the 16th ACM PODS Conference*, 1997.

Tim Bray, Jean Paoli, and C. M. Sperberg-McQueen. *Extensible Markup Language 1.0*, 1998. http://www.w3.org/TR/1998/REC-xml-19980210.

Jirapun Daengdej and Dickson Lukose. How case-based reasoning and cooperative query answering techniques support RICAD? In *Proceedings of ICCBR-97*, pages 315–324. Springer, 1997.

Michelle Doyle, Maria Angela Ferrario, Conor Hayes, Pádraig Cunningham, and Barry Smyth. CBR Net:- smart technology over a network. Technical Report TCD-CS-1998-07, Trinity College Dublin, 1998.

Jeremy Ellman. An application of case based reasoning to object oriented database retrieval. In Ian Watson, editor, *First United Kingdom Workshop on Case-Based Reasoning*. Springer, 1995.

Dan Gardingen and Ian Watson. A web based case-based reasoning system for HVAC sales support. In *Applications & Innovations in Expert Systems VI*. Springer, 1998.

Conor Hayes and Pádraig Cunningham. Shaping a CBR view with XML. In *Proceedings of ICCBR-99*, 1999. To Appear.

Conor Hayes, Pádraig Cunningham, and Michelle Doyle. Distributed CBR using XML. In *Proceedings of the KI-98 Workshop on Intelligent Systems and Electronic Commerce*, number LSA-98-03E. University of Kaiserslauten Computer Science Department, 1998.

International Organization for Standardization, Geneva, Switzerland. *ISO 8879: Information Processing – Text and Office Systems – Standard Generalized Markup Language (SGML)*, 1986.

H. Kitano and H. Shimazu. The experience sharing architecture: A case study in corporate-wide case-based software quality control. In *Case-Based Reasoning: Experiences, Lessons, and Future Directions*. AAAI Press, 1996.

Michael Manago, Ralph Bergmann, Noël Conruyt, Ralph Traphöner, James Pasley, Jacques Le Renard, Frank Maurer, Stefan Wess, Klaus-Dieter Althoff, and Sylvie Dumont. CASUEL: A common case representation language. INRECA Consortium. Available on the World-Wide Web at http://wwwagr.informatik.uni-kl.de/∼bergmann/casuel/CASUEL_toc2.04.fm.html, 1994.

Arijit Sengupta. Toward the union of databases and document management: The design of DocBase. In *Proceedings of COMAD-98*. Tata McGraw Hill, 1998.

Hideo Shimazu, Hiroaki Kitano, and Akihiro Shibata. Retrieving cases from relational databases: Another stride towards corporate-wide case-base systems. In *Proceedings of IJCAI-93*, 1993.

Hideo Shimazu. A textual case-based reasoning system using XML on the world-wide web. In *Proceedings of the Fourth European Workshop on Case-Based Reasoning*. Springer, 1998.

Markus Stolpmann and Stefan Wess. *Intelligente Systeme für E-Commerce und Support*. Addison Wesley, 1998.

W3C. *XML-QL: A query language for XML*, 1998. http://www.w3.org/TR/1998/NOTE-xml-ql-19980819/.

I. Watson. *Applying Case-Based Reasoning: Techniques for Enterprise Systems*. Morgan Kaufmann, San Mateo, CA, 1997.