The paper titled " DEVSML 2.0: The Language and the Stack " was presented at DEVS Symposium, Spring Simulation Multiconference 2012 at Orlando, FL.

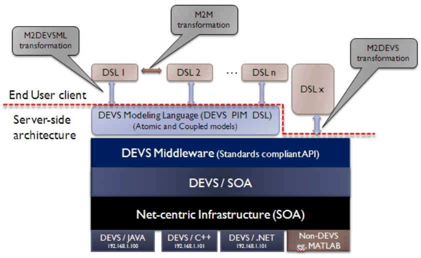
The earlier version of DEVSML stack [1,2] developed models in Java and in platform independent DEVS Modeling language that used XML as a means for transformation. The model semantics were bound together by XML. The latest version of the DEVSML, the language, is based on EBNF grammar and is supported by DEVS middleware API. The middleware is based on DEVS M&S Standards compliant (under evaluation) API and interfaces with a net-centric DEVS simulation platform such as a service oriented architecture (SOA) that offers platform transparency. With the maturation of technologies like Xtext [3] and Xpand [4] we have now extended the concept of XML-based DEVSML to a much broader scope wherein Domain Specific Languages (DSL) can continue to be expressed in all their richness in a platform independent manner that is devoid of any DEVS and programming language constructs. The key idea being domain specialists need not delve in the DEVS world to reap the benefits of DEVS framework.

The DEVSML 2.0 stack in figure below adds three transformations at the top layer:

1. Model-to-Model (M2M)

2. Model-to-DEVSML (M2DEVSML)

3. Model-to-DEVS (M2DEVS


M2DEVSML transformation
M2M transformation
M2DEVS transformation
DSL 1
DSL 2
...
DSL n
DSL x
End User client
Server-side architecture
DEVS Modeling Language (DEVS PIM DSL) (Atomic and Coupled models)
DEVS Middleware (Standards compliant API)
DEVS / SOA
Net-centric Infrastructure (SOA)
DEVS / JAVA 192.168.1.100
DEVS / C++ 192.168.1.101
DEVS / .NET 192.168.1.101
Non-DEVS eg. MATLAB

The end-user as indicated in figure above will develop models in their own DSL and the DEVS expert will help develop the M2M and M2DEVSML transformation to give a DEVS backend to the DSL models. While M2DEVSML transformation delivers an intermediate DEVS DSL (the DEVSML DSL), the M2DEVS transformation directly anchors any DSL to platform specific DEVS. There are many DEVS DSLs that implement a subset of rigorous DEVS formalism. One example of DEVS DSL is XML-based Finite Deterministic DEVS (XFD-DEVS) [5]. DEVSSpecML [6] built on BNF grammar is another example of DEVS DSL. DSLs can be created using many available tools and technologies such as: Generic Modeling Environment (GME) [7], Xtext , Ruby, Scala and many others. DSL writing tools like Xtext, Ruby, etc. focusing directly on the EBNF grammar provide a much easier foundation to develop the Abstract Syntax Tree (AST) for M2M transformations. The rich integration and code generation capabilities with open source tools like Eclipse give them strong acceptance in the software modeling community.

The addition of M2M, M2DEVSML and M2DEVS transformations to the DEVSML stack adds true model and simulator transparency to a net-centric M&S SOA infrastructure. The transformations yield models that are platform independent models (PIMs) that can be developed, compared and shared in a collaborative process within the domain. Working at the level of DEVS DSL allows the models to be shared among the broad DEVS community that brings additional benefits of model integration and composability. The extended DEVML stack allows DSL-s to interact with DEVS middleware through an API. This capability enables the development of simulations that combine and execute DEVS and non-DEVS models [7]. This hybrid M&S capability facilitates interoperability.

References:

[1] Mittal, S, Martin, JLR, Zeigler, BP, **DEVSML: Automating DEVS Simulation over SOA using Transparent Simulators** , DEVS Syposium, 2007

[2] Mittal, S, Martin, JLR, Zeigler, BP, **DEVS-Based Web Services for Net-centric T&E** , Summer Computer Simulation Conference, 2007

[3] **Xtext** Language Development Framework accesible at: http://www.eclipse.org/Xtext/

[4] **Xpand** Model Transformation Framework accessible at:

http://www.eclipse.org/modeling/m2t/?project=xpand

[5]
Mittal, S, Zeigler, BP, Ho, MH,
XFDDEVS: XML-Based Finite Deterministic DEVS
last accessed Jan 2011 at:
http://www.duniptechnologies.com/research/xfddevs/

[6]
Hong. KJ, Kim, TG,
DEVSpecL-DEVS specification language for modeling, simulation and analysis of discrete event systems
,Information and Software Technology, Vol. 48, No. 4, pp. 221 - 234, Apr., 2006

[7]
Martín, JLR, Moreno, A, Aranda, J, Cruz, JM,
Interoperability between DEVS and non-DEVS models using DEVS/SOA
. In SpringSim'09: Proceedings of the 2009 spring simulation multiconference: 1-9 (San Diego, CA, USA, 2009)

[8]
Saurabh Mittal, Scott A Douglass,
Net-centric ACT-R Based Cognitive Architecture With DEVS Unified Process
, Spring Simulation Multiconference, Boston, April 2011

[9]
Saurabh Mittal, Scott A Douglass,
From DSLs to DEVS Components: Applications to Cognitive M&amp;S
, Spring Simulation Multiconference, Boston, April 2011

[10] Saurabh Mittal, Scott A Douglass,
DEVSML 2.0: The Language and the Stack
, Spring Simulation Multiconference, Orlando, FL, March 2012.