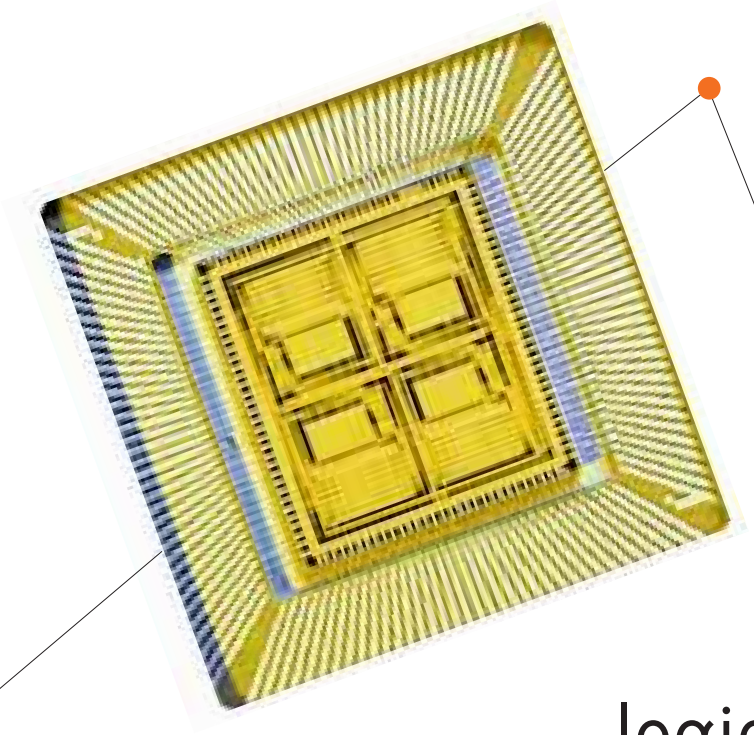
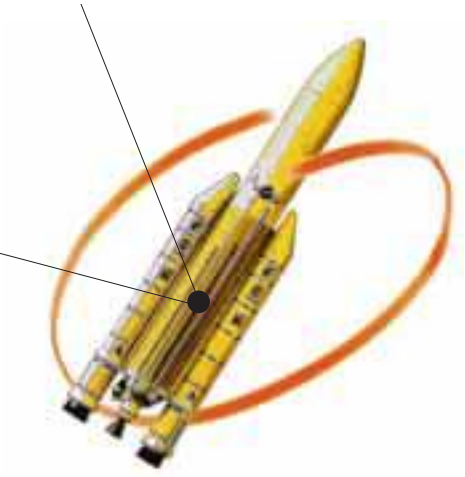
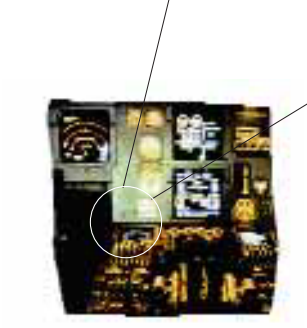


logiciels : objectif zéro faute



Écrire des programmes sans erreurs de syntaxe, certains programmeurs y arrivent du premier coup. Mais en écrire qui soient à la fois complexes et dépourvus de fautes logiques ou de bogues, voilà qui relève de l'exploit, sinon de l'utopie. Qu'à cela ne tienne : les outils de vérification et de test se multiplient et se perfectionnent.



À l'heure où l'informatique est omniprésente, on ne peut s'empêcher d'être par moments anxieux : les logiciels qui commandent le réacteur nucléaire tout proche vont-ils bien fonctionner en toutes circonstances ? Ceux qu'utilise ma carte bancaire ne risquent-ils pas de me dépouiller sans que je m'en aperçoive ? Tous les circuits électroniques de l'avion que je vais prendre sont-ils parfaitement compatibles entre eux ? Rassurons-nous : les logiciels en service sont en général préalablement validés et testés, et cela d'autant mieux que leur dysfonctionnement peut avoir des conséquences graves. Mais les validations et les tests demandent du temps, des efforts... qui se paient. Qui plus est, comment être sûr que tout a été vérifié ?

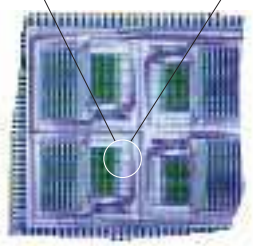
Là réside l'intérêt des recherches en vue de systématiser et d'automatiser ces procédures. Le problème se pose surtout pour les systèmes distribués ou parallèles, qui comportent plusieurs éléments en communication et en interaction. Il faut garantir, par exemple, qu'il n'y aura pas de blocages dans le protocole de communication, situation où deux logiciels attendent l'un de l'autre une information nécessaire à la poursuite de leur exécution, ou que plusieurs logiciels n'auront pas besoin d'écrire simultanément à une même adresse

De telles recherches sont menées par plusieurs projets au sein de l'INRIA, et peuvent comporter un aspect théorique important. En effet, l'une des démarches consiste à analyser la sémantique du langage de programmation afin d'obtenir des modèles formels des logiciels considérés. Cette étape facilite l'élaboration d'outils de vérification proprement dite des logiciels. La vérification peut en effet se faire en essayant de prouver formellement des propriétés et des critères de bon fonctionnement. Mais la vérification repose le plus souvent sur la force brute : examiner, pour chaque jeu de données, le comportement du système. Cela revient à explorer tous les états par lesquels passe le système (que l'on modélise par un "graphe d'états"). Le problème, c'est que le nombre de configurations à explorer est immense, même si les éléments du système sont à nombre fini d'états – ce qui, en pratique, est souvent le cas. La difficulté est donc de réaliser l'exploration de façon "intelligente", en prouvant certaines propriétés génériques, en se restreignant à des cas pertinents ou à des sous-systèmes, etc., cela afin d'éviter une vérification exhaustive bien trop gourmande en temps ou en mémoire. Enfin, il s'agit de développer des logiciels maniables par des utilisateurs non spécialistes, aspect qui est loin d'être négligeable si l'on veut séduire les industriels.

L'un des projets qui œuvrent en ce sens est **MEIJE,** particulièrement autour du langage Esterel qu'il a développé. Il s'agit d'un langage de programmation adapté aux systèmes de contrôle, qui doivent réagir en temps réel à un environnement externe et qui fonctionnent de manière "synchrone" entre composants parallèles. Les applications abordées par MEIJE portent donc naturellement sur les systèmes embarqués (avions, voitures, circuits matériels, etc.), domaine où la fiabilité des logiciels est cruciale. MEIJE collabore ainsi avec Dassault Aviation sur la modélisation des systèmes embarqués sur des avions, dans le but de leur vérification. De même, dans le domaine de la microélectronique, MEIJE expérimente l'utilisation des techniques Esterel pour la synthèse et la conception de circuits intégrés : ces dispositifs équivalent, du point de vue formel, à des systèmes logiciels parallèles.

L'action **VASY,** elle, contribue au développement d'une boîte à outils de compilation et de vérification, dénommée CADP et diffusée à plus de 165 sites dans le monde. Cet environnement s'articule autour de Lotos, langage permettant de décrire les protocoles pour systèmes distribués et parallèles "asynchrones". Le choix s'est porté sur ce langage car il constitue le seul de ce type ayant le statut de norme internationale et dont la sémantique est rigoureusement définie. Les applications de la boîte à outils CADP vont des bases de données aux réseaux de communication, en passant par les protocoles de sécurité cryptographique pour le commerce électronique. CADP est actuellement utilisée par Bull (dans le cadre du GIE Dyade, associant Bull et l'INRIA) afin de valider les architectures multiprocesseurs que cette société élabore pour les futurs serveurs haut de gamme. CADP a aussi permis de déceler une erreur dans le bus "Firewire" (norme IEEE 1394), bus série à haut débit pour les microordinateurs auquel se sont ralliés les principaux constructeurs informatiques, éditeurs de logiciels et fabricants d'appareils audiovisuels. Cette erreur était quasiment impossible à détecter à la main, puisqu'elle apparaît au bout d'une séquence bien précise d'une cinquantaine d'opérations !

Quant au projet **PAMPA,** il développe entre autres des outils de validation dans le cadre de la méthode orientée objets UML (*Unified Modeling Language*), méthode très largement utilisée dans le monde industriel pour construire des programmes et les modéliser. Un des points forts de PAMPA est la mise au point, en commun avec le laboratoire Vérimag du CNRS, d'un logiciel appelé TGV (*Test Generation with Verification technology*) qui produit automatiquement des tests de conformité. Armé de cet outil, PAMPA collabore, par exemple, au projet européen MODISTARC, dont l'objectif est de mettre en place une méthodologie de tests pour les protocoles de communication et de gestion, embarqués sur automobiles. Mais, une grande part des applications des outils de validation concerne le domaine des télécommunications, où les systèmes sont, par essence, répartis. Aux côtés de quatre autres équipes de l'INRIA, PAMPA coopère ainsi avec la société Alcatel pour la conception d'outils de manipulations formelles au sein d'une chaîne de développement de logiciels de télécommunications (définie par Alcatel).



Contacts chercheurs

MEIJE (1) : Robert de Simone, robert.de_simone@inria.fr
VASY : Hubert Garavel, hubert.garavel@inria.fr
PAMPA (2) : Claude Jard, claude.jard@inria.fr

(1) projet commun avec le Centre de Mathématiques Appliquées (CMA : Ecole des Mines de Paris)