Pour information : 4 matières de poids identique dans l'UE 1

UE1 : 16 ECTS au total

Pour les FISA, 3 matières de poids identique, pour un total de 9 ECTS. Le projet de semestre 7 est dans l'UE2

Algorithmique et complexité

Responsable: Johny Bond

Résumé: Complexité : notions de base de complexité. Le modèle de Turing. Classes P et NP. Problèmes NP-complets. Preuves de NP-complétude. Introduction aux algorithmes d'approximation. Calculabilité.

Objectifs

- Donner les bases de la NP-complétude.
- Reconnaître un problème difficile Niveau: Maîtrise
- Connaître la hierarchie des différents types d'algorithmes d'approximation et savoir les utiliser à bon escient Niveau: Maîtrise

Contenu

- Introduction, Machine de Turing
- Machines de Turing
- Les classes P et NP
- Le théorème de COOK
- Problèmes NP-complets connus
- La complexité de certains jeux
- Approximation

References

- C.H. Papadimitriou : "Computational Complexity", Addison-Wesley, 1994
- D. Harel : "Algorithmics : The spirit of computing", Addison Wesley
- M. Sipser : "Introduction to the Theory of Computation", PWS, 1997 (deuxième édition en parue en février 2005
- M.D. Davis, R. Sigal, E.J. Weyuker : "Computability, Complexity + Languages", Academic Press
- M.R. Garey et D.S. Johnson : "Computers and intractability. A guide to the theory of NP-completeness", Freeman, New York, 1979
- Pierre Wolper : Introduction à la Calculabilité, Dunod 2006
- Sanjeev Arora, Boaz Barak : Computational Complexity : A Modern Approach, Cambridge University Press, 2009
- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest : Introduction to algorithms, MIT Press, 1990

Conception logicielle

Responsable: Philippe Collet

Résumé: Présentation et utilisation des schémas de conception

Objectifs

- Réaliser des conceptions robustes et facilement extensibles.
- Savoir choisir, appliquer et composer les schémas de conception Niveau: Maîtrise
- Savoir décider de l'opportunité de la mise en oeuvre d'un schéma de conception Niveau: Maîtrise

Contenu

- Les schémas de conception: de structure, de comportement, de création
- Etude détaillées de quelques schémas de conceptions, reposant sur des exemples pour lesquels sont présentées différentes solutions: des plus naïves, aux plus élaborées qui mettent en oeuvre les schémas de conception améliorant extensibilité, réutilisabilité et documentation. Pour chaque schéma, mise en évidence des conséquences de sa mise en oeuvre.

References

- Design Patterns: Elements of Reusable Object-oriented Software. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Addison Wesley, 1995
- UML et les Design Patterns. Craig Larman. CampusPress , 2003

Programmation multi-paradigmes

Responsable: Julien Deantoni

Résumé: Apprentissage des bases du langage C++

Objectifs

- Acquérir les bases de C++ permettant de développer des programmes procéduraux utilisant la STL et des programmes orientés objets.
- Paradigmes fondamentaux de programmation (OO, impératif, généricité) Niveau: Maîtrise

Contenu

- Introduction à C++ et la notion de classe (template)
- C++ procédural et STL
- Définition des classes C++ comme types abstraits de données
- Héritage et programmation à objets en C++
- Récapitulation des mécanismes fondamentaux de C++

References

- Bjarne Stroustrup : Programming (Principles and Practice using C++)
- Stanley Lippman, Barbara Moo : C++ Primer

Projet Semestre 7

Responsable: Philippe Collet

Résumé: Ce projet de 3 semaines à temps plein permet de mettre en applications les enseignements formels et appliqués des autres cours sur un problème original.

Objectifs

- Mobiliser des connaissances acquises dans plusieurs enseignements pour aider les élèves à faire le lien
- Gérer son temps et les taches de dévelopement Niveau: Expert
- Travailler en accord avec un client Niveau: Maîtrise
- Livrer un projet logiciel Niveau: Expert

Contenu

- 3 semaines de travail à temps plein sur le projet

Choisir 3 options de poids équivalent parmi les 6 de l'UE 2, dans chaque paire, un cours, il y a 3 paires :

**Machine Learning/Sécurité Logicielle*
**Langages du web/Compilation*
**Finite State Machines/Gestion de la concurrence*

UE2 : 9 ECTS au total

Pour les FISA, UE2 contient le projet S7 (coeff 4) obligatoire, et une seule option doit être choisie parmi la liste (coeff 3) : ce sera soit Sécurité Log. soit Machine Learning.

Compilation

Numerus clausus : 1 groupe de TD

Responsable: Erick Gallesio

Objectifs

- Assimiler les théories, méthodes et techniques qui permettent de décrire et concevoir des langages. comprendre les principes de conception des lsangages de programmation et les transformations de programmes réalisées par les compilateurs
- Comprendre finement les différents aspects des langages de programmation, en particulier les vérifications et transformations effectuées par les compilateurs.
- Assimiler les techniques qui permettent l'écriture de compilateurs.
- Mettre en pratique les principes et méthodes étudiés. Il s'agit de réaliser, dans le cadre du projet de compilation, un compilateur pour un petit langage de programmation.

Contenu

- Introduction à la compilation
- Analyse lexicale et LEX
- Rappels sur les langages formels (grammaire algèbriques, arbres de dérivations, ambiguité, élimination de la récursivité gauche)
- Analyse syntaxique non déterministe (analyse par descente récursive, analyse prédictive non récursive, retour arrière (backtracking))
- Analyse Syntaxique descendante
- Analyse syntaxique ascendante
- Traduction dirigée par la syntaxe
- Générateurs LR()
- Sémantique statique des langages de programmation
- Traitements "source to source"

References

- A first course using ANSI C, LEX and YACC, J. P. Bennett,2nd edition, McGraw Hill (1996)
- Compilateurs, D.Grune et all , Dunod (2002)

- Compilateurs: Principes, techniques et outils,A. Aho, R. Sethi, J. Ullman; InterEditions (1991)
- Crafting a Compiler, C.Fischer, R.LeBlanc , Benjamin Cummings Series (1988)
- Lex and Yacc,JR Lewine, T. Mason, D. Brown; O'Reilly and Associates, Inc (1992)
- Modern Compilation Implementation, A. Appel; Cambridge University Press (1998
- The Essence of Compilers, R. Hunter , Prentice hall (1999

Acquis

- Savoir réaliser un compilateur pour un petit langage de programmation Niveau: Maîtrise
- Utilisation de générateurs de type lex et yacc Niveau: Maîtrise
- Utilisation de techniques de compilation pour des transformations "source to source" Niveau: Maîtrise

Finite State Machines

Numerus clausus : 2 groupes de TD

Responsable: Julien Deantoni

Résumé: Ce cours permet de souligner une utilisation pragmatique des machines à états finis dans différents pans de l'informatique. Le cours montrera l'utilisation de machines à états pour la définition de protocole de communication aussi bien que pour la définition du contrôle d'un système embarqué. Différentes classes de machines à états seront introduites conjointement avec leur outil. L'objectif principal est de comprendre le rôle central que prennent ces représentations dans l'informatique de tous les jours. Les mises en œuvres varieront de la génération de code C et Java à l'utilisation de langages formels comme les automates temporisés.

Objectifs

- Permettre aux étudiants de maîtriser la notion de machine à états finis et l'utilisation de variantes spécifiques selon l'utilisation qu'ils désirent en faire
- La compréhension de l'utilisation des automates et autres langages formels de manière pragmatique dans l'informatique

Contenu

- Introduction des machines à états finis et rappels sur les automates
- Utilisation dans différents pans de l'informatique.
- Définition de protocole de communication, définition du contrôle d'un système embarqué, etc
- Présentation de différentes classes de machines à états et mise en oeuvre dans des outils
- Génération de code C et Java
- Utilisation de langages formels comme les automates temporisés.

Gestion de la concurrence

Numerus clausus : 2 groupes de TD

Responsable: Michel Riveill

Résumé: Connaitre les principaux problèmes posés par le partage de ressources et les solutions pouvant être apportées que ce soit dans une architecture avec mémoire commune ou une architecture répartie sans

mémoire commune. Les problèmes concernent la gestion de la famine et des interblocages. Les modèles étudiés concernent les rendez-vous, la barrière, lecteur-redacteur, producteur-consommateur en architecture avec ou sans mémoire commune.

Objectifs

- Construire des programmes multi-thread ou multi-processus correct.
- Connaitre les principes des outils de preuve (model-checking) et savoir les utiliser dans des cas simples.
- Connaitre les outils usuels du domaine (verrou, sémaphore, moniteur) et savoir les utiliser dans différents langages (java), OS (Unis) ou bibliothèque (pthread)
- Mettre en place une méthodologie de preuve par model-checking: modélisation du problème, écriture de la propriété à vérifier, vérification et si c'est OK, traduction du modèle dans un langage exécutable Niveau: Notions
- Programmer les schémas usuels de synchronisation avec des sémaphores ou des moniteurs, en C ou en Java: Section critique, Rendez-vous, Lecteur-rédacteur, Tampon Niveau: Maîtrise
- Mettre en oeuvre les schémas étudiés dans un cas réel (projet) Niveau: Applications

Contenu

- Introduction à la programmation parallèle et concurrente
- Modélisation FSP : processus séquentiel (FSP + Java)
- Modélisation FSP : composition de processus (FSP + Java)
- Verrou, Moniteurs et Sémaphores (FSP + Java)
- Propriétés de sûreté et vivacité (FSP + Java)
- Interblocage et Transaction : famille de solutions (FSP + Java)
- Communication et synchronisation entre thread Posix
- Communication et synchronisation entre processus Unix
- Synchronisation non bloquante
- Communication et synchronisation dans un système distribués : approche client-serveur et approche P2P
- Temps, ordre et état dans les systèmes répartis
- Gestion "cohérente" de la mémoire

References

- Michel Raynal, Concurrent Programming - Algorithms, Principles, And Foundations, Springer - 2013
- Assez proche du cours : Magee + Jeff Kramer, Concurrency: State Models + Java Programs, Weiley, 2006
- Synchronisation et état global dans les systèmes répartis. (Tome 2 - une introduction aux principes des systèmes répartis). Eyrolles, collection EDF, 1992, 228 p.

Langages du web: schémas et transformations XML, JSON et Frameworks Web

Numerus clausus : 3 groupes de TD

Responsable: Catherine Faron-Zucker, Rémi Pourtier

Résumé: Ce cours présente les langages standards clés des technologies XML: le langage XPath d'expressions de localisation, le langage XSD de définition de modèles de documents XML et le langage XSLT de transformation de documents XML. Il est illustré sur plusieurs dialectes XML. La seconde partie du cours sera dédiée aux frameworks/libraries front end orientés composants. Un premier travail sera effectué autour de ReactJS pour vous permettre d'apprendre cette technologie et ensuite une comparaison poussée sera conduite entre ReactJS et Angular : Quelles sont les différences et points communs ? La réponse à cette question se fera à travers l'étude approfondie des concepts, des architectures et des choix d'implémentation. Ce travail d'apprentissage rapide d'une technologie et de comparaison poussée avec l'existant est important et très recherché par les entreprises.

Objectifs

- Maîtrise des langages XML, XPath, XSD, XSLT et des concepts de documents et données structurés, de modèle de documents et de transformation de documents.
- Modélisation XML - Schémas XML Niveau: Maîtrise
- Transformation de données XML avec XSLT Niveau: Maîtrise
- Maitrise des concepts de ReactJS
- Comparaison approfondie entre deux technologies : ReactJS et Angular : concepts, architectures et choix d'implémentation
- Comprendre les besoins derrière les frameworks/libraries front end
- Développer la capacité d'apprentissage rapide d'une nouvelle technologie et pouvoir en extraire les principaux concepts

Contenu

- XML et espaces de nomage
- XPath : Expression de chemins de localisation
- XSD : modèles de documents XML
- XSLT : transformation de documents XML
- Implémentation d'une application avec ReactJS
- Étude comparative entre ReactJS et Angular
- Présentation des résultats
- Sensibilisation aux performances des applications web et l'impact des frameworks/libraries

References

Machine Learning

Numerus clausus : 3 groupes de TD, incluant les étudiants FISA

Responsable: Diane Lingrand

Résumé: Ce cours s'articule autour d'un projet fil rouge : le développement d'une application en C++ de reconnaissance de la langue des signes en temps-réel à partir d'une webcam. Le cours présente les principes de l'apprentissage artificiel et de vision par ordinateur, détaille les mécanismes des algorithmes classiques d'apprentissage statistique qui permettent ensuite aux étudiants de développer leur application en assemblant des briques logicielles présentes dans la librairie standard OpenCV.

Prerequis:

- Calcul du gradient pour minimiser/maximiser un critère
- Programmation C++

Objectifs

- Apprendre des algorithmes très classiques de l'apprentissage automatique (Boosting, réseaux de neurones, forêts aléatoires).
- Aborder les notions de bases de la vision par ordinateur favorisant l'ouverture à des projets motivants autour de la robotique, du multimédia
- Principes de bases de l'apprentissage automatique Niveau: Maîtrise
- Bases en analyse d'images et de vidéos Niveau: Applications
- Analyse statistique de données (histogrammes, distributions, estimation) Niveau: Notions
- Principes de la vision artificielle par ordinateur, introduction à la vision en robotique Niveau:

Contenu

- Principes du boosting
- Algorithme AdaBoost en détail
- Histogrammes
- Algorithmes Mean-Shift et CAMShift
- Réseaux de neurones
- Détection de contours
- Extension de la détection du visage à la détection de piétons
- Détection de mouvement
- Reconnaissance de gestes
- Extension à la reconnaissance de gestes en 3D
- Apprentissage par forêts aléatoires

References

- Computer Vision in C++ with the OpenCV Library By Gary Bradski, Adrian Kaehler, O'Reilly Media, October 2013, 575
- Mastering OpenCV with Practical Computer Vision Projects, by Daniel Lélis Baggio, Shervin Emami, David Millán Escrivá, Khvedchenia Ievgen..., Packt Publishing, December , 2012
- OpenCV 2 Computer Vision Application Programming Cookbook, by Robert Laganière, May 2011

Sécurité logicielle

Cours donné in English if needed

Numerus clausus : 2 groupes de TD, incluant les étudiants FISA

Responsable: Yves Roudier

Résumé

Les logiciels constituent aujourd'hui le pilier de l'économie numérique et des réseaux. Ce cours introductif vise à étudier les risques posés par leur code et leur architecture, notamment quand elle est distribuée, aussi bien que les moyens de les sécuriser. Il se situe à la frontière entre sécurité informatique et génie logiciel. Un panorama des différentes techniques disponibles tout au long du cycle de vie logiciel (ingénierie des exigences et des modèles, codage sécurisés, support fourni par langages et environnements d'exécution sécurisés, test de programmes et certification, déploiement sécurisé) sera notamment dressé dans ce cours.

Objectifs

- Définition d'exigences de sécurité ----- Niveau: Applications
- Programmation sécurisée ---- Niveau: Maîtrise
- Authentification, Contrôle d'accès ---- Niveau: Applications
- Utilisation de bibliothèques cryptographiques ---- Niveau: Applications
- Test de sécurité ---- Niveau: Applications

Contenu

- Programmation sécurisée : vulnérabilités fréquentes (mémoire,processus et entrées/sorties) et bonnes pratiques
- Etude des bibliothèques de sécurité et cryptographiques de quelques langages
- Test de sécurité pour programmes et piles de protocoles (fuzzing, analyses statique et dynamique)
- Environnement d'exécution sécurisés : OS (architecture de sécurité et contrôle d'accès logique), virtualisation (JVM, machines virtuelles, micronoyaux), trusted computing (TPMs, cartes à puce)
- Certification et déploiement sécurisé de logiciels (mécanismes de signature de code)
- Ingénierie des exigences liées à la sécurité et à la vie privée et ingénierie des modèles pour la définition d'architectures de sécurité logicielles

References

- Gildas Avoine, Pascal Junod, Philippe Oechslin, Sylvain Pasini. Sécurité informatique, Cours et exercices corrigés. Vuibert.
- Ross Anderson. Security Engineering. Wiley. (aussi disponible en ligne : http://www.cl.cam.ac.uk/~rja14/book.html)

Ce cours est commun Master 1 EIT Digital Financial technologies

Pour information : une des 2 matières (à coeff 2) dans l'UE 3 « SHS »

UE1 : 3 ECTS au total

Gestion d'entreprise

Responsable: Christine Bachelot

Résumé: Le module doit permettre de comprendre les indicateurs clés d'une bonne gestion financière de l'entreprise, d'analyser et de diagnostiquer la santé financière d'entreprise par le biais de cas d'entreprises.

Objectifs

- Déterminer le seuil de rentabilité et le point mort lors de la gestion d'un projet
- Appréhender la situation financière : CA, bénéfice, trésorerie, EBE Niveau: Applications
- Calculer un coût de revient simple et un seuil de rentabilité Niveau: Applications
- Pratiquer les intérêts simples et composés Niveau: Applications
- Comprendre la capitalisation et l'actualisation Niveau: Applications
- Établir un tableau de prêt bancaire (calcul d'annuités) Niveau: Applications

Contenu

- Analyse du bilan fonctionnel
- Analyse du compte de résultat : les Soldes Intermédiaires de Gestion.
- Les moyens de financement de l'entreprise
- Les coûts

References

- Comptabilité analytique de GOUJET, RAULET éditions DUNOD
- Comptabilité générale de MAESO, PHILIPPS, RAULET éditions DUNOD
- Contrôle de gestion de G.LANGLOIS, M.BRINGER éditions FOUCHER
- Introduction à l'analyse financière de A.PLANCHON éditions DUNOD