XML REPOSITORY SEARCHER-BROWSER SUPPORTING FINE-GRAINED ACCESS CONTROL

R. Steele,* W. Gardner,* and T.S. Dillon*

Abstract

The widespread use of eXtensible Markup Language (XML) for data representation and exchange has led to increasing research interest in methods for XML content searching, presentation, and access control. This paper presents an XML repository searcher-browser application with a declarative role-based access control framework; the proposed access control model allows the definition of a fine-grained access policy to be applied to the underlay data content. The auto-generated user interface for the accessing of XML content provides a light-weight application that, while taking into account the access control policy, is also suitable for distributed mobile applications.

Key Words

Access control, XML, Web application, security, Xplorer, auto-generated user interface

1. Introduction

eXtensible Markup Language (XML) [1] is a widely accepted standard for structuring data by adding metadata to elements using self-descriptive tags. It is increasingly used for encoding all kinds of documents, such as product catalogues, digital libraries, and electronic health records (EHRs). The widespread success of XML used in information interchange has led to much interest and enhanced opportunities for its use in many aspect of software development. This growth has also led to increasing interest in issues regarding XML content searching and access control. However, current end-user applications lack an expressive mechanism for securely and effectively searching and browsing semi-structured content by using more than just keywords. Current keyword-oriented searching mechanisms suffer from limited query interface and limited customization to individual users, and often return inaccurate query results. This can result in a user having to filter through numerous mostly irrelevant results to find the right information.

In this paper, an access control model and an auto-generated multi-field search interface (called Xplorer) for XML repositories are presented. One of the critical issues related to search of XML content is the protection of sensitive information against unauthorized access. This requirement can be further defined into two levels: first is the access to information at the document level, and on a lower level is the access control to content segments within the same document. The requirement of access control is further complicated when the user perspective is also taken into consideration. The underlying properties of XML potentially support sophisticated and fine-grained differentiation of access capabilities between different users and contents. Xplorer aims to provide a framework for secure access to XML repositories by defining an access control vocabulary to encode data access constraints. The proposed vocabulary not only has powerful expressiveness to encode fine-grained access rights on different documents or parts of a document, but also dramatically reduces UI development and maintenance time, as the user interface and the access constraints are not hard-coded in the application, and this also enables flexible customization at a low cost.

The rest of this paper is organized as follows. Section 2 contains a brief survey of related work. The details of the role-based access control model are presented in Section 3, followed by discussion of the search-browse user interface engine in Section 4. An example implementation of the system for an electronic health record is shown in Section 5. Section 6 concludes the paper.

2. XML Data and Security

The appropriate presentation of information is an important aspect in any information system design, and is more so for the case of web/mobile-based applications [2]. To transform data into knowledge, information must be supported by an appropriate search and viewing mechanism that will allow the user to utilize them; this is true regardless of what data format the information is stored in. This is evidenced in the works of Odeview [3, 4] and Pesto

* Faculty of Information Technology, University of Technology, Sydney, NSW, Australia; e-mail: {rsteele, wgardner, tharam} @it.uts.edu.au

Recommended by Dr. Hai Jin
(paper no. 202-1861)

[5], where a graphical-based application was proposed for the browsing and querying of object-oriented databases. Other work in this area includes XQForms and QURSED [6, 7], but this work has looked at simpler schemas and at the creation of developer tools rather than auto generation of a user interface for end-user applications. On the other hand, BBQ [8] is one of the more recent works that has addressed the searching and accessing of XML-based data; a new underlying query language XMAS provides the support for the query of XML content. Although BBQ provided sufficient support in the querying of XML data, it lacks in providing an information presentation that is suitable for end users where the semantics of the data can be utilized.

Several works in the literature offer approaches to defining and enforcing access rights on XML documents. Kudo and Hada [9] proposed the XML Access Control Language (XACL). XACL is used to specify an object-subject-action-condition oriented access control policy. It supports flexible provisional authorization to a document based on whether certain conditions are met; for example, the subject is allowed access to confidential information, but the access must be logged. Bertino *et al.* [10, 11] defined the Author-X system as a suite of tools focusing on access-control enforcement and security administration for XML documents. Damiani *et al.* [12, 13] also specify a language for encoding access restrictions at the DTD/schema level or for individual XML documents. Gabillon and Bruno [14] implement access control by converting their "authorization sheet" to an XSLT document that can then extract a view of the accessible part of the corresponding XML document.

XACML (eXtensible Access Control Markup Language) [15] is an OASIS standard based on work including that of Kudo, Damiani, and Bertino. It standardizes an access request/response format, architecture of the policy enforcement framework, and so on, but it does not address deriving access control rules from the existing policy base. These approaches are based on XPath [16], XSL [17], and custom constructs that were developed to specify access conditions. Goel *et al.* [18] developed an XQuery-based [19] approach for deriving fine-grained access control rules from schema-level rules, document content, or rules on other documents. Miklau *et al.* [20] proposed cryptographic technique to ensure that published data are visible only to authorized users. More recently, a security views technique has been proposed in [21] that provides a grouped XML view consisting of only the information that the users (in that group) are authorized to access, but the proposed technique only supports DTDs.

3. Declarative Access Control Model

The Xplorer framework presented in this paper extends ideas from both XML access control and the research into search interface generation to provide an access control framework for viewer applications for semi-structured data. This is an extension to our previous work [22–24] to auto-generate interfaces from XML Schema, and it differs from other previous work in that it provides an access control model including the semantics of the access control privileges in terms of their interpretation by a generic semi-structured data viewer application. Clearly, there is a need for a simple and intuitive language to declaratively encode the semantics of an access control policy for XML content. Although other issues such as performance are also important in the design of XML-based application, our work here deals primarily with issues of scalability of access control deployment and performance in the aspect of end user data access efficiency.

The declarative role-based access control model (DRBACM) is defined by the scheme shown in Fig. 1. The model comprises three key components: `users`, `rolePrivileges`, and `privilegeSemantic`. Note that the key notion of this model is the ability to provide linking of relationships between privilege, role, and content all within a centralized yet flexible access model. Each component in the model provides features for defining the different aspects for a fine-grained access policy.

Firstly, the `users` component allows a role to be attached to a user, which forms the high-level access policy relationship between a specific user and his or her access rights. Note that each user can also have exceptions defined based on the `exceptionRule` component. The core access control policy, consisting of a set of declarative rules, is defined in the `rolePrivileges` component. It allows the definition of roles' privileges to various sensitivity levels and sections of the XML document and also includes conditional properties of the rules, and exceptions.

The sensitivity value is the only additional element that needs to be introduced into the XML data repository, which is stated in the `dataSensitivityFrom` and `dataSensitivityTo` elements to define the range of access a particular role may have on the data. In the XML data repository, the sensitivity level value of each element node is declared in the data schema as part of the appropriate element's attributes. A sensitivity level that is declared on a higher level of the element tree is cascaded down to the sub-elements. However, a local declaration on any sub-element can be use to override the cascaded value. By attaching the sensitivity level to the data schema it is possible to change access behaviour for all instance data of an element without making any changes to the instance documents—just a change to the schema is required. In addition, a new data element can easily be added to the data model, and the need to modify the access control policy in the security model every time when a new element is added is greatly reduced. The privilege classifications and their semantic relations, coupled with the use of data sensitivity level, provide a simple yet powerful framework for the definition of access control rules.

The access privilege is also defined in the access model scheme; however, this can be modified or completely replaced by a new set of privileges to suit the need of the situation. A default set of privileges as used in the implementation of Xplorer is shown in Table 1, and the semantic association of these privileges is shown in Fig. 2.

The Xplorer system provides an auto-generated multi-field search interface that relies on the schema of the XML data as well as the access control rules encoded using the

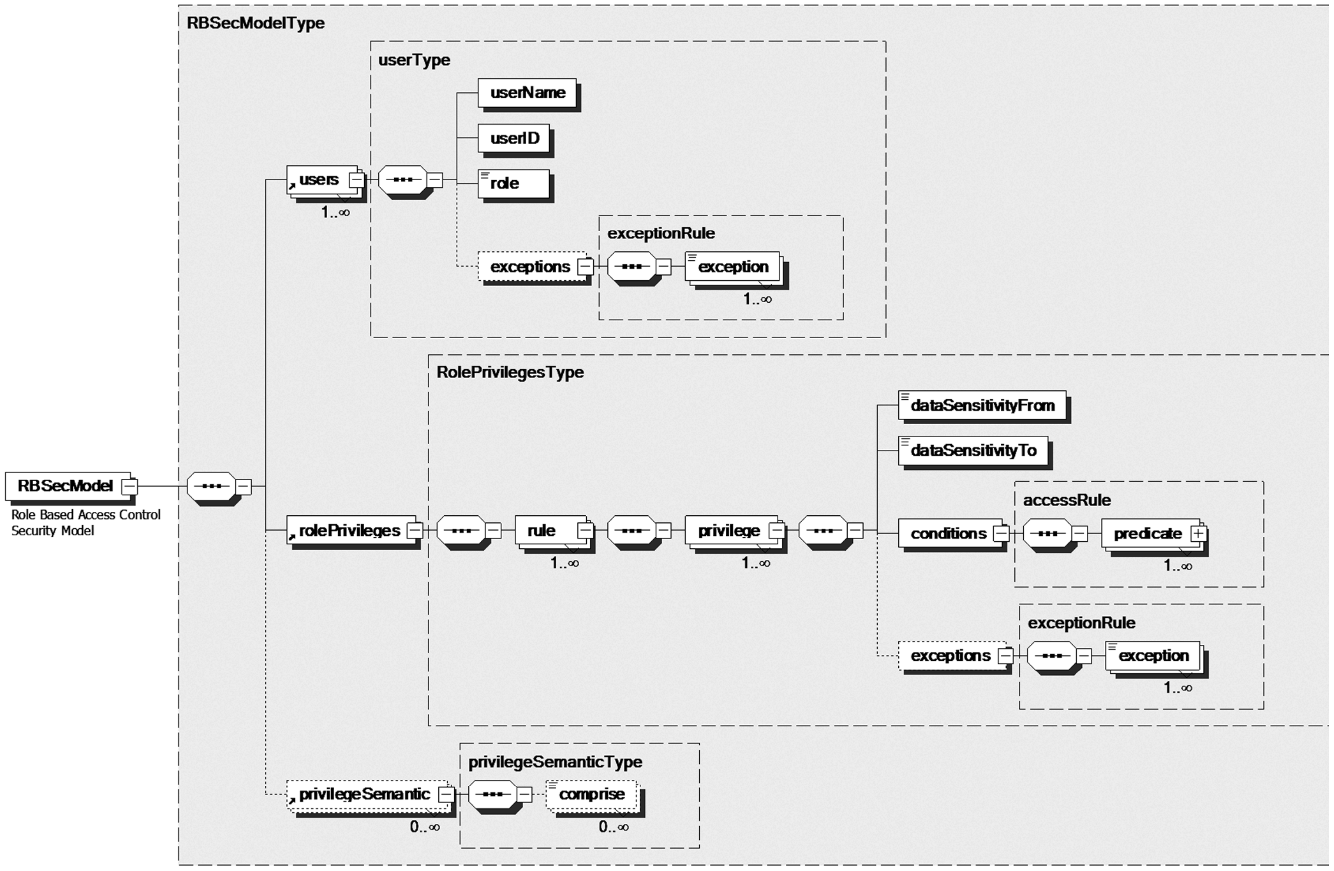


Figure 1. The Xplorer declarative role-based access control scheme model.

Table 1
Privilege Semantic in the DRBACM

Privilege	Meaning
Read	Only read the element content excluding sub-elements. This privilege can also be defined on selected attributes within an element.
Browse	Traverse through elements instance, read the elements' content and its child elements
Search	Specify which elements are searchable
Update	Refer to the ability to change the value of an attribute and text of an element
Add	Add new instance of an element/attribute
Delete	Delete an instance of an element/attribute
All	Allow full access to the whole document or part of a document
Deny	Deny access to an element/attribute

proposed constraints vocabulary. It allows searching of the XML repositories, utilizing the information in the XML semi-structured data tags without requiring end-users to use a complex XML query language. The query results will be intercepted by the Xplorer UI generator to provide a user-friendly and access-controlled view of the results. Because the vocabulary was defined to encode the access rules outside the raw instance data, this allows the access control policies, roles ,and rules to be altered easily without having to modify the data schema, data instances, or other business logic. Just by modifying the access control policy rules, new roles can be simply defined or existing roles altered with a high level of fine-grained expressive power. Once defined, the access control rules are dynamically taken into account by Xplorer (XQuery generator and the UI generator) to restrict access to the parts of the content documents the user (of a particular role) is allowed to see.

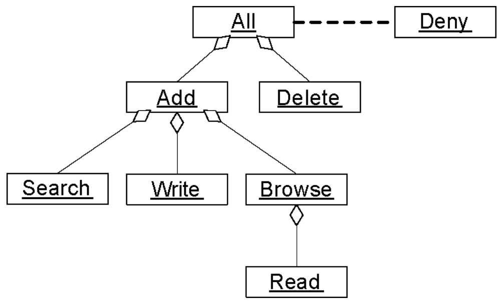


Figure 2. Role-based access control security model.

4. Interpreting Access Control Rules

For viewing of the XML repository, the security enforcement module of Xplorer will enforce access control rules either by refining the XQuery to run against the XML repository or by generating and applying XML transformations via dynamically generated XSLT from the access control rules. The Xplorer user interface will operate at any one time in one of the three modes: (1) Search, (2) View, and (3) Update (as shown in Fig. 3).

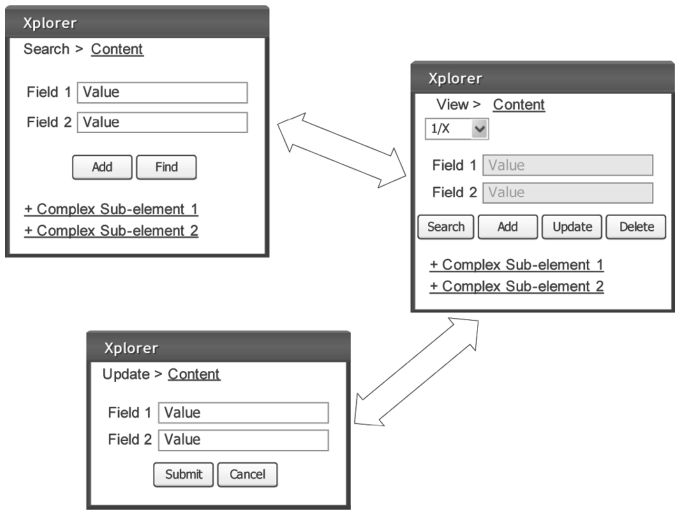


Figure 3. Operating modes of the Xplorer user interface.

The interpretation of the access policies occurs at three different but interrelated layers (Fig. 4). The access control enforcement module (of Xplorer) will apply the access control policy by first retrieving the access control rules that apply to the user, based on the role that is assigned to the user, follow by any user-specific exceptions, and then the actual privilege type of the elements that affects how the data are to be presented. There is a configuration parameter to instruct the framework to cache the access control for a specific period in order to avoid repetitive access control queries. The system will then replace any references to session variables (e.g., current user Id) with the actual values from the session object. If any of the access control rules apply to the requested data, then the system will refine the auto-generated query to ensure that the query requests only the data the user has access to.

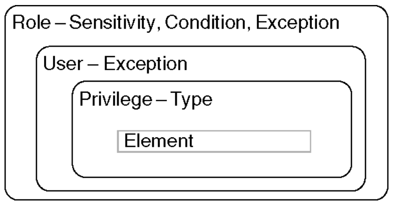


Figure 4. The way access control policies are interpreted by Xplorer.

In Search mode the user interface is generated from the XML data schema so as to only display search fields (i.e., the first/current level `simpleType` elements) corresponding to the elements from the schema for which the user has search privilege under the provided access control policy. In addition to the search fields, nested XML elements are shown as hyperlinks, and two buttons will be displayed, the Add button and the Find button. The Add button allows a new instance of the current element to be added, and the Find button submits the search request and will take the Xplorer into the View mode. The Search mode is the initial mode of the user interface if the element that the user gets into has more than one instance; otherwise the View mode will be initial mode. In this case, the user will get a view of content without the ability to search other records, and links to the appropriate sub-elements are also presented.

The View mode is generated from the XML instance elements returned by a search request. In this mode all the elements are read-only with values for `simpleType` displayed as text field and `complexType` elements displayed once again as hyperlinks so that the user can opt to drill-down into various parts of the currently selected element. Depending on the current user access policy, the View mode would provide a Search button to go back to the Search mode, an Add button that takes Xplorer to Update mode to add a sibling instance of the current element, an Update button to switch to the Update mode to edit the values of the current element, and a Delete button to delete the current instance element and its child elements.

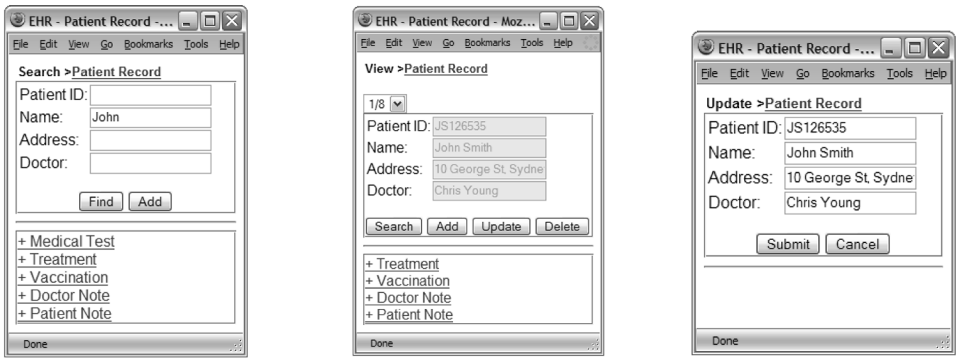
The Update mode is used to update values of an instance. This instance could be a new instance being added or an existing instance being modified. The interface is generated from the schema of the current element to be edited or created. In this mode all the elements to which the user has write privilege will be rendered as editable text boxes. Once the update is done (Submit clicked) the user is returned to the View mode.

5. Case Study: EHRs

Suppose that a hospital wants to impose the following security policy on an EHR system, with two roles needed in their security policy, namely Doctor and Patient:

- The doctor can access the records of all patients but only the `MedicalTest` results of his/her own patients, and a doctor is only allowed to view the `PatientNote`.
- The patient is only allowed to access all information on

```
<RolePrivileges>
  <rule RoleId=‘‘Doctor’’>
    <privilege PrivilegeType=‘‘All’’>
      <DataSensitivityFrom>1</DataSensitivityFrom>
      <DataSensitivityTo>4</DataSensitivityTo>
      <conditions>
        <predicate type=‘‘equal’’ scope=‘‘//MedicalTest’’>
          <parameter type=‘‘xpath’’>//PatientRecord/Doctor</parameter>
          <parameter type=‘‘variable’’>Server.GetValue(Session.UserId)</parameter>
        </predicate>
      </conditions>
      <exceptions>
        <exception type=‘‘Search’’>//PatientNotes</exception>
        <exception type=‘‘Browse’’>//PatientNotes</exception>
      </exceptions>
    </privilege>
  </rule>
  <rule RoleId=‘‘Patient’’>
    <privilege PrivilegeType=‘‘Browse’’>
      <DataSensitivityFrom>1</DataSensitivityFrom>
      <DataSensitivityTo>2</DataSensitivityTo>
      <conditions>
        <predicate type=‘‘equal’’ scope=‘‘//PatientRecord’’>
          <parameter type=‘‘xpath’’>//PatientRecord/PatientID</parameter>
          <parameter type=‘‘variable’’>Server.GetValue(Session.UserId)</parameter>
        </predicate>
      </conditions>
      <exceptions>
        <exception type=‘‘Add’’>//PatientNotes</exception>
      </exceptions>
    </privilege>
  </rule>
</RolePrivileges>
```


EHR - Patient Record -...
File Edit View Go Bookmarks Tools Help
Find
Add
+ Medical Test
+ Treatment
+ Vaccination
+ Doctor Note
+ Patient Note
Done
EHR - Patient Record - Moz...
1/8
Search
Add
Update
Delete
EHR - Patient Record -...
Submit
Cancel

Search >Patient Record
Patient ID:
Name: John
Address:
Doctor:

View >Patient Record
Patient ID: JS126535
Name: John Smith
Address: 10 George St, Sydne
Doctor: Chris Young

Update >Patient Record
Patient ID: JS126535
Name: John Smith
Address: 10 George St, Sydne
Doctor: Chris Young

(a) Search mode – for a Doctor (b) View mode – for a Doctor (c) Update – for a Doctor

Figure 5. Examples of the Xplorer user interface for the HER system.

his/her own record, except the section of `DoctorNote` and `MedicalTest.` A patient is also allowed to edit his or her own `PatientNote.`

The above constraints can be easily encoded using the proposed method by adding the data sensitivity level into the existing data model schema; the access policy would be defined as the example XML code show below. Three example user interfaces showing the different mode of Xplorer for the EHR base on this policy are shown in Fig. 5.

6. Conclusion

In this paper, we presented a declarative access control model for XML, and Xplorer, an engine to interpret access control rules and provide secure searching and browsing of XML repositories. First we defined an access control model, consisting of a set of privileges and an access control rule schema, which provides powerful expressiveness to encode access rules to semi-structured content. Second, we described the Xplorer engine that interprets the access control rules.

The XML browse-and-search technique is particularly well suited to mobile clients, as it can be used to produce a simple interface to incrementally browse the hierarchical data. The auto-generated interface saves on development effort and eases maintainability. With the proper architectural support, Xplorer can be implemented to provide distributed multilevel access control in an enterprise situation. Although large-sized data may be store centrally, the data content can be transmitted to the access point after the first level of access control policy has already been applied (e.g., from the central data source, to one hospital and/or department). At this stage, the internal (n level) access policy can then be applied, so that the data can only be accessed by the appropriate user.

Combining the auto-generated search interface framework with the role-based control policy, the Xplorer interface allows the user to search, view, and navigate through the XML repositories, by presenting simple type elements on the current level with the appropriate UI elements, while sub-levels are show as hyperlinks. This presents a generic technique to search and browse XML data that takes into account access control rules.

References

[1] W3c-Xml, Extensible markup language (xml), http://www.W3.Org/xml, 2004.

[2] W. Gardner, E. Chang, & T.S. Dillon, Analysis model of web user interface for web applications, *16th Int. Conf. on Software & Systems Engineering and Their Applications, ICSSEA 2003*, France, Paris, 2003, 250–256.

[3] R. Agrawal, N.H. Gehani, & J. Srinivasan, Odeview: The graphical interface to ode, *Int. Conf. on Management of Data*, Atlantic City, NJ, 1990, 34–43.

[4] S. Dar, N.H. Gehani, H.V. Jagadish, & J. Srinivasan, Queries in an object-oriented graphical interface, *Journal of Visual Languages and Computing*, *6*(1), 1995, 27–52.

[5] M. Carey, L. Haas, V. Maganty, & J. Williams, Pesto: An integrated query/browser for object databases, *Int. Conf. on Very Large Databases (VLDB)*, Mumbai, India, 1996, 203–214.

[6] M. Petropoulos, V. Vassalos, & Y. Papakonstantinou, Xml query forms (xqforms): Declarative specification of xml query interfaces, *Int. Conf. on World Wide Web*, Hong Kong, 2001, http://www10.org/cdrom/papers/335/index.html.

[7] Y. Papakonstantinou, M. Petropoulos, & V. Vassalos, Qursed: Querying and reporting semistructured data, *Int. Conf. on Management of Data*, Madison, WI, 2002.

[8] K.D. Munroe & Y. Papakonstantinou, Bbq: A visual interface for integrated browsing and querying of xml, *Int. Conf. on Very Large Databases (VLDB)*, Cairo, 2000.

[9] M. Kudo & S. Hada, Xml document security based on provisional authorization, *7th ACM Conf. on Computer and Communications Security (CCS)*, Athens, 2000, 87–96.

[10] E. Bertino, S. Castano, & E. Ferrari, Securing xml documents with author-x, *Internet Computing, IEEE*, *5*(3), 2001, 21–31.

[11] E. Bertino & E. Ferrari, Secure and selective dissemination of xml documents, *ACM Trans. on Information and System Security (TISSEC)*, *5*(3), 2002, 290–331.

[12] E. Damiani, P. Samarati, S. De Capitani Di Vimercati, & S. Paraboschi, Controlling access to xml documents, *Internet Computing, IEEE*, *5*(6), 2001, 18–28.

[13] E. Damiani, S.D.C.D. Vimercati, S. Paraboschi, & P. Samarati, A fine-grained access control system for xml documents, *ACM Trans. on Information and System Security (TISSEC)*, *5*(2), 2002, 169–202.

[14] A. Gabillon & E. Bruno, Regulating access to xml documents, *15th Annual Conf. on Database Security*, Niagara, Ontario, Canada, 2001, 299–314.

[15] Oasis, *Extensible access control markup language (xacml) version 1.0*, 2003 [cited 30 May 2004], http://www.oasis-open.org/committees/tc_home.php?wg_ abbrev=xacml.

[16] W3c-Xpath, Xml path language (xpath) version 1.0, http://www.W3.Org/tr/xpath, 1999.

[17] W3c-Xsl, Extensible stylesheet language (xsl), http://www.W3.Org/style/xsl, 2003.

[18] S.K. Goel, C. Clifton, & A. Rosenthal, Derived access control specification for xml, *Workshop on XML Security*, Fairfax, VA, 2003, 1–14.

[19] W3c-Xquery, Xquery 1.0: An xml query language, http://www.W3.Org/tr/xquery, 2004.

[20] G. Miklau & D. Suciu, Controlling access to published data using cryptography, *Proc. of the 29th VLDB Conf.*, Berlin, Germany, 2003.

[21] F. Wenfei, C. Chee-Yong, & G. Minos, Secure xml querying with security views, *SIGMOD*, Paris, 2004, 587–598.

[22] R. Steele, Y. Ventsov, & T. Dillon, Object-oriented database-based architecture for mobile enterprise applications, *IEEE ITCC04*, Las Vegas, 2004, 586.

[23] R. Steele, Y. Ventsov, & T. Dillon, Xml schema-based discovery and invocation of mobile services, *IEEE EEE04*, Taipei, 2004, 253–258.

[24] R. Steele & T. Dillon, Ontology driven system for mobile device access of electronic health records, *Proc. of 3rd Int. Conf. of Mobile Business 04*, New York, US, July 12–13, 2004.

Biographies

Robert Steele is the current Information Director of the ACM Special Interest Group on Mobility (ACM SIGMOBILE), is on the Steering Committee of the International Conference on Mobile Business (Steering Committee Co-Chair 2005–2006), is a general co-chair of the (IEEE-sponsored) 2005 International Conference on Mobile Business, and is a member of the editorial review board of the *International Journal of e-Business Research*. He is the recipient of four Australian Research Council grants in the areas of XML and security and of other grants for research excellence. His previous research led to the formation of a start-up company that has received in excess of $7 million of funding. He is the author of over 50 research articles and five patents.

William Gardner is currently a Ph.D. candidate at the Faculty of IT, University of Technology, Sydney. His work relates to developing models for the design of web application user interfaces including aspects of user security and access control constraints. He has previously worked as an application programmer and software architect in the development of usability metric software, and has published research articles that have appeared in international conference and journals.

Tharam S. Dillon published over 400 international journal and conference papers and has been asked to be program or general chair for several international conferences, including the IEEE Data Semantics Series, that have focused on the data semantics of e-commerce and multimedia systems. His work is widely cited. He was elected as a Fellow of the IEEE (USA) in 1998. He is editor-in-chief of three international journals, advisory editor of *IEEE Transactions on Neural Nets*, and is on the editorial board of several other international journals. He is also on the International Program Committee of several international conferences and has served on the Engineering Panels of the Research Grants Committee in Hong Kong. His previous work has also addressed large-scale systems and has been widely adopted in industry.