Shaping a CBR view with XML

Conor Hayes, Padraig Cunningham

Department of Computer Science
Trinity College Dublin
Conor.Hayes@cs.tcd.ie

Abstract. Case Based Reasoning has found increasing application on the Internet as an assistant in Internet commerce stores and as a reasoning agent for online technical support. The strength of CBR in this area stems from its reuse of the knowledge base associated with a particular application, thus providing an ideal way to make personalised configuration or technical information available to the Internet user. Since case data may be one aspect of a company's entire corporate knowledge system, it is important to integrate case data easily within a company's IT infrastructure, using industry specific vocabulary. We suggest XML as the likely candidate to provide such integration. Some applications have already begun to use XML as a case representation language. We review these and present the idea of a standard case *view* in XML that can work with the vocabularies or namespaces being developed by specific industries. Earlier research has produced version 1.0 of a Case Based Mark-up Language which attempts to mark-up cases in XML to enable distributed computing. The drawbacks of this implementation are outlined in this paper as well as the developments in XML that allow us to produce an XML "View" of a company's knowledge system. We will detail the benefits of our system for industry in general in terms of extensibility, ease of reuse and interoperability.

1 Introduction

Adding intelligence to Internet applications is an obvious role for Case-Based Reasoning (CBR). E-commerce sets out to sell products without the intervention of a sales-assistant and in the absence of human sales assistants there is a need for intelligent software assistants to lubricate the sales process. Since what *is* available is catalogue data and data on user behaviour and preferences CBR is an obvious technology to create these sales assistants. In this scenario, the obvious cases are descriptions of the commodities on sale and the task is to identify the case configuration that meets the user's requirements.[1] These cases might describe package holidays, hardware configurations, or real estate for instance.

The proposed standard for distributing data of this type on the Internet is XML (eXtensible Mark-up Language) so it is important that the CBR process can deal with data in this format. Indeed Shimazu (1998) and Watson & Gardingen(1998) have

[1] Several online CBR applications that conform to this secnario already exist; see http://wwwagr.informatik.uni-kl.de/~lsa/CBR/CBR-Homepage.html for some examples.

described CBR applications that receive cases in an XML format. We have already presented a proposal for CBML, a case description language based on XML (Hayes et al. 1998, Doyle et al. 1998).

XML is a description language that supports meta-data descriptions for particular domains and these meta-data descriptions allow applications to interpret data marked up according to this format. The meta-data description is the Document Type Declaration (DTD) and, for instance, a DTD for real estate will attach semantics to a document marked up in that format.

In (Hayes et al., 1998) we proposed a generic DTD for CBR called CBML that allowed cases to be marked up in an XML-based format. The major drawback of this approach was that data needed to be marked up in this CBR specific format but now the evolving potential of XML allows for an improvement on this idea. In an e-commerce situation domain specific DTDs exist and catalogue data will be marked up in this format – for instance RELML has been proposed as a standard for marking up real estate data. Any case-based assistant that would operate in this space would need to access this data. The new XML proposals for Namespaces and Schemas allow for a CBR *view* on this data and it is this approach that we describe here. This approach has the advantage that it uses existing XML data; it simply provides the appropriate CBR perspective on this data.

In section 2 we review two CBR applications that use XML as a case representation language to query a relational data base. In section 3 we examine and critique an earlier proposal for a standard case representation language in XML. We find that while the principle is still a sound one, the implementation is hampered by a failure to recognise the necessity of retaining the integrity of data marked up according to an industry specific vocabulary. The inability of the DTD to describe structured data objects such as a case base is also brought to light.

Section 4 reviews an XML standardisation project in the domain of Real Estate in the context of past CBR work in this domain. We argue that as standards such as The Real Estate Listing Mark-up Language (RELML) emerge, CBR techniques will have to integrate easily within these existing data structures. In section 5 we introduce the XML concepts of namespaces and schemas, which will allow us to integrate CBR with existing mark-up. We follow this in section 6 with our proposal for what a case namespace should look like.

2 Two XML–CBR applications

The Caret System by Hideo Shimazu is a development of earlier work on retrieving cases from a relational database (Shimazu 1998). It uses XML to mark up cases of natural language text describing technical support problems and solutions. Support staff mark up cases in XML and the documents are then parsed and stored in a relational database by the Caret system. Features either contain coded (discrete) data or textual data. However only the coded tags affect the retrieval mechanism.

The Caret system follows from work on the SQUAD system in which information retrieval using CBR is integrated with a relational database management system for reasons of security, data integrity, data standardisation and scalability (Kitano & Shimazu 1998). Indeed Kitano and Shimazu propose that CBR applications have been

too narrowly focused on domain specific problems. They suggest that a case based system should be viewed as a *medium* to be used in conjunction with the mainstream corporate information system. We would share this view, and we anticipate that a standard way of marking up cases will provide an opening in this respect.

The retrieval technique used in Caret is a version of the *Many are called Few are chosen* (MAC/FAC) retrieval methods outlined by (Gentner & Forbus 1991). This algorithm is chosen in order to allow SQL retrieval from the database without having to retrieve every record to compute similarity. Since it uses only coded tags as features by which to calculate similarity, Caret returns a rude subset of the case base relying on the client case adapter to further stream the matched queries.

Since Caret doesn't use the textual parts of its cases in its retrieval mechanism, it is not quite a textual Case Base Reasoning System. Its retrieval mechanism could as easily be applied to any type of data. The second example we look at applies ideas from the Caret system to a sales support system for the installation of air conditioning units.

The HVAC air conditioning sales support system uses a similarity table to send SQL queries to an Access data base of existing installations (Gardingen & Watson 1998). A Java servlet retrieves the cases and converts them to XML. The URL addresses are sent to a Java based client adapter which issues http requests for the XML files. The client then performs nearest neighbour ranking and displays the results.

Both implementations outlined use a vocabulary of user-defined tags and a DTD that assumes a case representation consisting of a feature list and corresponding values. In each implementation, the use of XML allowed the developer to define structured domain specific data. Furthermore XML data was downloaded by issuing HTTP requests from the client end.

However, these implementations highlight some shortcomings with representing case data in this way. Data typing is not possible, nor does or is there any allowance for feature weighting. Also Since DTD creation is a difficult task, it would make sense to use the standard DTDs or schemas which emerge for industry specific data. The task in this scenario would be to use the vocabulary from these DTDs or a combination of DTDs and render them in case form. What is being proposed is a facility to create a standard case *view* of data, baring in mind that XML is suited to full integration with database technology, and that XML documents can easily be created on the fly from an existing database or from several databases. However, as we shall discuss in the following two sections the current DTD model is not powerful or flexible enough to support this task.

The next section will look at our early attempts to create a standard case representation language. The shortcomings of this implementation anticipate the new W3C namespace recommendation and the W3C schema proposal which will allow us to create a standardised case view of existing corporate data.

3 Standard Case Base Representation language

We originally designed CBML as an XML application to facilitate the storage and exchange of case data over a network. The design was motivated by the need to initiate a standard for the exchange of case data, particularly in relation to distributed computing (Hayes et al. 1998, Doyle et al 1998). Version one of CBML was influenced by the functionality of the CASUEL language specification (INRECA 1994). Casuel was developed as an interface language between all INRECA component systems. It was intended to serve as a standard for exchanging information between classification and diagnostic systems that use cases. CBML was intended to establish the ground on which the CBR community could build a standard for the exchange and storage of case data on the Internet. Several shortcomings have emerged with this early implementation and we will deal with these below.

Despite the simplicity of this early implementation, CBML required a case structure file and case base file as well as their respective DTD files to describe each case file. This implies that at least two separate documents (depending upon whether the case structure file and the case base file include their respective DTDs) need to be transmitted. This requires a minimum of two http requests for each case base.

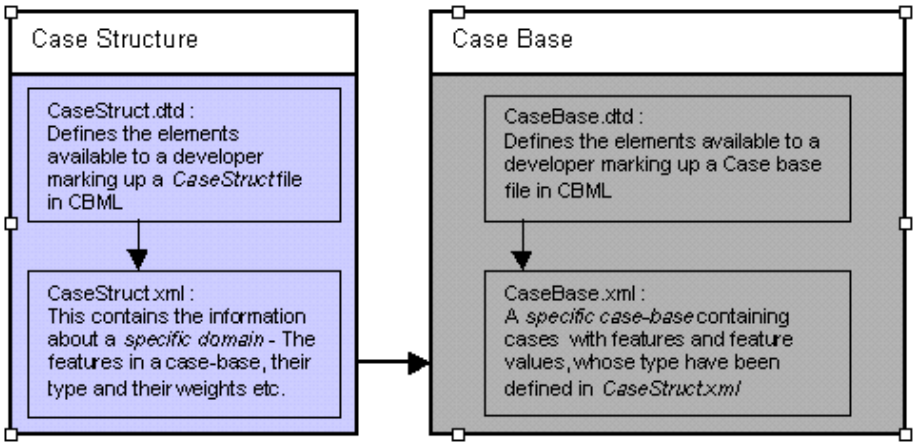

Fig. 1. The relationship between the four documents required to represent data in CBML

CBML is a flabby implementation which significantly increases the amount of data required to describe the case base. For instance, the case feature value pair `HolidayType = Bathing` is described in CBML as

```
<feature name="HolidayType">Bathing</ feature >
```

where the element `feature` and its attribute `name` are defined beforehand in the case base DTD. A less verbose way of describing the same pair would be to define the tag `<HolidayType>` thus allowing the XML feature value pair

```
<HolidayType>Bathing</HolidayType>.
```

From this example it is clear that the first implementation, as well as being over long, is not at all an intuitive representation of the feature value pair of this case. Secondly, and more importantly, in an attempt to maintain the generality of the language the portability of the data it describes has been compromised. Whereas the first example is meaningful to software that expects CBML data, it poses real problems to any software that does not expect the "feature" part of the "feature-value" pair to be represented as a value of the `name` attribute associated with the element <`feature>.`

Also, there was no way of elegantly merging elements defined according to another DTD with CBML elements without violating the readability and "meaningfulness" of the former. As in the example given earlier, a *bathing* holiday may be marked up as `<HolidayType>Bathing</HolidayType>` using a hypothetical travel agent's DTD. The CBML representation of this does not respect the mark-up for this type of data and imposes its own "case-centric" version:

```
<feature name="HolidayType">Bathing</feature>
```

Another drawback was the impossibility of having cases containing mixed data, that is, elements with tags defined according to different domains. These types of cases would occur in complex component based descriptions. Furthermore, In terms of XML document architecture, there is no logical connection between the case structure document and the case base document. (This shortcoming is dealt with in our new implementation). In this respect we were following the CASUEL syntax in which one document provides a case super-class from which a set of case instances are derived. We recognised that the super-class in our implementation was described in different parts by the casestruct.xml and the casebase.dtd since both these documents contribute to the description of a specific case structure within a domain. While the XML document architecture would suggest that the descriptive information in casestruct.xml should be contained in the casebase.dtd, it was not possible to merge these documents. Since the DTD only describes document structure it is not powerful enough to represent the extra information contained in the casestruct.xml document.

It became clear to us that the weakness of the CBML document architecture was tied to the limitations of the DTD to describe the type of data required by cases. Indeed, there is a growing opinion that the DTD is unsuited to the rigorous description demanded by data objects and that a new mechanism must be established[2]. This mechanism called a schema will support data typing, be easily extensible and achieve the inheritance requirements described earlier[3].

We will address the impact of this development in section 5.

---

[2] Connolly, D., Bray, T. W3C (1999) XML Activity Page, http://www.w3.org/XML/Activity.html

[3] Malhotra, A., Maloney, M., W3C (1999) XML Schema Requirements, http://www.w3.org/TR/NOTE-xml-schema-req

4 A Case Study

Before examining the feasibility of designing an XML based case based language, it is important to examine the situations in which such a language might be used and how it might interact with current data storage and representation techniques. We have examined already the case for Case Based Reasoning as a technology suited to the demands of internet commerce (Doyle et al. 1998). In drawing up a case based representation language, we have to examine whether we are helping or hindering the transmission of information that has been marked up in an *industry specific mark-up language*. It would seem to us that we should be looking to facilitate how data is marked up in other agreed DTDs rather than shoe-horning it into a "case-centric" mark-up language.

In the example given above, if the tag `<HolidayType>` were a standard tag for the travel industry, it would not make sense converting this to `<feature name="HolidayType">` simply to cater towards documents that store their travel data in the form of cases. We would argue that a case base document should be considered no differently that a document containing industry standard data.

A concrete example is the emergence of the Real Estate Listing Mark-up (RELML) language developed by OpenMLS and 4th World Telecom to facilitate searches on the web for real estate offered for sale from various agents through out the USA.[4] The motivation behind the RELML project is the recognition that local knowledge is important in describing and valuing property. Therefore rather than centralising this information, local real estate knowledge is marked up locally according to the publicly available RELML DTD. This data is then indexed by a crawler and from these indices searches can be performed from various web sites.

Providing a standard mark-up language allows different real estate agents list their properties in a form that can be easily searched, without losing the local knowledge which a centralised system might entail. More importantly, independent vendors are placed on a level footing with larger franchised businesses.

---

[4] OpenMLS, Real Estate Listing Management System, http://www.openmls.com
Rein, Lisa. (1998) The Business of Residential Listings. XML.com,
http://www.xml.com/xml/pub/98/08/real/openmls.html

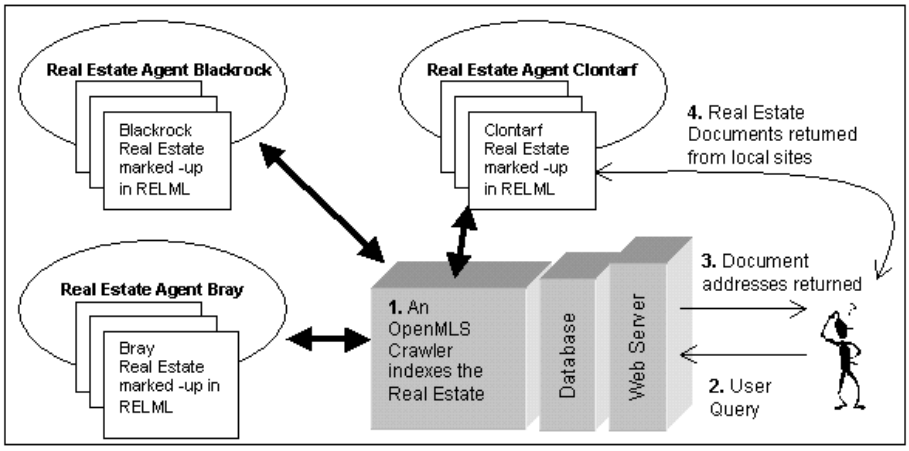

Fig. 2. The architecture proposed for the Real Estate Listings project consists of three tiers, where a crawler on the middle tier centrally indexes real estate data in XML from real estate member sites. Clients may then query this database and replies are delivered in raw XML or HTML depending on the ability of the client software.

Integrating CBR with XML and current database technology is particularly important if, as in this example, the case base is changing on a daily basis. The OpenMLS system anticipates the emergence of online brokerage facilities. Once the first wave of Internet commerce finishes, there still remains the problem of having a community of virtual shop fronts but no easy way to locate them, or compare prices and services. The next issue will be to establish an easy way to find a community of holiday vendors, for example, and be able to query their products quickly. This does not entail centralising knowledge, but exploiting the diversity of the Internet by providing a common language by which various vendors can market their goods. XML has already been suggested as a solution in this respect[5]. In the OpenMLS application the XML documents describing each property reside on the local real estate sites and are downloaded from the local site once a search at the broker end indicates a match. A better metaphor, therefore, than the virtual shopping mall would be a virtual brokerage house.

The type of scenario described by the OpenMLS system is readily amenable to the type of inexact searches suited to CBR. For instance, in a variation of the Rachmann CBR system, the indexed data from several real estate agents in a locality could be searched in order to determine a property valuation (Cunningham et al. 1994). The following example marked up in RELML is a slightly modified case taken from the property evaluation domain illustrated in Hanney's work (Hanney 1996).

[5] Microsoft Corp. (1998) Improving the Online Shopping Experience with XML http://www.microsoft.com/xml/scenario/junglee.asp

The RELML is a proposed standard which is not comprehensive enough to fully capture the details of residential real estate. However, it does give an indication of the benefits of an XML based industry standard mark up language and it certainly could be used in a case based search. The following illustrates a Dublin property marked up using pared down RELML:

```
<?XML version='1.0'?>
<RESIDENTIAL-LISTING VERSION='A1'>
<REMARKS> </REMARKS>
<GENERAL>
<IMAGE FORMAT='JPEG' WIDTH='150' HEIGHT='150'
SRC='http://www.hayesrealty.ie/search/homes/7466.jpg'/>
<TYPE>SINGLE-FAMILY</TYPE>
<PRICE>41500</PRICE>
<AGE UNITS='YEARS'>5</AGE>
<LOCATION COUNTRY='IRE' COUNTY='Dublin'>
<ADDRESS>127 Cabra road</ADDRESS>
<CITY>Dublin</CITY>
<ZIP>7</ZIP>
</LOCATION>
<STRUCTURE>
<NUM-BEDS>2</NUM-BEDS>
<NUM-BATHS>2</NUM-BATHS>
<BUILDING-AREA UNITS='SQ-FEET'>1100</BUILDING-AREA>
</STRUCTURE>
<DATES>
<LISTING-DATE>11/1/99</LISTING-DATE>
<LAST-MODIFIED>11/1/99</LAST-MODIFIED>
</DATES>
<LAND-AREA UNITS='ACRES'>0.75</LAND-AREA>
</GENERAL>
</RESIDENTIAL-LISTING>
</XML>
```

Fig. 3. A Dublin property marked up in basic RELML.

The case marked up in figure 3 gives no indication of what tags contribute to it being a case – what are its features, weighting information, which features may be constraints, what types are permissible as feature values etc. A CBR application processing this document would have to be hard coded with this information, and this coding would have to be changed every time a new feature is added or a constraint imposed, for example. This becomes a serious drawback when a product range changes or is updated.

What is required is an XML methodology that can provide a standard CBR view of data already marked up with user defined tags. This would provide searchable data to any CBR application that recognises the standard, and at the same time maintain the structural and descriptive integrity of the user-defined tags. To understand how this can be achieved we will briefly introduce the concept of XML namespaces and schemas.

5 Namespaces and Schemas

The namespace facility is an advanced feature of XML, outlined in a W3C recommendation as of January 1999.[6] Namespaces allow developers to uniquely qualify element names and relationships to avoid name collisions on elements that have the same name but are defined in different vocabularies. They allow tags from multiple name spaces to be mixed, which is essential if data is coming from multiple sources. Namespaces in XML are identified by a URI (Universal Resource Identifier) which allows each namespace to be universally unique. Every namespace is associated with a user defined prefix which allows the tags from each namespace to be distinguished even though they may in fact have the same name.

For example, an online bookstore may define the `<TITLE>` tag to mean the name of a book, contained only within the `<BOOK>` element. In a mailing list of customers, however, the `<NAME>` tag might indicate a person's position, for instance: `<TITLE>President</TITLE>`. Namespaces help define this distinction clearly.

```
<booksbought xmlns:bks="http://www.bookstore.com"
             xmlns:cst="http://www.bookstore.com/customerlist">
<book><bks:title>Fidelity in a Nutshell<bks:title><book>
<cst:name>W J Clinton<cst:title>President</cst:title></cst:name>
</booksbought>
```

Fig. 4. In this example the prefixes `cst` and `bks` denote the tag sets associated with the customerlist and book stock schemas respectively.

What is most important from the point of view of standardising a case based vocabulary is that namespaces allow us to define a unique case namespace that can be referenced by anyone wishing to mark up data in case format. Furthermore, namespaces will allow feature terms to be defined from several standard vocabularies (DTDs).

In section 5 we will present a case namespace which can be used to provide a standard case view of XML data. Before this however we wish to briefly discuss DTDs and Schemas, and the benefits adopting the latter will bring to case based data processing.

---

[6] World Wide Web Consortium 14-January-1999 Namespaces in XML, http://www.w3.org/TR/1999/REC-xml-names-19990114/

5.1 Schemas

In section 3 we acknowledged the insufficiency of our earlier implementation of a case base mark-up language. This was mainly due to two factors. Firstly, our implementation made no provision for dealing with data already marked up according to an existing DTD. Secondly, the DTD documents required by our implementation were not adequate to the task of describing case data. We began to view the caseStruct.xml document as a supplement to the case Base DTD, but a supplement that had no logical connection to the CaseBase within the standard document architecture outlined in XML 1.0[7] Its use would be limited to applications that were hard coded to recognise the CBML document structure. A better solution would have beeen to amalgamate the descriptive properties of the casestruct.xml and the DTD of the Case Base. However, the classical XML DTD is not descriptive enough to contain this additional information.

In fact, the legacy of a DTD as a descriptive syntax for document interchange makes it ill suited to the demands of data interchange (Boumphrey et al. 1998). Whereas document interchange is concerned mainly with document structure and the hierarchy of its elements, data interchange has more rigorous requirements such as an ability to constrain data types, provide easy extensions and inheritance facilities. Moreover, DTDs are unable to express the data relationships inherent in a relational database, nor do they provide support for the new W3C recommendation on namespaces.

It is unsurprising therefore that work has begun on finding an alternative to the DTD that has the descriptive powers required to adequately mark-up data. This alternative is called an XML schema and currently four such proposals are before the W3 consortium. Schema semantics constitute a superset of those provided by XML DTDs and are designed specifically for data interchange. Unlike the DTD, a schema document is itself an XML document with a mechanism somewhat analogous to but more expressive than a DTD for constraining document structure.

To date, the W3C Schema working group have published a requirements document for XML schemas and plan to deliver working drafts and proposed recommendations later this year [8]. Despite the immaturity of the Schema recommendation, we have decided in the following section to use one of the proposal notes (XML-Data)[9] as a basis for designing a case namespace[10]. While the details of the schema syntax are expected to change, the principles elucidated in the proposal note soundly reflect the issues surrounding XML data mark-up.

---

[7] W3 Consortium 10-February-1998. Extensible Mark-Up Language (XML) 1.0 http://www.w3.org/TR/REC-xml

[8] W3C XML Activity page http://www.w3.org/XML/Activity.html

[9] W3C Note 05 Jan 1998, XML-Data, http://www.w3.org/TR/1998/NOTE-XML-data/

[10] Microsoft Internet Explorer 5 provides support for a schema and data typing based on the XML-Data proposal note.

6 A Case Namespace

The XML namespace mechanism allows us to create a unique set of structured XML tags that can be referenced from within any XML document. By providing an agreed case namespace, we are essentially allowing any developer mark-up data in a standard case format. Data already marked up according to an existing DTD or schema can also be converted to case data by using the namespace facility to mix tags from different domains. We are attempting in this way to present the idea of CBR as a medium (Kitano & Shimazu 1995).

Figure 3 illustrates the document architecture for a case based view of real estate data using the namespace and schema mechanism. The real estate case schema is essentially a real estate case template created from the case namespace and the real estate namespace or tag set. Another way of visualising this is by imagining that the case namespace provides the pieces for a case wrapper around data tagged according to another namespace. The makeup of this template is determined by a domain expert using the case namespace and the real estate namespace. This is not simply a question of picking and mixing at will. Each namespace has rules that need to be complied with. For instance, within the case namespace the `case` element must contain a `featurelist` element and a `solution` element. Once the Real Estate Case Schema has been determined, an XML case base compliant with this can be created on the fly using technology such as active server pages.

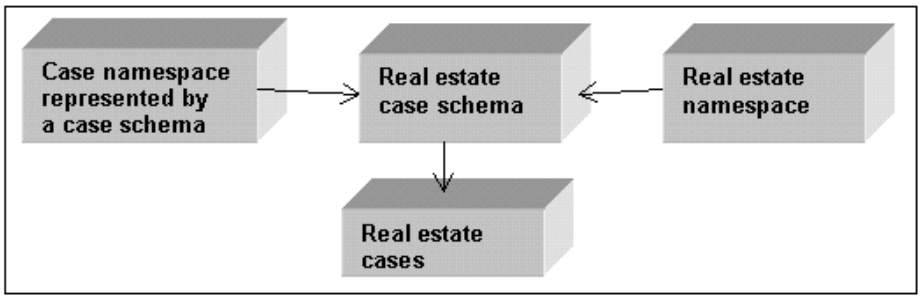

Fig. 5. The document architecture for a case-base marked up using a Real Estate namespace and a case namespace.

We have decided to use the well known travel domain[11] as a basis for comparison with CBML version 1.0. The small size of each case also allows us also demonstrate our concept with brevity. Figure 6 illustrates a simple case base in XML containing one case marked-up using the case namespace and a hypothetical travel agent namespace. Elements prefixed by the letter `c` belong to the case namespace. Since the travel agent namespace is declared first, it becomes the default namespace in this document and non-prefixed elements are understood to belong to this domain.

[11] Decision support in a travel agency, by Mario Lenz, GMD-FIRST, available at http://wwwagr.informatik.uni-kl.de/~bergmann/casuel/casebases.html

```
<?xml version='1.0'?>
<holidaycases xmlns:t="http://www.travelagent.com/travel"
                    xmlns:c="x-schema:caseschema.xml">
  <c:cases>
       <c:case caseid="1">
       <c:featurelist>
              <c:feature>
                     <holidayType>Bathing</holidayType>
              </c:feature>
              <c:feature><price>2498</price></c:feature>
              <c:feature>
                     <numberofPersons>2</numberofPersons>
              </c:feature>
              <c:feature><region>Egypt</region></c:feature>
              <c:feature><transport>Plane</Transport></c:feature>
              <c:feature><duration>14</duration></c:feature>
              <c:feature><season>April</season></c:feature>
              <c:feature><accomodation>2</accomodation></c:feature>
       <c:featurelist>
              <c:solution>
                     <hotel>Hotel White House, Egypt</hotel>
              </c:solution>
       </c:case>
  </c:cases>
  </holidaycases>
```

Fig. 6. A Case marked up using a case schema and a holiday schema

```
1.  <Schema name ="caseschema" xmlns="urn:schema-microsoft-com:xml-data"
2.  xmlns : dt="urn:schema-microsoft-com:datatypes">
   --
3.  <ElementType name="feature" content = "mixed" model="open"
    order="many">
4.  <AttributeType name="weight" required="no" dt:type="int"
    model="closed"/>
5.  <AttributeType name="constraint" required="no" dt:type="boolean"
    default="0"/>
6.  </ElementType>
7.  <ElementType name="featurelist" content="eltOnly" model="open"
    order=many">
8.  <element type="feature"/>
9.  </ElementType>
10. <ElementType name="solution" content="mixed" model="open"/>
11. <ElementType name="case" content="eltonly" order="one">
12. <AttributeType name="caseid" required="yes" dt:types="integer"
    model="closed"/>
13. <element type="featurelist"/>
14. <element type="solution"/>
15. </ElementType>
16. <ElementTye name="cases" content ="eltonly" model ="open"
    order="many">
17. <element type="case"/>
18. </ElementType>
   --
19. </Schema>
```

Fig. 7. An excerpt from a Case Schema based on XML-Data schema proposal.

The case namespace referenced in the `holidaycases` tag in figure 6 is delimited by the simple case schema shown in figure 7. This schema is easily extended and includes a facility for data-typing .

A full explanation of the syntax of the schema is outside the scope of this paper. However, a few points will make it a little clearer. The schema is an XML document itself, unlike the DTD which is defined according to EBNF notation. The `ElementType` tag (I will use *tag* in place of *element* here for purposes of clarity) defines the features that can appear in an XML document based on this particular schema. The `ElementType` tag contains a list of the elements permissible within the parent element. Constraints can be placed on these as to whether their presence is optional, their number and their type. For example, the `Featurelist` tag is defined in line 7 and it is allowed contain many Feature tags (line 8). Likewise the `AttributeType` defines an attribute associated with the element defined by an `ElementType` tag. For example, in lines 4-5 of figure 7 there are optional weight and constraint attributes associated with the feature element defined in line 3.

7 Conclusions

This paper presented the idea of a standard integrated CBR *view* of a company's information system. The ability of XML to integrate with relational database systems makes it a suitable candidate to represent this view. We have looked at two CBR systems that store cases in a database and use XML as a case representation format. These systems make use of the XML facility to create a custom tag set for each domain of use. We explain that since good DTD creation is not an easy task, and domain specific vocabularies are emerging it makes sense to find a standard way of representing case data without violating the syntax of the domain data. As an example, we look at an industry initiative in the real estate domain that uses a standard XML vocabulary, and suggest a role for CBR in such a scenario. We review CBML, an initial implementation of a standard case representation language using XML, and find it lacking for a number of reasons. Its syntax subsumes that of the domain data completely and its document architecture can only be understood by applications that have been designed to handle it. Our research into a solution has led us to the conclude that the current DTD model is inappropriate for the more rigorous requirements of data (as opposed to document) description. We then introduce two new initiatives stemming from the XML project - namespaces and schemas, which when used together allow us to create a powerful descriptive model for data which can be used as an alternative to the DTD. Arising from this we present the idea of a case namespace represented by a powerful case schema. The examples we present are taken from the holiday domain.

Further work will need to be done to refine our case namespace and explore its application in the realm of internet commerce, particularly in the area of internet brokerage. Schemas offer the advantage of easy extensibility, inheritance and data-typing support - none of which can be achieved with the current DTD model. The goal is to develop an XML case schema that fully exploits these features.

References

[1] Boumphrey, F. et al. (1998) XML Applications, Wrox Press, pgs. 97 -130

[2] Cunningham P., Finn. D., Slattery, S. (1994) Knowledge Engineering requirements in Derivational Analogy in Topics in Case Based Reasoning, Lecture notes in Artificial Intelligence, S. Wess, K-D Althoff, M.M Richter eds., pp234-245, Springer Verlag, 1994

[3] Doyle, M., Ferrario, M.A, Hayes, C., Cunningham, P., Smyth, B. (1998) CBR Net: Smart Technology Over a Network. TCD Technical Report TCD-CS-1998-07 - available at http://www.cs.tcd.ie/publications/tech-reports/tr-index.98.html

[4] Gentner, D., and Forbus, K. D. 1991. MAC/FAC: A model of similarity based access and mapping. In Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society. Northvale, NJ: Erlbaum

[5] Gardingen D., Watson I. (1998). A Web based Case-Based Reasoning System for HVAC Sales Support. Proceedings of British Expert Systems conference 1998.

[6] Hanney, K 1996. Learning Adaptation Rules From Cases. MSc. Thesis. Computer Science Department, Trinity College Dublin.

[7] Hayes C., Cunningham P., Doyle M. (1998) Distributed CBR using XML in proceedings of the Workshop: Intelligent Systems and Electronic Commerce, Bremen, September 15-17 1998. Also available as TCD technical report TCD-CS-1998-06
http://www.cs.tcd.ie/publications/tech-reports/tr-index.98.html

[8] INRECA consortium.(1994). Casuel: A Common Case Representation Language, available at http://wwwagr.informatik.uni-kl.de/~bergmann/casuel/CASUEL_toc2.04.fm.html

[9] Kitano, H. & Shimazu, H. (1996) The Experience Sharing Architecture: A Case Study in Corporate-Wide Case-Based Software Quality Control. In Case-Based Reasoning: Experiences, Lessons & Future Directions. Leake, D.B. (Ed.) pp 235-268. AAAI Press/The MIT Press Menlo Park, Ca, US.

[10] Shimazu, H. (1998). Textual Case Based Reasoning using XML on the World-wide Web in Advances in Case Based Reasoning, proceedings of 4t^{h} European workshop on CBR (EWCBR),
Springer Verlag LNAI

[11] Wilke, W., Lenz, M., Wess, S. (1998). Intelligent Sales Support with CBR. In Case-Based Reasoning Technology: from foundations to applications. Lenz, M., Bartsch-Sporl, B., Burkhard. H-D & Wess, S. (Eds.). Lecture Notes in AI#1400 91-113. Springer-Verlag, Berlin.