# XML Security in Certificate Management – XML Certificator

Joan Lu* and Nathan Cripps*

*School of Computing and Engineering, University of Huddersfield, UK*

**Abstract:** The trend of rapid growing use of XML format in data/document management system reveals that security measures should be urgently considered into next generation's data/document systems. This paper presents a new certificate management system developed on the basis of XML security mechanisms. The system is supported by the theories of XML security as well as Object oriented technology and database. Finally it has been successfully implemented in using C#, SQL, XML signature and XML encryption. An implementation metrics is evidently presented.

## INTRODUCTION

XML has rapidly become the de facto standard for document and data exchange since its recent birth in 1998 [1]. The extent of this growth is indicated by the growing areas of research into XML and its applications. This trend has pushed the need for suitable standards and specifications for information representation [2].

In lay of recent research concerning XML and security [3], this paper details the progress of a prototype implementing XML security technology in certificate management system.

The rapid and successful growth of XML has provided a new set of security problems particular because it has been used for data exchange and furthermore storage. Problems include:

1. The management of data in different formats
2. Traditional technologies do not support inter-document level encryption
3. SSL (Secure Sockets Layer) only secures data for the period of a handshake [4].

The objective of this research is to develop an XML based certificate system called XML Certificator which will model the content and security management of heterogeneous certificates. This could include scientific calibration certificates developed within the software community in order to authenticate the results of a calibration experiment. The benefits of security with certificates are as significant as with any other documents because they are the based on trust. The prototype will provide the means for editing, transforming, saving and loading certificates. XML technology enables the secure transmission of information at any level of a document. Together with the theory of security, XML can represent digital signatures for signatures that require validation and creation within a content management environment.

Provided in the next section is the methodology regarding the three key theories and technologies used. Requirements of the system are also mentioned here.

*Address correspondence to these authors at the School of Computing and Engineering, University of Huddersfield, UK;
E-mail: j.lu@hud.ac.uk; n.cripps@hud.ac.uk

## METHODOLOGY

### 1. Underpinning Theories

Three key theories are utilized within this project.

#### *a. XML Security Theory*

XML is a text based standard that is both human and machine readable. The aim is to standardize the exchange of data within a common framework and thus lessens both computation time and understandability. XML Security plays a small but important part among the vast field of XML. The world leading professional body W3C working groups have developed several specifications, i.e. XML signature, XML encryption/decryption, XML key management, underpinned by the theories of XML security [1].

It is based upon PKI (public key infrastructure) fundamentals which is heavily mathematically based and proven [3]. The basis hinges on keys of asymmetric or symmetric type. Two parties whom both require access to a secure document either use the same key (symmetric) or separate keys (asymmetric). Both types have their advantages and disadvantages [5] but the latter has received the greater attention [6, 7].

#### *b. Object Oriented (OO) Theories*

Object oriented theories achieve advancement over procedural programming [3]. The fundamental underlying aspect is the modeling of any concept as an object, whether it is abstract or real. The essence of OO modeling is to deliver three main principles, namely: encapsulation, polymorphism and inheritance.

1. Encapsulation The hiding of properties belonging to the class. Manipulation is restricted through specific methods only.
2. Polymorphism Many forms of a type and the ability to version methods with signatures.
3. Inheritance. Enables code reuse with the ability to inherit the properties and methods of an object.

#### *c. Database Theory*

The most commonly used type of database is the relational [8-10]. This basis of which all data is modeled as relations with common relationships used to link them together.

1874-947X/09 

Later version incorporate OO theory with an object-relational framework added on top of the relational layer. DDL (Data Definition Language) is used to model the relations and DML (Data Manipulation Language) provides for the manipulation of the data. Corresponding functions provided in Object Oriented Database management systems are ODL – Object Definition Language and OQL Object Query Language. The DML is based upon relational algebra. Theory of relationships is based upon two types of relational integrities:

1. Entity integrity: No primary key must be null.
2. Referential integrity: A Foreign key must be null or must match a primary key.

## 2. Technologies and Standards Involved

### *a. Technologies Involved*

#### ***XML Security Technologies***

Existing technologies include SSL (Secure Sockets Layer) and TLS (Transport Layer Security). Like XML security PKI provides the mathematical basis of their security. XML security, however, holds several higher cards in its hand:

1. XML security is portable because it can become part of the data [2]. SSL and TLS only secure data during transportation for length of a handshake [8].
2. XML security has a granular structure and there are no limits to the number of referrals to a document [11]. Security for a whole or part of a document is specified.
3. Non-XML documents may be secured.

Key languages and specifications include XML Encryption, XML Digital Signature and XML Key Management Systems. Together they endeavor to adopt the following three requirements [12]:

1. Authentication: The client or receiving party can be certain of the origin of a document. It is indisputable.
2. Data Integrity: Secured data is identical to the original data as a client can be assured there have been no alterations of the data during transmission over a network.
3. Confidentiality: Only authorized personnel can access the data. It remains unrecognizable to those who do not have permission to view the data.

**XML Encryption Syntax is listed in List 1**

```
<EncryptedData ID? Type?>
   <EncryptionMethod/>?
   <ds:KeyInfo>
      <EncryptedKey>?
      <AgreementMethod>?
      <ds:KeyName>?
      <ds:RetrievalMethod>?
      <ds:*>?                    ? = zero or one occurrence
   </ds/KeyInfo>?                + = one plus occurrences
   <CipherData>                  * = zero plus occurrences)
      <CipherValue>?
      <CipherReference URI?>?
   </CipherData>
   <EncryptionProperties/>?
</EncryptedData>
```

- <EncryptedData ID? Type ?> - The root element of XML Encryption. The ID attribute allows multiple encryption elements by providing an identifier for each instance. Type identifies the type of plaintext for example an image or an XML element.
- <CipherData> - A child of <EncryptedData ID? Type ?>. Responsible for storing the actual encrypted value.
- <CipherValue>? – The actual encrypted data represented as Base-64-encoded data.
- <EncryptionMethod/> - Identifies the encryption algorithm.
- <ds:KeyInfo> - Reused from the digital signature specification as described above.

<EncryptionProperties/> - For the inclusion of other properties and semantics

#### ***Object Oriented Technologies***

C# developed by Microsoft is a platform independent high level OO language. C# was unveiled in July 2000 and takes as a basis of the advantages of all previous OO languages [13]. Provided by Microsoft is a rapid application development model which sits within an infrastructure of a high-level abstraction of the operating system, namely the .NET framework. C# executes within a common runtime language runtime environment (not too dissimilar to Java) which is responsible for memory management, references, garbage collection, type checking, exception handling and compiling. A number of class libraries known as the FCL (Framework Class Library) supports OO functions. FCL base is the lowest level supporting basic classes such as input/output, security and threading. The next tier extends this basic class to support data management and manipulation including SQL and XML. Top tiers allow the creation of web applications and web services.

Due to the support of OO technology and XML security, C# presents itself as an outstanding candidate for the proposed prototype. XML standards and web related technologies are supported in .NET framework.

#### ***Database Technologies***

The language of use for both DDL and DML is SQL (Structures Query Language) based upon relational algebra. What are the benefits of using a relational database for the storage of XML data? Firstly, relational databases are the most commonly used and therefore have a great support structure in place. Secondly, the ability to retrieve and update documents or parts of them is relatively easier when storing XML documents as relations [5]. Several drawbacks have to be considered as a result:

1. The mapping of XML to SQL relations is computationally costly compared to storing XML in its native form.

**Table 1. .NET 2.0 Supported XML Technologies**

| Standard | Reference |
|---|---|
| XML 1.0 | http://www.w3.org/TR/2004/REC-xml/ |
| XML namespaces 1.1 | http://www.w3.org/TR/xml-names11/ |
| XML Schema 1.1 | http://www.w3.org/XML/Schema |
| DOM Level 1 and Level 2 Core | http://www.w3.org/TR/DOM-Level-2-Core/ |
| XPath 1.0 | http://www.w3.org/TR/xpath |
| XSLT 1.0 | http://www.w3.org/TR/xslt |
| SOAP 1.2 | http://www.w3.org/TR/soap12-part1/ |
| XML Signature 1.0 | http://www.w3.org/TR/Xmldsig-core/ |
| X-Query 1.0 | http://www.w3.org/TR/xml/query |
| Xml Encryption 1.0 | http://www.w3.org/TR/Xmlenc-core/ |


**Fig. (1).** Architecture of system.

2. Restrictions of relational database management systems enforce maximum character size upon its data types. A large XML document or element may be too large to store within a single field. Therefore there is the need to store the XML as a BLOB (Binary Large Object) or a CLOB (Character Large Object). This then leads to further problems such as the querying of a BLOB.

All three technologies partake in a three tier model as shown below in Fig. (**1**).

## Standards Involved

There are number of XML based standards supported by .NET are listed in Table **1**.

## SYSTEM DESIGN

Fig. (**1**) presents the architecture for the proposed system involving three technologies as mentioned above. The presentation layer involves the graphical user interface. Here the user may publish, edit and sign a certificate. The communication layer provides the link between the application and the relational database, which leads to the data storage layer used to store certificate and signature data.

Included within these layers the following technologies shall be used:

- OO technologies, i.e.
  - C#: Implementation of the system. Provides a high level OO language with XML support.
  - UML: Modeling the OO design which helps to lower development time.
- XML technologies, i.e.
  - XML signature, encryption/decryption: For security and transportation of certificate data.
  - Document Object Mode, i.e. DOM: Store XML in memory. Enables use of XPath to quickly navigate an XML document. Also allows for manipulation of the document.
  - XPath: To query and navigate an XML document.
  - XSLT: Transform XML to other formats such as HTML.
  - DTD: Structuring and modeling XML providing uniformity.
  - Schema: Structuring and modeling XML with multiple data types.
- SQL: i.e.
  - DDL and DML: Storing and manipulating relational data.

The model overview of the system is shown in Fig. (**2**). Tools available to the client allow for publishing, signing and editing an XML certificate. The database is provided by Microsoft SQL Server 2000 onwards and requires an SQLOLEDB connection from the client. Normalized relational versions of XML signatures and certificates will be stored within the server. The transformation engine converts certificates to XML, PDF or HTML for publishing. The client is provided with a graphical user interface (GUI) to utilize the facilities of X-Certificator. The possibility of incorporating access control services and a key manager has been recognized as indicated in the figure. Access control will enforce limits of use onto the client depending upon their

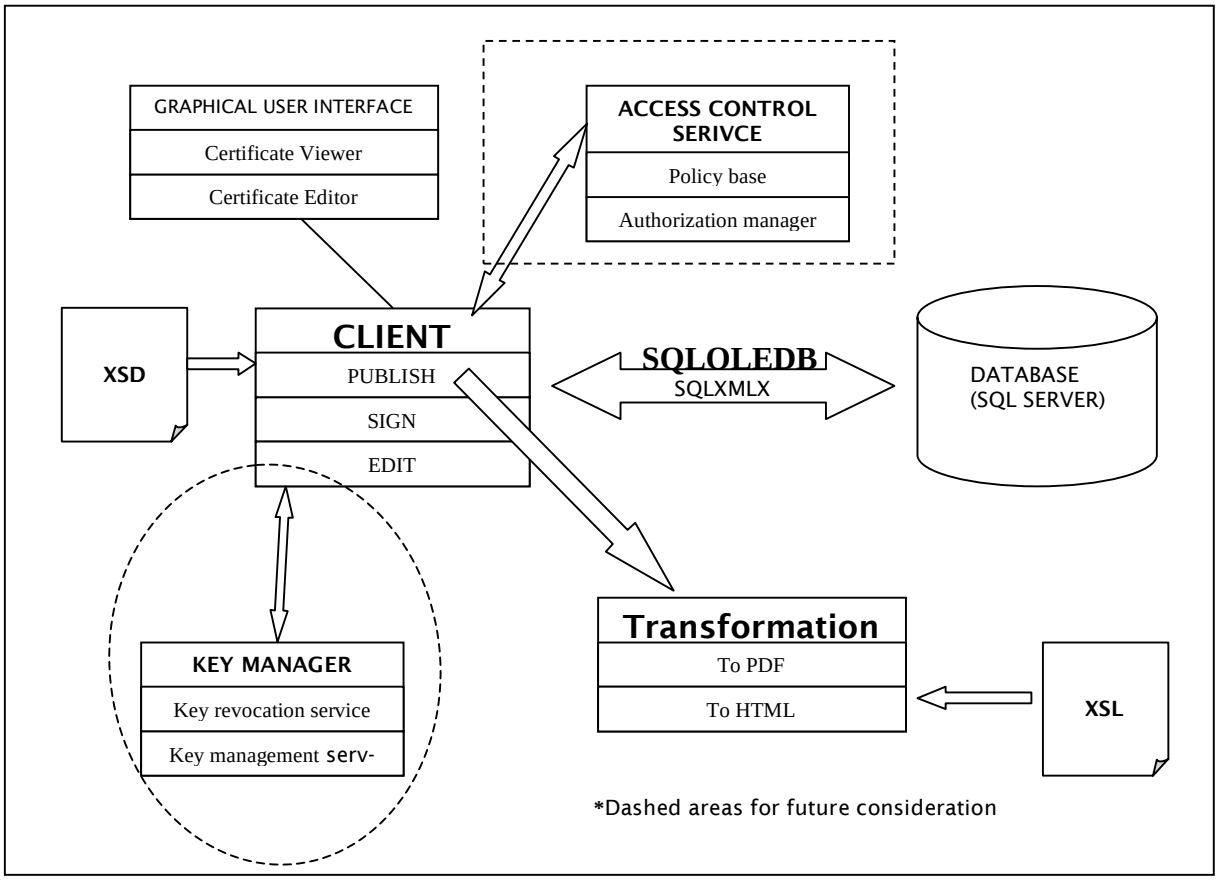


**Fig. (2).** Model overview.

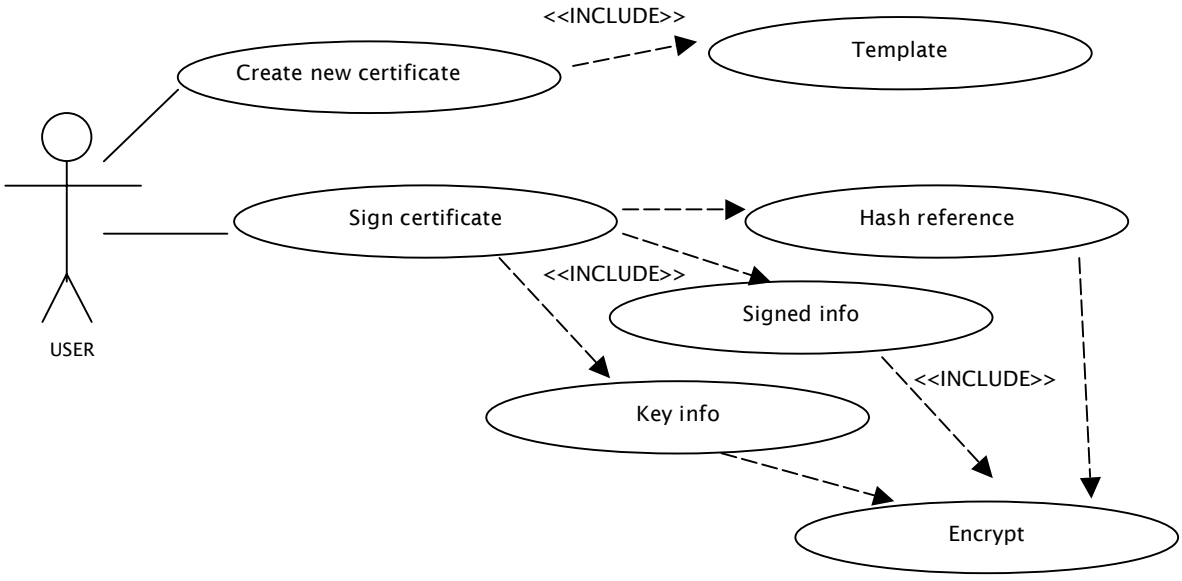


**Fig. (3).** Specification use case.

rights or attributes. A Key manager provides a service for linking keys with their respective owners. This is important for PKI technologies such as XML digital signature.

### Design Specification

Fig. (**3**) shown below, demonstrates the intended use case of the prototype. The user has various options available to them which are supplied by a graphical user interface (GUI).

### DESIGN GOALS:

- To use current security technologies for certificates.
- To use digital signatures for authentication.
- To hold data securely in XML format.
- To represent data in XML for maximum use of it's extensibility

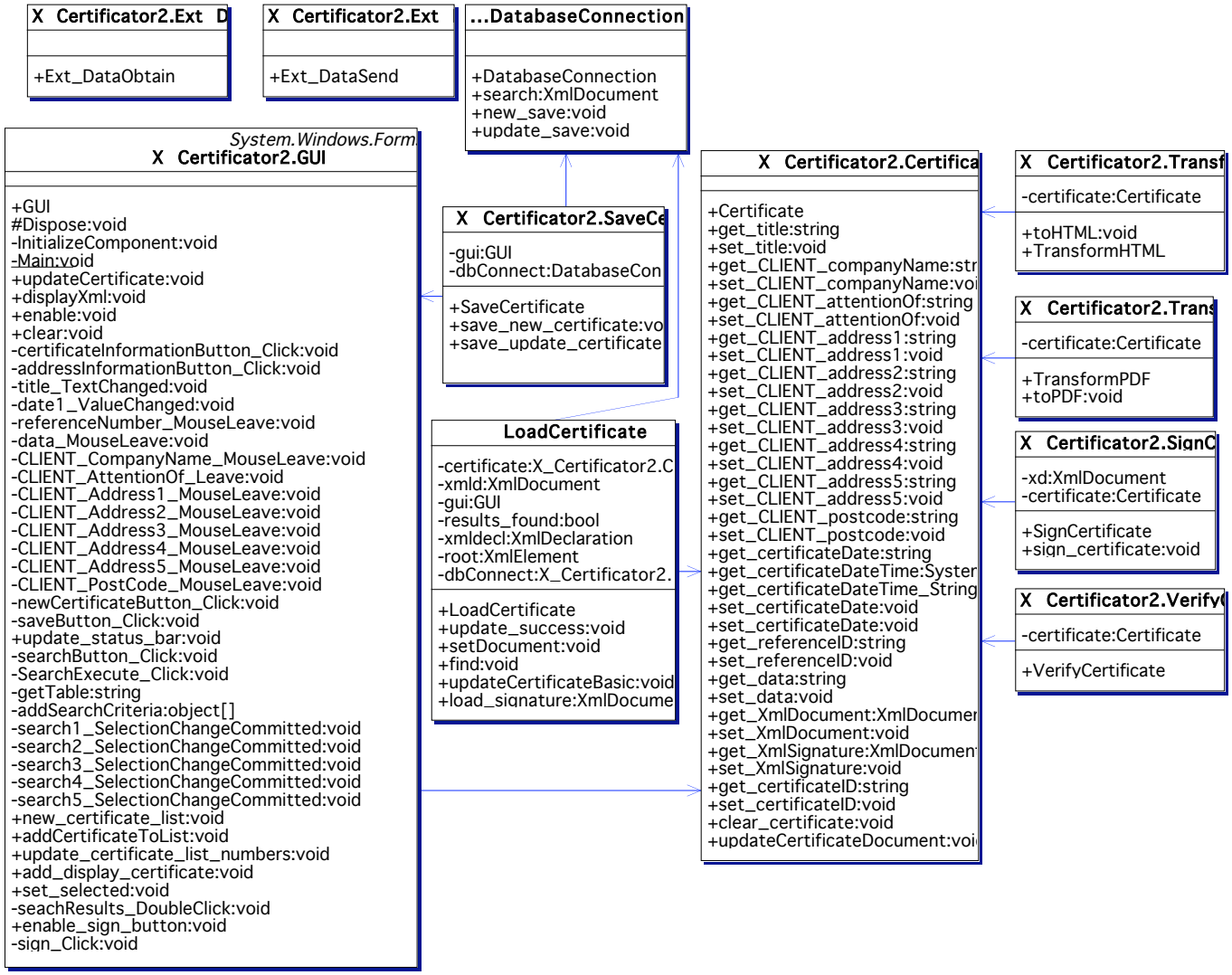


**Fig. (4).** Class diagram.

- To store and retrieve certificates from a database.
- To store and retrieve digital signature from a database.

## IMPLEMENTATION

### 1. UML Class Diagram

The main class as featured in Fig. (**4**) is the Certificate class. Here the class models basic details of a certificate. Should a certificate require greater detail the Certificate class may be inherited. The LoadCertificate class represents the bridge between the certificate object and connecting to the database. The SaveCertificate class performs a similar job except for the purpose of saving or updating a certificate. The GUI class holds all the user interface code and implements event methods to listen for the users requests.

Table **1** presents the metrics for the current implementation as provided by Borland® Together® ControlCenter™ 6.2. Number of operations (NOO) is the greatest for the GUI class which holds the user interface code. Certificate class provides 35 methods (NOM) and the majority of these are accessory methods. In total 2182 lines of code (LOC) have been written and the GUI class represents over half the total amount. As the number of lines (LOC) it suggests that the Certificate and GUI class are the most complex part of the program. This is indicated by the Cyclomatic complexity (CC) value.

The display XML method belongs to the GUI class. The XML Document as rendered in a DOM model is passed as input. It is streamed to file with a random file name (Line 976). The Internet Explorer object may then add this temporary file in order to display the XML certificate. It will also be possible to display an HTML transformed certificate with this object once implemented.

### 2. Implementation Metrics

XML – based security system is implemented for the following functionalities: i.e. default function, certificate issued, database connections, graphic user interface, loading, saving and signing certificates. The full definitions of Acronyms used in metrics are listed in the appendix.

**Table 2. Implementation Metrics**

| Metric* | Default | Certificate | Database Connection | GUI | Load Certificate | SaveCertificate | SignCertiticate |
|---|---|---|---|---|---|---|---|
| AC | 47 | 47 | 0 | 357 | 55 | 18 | 18 |
| CC | 92 | 39 | 4 | 92 | 26 | 5 | 3 |
| CR | 15 | 15 | 33 | 19 | 31 | 29 | 35 |
| DAC | 14 | 1 | 0 | 14 | 6 | 2 | 2 |
| DOIH | 12 | 1 | 1 | 1 | 1 | 1 | 1 |
| LOC | 2182 | 258 | 101 | 1417 | 238 | 63 | 58 |
| MNOP | 4 | 3 | 4 | 2 | 2 | 1 | 1 |
| MSOO | 15 | 4 | 1 | 15 | 15 | 2 | 2 |
| NOA | 69 | 15 | 0 | 69 | 7 | 2 | 2 |
| NOAM | 41 | 36 | 4 | 41 | 6 | 3 | 2 |
| NOC | 11 | 1 | 1 | 1 | 1 | 1 | 1 |
| NOCC | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| NOCON | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| NOM | 110 | 50 | 3 | 110 | 12 | 4 | 3 |
| NOO | 41 | 35 | 3 | 41 | 5 | 2 | 1 |
| PprivM | 64 | 29 | 0 | 88 | 54 | 40 | 50 |
| PprotM | 1 | 0 | 0 | 1 | 26 | 0 | 0 |
| PpubM | 35 | 71 | 100 | 11 | 46 | 60 | 50 |
| TCR | 17 | 17 | 50 | 23 | 44 | 41 | 54 |
| WMPC1 | 92 | 39 | 4 | 92 | 26 | 5 | 3 |
| WMPC2 | 102 | 57 | 13 | 102 | 14 | 6 | 3 |

*see appendix for definitions.

## RESULTS

Features demonstrated: 1) GUI for viewing, editing and creating a certificate; 2) Search facility; 3) Loading a certificate.

Currently implemented software is a workable graphical user interface (GUI) as shown below in Fig. (**5**). Split up into two sections, the left side provides an editable form version of a certificate. The right hand side shows an XML view of the certificate with updates in real time as and when changes are made via the editable boxes.

Also provided is another editable page for address or client information. Tabbed buttons at the top of the editable panel provide access to and from the different pages. The prototype currently models basic certificate information only but with class inheritance more information may be modeled. Fig. (**5**) shows an edited certificate.

The ability to save certificates is implemented. This includes the option of saving a new certificate or the update of an edited certificate. Fig. (**5**) illustrates the search facility as well. Up to 5 boxes are provided to filter a search of the database. For example, the user may request certificates with a specific ID and title. Should there be multiple returned results, these are presented as a selectable list to the user.

## DISCUSSION

Updating SQL server with XML: Updating multiple tables with one XML document. Use a unique ID from first table (only created on insert into a table (to be inserted from XML document)) for the foreign key of record in another table (to be inserted from XML document). A simple solution was found in using stored procedures within SQL server. The c# system sets up an SQL connection and calls the stored procedure passing the XML document in question as a parameter. The stored procedure holds the unique generated ID when created, as a variable and as such it is available as a foreign key value for the second table.

Problems with multiple certificates returned as a result: No root tag therefore will not validate as a well formed document. The built in XML API will not accept a non-well formed XML document. Solution has not been implemented but will include a collection of certificates each with a DOM certificate model.

Storage of a signature – no data manipulation is required. Therefore it stores as a BLOB (ntext) column in SQL server. As Signature ID as primary key so certificate table can be related to the relation with foreign key.

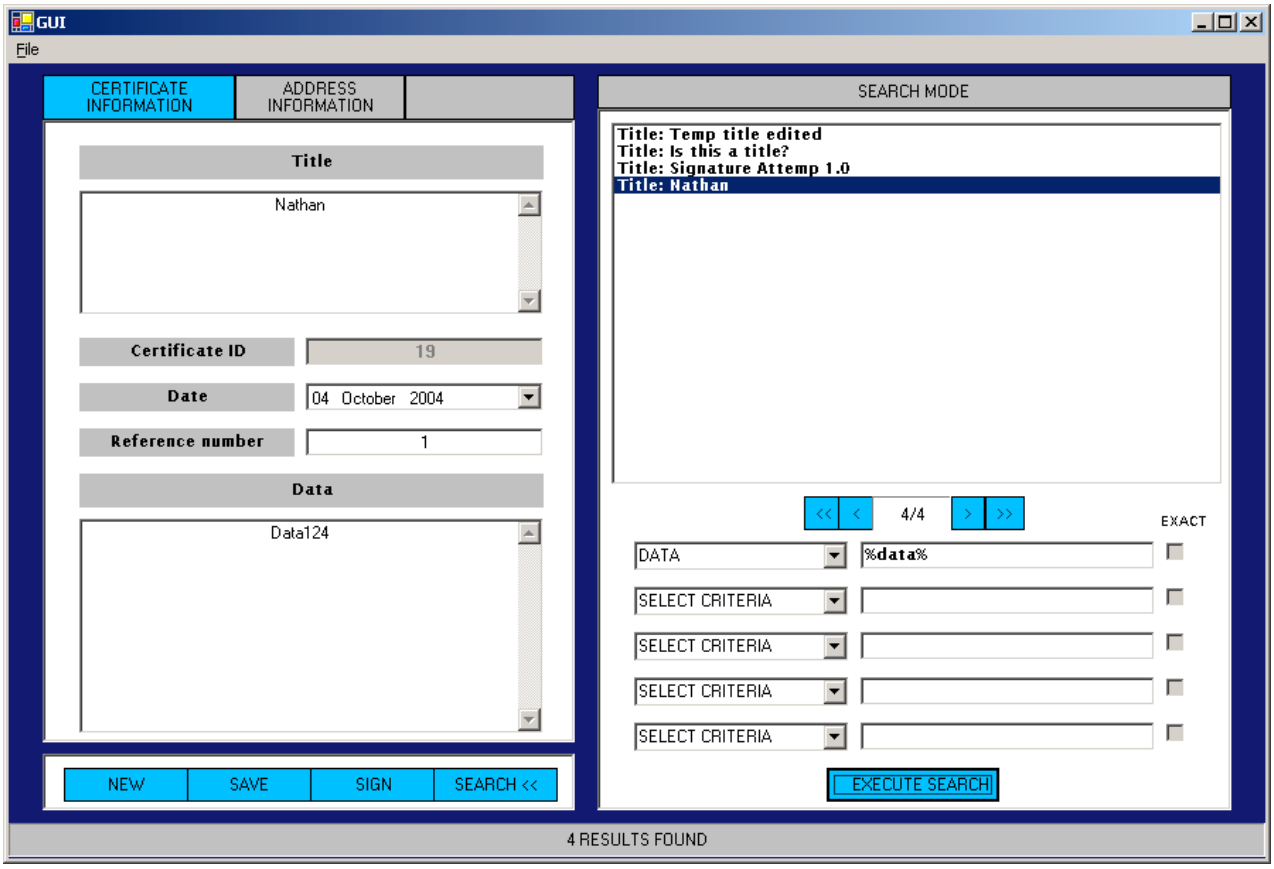


**Fig. (5).** X – Certificator mode1.

## APPENDIX:

### Metric Definitions

| Metric Acronym | Definitions | Metric Acronym | Definitions |
|---|---|---|---|
| AC | Attribute complexity | NOCC | Number of child classes |
| AHF | Attribute hiding factor | NOCON | Number of constructors |
| AIF | Attribute inheritance factor | NOM | Number of methods |
| CC | Cyclomatic complexity | NOO | Number of operations |
| CR | Comment ratio | NOOM | Number of overridden methods |
| DAC | Data abstraction coupling | PF | Polymorphism factor |
| DOIH | Depth of inheritance hierarchy | PIntM | Percentage of internal members |
| LOC | Lines of code | PPIntM | Percentage of protected internal members |
| MHF | Method hiding factor | PPrivM | Percentage of private members |
| MIF | Method inheritance factor | PProtM | Percentage of protected members |
| MNOP | Maximum number of parameters | PPubM | Percentage of public methods |
| MSOO | Maximum size of operation | TCR | True comment ratio |
| NOA | Number of attributes | WMPC1 | Weighted methods per class 1 |
| NOAM | Number of added methods | WMPC2 | Weighted methods per class 2 |

Changes to the design: To store the signature as a blob in SQL server as apposed to normalizing a relational version. There is no need to find part of a signature while in relational form. Only need a digital signature ID for retrieval.

As mentioned above restrictions of DBMS data types are due to maximum characters. Therefore it needs to store as a BLOB of CLOB which itself leads to other problems e.g. querying a BLOB. Solution is to create a DOM model and

query with XPath. But then drawbacks of a DOM model come into play.

A detached digital signature requires a physically stored reference. The Certificate is mapped as a DOM and therefore only represented in memory. In order to reference the XML document, compute the signature and create a temporary file. This file must also be created for signature validation.

## CONLUSION

This paper initially introduces the proposed prototype and the features that it will provide. Methodology of the system describes the decisions taken and the reasons behind them. This includes the theoretical and technological aspects underpinning the design. The three main theories include object oriented technology, XML security and relational database theory. Architecture of the system is included to show the system. Code snippets and UML diagrams support the current stage of implementation.

From theoretic approach to implementation of system, a prototype of XML based certificate system has been developed with the achievement of signature, authentication and verification of XML documents.

With an application of C# and .NET platform, Web integration is easily to be achieved. This is another advantage of Microsoft products as it can be linked to SOAP protocol in future industrial applications.

## REFERENCES

[1] W3C. Available from: http://www.w3.org/XML/ [Accessed: September 14, 2004].

[2] Lu Z. A survey of XML applications on scientific technology. Int J Software Eng Knowl Eng 2005; 1: 1-33.

[3] Lu Z, Cripps N. XML approach to key management system in scientific documents. International Conference of Internet Computing: CSREA, USA 2005.

[4] Williams K. Professional XML Databases. Wrox Press 2000.

[5] Kudrass T. Management of XML documents without schema in relational database systems. Information and Software Technology. Elsevier 2002; Vol. 44: pp. 269-275.

[6] Lu EJ-L, Chen R-F. An XML multisignature scheme. Applied Mathematics and Computation. Elsevier 2004; Vol. 149: pp. 1-14.

[7] Lekkas D, Gritzalis D. Cumulative notarization for long-term preservation of digital Signatures. Computers & Security. Elsevier 2004; Vol. 23: pp. 413-424.

[8] Lim BBL. Incorporating WS-Security into a Web services-based portal. Emerald Group Publishing Limited 2004; Vol. 12: pp. 206-217.

[9] Lu Z, Cripps N. Impact of XML applications on interdisciplinary and multidisciplinary applications. International Conference on Systemics, Cybernetics and Informatics. International Institute of Informatics and Cybernetics, Orlando, Florida, USA 2005.

[10] Lu J, Fox R. Effectively Storing and Querying XML. Proceedings of International Conference on Internet Computing and Conference on Computer Games Development: CSREA Press 2007.

[11] Selkirk A. XML and security. BT Tech J 2001; 19: 23-34.

[12] Bertino E. XML security. Information Security Technical Report. Elsevier 2001; Vol. 6: pp. 44-58.

[13] Liberty J. Building. NET applications: Programming C# 3rd ed. O'Reilly 2003.

Received: March 31, 2008 Revised: April 04, 2008 Accepted: April 28, 2008