Modèles et langages pour la programmation par composants

Pascal Fradet, Alain Girault, Gregor Goessler

INRIA Rhône-Alpes

Définition : composant

Programme informatique encapsulé dans une interface.

Cette interface indique :

- quel est le type du composant,
- éventuellement quel est son comportement,
- et quels sont les ports d'entrée et de sortie

Introspection : Opération qui permet à un composant de connaître son propre contenu : sous-composants, comportement...

Programmation par composants

Consiste à concevoir un système complexe en terme de réseau de composants reliés entre eux par des connecteurs, qui collaborent pour réaliser les fonctionnalités désirées.

Connecteur : implémente un protocole spécifique de communication. Il permet de connecter des composants.

Composition : deux composants peuvent être composés pour former un composant plus complexe.

Classification

Modèles formels

Modèles non formels

Modèles statiques

Modèles dynamiques

Modèles formels statiques

Reactive Modules

[Alur & Henzinger, LICS 1996]

Modèle de spécification.

Il combine exécution synchrone et asynchrone.

Plusieurs niveaux d'abstraction spatiale et temporelle

Deux opérateurs d'abstraction : cachage et abstraction

Cela permet de regrouper un nombre arbitraire de pas de calcul en un seul pas

Ptolemy II

[Lee et al, RR UCB, 2003]

Langage de modélisation hétérogène et hiérarchique.

Plusieurs modèles de calcul (« domaines »).

Les composants (« acteurs ») interagissent en fonction du domaine.

Hiérarchie : un composant dans un domaine A peut être raffiné en plusieurs sous-composants

Hétérogénéité : ces sous-composants peuvent être dans un autre domaine B

Polymorphie : certains composants peuvent fonctionner dans plusieurs domaines.

Sémantique : basée sur le « tagged signal model » [Edwards, Lavagno, Lee & Sangiovanni-Vincentelli, PIEEE, 1997].

Prometheus

[Goessler, RT-TOOLS, 2001]

Outil de modélisation pour spécifier et composer des systèmes temps-réel.

Chaque composant a des propriétés de sûreté, vivacité, et une priorité

PROMETHEUS établit formellement la propriété globale de sûreté, de vivacité, et la cohérence des priorités.

Voir [Goessler & Sifakis, SCP, 2004] pour des recherches plus récentes.

Modèles formels dynamiques

Shift

[Deshpande, Göllü & Semenzato, IEEE TAC, 1998]

Langage orienté objet.

Les composants sont des automates hybrides.

Dynamicité : un composant peut créer dynamiquement d'autres composants, un composant peut se suicider, et des composants peuvent se synchroniser dynamiquement.

SHIFT est utilisé uniquement pour faire de la simulation.

Il n'existe pas de compilateur.

Reactive Machines

[Susini, Boussinot & Hazard, RTCSA, 1998]

Modèle dédié à la programmation réactive

Composant = machine réactive (RM).

Connecteur = synchroniseur (SYNC).

Une seule sémantique de la communication dans les SYNC : synchrone

Mais sans réaction instantanée.

Implémentation dans SUGARCUBES [Boussinot & Susini, SPE, 1998].

Modèles non formels statiques

Architecture Description Languages

Synthèse jusqu'à 2000 : [Medvidovic & Taylor, IEEE TSE, 2000]

ADAGE [Coglianese & Szymanski, AGARD, 1993]

AESOP [Garlan, Allen & Ockerbloom, SIGSOFT, 1994]

ARCHJAVA [Aldrich, submitted, 2004]

C2 [Medvidovic, Oreizy, Robbins & Talyor, SIGSOFT, 1996]

METAH [Binns & Vestal, RTSS, 1993]

RAPIDE [Luckham et al, IEEE TSE, 1995]

SADL [Moriconi, Quian & Riemenschneider, IEEE TSE, 1995]

UNICON [Shaw et al, IEEE TSE, 1995]

WRIGHT [Allen & Garland, ICSE, 1994]

Méta ADL : ACME [Garland, Monroe & Wile, CASCON, 1997]

Metropolis

[Balarin et al, CODES, 2002]

Environnement de conception.

Un langage (« meta-model »), une méthodologie, et des outils.

Raffinement des spécifications de haut niveau jusqu'à l'implémentation sur une plate-forme.

Séparation des aspects calcul, communication et coordination.

Koala

[van Ommering, van der Linden, Kramer & Magee, Computer, 2000]

Modèles à composants logiciels.

Séparation de l'implémentation et de la configuration.

Introspection pour faire des composants adaptatifs.

Modèles non formels dynamiques

Enterprise Java Beans

Architecture serveur à composants.

Très utilisé dans l’industrie.

Fractal

[Bruneton, Coupaye & Stefani, WCOP, 2002]

Modèle basé sur le Kell Calculus [Stefani, RR INRIA, 2003] : Fraktal

Chaque composant consiste en deux parties : la partie fonctionnelle et un contrôleur qui gère des propriétés non-fonctionnelles : introspection, configuration, sécurité...

Les contrôleurs permettent une reconfiguration dynamique.

Inversion du contrôle : configuration des composants par une entité séparée.

JULIA en est une implémentation du modèle FRACTAL en JAVA.

THINK en est une implémentation du modèle FRACTAL en C.

RMA

Reconfiguration Management Architecture [Moessner, Hope, Cook, Tuttlebee & Tafazolli, IEICE TCOM, 2002]

Sorte d’ADL dédiée à la radio logicielle.

Permet la reconfiguration des couches réseau.

Différentes fonctinalités côté terminal et côté réseau.

Implémentation middleware.

Langages et outils pour programmer les composants

Quelques exemples de langages

Langages synchrones [Benveniste et al, PIEEE, 2003]

Interface Automate [de Alfaro & Henzinger, FSE, 2001]

E-LOTOS [Garavel & Sighireanu, FMICS, 1998]

Un Langage pour la Mobilité (ULM) [Boudol, ESOP, 2004]

IF-2 [Bozga, Graf & Mounier, CAV, 2003]

SYSTEMC [Grötker, Liao, Martin & Swan, Kluwer, 2002]

Mathlab/Simulink

CORBA components [OMG, 2002]

Unified Modeling Language (UML) [Selic & Rumbaugh, RR, 1998]

ERLANG [Armstrong, Virding, Wikström & Williams, Prentice Hall, 1996]