TOPME: AN XML-BASED CLIENT-SERVER FRONT-END FOR THE DISTRIBUTED MANAGEMENT OF CLINICAL PROTOCOLS FOR TOPS

Richard Jones, Kudakwashe Dube and Wu Bing
Department of Computer Science,
(Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland.
Emails: richard.jones@student.dit.ie, {kudakwashe.dube, bing.wu}@dit.ie)

ABSTRACT

This paper presents a distributed XML-based client-server front-end called the clinical Test-Ordering Protocol Management Environment (TOPME). TOPME presents to clinicians, in an easy-to-use way, the complex clinical protocol management functionality of TOPS, a clinical Test-Ordering Protocol System.

KEYWORDS: XML, user interface, clinical guidelines and protocols, ECA rule

INTRODUCTION

At a little over five years old, the eXtensible Mark-up Language (XML) [1] has become a leading standard for the storage, presentation and transport of data. Despite XML being a language and platform independent technology, huge amounts of data are still, and may continue to be, stored and processed on language and platform dependent systems. XML could be seen as a means for the integration of heterogeneous systems. TOPS [2], a clinical Test-Ordering Protocol System, is an application that was developed for the specification, storage, execution, querying and manipulation of clinical guidelines and protocol systems using the Event-Condition-Action (ECA) rule paradigm. TOPS currently requires a distributed front-end, that should present to the user the complex TOPS functionality in a simple and user-friendly way. TOPME addresses this requirement. This paper introduces TOPME a distributed XML based client-server application for TOPS. The rest of this paper is organised as follows. Section 2 outlines the background to the TOPS system. Section 3 discusses related work. Section 4 identifies the need for TOPME. Section 5 introduces the basic concepts relevant to TOPME. Section 6 introduces PLANX, the XML implementation of Protocol specification *LAN*guage, (PLAN). Section 7 presents the TOPME architecture. Section 8 presents the management of TOPS objects through TOPME. Section 9 outlines future work related to TOPME. Section 10 concludes and discusses the TOPME application.

BACKGROUND

The TOPS Project is an on-going research project within the Department of Computer Science at the Dublin Institute of Technology. The aim in TOPS is to investigate the problem of supporting clinical guidelines and protocols through the application of IT technologies. A major outcomes of the TOPS Project are a framework, concepts, methods and a mechanism for clinical protocol management. The proof-of-concepts system, TOPS, has been developed to address the need for computerised support for clinical guideline and protocol management. Clinical guidelines and protocols can be defined as "systematically developed statements to assist practitioners and patient decisions about appropriate healthcare for specific clinical circumstances"[3]. The class of clinical guidelines/protocols that are focused on within this project are protocols for clinical laboratory test ordering for the long-running management of a patient's condition. The specific technology that TOPS is based upon is the ECA paradigm as defined in active databases [4, 5]. TOPS is a prototype system that implements these outcomes using PLAN [6], which is a high level generic language for defining or specifying protocols using the ECA rule paradigm.

RELATED WORK

Attempts to reduce costs, practice variation and inappropriate resource utilisation in healthcare have led to the formalisation of medical domain information and knowledge, acquired through experience and research, to create clinical guidelines and protocols with the aim of incorporating them into daily practice [3]. Research, development and practice on computerised clinical guidelines have led to the development of structured formal guideline methods and mechanism, which include EON [7], Asgaard/Asbru [8], GLIF [9] and our own work, TOPS [2]. Among the structured text-based approaches that use HTML and XML and the Internet, the most interesting projects, from the point of view of our work, is the Guideline Element Model (GEM) [10]. GEM focused on the definition of clinical protocols and guidelines in XML using the technology of Document Type Definitions (DTD). The problem of the integration of legacy healthcare information systems with clinical guideline and protocol and electronic healthcare systems poses a major challenge to which XML technology and web services can play a significant role. The work presented in this paper gets its inspiration from GEM, which is an emerging standard as a guideline XML specification language [10].

PROBLEM DEFINITION

XML has now become an important standard format for storage, dissemination and presentation of data due to its platform and language independent nature. Using XML to wrap the existing TOPS application would give other systems access to TOPS functionality. Utilising a web services model would give TOPS a new distributed environment for management of clinical guidelines and protocols. Currently, TOPS does not have a graphical user interface. The clinician's interaction with the system is through a command line interface. TOPME presents to the user TOPS functionality in an easy to view fashion.

BASIC CONCEPTS

Before proceeding, basic concepts should be defined. The *eXtensible Mark-up Language* (XML)[1] is a simple, flexible data format derived from Standard Generalised Mark-up Language (SGML ISO 8879) and introduced to meet the challenges of large-scale electronic publishing. XML allows documents authors to create tags that accurately define the data they represent. XML was. The definition of PLAN in XML will be defined using *XML schemas*. The purpose of XML Schemas (XSD) [11] is to define and describe a class of XML documents by using constructs to constrain and document the semantics, usage and relationships of their constituent parts: data-types, elements and their content, attributes and their values, entities and their contents and notations. Schema constructs may also provide for the specification of implicit information such as default values. Schemas document their own meaning, usage, and function. Thus, the XML schema language can be used to define, describe and catalogue XML vocabularies for classes of XML documents. XML schemas are regarded as the next logical step forward from Document Type Definition (DTD). TOPME will use *web services,* such as SOAP, WSDL, UDDI and HTTP, to transport information via the Internet. A web service is a software system identified by a *Uniform Resource Identifier* (URI), whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by the *Internet protocols* such as HTTP [12]. XML, XML Schema Language and Web Services provide the framework for the TOPME application.

PLANX: XML VERSION OF PLAN

```
33. <dynamic-rule> ::= <rule-header>,
[<description>,] <event_spec>,
<condition_spec>, <action_spec>;
34. <event_spec> ::= On: <event>
( [<parameter_list>] )
35. <condition_spec> ::= IF: <condition> |
<condition> [AND | OR] <condition_spec>
36. <action_spec> :: DO: <action>
( [<parameter_list>] )
```

Figure 1: BNF/EBNF definition of a dynamic rule

PLANX is the XML implementation of PLAN. Figure 1 is an extract for the initial definition of PLAN in BNF/EBNF syntax, which illustrates an ECA rule. An ECA rule is referred to as a dynamic rule in TOPS. The dynamic rule specification is an extract from [13].

PLANX is defined using XML schemas, which specify the names of all the attributes and their data-type [14]. Figure 2 illustrates an ECA rule schema corresponding to a dynamic rule in TOPS. Due to space limitation, the XML specifications of the event, condition and action components of the ECA rule are not presented in Figure 2.An ECA rule defined in PLANX consists of the following elements: the name of the rule, the description associated with the rule, the creator identification number. The ECA rule also consists of an ECA event, which describes an occurrence of something defined within a particular clinical domain such as the arrival of new results, the check-in of a patient and patient state changes.

```
<xsd:complexType
name="eca_rule"><xsd:sequence><xsd:element
name="name" type="xsd:string"/><xsd:element
name="description"
type="xsd:string"/><xsd:element
name="creator_ID" type="xsd:long"/><xsd:element
name="ecaEvent">
<event specification></xsd:element><xsd:element
name="ecaCondition">
<condition specification></xsd:element>
<xsd:element name="ecaAction">
<action specification></xsd:element>
</xsd:sequence> </xsd:complexType>
```

Figure 2: Schema definition of a dynamic rule

An ECA condition triggers the mechanism that leads to a decision on whether or not an appropriate clinical intervention is required; examples could be presence of a clinical test result that exceeds a specified threshold. An ECA action is a set of operations meaningful to the clinical domain such as informing a clinician and altering another rule.

TOPME ARCHITECTURE

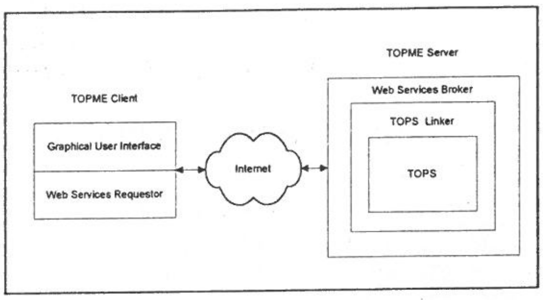

Figure 3: Overview of TOPME architecture

The TOPME architecture is based on the client-server model. The client application will reside in the clinician's desktop environment. The server application shall reside in the same host machine as the TOPS application. Figure 3 illustrates the conceptual architecture of TOPME. The TOPME client application consists of two components the graphical user interface, which shall handle the interaction between TOPS and the clinician. The aim of the GUI is to make interaction as easy as possible. The second component is the web service requestor this will communicate with the TOPME server and request services, these requests will be of the form of request and response. The TOPME server application can be considered to be a wrapper around TOPS. The web services broker will handle communication between the server and clients. The TOPS linker component controls the interaction between TOPME and TOPS. An overview of this interaction is given in the next section. The last component is TOPS, TOPS provides its API for TOPME to interact with and can be considered in as a black box for the purpose of this paper.

CREATION AND MANAGEMENT OF TOPS OBJECTS THROUGH TOPME

The strategy adopted in TOPME for managing TOPS objects is to i) create XML specifications of TOPS objects; ii) map the XML specification to a TOPS object; and iii) submitting the object to the TOPS engine; iv) communicate with TOPS for performing managements operations on the submitted object. The management of TOPS objects through TOPME will be briefly illustrated within the next two sections and demonstrated by using the TOPS Category object.

Figure 4 illustrates the XML document for the insertion of a new category into TOPS. A TOPS category includes the following information i) the category name, ii) a description for this category and iii) the category creator.

```
<?xml version="1.0">
<Category xmlns="http://www.example.com">
<name>Cancer</name>
<description>Cancer treatment</description>
<creator>Richard Jones</creator>
</Category>
```

Figure 4: XML definition of a Category

It is the responsibility of the TOPS Linker to convert the TOPME XML Category in a valid TOPS Category Object. Figure 5 shows this procedure. The TOPS Linker utilises the functionality of data binding. Once the TOPS category is created, TOPS looks after the insertion of the category into the TOPS system.

Once a clinical protocol is within TOPS and is instantiated (i.e. associated with a patient) its important to manage both its specification and its execution in relation to a patient. Illustrated in Figure 6 is the functionality for the management of a category. The term management in the context of TOPS and TOPME is a general term for specification, execution, manipulation and querying. Figure 6 illustrates the management APIs available in TOPS for TOPME.

The `add()` methods inserts the category. The `getNames()` method returns a list of the current categories, the retrieveId allows for the return of a category id, which is required when inserting protocols. The `retrieveCategories()` method allows for the return of a one-dimensional array of category names and their category id. There is also present a `delete()` method to remove a category. Also for category name, description, creator and identification there are getter and setter functions available. For all the other TOPS objects API's exist for the management of them.

```
// Create a new TOPS Category
private Category _category;
_category = new Category(xml.getName(),
xml.getDescription(), xml.getCreator());
```

Figure 5: Creation of TOPS Category from XML document

```
public void add()
public String[] getNames()
public long retrieveId(String categoryName)
public String[] retrieveCategories()
public void delete()
```

Figure 6: API for the management of a TOPS category

TOPME has been implemented in Java and uses Tomcat as a container for the web services. TOPME uses SOAP as its transport protocol and WSDL to define the web services. The information contained within the XML document is extracted using XML data binding and then used to create a TOPS object, which is submitted to TOPS together with an appropriate message for handling it.

FUTURE WORK

Areas of future work that needs further investigations are: i) exploring the possibility of using a purely XML storage format instead the current relational format; ii) improving security in web services; and iii) the area of the dynamic visualisation of an executing patient plan is a hugely significant area that could be utilised to give extremely in-depth view of the long-term treatment of a patient(s) condition.

SUMMARY AND CONCLUSION

This paper has presented TOPME as a distributed web service-based client-server for management of clinical protocols using the existing TOPS engine. The XML schema definition of PLANX, an XML based version of the ECA rule-based clinical protocol specification language, PLAN, was presented. The architecture of TOPME was also presented. The main components of TOPME were described. The strategy for the creation and management of the TOPS objects was presented. TOPME has been implemented using Java, SOAP, WSDL and Tomcat. The emerging XML technology promises to be of strategic significance in the management clinical guidelines and protocols.

References

1. XML Working Group, *Extensible Markup Language (XML).* http://www.w3c.org/xml. 2003, World Wide Web Consortium (W3C).
2. Wu, B. and K. Dube. *Applying the Event-Condition-Action Mechanism in Healthcare: A Computerised Clinical Test-Ordering Protocol System (TOPS).* in *3rd International Symposium on Cooperative Database Systems for Advanced Application (CODAS 2001).* 2001. Beijing, China: IEEE Computer Society, Los Alamitos, California.
3. Institute of Medicine (IOM), *Guidelines for clinical practice: from development to use.* Institute of Medicine (IOM), ed. M.J. Field and K.N. Lohr. 1992, Washington, D.C.: National Academy Press.
4. Widom, J. and S. Ceri, *Active Database Systems: Triggers and Rules for Advanced Database Processing*, ed. J. Widom and S. Ceri. 1996: Morgan Kaufmann.
5. Dittrich, K.R., S. Gatziu, and A. Geppert. *The Active Database Management System Manifesto: A Rulebase of ADBMS Features.* in *2nd Workshop on Rules in Databases (RIDS).* 1995. Athens, Greece: Springer.
6. Wu, B. and K. Dube. *PLAN: a Framework and Specification Language with an Event-Condition-Action (ECA) Mechanism for Clinical Test Request Protocols.* in *34th Hawaii International Conference on System Sciences (HICSS-34):: the Mini-Track in Information Technology in Healthcare.* 2001. Maui, Hawaii: IEEE Computer Society, Los Alamitos, California.
7. Musen, M.A., et al., *EON: A component-based approach to automation of protocol-directed therapy.* JAMIA, 1996. **3**(6): p. 367-88.
8. Shahar, Y., S. Miksch, and P. Johnson, *The Asgaard Project: A task-specific framework for the application and critiquing of time-oriented clinical guidelines.* Artificial Intelligence in Medicine, 1998. **14**: p. 29-51.
9. Ohno-Machado, L., et al., *The GuideLine Interchange Format: A Model for Representing Guidelines.* JAMIA, 1998. **5**(4): p. 357-372.
10. Shiffman, R.N., et al., *GEM: A proposal for a more comprehensive guideline document model using XML.* JAMIA., 2000. **7**: p. 488-498.
11. XML Schema Working Group, *XML Schema version 1.1,* http://www.w3.org/XML/Schema. 2003, World Wide Web Consortium (W3C).
12. Web Services Architecture Working Group, *Web Services Architecture Requirements.* http://www.w3.org/TR/wsa-reqs#RFC2396. 2002, World Wide Web Consortium.
13. Dube, K., *PLAN Language Syntax in Backus-Naur Form (Revised).* 2001, Computer Science Department, School of Computing, Dublin Institute of Technology: Dublin. p. 6.
14. Jones, R., *PLANX: PLAN goes XML.* 2002, Dublin Institute of Technology: Dublin. p. 6.

Proceedings of the 7th International Conference for Young Computer Scientists
August 8-10, 2003, Harbin, P.R. China

Exploring the New-generation Computing Technology

Edited by
Xiaofei Xu
Yadong Wang
Jiqing Han
Shouxu Jiang

International Academic Publishers
World Publishing Corporation

ISBN 7-5062-6084-0

9 787506 260848 >