Syllabus du M1 Sciences Du Logiciel

UI/UX et applications frontales

S7, 6 ECTS

Objectifs :

La partie du module dédiée à l'UI/UX a pour objectif de permettre aux étudiants de maîtriser les principes de conception, programmation et évaluation d'interfaces utilisateur et techniques d'interaction avec pour objectif la production de systèmes informatiques utilisables.
La partie du module dédiée aux applications frontales vise à étudier les langages côté client (*front end*) et à savoir mettre en œuvre un framework dans le cadre d'une application web dynamique de type "Single Page Application".
Compétences :

- Proposer et mettre en œuvre une démarche de conception centrée utilisateur et un ensemble de techniques associées pour le développement d'applications informatiques.
- Concevoir et implémenter la partie représentation d'une application web réactive en m'appuyant sur un framework dédié.

Contenu :

Partie "UI/UX" :

1. Conception centrée utilisateur (UCD) selon le standard ISO 9241-210, règles et standards Introduction à l'analyse des utilisateurs (personas, modélisation des tâches)
2. Etude et mise en oeuvre de techniques de conception: prototypage basse, moyenne et haute-fidélité
3. Etude et mise en oeuvre de techniques d'évaluation analytiques (cheminement centré tâches, évaluation heuristique, prédictions à base de modèles) et empiriques (tests d'utilisabilité)
4. Retour sur investissement de l'UCD
5. UX et ses différentes dimensions

Partie "Front" :

1. Rudiments d'un langage côté client (Javascript ou typescript) : syntaxe du langage, utilisation dans le navigateur, les principales structures de données, etc
2. Spécificités d'un langage côté client (Javascript ou typescript) : le modèle objet du langage, la puissance des fonctions, …
3. Etude d'un framework de type front end (Angular, React ou Vue.js)
4. Manipulation du DOM et notion de DOM virtuel
5. Gestion des événements et des formulaires
6. Routage et application "Single Page Application"

Techno utilisées en TP : outils de prototypage et de modélisation des tâches, Javascript ou typescript et framework de type front end, node.js

Pré-requis :

- programmation orientée objet et programmation fonctionnelle, modélisation UML, éléments du langage HTML et CSS

Références bibliographiques :

- Standard ECMA-262 : https://www.ecma-international.org/publications/standards/Ecma-262.htm
- JavaScript and JQuery: Interactive Front-End Web Development. Jon Duckett
- ISO 9241-210 – Part 210
- Nielsen, J. Usability Engineering. 1993. Morgan Kaufmann Publishers
- Dix, A. et al. 2003. Human-Computer Interaction. Prentice-Hall

Mots-clés :
- Programmation front end, framework, DOM, single page application.
- UI, Conception centrée utilisateur, prototypage, modèles de tâches, UX

Modalités d'évaluation :
- TP noté
- Contrôle terminal (devoir surveillé)

Modélisation, Conception, Développement Collaboratif

S7, 6 ECTS

Objectifs :

Ce module vise la modélisation et la conception des logiciels complexes, avec une sensibilisation aux spécificités du développement collaboratif.

Les aspects traités dans ce cours sont la modélisation, son utilisation dans le cadre d'une démarche de développement orientée objet, la spécification de contraintes afin de rendre les modèles cohérents, les patrons de conception orientés objet et leur utilisation dans le cadre d'un problème de conception, et la mise en œuvre d'un logiciel conformément aux exigences exprimées et aux bonnes pratiques relatives au développement collaboratif.

Compétences :

- Je conçois et je mets en œuvre la modélisation d'un système à partir d'une spécification des exigences, en appliquant un processus de développement préétabli et en utilisant le langage UML
- Je conçois et je mets en œuvre ma solution au moyen du ou des patrons de conception adéquats, sélectionnés et adaptés sur la base de leur description, selon une méthodologie adaptée au développement orientée objet.
- Je contribue au développement collaboratif d'un logiciel conformément au workflow retenu par l'équipe de développement tout en maintenant son référentiel d'exigences.

Contenu :

Partie I. Modélisation et Conception

- Gestion de la complexité des logiciels, techniques de gestion de la complexité (décomposition vs. abstraction)
- Modélisation avec UML
    - modélisation des exigences
    - modélisation structurelle
    - introduction à la spécification de contraintes avec OCL
    - modélisation du comportement
- Processus / Démarche de développement
    - Modélisation du contexte métier
    - Modélisation des besoins
    - Conception et réalisation des besoins

Partie II. Patrons de conception

- Conception à base de patrons
    - qu'est-ce qu'un patron de conception, catégories de patrons
    - description et classification des patrons de conception
    - utilisation des patrons dans la conception
    - principaux patrons du "Gang of Four" : Stratégie, Médiateur, Observateur, Décorateur, Fabrique abstraite, Fabrication…

- composition de patrons

Partie III. Développement collaboratif

- Introduction au développement collaboratif (droit d'auteur, licences libres…)
- Introduction à la gestion des exigences et des versions d'un logiciel
- Gestion de version décentralisée et modèles de développement associés

Pré-requis :

- Conception et programmation orientée objet (Java)
- Notions d'UML (diagrammes de classes et de séquence)
- Notions de processus de développement

Références bibliographiques :

- Steve Tockey. How to Engineer Software: A Model-Based Approach, John Wiley & Sons, 2019
- J Warmer, A Kleppe. The Object Constraint Language, Addison-Wesley, 2003
- E. Freeman & E. Freeman. Head First Design Patterns, O'Reilly, 2005
- Pascal Roques, UML2 - Modéliser une application Web, 4e édition, Eyrolles
- Scott Chacon & Ben Straub. Pro Git, Apress, 2e édition, 2014

Mots-clés :

Conception, Modélisation, Objet, UML/OCL, Design Patterns, Développement Collaboratif

Modalités d'évaluation :

- CC: Evaluation d'un mini-projet de conception mené pendant les séances de TP
- CT: Epreuves écrites

Théorie des langages

S7, 6 ECTS

Objectifs :

Cette UE a deux objectifs :

- comprendre le fonctionnement d'un analyseur de source et l'obtention de code intermédiaire (front-end) et,
- exploiter cette représentation pour générer et optimiser des codes exécutables (back-end) tout en mettant en oeuvre des stratégies de vérification afin d'assurer la correction du compilateur.

Compétences :

- mettre en oeuvre un analyseur de code source,
- développer des traducteurs vérifiés appliqués à la représentation intermédiaire du programme,
- savoir optimiser les performances d'un programme traduisant des requêtes relationnelles

Contenu :

- Compilation de langage impératif
    - Exécution et compilation
    - Analyse syntaxique
    - Génération de code
- Vérification
    - Preuve assistée
    - Modélisation d'AST typés
    - Sémantique de langages
    - Transformations vérifiées
- Compilation d'une requête en langage déclaratif
    - Introduction et motivations
    - Optimisation de code
    - Génération de code

Les TPs consistent en la mise en œuvre d'un mini-projet permettant de réaliser un compilateur composé d'un analyseur pour un langage de programmation réel exécuté par une machine virtuelle.

La première partie consiste à réaliser un mini-compilateur de l'analyse du source à la génération du code: analyse lexicale, analyse syntaxique, construction des arbres de syntaxe abstrait et génération de code. En seconde partie, des optimisations vérifiées seront réalisées sur la représentation intermédiaire . L'environnement de preuve interactive utilisé pour cela (Coq) est d'abord présenté avec un rappel des notions sous-jacentes, puis un ensemble d'exemples de sémantiques de langages de programmation est formalisé, avant de modéliser et vérifier une transformation agissant sur la représentation intermédiaire.

Pré-requis :

Théorie des langages, preuve interactive, connaissances en bases de données relationnelles

Références bibliographiques :

- A. Aho et al. Compilateurs : principes, techniques et outils. Pearson Education.
- Y. Bertot. Coq in a Hurry. EJCP, 2016.
- M. Bouzeghoub et al. Systèmes de BD : des techniques d'implantation à la conception de schémas. Eyrolles

Mots-clés :
Compilation, optimisation, génération de code, analyse syntaxique, représentation intermédiaire, sémantiques, preuve assistée, transformation vérifiée

Algorithmique avancée

S7, 6 ECTS

Objectifs :

Après des rappels de complexité des algorithmes, illustrés sur des opérations sur des structures de données avancées, ce module présente les principales méthodes et algorithmes pour modéliser et résoudre les problèmes de décision et d'optimisation linéaire et / ou combinatoire. Après validation de ce module, les étudiant-e-s sauront :

- reconnaître un problème d'optimisation combinatoire difficile
- modéliser des problèmes en flots, en programmation linéaire (avec ou sans variable entière), en programmation par contraintes
- choisir et utiliser efficacement un outil de résolution approprié
- implémenter les opérations sur les arbres de recherche et les tas, et des algorithmes de résolution exacte ou incomplète pour un problème d'optimisation combinatoire, et analyser leur complexité

Contenu :

- Structures de données avancées (tas, arbres de recherche)
- Flots : réseaux de transport sans, calcul de flots maximum
- Programmation linéaire : formalisme, résolution graphique, problème dual
- Problèmes classiques d'optimisation combinatoire difficile : partitionnement, ordonnancement, routage, ...
- Algorithmes approchés, coefficient d'approximation
- Le problème SAT, méthode DPLL
- Classes de complexité : La classe NP, réductions, NP-complétude et classes d'approximation
- Formalismes génériques et recherches exhaustives : PPC, backtrack, heuristiques, forward checking ; PLNE, relaxation ; modélisation
- Méthodes incomplètes : principe des recherches locales / sur populations ; méta-heuristiques (liste tabou, ...)

Pour chaque type de problème d'optimisation ou de décision abordé, on étudie la complexité des algorithmes et la modélisation. Les TP consisteront en l'implémentation d'un solveur de type "backtrack", en l'implémentation d'algorithmes de recherche locale, et en l'utilisation de solveurs de PPC / PLNE.

Pré-requis : Graphes, Structures de Données, Complexité

Références bibliographiques :

- Algorithmique. T. Cormen, C. Leiserson, R. and C. Stein. 3ème éd., 2010
- Algorithms. S. Dasgupta, C.H. Papadimitriou, U.V. Vazirani. McGraw-Hill, 2006.

Mots-clés :

Flots, Programmation Linéaire, optimisation combinatoire, SAT, CSP, classes de complexité, Méta-Heuristiques, Tas binomiaux, B-arbres.

Parallélisme

S7, 6 ECTS

Objectifs :

L'objectif de ce module d'enseignement est d'introduire les fondements du parallélisme sur des architectures distribuées ou massivements parallèles (GPU) :

- les modèles du parallélisme (synchrone/asynchrone)
- l'abstraction d'algorithmes parallèles (Réseaux de Petri)
- les concepts de coopération et de synchronisation
- les modèles permettant d'atteindre un parallélisme efficace (MPI, Cuda)

A l'issue de cet enseignement, l'étudiant sera capable de concevoir, analyser et évaluer des algorithmes parallèles. L'étude des approches et API MPI et Cuda permettra à l'étudiant de résoudre des problèmes à l'aide d'algorithmes parallèles efficaces. La présentation des "conditions" complètera la formation de l'étudiant quant à la synchronisation d'activités parallèles.

Contenu :

La pédagogie se basera sur plusieurs mises en situation (TD et TP) pour intégrer les particularités du parallélisme.

- Parallélisme/synchronisation
    - moniteurs de Hoare, conditions
    - réseaux de Petri
- Modèle MPI
    - modèle et primitives
    - parallélisation et implémentation d'algorithmes
- Modèle Cuda
    - modèle de programmation et API
    - modèle d'exécution, mémoires et optimisation
    - applications, bibliothèques et outils
- Master Class

Pré-requis :

Algorithmique avancée, programmation concurrente, processus, threads, variables partagées, architecture des ordinateurs, réseau, abstraction de problèmes

Références bibliographiques :

- Fundamentals of Parallel Multicore Architecture, Chapman and Hall/CRC, Y. Solihin
- Principles of Concurrent and Distributed Programming, Addison-Wesley.
- Communication and Concurrency, Prentice Hall Int. Series in Computer Science, R. Milner.

Mots-clés :

Architectures parallèles, Modèles parallèles, Modèles répartis, cohérence de données, expressions et conditions de synchronisation, MPI, CUDA

Ingénierie des Systèmes Interactifs et des Applications Web Dynamiques

S8, 6 ECTS

Objectifs :

La partie “ingénierie des applications web dynamiques” se positionne côté "back office" et est dédiée à des aspects de conception (archi. multi-couches, rôle d'un "framework") et technologiques (J2EE : servlet, JSP, ORM, etc).

Compétences :

- Mettre en œuvre le patron MVC dans une architecture multi-couches
- Faire cohabiter une représentation relationnel d’un SGBD avec une représentation objet

La partie “Ingénierie des Systèmes Interactifs” (ISI) vise la maîtrise de la modélisation et de la programmation des systèmes interactifs (SI).

Compétences:

- Architecturer et modéliser les SI pour garantir utilisabilité, modifiabilité et fiabilité
- Exploiter le design pattern MVC
- Valider la fiabilité d’un SI (test et vérification)
- Mettre en œuvre dans un environnement de programmation par événement

Contenu :

Partie “Applications web dynamiques” :

1. Eléments d'architecture répartie
   - client-serveur vs. n-tier
   - MVC1 et MVC2
2. Le rôle d'un framework
   - injection de dépendances
   - inversion de contrôle
3. La plateforme J2EE
   - Cycle de vie d'une servlet et d'une JSP
   - Syntaxe JSP
   - Portée et communication par objets implicites
   - Cohabitation d'un modèle objet et relationnel d'un SGBD : JDBC, JPA

Techno utilisées en TP : Spring Boot; Hibernate; Thymeleaf

Partie “Systèmes interactifs” :

1. Principes architecturaux des systèmes interactifs (modèles de Seeheim et ARCH)
2. Design pattern MVC, de sa mise en oeuvre en Java et de son intégration dans ARCH
3. Principes de modélisation des systèmes interactifs à base d’automates à états finis étendus.
4. Implémentation à base de modèle dans un environnement de programmation par événement
5. Description de propriétés de systèmes interactifs et vérification sur modèles. Comment gérer utilisabilité, modifiabilité et fiabilité dans un même cadre méthodologique
6. Mise en œuvre des principes de validation : expression et vérification de propriétés, définition et mise en œuvre de tests sur des systèmes interactifs

Techno utilisées en TP : NetBeans, Java Swing

Pré-requis :

Connaissances requises : programmation orientée objet (notamment Java), modélisation UML, éléments du langage HTML, modèle relationnel pour les données, bibliothèque Java SWING et programmation par événement, modélisation par automates

Références bibliographiques :

- Conception d'applications en Java/JEE. Jacques Lonchamp. Dunod.
- J2EE : https://jakarta.ee/
- Buxton, W., 1990. A three-state model of graphical input. IFIP TC13 Conference on HCI, North-Holland Publishing Co., 449-456
- L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, (3rd edition), Addison-Wesley, 2012.

Mots-clés :

- Application répartie, persistance, J2EE
- Modélisation de systèmes interactifs, Architecture logicielle, fiabilité, modifiabilité utilisabilité, vérification, test

Modalités d'évaluation :

- TPs notés
- Contrôle terminal (devoir surveillé)

Modèles et Architectures des Applications Réparties (MARA)

S8, 6 ECTS

Objectifs :
Ce cours présente les méthodes et outils permettant de concevoir et de gérer les applications informatiques distribuées et hétérogènes.
La première partie du cours est consacrée à la représentation de données structurées et l'interopérabilité, avec un focus particulier sur les langages XML (modélisation via différents types de grammaires, langages de transformation et d'interrogation), et sur l'approche orientée services (aspects architecturaux, conception, développement et publication).
Dans la deuxième partie, on poursuivra l'étude des Design Patterns du GoF initiée en S7 au tronc commun. Cette partie présentera aussi des patterns de programmation concurrente et répartie. En outre, on étudiera les bases de la programmation concurrente en Java et de la programmation répartie avec Java RMI.

Contenu :
Partie I : Interopérabilité des données et approches orientées services

1. Gestion de build et des dépendances dans un projet Java
    - Apache Maven
2. Représentations de données structurées
    - Modèles de données XML : DTD et schémas XML
    - Transformations et extractions d'informations : XSLT, XPath
    - Parsing de documents XML : DOM, SAX, StAX, JAXB
    - Comparaison avec JSON (JSON Schema, JSONP, JSONB)
3. Approches orientée services
    - Services Web SOAP : WSDL, SOAP, UDDI
    - Méthodologies de conception contract-first et code-first
    - Web Services REST

Partie II : Design patterns (compléments), Programmation concurrente et répartie en Java

1. Compléments sur les design patterns du GoF

2. Programmation concurrente en Java et multithreading
    - Multithreading en Java : classe Thread, interface Runnable, synchronisation de threads, attentes et notifications
    - Design patterns de programmation concurrente (Objet actif)

3. Programmation répartie en Java
    - Introduction au middleware, principes des middleware à objets
    - Design patterns de programmation répartie
    - Java RMI
    - Conception et programmation à base d'objets répartis en Java RMI

Pré-requis :
Programmation orientée objet (notamment Java), connaissances élémentaires en bases de données et en programmation répartie (modèle client-serveur)

Références bibliographiques :

- N. Bradley : The XML companion, Addison-Wesley, 3rd edition, 2001
- R. Fielding : Architectural Styles and the Design of Network-based Software Architectures, PhD thesis, 2000
- E. Freeman, E. Freeman, K. Sierra, B. Bates : Head First Design Pattern, O'REilly, 2006
- C. Delannoy : Programmer en Java, 9e édition, Eyrolles, 2014

Mots-clés :

Interopérabilité, données structurées, services web, design patterns, concurrence, répartition, middleware, Java

Modalités d'évaluation :

- Contrôle sur table
- Rapport
- Code

Gestion de Projets de Recherche, Industriels et Agiles

S8, 9 ECTS

Objectifs :

L'objectif de l'UE est d'initier les étudiants au monde de la recherche et de les former à la production d'un état de l'art, à celle de la gestion de projet industriel ou de recherche en focalisant sur les aspects méthodologiques selon le PMBoK(1) ou les valeurs de l'agilité(2).

Compétences :

- Faire une recherche bibliographique sur un sujet de recherche ou industriel.
- Planifier, coordonner, mesurer, surveiller, contrôler et analyser les activités d'un processus outillé de développement logiciel selon le PMBoK
- Travailler en équipe.
- Appliquer la méthode SCRUM pour la conduite d'un projet tout en respectant les valeurs de l'agilité.
- Présenter la réalisation d'un livrable sous forme écrite ou orale.

Contenu :

Le contenu est divisé en trois parties :

1 - Initiation à la recherche et la production d'état de l'art :

- Présentation des sociétés savantes, de l'organisation et de l'évaluation de la recherche
- Présentation synthétique d'un sujet de recherche
- Principes de la rédaction d'articles de recherche (état de l'art, intégration de citations et références)

2 - Gestion de projet selon le PMBoK

- Introduction au management de projet informatique
- Cycles de vie du projet
- Les groupes de processus et les domaines de connaissances du management de projet
- Le management du contenu du projet, des ressources, des parties prenantes, de la communication, des délais, des risques et de la qualité

3 - Méthodes agiles et pratique SCRUM

- Les problèmes liés aux processus de gestion de projets de type linéaire
- Le Manifeste Agile : les valeurs et les principes.
- Scrum
    - L'équipe Scrum
    - Les cérémonies Scrum
    - Les artefacts de Scrum

La production de l'état de l'art est mise en pratique à travers un projet en groupe de 4 pour un volume de 50h/étudiant.

La gestion de projet est analysée en TD d'après l'expérience pratique des étudiants selon les domaines de connaissances, les groupes de processus et la pratique de la méthode Scrum.

Pré-requis :

- Expérience de développement individuelle ou collective

Références bibliographiques :

(1) Project Management Book of Knowledge - Project Management Institute - https://www.pmi.org/pmbok-guide-standards

(2) Manifeste Agile - https://agilemanifesto.org

(3) Le métier de chercheur. Regard d'un anthropologue. Bruno Latour. INRA Editions, 2001

(4) SCRUM - Scrum Guide - https://www.scrum.org

Mots-clés :
Recherche scientifique, état de l'art, rédaction d'articles, gestion de projet, PMBoK, cycles de vie, processus, domaines de connaissances, agilité, SCRUM.

Modalités d'évaluation :
Evaluation de livrables :

- Liste de références, résumés d'articles, article de synthèse, poster
- Plan de management de projet
- Contrôle de connaissances des définitions du PMBoK et de SCRUM
- Artefacts SCRUM
- Présentations orales de livrables

Professionnalisation en Sciences Du Logiciel

S8, 6 ECTS

Objectifs :

Le stage ou l'alternance ont pour but de mettre les étudiants en situation réelle, en entreprise ou en laboratoire, pour connaître et approfondir le monde professionnel orienté industrie ou recherche; de mettre en application les connaissances acquises au cours de leur cursus; de les préparer au stage long de Master2 et de favoriser la future insertion professionnelle en apportant une expérience aux étudiants.

Compétences :

- Définir son projet professionnel et ses compétences pour rechercher son insertion idéal
- Construire un CV personnalisé et rédiger une lettre de motivation spontanée et en réponse à une annonce
- Adopter la pratique professionnelle définie dans l'équipe d'accueil en entreprise ou en laboratoire
- Présenter à l'écrit et à l'oral ses activités en situation professionnelle

Contenu :

Cours magistral :

- Définition d'une compétence
- Définition d'un projet professionnel et en particulier des compétences pour rechercher un stage
- Construire un CV personnalisé et rédiger des lettres de motivation spontanée et en réponse à une annonce

Travaux dirigés :

- Identification des compétences individuelles
- Rédaction itérative du CV (3 versions)
- Rédaction itérative de lettres de motivation (3 versions)

Stage ou alternance (12 semaines ETP minimum) :

- Recherche d'un stage ou d'une alternance
- Réalisation des activités en situation professionnelle
- Rédaction d'un rapport de fin d'activité
- Présentation orale des activités en situation professionnelle

Pré-requis :

Expérience de développement logiciel académique

Mots-clés :

Stage, alternance, CV, lettre de motivation, rapport, soutenance

Modalités d'évaluation :

Évaluation des productions suivantes : CV, lettre de motivation, travail en situation professionnelle, rapport de stage, soutenance de fin d'activité, respect des échéances.

Anglais

S8, 3 ECTS

Objectifs :

Niveau C1 du CECRL (Cadre Européen Commun de Référence pour les Langues)Permettre aux étudiants de développer les compétences indispensables à la réussite dans leur future vie professionnelle en contextes culturels variés.Acquérir l'autonomie linguistique nécessaire et perfectionner les outils de langue spécialisée permettant l'intégration professionnelle et la communication d'une expertise scientifique dans le contexte international.

Compétences :

- S'exprimer avec aisance à l'oral, devant un public, en usant de registres adaptés aux différents contextes et aux différents interlocuteurs.
- Se servir aisément d'une langue vivante autre que le français : compréhension et expression écrites et orales
- Comprendre un article scientifique ou professionnel rédigé en anglais sur un sujet relatif à leur domaine
- Interagir à l'oral en anglais : réussir ses échanges formels et informels lors des colloques, réunions ou entretiens professionnels.
- Défendre sa candidature par écrit (CV) ou à l'oral (entretien de recrutement) en anglais

Contenu :

Développer :

- les compétences liées à la compréhension de publications scientifiques ou professionnelles rédigées en anglais ainsi que les compétences nécessaires à la compréhension de communications scientifiques orales.
- les outils d'expression permettant de maîtriser une présentation orale et/ou écrite et d'aborder une discussion critique dans le domaine scientifique
- la maîtrise des éléments d'argumentation critique à l'oral et/ou à l'écrit d'une publication scientifique
- une réflexion plus large sur leur place, leur intégration et leur rayonnement en tant que scientifiques dans la société, abordant des questions d'actualité, d'éthique, d'intégrité...

Pré-requis :

Niveau B2 du CECR

Mots-clés :

Projet Anglais scientifique Rédaction Publication Communication esprit critique scientifique interculture