Applications Concurrentes :

Conception,

Outils de Validation

ACCOV_B

Présentation du cours

Plan du tome 1

Rappel : le tome 2 traite des outils de spécification et de validation

ACTUALITE DU COURS

Avec l'extension du Web, de Java et de Ada95, des outils de programmation réseau, avec la programmation par événements associée à des "threads" et à des actions réparties, avec l'approche client-serveur, avec le poste à poste (« peer to peer »), il va être de plus en plus facile de faire de la programmation concurrente ou parallèle. C'est facile a programmer, mais c'est difficile à faire correctement. Il y aura beaucoup de programmes faux parce que concurrents.

Il faut sensibiliser les auditeurs aux bons paradigmes, aux bons "patterns" de concurrence ou de parallélisme, bons parce que bien connus et bien éprouvés par ceux qui développent des programmes concurrents depuis longtemps, mais aussi parce qu'ils peuvent être prouvés et validés avec des outils de validation. Or ces outils sont disponibles au CNAM et sur le Web, et ils sont utilisés pour des programmes et des problèmes réels. Il est donc d'actualité d'enseigner ces "patterns" et leur évaluation, pour qu'ils soient utilisés dans le développement d'applications et de systèmes comportant des processus concurrents.

Ce n'est pas seulement nous qui le disons, c'est aussi Doug Lea, le célèbre auteur de : *Concurrent Programming in Java : Design Principles and Patterns*, Addison Wesley 97, et professeur à l'université de New-York. Dans la revue "IIE Concurrency", Parallel, Distributed & Mobile Computing, vol.6, n° 4, oct.-dec. 1998, à la rubrique Object-Oriented Applications, il a publié un article "Patterns and the democratization of concurrent programming", pages 11-13, où il écrit :

"The number of concurrent programmers has skyrocketed over the past few years and will continue to do so."..."Although only a small proportion of code found in today's production programs, applications, and systems directly exploits concurrency, every programmer should at least be aware of concurrency."..."Concurrent programming's diversity appears in the wide range of designs and design approaches seen in practical applications."..."Concurrency techniques have become indispensable for programmers who create the highly available services and reactive applications that are now commonplace."..."Despite its increasing centrality, concurrency remains intrinsically more complex, unfamiliar, and prone to design and programming errors than most other aspects of programming."..."Many programmers increasingly cope with new, concurrency-driven complexities. And without expert guidance, they're also expected to occasionally fail."..."However, these multithreading constructs - threads, locks, monitors, sockets, and related interprocess messaging support - are now commonly accepted as the standard basis for building practical concurrent programs. And they can serve as building blocks for nearly any other style of concurrency support a programmer desires (such as Ada tasks)."..."Concurrent design-pattern descriptions thus typically address design forces and integrity issues."..."Temporal dimensions of correctness introduced by concurrency - safety and liveness (including fault-tolerance) - are central concerns in any concurrent design and its implementation."

En s'intéressant, au premier chef, aux systèmes multiprocesseurs (architectures à couplage fort), ce cours peut utilement compléter le cours d'approfondissement "Systèmes et Applications répartis" qui s'applique exclusivement aux aspects soulevés par la répartition (architectures à couplage faible). Ce cours présente aussi un grand intérêt pour les applications en temps réel, que l'informatique y soit embarquée ou enfouie, où les conséquences des fautes de programmation sont souvent dramatiques.

OBJECTIFS DU COURS

Au niveau de l'analyse opérationnelle des applications coopératives, centralisées, temps réel ou distribuées, appréhender les structures qui concernent l'exécution des processus concurrents, le partage des ressources et des objets et qui expriment leurs relations et leur contrôle au moment de l'exécution. Dégager les paradigmes de la concurrence et les patrons de construction ("design patterns") qui sous-tendent les architectures logiques de ces applications, en étudier la programmation avec les mécanismes des langages, apprendre à en faire la validation en utilisant des outils disponibles, connaître les principaux motifs d'erreur de parallélisme ("error patterns").

Savoir ainsi maîtriser les difficultés de la concurrence lors de la construction, la modification, l'intégration ou la maintenance d'applications comportant des processus coopérants, locaux ou distants. Disposer de techniques utiles pour réceptionner les systèmes informatiques de ces applications, pour les soumettre à des tests couvrant les principaux cas d'erreurs de gestion de la concurrence, et pour faire la recette par validation de travaux en sous-traitance ou en régie.

ORGANISATION DU COURS

Cette valeur comprend une présentation informelle de problèmes de parallélisme, de concurrence, de distribution et de temps réel, en s'appuyant sur des langages de programmation intégrant concurrence et répartition (comme Java et Ada95). On développe plus particulièrement les solutions concernant le partage des ressources et la concurrence entre les processus. Puis on présente des outils de validation de propriétés de sûreté (non interblocage, "no deadlock") et de vivacité (non famine, équité, respect des échéances). On met l'accent sur les performances et le respect des contraintes temporelles des solutions présentées et validées. On utilise, pour valider les problèmes présentés informellement, des outils tels que GreatSPN, CPN-AMI, DesignCPN (disponibles sur le Web), et Quasar (produit au CNAM).

Cet enseignement intègre aussi, selon la demande des auditeurs, des présentations et analyses d'applications transactionnelles, d'applications client-serveur avec Corba, Java, Ada95 ou autres, d'applications temps réel avec des contraintes d'échéances temporelles, ou d'expression de qualité de service (QoS) pour le multimédia. On examine de même le traitement de la concurrence des processus et du partage des ressources dans les plates-formes standard (Posix, Unix, Linux, WindowsNT, MacOs8, Chorus, Mach, Jaluna), les plates-formes objets (Corba, Java), plates-formes temps réel (VxWorks, pSoS, Vrtx, Rtc, LynxOs, T-Smart), des supports de mobilité ou de coopération poste à poste (pair à pair, « peer to peer »).

CONTENU

Nature et structuration des applications comportant des processus concurrents.

Contrôle de concurrence dans les moniteurs transactionnels, dans les systèmes d'information, dans les applications temps réel, dans le multimédia. Contrôle de concurrence dans les plates-formes systèmes standard (Posix, Unix, Linux, WindowsNT, MacOs8, Chorus, Mach, Jaluna,...), les plates-formes objets (Corba, Java), plates-formes temps réel (VxWorks, pSoS, Vrtx, Rtc, LynxOs, T-Smart), PVM, MPI, les mobiles, le "groupware".

Regroupement et hiérarchie de processus, modèle client-serveur, modèle transactionnel. Gestion mémoire des processus et des modules partagés. Allocation dynamique de la mémoire. Partitionnement pour la répartition et la distribution.

Les paradigmes de la concurrence et les archétypes de programmation ('design patterns').

Exclusion mutuelle, élection, producteur-consommateur, lecteurs-rédacteurs, client-serveur, problèmes liés aux pannes, diffusion atomique ordonnée, interblocage, famine, terminaison.

Mécanismes de bases (processus, sémaphores, moniteurs, la classe thread et les méthodes synchronized dans Java, tâches et objets protégés dans Ada95, communication synchrone et asynchrone, messages, boîtes aux lettres, tableau noir, invocation à distance, rendez-vous, RPC et proxys, transaction). Programmation des paradigmes. Modularité et objets concurrents.

La spécification des problèmes et les outils de vérification et de validation des solutions.

Aperçu des méthodes de spécification : automates, états-assertions, méthode B, logique temporelle, réseaux de Petri.

Techniques d'analyse des réseaux de Petri (analyse structurelle), "Model-checking" (logique temporelle), automates synchronisés. Utilisation des outils de simulation et de validation : GreatSPN, CPN-AMI, DesignCPN (sur le Web).

Ouvertures.

Évaluation quantitative : temps de réponse, échéance, gigue. Contraintes de précédence. Qualité de service.
Traite-interruptions, transfert asynchrone (signaux, AST), modes de fonctionnement, sécurité, distribution et placement.
La concurrence dans la gestion des flux d'ateliers (« work flow management »).

Méthode d'enseignement (cours + ED +TP intégrés)

Présentation et discussion d'exemples de schémas de concurrence exprimés en Ada, Java, Posix... et fournis sous forme de programmes exécutables.
Validation des exemples en utilisant des outils spécifiques.
Projet en utilisant des logiciels libres : étude, programmation et validation des schémas de concurrence dans des architectures logiques simples comportant des processus concurrents et de l'allocation dynamique.
A cause du projet et du tutorat qu'il implique, le nombre d'auditeurs inscrits est limité à 30.

Polycopiés Ouvrages Multimédia

BESANCENOT J., CART M., FERRIÉ J., GUERRAOUI R, PUCHERAL P., TRAVERSON B. *Les systèmes transactionnels*., 415 p. (Hermès 1997)
BLAIR G., STEFANI.J.B. *Open distributed processing and Multimedia*., 452 p. (Addison-Weslay 1998)
[BREGUET 99] P. Breguet, L. Zaffalon, *Programmation séquentielle avec Ada* 95, Presses Polytechniques et Universitaires Romandes 1999
BURNS A., WELLINGS A., *Concurrency in Ada*., 396 p.(Cambridge University Press 1995)
[CHARON 00] I. Charon, *Le langage Java*, Hermès Science 2000
[CROCUS 75] CROCUS, *Systèmes d'exploitation des ordinateurs*, Dunod 1975 (364 pages)
depuis juin 2002, disponible gratuitement sur le serveur http://cnum.cnam.fr
LEA D. *Concurrent programming in Java*., 339 pages (Addison Wesley 1997).
[SILBERSCHATZ 01] A. Silberschatz, P. Galvin, G. Gagne, *Principes appliqués des systèmes d'exploitation, avec Java*, Vuibert 2001 (800 pages)
ZAFFALON L., BRÉGUET P. *Programmation concurrente et temps réel avec Ada 95*, 559 p. (Presses Polytechniques et Universitaires Romandes).

DOCUMENTATION SUR INTERNET

Département informatique du CNAM *deptinfo.cnam.fr/Enseignement*
Outils de validation *www.daimi.au.dk/PetriNets/tools*
Outils de validation *quasar.cnam.fr*
DOCUMENTATION ADA
version reliée du **manuel de référence Ada** au prix de 22.95 euros (en 2004)
http://www.springeronline.com/sgw/cda/frontpage/0,11855,5-40280-22-2173562-0,00.html
Adalog *http://perso.wanadoo.fr/adalog*
Groupe Ada-France *ada-France.org*
Ada Europe *ada-europe.org*
ACM SigAda *www.acm.org/sigada*
Ada Home *www.adahome.com* **ou** *lglwww.epfl.ch/ada*
ACT Europe &GNAT *: http://libre.act-europe.fr/*
COMPILATEUR ADA 95 GRATUIT (PC, LINUX,..)
ACT Europe &GNAT :: *http://libre.act-europe.fr/*
Adalog :: *http://perso.wanadoo.fr/adalog/freeada1.htm*
ENSMA :: *http://www.lisi.ensma.fr/ftp/enseignement/ada/*
SITES POUR LA PROGRAMMATION CONCURRENTE
http://stwww.weizmann.ac.il/g-cs/benari/cp.HTM
http://www.seas.gwu.edu/~mfeldman/cs2-book/chap15.html
N'oubliez pas de rendre visite au serveur du département d'informatique
http://deptinfo.cnam.fr/Enseignement/

et au serveur de la bibliothèque numérique du CNAM
http://cnum.cnam.fr

Le serveur pédagogique de SPECIF recense les supports de cours qui sont en ligne et en particulier des cours systèmes qui sont en ligne.
http://rangiroa.essi.fr/specif/spedago/index.html