Un méta-modèle pour la représentation de systèmes à structure dynamique applicable à la reconfiguration dynamique partielle

Clément Foucher Min Zhu
LAAS-CNRS, Université de Toulouse, UPS, Toulouse, France
clement.foucher@laas.fr min.zhu@laas.fr

1 Introduction

Un modèle à structure dynamique est une modèle qui peut évoluer au cours du temps, non seulement par la modification de paramètres mais également de sa structure même. La structure d'un système peut être représentée par une hiérarchie de composants, chaque composant pouvant soit exposer un comportement, soit être découpé en sous-composants. On appelle les composants exprimant un comportant des composants « atomiques », tandis que les composants comprenant plusieurs autres composants sont nommés « couplés ». La notion de structure dynamique représente donc une notion de changement des composants constitutifs du système par ajout, suppression ou remplacement des composants et de leurs connexions.

Dans ce document, nous décrivons la syntaxe proposée pour PrDEVS, un méta-modèle à structure dynamique, et son application aux systèmes matériels reconfigurables.

2 Choix du méta-modèle

Le formalisme DEVS (Discrete-Event System Specification) a été proposé par Zeigler [4] comme un outil permettant la modélisation de systèmes en utilisant une structure hiérarchique, et leur simulation selon une sémantique à évènements discrets. La simulation à évènements discrets propose un paradigme différent de la simulation à temps discret. La simulation à temps discret découpe le temps selon des intervalles fixes (même s'il peuvent parfois être modifiés au cours de la simulation), tandis que la simulation à évènements discrets détermine le temps auquel le prochain évènement va se produire et « saute » directement à cet instant. Cette vision différente comporte des avantages sur la simulation à temps discret, tels que la minimisation des calculs inutiles (on ne fait pas de calcul s'il n'y a pas eu d'évènement), ou au contraire une granularité de calcul plus fine en cas de variations rapides d'un signal par la production d'évènements plus rapprochés.

DEVS n'est pas prévu pour permettre la modification de la structure d'un modèle au cours de sa simulation. Néanmoins, plusieurs extensions ont été proposées dans ce but, tel que Dynamic Structure DEVS (DSDEVS) [1] ou Dynamic DEVS (DynDEVS) [3]. Mais des limitations dans ces extensions nous ont poussé à définir une nouvelle extension. Parmi les limitations, on notera un ensemble des structures possibles figé, qui prévoit à l'avance toutes les configuration possibles de la structure du système, ou encore l'absence de gestion de la modification de l'état d'un composant de manière externe.

3 Syntaxe du méta-modèle PrDEVS

Plutôt que d'utiliser une syntaxe basée sur des évolutions de la structure générale du modèle, d'un état initial vers un état final, nous avons choisi de présenter une syntaxe basée sur l'ajout et la suppression de composants, ainsi que de sauvegarde et de chargement de l'état d'un composant (son « contexte »). Cette méthode, plus proche de l'approche « new/delete » de la programmation objet nous semble plus adaptée par rapport aux méthodes utilisées actuellement dans la conception d'un système, notamment dans un système matériel reconfigurable dans lequel on ajoute ou supprime des composants. Par ailleurs, PrDEVS met en œuvre l'approche d'ingénierie dirigée par les modèles [2] de l'OMG, dans laquelle on sépare clairement le modèle de haut niveau (Platform Independent Model – PIM) du modèle de la plateforme d'exécution (Platform Definition Model – PDM), la fusion d'un PIM pouvant être réalisée avec différents PDM pour former un modèle final spécifique (Platform Specific Model – PSM).

Dans notre syntaxe, un composant racine, le $PRDEVS$, contient toutes les informations du système :

$$PRDEVS = < L, L_\Phi, C^{Top} >$$

Dans lequel L est la bibliothèque de composants disponibles, L_Φ la bibliothèque de contextes, et C^{Top} un composant couplé.

Les composants couplés sont définis, d'une manière proche de DEVS, par la syntaxe suivante :

$$N^{nid} =< X, Y, D, EIC, EOC, IC >$$

Ici, X et Y représentent respectivement les entrées et les sorties du composant, D l'ensemble de composants contenus dans le couplé et EIC, EOC et IC les connexions entre ces composants.

Enfin, les composants atomiques présentent la structure suivante :

$$M_t^{mid} =< X, Y, S, s_0, \delta_{ext}, \delta_{int}, \delta_{con}, \lambda_{SC}, \lambda, \tau >$$

S représente l'ensemble des états possibles du composant, s_0 l'état initial, les δ sont les fonctions de transition entre les états du système, λ la fonction de sortie et τ la fonction de temps. Par rapport à DEVS, nous rajoutons ici une fonction λ_{SC}, qui émet une sortie spéciale à destination du simulateur, demandant une modification de la structure du système.

La nouveauté est donc principalement contenue dans cette fonction λ_{SC}. Celle-ci produit un appel de routine, mathématiquement définie comme un ajout ou une suppression d'élément dans un ensemble, qui impacte la structure du système. On constate aussi la notion d'id (nid et mid), ou identifiant d'un composant. L'id est généré à la création du composant, et permet d'identifier celui-ci de manière unique dans les appels de fonctions.

Par ailleurs, PrDEVS prend en charge explicitement l'état d'un composant, dans sa notion de contexte. Le contexte d'un composant est défini par son état $s \in S$.

Dans DEVS, un composant dispose d'un état initial $s_0 \in S$, puis celui-ci évolue au cours de la simulation. Dans PrDEVS, nous identifions précisément l'état d'un composant en permettant de récupérer celui-ci par un appel de λ_{SC} pour le stocker dans la bibliothèque L_Φ, ou à l'inverse de remplacer le contexte d'un composant par un contexte déjà stocké dans L_Φ. Cette méthode permet notamment d'envisager la suspension d'un composant du système, sa suppression, puis la possibilité de repartir ensuite de son état précédent si celui-ci a été sauvegardé.

4 PrDEVS appliqué à la reconfiguration partielle

L'approche utilisée dans PrDEVS se base à la fois sur un formalisme mathématique rigoureux (DEVS), et sur les pratiques réelles des développeurs (ajout/suppression de composants, sauvegarde de contexte). Un exemple de PDM a été proposé se basant sur un FPGA découpé en zones reconfigurables reliées par un bus, représenté sur la figure 1.

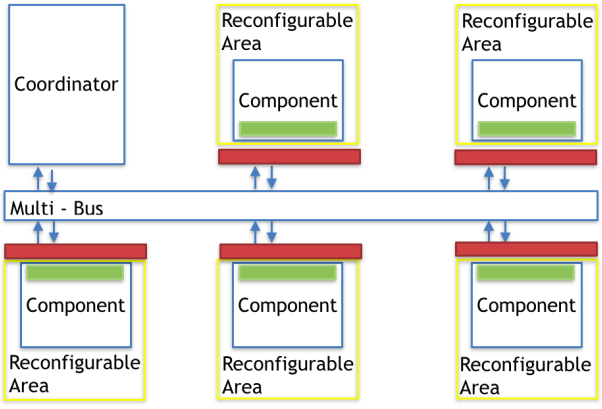


FIGURE 1 : PDM matériel pour PrDEVS

Sur la figure, la partie connexion, en rouge, contient des tables d'adressage faisant le lien entre les composants (représentant les connexions de type EIC, EOC et IC), qui peuvent évoluer. Le composant en lui-même est situé sur une zone reconfigurable, en jaune sur la figure, ce qui permet sa modification. Enfin, en vert, la partie contexte est une zone mémoire également modifiable.

Le bus contient la connectique, à la fois pour la communication entre les composants, la mise en œuvre des fonctions de changement de structure, et l'ordonnancement de l'exécution, car il s'agit d'un modèle évoluant en temps simulé. Le coordinateur, quant à lui, est un composant spécifique qui gère l'interprétation de la sémantique de PrDEVS, notamment la réponse aux appels de fonctions λ_{SC} et l'ordonnancement des cycles de simulation. C'est notamment le coordinateur qui va héberger la bibliothèque de contexte, venir mettre à jour les tables d'adressage et le contexte des composants, ainsi que faire les appels de reconfiguration dynamique partielle.

5 Conclusion

Dans ce document, nous avons présenté un méta-modèle dont la syntaxe et la sémantique sont des dérivés de DEVS, un formalisme mathématique pour la modélisation hiérarchique de systèmes et leur simulation à évènements discrets. Notre méta-modèle, PrDEVS, supporte les évolutions de structure dynamiques du modèle simulé, tout en adaptant l'approche pour se rapprocher de celle généralement utilisée dans la spécification de systèmes : la création et la suppression explicite de composants. Par ailleurs, PrDEVS apporte le support de la gestion des contextes des composants, ce qui permet d'envisager des suspensions d'exécution de composants par suppression, tout en étant capable de redémarrer ceux-ci dans l'état où ils étaient avant leur suppression. L'approche d'ingénierie dirigée par les modèles, qui sépare le modèle de l'application de celui de la plateforme, permet d'envisager différentes mises en œuvres du simulateur, tant en logiciel qu'en matériel. Un déploiement matériel sur FPGA dynamiquement reconfigurable est par ailleurs proposé en guise d'exemple.

Références

[1] F. J. Barros. Modeling formalisms for dynamic structure systems. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 7(4) :501–515, Oct. 1997.

[2] Object Management Group. Mda - the architecture of choice for a changing world, 2016. [Online ; accessed 19-January-2017].

[3] A. M. Uhrmacher. Dynamic structures in modeling and simulation : A reflective approach. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 11(2) :206–232, Apr. 2001.

[4] B. P. Zeigler, H. Praehofer, and T. G. Kim. *Theory of modeling and simulation : integrating discrete event and continuous complex dynamic systems*. Academic press, Orlando, FL, USA, 2nd edition, 2000.