Syllabus du M2 Sciences Du Logiciel

Ingénierie des systèmes et des modèles (ISM)

S9, 6 ECTS

Objectifs :

Ce module fournit une initiation à la modélisation système (MBSE) dans le contexte de la conception dirigée par les modèles de systèmes hétérogènes comportant du logiciel et des composants mécaniques, capteurs, actionneurs, … et faisant intervenir des spécialistes de différents domaines. Seront évoqués l'ingénierie des exigences, la notion de ligne de produits, la modélisation SysML et l'étude des méthodes et outils de métamodélisation, d'édition, de vérification et de transformation de modèles.

Les compétences attestées seront la capacité de modéliser des exigences système en SysML et de définir des langages dédiés outillés.

Les étudiants auront aussi acquis la capacité de s'intégrer dans des équipes d'ingénierie système et seront familiarisés avec les notions de lignes de produits.

Contenu :

Cours / TD :

1. Introduction à l'ingénierie système
2. Ingénierie des exigences

TD : Étude d'un cahier des charges

1. Lignes de produits
2. Modélisation système en SysML
3. Langages dédiés et méta-modélisation
4. Expression et vérification de propriétés statiques
5. Génération d'éditeurs arborescents, textuels et graphiques
6. Scénario d'exécution et simulation de modèles
7. Transformation de modèles

TPs:

1. Prise en main d'un environnement SysML
2. Etude de cas SysML
3. Méta-modélisation d'un langage dédié
4. Génération d'un éditeur arborescent et textuel
5. Expression et vérification de propriétés statiques
6. Définition d'un langage cible et expression d'une transformation de modèle

Pré-requis :

- modélisation UML
- langage OCL

Références bibliographiques :

1. OMG SYSML
2. Ingénierie dirigée par les modèles. Des concepts à la pratique. JM Jezequel, B. Combemale, D. Vojtisek, Ellipses, 2012
3. Model-Driven Software Engineering in Practice. M. Brambilla, J. Cabot, Morgan & Claypool Publishers 2012

4. https://www.eclipse.org/Xtext/
5. https://www.uml-sysml.org/sysml/

Mots-clés :

- Ingénierie Système, Ingénierie des exigences, modélisation système, ligne de produits, ingénierie des modèles, vérification de modèles, transformation de modèles

Modalités d'évaluation :

- Travail de groupe: études de cas en ingénierie système et en ingénierie des modèles
- Contrôle individuel

Développement Orienté Plateforme

S9, 6 ECTS

Objectifs :
Ce module a pour objectif de renforcer les connaissances dans le développement de plateformes applicatives. Il se compose de trois parties : la première vise à acquérir une connaissance générale sur la mise en œuvre des architectures multicouches dans le cadre de la plateforme .NET ; la deuxième a pour objectif le développement d'applications natives Android ; la troisième concerne le développement cross-plateforme d'applications mobiles.

Connaissances :
- Frameworks de développement et leurs liens avec les plateformes

Compétences :
- Argumenter sur le choix de la plate-forme à utiliser
- Développer une application web et/ou mobile
- Développer une application mobile en coopération avec des services en ligne

Contenu :

Développement orienté .NET 10hC, 10hTP
- Machine virtuelle
- Langage C# et environnement de développement
- Mise en oeuvre des architectures orientées services, services web

Développement d'applications mobiles natives pour Android 4hC, 20hTP
- Langage Kotlin
- Outil de build Gradle
- Framework Android

Développement d'applications cross-plateformes 4hC, 8hTP
- Langage Dart et framework flutter
- Développement de composants Flutter
- Mise en oeuvre de consommateurs de services web

Pré-requis :
- Connaissance des architectures multi-couches, MVC
- Programmation orientée objet
- Mise en place d'un Web Service REST
- Manipulation du format de donnée JSON

Références bibliographiques :
- .Net (https://docs.microsoft.com/fr-fr/dotnet)
- Android (https://developer.android.com/docs)
- Flutter (https://flutter.dev/docs)

Mots-clés :
- Plateforme applicative, .NET, Développement mobile, Kotlin, Android, Cross-platform

Modalités d'évaluation :

- Projet court en groupe de développement d'application mobile
- Challenge de développement mobile sur une journée
- Contrôle terminal individuel

Sécurité, Test et Optimisation des Applications Web (SécuTOAW)

S9, 6 ECTS

Objectifs :

Cette UE comporte deux matières dédiées aux applications Web : Sécurité et Test et Optimisation

- La partie Sécurité a pour objectif de sensibiliser les étudiants aux problèmes de sécurité spécifiques aux applications Web et de présenter les solutions architecturales et techniques pour y faire face.
- La partie TOAW a pour objectif d'une part de présenter les techniques de test des applications objet et d'autre part de renforcer les connaissances dans le développement des applications web côté backoffice.

  Compétences :
  - Effectuer une campagne de tests en utilisant des outils de test appropriés axés sur des attributs de qualité spécifiés par l'équipe qualité et le client.
  - Optimiser l'efficacité et la qualité d'une application web côté serveur.

Contenu :

Partie Sécurité :

1. Contexte technique et juridique de la sécurité informatique
   - Présentation d'incidents, temps moyen de survie
   - Cybercriminalité, intelligence économique et sécurité informatique
   - Lois relatives à la sécurité informatique, vie privée et secret des correspondances
2. Risques, taxonomie d'incidents, exemples
   - Risques et évolutions des risques
   - Taxonomie d'incidents et d'insécurités
   - Erreurs de codage : injection de commandes, dépendances indirectes, contournement d'authentification
3. Mise en place de la sécurité informatique
   - Architectures et réseaux sécurisés
   - Développement sécurisé, Owasp, CWE, Analyse des CVEs
   - Relais et firewall applicatifs
   - Maîtrise des flux applicatifs, intégration dans un environnement sécurisé
   - Silos et conteneurs applicatifs

Partie TOAW :

1. Test des applications objet :
   - tests unitaires, d'intégration et fonctionnels
   - mock objects
2. Développement avancé côté backoffice :
   - cohabitation avancée d'un modèle objet et relationnel d'un SGBD : gestion des relations, cascading, cas de l'héritage
   - optimisation des requêtes : cache, n+1 select, pagination
   - gestion transactionnelle
   - concurrence et optimistic locking

- asynchronisme

Technos : JUnit, Mockito, Cucumber, Spring Boot

Pré-requis :
- Programmation orientée objet, développement élémentaire côté backoffice
- Développement logiciel et web
- Connaissances de base en protocole réseau (IP, TCP, UDP, HTTP)
- Systèmes d'exploitation

Références bibliographiques :
- "Test Driven Development: By Example". Kent Beck, 2003.
- "xUnit Test Patterns: Refactoring Test Code". Gerard Meszaros, 2007.
- J2EE : https://jakarta.ee/

Mots-clés :
- Modèles de sécurité; politiques et propriétés de sécurité, architectures de sécurité, détection d'intrusion
- Test, backoffice

Modalités d'évaluation :
- Rendus de TP
- TP noté
- Contrôle terminal

Vérification et validation, analyse formelle (VVAF)

S9, 6 ECTS

Objectifs :

Ce module fournit une initiation à l'application de méthodes formelles pour la spécification et la vérification de systèmes dits critiques pour lesquels une défaillance peut avoir des conséquences graves. Après une présentation de langages d'expression de propriétés et de comportements, on s'interessera à trois méthodes: la construction correcte de spécification par raffinements, la vérification de comportements temporels puis temporisés par model checking et l'interprétation abstraite.

Les compétences attestées seront la capacité d'exprimer des propriétés ou des comportements dans des formalismes permettant leur vérification.

Les compétences acquises comprendront aussi la connaissance des principes sur lesquels reposent les outils de model checking et d'interprétation abstraite.

Contenu :

Cours/TD:

1. Introduction
    a. Systèmes réactifs, temps réel et critiques
    b. Model Checking, Preuve, Raffinement
2. Spécification de propriétés
    a. Logique classique et théorie des ensembles
    b. Logique temporelle linéaire et arborescente
    c. Logiques temporisées
3. Spécification et vérification de comportements
    a. Automates finis et symboliques (Event-B)
    b. Langages pour l'algorithmique distribuée, espace d'état
    c. Automates de Büchi et logique temporelle linéaire
    d. Automates temporisés
4. Développement par raffinements (Event-B)
5. Vérification par interprétation abstraite

TPs:

1. Modélisation d'algorithmes distribués, expression et vérification de propriétés
2. Développement par raffinements
3. Modélisation et vérification de systèmes temps-réel
4. Vérification par interprétation abstraite

Pré-requis :

logique classique, automates, algorithmique

Références bibliographiques :

- Emerson. Verification of Reactive Systems: Formal Methods and Algorithms. Springer.
- Abrial. Modeling in Event-B: System and Software Engineering. CUP
- R. Alur and D. L. Dill. A theory of timed automata. TCS, 1994.
- P. Cousot, R. Cousot. Abstract Interpretation... POPL 1977
- http://www.uppaal.org
- https://frama-c.com

Mots-clés :

spécification, vérification, model checking, interprétation abstraite

Modalités d'évaluation :

- Evaluation de rendus de TP
- contrôle terminal

Professionnalisation en Sciences Du Logiciel (partie 2)

S9, 6 ECTS

Objectifs :
La mise en situation professionnelle, dans le cadre d'un contrat d'apprentissage ou professionnalisation, ou sous convention de stage, est répartie sur les deux semestres. Elle s'effectue en alternance avec les cours (3 jours par semaine en entreprise ou en laboratoire - 2 jours en formation). Elle permet aux étudiants de contribuer à un projet dans un contexte professionnel sur une longue période (début entre septembre et novembre, fin en juillet) et de participer à différentes phases. L'objectif est de compléter la formation des étudiants au métier d'ingénieur ou de chercheur, et de faciliter ainsi leur insertion et leur devenir professionnel.

Contenu :
Il n'y pas d'enseignement "traditionnel" dans le cadre de cette UE. Cependant, les étudiants sont accompagnés individuellement dans leur recherche d'un poste (préparation de CV, LM, et entretiens de recrutement). Par ailleurs, les étudiants sont suivis par un enseignant.

- Recherche d'un poste dans une entreprise ou un laboratoire de recherche
- Réalisation d'une mission de niveau ingénieur en entreprise ou travail d'initiation à la recherche dans un laboratoire (mission initiée dans cette UE du S9 et poursuivie au S10)
- Communication orale sur la mission

Pré-requis :

- Expérience acquise dans un précédent stage ou dans le cadre d'un projet universitaire ou personnel
- Connaissances méthodologiques
- Connaissances théoriques et techniques en fonction de la mission

Mots-clés :
Professionnalisation, Expérience, Compétences, Candidature, Communication orale

Modalités d'évaluation :
Présentation orale : description du contexte de la mission, des objectifs de l'étudiant et de l'entreprise, de l'environnement méthodologique, technique et humain, et du planning prévisionnel de la période de professionnalisation.

CT: 100% (présentation orale) en session 1 et 2

Agilité à l'échelle et relation MOA-MOE (AGIREMm)

S10, 6 ECTS

Objectifs :

L'objectif de l'UE est d'approfondir les connaissances et compétences des étudiants pour les pratiques agiles dans des dimensions différentes (taille de l'équipe, dépendances et synchronisation dans les grands projets, gestion des flux...) ainsi que pour les formes de collaboration impliquant le triangle MOA-AMO-MOE (Maître d'ouvrage - assistant à maîtrise d'ouvrage et maître d'oeuvre).

- Insérer ses activités de projet dans une pratique SAFe, Lean ou Kanban
- Identifier les points d'amélioration de la pratique agile de l'équipe
- Définir les pratiques agiles à mettre en oeuvre pour un projet donné
- Faciliter la relation MOA-MOE à travers le rôle de l'AMO

Contenu :

- Lean, Just In Time, filiation Lean eXtreme Programming Agile SAFe
- XP 3 cercles, qualité intrinsèque, test 1st Prog. Clean Code
- Application à une étude de cas : TDD, refactoring, conception émergente, conception collaborative
- Application à une étude de cas : standards de codage, intégration continue, métaphore, stories tests…
- Design (au sens historique), collaborative design, set-based design
- Kanban : flux tiré, WIP, goulot d'étranglement, VSM, intro vers SAFe (Lean Entreprise)
- SaFE : rôles et études de cas
- Introduction à Spotify

- Rôle et périmètre d'action de l'AMO
- AMO : facilitateur de projet

Pré-requis :

- Connaissances des valeurs de l'agilité et de la méthode SCRUM
- Gestion de projet selon le PMBoK

Références bibliographiques :

- Manifeste pour le développement agile de logiciel, http://agilemanifesto.org/
- SAFe - Scaled Agile Framework, https://www.scaledagileframework.com

Mots-clés :

Lean, Kanban, SaFE, Spotify, MOA-AMO-MOE

Modalités d'évaluation :

Evaluation de livrables :

Plan de management de projet
Contrôle de connaissances
Artefacts SCRUM
Gestion des réunions

Architecture Logicielle et Composants Logiciels (ALCoL)

S10, 6 ECTS

Objectifs :
Cette UE a pour objectif :

- de présenter les problèmes, les concepts et les techniques relatifs à l'architecture logicielle. On étudiera comment, en s'appuyant sur les exigences de différentes natures, concevoir et documenter l'architecture d'un système logiciel
- de présenter la problématique de la réutilisation, d'étudier les principes de la conception et de la programmation à base de composants logiciels,
- d'étudier les principes de la conception et de la programmation à base d'entité autonomes et communicantes : les acteurs
- de donner une introduction à l'intelligence artificielle distribuée et aux systèmes multi-agents

Contenu :
1. ARCHITECTURE LOGICIELLE (2 ECTS) :
Introduction à l'architecture logicielle : problématique, définitions

- Exigences fonctionnelles et extrafonctionnelles, parties prenantes
- Styles (patterns) d'architecture
- Conception architecturale : méthodes itératives, ADD
- Documentation architecturale : points de vue et vues
- Conception et documentation

2. CONCEPTION ET PROGRAMMATION PAR COMPOSANTS (2 ECTS)
Problématiques de la réutilisation et de la flexibilité logicielles

- Composants logiciels, connecteurs, interfaces fournies et requises, composition
- Configuration, reconfiguration
- Modèles de composants « académiques » et « industriels »
- Les composants dans UML2
- Ingénierie des composants logiciels

3. CONCEPTION ET PROGRAMMATION CONCURRENTE PAR ACTEURS (2 ECTS)

- Modèles d'interaction (communication, synchronisation) : échanges de messages, futurs, client-serveur
- Patterns de programmation concurrente
- Modèle de conception et de programmation par acteurs
- Conception à base d'acteurs, mise en oeuvre avec AKKA

4. OUVERTURE VERS LES SYSTEMES MULTI-AGENTS ET L'INTELLIGENCE ARTIFICIELLE DISTRIBUEE (décentralisation, autonomie, interaction, émergence)

Pré-requis :

- Conception et programmation objet, UML, Design patterns, Multithreading

Références bibliographiques :

- Software Engineering (10th ed), I. Sommerville, Pearson, 2016
- Software Architecture in Practice (3rd ed), L. Bass et. al., Addison Wesley, 2013
- Pattern-Oriented Software Architecture - Patterns for Concurrent and Networked Objects, D. Schmidt et al., Wiley, 2008
- AKKA : https://akka.io/

Mots-clés :

- Conception, exigence, flexibilité, réutilisation, composants, communication, concurrence, distribution

Modalités d'évaluation :

- Contrôle sur table
- Rapport
- Code

DevOps et architectures micro-services

S10, 3 ECTS

Objectifs :
Cette UE porte sur les problèmes et les méthodes, techniques et outils intervenant dans la vie d'un logiciel complexe, en particulier dans le développement d'une application distribuée en micro-services.
Compétences visées :

- Définir un pipeline de CI/CD pour automatiser la vérification/validation et le déploiement
- Architecturer une application web comme application composite à base de conteneurs
- Faire évoluer l'application vers une architecture micro-services

Connaissances acquises :

- Intégration continue au sens de Martin Fowler, et dans un workflow décentralisé (feature branches, PRs)
- Automatisation d'une configuration d'une plateforme de CI/CD
- Orchestration de conteneurs
- Spécificités d'une architecture micro-services
- Manipulation des métriques techniques et fonctionnelles

Contenu :
Le cours se découpe en trois parties complémentaires, et les éléments du cours de chaque partie sont mis en pratique à plusieurs niveaux (application directe du cours par un TP en monôme ou binôme et/ou mise en œuvre des méthodologies sur un cas d'étude réalisé en séance de TD dans un contexte agile) :

1. Vérification, validation et intégration continue
2. Virtualisation applicative, orchestration de conteneurs et déploiement continu
3. Architectures micro-services

Pré-requis :

Programmation orientée objet (Java), Développement web (Jakarta EE, JavaScript), Architecture REST, Tests unitaires et d'intégration, Gestion de version (git), Environnement de développement linux (shell)

Références bibliographiques :

- Kent Beck. Embracing change with extreme programming. Computer, 32(10):70--77, 1999.
- Ian Miell & Aidan Hobson Sayers. Docker in Practice. Manning Publications, 2016.
- Docker CE (https://www.docker.com/)
- GitHub Actions (https://help.github.com/en/actions)
- Ansible (https://docs.ansible.com/ansible/latest/index.html)

Mots-clés :

- Automatisation, DevOps, CI/CD, Docker, Microservices, Métriques

Modalités d'évaluation :

- Rendus de TD/TP (40%), ABI=ABJ=0
- Contrôle terminal (60%), ABI=ABJ=DEF

Professionnalisation en Sciences Du Logiciel (partie 3)

S10, 12 ECTS

Objectifs :
La mise en situation professionnelle, dans le cadre d'un contrat d'apprentissage ou professionnalisation, ou sous convention de stage, est répartie sur les deux semestres. Elle s'effectue en alternance avec les cours (3 jours par semaine en entreprise ou en laboratoire - 2 jours en formation). Elle permet aux étudiants de contribuer à un projet dans un contexte professionnel sur une longue période (début entre septembre et novembre, fin en juillet) et de participer à différentes phases. L'objectif est de compléter la formation des étudiants au métier d'ingénieur ou de chercheur, et de faciliter ainsi leur insertion et leur devenir professionnel.

Contenu :
Il n'y pas d'enseignement "traditionnel" dans le cadre de cette UE. Les étudiants sont néanmoins suivi par un enseignant.

- Réalisation d'une mission de niveau ingénieur en entreprise ou travail d'initiation à la recherche dans un laboratoire (poursuite de la mission initiée au S9)
- Communication orale et écrite sur la mission

Pré-requis :
- Expérience acquise dans un précédent stage ou dans le cadre d'un projet universitaire ou personnel
- Connaissances méthodologiques
- Connaissances théoriques et techniques en fonction de la mission

Mots-clés :
Professionnalisation, Expérience, Compétences, Communication

Modalités d'évaluation :
- évaluation du travail réalisé (y compris le volet "communication")
- rapport
- soutenance