TURTLE : un pont entre UML et RT-LOTOS

Pierre de Saqui-Sannes[(1)(2)], Ludovic Apvrille[(3)], Christophe Lohr[(4)], Jean-Pierre Courtiat[(2)]

(1) ENSICA, 1 place Emile Blouin, 31056 Toulouse Cedex 05, France
(2) LAAS-CNRS, 7 avenue du Colonel Roche, 31077 Toulouse Cedex 04, France
(3) ENST, Institut Eurecom, BP 193, 2229 route des Crêtes, 06904 Sophia-Antipolis Cedex, France
(4) Université Concordia, Dpt. Génie Electrique, 1455 de Maisonneuve O., Montréal, Qc., H3G 1M8 Canada

desaqui@ensica.fr, ludovic.apvrille@enst.fr, lohr@ece.concordia.ca, courtiat@laas.fr

Résumé. Le profil UML TURTLE étend les diagrammes de classes et les diagrammes d'activités par des opérateurs de composition et des opérateurs temporels. La sémantique formelle du profil est exprimée en RT-LOTOS. Les outils TURTLE incluent un éditeur de diagrammes, un générateur de code RT-LOTOS et un outil de validation formelle qui permet la détection au plus tôt des erreurs de conception et la confrontation d'une architecture et de ses modèles de comportement à des exigences temporelles.

1 Positionnement de l'outil TURTLE

Face à un processus de normalisation à l'OMG qui fit dans un premier temps [6] abstraction des spécificités des systèmes temps réel [5], le développement du profil UML temps réel TURTLE (Timed UML and RT-LOTOS Environment [1]) répond au besoin de doter la notation UML d'une sémantique formelle et d'adosser la validation formelle de modèles UML temps réel à des techniques éprouvées. Le profil TURTLE est supporté par une chaîne d'outils qui va de l'édition de diagrammes à la validation d'un modèle par combinaison d'une exploration partielle sous forme de simulation et de vérification à base d'analyse d'accessibilité exploitée au moyen de «model checking» ou de minimisations. Ce prototype d'outil s'adresse à tout concepteur/trice de système temps réel, système embarqué ou protocole de communication désireux/se de détecter au plus tôt les dysfonctionnements logiques et temporels potentiels sur un modèle de haut niveau indépendant d'un langage d'implantation ou d'un système d'exploitation spécifique.

2 Du langage formel RT-LOTOS au profil TURTLE

Le profil TURTLE étend la notation UML au niveau des diagrammes de classes et des diagrammes d'activités, respectivement dédiés à la structuration d'un système en classes et à la description des comportements internes des dites classes. Il s'agit en l'espèce de *Tclasses* qui sont autant de classes stéréotypées dotées de portes de communication par rendez-vous. Les associations entre *Tclasses* sont attribuées par des opérateurs de composition, ce qui permet d'exprimer explicitement les concepts de parallélisme, séquence, synchronisation, invocation et préemption entre tâches décrites par des *Tclasses*. Les diagrammes d'activités ont été étendus par des opérateurs temporels pour décrire un délai fixe, un délai non déterministe qui associé au premier permet de travailler avec des intervalles temporels, et une offre limitée dans le temps.

Le fait d'exprimer la sémantique formelle du profil en RT-LOTOS [4] a permis de réutiliser l'outil RTL (Real-Time LOTOS Laboratory [8]) à des fins de validation comportementale et temporelle de modélisations TURTLE.

3 Un outil de conception et validation formelle

Le profil TURTLE est supporté par une chaîne d'outils :

- TTool [9] (abréviation de *TURTLE Toolkit*), développé à l'ENST, inclut un éditeur graphique de diagrammes TURTLE, un vérificateur de syntaxe, un générateur de code RT-LOTOS et, enfin, des outils de visualisation et d'analyse des résultats de simulation et de validation.
- RTL [8], développé au LAAS-CNRS, admet en entrée une spécification RT-LOTOS et permet de réaliser, soit des simulations aléatoires pour une durée donnée, soit une vérification basée sur l'analyse exhaustive des états du système considéré. Lorsque le système est fini - ou tout au moins de taille «raisonnable» - RTL peut générer un graphe d'accessibilité optimisé qui met en évidence la progression du temps.

Un lien entre les deux outils permet à un ingénieur de générer des traces de simulation ou un graphe d'accessibilité de façon transparente depuis TTool, c'est à dire sans écrire ou examiner une seule ligne de code RT-LOTOS. La chaîne de traitement mise en œuvre est représentée à la Figure 1.

TTool facilite la phase de validation en établissant une correspondance entre les actions des traces de simulation ou du graphe d'accessibilité, et celle de la modélisation TURTLE.

En termes de vérification, le rôle de RTL est de construire un automate temporisé et un graphe d'accessibilité. Ces derniers peuvent être exploités par des outils externes : Kronos [7] pour vérifier sur l'automate temporisé la satisfaction de formules logiques décrivant les propriétés requises; Aldébaran [3] pour minimiser le graphe d'accessibilité par rapport à une relation d'équivalence.

Pour faciliter l'analyse, TTool permet d'adjoindre au diagramme de classes des d'observateurs modélisés par des classes TURTLE non intrusives. La recherche des transitions d'erreur de ces observateurs au niveau des traces de simulation ou du graphe d'accessibilité permet de mettre en évidence

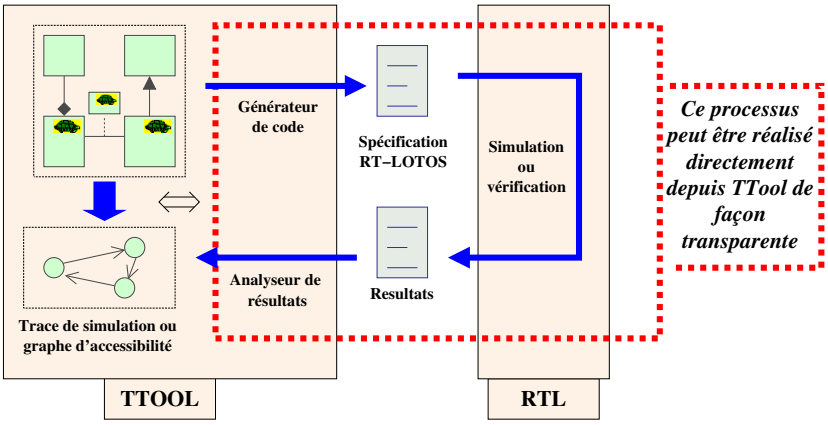

FIG. 1 – *Le processus de validation de diagrammes TURTLE*

la violation de la propriété observée. Cette technique a été exploitée pour prouver la continuité de service dans une procédure de reconfiguration dynamique de logiciel embarqué à bord de satellite [2].

4 Etude de cas : un distributeur de café

Dans les temps impartis à une démonstration, le maintenant très classique distributeur de café servira d'exemple de modélisation TURTLE complétée d'une validation comportementale et temporelle. Le distributeur considéré produit un café ou un thé après que l'utilisateur ait inséré deux pièces de monnaie. L'architecture du système comporte quatre *Tclasses* (Figure 2). *Wallet* représente la gestion de la monnaie par l'utilisateur et *Machine* implante la fonction principale du distributeur. A cela s'ajoutent deux boutons poussoirs aux noms explicites: *CoffeeButton* et *TeaButton*. Une relation de synchronisation entre *Wallet* et *Machine* permet de modéliser l'insertion et le rejet des pièces. Les relations de synchronisation entre *CoffeeButton* et *Machine* d'une part et entre *TeaButton* et *Machine* d'autre part, servent à transmettre à *Machine* un signal de type «bouton poussé».

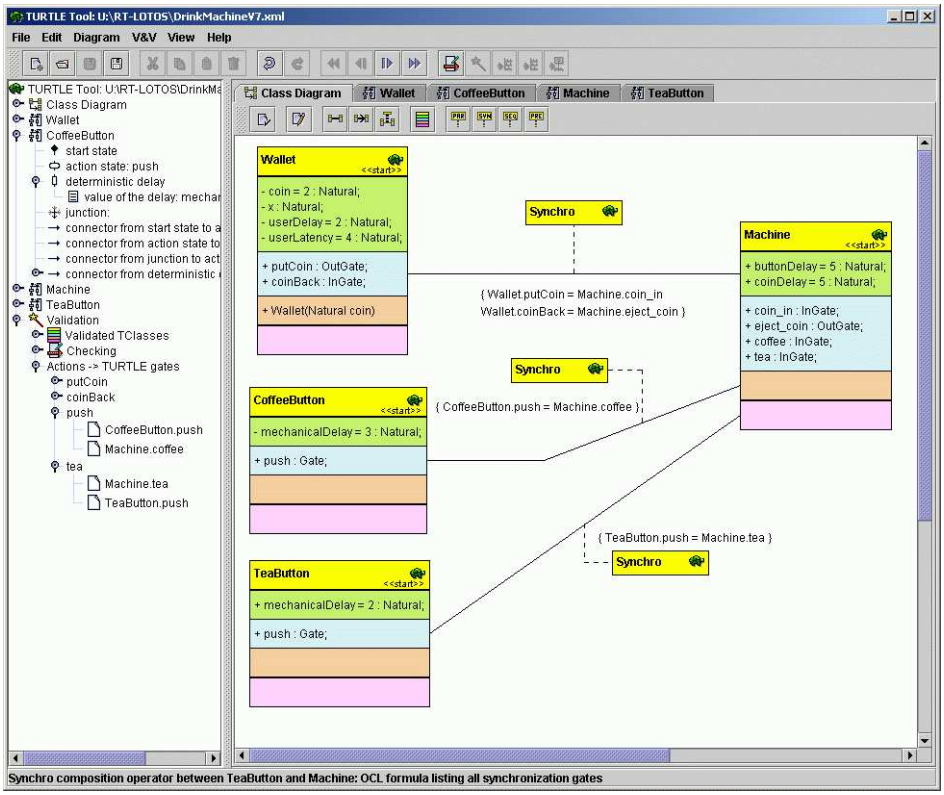

FIG. 2 – *Capture d'écran de TTool lors de l'édition du diagramme de classes du distributeur de café*

Le comportement de chacune des *Tclasses* est donné par un diagramme d'activités réalisé avec TTool. Faute d'espace, nous ne présentons pas ces diagrammes d'activités.

Le comportement logique et temps-réel de ce distributeur de café a été analysé de manière complémentaire sur des traces de simulation et des graphes d'accessibilité (Figure 3). Du point de vue logique, les graphes ont montré qu'il faut avoir au moins deux pièces dans le porte-monnaie pour obtenir une boisson. Du point de vue temporel, on a pu faire varier les attributs des classes et montrer que «*userDelay* < *coinDelay*» est une condition nécessaire à l'obtention d'une boisson.

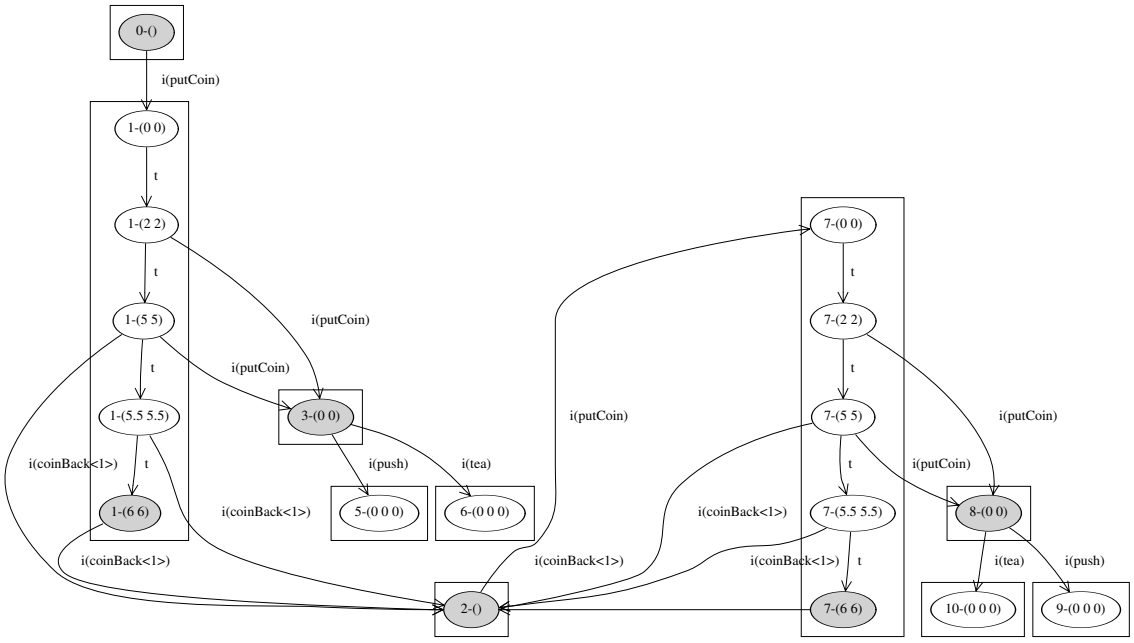

FIG. 3 – *Graphe d'accessibilité obtenu pour un portefeuille comportant deux pièces de monnaie*

Références

[1] L. Apvrille, P. de Saqui-Sannes, C. Lohr, P. Sénac, and J.-P. Courtiat. A New UML Profile for Real-time System Formal Design and Validation. In *UML'2001*, number 2185 in LNCS, pages 287–301, Toronto, Canada, October 2001. Springer.

[2] L. Apvrille, P. de Saqui-Sannes, P. Sénac, and C. Lohr. Verifying Service Continuity in a Dynamic Reconfiguration Procedure: Application to a Satellite System. *Automated Software Engineering*, 11(2), April 2004.

[3] CADP. `http://www.inrialpes.fr/vasy/cadp/`.

[4] J.-P. Courtiat, C.A.S. Santos, C. Lohr, and B. Outtaj. Experience with RT-LOTOS, a Temporal Extension of the LOTOS Formal Description Technique. *Computer Communications*, 23(12):1104–1123, 2000.

[5] Object Management Group. UML 2.0 Superstructure Specification, August 2003. `http://www.omg.org/docs/ptc/03-08-02.pdf`.

[6] Object Management Group. Unified Modeling Language Specification Version 1.5, March 2003. `http://www.omg.org/docs/formal/03-03-01.pdf`.

[7] KRONOS. `http://www-verimag.imag.fr/TEMPORISE/kronos/`.

[8] Real-Time LOTOS Laboratory. `http://www.laas.fr/RT-LOTOS`.

[9] TTOOL. `http://www.eurecom.fr/~apvrille/TURTLE/index.html`.