SOLUTIONS FOR THE DEVELOPMENT OF DECISION SUPPORT SYSTEMS

Cristina Ofelia STANCIU
"TIBISCUS" UNIVERSITY OF TIMIŞOARA, FACULTY OF ECONOMICS

Abstract:
Storing data used for a decision support system in the XML file has been inspired by the open source project WEKA. The idea of using the C# programming language together with the XML format has proven to be a great success, as they are new and modern technologies. The assembly of the data representation in the XML format, together with the XSD schema and applications developed with object oriented programming languages that offer function libraries for processing these data models represents a powerful, efficient and mostly elegant solution.

Key words: *decision support systems, programming language, C#, XML*

JEL classification: D80, M15

The boundaries of structural programming have imposed the need of finding a new programming model, which would adequately contribute to data abstraction. Object based languages (like Ada and CLOS) and the object oriented languages (like Smalltalk, C++ and Java) are based on this model. The object oriented approach offers tools for a natural representation of the problem's elements.

The object is defined [Stoicu 00] as "an entity that includes data structures as well as operations". An object is described by an identifier and a series of properties that are able to support different actions.

Object oriented programming is [Booch 94] "an implementation method that uses programs that are organized as cooperating object collections, each object being the instance of a class; each class belongs to a class hierarchy and the classes are related by inheritance relations."

The last decade has been influenced by the object oriented programming languages, most of the computer based information systems being developed using languages as Delphi, Visual Basic, C++ or Java. There is a relatively recent language, C#, which is considered a descendant of C++.

In July 2000, with the appearance of C#, Microsoft has launched the .NET framework, that mainly is a frame that offers a new programming interface and integrates technologies such as ASP, XML, SOAP, WSDL which this far have only been used by Microsoft separately. The .NET also allows the integration of different programming languages for developing applications, making possible to import classes written in another language that the one the programmer is being familiar with. This feature is offered by the CLS (Common Language Specification) that imposes the minimum conditions that need to be achieved for a language to be integrated in the .NET language family.

The main components of the .NET development environment are:

- its „official" languages: C#, Visual Basic .NET, Managed C++, Jscript .NET;
- CLR (Common Language Runtime), the common platform of the above mentioned languages, that includes a virtual machine similar to the Java virtual machine;
- FCL (Framework Class Library), a collection of class libraries.

Compiling the programs in .NET will not generate, as one would expect, an executable format, but an intermediary format called MSIL (Microsoft Intermediate Language), that is afterwards executed by the CLR. The intermediary program is once again compiled b the JIT (Just In Time) compiler that will produce the executable format.

Figure 1 shows the .NET framework components. The main classes are based on the CLR, followed by a data level and XML classes, and at the superior level there are classes for developing web service and Windows applications.

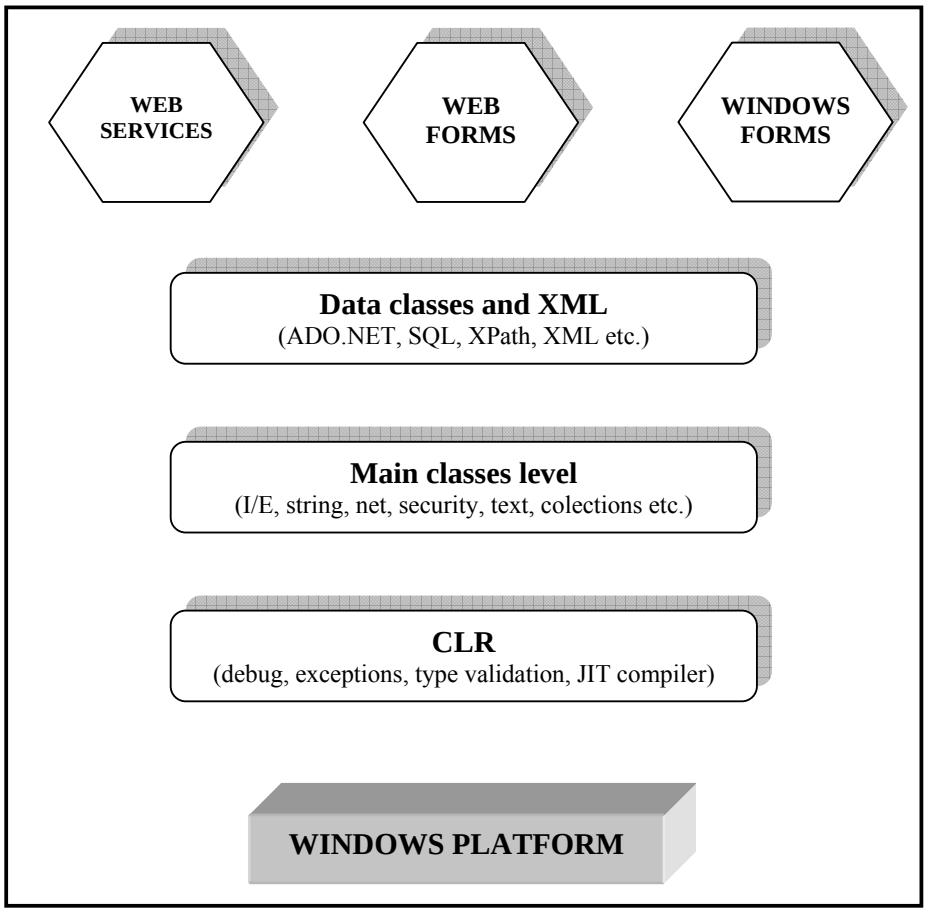

Figure 1. *The .NET framework components*

The development of the C# programming language was part of the .NET initiative and has received the ECMA (European Computer Manufacturers Association) and ISO standards, the syntax being based on the C++ syntax, but also using features from another languages, such as Delphi and Java.

According to the ECMA standards, the new programming language was aiming the following:

- C# was meant to be a simple, modern, object oriented language;
- the language must include features for data type validation, for detection of use of uninitialized variables;
- the C# language is used for complex programs, but also for smaller programs with dedicated functions;

- the C# language does nor offer the same results regarding performance or size as the C language or assembly language, in spite of the fact that I doesn't require too much memory or processor.

The main differences between C# and other languages, such as C or C++, are the following:

- there are no global variables or functions. All the methods and members will be declared within the classes;
- C# admits a logic type, bool, used in conditional situation, as an *if* or a *while*, that require the evaluation of a Boolean expression;
- multiple inheritance is not possible, although a class can implement any number of interface;
- the managed memory can be released using the *garbage collection* method that prevents memory "leaks";
- the type safety is stronger in C# than in C++. No implicit conversions are possible between the Boolean and integer type, and any conversion defined by the programmer has to be marked as explicit or implicit;
- C# has currently 77 reserved words.

In spite of the positive feed back that C# has received since the launch, there are also less favorable aspects of this programming language. A first aspect was that the programs written in C# demand more resource than similar applications written in other languages. Another aspect shown by the critics was that C# has been implemented for .NET, which is only available for the Windows operating system, but this observation is only valid for the non-standard libraries developed by Microsoft than for the C# language. As proof, there are other programming environments, such as Mono or DotGNU that support C #programs, including on other platforms (Linux, BSD, Mac OS X).

Regardless the minuses given to the C# language, it remains a powerful language that can be successfully used in developing advanced computer based information systems, as the decision support systems are. It has been mentioned before that the .NET framework offers a new programming interface and integrates technologies such as ASP, XML etc. This is where the idea of using the XML format in developing a decision support system came from.

XML is a modern format and most of the high level visual programming languages (such as Visual C++, Visual C#, Java) are capable to process or manage XML files and the data from those files.

Modeling data in XML format allows data validation by defining XML schemas. The XSD (*XML Schema Definition*) files define rules and relations between the elements and attributes using an XML file, and offer support for namespaces, data types and more advanced features. An XML schema defines: the elements that appear in an XML document, the possible attributes of the elements, the elements with descendants, if an element is void or contains text etc.

The data models represented in the XML format together with the XSD schemas and some applications developed in programming languages that offer libraries of functions for processing these data models ((.NET, Java packages, Qt) represent a powerful and efficient solution, and is an elegant solution from the object oriented programming point of view.

There are two main methods for parsing XML files: SAX method and DOM method. Each of these two methods has its own advantages, in confliction with the opposite method.

- SAX (*Simple API for XML*) is a method based on events: when parsing each XML entity of the model there is an e emitted signal that the software developer

receives. The most important features of the SAX parser are the rapidity and simplicity.

- DOM (*Document Object Model*) is based on loading the whole XML file in the memory and then the software developer can use all its elements. An advantage is that this parsing method allows at any moment access to any node of the XML file through the functions and data types offered by the DOM function library implemented in a certain programming language. The most important features of the DOM parser are: model hierarchy and serial parsing.

In particular, considering the case of developing a decision support system, the XML format is well worth being taken into consideration, as it can easily represent and treat tree structures, it is easy to use and flexible.

The idea of storing data for a decision support system with the XML format was inspired by the WEKA (Waikato Environment for Knowledge Analysis) project, developed for research by a team from the Waikato University in New Zeeland.

WEKA is a system used for Machine Learning applications, offering a uniform user interface and a large number of algorithms, including rule induction algorithms (1R, T2, Induct), instance based algorithms (IB1-4, PEBLS, K*), regression algorithms (M5), relational rules algorithms (FOIL). WEKA is a very good solution for Data Mining problems, helping any specialist in a certain field to extract valuable knowledge from large data bases.

WEKA is an open source tool, developed in Java, that includes a series of Data Mining and Machine Learning techniques, even an implementation of the C4.5 decision tree. Decision trees are built automatically once the parameters are set. The system includes example data sets, and obviously also allows using personalized data.

The data representation in WEKA is realized using the ARFF (Attribute-Relation File Format) file format, which is a text file that contains a list of instances and a set of attributes. An ARFF file has two parts: the header part and data part. The header of an ARFF file contains the name of the relation, a list of attributes and the type of these attributes. The data part of the ARFF file consist in enumerating, in separate rows, the instances referring to each attribute defined in the file's header.

Regarding the data section, each instance is written on a separate row, and the values of the attributes of each instance will be comma separated, and obviously in the same order with the order the attributes have been declared in. The unknown values of the attributes will be marked with the "?" character and are considered missing values.

Similarly to the WEKA files, in the XML files one can first define the attributes and their type, and then the actual data, the instances, by enumerating the values of the previously defined attributes. To show the enhancements offered by the XML files, it is important to know that the XML file can be validated at loading, using an XML validation schema and the functions offered by the class libraries that also include the XML parser. The XML file can be validated once again when it is written on the disc, in order to check if the file will be correctly written.

Storing data used for a decision support system in the XML file has been inspired by the open source project WEKA. The idea of using the C# programming language together with the XML format has proven to be a great success, as they are new and modern technologies. The assembly of the data representation in the XML format, together with the XSD schema and applications developed with object oriented programming languages that offer function libraries for processing these data models represents a powerful, efficient and mostly elegant solution.

REFERENCES

1. Averweg, U. R., Erwin, G. J. - *Critical Success Factors for Implementation of Decision Support Systems in South Africa*,s Proceedings of the 32nd Hawaii International Conference on System Sciences, 1999
2. Cojocariu, A., Stanciu, Cristina-Ofelia, *Informatică de gestiune,* Editura Eurostampa, Timişoara, 2008
3. Ganguly, A. R., Gupta A., *Data Mining Technologies and decision Support Systems for Business and Scientific Applications*, Encyclopedia of Data Warehousing and Mining, Blackwell Publishing, 2005
4. Holmes, G., Donkin, A., Witten, I.H., *Weka: A machine learning workbench* – Proceedings of Second Australia and New Zealand Conference on Intelligent Information Systems, Brisbane, Australia, 1994
5. Power, D., *Categorizing Decision Support Systems: A Multidimensional Approach*, in volume *Decision Making Support Systems: Achievements, Trends and Challenges for New Decade*, Idea Group Publishing, 2003
6. Power, D., *Decision Support Systems Hyperbook*., Cedar Falls, IA: DSSResources.COM, 2000, http://dssresources.com/dssbook/.
7. Sommerville, I., *Software Engineering (6th Edition)*, Addison – Wesley Publishing, 2000
8. Turban E., Aronson J., *Decision Support Systems and Intelligent Systems*, Prentice Hall, SUA, 2001
9. van der Vlist, E., The W3C's Object-Oriented Descriptions for XML, O'Rellys Publishing, 2002
10. Witten, I. H., Frank, E., *Data Mining – Practical Machine Learning Tools and Techniques, Second Edition*, Norgan Kaufmann Publishing, 2005
11. Yergeau, F., colab, *Extensible Markup Language 1.0. (Fourth edition)*, (http://www.w3.org/TR/REC-xml/ - W3C Recommendation 16 August 2006, edited in place 29 September 2006)