Using XML in simulation modelling

Automatic code generation for XML-based models

Bourouis Abdelhabib

Département d'informatique

Institut des Sciences Exactes
Centre Universitaire Larbi Ben M'Hidi
4000 Oum El Bouaghi

ALGERIE
habib.bourouis@univ-batna.dz

Belattar Brahim

Département d'informatique

Faculté des Sciences de l'ingénieur
Université Colonel Hadj Lakhdar
5000 Batna

ALGERIE
belattarb@univ-batna.dz

ABSTRACT. This paper is about using xml standard to enhance simulation models exchange, reuse and high level interoperability. We present the concept and demonstrate the utility of the QNML, which is an XML dialect for describing queueing networks. Our work is split into three stages. Firstly, a graphical modelling framework allows users to visually specify their models and generates the equivalent QNML document. In a reciprocal way, graphical models may also be generated from those XML documents. Secondly, XML models are automatically translated into executable simulation models using XSLT. Finally, simulation trace and results are also described in XML to facilitate further processing.

KEYWORDS: discrete event simulation, web-based simulation, queueing networks, XML schema, XSLT, graphical modeling.

1. Introduction

Simulation models refer narrowly to executable ones, coded directly using simulation or programming languages. At this level, there is a lack of interoperability, so models exchange and reuse are not assured between simulation and/or analysis tools. The need for powerful and common accepted standards, especially those XML-based was highlighted in [07] and [12]. Since modelling is an interesting early step in simulation, we are interested in developing a simulator-neutral model description language for queueing networks that facilitates automatic simulation code generation.

Our work is subscribed under web-based simulation, which integrates web technologies with the field of simulation. It has been introduced recently and is currently the subject of much interest to both simulation researchers and simulation practitioners. This includes for example the use of HTML, CGI scripts, java applets, servlets, RMI, VRML and particularly the XML standard. Web-based simulation packages provide several beneficial features like wide availability, controlled access, efficient maintenance and increased integration.

The web language XML represents a new way for organizing knowledge and information. It is vendor and platform independent. XML-based applications can export the contents of internal structures in such a way that another application can easily import them. The same XML source document could be read by many applications, each of which can transform the data into its unique input requirements, using the XML family of specifications. XML validation capabilities ensure that data files can be error free, eliminating many runtime application problems.

Our work is also based on JAPROSIM, see [01], as a simulation language, but any other simulation/programming language may be used instead, as a target for coding XML-based models. The second section discusses related work and similar researches. The third section is about the use of XML within the JAPROSIM library, both for modelling and output results. We will focus particularly on the Queueing Network markup Language (QNML) and the automatic simulation code generation. The last one exposes some interesting conclusions and future improvements.

2. Related work

A number of web-based discrete event simulation toolkits written in Java have been developed. Some examples are: JSIM [09] which supports the distribution of both simulation models and results, since they can be accessed via the web and SIMJAVA [06] uses RMI. Other researches focused on the interoperability of models using XML. In [04] an XML-based Supply Chain Simulation modelling Language was proposed and a specific

algorithm was used to map models into a simulation language. A similar work was presented in [14] using an XML-based job definition format for production chains. In [08] formalism for composing simulations from XML-specified components was proposed. Canonico in [03] worked on developing an XML-based modelling language for communication networks and uses Extensible Stylesheet Language Transformations (XSLT) to transform the model into ns-2. The IBM initiative Simulation Reference Markup Language (SRML) is a language for adding behaviour to XML documents. It combines XML and scripts to encode both the structure and behaviour of all items comprising a simulation [10].

The importance of our work is that Queueing Networks (QN) are not a domain specific formalism. Hence QNML is extensible and could be used in several domains like communication networks, production and supply chains, computer systems, transportation and logistics ...etc. In addition, it is a pure XML without any scripting. A graphical modelling tool and an automatic translation, using XSLT, to an executable simulation model, based on JAPROSIM, are offered.

3. JAPROSIM and XML

JAPROSIM is part of a whole project that aims at providing a kernel for Discrete Event Systems (DES) modelling and simulation. The project includes a graphical modelling and explanation facilities. XML is used to:

– Describe simulation models in the form of QN using QNML at a conceptual level, resulting in a high level of interoperability. Models so obtained are simulator-neutral and could be exchanged and reused by translating them into any simulation/programming language.

– Describe simulation trace and results using Simulation Output Description Language (SODL). Hence, results could be shown to the user in a variety of graphical presentations (pie charts, histograms ...etc). Simulation trace is well structured to facilitate further processing (searching explanations of ambiguous scenarios or results).

– Translate QNML models to a specific simulation/programming language using XSLT. Hence, Simulation models become runnable simulation applications.

The entire process is shown in the Figure 1. In the first stage, user can visually build the conceptual model using a graphical modelling tool. Models are saved in XML documents according to QNML. The graphical modelling tool allows proceeding backward to obtain a graphical model from its corresponding XML document. During the second stage, an automatic translation tool generates the simulation program. In our case, all needed Java classes according to the JAPROSIM library. The obtained simulation model is straight executable. It is possible to translate QNML documents to theoretically any simulation

language, since interoperability is guaranteed by this description language. In the later stage and after running the simulation model, the trace and results are also described and saved in XML using OSDL. This facilitates simulation output analysis and exchange.

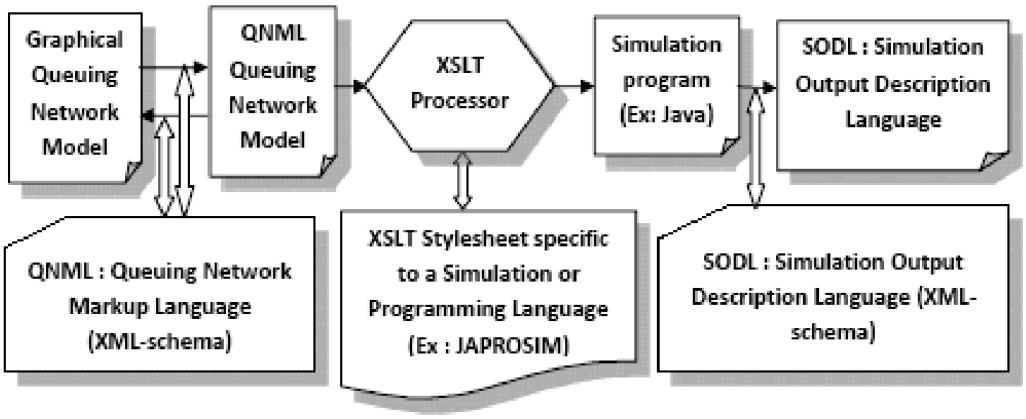

Figure 1. The use of XML and JAPROSIM in Simulation modelling

3.1. Queueing Networks Markup Language

Queueing networks are commonly used to model a wide range of discrete event systems. Kendall's notation is a mean for modelling using QN especially in case of simple systems. For complex ones, a graphical model with textual annotations is used. To simulate such systems, the QN model is commonly coded in a simulation or programming language, so simulation models could not be exchanged nor reused. The description of a QN may be graphical and/or textual, but there is no standard description language to do so. In addition, the description language must allow machine processing and human handling. Generally, developing such a language for describing conceptual models, not only for QN, is a real challenge for simulation modelling.

Discrete event models are made up by two parts. The static part describes all the components with their attributes, and the dynamic one describes the behaviour of each component. XML is not a programming language. It is used for structuring information and knowledge, so the description of the dynamic part of a DES using XML is not evident. The main advantage of modelling using QN is that these two parts are merged. Describing the static part of a QN, which means identifying all stations with their characteristics and interconnections, leads to implicit dynamic part description. This is why QN models are suitable to be described using XML.

In its simplest form, a QN is a set of service stations; each one is distinguished by an identifier, characterized by a number of parallel identical servers, a waiting queue with limited or unlimited capacity, a service probability distribution and discipline. A service station may or not receive transaction from outside of the system, but must have at least an output, towards another station or outside the system. Inputs from outside are branded by identifiers and have probability distributions to describe arrival processes. If not mentioned, transactions will have the same priority. An output is identified by a reference to the destination station and a routing probability.

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" version="1.0">
  <xsd:element name="qnet">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="desc"/>
        <xsd:element ref="station" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="station">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="distribution"/>
        <xsd:element ref="jobs" minOccurs="0"/>
        <xsd:element ref="input" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="output" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="identifier" type="xsd:positiveInteger" use="required"/>
      <xsd:attribute name="nbServers" type="xsd:positiveInteger" use="required"/>
      <xsd:attribute name="discipline" type="disc" use="required"/>
      <xsd:attribute name="alt-disc" type="disc" use="optional" default="FIFO"/>
      <xsd:attribute name="capacity" type="xsd:integer" use="optional" default="-1"/>
      <xsd:attribute name="batchmin" type="xsd:positiveInteger" use="optional" default="1"/>
      <xsd:attribute name="batchmax" type="xsd:positiveInteger" use="optional" default="1"/>
      <xsd:attribute name="slice" type="xsd:decimal" use="optional" default="1"/>
      <xsd:attribute name="preemption" type="xsd:boolean" use="optional" default="false"/>
    </xsd:complexType>
  </xsd:element>
```

Figure 2. *Fragment of the XML schema for QNML*

XML is a meta-language that encourages deriving task-specific languages using Document Type Definitions (DTDs) or XML-schemas. It is possible to validate an XML document against (DTDs), but much of XML's power comes from the ability to define an XML-schema, which is the foundation of a task-specific language. However a schema has several advantages over DTD. It is written in XML, which facilitate its processing, comes with a rich and extensible set of predefined types and permits in addition to specify occurrences of elements. The XML-schema fragment in Figure 2 summarizes most of the

rules, vocabulary and data types needed to describe a QN model as discussed previously. It defines an XML dialect named QNML.

3.2. Automatic simulation code generation

The code generation process is a model transformation operation. More accurately, it is considered as graph transformation since models here are simply graphs. The source model described in XML is a graph (a tree of nodes) according to the Document Object Model (DOM) API. The Target model is a java application and is also a graph seen as an Abstract Syntax Tree. This process is guided by a graph grammar which is a set of rules in the form of: *Left Hand Side (LHS) Right Hand Side (RHS)*, to be applied to the source graph. Our module for automatic code generation is based on the theory of model transformation by graph transformation [11]. The main advantage using this theory is especially the formal and clarity of mapping between models. Instead of coding transformation rules in a programming language, the use of XSLT rules helps maintaining and reusing the translation software for future versions or other simulation languages. Figure 3 shows an XSLT template used to translate an XML node into Java code according to JAPROSIM, represented as literal text.

```
<xsl:template match="input">
<xsl:text>class Transaction</xsl:text>
<xsl:value-of select="../@identifier"/><xsl:text>_</xsl:text>
<xsl:value-of select="./@identifier"/><xsl:text> extends Transaction {
</xsl:text>
      <xsl:choose>
        <xsl:when test="distribution/@function[contains(., 'Exponential')]">
          <xsl:text> static Exponential arrival = new Exponential(</xsl:text>
          <xsl:value-of select="dist/@param"/><xsl:text>);
          </xsl:text>
        </xsl:when>
```

Figure 3. *Fragment of the XSLT Stylesheet used in translation*

XSLT is a powerful language for manipulating the data in XML documents. It provides a series of operations and manipulators, while XML Path Language (XPath) provides precision of selection and addressing. So in order to translate the conceptual model into an executable simulation model, we used an XSLT stylesheet and an XSLT processor. The XML model description (QNML file) is transformed into the corresponding set of Java classes according to JAPROSIM. We integrated this module within the graphical modeling interface. It is extensible for other simulation languages easily by associating the corresponding XSLT stylesheets. Hence, the user will be free to choose his target simulation language.

It is important to mention that various graph description languages were developed. GraphML [02] and GXL [13] are XML-based but GML [05] is not. They deal with any type of graph and are based on the graph theory. The lack of a standard language widely adopted and since QNML also describes a graph, it is important to note that we can use XSLT to easily transform QNML documents into other graph description languages for other types of processing (particularly visualization/modification).

4. Conclusion and future work

The Queueing Network Markup Language (QNML) we presented is a very comprehensive vendor and platform independent exchange format based on the XML-Standard. It encourages model exchange and reuse. It is also easily transformable to an executable simulation model using XSLT. An automatic translation tool was developed according to the JAPROSIM library using an XSLT-stylesheet to obtain the complete set of java classes that implement QNML models. The use of XML was extended to simulation trace and output, which not only enhance data exchange, but make easy the process of simulation output analysis.

QNML is extensible and may be useful to generalize for discrete event process interaction simulation modelling. In a similar way, the simulation code generator is extensible and could be improved giving the user a wide range of target simulation languages by associating appropriate XSLT-stylesheets. It is also possible to make use of ontologies since they can be seen within computer simulation as a logical next phase of the web-based modelling and simulation thrust, where the emphasis is on knowledge and its representation rather than on run-time characteristics.

Bibliography

[01] Bourouis. A, Belattar. B, « JAPROSIM: A Java Framework for Discrete Event Simulation », in *Journal of Object Technology*, vol. 7, no. 1, January-February 2008, pp. 103-119, http//www.jot.fm/issues/issue_2008_01/article3/

[02] Brandes. U, Eiglsperger. M, Herman. I, Himsolt. M, Marshall. M. S, « GraphML progress report: Structural layer proposal ». *Proc. 9th Intl. Symp.* Graph Drawing (GD '01), Lecture Notes in Computer Science 2265:501–512. Springer. 2002.

[03] Canonico. R, Emma. D, Ventre. G, « An XML Description Language for Web-Based Network Simulation ». *The 7th IEEE International Symposium on Distributed Simulation and Real-Time Applications, 76—83*. 2003.

[04] Chatfield D. C, Harisson T. P, Hayya J. C, « XML-based Supply Chain Simulation Modeling. *Proceedings of the 2004 Winter Simulation Conference.* 2004.

[05] Himsolt M, « GML, A portable Graph File Format ». Information on GML is available on the world wide web at http://www.uni-passau.de/Graphlet/GML. 1995

[06] Howell. F, McNab. R, « simjava: a discrete event simulation package for Java with applications in computer systems modeling ». *The first International Conference on Web-based modeling and Simulation, San Diego CA, Society for Computer Simulation.* 1998.

[07] Kilgore, R. A. « Open source simulation modeling language (SML) ». *Proceedings of the 2001 Winter Simulation Conference*, ed., B. Peters,J. Smith. Piscataway, NJ: 2001

[08] Mathias. R, Adelinde. M. U, « Composing simulations from XML-specified model components ». *Proceedings of the 2006 Winter Simulation Conference.*2006.

[09] Miller. J. A, Rajesh. N, Zhiwei. Z, Hongwei. Z, « JSIM: A Java-Based Simulation and Animation Environment ». *Proceedings of the 30th Annual Simulation Symposium (ANSS'97), 31--42, Atlanta, Georgia.* 1997.

[10] Reichenthal. S. W, « SRML - Simulation Reference Markup Language ». http://www.w3.org/TR/2002/NOTE-SRML-20021218. 2002.

[11] Rozenberg, G, « *Handbook of Graph Grammars and Computing by Graph Transformation, Vol. 1: Foundations* ». Singapore: World Scientific.1997.

[12] Wiedemann. T, « Next generation simulation environments founded on open source software and XML-based standard ». *Proceedings of the 2002 Winter Simulation Conference.* 2002.

[13] Winter. A, « Exchanging Graphs with GXL ». *Proc. 9th Intl. Symp.* Graph Drawing (GD'01), Lecture Notes in Computer Science 2265:485–500. Springer. 2002.

[14] Wolfgang. K, « Simulation of the production chain by use of an XML-based job definition format ». *Proceedings of the 14th European Simulation Symposium.* 2002.

Biographies

BOUROUIS Abdelhabib received his BS degree in Computer science from the University of Constantine in 1999 and his MS degree from the University of Batna in 2003 where he is preparing a PhD degree. He is a lecturer at the University of Oum el Bouaghi since 2003.

BELATTAR Brahim is a professor at the University of Batna since 1992. He has also taught at the University of Constantine from 1982 to 1985. He received his BS degree in Computer science from the University of Constantine in 1981 and his MS and PhD degrees from the University Claude Bernard of Lyon (French) respectively in 1986 and 1991.