Action Spécifique 2011 GeMoC du GDR GPL

« Ingénierie du logiciel pour les systèmes hétérogènes »

`http://www.combemale.fr/research/gemoc/as2011/`

RAPPORT D'ACTIVITÉ

Benoit Baudry*, Benoit Combemale*, Julien DeAntoni** et Frédéric Mallet**
* Equipe Triskell (UMR IRISA), Rennes
** Equipe AOSTE (INRIA - UMR I3S - UNS), Sophia Antipolis

Contact principal : Benoit Combemale
IRISA, Bureau F236, Campus de Beaulieu, 35042 Rennes, France
`benoit.combemale@irisa.fr`

13 janvier 2012

Résumé

L'Action Spécifique 2011 GeMoC est une action à court terme (mai 2011 - décembre 2011) soutenue par le GDR CNRS *Génie de la Programmation et du Logiciel* (GPL) pour réaliser une étude sur l'activité menée en France et à l'international dans le domaine de *« l'ingénierie du logiciel pour les systèmes hétérogènes »* .

Après un rappel du contexte et des objectifs de l'AS GeMoC (Section 1), nous présentons dans ce rapport d'activité le résultat des différents travaux menés. Après le détail des différents évènements réalisés (Section 2), nous proposons une terminologie précise caractérisant les différentes hétérogénéités identifiées dans l'ingénierie du logiciel (Section 3). Sur la base de cette terminologie, nous proposons dans la section 4 la synthèse d'une revue systématique et comparative sur une des hétérogénéités identifiées : *la modélisation hétérogène*. Nous présentons ensuite l'organisation et les conclusions de la journée nationale de travail organisée à Paris et rassemblant de nombreux acteurs français représentant les différentes hétérogénéités identifiées (section 5). Nous concluons ce rapport par un bilan et un ensemble de propositions résultant des échanges menées dans le contexte de l'AS GeMoC en vue de structurer l'activité sur le thème de l'hétérogénéité dans l'ingénierie du logiciel (Section 6).

1 Description de l'Action Spécifique

1.1 Contexte

La complexité grandissante des systèmes embarqués actuels nécessite une communication forte entre les différents acteurs métiers du développement. Chacun de ces acteurs utilisent des langages et des formalismes différents, issus de leur domaine d'activité et avec leur propre sémantique. Ces différents langages reposent sur des modèles formels différents qui permettent de répondre d'une manière spécifique à certaines contraintes comme des contraintes de sûreté, d'accessibilité, de respect d'exigences temporelles, de sécurité, etc. Ces modèles formels reposent sur des modèles de calcul décrivant précisément la façon dont les différents processus s'exécutent, communiquent et se synchronisent. Ils contribuent à donner la sémantique d'exécution et de synchronisation de l'entité sur laquelle ils sont appliqués. Puisqu'un même système embarqué est conçu à partir de plusieurs langages ayant chacun leur propre modèle de calcul, il est nécessaire de savoir les composer de manière fiable. Un des freins majeur à cette composition est dû au fait que le modèle de calcul est fondu dans le modèle fonctionnel métier. Ceci est notamment dû à l'absence d'un langage spécifique pour la description explicite du modèle de calcul et ce malgré le nombre abondant, en France et à l'étranger, de travaux théoriques sur la composition de modèles de calcul.

L'ingénierie dirigée par les modèles (IDM) fourni des techniques et des outils pour manipuler les modèles directement, raisonner sur leur composition, les transformer et générer le code d'implantation. Ceci permet en particulier d'avoir un flot intégré où modèles d'analyse et de conception sont traités dans le même espace technologique. Lorsque l'IDM est appliquée à la conception et l'analyse de systèmes hétérogènes à logiciel prépondérant, il est fondamental de dissocier les modèles fonctionnels et leurs sémantiques associées (manipulation / transformation de données) de ceux qui caractérisent la sémantique de calcul, de communication et de synchronisation. Chacun de ces modèles doit pouvoir être manipulé séparément avant assemblage/tissage, vérification et déploiement par génération ou interprétation.

1.2 Objectifs

Le but de cette action spécifique, appelé GeMoC, est d'avoir une réflexion sur les efforts actuellement menés en France pour la définition de systèmes logiciels hétérogènes au regard de la compétition internationale. En particulier, nous proposons un état de l'art et des verrous technologiques et scientifiques restant à lever pour la composition formelle de modèles métiers et de leurs modèles de calcul associés. Nous proposons également une identification des acteurs et des actions menées ou en cours. Même s'il est très clair que la modélisation et la programmation de systèmes hétérogènes à logiciel prépondérant relèvent directement du GDR GPL, nous anticipons que le groupe de travail puisse aller au delà et être constitué d'équipes impliquées dans plusieurs GDRs. C'est justement l'objectif de l'AS d'identifier précisément le périmètre d'un groupe de travail avec les acteurs français du domaine et de déterminer son positionnement au sein des organisations existantes telles que le GDR. Une fois le groupe de travail constitué, il se donne la mission d'encourager les échanges entre équipes et notamment de :

- coordonner les efforts nationaux ;
- organiser des manifestations scientifiques ;

– encourager l'intégration d'outils pour constituer une plate-forme logicielle open-source ;
– coordonner les réponses à des appels à projets nationaux ou internationaux.

L'AS GeMoC vise donc à répondre à deux objectifs principaux :
– Un état de l'art et des verrous actuels en se concentrant sur un périmètre limité et identifié de l'hétérogénéité dans l'ingénierie du logiciel ;
– Une proposition de structuration de la communauté française travaillant dans le domaine de l'ingénierie du logiciel pour les systèmes hétérogènes.

2 Calendrier des évènements

Nous rappelons ci-dessous la liste des évènements organisés dans le cadre de l'AS GeMoC.

1. 13 mai 2011 : l'AS est acceptée par le GDR GPL
2. 15 mai 2011 : mise en ligne de la page web de l'AS (cf. `http://www.combemale.fr/research/gemoc/as2011/`)
3. 28 juin 2011 : journée de travail en visio-conférence (Benoit Baudry, Benoit Combemale, Julien DeAntoni et Frédéric Mallet)
4. 14 septembre 2011 : journée de travail en visio-conférence (Benoit Baudry, Benoit Combemale, Julien DeAntoni et Frédéric Mallet)
5. 14 novembre 2011 : journée de travail à Sophia Antipolis (Benoit Combemale, Julien DeAntoni et Frédéric Mallet)
6. 23 et 24 novembre 2011 : journées de travail à Paris (Benoit Baudry, Benoit Combemale, Julien DeAntoni et Frédéric Mallet)
7. 25 novembre 2011 : journée nationale de travail à Paris avec l'ensemble de la communauté française ayant répondu à l'appel (cf. `http://www.combemale.fr/research/gemoc/as2011/#workshop`).

Le financement accordé à l'AS GeMoC par le GDR GPL a permis de prendre en charge les déplacements des organisateurs aux différents évènements ainsi que le déjeuner de l'ensemble des participants à la journée nationale de travail à Paris.

3 Hétérogénéité dans l'ingénierie du logiciel

« extrait d'un document de travail de l'AS GeMoC »

We identify three kind of heterogeneity in software engineering :
– Heterogeneous systems : orchestration of sub-systems (e.g., ULS)
– Heterogeneous execution platforms : cooperation of execution platforms (e.g., simulation engines)
– Heterogeneous modeling : cooperation of domain specific models

These three kinds of heterogeneity still independent and possibly complementary. Indeed, a uniform system (i.e., a system that could not be decomposed into autonomous sub systems) can be described by separating the concerns through heterogeneous modeling languages (structure, behavior, time...), and inversely an heterogeneous system (*e.g.,* System of Systems) can be uniformly described at a higher level of abstraction. In both cases, and depending of the intrinsic nature of the system (performance, distributed, ...), the system can be heterogeneously executed, both at design time (cooperation of simulation engine) and at runtime (using different execution "units"). By the way, the three kinds of heterogeneity can be combined and can be independently addressed. We focus in this survey on heterogeneous modeling.

4 Etat de l'art sur la modélisation hétérogène

4.1 Critères d'évaluation

« extrait d'un document de travail de l'AS GeMoC »

We are currently writing a survey whose goal is to classify various approaches to heterogeneity according to specific criteria. In this section we give an unambiguous description of such criteria and motivates the choices of the criteria while providing definitions needed to understand semantics of programming/modeling languages according to the state of the art.

First, a well accepted basis is that software languages (including programming and modeling languages) are built upon two fundamental features : syntax and semantics. Syntax refers to the concepts and relations that specify a well-formed program or model. Semantics refers to the meaning of these programs or models [15]. In the following, we detail the classical understanding of syntax and semantics according to the domain of use. We also use the term *mogram* to describe a program as well as a model [5].

The syntax of a language `L1` can be either concrete or abstract. The concrete syntax is usually the one finally used for someone who wants to write mograms in the `L1` language. This syntax can be textual or graphical. A textual concrete syntax can for instance be specified using a notation like BNF (Backus-Naur Form) as usually done in the context of programming language. BNF and their variant provide an elegant way to specify a concrete syntax. However, one limitation of the concrete syntax is that it specifies the way the language looks like but does not focus on its structural aspect. The abstract syntax avoids the specification of specific keywords and syntactic conventions to keep only the structure of a language. Since the middle sixties, researchers have shown the importance to consider the abstract syntax of a language to reason about its meaning [7, 8]. The use of an abstract syntax is currently promoted by the Model Driven Engineering (MDE). In an MDE development the abstract syntax is a first-order citizen whose definition is given via a metamodel. The main artifact is then a model, i.e., a manageable artifact that represents the abstract syntax of a specific instance of the metamodel. The abstract syntax can then be linked to a concrete syntax to ease the edition. Because the abstract syntax keeps only the structural aspects of specifications, its definition, i.e., the metamodel, is a set of concepts and relations. While the formal semantics of models are seldom dealt with, the abstract syntax of programs has been largely used to specify the semantics of programs [15, 17, 9]. The semantics of a program can mainly be given in five different ways :

operational the semantics is defined by an abstract interpreter of the language where rules define how operators modify the system state [9, 17] ;

denotational the semantics is defined by an association of each language construction to a mathematical function [9, 17, 16] ;

transformational the semantics is defined by reducing constructs of the language to more elementary ones by means of transformations into a simpler language whose semantics is already given [12].

axiomatic the semantics is defined by a mathematical theory associated with each language elements to enable the proof of some properties [9, 17, 3].

attribute grammar the semantics is defined by decorations of a context free grammar with attributes you are interested in. Basically attributes can take values from arbitrary domains and arbitrary functions can be specified, written in a language of choice, to describe how attributes values in rules are derived from each other [6].

Among these very common ways to specify formally a language semantics, each has its own pros and cons but an often desired goal is to avoid over specification because it reduces interpreter/compiler possibilities. It is possible to specify the semantics of concurrent program with such techniques (see chapter 14 of [17] for a quick overview). However, such approaches are tedious and each concurrency or synchronization operators must be formalized for each language that deals with concurrency.

Concurrency theory has been heavily inspired by the works of Hoare and Milner on process algebras, and also by the works on general net theory (Place/Transition nets). Milner's work on CCS (Calculus of Communicating System [11]) and Hoare's work on CSP (Communicating Sequential Processes [4]) ; were originally inspired from an inherently concurrent model called the "Actor Model" [2]. They independently suggested the same novel communication primitives : an atomic action of synchronization, with the possible exchange of values, as the central primitive of communication. An important point of such languages is that they provide explicit operators to capture concurrency and synchronization.

During the same period, the notion of MoC (*Model of Computation*, including time and communication) appears with work on digraphs [1] and the use of Petri nets techniques [14, 13]. These models have largely been refined and studied to specify the concurrency within and inter computations [10]. Even though multiple variants exist, the main idea is to represent the concurrent computations as nodes of a labeled directed graph. Each node of the graph corresponds to an operation in the computation and each edge represents a queue of data directed from one node to another. The operation execution takes data/tokens off the incoming edges and places results on outgoing edges. An operation can "fire" only if there are sufficient tokens on incoming edges. By doing so, MoCs focus on the data dependencies of programs. More precisely, they represent causality relationships between computations so that they can extract the potential parallelism from the causalities. They do not focus on the operation(s) performed by a node while it was the case for usual specification of the formal semantics of languages. It is also noticeable that the evolution of the notion of MoC put a large emphasis on the role of time and its impact on the functional specification.

Based on the different approaches used in the state of the art to implement languages, we defined four criteria on languages that enable the comparison between heterogeneous approaches

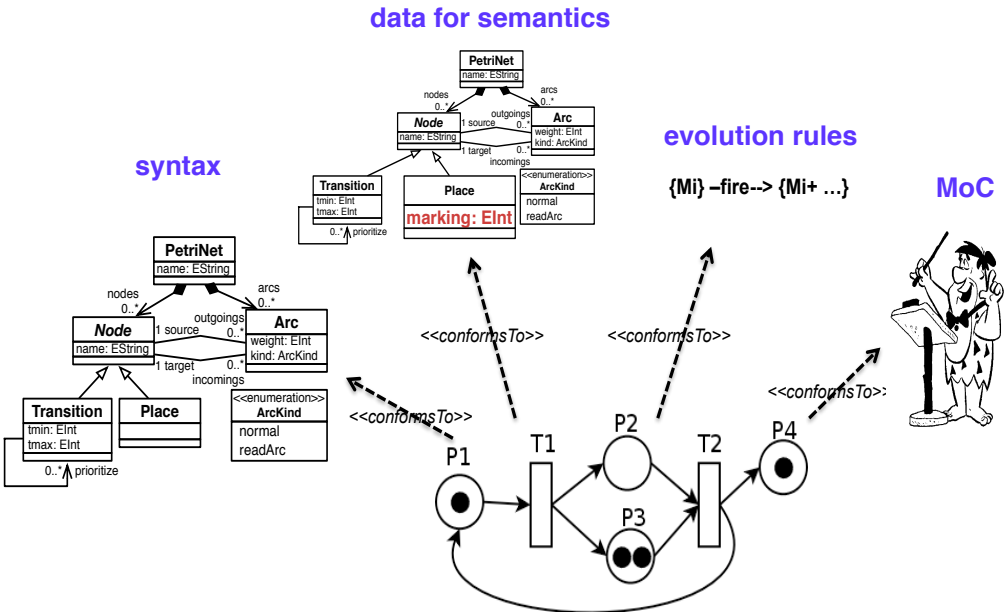

FIGURE 1 – Illustration of our four criteria on the Petri net language

and semantics :

1. syntax : it defines the concepts of the language as well as their relationships. It can either be concrete or abstract, based on the Model driven engineering technology or not.
2. data for semantics : it models the set of data needed to encapsulate the state of a system and on which the semantics is specified. It can be seen as the data part of an attribute grammar or as the data used in the specification of the system state in structural operational semantics rules. It can also be the domain on which the functions of a denotational semantics are applied.
3. evolution rules : It specifies how the data for semantics evolves according to the operator of the language. It is close to the function that can be defined in an attribute grammar or in the denotational semantics or close to the effect part of a structural operational semantics rule. It can also be seen as the set of transformations applied on the data in a transformational approach. We limit these evolution rules to define only the data manipulation and propose to separate all synchronization, concurrency and time, which is the matter of the MoC.
4. MoC : it defines the way the different evolution rules are scheduled. To do so, causalities, concurrency and time are specified and linked to the activation of evolution rules. By splitting evolution rules and MoC, it is easier to point at the changes of the system state from the ordering in which the changes occur.

We illustrate in Figure 1 our four criteria based on the Petri net language.

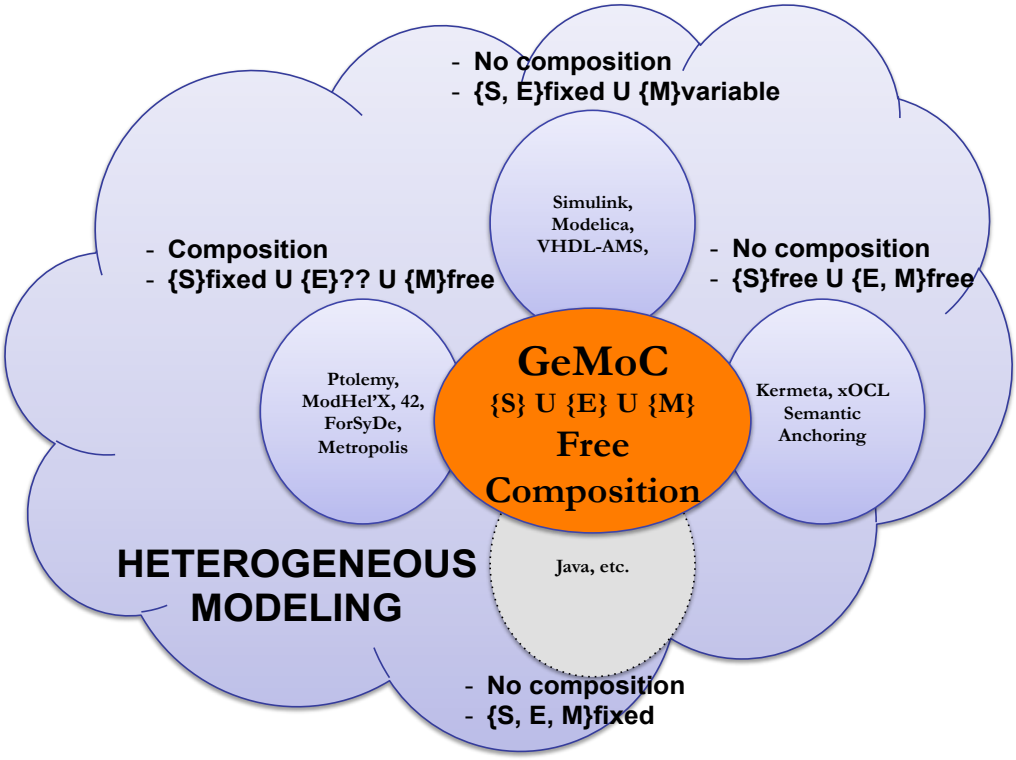

FIGURE 2 – Preliminary results of our survey on heterogeneous modeling

4.2 Résultats préliminaires de la classification

Sur la base des critères présentés dans la section précédente, nous avons établi un état de l'art et une classification des approches existantes. La Figure 2 propose un aperçu de cette classification où : S = syntax, E = evolution rules, et M = MoC. Notons que les informations dynamiques non pas d'impact dans la classification.

Dans cette figure, nous proposons quatre classes principales d'approches existantes, et positionnons les perspectives vis-à-vis de celles-ci (appelé GeMoC dans la figure). Pour chaque classe, nous précisons si les approches correspondantes offrent des mécanismes de composition des langages ($< s1, e1, d1, m1 > \circ < s2, e2, d2, m2 >$). Pour chaque critère, nous précisons également s'il est *free* (can be defined by users), *variable* (can be chosen by users) ou *fixed* (imposed by the approach).

Les travaux de recherche menés dans le cadre de l'AS GeMoC ont permis d'établir un état de l'art sur la modélisation hétérogène. Nous proposons dans ce rapport une synthèse de cet état de l'art et nous rédigeons actuellement un article en vue d'une soumission au début de l'année 2012.
Nous prévoyons également de proposer un numéro spécial dans une revue internationale sur le thème de l'hétérogénéité dans l'ingénierie du logiciel. Ce numéro spécial aura pour objectif d'identifier et de rendre visible les travaux et les résultats motivés par la même problématique, mais menés jusqu'à lors dans différentes sous communautés de l'ingénierie du logiciel.

5 Journée nationale de travail sur l'ingénierie du logiciel pour les systèmes hétérogènes

Webpage : `http://www.combemale.fr/research/gemoc/as2011/#workshop`

Présentation de la journée nationale de travail Cette journée nationale de travail, organisée le 25 novembre 2011 par l'AS GeMoC, a permis de rassembler les acteurs majeurs de l'ingénierie du logiciel en France afin de faire émerger un groupe de travail sur le thème des systèmes hétérogènes. Cette journée avait en particulier l'objectif d'identifier le périmètre d'un groupe de travail ainsi que ses axes scientifiques structurant. Concrètement, l'objectif était d'établir un ensemble de propositions à destination du GDR GPL en vue de structurer l'activité de recherche et de développement sur le thème de l'ingénierie du logiciel pour les systèmes hétérogènes.

En particulier, cette journée de travail fut l'occasion d'explorer les initiatives visant à faire coopérer différents points de vue hétérogènes : de la conception, ... à la simulation, ... à l'exécution. Parmi les sujets de discussions :

- Modélisation hétérogène : ingénierie des langages, composition de modèle, variabilité
- Simulation hétérogène : modèle de calcul, test et vérification, ...
- Plate-forme d'exécution hétérogène : création et configuration dynamique de plate-forme d'exécution, models@runtime, processus d'entreprise

– Domaine d'application : système de systèmes, système réactif, système embarqué, système autonome, ...

Programme Suite à l'appel à participation, nous avons reçu de nombreuses propositions de participation et de présentation. Nous avons volontairement restreint les exposés à la matinée pour consacrer l'intégralité de l'après midi à des discussions entre l'ensemble des participants.

– 09h00-09h30 : accueil / café
– 09h30-10h00 : présentation de l'AS et tour de table
– 10h00-12h00 : exposés
  – Valerie Issarny (INRIA Rocquencourt) : Interopérabilité dans les systèmes de l'Internet du futur
  – Jacques Malenfant (LIP6) : Composabilité et interfaces riches
  – Cécile Hardebolle (Supélec) : Modèles hétérogènes et adaptation sémantique dans ModHel'X
  – Laurent Rioux (THALES Research & Technology) : Vers une IDM pour des architectures
– 12h00-14h00 : déjeuner (pris en charge par l'AS)
– 14h00-14h30 : présentation des résultats de l'AS
– 14h30-15h30 : discussions sur les résultats de l'AS
– 15h30-16h00 : pause café
– 16h00-17h30 : discussions sur l'identification d'un groupe de travail

La présentation faite par les membres de l'AS GeMoC au cours de cette journée de travail est disponible à l'adresse : `http://www.combemale.fr/research/gemoc/as2011/20111125_workshop.pdf`. Les présentations des exposés au cours de la matinée sont disponible sur la page Internet de l'AS.

Participants La journée nationale de travail a réuni 27 participants issus de 16 groupes de recherche différents.

– Idir Ait Sadoune (Supélec)
– Benoit Baudry (INRIA Rennes)
– Nelly Bencomo (INRIA Rocquencourt)
– Réda Bendraou (LIP6)
– Mireille Blay (I3S)
– Frédéric Boulanger (Supélec)
– Pierre Boulet (LIFL)
– Joël Champeau (ENSTA-Bretagne)
– Cauê A. Clasen (EMN)
– Benoit Combemale (IRISA)
– Xavier Crégut (IRIT)
– Julien De Antoni (I3S)
– Jérome Delatour (ESEO)
– Sophie Ebersold (IRIT)
– Mahmoud El Hamlaoui (IRIT)
– Nikolaos Georgantas (INRIA Rocquencourt)
– Sébastien Gerard (CEA)
– Marie-Pierre Gervais (LIP6)
– Cécile Hardebolle (Supélec)

- Valerie Issarny (INRIA Rocquencourt)
- Ali Koudri (THALES Research & Technology)
- Jean-Christophe Le Lann (ENSTA-Bretagne)
- Jacques Malenfant (LIP6)
- Frédéric Mallet (I3S)
- Marc Pantel (IRIT)
- Laurent Rioux (THALES Research & Technology)
- Xavier Thirioux (IRIT)

Cette journée nationale de travail sur l'ingénierie du logiciel pour les systèmes hétérogènes a réunis 27 participants issus de 16 groupes de recherche différents. Les exposés ont permis de donner un bon aperçu de la diversité des travaux menés dans l'ingénierie du logiciel pour aborder les différentes problématiques liées à l'hétérogénéité. A l'issue des quatre exposés, l'ensemble des participants ont mené une discussion pour placer les différentes approches selon une taxonomie et une terminologie proposées par les organisateurs (cf. section 3), et ont mené une réflexion sur l'avenir du groupe de travail. Les participants ont unanimement exprimé le souhait d'identifier les acteurs des différentes sous communautés de l'ingénierie du logiciel travaillant sur les problématiques de l'hétérogénéité. Pour cela, les participants ont proposé la mise en place à court terme d'une page Internet identifiant l'ensemble des acteurs, d'une liste de diffusion pour faciliter les échanges, et d'essayer de re-conduire annuellement une journée de travail similaire à celle réalisée le 25 novembre 2011.

6 Conclusion

Bilan Le financement accordé à l'AS GeMoC a permis de prendre en charge le déplacement des organisateurs à plusieurs réunions de travail sur l'état de l'art dans le domaine de l'ingénierie du logiciel pour les systèmes hétérogènes. Il est important de noter que malgré le côté austère de la chose, les organisateurs ont privilégié le travail en visio conférence afin de permettre le financement d'une journée nationale plus chaleureuse. En particulier, le financement accordé à l'AS a permis de prendre en charge le déjeuner des participants à la journée nationale de travail le 25 novembre 2011 à Paris.

L'AS a atteint ses objectifs à la fois scientifiques par l'élaboration de l'état de l'art, et d'animation de la communauté scientifique par l'organisation de la journée nationale de travail.

Perspectives Nous prévoyons de soumettre un article au début de l'année 2012 sur la base de l'état de l'art réalisé au cours de l'AS GeMoC. Nous souhaitons également essayer de rendre visible les travaux sur l'hétérogénéité réalisés dans différentes communautés de l'ingénierie du logiciel en proposant en 2012 un numéro spécial d'une revue internationale.

La réunion nationale de travail à Paris a permis de réunir de nombreux acteurs français de la communauté de la programmation et du logiciel et donc relevant du GDR GPL. Cette première journée a permis d'identifier un groupe de travail et de structurer les différentes contributions françaises vis-à-vis de la classification élaborée dans l'état de l'art. Les participants ont unanimement

émis le souhait de poursuivre cet effort de structuration et d'identification dans les années à venir.

Au regard des résultats de cette *Action Spécifique* et du souhait de la communauté du logiciel, nous sollicitons le GDR GPL pour renouveler son soutien financier et permettre ainsi une consolidation des efforts réalisés au cours de cette première année. En particulier, nous sollicitons le GDR GPL pour une nouvelle action en 2012 permettant une consolidation du groupe de travail et l'organisation de nouvelles journées de travail.

Benoit Baudry, Benoit Combemale, Julien DeAntoni et Frédéric Mallet
Les organisateurs de l'Action Spécifique 2011 GeMoC du GDR GPL

Références

[1] Vinton Gray Cerf. *Multiprocessors, semaphores, and a graph model of computation*. PhD thesis, 1972. AAI7222158.

[2] C. Hewitt, P. Bishop, and R. Steiger. A universal modular actor formalism for artificial intelligence. In *Proceedings of the 3rd international joint conference on Artificial intelligence*, pages 235–245. Morgan Kaufmann Publishers Inc., 1973.

[3] C. A. R. Hoare. An axiomatic basis for computer programming. *Commun. ACM*, 12 :576–580, October 1969.

[4] C. A. R. Hoare. Communicating sequential processes. *Commun. ACM*, 21 :666–677, August 1978.

[5] Anneke Kleppe. *Software Language Engineering : Creating Domain-Specific Languages Using Metamodels*. Addison-Wesley Professional, 2008.

[6] Donald E. Knuth. Semantics of context-free languages. *Theory of Computing Systems*, 2 :127–145, 1968. 10.1007/BF01692511.

[7] J. McCarthy. Towards a mathematical science of computation. *Information Processing*, 62 :21–28, 1962.

[8] J. McCarthy and J. Painter. Correctness of a compiler for arithmetic expressions. *Mathematical Aspects of Computer Science*, 19, 1967.

[9] Bertrand Meyer. *Introduction to the theory of programming languages*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1990.

[10] R.E. Miller. A comparison of some theoretical models of parallel computation. *Computers, IEEE Transactions on*, 100(8) :710–717, 1973.

[11] R. Milner. *Communication and concurrency*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989.

[12] P. Pepper. A study on transformational semantics. In Friedrich Bauer, Manfred Broy, E. Dijkstra, S. Gerhart, D. Gries, M. Griffiths, J. Guttag, J. Horning, S. Owicki, C. Pair, H. Partsch, P. Pepper, M. Wirsing, and H. Wössner, editors, *Program Construction*, volume 69 of *Lecture Notes in Computer Science*, pages 322–405. Springer Berlin / Heidelberg, 1979. 10.1007/BFb0014674.

[13] C.V. Ramamoorthy and G.S. Ho. Performance evaluation of asynchronous concurrent systems using petri nets. *IEEE Transactions on Software Engineering*, 6 :440–449, 1980.

[14] C. Ramchandani. Analysis of asynchronous concurrent systems by timed petri nets. Technical report, Cambridge, MA, USA, 1974.

[15] David A. Schmidt. Programming language semantics. In *In CRC Handbook of Computer Science*, pages 28–1. CRC Press, 1995.

[16] Joseph E. Stoy. *Denotational Semantics : The Scott-Strachey Approach to Programming Language Theory*. MIT Press, Cambridge, MA, USA, 1981.

[17] Glynn Winskel. *The formal semantics of programming languages : an introduction*. MIT Press, Cambridge, MA, USA, 1993.