HAL
open science

JeB : un environnement de simulation en JavaScript pour B événementiel

Faqing Yang, Jean-Pierre Jacquot

To cite this version:

Faqing Yang, Jean-Pierre Jacquot. JeB : un environnement de simulation en JavaScript pour B événementiel. Approches Formelles dans l'Assistance au Développement de Logiciels (AFADL), Apr 2013, Nancy, France. hal-00908037

HAL Id: hal-00908037

https://inria.hal.science/hal-00908037v1

Submitted on 22 Nov 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

JeB : un environnement de simulation en JavaScript pour B événementiel

Faqing Yang and Jean-Pierre Jacquot

DEDALE Team – Université de Lorraine – CNRS UMR 7503 LORIA
Vandoeuvre-lès-Nancy, France
{firstname.lastname}@loria.fr

Résumé Ce papier présente un outil, JeB, qui peut automatiquement générer des simulateurs de modèles B événementiel. Il fournit un environnement graphique d'exécution et l'interface pour combiner les codes écrits par l'utilisateur. Il peut être utilisé pour la visualisation et la validation des modèles B événementiel. Les simulateurs sont implantés en HTML et JavaScript. On peut simuler les modèles B événementiel dans les navigateurs actuels.

1 Introduction

L'usage des méthodes formelles telles que B [1] ou B événementiel [2] dépend crucialement des outils. Il existe des outils, les animateurs, comme Brama [1], ProB [3] ou AnimB [2], qui permettent d'observer comment l'état d'un modèle B événementiel évolue en fonction du déclenchement des événements. Malheureusement, ces outils ont des limites :

- le texte de la spécification doit être modifié pour obtenir une version animable,
- les constantes du modèle doivent être codées en extension pour que l'animateur puisse utiliser des stratégies d'énumération,
- certains domaines sont trop complexes pour que les valeurs puissent être énumérées efficacement.

Dans le cas où l'animation échoue, une façon de contourner ces limitations est de traduire le modèle B événementiel vers un langage de programmation :

- B2C [6] traduit vers C le sous-ensemble de notations B événementiel utilisées dans la spécification d'un processeur électronique,
- EB2ALL [3] est issu de B2C dont il élargit l'ensemble des notations traduites. Il permet la traduction vers plusieurs langages.

La contrainte principale de ces traducteurs tient à la nature des modèles traduits. Ceux-ci doivent être déterministes, en particulier, le choix des événements et leur ordre d'exécution est fixé.

1. http ://www.brama.fr
2. http ://wiki.event-b.org/index.php/AnimB
3. http ://eb2all.loria.fr/

Nous pensons qu'il est possible d'associer les deux idées, animation et traduction, pour générer des simulateurs. JeB est un outil basé sur les avantages de ces idées, de plus il ajoute :

- la simulation de machines de niveau d'abstraction quelconque,
- la prise en compte du non-déterminisme et des quantifications pour les domaines finis,
- une interface pour introduire du code écrit par l'utilisateur,
- la connexion à des outils graphiques pour la visualisation et la validation,
- des obligations de preuves pour garantir la correction d'usage de la simulation.

2 Implantation

JeB est implanté en Java (traducteur), JavaScript (bibliothèque pour B événementiel, contrôleur, code généré) et HTML (interface graphique). Les composants principaux sont les suivants :

- un traducteur : installé comme un *plug-in* Rodin [5], il génère automatiquement des simulateurs en JavaScript pour chaque machine ;
- une bibliothèque : elle implante toutes les notations mathématiques de B événementiel et interprète les formules pendant l'exécution ;
- un contrôleur : il ordonnance l'exécution des événements et permet aux utilisateurs de piloter la simulation à partir de l'interface graphique ;
- des simulateurs générés : il y a un simulateur par machine. Chacun comporte une page HTML pour l'interface graphique et une traduction en JavaScript des variables, invariants, événements et contextes ;
- des fichiers de configuration : ils définissent les paramètres des simulations et contiennent les fonctions codées à la main en JavaScript.

Le non-déterminisme intrinsèque de B événementiel fait qu'à chaque pas de l'exécution, il peut y avoir un choix de l'événement à déclencher. Nous avons donc adopté une stratégie proche des animateurs qui passe par trois étapes :

- détermination de l'ensemble des événements déclenchables (toutes les gardes sont évaluées),
- choix d'un événement à déclencher,
- réalisation de l'action de l'événement choisi.

Cette stratégie impose une traduction de l'événement qui doit séparer les gardes et les actions en deux parties. Le choix de l'événement est soit contrôlé par un ordonnanceur, soit demandé à l'utilisateur. Un ordonnanceur qui choisit aléatoirement l'événement est fourni par défaut ; l'utilisateur peut le remplacer pour implanter d'autres stratégies.

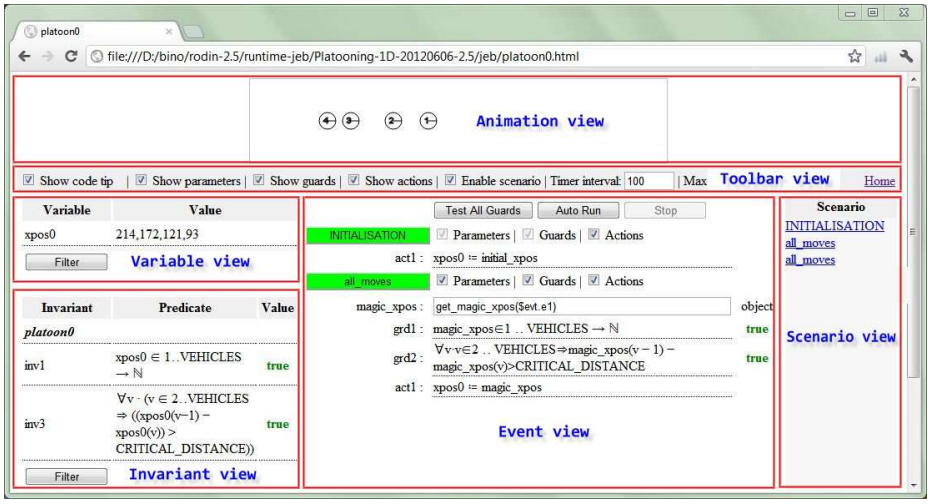

FIGURE 1. Fenêtre de simulation

3 Mode d'emploi

La construction et l'exploitation des simulations est un processus de collaboration entre les outils et les humains. Il passe par quatre étapes :

- la génération des simulateurs par activation du traducteur de JeB,
- la valuation des constantes et la programmation des fonctions codées à la main dans les fichiers de configuration,
- la mise en place de l'affichage graphique des états (optionnel), et
- l'observation et l'analyse des simulations.

Dans la pratique, nous faisons des observations à travers l'interface graphique représentée Figure 1. La zone supérieure est une vue graphique des états. La zone en-dessous est une barre d'outils qui définit les paramètres principaux de la simulation. La zone la plus basse est organisée en trois colonnes. La colonne de gauche est la vue des variables et des invariants ; la colonne du milieu présente la vue des événements ; la colonne de droite est la vue scénario qui permet l'analyse des historiques.

4 Application

Nous avons utilisé JeB sur plusieurs études de cas dont les deux spécifications de *platooning* présentées dans [7] et la spécification du stimulateur cardiaque présentée dans [4]. Le coût [4] de la simulation des trois modèles [5] est ré-

4. Il est compté en JeB version 0.4.5.
5. Les simulateurs sont accessibles à l'adresses : http ://dedale.loria.fr/ ?q=en/JeB.

	1D Platooning			2D Platooning			Stimulateur cardiaque		
Éléments	**Manuel**	**Total**	**taux**	**Manuel**	**Total**	**taux**	**Manuel**	**Total**	**taux**
ensembles	0	0	-	1	1	100%	1	1	100%
constantes	6	15	40%	24	50	48%	15	15	100%
fonctions de paramètre	2	11	18%	2	43	5%	0	26	0%
fonctions d'animation	2	2	100%	2	2	100%	N/A	N/A	N/A
taille de code (KB)	3	290	1%	7	890	1%	1	954	1%

TABLE 1. Résumé du coût de création des simulations

sumé dans le Tableau 1. Il est intéressant de noter que l'utilisation de JeB reste d'un coût raisonnable. Il peut être utilisé en tant qu'outil de routine pendant tout le développement.

5 Conclusion

La généralisation de l'usage des méthodes formelles est très dépendante des outils. Nous proposons un outil pratique pour simuler les modèles B événementiel. Notre simulateur permet d'exécuter nos modèles du *platooning* en 2 dimensions alors que les animateurs comme Brama ou ProB échouent. Nous poursuivons le travail sur JeB dans trois directions. La première est d'améliorer la bibliothèque pour B événementiel. La deuxième est la facilité d'utilisation de JeB au cours des développements. Ce travail poursuivra l'application de JeB à de nombreuses autres spécifications complexes. La troisième s'agit d'intégrer des obligations de preuves pour garantir la correction d'usage de la simulation.

Références

1. Abrial, J.R. : The B Book. Cambridge University Press (1996)
2. Abrial, J.R. : Modeling in Event-B : System and Software Engineering. Cambridge University Press (2010)
3. Leuschel, M., Butler, M. : ProB : An Automated Analysis Toolset for the B Method. Journal Software Tools for Technology Transfer 10(2), 185–203 (2008)
4. Méry, D., Singh, N.K. : Pacemaker's Functional Behaviors in Event-B. Technical report, MOSEL - INRIA Lorraine - LORIA (2009), http ://hal.inria.fr/inria-00419973
5. RODIN : Rigorous Open Development Environment for Complex Systems. Website (2012), http ://www.event-b.org
6. Wright, S. : Automatic Generation of C from Event-B. In : Workshop on Integration of Model-based Formal Methods and Tools (2009)
7. Yang, F., Jacquot, J.P. : Scaling Up with Event-B : A Case Study. In : The 3rd NASA Formal Methods Symposium (NFM'11). LNCS, vol. 6617, pp. 438–452. Springer Berlin / Heidelberg, California, USA (2011)