Génération automatique d'observateurs pour la vérification formelle d'exigences temporelles

B. Fontan*, — P. de Saqui-Sannes*,** — L. Apvrille *****

** Laboratoire d'Analyse et d'Architecture des Systèmes, Université de Toulouse*
7 avenue du Colonel Roche 31 077 Toulouse cedex 04, France

*** ENSICA, Université de Toulouse, 1 place Emile Blouin 31 056 Toulouse cedex 05, France*
{bfontan, desaqui}@ensica.fr

**** GET/ENST, 2229 route des Crêtes, B.P. 193, 06904 Sophia-Antipolis Cedex, France*
ludovic.apvrille@telecom-paris.fr

MOTS-CLÉS : *Exigences temporelles, SysML, UML, vérification formelle.*

Doté d'une sémantique formelle, outillé [4] [3], le profil UML temps réel TURTLE (*Timed UML and RT-LOTOS Environment*) couvre les phases d'analyse, conception et prototypage des systèmes distribués [1]. Son utilisation est d'autant plus justifiée lorsque le système à concevoir nécessite de vérifier formellement des exigences temporelles et de tracer les dites exigences au long du cycle de développement du système. L'objectif de ce poster est de montrer les dernières avancées en matière de traitement des exigences temporelles : expression de ces exigences dans un langage dédié, synthèse automatique d'observateurs pour guider la vérification et construction de matrices de traçabilité.

En termes de langage, TURTLE a été étendu avec les diagrammes d'exigences de SysML (*Requirement Diagram* ou *RD).* A partir d'une exigence informelle, il est possible de dériver une exigence temporelle exprimée formellement dans un langage graphique à base de *Timing Diagrams* UML enrichis de points d'observations. Ce langage graphique défini en [2] s'appelle TRDD (*Timing Requirement Description Diagram*)

Comment vérifier formellement une exigence temporelle exprimée en TRDD ? La réponse est la principale contribution de ce travail : par la synthèse d'un observateur qui sera « greffé » aux diagrammes de conception définissant l'architecture objet et les comportements des objets composant le système en cours d'étude.

Plus précisément, la figure 1 illustre le fait que l'outil TTool [4] traduit les diagrammes TURTLE dans un format intermédiaire appelé TIF (*TURTLE Intermediate Format*) qui sert de point de départ de la génération de spécifications formelles (RT-LOTOS), ces dernières pouvant alors formellement vérifiées à l'aide

de l'outil RTL [3]. C'est au niveau du TIF que les observateurs construits à partir d'exigences temporelles exprimées en TRDD vont être composés avec les diagrammes de conception du modèle du système. Le modèle TIF résultant est ensuite traduit dans une spécification RT-LOTOS sur laquelle RTL met en œuvre une analyse d'accessibilité temporisée. TTool analyse le graphe d'accessibilité et en particulier les étiquettes de transitions relatives aux synchronisations entre observateur et objet observé. De là il conclut sur la (non) satisfaction de l'exigence et reporte le résultat dans une matrice de traçabilité.

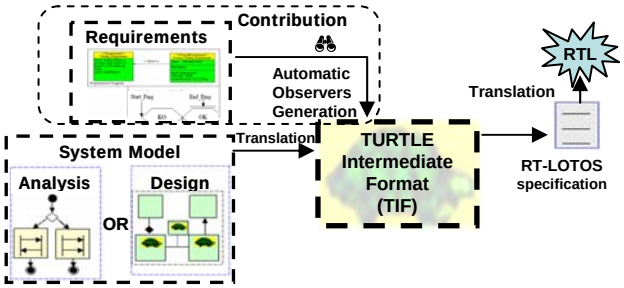

Figure 1. Positionnement de la génération automatique des observateurs

La synthèse d'observateur peut être ainsi résumée. Les informations du TRDD permettent de construire un squelette de comportement de l'observateur (cf. la table de traduction et algorithmes en [2]). Des algorithmes complémentaires sont ensuite appliqués avec pour principe de base qu'un observateur doit rester passif durant la phase d'observation et, notamment, ne pas bloquer les objets observés. Néanmoins, en cas de violation d'une exigence de haute criticité, l'observateur doit stopper l'exécution de la vérification du système. Il devient ainsi intrusif.

En conclusion générale, nous pouvons remarquer que notre discussion s'est focalisée sur la vérification formelle à base de spécifications formelles RT-LOTOS générées par TTool. Ce même outil génère également du code Java en exploitant les diagrammes de composants et de déploiement en plus des diagrammes d'analyse et de conception déjà mentionnés [1]. Le concept d'observateur utilisé ici en termes de vérification pourrait être transposé à des « sondes » applicables aux prototypes de code Java engendrés par TTool.

Bibliographie

[1] L. Apvrille, P. de Saqui-Sannes, R. Pacalet, A. Apvrille, « Un environnement de conception de systèmes distribués basé sur UML », Annales des Télécommunications, Vol. 61, n 11/12, pp. 1347-1368, Nov. 2006.

[2] B. Fontan, P. de Saqui-Sannes, L. Apvrille., « Génération automatique d'observateurs à partir d'exigences temporelles », Rapport Technique DMI ENSICA, Toulouse, février 2007. http://dmi.ensica.fr/spip.php?article776

[3] Real-Time Lotos Laboratory (RTL), http://www.laas.fr/ RT-LOTOS

[4] TURTLE toolkit (TTool), http://labsoc.comelec.enst.fr/turtle/