# Préface

Au cours des trente dernières années, le temps réel s'est progressivement établi comme une discipline à part entière qui rassemble aujourd'hui une forte communauté issue à la fois du monde académique et de l'industrie. Ce traité en deux volumes a pour objectif de mieux faire connaître cette discipline: ses enjeux, les méthodes et formalismes qui lui sont spécifiques, les outils existants, les résultats connus et, naturellement, les recherches encore à mener.

L'informatique temps réel est une discipline qui concerne le contrôle ou le suivi de systèmes qui évoluent dynamiquement. Ces systèmes peuvent être des équipements de production ou de transport de produits, de matières, d'énergie ou d'informations. À ce sens, indubitablement, le temps réel entretient des liens étroits avec d'autres disciplines : l'automatique qui définit les cahiers des charges sous-jacents, la recherche opérationnelle pour les techniques de maîtrise et d'organisation de systèmes discrets, l'ingénierie du logiciel pour la mise en œuvre d'un développement sûr et de qualité, les processus stochastiques en raison de la connaissance le plus souvent incomplète ou imparfaite des systèmes et de leur environnement. En fait, ce qui fait l'essence de l'informatique temps réel est une exigence de garantir le niveau de performances temporelles requis par la dynamique et la sûreté des systèmes contrôlés ou surveillés : les résultats doivent non seulement être corrects en valeur mais ils doivent aussi être produits et délivrés aux bons instants. Le plus souvent cette exigence se traduira en des contraintes sur les dates de fin d'exécution des activités, on parle alors de contraintes d'échéances. Garantir le respect de ces contraintes repose sur un ensemble de techniques de vérification formelle dont la plupart sont présentées dans le tome 1 de ce traité.

Les systèmes temps réel sont généralement « embarqués », c'est-à-dire situés à l'intérieur même de l'environnement avec lequel ils doivent interagir, comme un calculateur dans une voiture ou un avion. En plus des exigences de performances temporelles, ils sont alors assujettis à de fortes contraintes d'encombrement, de coût et parfois d'énergie, ce qui limite d'autant la puissance de calcul et l'espace mémoire disponibles et constitue une différence fondamentale avec l'informatique traditionnelle où globalement les ressources disponibles augmentent exponentiellement au rythme de la loi de Moore. Les domaines d'applications intégrant de manière concrète l'informatique temps réel sont nombreux et de natures variées : contrôle de systèmes automatisés de production, surveillance de systèmes, aide au pilotage de véhicules, comme par exemple des systèmes électroniques de freinage offrant une meilleure sécurité, ou contrôle de flux dans les réseaux tels l'internet pour des applications critiques de télécommande, etc. Et l'on voit que le temps réel est loin d'être une discipline désincarnée de la réalité et qu'il peut être source de progrès dans notre quotidien.

Ce traité est organisé en deux volumes et vingt et un chapitres, rédigés pour la plupart par des conférenciers de la quatrième École d'été Temps Réel 2005 (ETR'05). Cette manifestation, soutenue par le GDR ARP (Architecture, Réseau et système, Parallélisme) du CNRS et l'INRIA, est un événement phare pour la communauté temps réel francophone qui a rassemblé, lors de sa dernière édition à Nancy, 110 chercheurs et industriels du 13 au 16 septembre 2005.

Le premier volume du traité, intitulé *Systèmes Temps Réel : Techniques de Description et de Vérification*, a pour objet de faire connaître les principaux formalismes et les outils logiciels associés qui offrent des solutions pour spécifier un système temps réel et vérifier, avant le déploiement du système, le respect des propriétés attendues. Les méthodes formelles constituent à l'évidence un facteur de progrès considérables vers une informatique « sûre de fonctionnement » mais leur utilisation ne pourra se faire à grande échelle que si les outils logiciels correspondants sont développés. C'est pourquoi le premier tome est délibérément orienté « outils » de façon à proposer des solutions concrètes à l'utilisateur potentiel de méthodes formelles. Au-delà des outils, le but est également de faire mieux connaître les formalismes sous-jacents et, c'est notre souhait, de montrer comment les méthodes formelles peuvent améliorer la qualité et la sûreté des systèmes.

Ce second tome du traité, intitulé *Systèmes Temps Réel : Ordonnancement, Réseaux, Qualité de Service*, est consacré aux mécanismes exécutifs permettant l'obtention d'une « qualité de service » temps réel. En particulier, le problème crucial du choix, de la configuration et de l'implémentation des stratégies d'ordonnancement de tâches sera largement traité. Une seconde partie de ce tome est dédiée aux réseaux et protocoles de communication temps réel qui permettent à des entités distantes d'un même système de s'échanger des informations dans des délais

contraints. C'est typiquement le cas, étudié en détail dans le chapitre 10, d'applications embarquées dans les véhicules qui sont distribuées sur plusieurs calculateurs connectés sur un réseau et qui s'échangent des informations dont la durée de vie est fortement limitée dans le temps. À titre d'exemple, certains véhicules haut de gamme intègrent jusqu'à soixante-dix calculateurs entre lesquels s'échangent environ deux mille cinq cents informations ; un grand nombre de ces informations sont soumises à des contraintes de temps strictes qui, pour certaines, peuvent être de l'ordre de quelques millisecondes.

Le premier chapitre, rédigé par Pascal Richard et Frédéric Ridouard, fait un tour d'horizon des algorithmes d'ordonnancement temps réel de tâches dans le contexte monoprocesseur. C'est sans conteste, le domaine du temps réel qui a donné lieu au plus grand nombre de travaux au cours des trente dernières années et des solutions efficaces, voire optimales, existent pour les contextes applicatifs les plus courants.

Joël Goossens, dans le second chapitre, traitera du cas où l'ordonnancement est réalisé sur une plate-forme multiprocesseur. Ce choix de conception peut s'avérer judicieux pour des raisons de coûts (à puissance de calcul équivalente, une solution multiprocesseur est souvent moins coûteuse) mais aussi de facilité et de rapidité de développement par exemple lorsque des processeurs dédiés, implantant des fonctionnalités de haut niveau, sont utilisés. L'ordonnancement temps réel multiprocesseur est encore aujourd'hui un domaine de recherche largement ouvert qui pourrait donner lieu à des développements importants dans les années à venir compte tenu des besoins pratiques et de l'intérêt grandissant de la communauté scientifique.

Le chapitre 3, rédigé par Daniel Simon, Olivier Sename et David Robert, se situe dans le contexte de la conception des applications industrielles de contrôle-commande. L'idée défendue est de concevoir conjointement algorithmes de commande et algorithmes d'ordonnancement, en étudiant l'influence des choix faits sur le processus physique sous contrôle. Ce sont des travaux d'importance pratique qui comblent un vide que l'on peut probablement attribuer historiquement au manque d'interactions entre les communautés informatique temps réel et automatique. Les choses ont aujourd'hui changé à cet égard et ce domaine de recherche est actuellement en plein essor.

Dans le chapitre 4, Nicolas Navet et Bruno Gaujal, présentent les stratégies d'ordonnancement permettant de minimiser la consommation en énergie des processeurs. Ce problème est devenu crucial dans la conception de tous les systèmes électroniques dont l'alimentation est assurée par des batteries. En effet, l'augmentation continue des performances et des fonctionnalités nécessite l'utilisation de processeurs fonctionnant à des fréquences toujours plus élevées, et donc consommant de plus en plus d'énergie, alors que parallèlement la technologie

des batteries ne progresse pas au même rythme. L'ordonnancement sous contrainte d'énergie étend la problématique de l'ordonnancement en intégrant une nouvelle dimension qui est la vitesse du processeur, ce qui remet en cause beaucoup des solutions existantes.

Le problème de l'estimation des pires temps d'exécution de code logiciel est l'objet du chapitre 5 rédigé par Isabelle Puaut. Toutes les techniques de vérification de propriétés temps réel présupposent des informations sur les temps d'exécution, et, lorsque que l'on recherche des garanties déterministes, la connaissance des pires temps d'exécution des tâches est nécessaire. L'enjeu est de taille : plus la connaissance des pires temps sera précise et plus l'architecture matérielle sera dimensionnée au plus juste. En pratique, la complexité croissante du matériel (différents niveaux de caches, *pipelining*, etc.) rend ce problème extrêmement délicat. Ce chapitre fournit des éléments de réponse et propose des voies de recherche, par exemple, sur des processeurs ou des compilateurs spécialisés, pour estimer de façon plus exacte les pires temps d'exécution.

Lorsque l'on fait un tour d'horizon des langages de programmation les plus couramment utilisés, on est frappé par le fait que peu permettent de gérer la concurrence des activités et qu'aucun ne permette de manipuler le temps, et, en particulier, les contraintes de temps, de façon explicite. Le langage Java et ses diverses extensions temps réel proposent à cet égard des initiatives louables : modèle de concurrence, synchronisation et partage de ressources, gestion mémoire prévisible, etc. Marc Richard-Foy, dans le chapitre 6, présente les fonctionnalités de Java et de certaines de ses extensions qui présentent un intérêt pour le développement d'applications temps réel. Des idiomes de programmation, démontrant l'utilisation concrète des primitives du langage, sont également proposés. Ce chapitre apporte de précieux éléments de réflexion pour décider de l'adéquation du langage Java à des besoins particuliers de temps réel.

Les chapitres 7 à 11 sont consacrés aux réseaux et protocoles de communication temps réel. Zoubir Mammeri dans le chapitre 7 introduit le concept de qualité de service temps réel et, en particulier, identifie les diverses techniques qui permettent son obtention, que ce soit sur des réseaux locaux ou des réseaux ouverts comme l'Internet où la multiplicité des flux concurrents rend le problème beaucoup plus difficile. Dans ce contexte, un élément de solution est proposé par l'expression de contraintes de qualité de service dites « (m,k) firm » qui est le sujet du chapitre 8 rédigé par Ye-Qiong Song, Anis Koubaa et Jian Li. Un flux est dit « (m,k) firm » si au moins m paquets parmi k paquets consécutifs sont transmis en respectant leur échéance. Cette contrainte exploite le fait que beaucoup d'applications, dans le domaine du multimédia en particulier, sont capables de tolérer ou de s'adapter, dans une certaine limite, spécifiée par la contrainte (m,k)-firm, à une variation des performances du réseau. L'utilisation de ce type de contraintes permet une

dégradation contrôlée de la qualité de service de flux en concurrence lorsque les ressources réseaux sont insuffisantes.

Jean-Pierre Thomesse, dans le chapitre 9, définit les principes généraux des protocoles de niveau *Medium Access Control* (MAC) et identifie les principales normes et produits existants. Dans un profil de communication, les protocoles de niveau MAC ont cette fonction essentielle de gérer les accès concurrents à la ressource partagée qu'est le canal de communication. Selon l'algorithme de partage du canal, les délais d'attente vont différer, ce qui a un impact direct sur la qualité de service temps réel. Les protocoles de niveau MAC ont donné lieu dans les vingt-cinq dernières années à un nombre considérable de recherches et de produits commerciaux, et des solutions réellement efficaces, et souvent standardisées, sont aujourd'hui disponibles. C'est le cas en particulier dans le domaine des réseaux embarqués dans les véhicules qui fait l'objet du chapitre 10 rédigé par Françoise Simonot-Lion et Nicolas Navet. Les fonctions embarquées dans les véhicules sont de plus en plus nombreuses et certaines, relatives au freinage, à la direction ou la transmission, sont critiques du point de vue de la sécurité. Cette préoccupation de sécurité est naturellement partagée avec d'autres secteurs des transports, comme l'avionique ou le ferroviaire, mais les contraintes de coûts propres à l'industrie automobile nécessitent le développement de solutions techniques spécifiques, qui sont examinées dans ce chapitre.

La technologie de réseau local la plus répandue est sans conteste aujourd'hui Ethernet et diverses propositions ont été faites pour l'adapter à une utilisation en informatique industrielle où des contraintes de temps et, plus généralement de sûreté de fonctionnement, existent. Jean-Dominique Decotignie, dans le onzième et dernier chapitre de ce tome 2, analyse les forces et faiblesses des solutions en concurrence et donne un aperçu sur ce qu'il reste à développer.

Pour conclure, je voudrais exprimer ma gratitude à M. Jean-Charles Pomerol, directeur éditorial, pour la confiance qu'il a accepté de me témoigner et remercier chaleureusement les auteurs et les relecteurs des chapitres qui ont offert de leur temps et partagé leur expertise dans la rédaction de ce livre. Je remercie également Françoise Simonot-Lion et Olivier Zendra pour leurs remarques et suggestions sur ce chapitre introductif.

Enfin, j'espère que cet ouvrage sera, pour vous, chers lecteurs, une source d'information complète, fiable et bien documentée sur les systèmes temps réel. Je vous en souhaite une excellente lecture,

Nicolas NAVET