

# A Design of a UML Profile for Meta-computing

Julien Taillard

*Laboratoire d'Informatique  
Fondamentale de Lille  
Université des Sciences  
et Technologies de Lille  
taillard@lifl.fr*

Jean-Luc Dekeyser

*Laboratoire d'Informatique  
Fondamentale de Lille  
Université des Sciences  
et Technologies de Lille  
dekeyser@lifl.fr*

Francis Piriou

*Laboratoire d'Électrotechnique  
et d'Électronique de Puissance  
de Lille  
Université des Sciences  
et Technologies de Lille  
francis.piriou@univ-lille1.fr*

## 1 Introduction

Meta-computing and grid computing are an important research area of modern distributed computing. A lot of work is made to facilitate the writing of distributed applications. Most of these approaches proposed language extensions or libraries. These approaches do not facilitate readability, model reuse, nor portability. Unified Modeling Language (UML), which is a standard for visual modeling, can help users to specify distributed applications.

We are going to design a UML profile to describe distributed softwares, hardware platforms, and the mapping of the applications to the hardware target. Then, using model to model transformations, re-factoring and refinement of models, we can produce automatically an optimized distributed software for the targeted hardware, using different languages or libraries.

## 2 UML for meta-computing

The aim of meta-computing is to use distributed and heterogeneous architectures. It is considered as the future of parallel computing.

There are many frameworks and middleware to make programs for such architecture like Message Passing Interface (MPI) [4], High Performance Fortran (HPF) [3] or Common Component Architecture (CCA) [1].

Unfortunately, writing distributed applications requires being a specialist.

UML can be useful to model such applications. UML can be profiled for a specific domain such as High Performance Computing. Two profiles allow a clear separation to model software and hardware architecture. The main interests are:

- To refrain from designing programming languages to

express the two different models, application and hardware architecture.

- To profit from all the new systems dedicated to High Performance Computing without having to re-formalize these two models.
- To use a single modeling environment, possibly supporting a visual specification.
- To benefit from standard formats for exchange and storage.
- To be able to express transformation rules from model to model in a Model Driven Architecture (MDA) style. The MDA is based on models describing the systems to be built. A system description is made of numerous models, each model representing a different level of abstraction. The modeled system can be deployed on one or more platforms via model to model transformations.

We have already developed such tool which allows to model software and hardware separately. Gaspard [7] is a co-design environment for Multiprocessors System On Chip which allows to model software and hardware independently. Then, software can be mapped on hardware architecture and simulation or synthesis code of the systems can be generated. It is quite intellectually pleasant to use the same models for System on Chip and meta-computing.

### 2.1 UML profile

The UML profile has to model different aspects: hardware architecture, parallel software, and association and mapping between both. Lots of hardware modeling techniques and/or concepts have already been proposed as extensions to UML. The Gaspard hardware UML profile [5]

will be used. The modeling of distributed components is defined in some models like Corba CCM [2], but it is not specialized for High Performance Computing. Consequently, modeling of high performance applications is not very developed.

Pllana and Fahringer [8] have proposed to use activity diagrams to model distributed software. Activity diagrams are a simple and attractive way to model parallel applications, but this approach seems very close to programming model like message passing and shared memory. However we want to model softwares without execution model pre-occupation.

Gaspard introduces an UML profile for software which is specialized in data-parallel applications: it is expressed by a graph on dependency in a component-based design with data-flow/control-flow technic. Repetitive multiplicity has been proposed to express both data parallelism and regular topology of hardware.

Once software and hardware have been modeled, software can be mapped on hardware with the association model. Then, models can be exported in XMI (XML Meta-data Interchange) to have a model representation on file.

## 2.2 Transformation and code generation

When the transformation rules will be written, a transformation engine like ModTransf [6] can be used to do models transformation and code generation. Two kinds of transformation will be made. First, a Transfer Level Model (TLM) will be introduced. This model will be used to model communications in software according to the mapping of software on hardware and by respecting the dependencies. Then, from the transfer level model, it will be possible to generate code towards different platforms.

## 3 Conclusion and Future Work

More people should be able to use meta-computing with this profile because it should simplify the writing, maintaining and reuse of those applications. Additionally, applications such as numerical simulations could benefit from this technic by easily being adapted to distributed applications, reducing their execution time. This is an ongoing project.

First, the definition of the UML profile for meta-computing has to be evaluated, maybe extended and finalized, and a meta-model of association has to be proven adequate to the concepts of meta-computing. A new meta-model has to be proposed at the transaction level, as it is quite different from our TLM meta-model for SystemC simulation. In addition, we should specify the transformation rules (from the models) to make code generation and optimize this code to targeted hardware. To finish, we

will validate this method on a 3D electromagnetic simulation code which has been developed in L2EP (Laboratoire d'Électrotechnique et d'Électronique de Puissance de Lille).

## References

- [1] Cca. World Wide Web document, URL: <http://www.cca-forum.org>.
- [2] Corba ccm. World Wide Web document, URL: <http://openccm.objectweb.org/doc/ccm.html>.
- [3] High performance fortran. World Wide Web document, URL: <http://dacnet.rice.edu/Depts/CRPC/HPFF/index.cfm>.
- [4] Message passing interface. World Wide Web document, URL: <http://www-unix.mcs.anl.gov/mpi/>.
- [5] A. Cucurru. *Modélisation unifiée des aspects répétitifs dans la conception conjointe logicielle/matérielle des systèmes sur puce à hautes performances*. PhD thesis, Université des sciences et technologies de Lille, Laboratoire d'informatique fondamentale de Lille, Nov. 2005. (In French).
- [6] C. Dumoulin. ModTransf: A model to model transformation engine, Nov. 2004. <http://www.lifl.fr/west/modtransf>.
- [7] Laboratoire d'informatique fondamentale de Lille. Gaspard home page. <http://www.lifl.fr/west/gaspard/>, 2005.
- [8] S. Pllana and T. Fahringer. On customizing the uml for modeling performance-oriented applications. In *UML*, pages 259–274, 2002.