C/C++, programmer ses applications en Multicore

Cours Pratique de 3 jours - 21h
Réf : MUC - Prix 2024 : 1 870€ HT

Vous découvrirez les architectures Multicore et leur programmation, les techniques de mise en œuvre d'une approche multithread ou multiprocessus et les langages dédiés à la programmation parallèle. Vous étudierez également les contraintes en matière de synchronisation d'accès aux données et les précautions à prendre.

OBJECTIFS PÉDAGOGIQUES

À l'issue de la formation l'apprenant sera en mesure de :

Maîtriser les enjeux de la programmation Multicore

Concevoir et développer des applications à base de threads et de processus

Maîtriser les modèles de programmation parallèle et les librairies disponibles

Déboguer et profiler des applications Multicore

TRAVAUX PRATIQUES

Les travaux pratiques seront réalisés en C/C++ sous Visual Studio en environnement Windows.

LE PROGRAMME

dernière mise à jour : 08/2018

1) Introduction

- Enjeux de la programmation Multicore.
- Tableau des technologies utilisables : processus, thread et parallélisme.
- Description du fonctionnement d'un processeur.
- Architecture en "Hyperthreading".
- Architectures des processeurs INTEL et AMD.
- Architectures NVidia et API.
- Architecture en mémoire partagée vs mémoire distribuée.

2) Modélisation des applications

- Importance des aspects modélisation.
- Patterns de mise en parallèle des traitements.
- Utilisation des mécanismes asynchrones.
- Développer une nouvelle application : précautions et modélisation. Eviter les "singletons".
- Modifier une application existante en Multicore.
- Choix d'architecture : un compromis synchronisation et performance. Choix multiprocessus/multithreads.

3) Threads

- Apport des threads dans une application industrielle.
- Ordonnancement des threads.
- Gestion des stacks et "call stack" dans les threads.
- Débogueurs multithreads.
- Gestion des objets de synchronisation : sections critiques, Mutex et Sémaphores.
- Développer "thread safe".
- API de threads TBB, Clik++, C++11, boost threads, pthreads.

Travaux pratiques : Threads et synchronisation en C/C++.

PARTICIPANTS

Développeurs, architectes logiciels, chefs de projet.

PRÉREQUIS

Bonnes connaissances de C ou de C++. Connaissances de base des concepts liés aux applications Multicore.

COMPÉTENCES DU FORMATEUR

Les experts qui animent la formation sont des spécialistes des matières abordées. Ils ont été validés par nos équipes pédagogiques tant sur le plan des connaissances métiers que sur celui de la pédagogie, et ce pour chaque cours qu'ils enseignent. Ils ont au minimum cinq à dix années d'expérience dans leur domaine et occupent ou ont occupé des postes à responsabilité en entreprise.

MODALITÉS D'ÉVALUATION

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques…
Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

MOYENS PÉDAGOGIQUES ET TECHNIQUES

• Les moyens pédagogiques et les méthodes d'enseignement utilisés sont principalement : aides audiovisuelles, documentation et support de cours, exercices pratiques d'application et corrigés des exercices pour les stages pratiques, études de cas ou présentation de cas réels pour les séminaires de formation.
• À l'issue de chaque stage ou séminaire, ORSYS fournit aux participants un questionnaire d'évaluation du cours qui est ensuite analysé par nos équipes pédagogiques.
• Une feuille d'émargement par demi-journée de présence est fournie en fin de formation ainsi qu'une attestation de fin de formation si le stagiaire a bien assisté à la totalité de la session.

MODALITÉS ET DÉLAIS D'ACCÈS

L'inscription doit être finalisée 24 heures avant le début de la formation.

ACCESSIBILITÉ AUX PERSONNES HANDICAPÉES

Vous avez un besoin spécifique d'accessibilité ? Contactez Mme FOSSE, référente handicap, à l'adresse suivante psh-accueil@orsys.fr pour étudier au mieux votre demande et sa faisabilité.

4) Processus

- Espaces d'adressage des processus, organisation.
- Critères de choix d'une approche multiprocessus.
- Techniques de communication interprocessus (IPC).
- Outils de debugging multiprocessus.
- Avantage et inconvénients des techniques multiprocessus.

Travaux pratiques : Gestion de traitements asynchrones avec l'API C/C++.

5) La programmation parallèle

- Apport et objectifs de la programmation parallèle.
- La librairie "OpenMP" C++ (programmation mémoire partagée).
- La librairie "OpenMPI" (programmation mémoire distribuée).
- Utiliser les GPU des cartes graphiques pour le calcul.
- Kits de NVidia (CUDA) et ATI.
- La librairie "OpenAcc" pour la programmation GPU.
- La librairie "OpenCL" pour la programmation parallèle CPU et GPU.

Travaux pratiques : Paralléliser des algorithmes avec "OpenMP" en C++. Utilisation de l'API OpenCL.

6) Synthèse et conclusion

- Conclusion des techniques étudiées.
- Avenir du C++ avec le multicore.

LES DATES

CLASSE À DISTANCE
2024 : 17 juil., 06 nov.

PARIS
2024 : 22 avr., 10 juil., 23 oct.