

# XML security in certificate management

Joan Lu, Nathan Cripps and Chen Hua\*

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

*J.lu@hud.ac.uk*

\*Institute of Technology, Xi'an, Shaanxi, P. R. China

## 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.*

## 1. 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. In lay of recent research concerning XML and security [2], 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 [3].

The system 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 document because they are 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. Progress is shown in this report with annotated screenshots and UML documentation. Following sections discuss the problems faced during the implementation and a review of what has been achieved. The report finishes with a recommendation of the next procedural stage.

## 2. Methodology

### 2.1 Underpinning Theories

3 key theories are utilized within this project.

#### 2.1.1 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.

It is based upon PKI (public key infrastructure) fundamentals which is heavily mathematically based and proven. 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 [6] but the latter has received the greater attention.

### 2.1.2 Object Oriented (OO) theories

Object Oriented techniques achieve advancement over procedural programming. 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.

### 2.1.3 Database theory.

The most commonly used type of database is the relational [4]. This basis of which all data is modeled as relations with common relationships used to link them together. 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. 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.

## 2.2 Technologies involved

### 2.2.1 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 (10). SSL and TLS only secure data during transportation for length of a handshake [4].
2. XML security has a granular structure and there are no limits to the number of referrals to a document [7]. 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 [8]:

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 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.

### 2.2.2 OO technologies

C# developed by Microsoft is a platform independent high level OO language. C# was unveiled in July 2000 and takes as a basis the advantages of all previous OO languages [9]. 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.

### 2.2.3 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 [6]. 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.
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.

### 2.3 System design



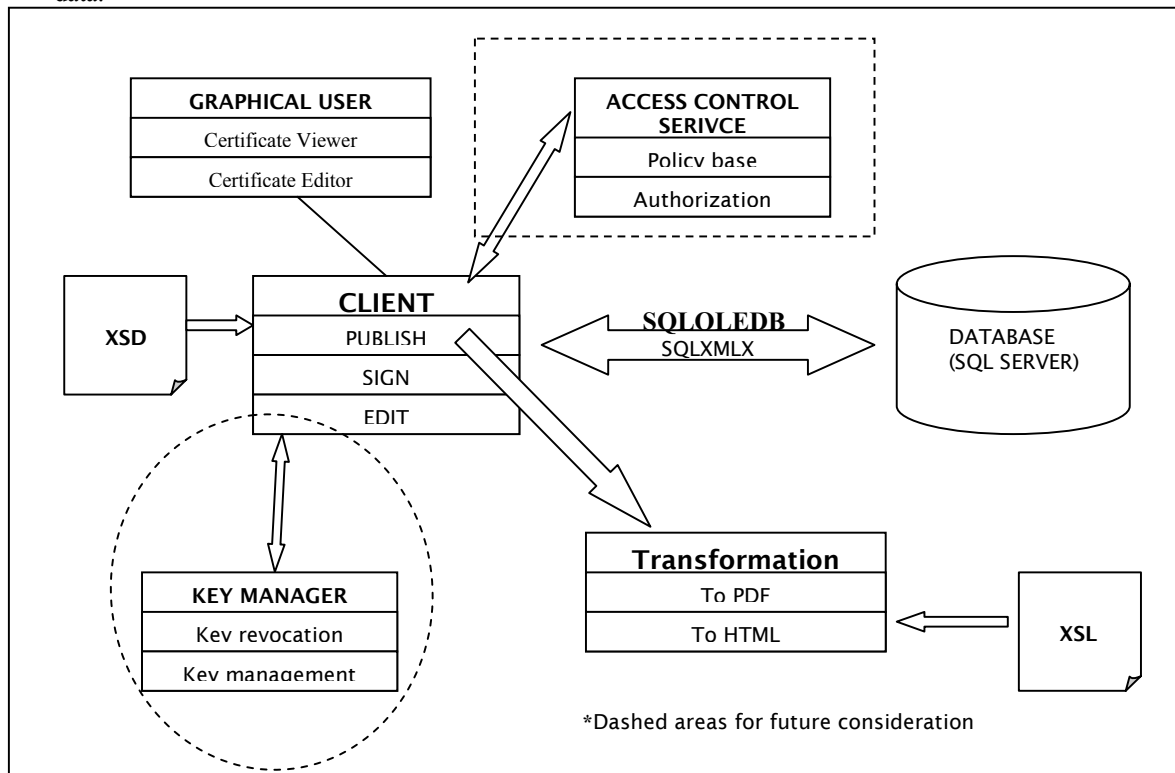
**Figure 1: architecture of system**

Fig 1 presents the architecture for the proposed system involving the 3 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:

- C#: Implementation of the system. Provides a high level OO language with XML support.
- XML: For security and transportation of certificate data.

- 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.
- SQL: Storing and manipulating the relational data.
- UML: Modeling the OO design which helps to lower development time.



**Figure 2: overview model**

The architecture of the proposed system is shown in fig 1. Tools available to the client allow for publishing, signing and editing a XML certificate. The database is provided by Microsoft SQL Server 2000 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 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.

### 2.3.1 Design specification

#### Goals:

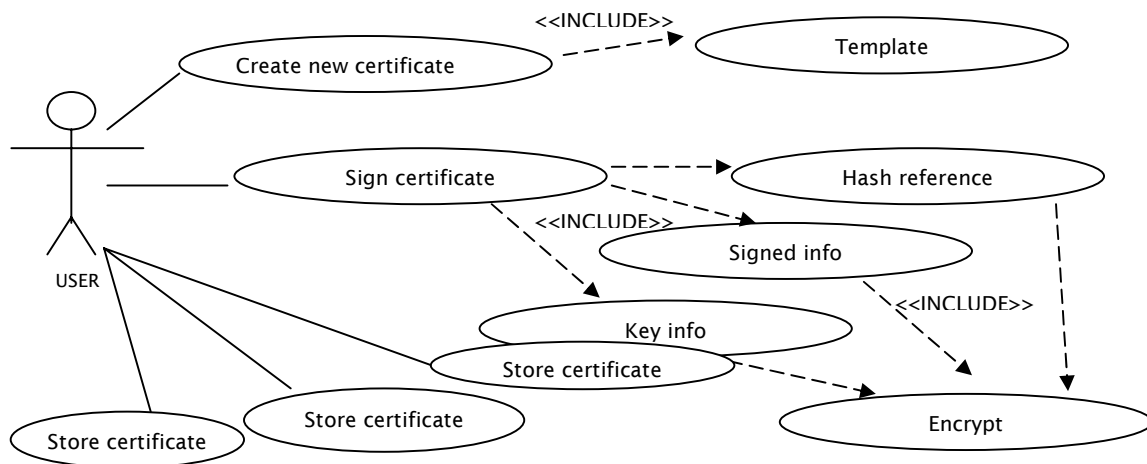
- To use current security technologies for certificates.
- To use digital signatures for authentication.
- To securely hold data in XML format.

- To represent data in XML for maximum use of its extensibility
- To store and retrieve certificates from a database.
- To store and retrieve digital signature from a database.

Figure 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).

## 3. Implementation

### 3.1 UML Class diagram



**Figure 3: Specification use case**

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 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) trend suggest the Certificate and GUI class are considered to be the most complex. 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 a HTML transformed certificate with this object once implemented.

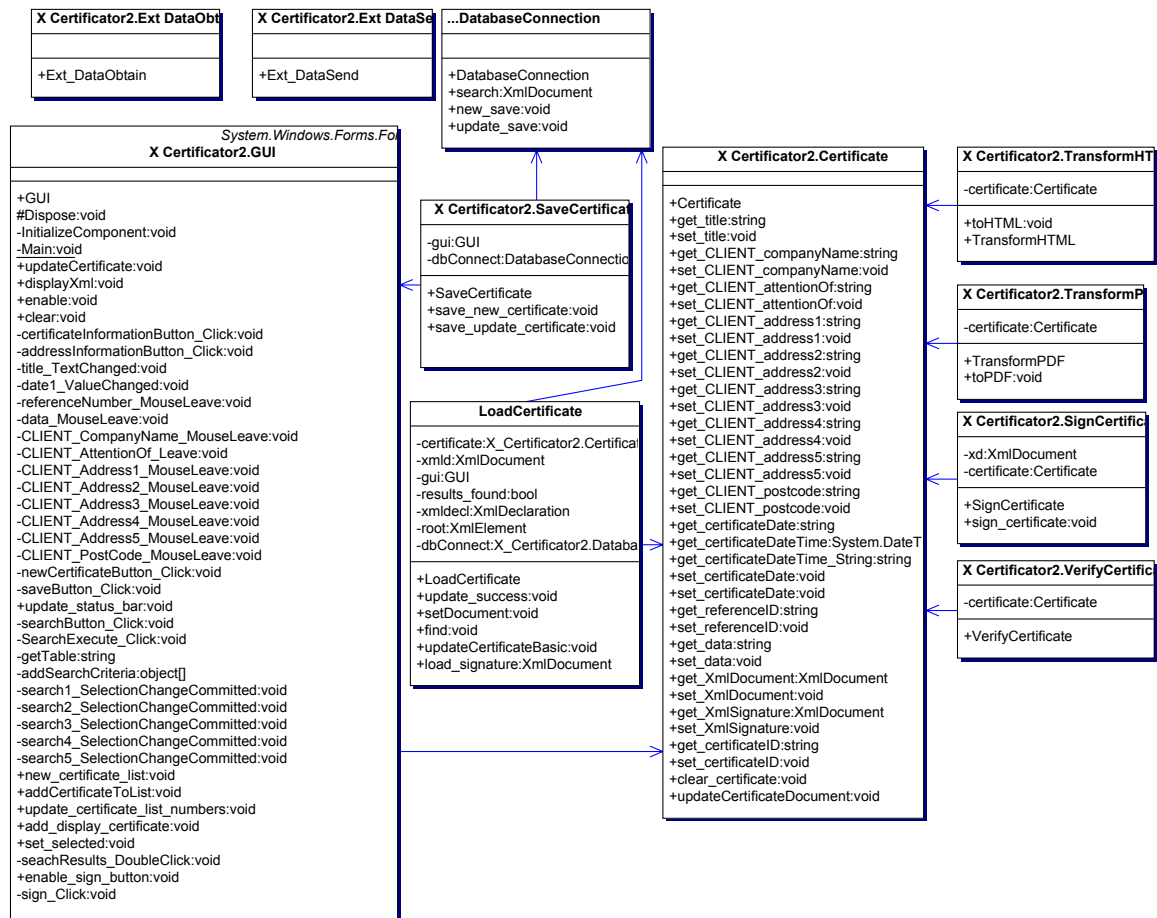


Figure 4: class diagram

### 3.2 Metrics

Table 1: Implementation metrics

Metric*	Default	Certificat e	Database Connectio n	GUI	LoadCertificat e	SaveCertificat e	SignCertificat e
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

## 5. 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.

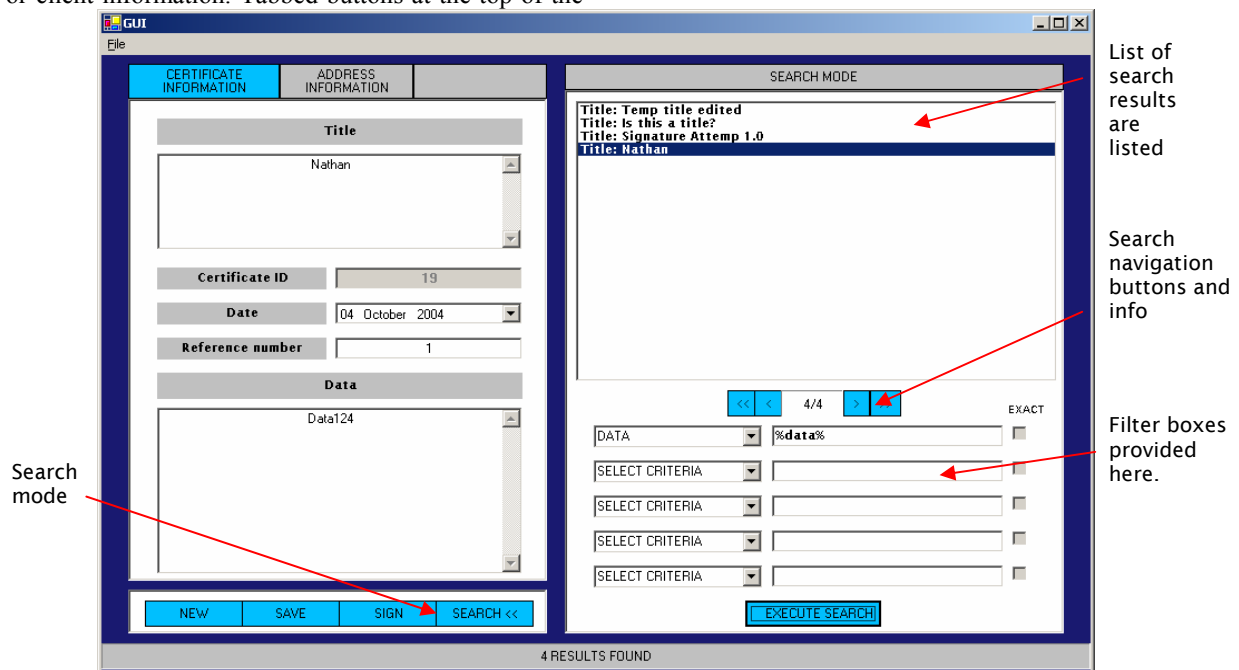


Figure 5: search mode

## 5. 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 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 of it required. Therefore store 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.

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 due to maximum characters. Therefore need 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 and therefore compute the signature a temporary file had to be created. This file must also be created for signature validation.

## 6. Conclusion

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. Results are provided with screenshots of the prototype. A discussion features above describing the problems encountered.

## References

- [1] W3C Available from: <<http://www.w3.org/XML/>> [Accessed: 14/9/04].
- [2] Lu, Z. and Cripps, N., XML Approach to Key Management System in Scientific Documents, *International Conference of Internet Computing*, USA, 2005, ISBN: 1-932415-69-6
- [3] Williams K, (2002), "Professional XML Databases", Wrox Press.
- [4] Lim B B L (2004), "Incorporating WS-Security into a Web services-based portal", Vol 12, Issue 3, pp. 206 – 217.
- [5] Dournaee B, (2002), "XML Security", McGraw-Hill.
- [6] Kudrass T, (2002), "Management of XML documents without schema in relational database systems", *Information and Software Technology*, Vol 44, pp. 269-275.
- [7] Selkirk A, (2001), "XML and security", *BT Technical Journal*, Vol 19, issue 3, pp. 23 – 34.
- [8] Bertino, E (2001), "XML security", *Information Security Technical Report*, Vol 6, Issue 2, pp. 44 – 58.
- [9] Liberty J, (2003), "Building .NET applications: Programming C# 3<sup>rd</sup> Edition", O'Reilly.

## Appendix: Metric definitions

Metric Acronym	Definition	Metric Acronym	Definition
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