REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

OFFRE DE FORMATION PARCOURS INGENIEUR D'ETAT

Etablissement	Faculté / Institut	Département
-	Informatique	

Domaine	Filière	Spécialité
Mathématiques et Informatique MI	INFORMATIQUE	Software Engineering

الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

عرض تكوين مسار مهندس دولة

المؤسسة	الكلية/ المعهد	القسم
-	الإعلام الآلي	

الميدان	الشعبة	التخصص
رياضيات و إعلام آلي	إعلام آلي	هندسة البرمجيات

SOMMAIRE

I – Fiche d'identité de la Formation

1 - Localisation de la formation :

Faculté : Informatique

Département : ….

Références de l'arrêté d'habilitation de la formation (joindre copie de l'arrêté)

2- Partenaires extérieurs : (Champ obligatoire)

- Autres établissements partenaires :

….

- Entreprises et autres partenaires socio économiques :

…

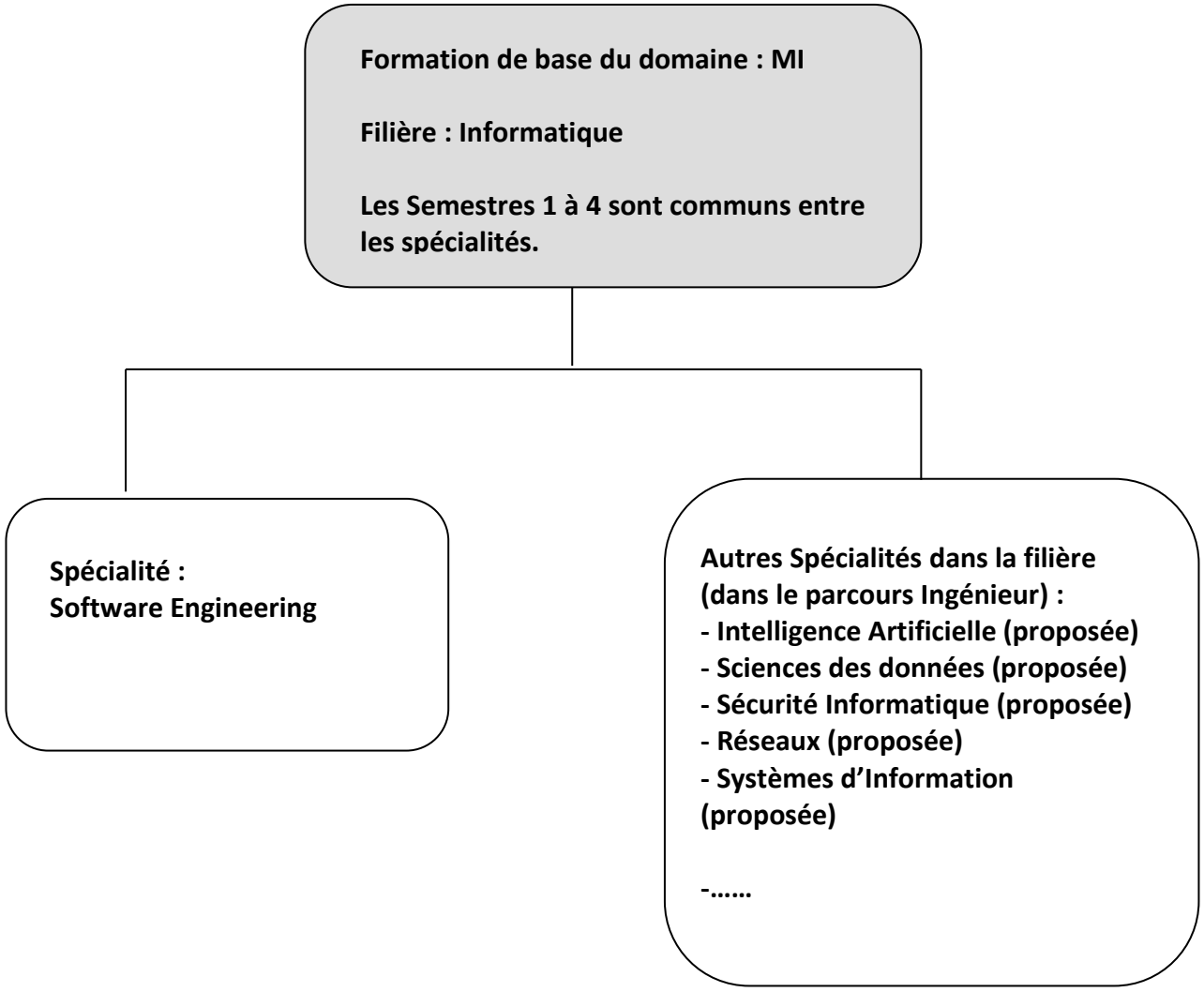
- Partenaires internationaux :

….

3 – **Contexte et objectifs de la formation**

A – Organisation générale de la formation : position du projet (Champ obligatoire)

Si plusieurs spécialités sont proposées ou déjà prises en charge au niveau de l'établissement (même équipe de formation ou d'autres équipes de formation), indiquer dans le schéma suivant, la position de ce projet par rapport aux autres parcours.



B - Objectifs de la formation (Champ obligatoire)

(Compétences visées, connaissances acquises à l'issue de la formation- maximum 20 lignes)

Le monde de l'informatique étant en évolution croissante, la présente offre de formation en Software Engineering vise les principaux objectifs suivants :

- Former des ingénieurs ayant des compétences en ingénierie des logiciels leur permettant de maitriser le cycle de développement de bout en bout, de l'analyse des besoins jusqu'à la mise en place de prototypes, tests, déploiement et maintenance de solutions logicielles.
- A travers cette formation, les étudiants pourront maitriser notamment, des aspects avancés des bases de données (distribuées, semi-structurées, NoSQL, …), les méthodes de développement agiles, le développement collaboratif, l'ingénierie des systèmes d'information modernes, les technologies du Web (IoT, Cloud computing, Big data …), le développement mobile ainsi que les aspects de qualité (assurance qualité) et de sécurité qui sont importantes pour tout produit logiciel.
- Former des compétences permettant de développer des solutions support aux métiers des entreprises, pouvant aussi assister les décideurs dans la prise de décision à travers la maitrise des outils d'analyse de données massives et variées.
- Des notions académiques approfondies seront enseignées aux étudiants, leur permettant également de poursuivre des études de post-graduation

C – Profils et compétences visées (Champ obligatoire) *(maximum 20 lignes) :*

Les compétences visées à l'issue de cette formation sont :

- Maitriser le développement d'un produit logiciel de qualité
- Savoir gérer un projet de développement logiciel
- Maitriser la collaboration dans le cadre d'un projet de développement
- Maitriser les outils de développement et les environnements technologiques actuels (Cloud, Big data, NoSQL, …)
- Savoir répondre aux besoins de l'entreprise et des décideurs, à travers la maitrise des méthodes et outils informatiques adéquats
- Maitriser les nouvelles technologies et l'impact de ces dernières sur le SI de l'entreprise

D – Potentialités régionales et nationales d'employabilité (Champ obligatoire)

Les retombées de cette formation concernent aussi bien le contexte régional que le contexte national au vu des besoins immenses en matière de compétences dans le domaine du développement de logiciels (à tous les niveaux) pour le secteur économique public et privé.
Les débouchés en matière d'employabilité concernent les profils suivants :

- Ingénieur développeur,
- Ingénieur software,
- Ingénieur devOps,
- Ingénieur design, développement et maintenance logicielle,
- Etc.

E – Passerelles vers les autres spécialités (Champ obligatoire)

Les passerelles peuvent se faire avec d'autres spécialités d'ingénieur en Informatiques (Intelligence Artificielle, et Administration et sécurité des systèmes et des réseaux) du fait que la formation actuelle en Software Engineering renferme des crédits fondamentaux en informatiques qui peuvent se faire valoir comme crédits d'autres spécialités en informatique.

F – Indicateurs de performance attendus de la formation (Champ obligatoire)

(Critères de viabilité, taux de réussite, employabilité, suivi des diplômés, compétences atteintes…)

- Comités pédagogiques,
- Réunions-bilans périodiques,
- Suivi du placement des étudiants dans le secteur économique

4 – Moyens humains disponibles

A – Capacités d'encadrement (exprimé en nombre d'étudiants qu'il est possible de prendre en charge)

B –Equipe pédagogique interne mobilisé pour la spécialité

C : Equipe pédagogique externe mobilisée pour la spécialité : (à renseigner et faire viser par la faculté ou l'institut)

Nom, prénom	Etablissement de rattachement	Diplôme graduation	Diplôme de spécialité (Magister, doctorat)	Grade	Matière à enseigner	Emargement

Visa du département

Visa de l'établissement

D : Synthèse globale des ressources humaines mobilisées pour la spécialité :

Grade	Effectif Interne	Effectif Externe	Total
Professeurs			
Maîtres de Conférences (A)			
Maîtres de Conférences (B)			
Maître Assistant (A)			
Maître Assistant (B)			
Autre (*)			
Total			

(*) Personnel technique et de soutien

5 – Moyens matériels spécifiques à la spécialité

A- Laboratoires Pédagogiques et Equipements : Fiche des équipements pédagogiques existants pour les TP de la formation envisagée (1 fiche par laboratoire)

Intitulé du laboratoire : Laboratoire de TP de la faculté d'Informatique/ laboratoire de TP de la faculté des Mathématiques

Capacité en étudiants :

N°	Intitulé de l'équipement	Nombre	Observations
1	Micro-ordinateurs (…)		
2	Routeurs		
3	Switch		
4	KINECT et caméras		
5	TINY Machine learning (Arduino)		
	…		
	Logiciels disponibles		
6	Systèmes d'exploitation Linux Environnement de programmation parallèles et distribués		
7	Logiciels open source pour le développement de jeux sur Android		

B- Terrains de stage et formations en entreprise (voir rubrique accords / conventions) **:** (Champ obligatoire)

Lieu du stage	Nombre d'étudiants	Durée du stage
		1 semestre
		1 semestre
		1 semestre
		1 semestre
		1 semestre
		1 semestre
		1 semestre

C- Documentation disponible au niveau de l'établissement spécifique à la formation proposée (Champ obligatoire) :

La faculté d'Informatique dispose d'une bibliothèque Interne accessible à tous les étudiants de la filière.

D- Espaces de travaux personnels et TIC disponibles au niveau du département de l'école :

…..

E- Support d'apprentissage

Indiquer la plateforme de diffusion des enseignements :

Type de Plateforme (Moodle,….)	*Etablissement parraineur*	*Lien de la plateforme*
Moodle		https://campusvirtuel.**université**.dz/
Google Classroom	Google	https://classroom.google.com/
Google meet	Google	https://meet.google.com/

II – Fiche d’organisation semestrielle des enseignements de la spécialité

1- Semestre 5 :

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff.	Crédits	Mode d'enseignement		Mode d'évaluation	
	14-16 sem	C	TD	TP	Autres			A distance	En présentiel	Continu	Examen
UE fondamentales											
UEF1(O/P)											
Algorithmique et Complexité Avancées	**84H**	**03H00**	01H30	01H30		**4**	**6**		X	40	60
Génie logiciel (GL)	84H	03H00	01H30	01H30		**4**	**6**		X	40	60
UEF2(O/P)											
BDD : Administration et Architecture	63H	01H30		03H00		**4**	**5**		X	40	60
Système d'Exploitation : Synchronisation et Communication (SYS2)	84H	03H00	01H30	01H30		**3**	**5**		X	40	60
UE méthodologique											
UEM1(O/P)											
Techniques d'Optimisation (TOp)	42H	01H30	01H30			**3**	**5**		X	40	60
UE découverte											
UED1(O/P)											
Fondements de l'IA (FIA)	42H	01H30		01H30		2	3		X	40	60
Total Semestre 5	**399H**	**13H30**	**06H00**	**09H00**		**20**	**30**				

- Volume hebdomadaire = 28H30 / semaine
- 7 Matières par semestre.

2- Semestre 6 :

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff.	Crédits	Mode d'enseignement		Mode d'évaluation	
	14-16 sem	C	TD	TP	Autres			A distance	En présentiel	Continu	Examen
UE fondamentales											
UEF1(O/P)											
Conception de logiciels	63H	01H30	01H30	01H30		**4**	**5**		X	40	60
Programmation WEB (PWEB)	63H	01H30		03H00		3	5		X	40	60
UEF2(O/P)											
BDD : Optimisation et gestion des accès concurrents	63H	01H30	01H30	01H30		3	5		X	40	60
Compilation 1 (COMPIL1)	84H	03H00	01H30	01H30		4	6		X	40	60
UE méthodologique											
UEM1(O/P)											
Analyse numérique	84H	03H00	01H30	01H30		**4**	**5**		X	40	60
UE découverte											
UED1(O/P)											
Introduction à la sécurité Informatique (ISEC)	63H	01H30	01H30	01H30		2	4		X	40	60
Total Semestre 6	**420H**	**12H00**	**07H30**	**10H30**		**20**	**30**				

- Volume hebdomadaire = 30H00 / semaine.
- 6 Matières par semestre.

3- Semestre 7 :

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff.	Crédits	Mode d'enseignement		Mode d'évaluation	
	14-16 sem	C	TD	TP	Autres			A distance	En présentiel	Continu	Examen
UE fondamentales											
UEF1(O/P)											
Concepts avancés de BD	63h	1h30	1h30	1h30		3	4		x	40%	60%
Gestion de projets (GP)	63h	1h30	1h30	1h30		3	4		x	40%	60%
Web avancé et Micro-services	42h	1h30		1h30		2	3		x	40%	60%
UEF2(O/P)											
Data-Mining	63h	1h30	1h30	1h30		3	5		x	40%	60%
Compilation 2	63h	1h30	1h30	1h30		3	5		x	40%	60%
UE méthodologie											
UEM1(O/P)											
Méthodes de management agiles	42h	1h30		1h30		2	3		x	40%	60%
Réseaux et protocoles	42h	1h30		1h30		2	3		x	40%	60%
UE découverte											
UED1(O/P)											
IHM : Conception et évaluation des interfaces	42h	1h30		1h30		2	3		X	40%	60%
Total Semestre 7	**420h**	**12h00**	**6h00**	**12h00**		**20**	**30**				

- Volume hebdomadaire = 30H00 / semaine.
- 8 Matières par semestre.

4- Semestre 8 :

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff.	Crédits	Mode d'enseignement		Mode d'évaluation	
	14-16 sem	C	TD	TP	Autres			A distance	En présentiel	Continu	Examen
UE fondamentales											
UEF1(O/P)											
Architecture et Gestion des Systèmes d'Information Avancés	63h	1h30	1h30	1h30		3	4		X	40%	60%
Big-Data et Base de données NoSQL	63h	1h30	1h30	1h30		3	4		X	40%	60%
UEF2(O/P)											
Architectures Logicielles	42h	1h30		1h30		2	4		X	40%	60%
Modèles et Gestion de Procédés Logiciels	63h	1h30	1h30	1h30		3	4		X	40%	60%
Test Logiciel et Assurance qualité	42h	1h30		1h30		2	4		X	40%	60%
UE Méthodologie											
UEM1(O/P)											
Modélisation et Evaluation des Performances	42h	1h30	1h30			2	3		X	40%	60%
Systèmes d'exploitation Mobiles	42h	01h30		1h30		2	3		X	40%	60%
UE Transversale											
UET1(O/P)											
Projet Pluridisciplinaire	42h			03H00		3	4		X		100%
Total Semestre 8	**399h**	**10h30**	**6h00**	**12h00**		**20**	**30**				

- Volume hebdomadaire = 28H30 / semaine.
- 8 Matières par semestre.

5- Semestre 9 :

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff.	Crédits	Mode d'enseignement		Mode d'évaluation	
	14-16 sem	C	TD	TP	Autres			A distance	En présentiel	Continu	Examen
UE fondamentales											
UEF1(O/P)											
Méthodes formelles pour GL	42h	1h30	1h30/15J	1h30/15J		3	4		X	40%	60%
Développement de logiciels embarqués	42h	1h30		1h30		3	4		X	40%	60%
UEF2(O/P)											
Conception de jeux vidéo : Théorie et Pratique	42h	1h30		1h30		3	4		X	40%	60%
Internet of Things (IoT) : Concepts et développement	63h	1h30	1h30	1h30		4	5		X	40%	60%
UE méthodologie											
UEM1 (O/P)											
DevOPs & Cloud Computing	84h	3h00		3h00		4	5		X	40%	60%
Sécurité Logicielle	42h	1h30		1h30		2	3		X	40%	60%
Développement mobile	42h	1h30		1h30		2	3			40%	60%
UE transversales											
UET1(O/P)											
Aspects juridiques	21h	1h30				1	2		X		100%
Total Semestre 9	**378h**	**13h30**	**2h15**	**11h15**		**22**	**30**				

- Volume hebdomadaire = 27H00 / semaine.
- 8 Matières par semestre.

6- Semestre 10 :

Domaine : Mathématiques et Informatique
Filière : Informatique
Spécialité : Software Engineering

Stage en entreprise sanctionné par un mémoire et une soutenance en présentiel ou à distance.

	VHS	Coeff	Crédits
Travail Personnel	30H	22	30
Stage en entreprise	/	/	/
Séminaires	/	/	/
Autre (préciser)	/	/	/
Total Semestre 10	420H	22	30

5- Récapitulatif global de la formation : (indiquer le VH global séparé en cours, TD, pour les 10 semestres d'enseignement, pour les différents types d'UE)

VH \ UE	UEF	UEM	UED	UET	Total
Cours	84	24	6	6	120
TD	50.25	7.5	1,5	0	59.25
TP	54.75	19.5	6	3	83.25
Travail personnel	30	0	0	0	30
Autre (préciser)	0	0	0	0	0
Total	219	51	13.50	9	292,5
Crédits	226	53	14	7	**300**
% en crédits pour chaque UE	75.33 %	17.66 %	4.66 %	2.33 %	100%

III - Programme détaillé par matière

(1 fiche détaillée par matière)

(tous les champs sont à renseigner obligatoirement)

Semestre : ***S5***

Unité d'enseignement : UEF1

Matière : ***Algorithmique et complexité avancées***

Crédits : 6

Coefficient : 4

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

L'objectif de ce cours est d'insister sur les critères de qualité d'un algorithme, notamment la validité et la complexité. L'étudiant ayant déjà un pré-requis sur la notion de complexité des algorithmes (module en S3), apprend à mieux analyser (en termes de validité et de complexité) les algorithmes des plus simples aux plus complexes, utilisant aussi les structures de données avancées (graphes, arbres, tas, tables de hachage). A l'issue de ce cours, L'étudiant devra savoir classer les problèmes (classes N, NL, P, NP, NP-complet), avoir une idée sur les problèmes les plus difficiles (les 21 problèmes NP-complets de Karp), savoir montrer la NP-complétude d'un problème, en utilisant le technique de réduction polynomiale).
Le cours est appuyé par des exercices de TD ainsi que des implémentations en TP, permettant d'évaluer une complexité expérimentale afin de justifier la complexité théorique établie.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Notions d'algorithmique, structures de données simples et avancées, éléments d'introduction à la complexité.

Contenu de la matière

Chapitre 1 : Théorie de la complexité

1) Introduction à la théorie de la complexité
2) Définition et analyse de la complexité d'un algorithme (complexité spatiale et temporelle, Exemples, complexité théorique vs pratique).
3) Types de complexité
4) Classes de complexité des problèmes (Définition des classes P et NP, Problèmes de décision et problèmes d'optimisation, Relation entre les classes P et NP (problème P vs NP)
5) Algorithmes déterministes et non déterministes
6) La classe NP-complet (Définition de la classe NP-complet et de ses propriétés, Technique de Réduction)
7) Exemples de problèmes NP-complets
   - Le problème SAT
   - Le problème du sac à dos
   - Le problème de la clique
   - Le problème du cycle hamiltonien
   - Le problème du voyageur de commerce

Chapitre 2. Preuve d'algorithmes

1) Tests et preuve théorique
2) 2 Propriété de terminaison

- Terminaison d‟un algorithme itératif
- Terminaison d‟un algorithme récursif
- Exemples

3) 3. Propriété de validité (correction)
   - Validité d'un algorithme itératif (invariant de boucle)
   - Preuve par induction
   - Validité d'un algorithme récursif
   - Validité d'un algorithme récursif (relation de récurrence)

Chapitre 3 : Paradigme « Diviser pour régner » et complexité

1) Introduction au paradigme « Diviser pour régner »
   - (Définition et principe du paradigme « Diviser pour régner »
   - Avantages et inconvénients du paradigme « Diviser pour régner »
   - Champs d'application du paradigme « Diviser pour régner »)
2) Stratégies de résolution de problèmes par « Diviser pour régner »
   - (Décomposition du problème en sous-problèmes plus simples, Résolution récursive des sous-problèmes, Combinaison des solutions des sous-problèmes)
3) Exemples d'algorithmes utilisant le paradigme « Diviser pour régner » & complexité
   - Tri par fusion (MergeSort)
   - Tri rapide (Quicksort)
   - Recherche linéaire vs recherche dichotomique
   - Suite de Fibonacci
   - Algorithme de Strassen pour la multiplication de matrices
   - Tours de Hanoi
4) Complexité d'algorithmes récursifs
   - Équation de récurrence pour les algorithmes récursifs
   - Les différentes variantes de master théorème pour la résolution d'équations de récurrence
5) Méthodes de résolution des équations de récurrence
   - Master théorème
   - Méthode par substitution
   - Méthode par deviner et tester (induction mathématique)
   - Arbre des appels récursifs

Chapitre 4 : La programmation dynamique

1) Introduction à la programmation dynamique
2) Différence entre programmation dynamique et recherche exhaustive
3) Exemples simples de problèmes résolus par programmation dynamique (ex:suite de Fibonacci)
4) Programmation dynamique vs Diviser pour régner
5) Approches de la programmation dynamique
   - Approche Top-Down (Memorization), Approche Bottom-Up (Tabulation)
6) Exemples d'algorithmes utilisant la programmation dynamique
   - Problème du sac à dos
   - Chemin le plus court
   - Séquence de nombres la plus longue croissante
7) Limites de la programmation dynamique

Chapitre 5 : Algorithmes gloutons

1) Introduction aux algorithmes gloutons
2) Exemples d'algorithmes gloutons

3) Analyse de la performance des algorithmes gloutons
4) Limites des algorithmes gloutons

Chapitre 6 : Analyse probabiliste et algorithmes randomisés

1) Introduction à l'analyse probabiliste et aux algorithmes randomisés
2) Analyse probabiliste d'algorithmes
3) Analyse de la complexité temporelle et spatiale des algorithmes randomisés
4) Exemples d'algorithmes randomisés
   - Tri randomisé
   - Recherche par hachage

Chapitre 7 : Algorithmes d'approximation

1) Définition et principe des algorithmes d'approximation
2) Techniques d'algorithmes d'approximation
3) Analyse de la performance des algorithmes d'approximation
4) Applications des algorithmes d'approximation

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

- Cormen, Leiserson, Rivest Stein, Algorithmique, cours exercices et problèmes, 3ème édition, Dunod, 2010.
- Ivan Lavallée, Complexité et algorithmique avancée, une introduction, édition Hermann 2008.
- Sylvain Perifel, Complexité algorithmique, Ellipses, 2014, 432 p. (ISBN - 9782729886929)
- Nicolas Hermann et Pierre Lescanne, Est-ce que P = NP ? Les Dossiers de La Recherche, 20:64–68, août-octobre 2005

Semestre : *S5*

Unité d'enseignement : UEF1

Matière : *Génie logiciel (GL)*

Crédits : 6

Coefficient : 4

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

- ✓ Acquérir les concepts de base du génie logiciel en termes d'objectifs de qualité, de coût et de délai.
- ✓ Maitriser les étapes d'un cycle de vie de logiciels
- ✓ Comprendre le processus de spécification et d'analyse des besoins en utilisant les diagrammes de cas d'utilisation UML
- ✓ Maîtriser les aspects de conception de logiciels
- ✓ Maîtriser la conception d'un diagramme de classes et de séquences UML.
- ✓ Acquérir les bases de la conception de qualité
- ✓ Exploiter les patrons de conception
- ✓ Maîtrise les techniques de test dans le but de détecter la présence d'erreurs dans un programme vis-à-vis de sa spécification.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

- ✓ *Maîtrise des concepts algorithmiques et de la programmation*
- ✓ *Notions de base en Programmation Orientée Objet.*

Contenu de la matière :

- ✓ **Chapitre 1 : Problématique et définitions (4 séances)**
  - Objectifs du génie logiciel
  - Définition d'un produit logiciel
  - Qualités d'un logiciel
  - Notion de cycle de vie d'un logiciel
  - Les étapes d'un cycle de vie de logiciels
  - Les différents modèles de cycle de vie (cascade, V, prototypage, spirale, incréments)
- ✓ **Chapitre 2 : Analyse & spécification des besoins (3 séances)**
  - Structure d'un cahier des charges
  - Modèles conceptuels
  - Besoins fonctionnels et non fonctionnels
  - Validation des besoins

- Introduction à UML : Modélisation unifiée des systèmes
- Définition des besoins à l'aide du diagramme de cas d'utilisation

✓ **Chapitre 3 : Conception des Logiciels (3 séances)**
- Notions de base et principes de la conception de logiciels
    - conception architecturale
    - conception détaillée
- Qualité de la conception et mesures : Couplage & Cohésion, Documentation
- Stratégies et méthodes de conception :
    - Conception Orientée Fonctions
    - Conception Orientée Données
    - Conception Orientée Objets

✓ **Chapitre 4 : Modélisation orientée objet à l'aide d'UML / Aspect Statique (2 séances)**
- Diagrammes de classes et d'objets.

✓ **Chapitre 5 : Modélisation orientée objet à l'aide d'UML / Aspect Dynamique (5 séances)**
- Les diagrammes d'interactions : de séquences et communication
- Les diagrammes d'états-transitions et d'activités.

✓ **Chapitre 6 : Les patrons de conception (3 séances)**
- Introduction aux patrons de conception
- Exemples : Singleton, Composition, Observateur …

✓ **Chapitre 7 : Tests des Logiciels (4 séances)**
- Introduction (origine et problématique
- Vérification statique (revue et inspection)
- Techniques de test
    - Test dynamique (structurel et fonctionnel)
    - Test statistique, de surcharge, de robustesse
    - Test Mutationnel (évaluer jeu de test)
- Automatisation des tests

✓ **Chapitre 8 : Concepts avancés (2 séances)**
- Conception Orientée Composants : Diagrammes de composants …
- Introduction aux méthodes Agiles (Méthodes XP, Méthodes SCRUM
- DevOps
- Référentiels de GL …

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références bibliographiques *(Livres et polycopiés, sites internet, etc)* :

Citer au moins 3 à 4 références classiques et importantes.

- ✓ Ian Sommerville. Software Engineering, Pearson Edition, 2015.
- ✓ Roger S. Pressman, Bruce Maxim. Software Engineering: A Practitioner's Approach 8th Edition, McGraw Hill, 2014.
- ✓ Stephens, Rod. Beginning software engineering, Wrox, a Wiley Brand, 2015.
- ✓ Systems Analysis and Design: An Object-Oriented Approach with UML 6th Edition, Alan Dennis, Barbara Wixom, David Tegarden, Wiley, 2020.
- ✓ Kim, G., Humble, J., Debois, P., Willis, J., & Forsgren, N. The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations. IT Revolution. 2021
- ✓ UML 2 : Modélisation des objets. Laurent Debrauwer et Fien Van der Heyde. Editeur ENI, Janvier 2013
- ✓ UML 2 par la pratique : Etude de cas et exercices corrigés. Pascal Roques. Editeur Eyrolles, Octobre 2011.
- ✓ Test logiciel en pratique Vuibert informatique John Warkins 2002.
- ✓ Object-Oriented Software Engineering: Using UML, Patterns and Java. Bernd Bruegge and Allen. H. Dutoit. Pearson International Edition, Prentice Hall, Second Edition, 2004.
- ✓ Architecture logicielle : Concevoir des applications simples, sûres et adaptables. Jacques Printz, Editeur Dunod, Juin 2012.
- ✓ Design Patterns pour Java : Mise en œuvre des modèles de conception en Java - Exercices et corrigés. Naouel Karam et Laurent Debrauwer, Editeur Eni, Novembre 2010.

***Semestre :* S5**

Unité d'enseignement : UEF2

Matière : ***BDD : Administration et Architecture***

Crédits : 5

Coefficient : 4

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

Ce module permet de cerner le niveau de stockage des données relationnelles**.** Seront présentées les contraintes d'intégrités **(**définition et représentation à l'aide d'un langage), l'organisation de données et structures d'accès, et les méta données, le mécanismes des vues ainsi que le contrôle d'accès.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoirsuivre cet enseignement – Maximum 2 lignes).*

Les concepts généraux de BD ainsi que leur modélisation et interrogation.

Contenu de la matière :

Chapitre 1 : Généralités sur les SGBD : concepts et architecture

Partie 1 : Présentation générale des BD et SGBD : Rappels

1- Généralités
   1-1 Définitions : BD et SGBD
   1-2 Historique des SGBD,
2- Niveaux de description et Architecture des SGBD
3- Objectifs et fonctionnalités d'un SGBD : Le SGBD ORACLE

Partie 2 : Gestion de l'intégrité

4- Vue d'ensemble des CI
   4-1 Les CI statiques
   4-2 Les CI dynamiques : les Triggers

Chapitre 2 : Les méta données

1- Introduction : nécessité de description des différents types de relations d'une base de données(relation système, relation de base, relation virtuelle)
2- Les concepts de base
   2-1 : les principaux catalogues ; Relation, Attribut, Index, Vue etc.
   2-2 : Gestion dynamique des catalogues
   2-3 : Le dictionnaire de données ORACLE

Chapitre 3 : Mécanisme des vues

1- Définition des vues
2- Interrogation à travers les vues
3- Mise à jour à travers les vues

Chapitre 4 : Gestion des privilèges d'accès

1- Privilèges système
2- Privilèges objets
3- Notion de rôle

Mode d'évaluation : Contrôle continu (40%), Examen (60%)

Références bibliographiques *(Livres et polycopiés, sites internet, etc)* :
Citer au moins 3 à 4 références classiques et importantes.

- J. Date. Introduction aux bases de données. Thomson publishing France 6ième édition. 1998
- C. Delobel et M. Adiba : bases de données et systèmes relationnels. Dunod 1982
- T. Connoly et Corolyn Begg. Systèmes de bases de données : approche pratique de conceptionde l'implémentation et de l'administration. Eyrolles 2005
- Frederic Brouard SQL CampusPress 2001
- Georges Gardarin : Les BD systèmes et leurs langages Eyrolles 2003
- Nacer Boudjlida : Gestion et Administration des Bases de Donnees. Application a Sybase etOracle. Dunod, Collection Sciences Sup. 2003.

Semestre : ***S5***

Unité d'enseignement : UEF2

Matière : ***Système d'Exploitation : Synchronisation et Communication (SYS2)***

Crédits : 5

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

Introduire la problématique du parallélisme dans les systèmes d'exploitation, étudier la mise en œuvre de la coopération à travers les outils de synchronisation et de communication et les problèmes associés.

Recommandation : Illustrer les différents concepts à l'aide d'outils (création de processus, signaux, sémaphores, mémoire partagée, files de messages, tubes) du système UNIX notamment dans les séances de TP.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Contenu de la matière :

Chapitre 1 : NOTION DE PARALLELISME/CONCURRENCE (10 %)

- Systèmes de tâches
- Parallélisme et langages évolués
- Déterminisme d'un système de tâches
- Parallélisme maximal

Chapitre 2 : SYNCHRONISATION DES PROCESSUS (20 %)

- Partage de ressources et des données
- Exclusion mutuelle
- Mécanismes d'exclusion mutuelle : variables d'états, solutions matérielles, les sémaphores, autres mécanismes de synchronisation

Chapitre 3 : COMMUNICATION ENTRE PROCESSUS (15 %)

- Communication par partage de variables : Modèle du producteur/ consommateur, modèle des lecteurs/ rédacteurs.
- Communication par échanges de messages : Désignation (directe : CSP, indirecte : les boites aux lettres)

Chapitre 4: MONITEURS ET REGIONS CRITIQUES (15%)

- Les moniteurs : Structure, implémentation, illustrations, variantes de moniteurs
- Régions critiques (Simples, Conditionnelles)

Chapitre 5: INTERBLOCAGE (20%)

- Caractéristiques, Modèles, Formalisation, Représentations graphiques
- Traitement de l'interblocage : Prévention, Détection- Guérison, Evitement

Chapitre 6: SYSTEMES DE FICHIERS (20%)

- Interface d'un système de fichiers : Attributs d'un fichier, opérations, types, structures, méthodes d'accès, répertoires.
- Implémentation d'un système de fichiers : Structure, organisation physique des fichiers et répertoires, implémentation, gestion des fichiers actifs et de l'espace libre, partage de fichiers, protection.
- Systèmes de fichiers multimédias
    - Types, structures et compression de fichiers multimédias
    - Opérations sur les fichiers multimédias
    - Organisation des fichiers multimédias

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références bibliographiques *(Livres et polycopiés, sites internet, etc)* :

Citer au moins 3 à 4 références classiques et importantes.

[1] Thomas Anderson and Michael Dahlin Operating Systems: Principles and Practice (Second Edition) Volume II: Concurrency ISBN : 978-0-9856735-4-3 Publisher: Recursive Books, Ltd.,. 2011-2015.

[2] M. J. Bach, traduit par G. Fellah, ''Conception du Système UNIX,'' Masson et Prentice Hall,1990.

[3] J. Beauquier, B. Berard,'' Systèmes d'exploitation : concepts et algorithmes,'' McGraw-Hill, 1990.

[4] Crocus,'' Systèmes d'exploitation des ordinateurs'', Dunod 1975.

[6] Patrick Cegielski, Conception de systèmes d'exploitation : le cas linux, Éditions Eyrolles, 16 Octobre 2003.

[7] N. B. Fontaine, P. Hammes, ''UNIX système V: Système et environnement'' Masson 1989.

[8] S. Krakowiak, ''Principes des systèmes d'exploitation des ordinateurs,'' Dunod 1987.

[9] J-L.Peterson, A.Silbershartz ''Operating Systems Concepts,''Addison-Wesly Publishing Company,Inc, 1983.

[10] A. Silberschatz, P. B. Galvin '' Principes des systèmes d'exploitation,'' 4 e Edition, Addison Wesley, 1994.

[11] William Stallings, Operating Systems: Internals and Design Principles (8th Edition), Publisher : Pearson; 8th edition (February 2, 2014).

[12] A. Tanenbaum, '' Modern Operating Systems, '' Third Edition Prentice Hall, 2009.

[13] A. Tananbaun, ''Systèmes d'exploitation,'' 3°édition, Pearson Edition. 2008.

[14] J-P. Verjus et al,''Synchronisation des programmes parallèles : Expression et mise en oeuvre dans les systèmes centralisés, édition 1983 Dunod.

Semestre : ***S5***

Unité d'enseignement : UEM1

Matière : Techniques d'Optimisation (TOp)

Crédits : 5

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

La recherche opérationnelle ainsi que les techniques d'optimisation revêtent une grande importance pour les ingénieurs informaticiens, particulièrement via les aspects algorithmiques. L'objectif du module est d'apprendre à l'étudiant à modéliser des problèmes combinatoires de différentes manières (programme linéaire, graphe, ...) et ce selon le type de problème. A travers ce cours, l'étudiant devra être capable d'identifier et de résoudre ces problèmes, trouver la solution optimale avec l'algorithme le plus adéquat.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Algèbre, Théorie des Graphes, Algorithmique 1, 2 & 3.

Contenu de la matière :

Chapitre 0 : La recherche opérationnelle. (1 séance : 5%)

Domaine de recherche opérationnelle.
Origine de la recherche opérationnelle.
Types de problèmes traités par la RO.
Brèves notions sur la complexité des algorithmes.
Domaines d'application de la recherche opérationnelle.

Chapitre 1 : Rappels. (2 séance : 12%)

Notions de base sur l'Algèbre linéaire.
Matrices : notions et éléments de base.
Systèmes linéaires.
Graphes : notions de base.
Aperçu sur quelques problèmes traités en théories des graphes : chemin optimal, ordonnancement.

Chapitre 2 : Programme linéaire. (2 séance : 15%)

Introduction par un exemple.
Forme générale, forme canonique, forme standard.
Modélisation.
Résolution graphique.

Chapitre 3 : Méthode du simplexe. (3 séance : 20%)

Algorithme général du Simplexe.
Algorithme à deux phases.

Chapitre 4 : La dualité en programmation linéaire. (2 séance : 12%)

PL dual.
Algorithme dual du simplexe.
Analyse post-optimale.

Chapitre 5 : Programmation linéaire en nombres entiers. (2 séance : 12%)

Introduction par un exemple.
Résolution graphique.
Méthode branch and bound (B&B).

Chapitre 6 : Programmation dynamique. (2 séance : 12%)

Introduction à la programmation dynamique.
Problème du sac à dos.

Chapitre 7 : Problème d'affectation (2 séances : 12%)

Couplage maximal.
Algorithme de l'Hongrois.

Bibliographie

1. J. Acher et J. Gardelle. *« Programmation linéaire »*. Livre. Editions Bordas. 1978.
2. A. Kaufmann et A. henry-Labordere. *« Méthodes et modèles de la recherche opérationnelle »*. Livre. Editions Dunod. 1974.
3. M. Sakarovitch. *« Optimisation Combinatoire – Tome 1 : Graphes et programmation Linéaire »*. Livre. Editions Hermann. 1984.
4. Roseaux. *« Exercices et problèmes résolus de recherche opérationnelle. Tome 1. Graphes: leurs usages, leurs algorithmes »*. Livre. Editions Dunod.1998.

Semestre : ***S5***

Unité d'enseignement : UED1

Matière : ***Fondements de l'IA (FIA)***

Crédits : 3

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

L'objectif de ce module est de présenter les fondements du domaine de l'intelligence artificielle. Dans ce cours, les principes de base des différents éléments de l'intelligence artificielle seront présentés : Logiques et Raisonnement, Résolution de problèmes, Méta heuristiques, Problème de satisfaction de contraintes, Apprentissage automatique et réseaux de neurones, Systèmes multi-agents.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Contenu de la matière :

Chapitre 1 : Logiques et Raisonnement

1.1 Logique propositionnelle
1.2 Logique du 1er ordre
1.3 Introduction aux logiques non classiques

Chapitre 2. Résolution de problèmes de planification

2.1. Représentation d'un problème par un espace d'états
2.2. Méthodes de recherche de solution dans les espaces d'états

Chapitre 3. Algorithmes pour les jeux

3.1 Algorithme Min Max
3.2 Algorithme Alpha Beta

Chapitre 4. Les méta-heuristiques

4.1 La recherche locale
4.2 Méthode Simulated Annealing (Recuit Simulé)
4.3 Algorithme génétique

Chapitre 5. Problème de satisfaction de contraintes (CSP)

5.1 Exemples de CSP
5.2 Recherche en arrière pour les CSPs (BackTracking)
5.3 Algorithme AC-3

Chapitre 6. Apprentissage automatique et réseaux de neurones

6.1 Algorithmes d'apprentissage automatique
6.2- Méthode des K plus proches voisins
6.3- Les réseaux de neurones

Chapitre 7 : Les systèmes multi-agents

7.1 Concepts de base

7.2 Les systèmes multi agents. (SMA) : Interaction, Communication, Négociation et Architecture

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références bibliographiques *(Livres et polycopiés, sites internet, etc)* :

Citer au moins 3 à 4 références classiques et importantes.

[1] Principles of Artificial Intelligence par Nils J. Nilson, 1982, Springer-Verlag Berlin Heidelberg.
[2] Essentials of Artificial Intelligence par Nils J. Nilson, 1993, Copyright: © Morgan Kaufmann
[3] Artificial Intelligence: A new synthesis, Nils Nilson, 1997, Elsevier.
[4]Artificial Intelligence: A Modern Approach par Stuart Russell and Peter Norvig, Ed. Pearson, 2016
[5] Rapport de synthèse - France Intelligence Artificielle. Ministère de l'Enseignement supérieur et de la Recherche, 2017.

Semestre : ***S6***

Unité d'enseignement : UEF1

Matière : ***Conception de logiciels***

Crédits : 5

Coefficient : 4

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Ce module vise à donner aux étudiants les compétences indispensables pour concevoir des logiciels de qualité. Il a pour but d'offrir une vision complète de la conception de logiciels, en couvrant à la fois les principes et les méthodes de base ainsi que les perspectives émergentes dans la matière. Le module s'article principalement autour de points suivants :

- Comprendre les principes de conception des gros logiciels.
- Savoir concevoir une expérience utilisateur intuitive.
- Comprendre les enjeux présents et futurs de la conception de logiciels.
- Savoir utiliser les principaux patrons de conception pour résoudre des problèmes de conception récurrents.
- Savoir utiliser des patrons d'architecture pour structurer une solution logicielle.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Génie Logiciel, Programmation orientée Objet

Contenu de la matière

- **Chapitre I : Introduction**

  Concepts clés de la conception de logiciels
  Les enjeux présents de la conception de logiciels
  Conception pour la scalabilité
  Conception pour l'expérience utilisateur (UX)

- **Chapitre II : Patrons de conceptions courants**

  Rappel sur les avantages et inconvénients des patrons, la description d'un patron, la classification
  Réutilisation de patrons de conception
  Exemples de patrons de conception

- **Chapitre III : Patterns architecturaux courants**

  Le patrons MVC
  Le patron PAC

- **Chapitre IV : Ingénierie Dirigée par les Modèles et conception de logiciels**

  La conception dans l'IDM
  La conception dans MDA (Model driven architecture)

- **Chapitre V : Les principes de conception**

  Principe de responsabilité unique

  Principe d'ouverture-fermeture

  Principe de substitution de Liskov

  Principe de séparation des interfaces

  Principe d'inversion des dépendances

- **Chapitre VI : Tendances futures de la conception de logiciels (optionnel)**

  Discussion sur les tendances futures de la conception de logiciels
  Impacts sur les processus de développement

- **Chapitre VII : Étude de cas d'un processus logiciel : 2TUP**

  Conception générique
  Conception préliminaire
  Conception détaillée

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

Dennis, Alan, Barbara Wixom, and David Tegarden. Systems analysis and design: An object-oriented approach with UML. John wiley & sons, 2015.

Kleppmann, Martin. "Designing Data-Intensive Applications: The Big Ideas Behind Reliable." Scalable, and Maintainable Systems 1st Edition–O'Reilly Media (2017).

Freeman, Eric, et al. Head First Design Patterns: A Brain-Friendly Guide. " O'Reilly Media, Inc.", 2004.

Freeman, Adam, and Adam Freeman. "The MVC Pattern, Projects, and Conventions." Pro ASP. NET Core MVC 2 (2017): 53-65.

Rubis, Ruslan. Patterns for Enterprise Application Design and Development. Diss. Florida Atlantic University, 2017.

Gamma, Erich, et al. "Elements of Reusable Object-Oriented Software." Design Patterns (1995).

Semestre : ***S6***

Unité d'enseignement : UEF1

Matière : ***Programmation WEB (PWEB)***

Crédits : 5

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

- Apprendre à créer des sites et applications Web.
- Maîtriser les standards du Web
- Maîtriser la programmation Web côté client et côté serveur.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

- Algorithmique et programmation en C, Bases de données et Langage SQL.

Contenu de la matière :

I. Chapitre I : Introduction au Web

- Historique et évolution du Web.
- Notions de bases du Web (HTTP, Client, Serveur, etc.).

II. Chapitre II : Langage HTML 5

- Historique et différentes versions
- Syntaxe du Langage : document bien formé, document conforme, outils de validation
- Présentation des éléments de base du Langage :
  - Sauts de ligne et séparateurs, titres, paragraphes, listes, tableaux, ….
  - Sections sémantiques (section, article, nav, …)
  - Les formulaires
- Les éléments multimédias (audio, vidéo,…) et les graphiques : Canvas et SVG

III. Chapitre III : Les feuilles de style CSS

- Séparation entre contenu et mise en forme et avantages
- Syntaxe de base de CSS :
  - Sélecteurs
  - Propriétés
- Media Queries et responsive design
- Gestion des conflits

IV. Chapitre IV : Le langage JavaScript

- Principe de la programmation web côté client
- Syntaxe de JavaScript
- Gestion des événements
- Objets JS prédéfinis

- Le DOM
- Gestion des exceptions

V. Chapitre V : Le Langage PHP 7

- Aperçu sur les langages de programmation coté serveur.
- Syntaxe du langage PHP.
- Fonctions PHP prédéfinies
- Traitement des formulaires en PHP
- Interaction avec les bases de données (MySQL).
- Sessions et cookies
- Gestion des fichiers

VI. AJAX

- Présentation
- Mode synchrone vs asynchrone
- L'objet XMLHTTPRequest
- Structure d'un code AJAX
- Interaction avec PHP

VII. Déploiement d'un Site/Application Web

- Modes de déploiement (Serveur local, hébergement en ligne).
- Etapes de déploiement
- Outils de déploiement.
- Notions de base sur la sécurisation d'un site Web (HTTPS)
- Exemple de déploiement.

VIII. Utilisation des Framework

- Limites de la programmation traditionnelle
- L'utilité des Framework
- Exemples de Framework CSS (ex. BootStrap)
- Exemples de Framework JavaScript (ex. JQuery ou Angular 2)

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références bibliographiques *(Livres et polycopiés, sites internet, etc)* :

Citer au moins 3 à 4 références classiques et importantes.

[1] World Wide Web Consortium, standards for web Design and applications : https://www.w3.org/standards/webdesign/
[2] Site Web partenaire du consortium W3C destiné aux développeurs Web. https://www.w3schools.com/
[3] Site Web officiel : Manuel du PHP. https://www.php.net/docs.php
[4] Rodolphe Rimelé « HTML 5 Une référence pour le développeur web », éditions Eyrolles, 2017.
[5] J. Engels « PHP 7 - Cours et exercices. », éditions Eyrolles, 2017.
[6] Mat Marquis « JavaScript pour les web designers », éditions Eyrolles, 2017.
[7] Michel Plasse « Développez en Ajax », éditions Eyrolles, 2006.

Semestre : ***S6***

Unité d'enseignement : UEF2

Matière : ***BDD : Optimisation et gestion des accès concurrents***

Crédits : 5

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

Maîtrise de concepts avancés des bases de données concernant l'optimisation et la gestion des accès concurrents ainsi que la reprise après pannes.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoirsuivre cet enseignement – Maximum 2 lignes).*

Les concepts généraux de BD ainsi que leur modélisation et interrogation..

Contenu de la matière :

Chapitre 1 : La mémoire relationnelle

1- La mémoire de stockage
2- Les différentes approches de stockage des données
3- Une approche de stockage : les B-arbres

Chapitre 2 : Fragmentation

1- Définition
2- Fragmentation horizontale
3- Fragmentation verticale
4- Fragmentation mixte

Chapitre 3 : Optimisation

1- Typologie de requêtes
2- Les différentes phases de traitement d'une requête
3- Optimisation de requêtes
    3-1 Méthode syntaxique
        a. Règles de transformation des arbres algébriques
        b. Algorithme simple de génération d'un plan de requêtes
    3-2 Estimation de coûts
        a. Utilisation de données statistiques

b. Techniques d'implémentation des opérateurs: cas de la jointure

Chapitre 4 : Gestion des accès concurrents :

1- Transaction : Définitions et caractéristiques
2- Les problèmes de concurrence
   2-1 Mise à jour perdue
   2-2 Non reproductibilité des lectures
   2-3 Dépendance non valide
3- Gestion de la concurrence
   3-1 Principe de sérialisation
   3-2 Les trois techniques de sérialisation : (états équivalents à ordonnancement série, permutation et/ou séparation des opérations permutables et/ou compatibles, et graphes deprécédence non cyclique)
   3-3 Les méthodes de sérialisation
      a- Par estampillage : estampillage simple, à deux estampilles
      b- Par verrouillage : différents types de verrouillage, graphe d'attente, V2P, verrousindéfinis

Chapitre 5 : Sureté de fonctionnement et reprise après pannes

1- Introduction : rôle du SGBD afférent à la gestion de pannes
2- Classification des pannes
3- Le processus de reprise après panne et états d'une transaction
4- Classification des mécanismes de reprise : mise à jour immédiates ou écriture en place, mise àjour différées ou écriture non en place, etc.
5- Les utilitaires de récupération : mécanismes de sauvegarde, outils de journalisation, utilitairesde points de contrôle (Checkpoint), gestionnaire de récupération
6- Les différentes Techniques de restauration : stratégies Undo, Redo etc.

Mode d'évaluation : Contrôle continu (40%), Examen (60%)

Références bibliographiques *(Livres et polycopiés, sites internet, etc)* :

Citer au moins 3 à 4 références classiques et importantes.

- J. Date. Introduction aux bases de données. Thomson publishing France 6ième édition. 1998
- C. Delobel et M. Adiba : bases de données et systèmes relationnels. Dunod 1982
- T. Connoly et Corolyn Begg. Systèmes de bases de données : approche pratique de conceptionde l'implémentation et de l'administration. Eyrolles 2005
- Frederic Brouard SQL CampusPress 2001
- Georges Gardarin : Les BD systèmes et leurs langages Eyrolles 2003
- Nacer Boudjlida : Gestion et Administration des Bases de Donnees. Application a Sybase etOracle. Dunod, Collection Sciences Sup. 2003.

Semestre : ***S6***

Unité d'enseignement : UEF2

Matière : ***Compilation 1 (COMPIL1)***

Crédits : 6

Coefficient : 4

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

L'objectif de ce cours sont multiples :

- Maitrise des langages de programmation : En effet, il est primordial de comprendre le fonctionnement des compilateurs afin de programmer efficacement,
- Conception et description de langages dédiés à des applications spécifiques. Le processus de compilation est souvent associé et restreint aux langages de programmation (procéduraux, orientés objets) alors qu'il peut être généralisé à une multitude de langages tels que :
    - les langages logiques (Prolog adapté aux stratégies d'essais et d'erreurs, aux problèmes de planification,…),
    - les langages fonctionnels (Lisp, Ocaml pour les applications de l'Intelligence Artificielle,…),
    - les langages de description de textes,
    - les langages de commandes de robots,
    - les langages à balise (HTML, XML,…),
    - les langages formels,
    - les langages de description et de conception de programme,
    - langage pour les protocoles (ASN1 : description des formats des messages),
    - les langages de test pour les protocoles (TTCN),
    - les langages de description des architectures (Rapid, Darwin,…),
    - les langages pour les techniques de preuves,
    - Les langages de conception du matériel (VHDL pour le matériel informatique),
    - les langages de modélisation graphique ou textuelle des systèmes et d'objets (tel que UML),
    - les langages de modélisation de la réalité virtuelle,
    - les langages Postscript,
    - les langages naturels,
    - les langages de description du Web intelligent (Web 2.0; web 3.0; web 4.0)
    - …

- Ecriture des compilateurs pour des processeurs spécifiques comme ceux présents dans les systèmes embarqués tels que les téléphones portables, les assistants personnels, les outils de positionnement (GPS), les caméscopes numériques, les systèmes de radiologie numériques,…
- Consolidation des connaissances acquises à travers différents modules à savoir les théories des Langages, Algorithmique, Architecture des ordinateurs, Théorie des graphes, Assembleur, les systèmes d'exploitation, la logique mathématique, …

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Théorie des langages, Architecture des ordinateurs, Algorithmique, Système d'exploitation, Logique Mathématique.

Contenu de la matière :

Introduction générale (2%)

1- Structure d'un compilateur
- 1.1 Phases d'analyse
  - 1-1-1 Analyse lexicale
  - 1.1.2 Analyse Syntaxique
  - 1-1-3 Analyse sémantique
- 1.2 Phases de production
  - 1.2.1 Génération de code
  - 1.2.2 Optimisation de code
- 1.3 Phases parallèles
  - 1.3.1 Gestion de la table de symboles
  - 1.3.2 Gestion des erreurs

2- Compilateurs, pré-compilateurs, traducteur et interpréteur

3- Rappels sur la théorie des langages
- 3.1 Les expressions régulières
- 3.2 Les automates
- 3.3 Equivalence entre expressions régulières et automates
- 3.4 Les grammaires

Chapitre I : Analyse lexicale (10%)
1- Introduction
2- Mise en œuvre d'un analyseur lexical
   - 2.1 Spécification des entités lexicales
   - 2.2 Correspondance entre expressions régulières et automates
   - 2.3 Algorithme d'analyse
   - 2.4 Création et gestion de la table des symboles
3- Gestion des erreurs lexicales

Chapitre II : Analyse syntaxique (45%)
1- Introduction
2- Les concepts de base
   - 2-1 Les formes normales des Grammaires
   - 2-2 Grammaire ε-libres et sans cycles
   - 2-3 Elimination de la récursivité à gauche directe et indirecte dans une grammaire
   - 2-4 Factorisation d'une Grammaire
3- Les méthodes d'analyse descendantes
   - 3.1 Les méthodes d'analyse non déterministes
   - 3.2 Les méthodes d'analyse déterministes
     - 3.2.1 La méthode d'analyse LL(1)
       - 3.2.1.1 Calcul des débuts et des suivants
       - 3.2.1.2 Construction la table d'analyse LL(1)
       - 3.2.1.3 Algorithme d'analyse LL(1)
     - 3.2.2 La Descente récursive

4- Les méthodes d'analyse Ascendantes déterministes
4.1 Analyse LR(1**)** par les contextes et les items
4.1.1 Construction de l'ensemble des contextes et des items LR(1)
4.1.2 Construction de la table d'analyse LR(1)
4.1.3 Algorithme d'analyse LR(1)
4.1.4 Analyse LR(K)

4.2 Analyse Simple LR(1) par les contextes et les items
4.2.1 Construction de l'ensemble des items LR(0)
4.2.2 Construction de la table d'analyse SLR(1)
4.2.3 Algorithme d'analyse SLR(1)

4.3 Analyse LALR(1) par les contextes et les items
4.3.1 Construction de l'ensemble des contextes et des items LR(1)
4.3.2 Construction de la table d'analyse LALR(1)
4.3.3 Algorithme d'analyse LALR(1)

5- Mise à jour de la table des symboles
6- Gestion des conflits dans le cas des tables d'analyse multi-définies
7- Relations entre les méthodes de l'analyse syntaxique
8- Gestion des erreurs syntaxiques
8-1 mode panique
8-2 Récupération au niveau syntagme

Chapitre III : Les formes intermédiaires (13%)

1- Introduction
2- La forme post-fixée
2.1 Evaluation des expressions arithmétiques et logiques
2.2 l'instruction d'affectation
2.3 Le branchement inconditionnel
2.4 Le branchement conditionnel
2.5 Les Instructions répétitives
2.6 Déclaration d'un tableau multidimensionnel
2.7 Référence à un élément d'un tableau multidimensionnel

3- Les quadruplets
3.1 l'instruction d'affectation
3.2 Le branchement inconditionnel
3.3 L'instruction conditionnelle
3.4 Les Instructions répétitives
3.5 Déclaration d'un tableau multidimensionnel
3.6 Référence à un élément d'un tableau multidimensionnel

4- Les triplets directs et indirects
4.1 L'instruction d'affectation
4.2 Le branchement inconditionnel

4.3 L'instruction conditionnelle
4.4 Les instructions répétitives
4.5 Déclaration d'un tableau multidimensionnel
4.6 Référence à un élément d'un tableau multidimensionnel

5- Les Arbres abstraits
4.1 L'instruction d'affectation
4.2 Le branchement inconditionnel
4.3 L'instruction conditionnelle
4.4 Les Instructions répétitives
4.5 Bloc d'instructions BEGIN-END
4.6 Déclaration d'un tableau multidimensionnel
4.7 Référence à un élément d'un tableau multidimensionnel

Chapitre IV : Traduction dirigée par la syntaxe (30%)

1- Schéma de traduction dirigée par la syntaxe dans le cas de l'analyse descendante
1.1 Les expressions arithmétiques
1.2 Les expressions logiques
1.3 Les structures de contrôle
1.3.1 Instruction conditionnelle
1.3.2 Instructions répétitives

2- Schéma de traduction dirigée par la syntaxe dans le cas de l'analyse ascendante
2.1 Découpage de la grammaire
2.2 Les expressions arithmétiques
2.3 Les instructions répétitives
2.4 Instruction du branchement inconditionnel

Travaux Dirigés :

Les travaux dirigés sont composés de 5 séries d'exercices permettant de couvrir l'ensemble du programme vu en Cours.

Série 1 : Grammaires/Automates et Analyse lexicale (1séance)
Série 2 : Analyse syntaxique descendante (3 séances)
Série 3 : Analyse syntaxique ascendante (3 séances)
Série 4 : Génération du code intermédiaire (3 séances)
Série 5 : Traduction dirigée par la syntaxe (3 séances)

Travaux Pratiques :

Les Travaux pratiques consistent à réaliser un compilateur d'un mini langage. Les séances seront organisées en 2 parties.

- La première partie sera dédiée à développer l'analyseur lexical (15%),
- La seconde partie sera consacrée à la mise en œuvre de la traduction dirigée par la syntaxe qui englobe (85%):
  - l'analyse syntaxique,
  - l'analyse sémantique,
  - la génération du code intermédiaire.

Le développement du compilateur pourra se faire de deux manières :

- soit en utilisant les générateurs des analyseurs lexicaux (tel que Flex) et les analyseurs syntaxico-sémantiques avec la génération du code intermédiaire (tel que Bison)
- soit en implémentant les algorithmes de l'analyse lexicale et de la traduction dirigée par la syntaxe (Analyse syntaxique, Analyse sémantique et Génération du code intermédiaire).

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références bibliographiques *(Livres et polycopiés, sites internet, etc)* :

Citer au moins 3 à 4 références classiques et importantes.

- Compilateurs: principes, techniques et outils - A. Aho, R. Sethi, J. Ullman – InterEditions.
- Compilateurs - D. Grune, H. Bal, C. Jacobs, K. Langendoen - Dunod.
- Principles of Compilers- V. Aho, J.D. Ullman - édition 2006.
- Modern compiler implementation in ML- A.W.Appel - Cambridge University Press 1998.
- The theory and practice of compiler writing - P.Q G. Sorenson - McGraw-Hill computer science series 1985.
- Flex & Bison - J.Levin- O'Reilly Media 2009.

Semestre : S6

Unité d'enseignement : UEM1

Matière *: Analyse numérique*

Crédits : 5

Coefficient : 4

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

Le module d'analyse Numérique est un module de base dans le domaine du calcul scientifique. Le calcul scientifique étant défini comme la branche qui permet d'implémenter une solution d'un problème sur un calculateur. L'objectif du module est l'acquisition des méthodes de discrétisation, des algorithmes de résolution des problèmes spécifiques posés dans le domaine de l'ingénierie informatique, l'analyse des algorithmes (rapidité, précision et souplesse), et enfin l'estimation des erreurs commises dans les modèles mathématiques employées ainsi que dans leur implémentation. Le domaine d'application est très vaste ; ça peut aller de la résolution d'un problème réel donné aux méthodes élaborées de l'apprentissage profond. Il est indispensable dans la formation d'ingénieur informaticien.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoirsuivre cet enseignement – Maximum 2 lignes).*

Algèbre1, Algèbre 2, Analyse Mathématique 1, Analyse Mathématique 2, Analyse Mathématique 3

Contenu de la matière :

Chapitre 1. Analyse d'erreurs

1.1 Introduction générale
1.2 Erreurs de Modélisation
1.3 Représentation des nombres sur un ordinateur :
1.3.1 Représentation des entiers signés
1.3.2 Représentation des réels.
1.3.3 Erreurs dues à la représentation.
1.4 Arithmétique Flottante
1.4.1 Opérations élémentaires
1.4.2 Opérations risquées
1.4.3 Evaluation des polynômes
1.5 Erreurs de Troncatures

Chapitre 2. Equations non linéaire

2.1 Introduction générale
2.2 Méthode de la bissection
2.3 Méthodes des points fixes :
2.3.1 Convergence de la méthode
2.3.2 Interprétation géométrique
2.3.3 Théorème d'Ostrovski.
2.3.4 Ordre de Convergence
2.3.5 Extrapolation d'Aitken
2.4 Méthode de Newton
2.4.1 Interprétation géométrique
2.4.2 Analyse de convergence
2.4.3 Théorème de Convergence Locale
2.5 Méthode de la sécante.
2.6 Applications

Chapitre 3.Systèmes d'équations linéaires

3.1 Introduction générale
3.2 Résolution des Systèmes linéaires triangulaires

Partie I Méthodes Directes

3.3 Elimination de Gauss
3.4 Décomposition LU ou l'interprétation matricielle de la méthode de Gauss
3.4.1 Décomposition de Crout
3.4.2 Décomposition LU et permutation de lignes
3.5 Décomposition de Cholesky
3.6 Méthode de GAUSS avec permutation
3.6.1 Pivot Partiel
3.6.2 Pivot Total
3.6.3 Pivot de la Toure (Rook Pivoting)
3.7 Systèmes tridiagonaux
3.8 Calcul de la Matrice inverse A^{-1}
3.9 Effets de l'arithmétique flottante
3.10 Conditionnement d'une matrice
3.8 Applications

Partie II Méthodes Itératives

3.9 Principe
3.10 Condition générale de Convergence
3.11 Applications :
3.11.1 Méthode de Jacobi
3.11.2 Méthode de Gauss-Seidel
3.11.3 Evaluation de l'erreur
3.11.4 Méthodes de relaxation et condition suffisante de convergence

Chapitre 4. Interpolation

4.1 Introduction
4.2 Matrice de Vandermonde
4.3 Interpolation de Lagrange
4.4 Polynôme de newton
4.5 Erreurs d'interpolation
4.6 Splines Cubiques :
4.6.1 Courbes de la forme $y = f(x)$

4.6.2 Splines paramétrées
4.7 Transformés de Fourier Discrètes.
4.8 Introduction au Nurbs :
4.8.1 B-splines
4.8.2 Génération de courbes

Chapitre 5. Différentiation et Intégration Numériques

1.1 Introduction
1.2 Différentiation Numérique
- 1.2.1 Différentiation d'ordre 1
- 1.2.2 Différentiation d'ordre supérieur

1.3 Intégration Numérique :
- 1.3.1 Formules de Newton -côtes simples et Composées
- 1.3.2 Méthode de Romberg
- 1.3.3 Quadratures de Gauss-Legendre
- 1.3.4 Intégration en utilisant les Splines
- 1.3.5 Applications

Chapitre 6. Calcul numérique des valeurs et vecteurs propres :

1.1 Introduction
1.2 Localisation des valeurs propres : Théorème de Gershgorin
1.3 Méthode de la puissance
1.4 Méthode de la puissance Inverse
1.5 Méthode QRApplications

Chapitre 7. Equations différentielles

3.1 Introduction
3.2 Méthode d'Euler Explicite
3.3 Méthode de Taylor
3.4 Méthode de Runge-Kutta
- 3.4.1 Méthode de Runge-Kutta d'ordre 2
- 3.4.2 Méthode de Runge-Kutta d'ordre 4
- 3.4.3 Analyse de l'erreur

3.5 Méthodes à pas multiples

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références bibliographiques *(Livres et polycopiés, sites internet, etc)* :

Citer au moins 3 à 4 références classiques et importantes.

[1]Rappaz, J. et M. Picasso: *Introduction a l'analyse numerique*. Presses polytechniques et universitaires romandes, Lausanne, 1998.

[2] Reddy, J.N.: *An Introduction to the Finite Element Method*. McGraw-Hill, New York, deuxieme edition, 1993.

[3] Asher, U. M. et Petzold L. R.: Computer Methods for Ordinary Differential Equations and Differential-Agebraic Equations. SIAM, Philadelphia, USA,1998.

[4] A. Fortin Analyse Numérique pour ingénieurs. Presses Internationales Polytechniques, 4ieme édition, 2011.

[5] F.Filbet Analyse Numérique-Algorithme et Etude Mathématique. Sciences Sup. Dunod, 2013.

[6] M., Benhamadou, A. Jeribi, Analyse Numérique Matricielle –Méthodes et Algorithmes, exercices et Problèmes corrigés . Edition Ellipses.2020

Semestre : ***S6***

Unité d'enseignement : UED1

Matière : ***Introduction à la sécurité d'Informatique (ISEC)***

Crédits : 4

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

Ce module permet d'initier les étudiants aux concepts de base de la sécurité d'informatique. En particulier, ce module consiste à :

- Présenter les aspects fondamentaux de la sécurité informatique.
- Identifier les principaux enjeux de la sécurité informatique.
- Distinguer et d'expliquer les différents types d'attaques auxquels un réseau ou un système informatique peut faire face.
- Etudier les principes et les techniques de la cryptographie.
- Sécuriser un site web par SSL.
- Sécuriser la messagerie par PGP.
- Protéger un hôte ou un réseau par un firewall et IDS.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Contenu de la matière :

Chapitre 1 : Introduction

1 Définition:

2 Buts de la sécurité

Confidentialité
Intégrité
Disponibilité
Non répudiation

3 Domaines d'application de la sécurité

Sécurité physique
Sécurité d'exploitation
Sécurité logique
Sécurité applicative
Sécurité des télécommunications

4 Attaques

4.1 Types d'Attaques

1. Script kiddie
    1. 2. Hacktivist
    2. Insider
    3. Organized crime
    4. Competitors

5. Open-source intelligence
6. Advanced persistent threat (APT)

4.2 Les types de malware

1. Viruses
2. Worms
3. Logic Bombs
4. Backdoor
5. Trojans
6. Remote access Trojan (RAT)
7. Ransomware
8. Keylogger
9. Spyware
10. Bots and Botnets
11. Rootkit

4.3 Reconnaître les attaques courantes

1. Social Engineering
2. Shoulder Surfing
3. Tricking Users with Hoaxes
4. Tailgating and Mantraps
5. Dumpster Diving
6. Watering Hole Attacks
7. Attacks via Email and Phone (spam, phishing, spear phishing, and whaling)

5 Politique de sécurité

Chapitre 2 : Cryptographie classique (chiffrement)

1. Introduction
2. Chiffre de César
3. Chiffrement monoalphabetique
4. Chiffrement par permutation par bloc
5. Chiffrement de Vigénaire
6. Chiffrement de HILL
7. Chiffre affine

Chapitre 3 : cryptographie moderne

1. Introduction
2. Classes de chiffrement moderne
    a. Chiffrement symétrique
    b. chiffrement asymétrique
3. chiffrement symétrique (AES)
4. chiffrement asymétrique (à clé publique)RSA: (Rivest - Shamir – Adleman)

    4.1 Exponentiation modulaire

    4.2 Exponentiation modulaire rapide
5. Notion de clé de session

    5.1 Protocole d’échange de clé
6. Enveloppe digitale
7. Condensé de message (haché)
8. Signature numérique

Chapitre 4 : Gestion des clés

1. Introduction
2. Système basé sur CA (Certification Authority)
3. Les principales fonctions
4. Architecture de l'infrastructure de gestion des clés
5. Certificat numérique
6. Installation d'autorité de certification sur windows serveur

Chapitre 5 : Protocoles sécurisés

1. Protocole SSL (Secure Socket Layer)
   - 1.1 Les fonctions de SSL
   - 1.2 Fonctionnement de SSL
   - 1.3 Surveillez les alertes SSL
   - 1.4 configurer un Service HTTPS dans IIS
2. SSH (Secure Shell)
3. IPSec (IP Secure)
   - 3.1 Architecture
   - 3.2 SA : Security Association
   - 3.3 SPD : Security Policy Database

Chapitre 6 : Le Protocole PGP

1. Introduction
2. PGP: (Pretty Good Privacy)
3. PGP: Algorithme
4. PGP: Authentification
5. PGP: Confidentialité
6. PGP: Compression
7. PGP: chiffrer ses mails sur windows

Chapitre 7 : Pare-feux et IDS

1. Introduction
2. Les Firewall
   - 2.1 Caractéristiques des Firewall
   - 2.2 Les types de Firewall
   - a) Le filtrage statique
   - b) Le filtrage dynamique (filtrage de paquet avec état)
   - c) Le filtrage applicatif

   a) Le filtrage statique
   - a.1 Les règles de filtrage des ACL
   - a.2 Filtrage statique par iptables
   - a.3 Gestion des règles par iptables

   b) Le filtrage dynamique (filtrage de paquet avec état)

   c) Le filtrage applicatif
   - 2.3 Les limites d'un firewall
3. Système de détection d'intrusions (IDS)
   - 3.1 Fonctions d'un IDS
   - 3.2 Classification des IDS

3.3 Exemple d'IDS

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références bibliographiques *(Livres et polycopiés, sites internet, etc)* :

Citer au moins 3 à 4 références classiques et importantes.

1. Practical Malware Analysis - Andrew Honig and Michael Sikorski. Published: February 29, 2012 by No Starch Press
2. The Hacker Playbook 3: Practical Guide To Penetration Testing - Peter Kim. Published: May 1, 2018 by Secure Planet
3. Sécurité informatique Ethical Hacking -Apprendre l'attaque pour mieux se défendre », Editions ENI - Octobre 2009 ISBN: 978-2-7460-5105-8
4. ETHICAL HACKING AND PENETRATION TESTING GUIDE, CRC Press Taylor & Francis Group ISBN-2015 : 13: 978-1-4822-3162-5
5. Ethical Hacking & Countermeasures- Threats & Defense Mechanisms, EC-Council | Press 2010 ISBN- 13 978-1-4354-8361-3
6. Mastering Hacking (The Art of Information Gathering & Scanning) - Harsh Bothra. Published: August 28, 2019
7. Stéphane Natkin, les protocoles de sécurité de l'internet, Dunod 2002
8. Bruce Schneier, cryptographie appliquée, Thomson Publishing, 1995
9. Bruno Martin, Codage, cryptologie et applications, Presses polytechniques et universitaires romandes, 2004
10. Stallings Williams, sécurité des réseaux, applications et standards, Vuibert Informatique, 2002
11. Solange Ghernaouti, Sécurité informatique et réseaux, 4ème édition Dunod, 2013
12. Gildas Avoine, Pascal Junod, Philippe Deschslin, sécurité informatique: cours et exercices corrigés, 2ème édition Vuibert 2009

Semestre : ***S7***

Unité d'enseignement : UEF1

Matière : ***Concepts avancés de bases de données***

Crédits : 4

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Ce cours fournit aux étudiants des connaissances théoriques et des compétences pratiques sur des sujets avancés dans les systèmes de bases de données et le big data. L'objectif principal de ce cours est d'élargir la vision des étudiants et d'introduire des sujets avancés, notamment les extensions orientées objet, relationnel-objet, les bases de données déductives et les langages d'interrogation appropriés, les données semi-structurées et XML ainsi que les bases de données distribuées. Les sujets supplémentaires couverts dans ce cours aideront les étudiants à devenir plus compétents dans le choix des technologies appropriées pour les projets et élargiront leur base de connaissances afin qu'ils aient une meilleure compréhension du domaine.

A la fin de ce cours les étudiants devront avoir une solide connaissance en :

Volet 1 : Modélisation des bases de données

- Modélisation de données orientée Objet.
- Modélisation des bases de données relationnelles-Objet.
- Modélisation des données semi-structurées
- Modélisation des bases de données déductives et autres types de bases de données

Volet 2 : Mise en œuvre de bases de données

- Utilisation de SGBD classique dans un environnement distribué (concept de SGBD distribué, transaction distribuée, requête distribuée).

Volet 3 : Travail collaboratif sous forme d'ateliers

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

BDD : Modélisation et exploitation (S5) et BDD : Administration et Architecture (S6).

Contenu de la matière

Chapitre I : Introduction aux SGBD orientés Objet

1. Evolution des SGBD
2. Limite des SGBDR
3. Généralités - Historique
4. Concepts fondamentaux des modèles objets

4.1 Objet, Type et Classe
4.2 Association
4.3 Autres concepts de l'approche OO
Agrégation, composition, héritage, liaison dynamique

5. Modélisation Objet avec UML: Unified Modeling Language
   5.1 Diagrammes de classes
   5.2 Diagrammes d'Etat/Transition
6. Caractéristiques des SGBD orientés objet
   6.1 Manifeste du système de BD OO
   6.2 Approches de développement de SGBDOO
   6.3 Avantages et inconvénients des SGBDOO

Chapitre II : Le Relationnel Objet : le Langage SQL3

L'apport des modèles orientés objet

1. Modèle Relationnel-Objet-SQL3
   1.1 Types abstraits de données et tables objets
   1.2 Identification d'objet OID
   1.3 Méthodes
   1.4 Héritage
   1.5 Large Object-LOB
   1.6 Vues Objet
   1.7 Association en SQL3

Chapitre III : Données Semi-structurées et XML

1. Notions de base de XML
2. Bases de données et XML
3. XML: un langage de balisage extensible
4. XML Schéma: un langage de typage des données XML
5. XDM: un modèle de données XML
6. XQuery: un langage d'interrogation de données XML

Chapitre IV : Généralités sur les bases de données distribuées

Introduction

1. Les bases de données distribuées (réparties)
   1.1 Définitions
   1.2 Caractéristiques et avantages de la répartition
2. Le Système de Gestion de BD distribué : SGBDD
3. Construction d'une base de données distribuée : les techniques de répartition de données
   3.1 Fragmentation (horizontale, verticale, mixte)
4. Architecture et fonctions d'un SGBDD
5. Transparences dans un SGBDD : 12 règles de « DATE »
6. Classification des approches de conception d'une BD distribuée : systèmes multi bases et systèmes fédérés

Chapitre VI : Gestion de Requêtes distribuées

1. Requêtes distribuées
2. Optimisation de requêtes
   2.1 Optimisation globale
   2.2 Optimisation locale
3. Stratégies d'évaluation de requêtes
   3.1 Algorithme d'optimisation R* (System R)
   3.2 Stratégies de jointure

Chapitre VII : Gestion de Transactions distribuées

1. Gestion de transactions : Définitions
2. Gestion de la concurrence : sérialisation distribuée, protocoles de verrouillage et d'estampillage
3. Validation et reprise
   3.1 Validation à deux phases 2PC,
   3.2 Validation à trois phases 3PC

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

- C. Delobel et M. Adiba, Bases de données et systèmes relationnels, Dunod 1982.
- Georges Gardarin : Les BD systèmes et leurs langages Eyrolles 2003
- G. Gardarin et P. Valduriez : Bases de données relationnelles: analyse et comparaison des systèmes Eyrolles, 1985
- C. J. Date 6ième édition traduit par Frédéric Cuppens : Introduction aux bases de Données. Internationale Thomson Publishing, 1998.
- T. Connoly et CorolynBegg. Systèmes de bases de données : approche pratique de conception de l'implémentation et de l'administration. Eyrolles 2005
- Emmanuel Bruno. 2013Bases de données - XQuery pour interroger des données XML - Eléments du langage et mise en œuvre - Cours et exercices corrigés (Niveau B).

Semestre : ***S7***

Unité d'enseignement : UEF1

Matière : ***Gestion de projets (GP)***

Crédits : 4

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

Décrire ce que l'étudiant est censé acquérir pour maitriser le bon déroulement d'un projet et assurer les enjeux aussi bien temporels et financière

- Permettre à l'étudiant de comprendre l'enjeu majeur de la gestion de projet.
- Initier l'étudiant au processus d'organisation et de planification.
- Entrainer l'étudiant à l'application de processus, méthodes et outils de planification.
- Initier l'étudiant aux environnements de gestion projet.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Contenu de la matière :

1. **Introduction**
   - Définition de concepts de base.
   - Notions de projet et gestion de projet.

2. **Les modèles de gestion de projet**
   - Les modèles de basés sur les livrables.
   - Les modèles basés sur le risque.

3. **Les éléments de Gestion de projet**
   - Les enjeux de gestion de projet.
   - Les activités de gestion de projet.
   - La structure de gestion de projet.
   - Les risques et la gestion de projet.

4. **L'organisation des équipes de programmation**
   - L'organisation de base.
   - Les outils de support.

5. **Les éléments de la planification**
   - La productivité du programmeur.
   - Echéance et jalonnement d'un projet.

6. **Le processus de planification**
   - Découpage et coordination des activités.
   - Les outils de planification (ordonnancement des activités et affectation des ressources).
   - Les environnements de planification (ex : MSPROJECT).

7. **Estimation des charges, délais et coût**
   - Les options alternatives : méthodes.
   - La précision de la taille des programmes.
   - Modèle d'estimation algorithmique.

8. **Approche Agile**
   - Principes et méthodes Agiles.
   - Présentation : Méthode Scrum. Méthode XP.

9. **Techniques d'analyse d'options et de décisions.**
   - Graphe polaire.
   - Matrice des bénéfices

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références bibliographiques *(Livres et polycopiés, sites internet, etc)* :

Citer au moins 3 à 4 références classiques et importantes.

- Principles of software engineering management by Tom GILB Edition Lavoisier.
- Software Engineering: A Practitioner's Approach by Roger S Pressman.
- Software Project Management in Practice by Pankaj Jalote.
- Génie logiciel: principes, méthodes et techniques by Alfred Strohmeier et Didier Buchs.

Semestre : ***S7***

Unité d'enseignement : UEF1

Matière : ***Web avancé et Micro-services***

Crédits : 3

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement : Apprendre à créer des applications Web avancées, Se familiariser avec l'architecture orienté-micro services et Apprendre à créer des clients mobiles hybrides

Connaissances préalables recommandées. *Développement de sites Web statiques et dynamiques, modèle MVC*

Contenu de la matière

Chapitre I : Introduction au Web Avancé

- Architecture d'une application Web en entreprise
- Rappel du modèle logiciel MVC.
- Les différentes couches d'une application monolithique traditionnelle (couche Métier, couche ORM, Services Web et applications Web tiers, clients lourd, client léger, client mobile, etc.).
- Limites des applications Web monolithiques

Chapitre II : Framework côté Client

- Limites du cycle de développement traditionnel côté client (JavaScript) sans Framework.
- Utilité des Framework
- Quelques exemples de Framework JavaScript (Angular, React, ...)

Chapitre III : Introduction aux Micro-Services

- Services Web et leur fonctionnement
- Principe et avantages des micro-services
- Passage d'une application Web monolithique à une architecture micro-services.

Chapitre IV: Mise en oeuvre d'une architecture micro-services

- Architecture micro-services avec Flask/Python
- Conception des micro-services avec Flask
- Déploiement des micro-services Flask
- Déploiement avec Docker

Chapitre V: Sécurisation des micro-services

- Bases de la sécurité Web (HTTP, socket, session, code status).
- Limites de l'authentification basée sur les sessions et les cookies, et types d'attaques.
- Introduction à JWT (Json Web Token), structure de JWT (header, payload, signature).

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références :

1. Moises Macero, Learn Microservices with Spring Boot, Editions Apress, 2017.
2. Christian Gammelgaard. Microservices in .Net, Editions Manning Publications Co. 2021.
3. Sam Newman. Monolith to Microservices : Evolutionary Patterns to transform your monolith. Editions OREILLY. 2019.
4. Julien Ponge. Architecture Microservices. Editions ENI. 2017.

Semestre : ***S7***

Unité d'enseignement : UEF2

Matière : ***Data-Mining***

Crédits : 5

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Ce cours permettra aux étudiants de comprendre les concepts clés du data-mining et d'apprendre à préparer et explorer les données pour leur bonne exploitation.
Les étudiants apprendront aussi les différentes techniques de data-mining telles que l'apprentissage supervisé (régression, classification) et non supervisé (clustering et recherche des associations).
Les étudiants apprendront également à évaluer les modèles et à choisir le modèle le plus approprié.
Les travaux pratiques permettront aux étudiants d'appliquer les techniques de data-mining à des problèmes réels dans divers domaines tels que la santé, la finance, le marketing, etc.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Entrepôts de données, Bases de données, Mathématiques (Analyse, Probabilités et statistiques).

Contenu de la matière

Chapitre 1 : Données, informations et connaissance

- Définitions générales
- Les types de connaissance
- Les enjeux de la connaissance

Chapitre 2 : Introduction au data mining

- Définitions et concepts clés
- Domaines d'application du data mining
- Les différentes étapes du processus de data mining

Chapitre 3 : Préparation des données

- Collecte de données
- Nettoyage des données
- Sélection des données.
- Transformation, normalisation des données
- Détection d'outliers
- Le fléau de la dimension
- Les Principales techniques de la réduction
- Etude de benchmarks de data mining

Chapitre 4 : Tâches et techniques du data mining

- Apprentissage non supervisé
  - Extraction des règles d'association (A priori, Eclat, FPgrowth, Charm Mafia, GenMax,)
  - Clustering
    - Par partionnement : K-Means, PAM
    - Hiérarchique : AGNES, DIANA
    - Par densité : DBSCAN
    - Les mesures de performances du clustering
- Apprentissage supervisé :
  - Méthodes de régression (uni et multivariée) et descente du gradient
  - Classification:
    - Régression logistique
    - Les plus proches voisins : K_NN
    - La méthode Naive Bayes
    - Les arbres de décision : C4.5, CART, CHAID

- **Évaluation des classificateurs**
  - Mesures de performance
  - Validation croisée
  - Sur-apprentissage, régularisation
  - Interprétation des résultats

Chapitre 5 : Technologies avancées du datamining

- SVM (Machines à vecteurs de support)
- Random Forest
- Réseaux de neurones
  - Perceptron
  - MLP (Multi layer Perceptron) et Rétropropagation
- Deep learning
  - RNN (Recurrent Neural Network) ET LSTM
  - CNN (Convolutional Neural network)
  - Transformers

Chapitre 6: Data mining distribué

- Intérêt de la distribution
- Distribution des données
- Approches hybrides (Traitement distribué, multi-agent)

Chapitre 7 : Text mining

- Analyse sémantique latente (LSA)
- Topic modeling et LDA (Latent Dirichlet Allocation)

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Bibliographie

1) Han, J., & Kamber, M. (2006). Data mining: Concepts and techniques (2nd ed.). Morgan Kaufmann Publishers.

2) Han, J., Pei, J., & Kamber, M. (2011). Data mining: Concepts and techniques (3rd ed.). Morgan Kaufmann Publishers.

3) Grossman, R. L., & Horn, W. (2004). A survey of data mining software tools. SIGKDD Explorations, 6(1), 20-33.

4) Witten, I. H., & Frank, E. (2005). Data mining: Practical machine learning tools and techniques (2nd ed.). Morgan Kaufmann Publishers.

5) Tan, P. N., Steinbach, M., & Kumar, V. (2005). Introduction to data mining (1st ed.). Addison-Wesley.

6) Aggarwal, C. C. (2015). Data mining: The textbook (1st ed.). Springer.

7) Daniel T.Larose, Chantal D. Larose, Data mining Découverte de connaissances dans les données, 2ième Ed., Vuibet. 2018.

8) Avrim Blum, John Hopcroft, and Ravindran Kannan, Foundations of Data Science, 2018.

9) UCI machine learning repository https://archive.ics.uci.edu/ml/index.php

.

Semestre : ***S7***

Unité d'enseignement : UEF2

Matière : ***Compilation 2***

Crédits : 5

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement (*Décrire ce que l'étudiant est censé avoir acquis comme compétences après le succès à cette matière – maximum 3 lignes).*

Les objectifs de ce cours sont multiples :

- Maîtrise des langages de programmation : En effet, il est primordial de comprendre le fonctionnement des compilateurs afin de programmer efficacement.

- Conception et description de langages dédiés à des applications spécifiques. Le processus de compilation est souvent associé et restreint aux langages de programmation (procéduraux, orientés objets) alors qu'il peut être généralisé à une multitude de langages tels que :
  - Les langages logiques (Prolog adapté aux stratégies d'essais et d'erreurs, aux problèmes de planification,…)
  - Les langages fonctionnels (Lisp, Ocaml pour les applications de l'Intelligence Artificielle,…)
  - les langages de description de textes,
  - les langages de commandes de robots,
  - les langages à balise (HTML, XML,…)
  - les langages formels
  - les langages de description et de conception de programme,
  - langage pour les protocoles (ASN1 : description des formats des messages)
  - les langages de test pour les protocoles (TTCN)
  - les langages de description des architectures (Rapid, Darwin,…)
  - les langages pour les techniques de preuves,
  - les langages de conception du matériel (VHDL pour le matériel informatique)
  - les langages de modélisation graphique ou textuelle des systèmes et d'objets (tel que UML),
  - les langages de modélisation de la réalité virtuelle
  - les langages Postscript
  - les langages naturels,…

- Ecriture des compilateurs pour des processeurs spécifiques comme ceux présents dans les systèmes embarqués tels que les téléphones portables, les assistants personnels, les outils de positionnement (GPS), les caméscopes numériques, les systèmes de radiologie numériques,…

Consolidation des connaissances acquises à travers différents modules à savoir les théories des Langages, l'algorithmique, Architecture des ordinateurs, la théorie des graphes, l'Assembleur, les systèmes d'exploitation, la logique mathématique,…

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Compilation 1, Théorie des langages, Architecture des ordinateurs, Système d'exploitation, Algorithmique

Contenu de la matière

Introduction générale

Chapitre I : Rappel sur les différentes phases d'un compilateur

Chapitre II - Environnement d'exécution

1- Allocation-Substitution
2- Organisation des données à l'exécution
   2-1 Allocation statique
   2-2 Allocation dynamique

Chapitre III- Optimisation de code

1- Introduction
2- Blocs de base et graphe de flot de contrôle
3- Principales optimisation**s**
   3-1 Sous-expressions communes
   3-2 Propagation des copies
   3-3 Elimination de code inutile
   3-4 Optimisation des boucles
4- Analyse globale des flot de données

Chapitre VI : Génération du code objet

1- Description de la machine virtuelle
2- Description de la machine cible
3- Langage assembleur & langage machine
4- Exemple de programme en assembleur
   4-1 L'instruction conditionnelle
   4-2 Les instructions itératives
   4-3 Les expressions arithmétiques
   4-4 Les données
      4-4-1 les données statiques
      4-4-2 les données dynamiques
5- Les procédures simples
6- Allocation des registres par coloriage de graphe

Chapitre V - Modèles alternatifs de programmation

16- Compilateurs des langages fonctionnels
17- Compilateurs des langages orientés objets
18- Compilateurs des langages scripts
19- Les parseurs dédiés

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

- *Compilateurs : principes, techniques et outils* - A. Aho, R. Sethi, J. Ullman
- *Compilateurs* - D. Grune, H. Bal, C. Jacobs, K. Langendoen - Dunod.
- *The Theory and Practice of Compiler Writing*- Sorensen
- *Modern Compiler Implementation In ML* - Andrew.W.Appel
- *Programming Language Pragmatics*, Michael Scott

Semestre : ***S7***

Unité d'enseignement : UEM1

Matière : ***Méthodes de management agiles***

Crédits : 3

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Ce cours donne des concepts avancés de la gestion de projet notamment les méthodes agiles.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Notions de gestion de projet

Contenu de la matière

1. Introduction et rappels de gestion de projets, les cycles traditionnels et WBS.
2. Définition de concepts de base.
3. Principes et méthodes Agiles dans la gestion de projet.
4. Responsabilité et Avantages d'un développement agile.
5. La méthode Agile SCRUM : Vue d'ensemble de SCRUM.

   5.1 L'approche itérative et incrémentale
   5.2 Le Cycle de Développement adopté
   5.3 Framework de SCRUM
   5.4 La théorie de SCRUM
   5.5 Les Artefacts SCRUM

6. Méthode XP.
7. Techniques d'analyse d'options et de décisions.

   7.1 Graphe polaire.
   7.2 Table des bénéfices.

8. Les modèles basés sur le risque.

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

Les méthodes agiles et la qualité Cours ENSG MOCCAND Guillaume
Les nouveaux tableaux de bord des managers Le projet Business Intelligence clés en main Alain Fernandez 6ème édition Eyrolles
Le chef de projet efficace 12 bonnes pratiques pour un management humain. Alain Fernandez Eyrolles 6ème édition
Guide PMBOK + Guide pratique Agile. Project Management Institute. *Edition PMI* Édition 2018

Semestre : ***S7***

Unité d'enseignement : UEM1

Matière : ***Réseaux et Protocoles***

Crédits : 3

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Suite à l'étude du module réseaux et protocoles, l'étudiant sera capable de comprendre, de configurer et d'utiliser au moins un protocole dans chaque couche de la pile protocolaire. Il sera capable d'analyser le fonctionnement des protocoles et d'en déduire les anomalies dues à des erreurs de configuration ou au matériel.

Connaissances préalables recommandées.

Initiation aux Réseaux Informatiques (RES1)

Contenu de la matière

Chapitre 1 : Architectures Tcp/Ip (2 séances)

- Rappel (protocoles Tcp/Ip)
- Adressage IP/MAC (Protocole ARP)
- Sous-adressage (statique/dynamique)

Chapitre 2 : Protocoles de routage (3 séances)

- Routage statique (rappel)
- Routage dynamique (RIP, OSPF, BGP)

Chapitre 3 : Réseaux locaux et réseaux étendues (3 séances)

- Réseaux locaux Virtuels (Vlan), Protocoles VTP, DTP
- Routage inter-Vlan
- Redondances dans les LANs (STP, EtherChannel, Protocole HSRP)
- Protocole PPP
- VPN (Protocole GRE)

Chapitre 4 : Protocoles applicatifs (3 séances)

- Protocoles web (HTTP et HTML)
- Protocoles de messagerie électronique (SMTP, POP et IMAP)
- Services de partage de fichiers (FTP et SMB)
- DHCP et DNS
- Telnet et SSH

Chapitre 5 : Protocole IPv6

- Adressage
- Transition IPv4/IPv6
- Services IPv6…

Chapitre 6 : Gestion des réseaux

- CDP, NTP
- SNMP
- Qualité de Service (QoS)
- Automatisation de gestion de la configuration (Ansible…)

Chapitre 7 : Ingénierie des Protocoles

- Introduction
- Spécification d'un protocole
- Cycle de Développement
- Vérification et Validation
- Modélisation
- Des protocoles pour des traitements répartis ouverts
- Conclusion

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références :

- Cours cisco
- Andrew S. Tanenbum, David J. Wetherall, 'Computer Networks', 5ème edition Pearson
- G.J. Holzmann. Design and validation of computer protocols. Prentice-Hall, 1991.
- J. Holzmann. The SPIN Model-Checker. Addison-Wesley, 2004.
- R.Lai, A. Jirachiefpattana. Communication protocol specification and verification.
- The Springer International Series in Engineering and Computer Science, 1998.
- A.Tanenbaum, N. Feamster and D. Witherall. Computer Networks. Edition 6th.
- Published by Pearson, 2020.
- FLEURY Éric, KAMOUN Farouk. CFIP'2006 : ingénierie des protocoles : mobilité, sans
- fil, et qualité de services (Actes du 12° colloque francophone sur l'ingénierie des
- protocoles).

Semestre : ***S7***
Unité d'enseignement : UED1

Matière : IHM : ***Conception et évaluation des interfaces***

Crédits : 3

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Confectionner des interfaces graphiques visuelles en respectant les critères ergonomiques et les standards du design des interfaces interactives et conviviales. L'étudiant pourra acquérir les connaissances suivantes :

- Connaissances des règles ergonomiques
- Connaissance d'une méthode de développent d'IHM
- Couplage avec la méthode de développement par objets
- Mise en œuvre de ces méthodes dans un projet

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Génie logiciel ; Programmation Orientée Objet

Contenu de la matière

Chapitre I : Notions fondamentales

1. Définitions : Interaction, Interactivité, IHM
2. Historique et évolution des IHMs
3. Conception d'IHM
   - Principes de base de la conception d'IHM
   - Tendances actuelles en matière de conception d'IHM
   - Défis et opportunités pour les IHMs dans le futur
4. Types d'interactions et de dispositifs IHM
5. Évaluation des IHMs : tests utilisateurs et mesures d'utilisabilité
6. Considérations sur l'accessibilité des IHMs

Chapitre II : Méthodologie de construction d'une IHM

1. Méthodologie Classique.
2. Etape d'identification : identification des domaines fonctionnels, Définition du modèle de l'utilisateur (notion de profil de l'utilisateur), Définition du modèle des taches (types des tâches) et environnement technique.
3. Etape d'analyse des taches (notion de séquence actions-objectifs).
4. Etape de modélisation (nécessité de choisir un modèle et une architecture).
5. Etape de spécification (cahier de charges)

- Etude des besoins pour l'IHM
- Spécification conceptuelle
- Spécification fonctionnelle
- Spécification syntaxique/ Spécification lexicale

Chapitre III : Modèles & architectures

1. Le Contrôleur de dialogue (définition & rôle).
2. Présentation du modèle Seeheim
3. Présentation du modèle PAC
4. Présentation du modèle MVC
5. Présentation des modèles à agents.

Chapitre IV : Règles ergonomiques dans les IHMs

1. Concepts de bases
2. Utilisabilité des systèmes interactifs
3. Heuristiques de Nielsen.
4. Critères ergonomiques de Bastien et Scapin
    a. Guidage
    b. Charge de travail
    c. Contrôle explicite
    d. Adaptabilité
    e. Gestion des erreurs
    f. Homogénéité / cohérence
    g. Signification des codes et dénominations
    h. Compatibilité
5. Règles d'or de Coutaz

Chapitre V : Conception d'interfaces multi utilisateurs

1. Etude comparative entre IHM mono utilisateur et multi utilisateur.
2. La méthode CCU (conception centrée sur l'utilisateur).
3. Exemples d'interfaces multi utilisateurs.

Chapitre VI : Interfaces adaptatives

1. Le Modèle de Vaudry.
2. Etude d'un exemple : Modèle à agents.

Chapitre VII : Les interfaces multimodales et les interfaces futures

1. Techniques d'interactions avancées, Réalité Augmentée, Interface Tangible, projection 3D, Analyse du mouvement)
2. Eléments de Programmation Visuelle.

Mode d'évaluation : (type d'évaluation et pondération)

Examen de TP (40%), Examen (60%)

Références

B. Shneiderman (1987). Designing the user Interface: Strategies for effective human computers, Edition Wesley.

J. Coutaz (1990). Interface homme-ordinateur, conception et réalisation. Dunod informatique.

J-B,Cramps(1998). Interfaces graphiques ergonomiques, conception et modélisation.

C.Kolski (2001). Analyse et conception de l'IHM : interaction homme-machine pour les SI Volume1, Traité IC2, série Informatique et Systèmes d'Information.

S-A Serengul (2006). The FastTrack to Human-Computer Interaction, (Paperback) Thomson Learning.

D. Benyon (2013). Designing Interactive Systems: A Comprehensive Guide to HCI, UX and Interaction Design, Pearson; 3 edition.

Kolski, C, Ezzedine, H et Abed, M (2001). Développement du logiciel : des cycles classiques aux cycles enrichis sous l'angle des IHM, ouvrage collectif, Analyse et conception de l'IHM, Interaction homme-machine pour les systèmes d'information Vol 1, Hermès, 2001, 250 p, ISBN 27462-0239-5, p. 23-49.

Yvonne Rogers, Helen Sharp et Jenny Preece (2011). Interaction Design: beyond human computer interaction (3rd edition)

Semestre : ***S8***

Unité d'enseignement : UEF1

Matière : ***Architecture et gestion des Systèmes d'information avancés***

Crédits : 4

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Les systèmes d'information modernes ou avancés ont su tirer profit des évolutions tant sur le plan des technologies que celui des modes d'organisation et de management. Tout système d'information est perceptible sous quatre dimensions que sont l'informationnel, l'organisationnel, le technologique et le managérial. Parmi les objectifs de cette première partie du module :

- Expliquer le principe de l'évolution des SI selon les 04 dimensions ;
- Comprendre le passage d'une culture verticale vers celle horizontale ;
- Différencier entre la vision orientée fonction et celle orientée processus d'entreprise ;
- Découvrir les enjeux de la cartographie des processus pour l'amont (stratégique) et l'aval (opérationnel) d'une organisation.
- Apprendre à manipuler les formalismes et les outils de modélisation d'entreprises et de processus ;
- Avoir une idée sur l'architecture des systèmes de gestion de workflow ou BPMS ;
- … etc.

De plus, les systèmes d'information ont connu des avancées technologiques qui font que le marché en la matière offre divers choix de solutions pour l'automatisation allant de la simple suite logicielle passant par les progiciels de gestion intégrés (ERP) jusqu'aux architectures urbanisées offrant une vision plus large et plus flexible pour la gestion des SI généralement complexes. Dans cette partie, nous verrons entre autres :

- Découvrir des concepts avancés comme l'agilité des SI ;
- Distinguer entre les visions intégrées et fragmentées dans les solutions d'automatisation de SI ;
- Comment sont apparus les premières formes d'ERP (ou MRP) ;
- Les ERP(s) modernes et leurs diverses classifications ;
- Les ERP(s) Open-Source : leurs forces et leurs faiblesses ;
- Comment configurer et exploiter un ERP ;
- Découvrir ce qu'est l'urbanisation en SI, ses fondements et principes ;
- Comprendre le projet d'urbanisation et sa gestion ;
- Manipuler quelques outils ;

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Systèmes d'information, Bases de données, Génie logiciel

Contenu de la matière

Chapitre I : Les Quatre Dimensions d'un SI

1. Notions et Rappels sur les SI
2. Dimensions de Reix
    a. Informationnelle
    b. Organisationnelle
    c. Technologique
    d. Managériale

Chapitre II : Le Business Process Management (BPM)

1. Modalités de la coopération/collaboration
2. Changement de culture
    a. Orienté donné vers orienté information
    b. Orienté fonction vers orienté processus
3. Evolution vers le BPM
    a. Visions W3C vs WFMC
    b. Autres consortiums
4. Concepts de base et classifications
    a. Processus
    b. Taxonomie des processus
5. Cycle de vie du BPM
    a. Modèles de cycle de vie
    b. CPI (ContinuousProcessImprovement)
    c. BPR (Business ProcessReingineering)
6. Modélisation des processus métier
    a. Objectifs de la modélisation graphique
    b. Quelques formalismes graphiques
    c. Le langage BPMN
7. Concepts et enjeux de la cartographie
    a. Cartographie du niveau stratégique
    b. Cartographie du niveau opérationnel
8. Le projet BPM (démarche, méthodologies et outils)

Chapitre III : La Technologie Workflow

1. Quelques définitions de base (processus, activité, acteur, rôle, contrôle de flux, instance, …)
2. Les modèles de référence et organismes de standardisation
    a. Modèle aux cinq interfaces
    b. Exemples d'outils conformes (Staffware, ProcessMaker)
3. Interopérabilité des systèmes de gestion de workflow
4. Architecture de workflow Inter-organisationnel
    a. Modèle en « sous-traitance »
    b. Modèle « faiblement couplé »
    c. Modèle « chaîné »
5. Workflow à base de services
    a. Eléments de SOA et services Web
    b. Le langage BPEL
6. Système de gestion de WF (WFMS)
    a. Architecture d'un WFMS

b. Quelques outils : Bonita, Oracle BPM et Yawl/Woped

Chapitre IV : Agilité des SI

1. Concepts et niveaux d'agilité
2. Architecture d'entreprise agile
3. Gouvernance d'un SI agile
4. Alignement stratégique du SI comme outil

Chapitre V : Évolution technologique des SI

1. Vision intégrée vs vision fédérée des SIQ (SAI)
2. L'EAI
3. L'ERP
   a. Architecture des premiers ERP : MRP
   b. ERP classique
   c. ERP vertical vs horizontal
   d. Les ERP et la vision processus : ERP vs BPMS
      Combinaison ERP et BPM
   e. Typologie et classification des ERP sur le marché
      Présentation de quelques ERP (Open-Concerto, Odoo, SAP, …)
   f. Le projet de mise en place d'un ERP dans une entreprise
      (Objectifs, coûts, couverture fonctionnelle, cahier des charges, conduite du changement, …)

Chapitre VI : Urbanisation des SI Complexes

1. Complexité des SI : un résultat de l'évolution d'entreprise
   a. Notion de valeur ajoutée : chaîne de Porter
2. Urbanisation : principes et fondements
   a. Bases et origines
   b. Terminologie
   c. Plan d'occupation du sol (POS)
3. Concepts d'une architecture urbanisée
   a. Processus
   b. Fonctions
   c. Applications
   d. Infrastructures
4. Le projet d'urbanisation : Équipes, processus et organisation

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

1. Robert Reix, Robert Fallery, Michel Kalika, Frantz Rowe (2016). "Systèmes d'information et ***management***". 7ème edition. Editeur : Vuibert.
2. Will Van Der Aalst, Kees Max Van Hee (2004). "***Workflow Management: Models, Methods, and Systems***". The MIT Press.
3. Will Van Der Aalst, (2016). "***Process Mining. Data Science in Action***". Editeur : Springer.

4. SetragKhoshafian, Marek Buckiewics (1998). "***Groupware et Workflow***". Editions Eyrolles.
5. SetragKhoshafian (2006). "***Service Oriented Enterprise***". Editions Taylor & Francis.
6. Jean-Louis Tomas, Yossi Gal (2011). "***ERP et conduite des changements : Alignement, sélection et déploiement***". Edieur : DUNOD.
7. Ellen Monk, Bret Wagner (2012). "***Concepts in Enterprise Resource Planning***". 4th Edition. Editor : Cengage Learning.
8. Jacques Sassoon (1998). "***Urbanisation des systems d'information***". Editor : Hermes Science Publication.
9. Philippe Eynaud, Daniel Alban, Julien Malaurent, Jean-Loup Richet, Claudio Vitari (2019). "***Information Systems Management : Governance, Urbanisation and Alignement***". Editeur : Wiley-ISTE.
10. CRIPP : Cadre de Référence International des Pratiques Professionnelles de l'audit interne. Etablit par l'IIA (Institute of InternalAuditors).

Semestre : ***S8***

Unité d'enseignement : UEF1

Matière : ***Big-data et Bases de données NoSQL***

Crédits : 4

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Les plateformes Big Data sont des systèmes distribués qui peuvent traiter de grandes quantités de données sur des clusters de serveurs. Elles sont utilisées dans tous les secteurs d'activité. Dans ce cours les étudiants vont se familiariser avec l'utilisation des plateformes Big Data actuelles et auront un aperçu des architectures Big Data basées sur le cloud telle que Hadoop, Spark et d'autres plateformes de Big Data basées sur SQL, telles que Hive. Les étudiants vont aussi Ce cours porte aussi sur les bases de données NoSQL. Les bases de données NoSQL ne tentent pas d'imposer un modèle relationnel. En fait, quatre modèles de données différents se sont distingués dans l'écosystème NoSQL : orientés clé-valeur, orientés document, orientés colonnes et orientés graphe. Pour chaque modèle NoSQL, son architecture est décrite et positionnée via le théorème CAP.
Après avoir suivi ce cours, les étudiants seront en mesure de :

- Distinguer les différents types de bases de données NoSQL.
- Comprendre l'impact du cluster sur la conception des bases de données.
- Enoncer le théorème CAP et en expliquer les points principaux.
- Expliquer comment HBase, MongoDB, Cassandra, Neo4j et Redis s'intègrent au théorème CAP.
- Travailler avec le système de fichiers distribués Hadoop (HDFS) comme base des technologies NoSQL.
- Exploiter les données HDFS avec Apache Spark-SQL et Apache Pig.
- Décrire la conception de HBase, MongoDB, Cassandra, Neo4j et Redis.
- Utiliser les langages de contrôle, de définition et de manipulation des données des bases de données NoSQL couvertes par le cours.
- Travailler en collaboration sous forme d'ateliers.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Modélisation et exploitation de BDD ; Administration et Architecture des BDD,

Concepts avancés des bases de données.

Contenu de la matière

Chapitre I : Big data et vue d'ensemble de BD NoSQL

1. Historique
2. BIG DATA
   2.1 Définition
   2.2 Caractéristiques du Big data
   2.3 Big data : Cas d'utilisation

3. BD NoSQL
   3.1 Revue du modèle relationnel
   3.2 Propriétés de l'ACID
   3.3 Le théorème CAP
   3.4 Taxonomie des bd NoSQL
      3.4.1 BD NoSQL clés-valeurs
      3.4.2 BD orientées colonnes
      3.4.3 BD orientées graphe
      3.4.4 BD orientée Document

Chapitre II : BD orientée clé-valeur et SGBD Redis

1. Le modèle de données clé-valeur
2. Redis en tant que cache
3. Commandes et Pipelining
4. Mécanismes de durabilité/persistance
5. Partitionnement avec Redis Cluster
6. Messagerie Publish/Subscribe
7. Notifications d'espace de clé
8. Suppression automatique à l'expiration de la clé
9. Chargement de données en masse
10. Transactions

Chapitre III : BD orientée document et SGBD MongoDB

1. Le modèle de données orienté documents
2. Documents et collections
3. Cas d'utilisation de MongoDB
4. Conception de MongoDB
5. MongoDB et le théorème CAP
6. Le langage de manipulation des données de MongoDB
7. Transactions, atomicité et documents
8. Durabilité et journalisation
9. Traitement par lots et agrégation
10. Indexation
4. MongoDB en tant que système de fichiers

Chapitre IV : BD orientée colonne et SGBD Cassandra

1. Le modèle de données orienté colonnes
2. Bases de données et tables
3. Colonnes, types et clés
4. Le langage de manipulation des données
5. L'architecture de Cassandra
6. Espaces clés, réplication et familles de colonnes
7. Le théorème CAP
8. Gestion des nœuds de cluster

Chapitre V : BD orientée graphe et SGBD Neo4j

1. Aperçu de la théorie des graphes
2. Le modèle de données orienté graphe
3. Cas d'utilisation des bases de données orientées graphe
4. Conception de Neo4j : Standalone et Cluster
5. Propriétés ACID et théorème CAP
6. Gestion des transactions avec JTA
7. Opérations CRUD avec l'API Neo4j Core
8. Naviguer dans les graphes avec l'API Traversal
9. L'API REST de Neo4j
10. Le langage de manipulation de données Cypher
11. Recherche dans les graphes

Chapitre VI : Ecosystème Hadoop

1. Introduction à Hadoop
2. HDFS (Hadoop Di03.2stributed File System)
3. MapReduce
4. YARN (YetAnother Resource Negotiator)
5. Hadoop ecosystem components
6. Sécurité et haute disponibilité avec Hadoop

Chapitre VII : Spark

1. Introduction à Spark (Les concepts de base de Spark & architecture)
2. Analyse de données avec Spark
3. Machine Learning avec Spark
4. Déploiement de Spark
5. Optimisation de performances avec Spark

Chapitre VIII : Autres types de bases de données

1. BD spatiale
2. BD à base ontologique
3. Streaming database

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

Aridhi, Sabeur, Philippe Lacomme, and Raksmey Phan. "Les bases de données NoSQL et le Big Data." (2014).
Sébastien Ferrandez. "MongoDB - Comprendre et optimiser l'exploitation de vos données (avec exercices et corrigés)". 2019.
Juvénal CHOKOGOUE. Hadoop - Devenez opérationnel dans le monde du Big Data. 2017.
Pirmin Lemberger et al. Big Data et Machine Learning - 2e éd. - Les concepts et les outils de la data science: Les concepts et les outils de la data science. 2016.
Sylvain Roussy. Neo4j Des données et des graphes – Prise en main (2e édition). 2016
Nishant Neeraj. Mastering Apache Cassandra - Second Edition Broché – 26 mars 2015
Juvénal CHOKOGOUE. Big Data & Streaming : Traitement streaming et temps réel des données en Big Data. 2019.
Sylvie Servigne.Fondements des bases de données spatiales. 2006
Stéphane Jean. Manipulation d'ontologies au sein de bases de données: OntoQL, un langage d'exploitation de bases de données à base ontologique Broché – 22 novembre 2011.

Semestre : ***S8***

Unité d'enseignement : UEF2

Matière : ***Architectures logicielles***

Crédits : 4

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

- ✓ Maîtriser l'ensemble des compétences indispensables au métier de développeur et d'architecte logiciel.
- ✓ Apprendre à utiliser les techniques de conception, de modélisation et d'implémentation selon l'approche composants, services et architectures logicielles.
- ✓ Apprendre à réfléchir au développement en termes d'architectures logicielles.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Génie logiciel, programmation orientée objet

Contenu de la matière

1. **Introduction et généralités**
   - Introduction.
   - Les architectures distribuées et les architectures centralisées.
   - Les architectures n-tiers.
   - Les modèles de structuration des systèmes et applications.
   - Présentation et comparaison des types d'architectures : multi-niveaux, clients/serveurs, extensibles, dynamiques,…etc.

2. **Les composants et les modèles de composants logiciels**
   - Introduction et définitions.
   - Object vs Composant.
   - Organisation logique d'un composant.
   - L'aspect structurel et comportemental.
   - L'approche par composant : principe et concepts de base.
   - Avantages et inconvénients.
   - Cycle de vie.
   - Présentation et études de quelques modèles de composants (EJB3, UML, .Net…etc.).

3. **Les architectures logicielles**
   - Introduction.
   - Principe et objectif.
   - Architecture vs. Conception.

- Les modèles d'architectures.

4. **Les langages de description d'architecture logicielle (ADLs)**
   - Introduction.
   - Les concepts de base des langages de description d'architecture (Les composants, les connecteurs et les configurations).
   - Description des principaux ADLs.
   - Classification des ADLs.
   - L'aspect statique dans les ADLs.
   - L'aspect dynamique dans les ADLs.
   - Vérification et validation.

5. **Conception architecturale**
   - Les bases de la conception architecturale.
   - Les approches de conception architecturale.
   - Développer un modèle architectural.
   - Exemple d'application.

6. **Les styles architecturaux**

7. **Les architectures orientées Services (SOA)**
   - Introduction.
   - Les concepts de base (service, annuaire de services, …etc).
   - L'architecture orientée services.
   - Les objectifs de l'architecture orientée services.
   - Les principes de l'architecture l'orienté service.
   - L'approche orientée services.
   - Technologies basées sur les principes de SOA.
   - Les services web : définitions et caractéristiques.
   - Les applications des services web.
   - Le fonctionnement des services web.
   - Architecture des services web.
   - Protocoles et normes.
   - Les architectures micro services.

8. **Le déploiement**
   - Introduction.
   - Déploiement sur des infrastructures centralisées.
   - Déploiement sur des infrastructures distribuées.
   - Déploiement sur le Cloud.
   - Autres solutions et technologies pour le déploiement.

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

- M. OUSSALAH. Ingénierie des composants: Concepts, techniques et outils. Éditeur Vuibert (20 juin 2005). ISBN-10 : 2711748367.
- Projet ACCORD (Assemblage de composants par contrats en environnement ouvert et réparti). Juin 2002.
- N. Medvidovic and R. N. Taylor, A Classification and Comparison Framework for Software Architecture Description Languages, IEEE Transactions on Software Engineering, Vol 26, no1, pp70-93, Janvier 2000.
- Jean-Michel Doudoux. Ensemble des cours de Jean-Michel Doudoux sur la technologie Java (Java SE, Java EE, Java ME) et Eclipse (https://www.jmdoudoux.fr/java/dej/titre.htm). Version 2.30 du 15/06/2022
- A. TASSO. Le livre de Java premier langage- Avec 109 exercices corrigés. 592 pages. ISBN-13 : 978-2212138542. ISBN-10 : 2212138547. Éditeur : Eyrolles; 9e édition (31 octobre 2013). Langue: Français.
- J. LONCHAMP. Conception d'applications en Java/JEE - 2e édition. Principes, patterns et architectures. Collection : InfoSup, Dunod. Parution : janvier 2019.
- J. ARLAT. Composants logiciels et sûreté de fonctionnement - intégration de COTS. Juin 2000.
- J. Fontanel, F. Lacomme, L. Ren. Les web services - Concevoir et utiliser des applications 2.0 C#, Java, PHP, API JavaScript, Android SDK, iOS SDK. Éditeur: Ellipses. Avril 2013.

Semestre : ***S8***

Unité d'enseignement : UEF2

Matière : ***Modèles et Gestion de Procédés Logiciels***

Crédits : 4

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Le module est conçu pour répondre aux besoins présents de la communauté tout en étant prospectif. Il se veut cependant simple et accessible pour les étudiants. Le module vise principalement la formation des étudiants pour :

- Comprendre les concepts de base des procédés de développement de logiciels.
- Maitriser leur gestion en terme de modélisation, orchestration et exécution.
- Maîtriser l'aspect automatisation relatif aux procédés de développement de logiciels.
- Connaître les langages et/ou outils les plus courants en la matière.
- Maitriser le volet collaboration et ses défis dans les procédés de développement de logiciels.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Module Génie logiciel du tronc commun

Contenu de la matière

- **Chapitre I : Introduction aux procédés de développement de logiciels** Définitions et objectifs des procédés de développement de logiciels
  Intérêts pour les projets de développements
  Les différents types et granularité de procédés de développement de logiciels

- **Chapitre II : La modélisation des procédés de développement**
  Aspect IDM pour les procédés de développement
  Types de modélisation : from-scratch, composants de procédés, patrons de procédés, framework de procédés (paramétrisation), ...
  Modélisation de procédés par et pour la réutilisation
  Les langages de modélisation de procédés

- **Chapitre III : L'exécution et l'orchestration des procédés de développement**
  Introduction et définitions
  Les avantages de l'exécution automatique de procédés de développement
  L'orchestration de procédés de développement et ses objectifs
  Les langages pour l'automatisation dans les procédés de développement

- **Chapitre IV : La collaboration dans les procédés de développement**

  Besoins et objectifs de la collaboration
  Fusion de procédés et défis (optionnel)
  Tâches collaboratives et principes de collaboration dans un projet de développement Les outils de collaboration

- **Chapitre V : Les tendances actuelles et futures des procédés de développement**

  L'optimisation des procédés de développement (optionnel)
  Les méthodes agiles et leur application dans les procédés de développement (optionnel)
  La qualité dans les procédés de développement (optionnel)
  L'intégration continue
  Les Procédés logiciels pour L'intelligence artificielle

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

Quaresma, José Augusto De Sena, and Sandro Ronaldo Bezerra Oliveira. "Teaching and learning strategies for software process subject." 2021 IEEE Frontiers in Education Conference (FIE). IEEE, 2021.

A. Hachemi et al. "Software process patterns: a roadmap." in AICCSA. IEEE. 2017.

Al-Saqqa S. et al. "Agile software development: Methodologies and trends." International Journal of Interactive Mobile Technologies 14.11 (2020).

KIM, Gene, HUMBLE, Jez, DEBOIS, Patrick, et al. The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations. IT Revolution, 2016.

Agh, Halimeh, Félix García, and Mario Piattini. "A checklist for the evaluation of software process line approaches." Information and Software Technology 146 (2022): 106864.

Roger, S. Pressman, and R. Maxin Bruce. Software engineering: a practitioner's approach. McGraw-Hill Education, 2015.

A. Hachemi et al. "POEML: a Process Orchestration, Execution, and Modeling Language." Journal of Software: Evolution and Process 34.6 (2022): e2456.

García-García, Julián A., et al. "Software process simulation modeling: systematic literature review." Computer Standards & Interfaces 70 (2020): 103425.

Van Der Aalst, Wil. Process mining: data science in action. Vol. 2. Heidelberg: Springer, 2016.

Xiaolong, He, et al. "Soft computing and decision support system for software process improvement: A systematic literature review." Scientific Programming 2021 (2021): 1-14.

Semestre : ***S8***

Unité d'enseignement : UEF2

Matière : ***Test Logiciel et Assurance Qualité***

Crédits : 4

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Le test a pour but de détecter la présence d'erreurs dans un programme vis-à-vis de sa spécification. Il cible la formalisation de critères pour guider la sélection des tests. Et éventuellement sa mise à l'épreuve : de la robustesse, des performances et des propriétés de sûreté de fonctionnement.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Notions fondamentales de développement et de spécification de logiciels

Contenu de la matière

1. Introduction et problématique

- Vérification statique (revue et inspection).
- Exécution symbolique (valeurs – variables).
- Techniques de test
- Notions et critères de l'assurance qualité logiciel.

2. Test dynamique (structurel et fonctionnel)

- Structurel et Critères de test structurel
- Fonctionnel

3. Test statique
4. Test Mutationnel (évaluer jeu de test) (Une Séance 1h 30 m).
5. La qualité des logiciels : Facteurs et évaluation de la qualité d'un logiciel.
6. Les modèles de qualité : SEI, CMMI,
7. Les méthodologies de développement et qualité.
8. Le processus et plan d'assurance qualité.
9. Les risques et la qualité
10. Automatisation des tests (Mise en œuvre : test unitaire et d'intégration) (Objet TP)

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

- Software engineering Addison-Wesely 6th ed.2001 I.Sommerville.
- Le genie logiciel et ses applications I.Sommerville Intereditions .
- Software Testing Techniques B. Bezier Van Nostrand Reinhold.
- Structured Testing T.J.McCabe.
- Test logiciel en pratique Vuibert informatique John Warkins 2002.
- http://www.faqs.org/faqs/software-eng/testing-faq/.
- Assurance qualité en conception by, Lavoisier.
- Du système d'assurance qualité à la certification ISO 9002 by FLEUR
- Software engineering Institute, SEI.org

Semestre : ***S8***
Unité d'enseignement : UEM1

Matière : ***Systèmes d'exploitation mobiles***

Crédits : 3

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Le cours permet à l'étudiant de comprendre le fonctionnement des systèmes d'exploitation mobiles en particulier Android tout en fournissant les concepts clés en matière de développement de systèmes et d'applications dans un environnement mobile.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Systèmes d'exploitation, Programmation orientée objet

Contenu de la matière

Chapitre 1 : Systèmes d'exploitation mobiles

1.1 Définition
1.2 Caractéristiques
1.3 Architecture générale d'un système d'exploitation mobile
1.4 Contraintes des système d'exploitation mobiles
1.5 Dispositifs mobiles
1.6 Catégories de Systèmes d'exploitation mobiles
  1.6.1 Systèmes propriétaires
  1.6.2 Systèmes libres (open source)
1.7 Exemples de systèmes d'exploitation mobiles
  1.7.1 Apple iOS
  1.7.2 Windows Phone
  1.7.3 Android
  1.7.4 Autres : Symbian OS, Blackberry OS, Harmony OS

Chapitre 2 : Système Apple iOS

2.1 Définition et caractéristiques
2.2 Historique et versions
2.3 Architecture d'iOS
  2.3.1 Couche Cocoa Touch
  2.3.2 Couche Media
  2.3.3 Couche Core services
  2.3.4 Couche Core OS
2.4 Gestion multi-tâches

Chapitre 3 : Système Windows Phone

3.1 Définition et caractéristiques
3.2 Historique et versions

3.3 Architecture de Windows Phone
3.3.1 Applications
3.3.2 Frameworks
3.3.3 CLR: Common Language RunTime
3.3.4 Bibliothèques
3.3.5 Noyau

Chapitre 4 : Système Android

4.1 Définition et caractéristiques
4.2 Historique et versions
4.3 Architecture d'Android
4.3.1 Couche application
4.3.2 Application framework
4.3.3 Bibliothèques
4.3.3.1 Native librairies
4.3.3.2 External native librairies
4.3.4 Android Runtime : Dalvik /ART
4.3.5 HAL : Hardware Abstraction Layer
4.3.6 Noyau linux
4.3.6.1 Gestion de processus et threads
4.3.6.2 Gestion de la mémoire
4.3.6.3 Gestion des périphériques
4.3.6.4 Gestion de fichiers
4.3.6.5 Gestion de l'énergie
4.3.6.6 Gestion du réseau
4.3.6.7 Binder (IPC)
4.3.6.8 Sécurité

Chapitre 5 : Use case : Développement mobile sous Android

5.1 Environnement de développement
5.1.1 Android SDK
5.1.2 SDK Manager
5.1.3 Android Virtual Device (AVD) Manager
5.1.4 Les outils SDK
5.1.5 IDE et Plugins
5.1.6 Langages de programmation
5.2 Structure d'un project Android
5.3 Construction d'application Android
5.4 Le fichier AndroidManifest
5.5 Composants d'une application Android : point de vue utilisateur
5.6 Composants d'une application Android : point de vue développeur
5.6.1 Composants applicatifs : Activité, services, fournisseur de contenu
5.6.2 Composants d'interaction : Intents, récepteur d'Intents, Notifications.
5.6.4 Permissions

Chapitre 6 : Applications Android

6.1 Les activités
6.1.1 Définition
6.1.2 Etats d'activités

6.1.3 Cycle de vie d'une activité
6.1.4 Activités et vues
6.2 Les ressources
6.2.1 Définition
6.2.2 Types de ressources
6.2.3 La classe R
6.2.4 Utilisation des ressources
6.3 Les Services
6.3.1 Types de services (local, distant)
6.3.2 Exemples des services
6.3.3 Cycle de vie d'un service
6.4 Les intentions (intents)
6.4.1 Définition
6.4.2 Attributs d'intent
6.4.3 Types d'intents
6.4.4 Résolution d'intents
6.4.5 Filtres d'intents
6.5 Les récepteurs d'intents (BroadcastReceivers)
6.5.1 Créations de récepteurs d'intents (statique et dynamique)
6.5.2 Diffusion d'intents
6.5.3 Réception et traitement d'intents
6.6 Les fournisseurs de contenu (Content Providers)
6.6.1 Content Provider et Content Resolver
6.6.2 Fournisseurs de contenu natifs
6.6.3 Modèles de données
6.6.4 Opérations sur les données
6.7 Les notifications
6.7.1 Mécanismes "toasts"
6.7.2 Gestionnaire des notifications
6.7.3 Sonnerie et vibreurs
6.7.4 Les alarmes
6.8 Les permissions
6.8.1 Définitions
6.8.2 Permissions sur les composants
6.8.3 Permissions sur les activités
6.8.4 Permissions sur les services
6.8.5 Permissions sur les récepteurs d'intents
6.8.6 Permissions sur les fournisseurs de contenu

Chapitre 7 : Gestion du matériel

7.1 Les ressources disponibles sur un appareil Android
7.2 Gérer le réseau Wi-Fi
7.2.1 Accéder à un réseau Wi-Fi
7.2.2 Activer et désactiver le Wi-Fi
7.2.3 Gérer les points d'accès
7.3 Gérer le Bluetooth
7.3.1 Les permissions
7.3.2 Préparer votre application
7.3.3 Rechercher d'autres appareils Bluetooth

7.3.4 Communication entre appareils Bluetooth

7.4 Les capteurs

7.4.1 Identification des capteurs

7.4.2 Utilisation des capteurs

7.4.3 Réception des données

Chapitre 8 : Processus, Threads et taches asynchrones

8.1 Processus et Threads Android

8.2 Types de processus

8.3 Priorités des processus

8.4 Mécanisme de threading d'Android

8.5 Communication inter threads et la classe Handler

8.6 Taches asynchrones (AsyncTask)

8.6.1 Définition

8.6.2 Types génériques d'AsyncTask

8.6.3 Cycle de vie d'AsyncTask

8.6.4 Utilisation d'AsyncTask

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu/Examen de TP (40%), Examen (60%)

Références

- Jonathan Levin. Android Internals: A Confectioner's Cookbook. 2014 -
- Karim Yaghmour. Embedded Android: Porting, Extending, and Customizing. 2013 - Earlence Fernandes. Instant Android Systems Development How-to. 2013
- Joshua J. Drake, Zach Lanier, Collin Mulliner, Pau Oliva Fora, Stephen A. Ridley, Georg Wicherski. Android Hacker's Handbook. 2014
- Damien Guignard, Julien Chable, Emmanuel Robles. Programmation Android : De la conception au déploiement avec le SDK Google Android 2. Edition Ayrolles. 2011
- Pierre Nerzic. Programmation mobile avec Android. Support de cours, 2023.

Semestre : ***S8***

Unité d'enseignement : UEM1

Matière : ***Modélisation et Evaluation des performances de systèmes (MEPS)***

Crédits : 3

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Les objectifs de ce module sont multiples. Le premier objectif serait de familiariser les étudiants aux principes de la modélisation et de l'évaluation des performances des systèmes de manière générale et plus particulièrement des systèmes informatiques logiciels et matériels.
Un autre objectif consiste à sensibiliser l'étudiant au fait que le développement de tout système informatique ne doit aboutir à un système sous-dimensionné tout en évitant le surdimensionnement. Pour cela, développer un système adapté, en respectant le plus possible les objectifs du cahier des charges, est une démarche qui passera obligatoirement, au cours de la phase de conception, par une étape de modélisation en choisissant le formalisme le plus adéquat, suivie d'une étape d'analyse et d'évaluation des performances.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Probabilités et Statistiques

Contenu de la matière

Introduction générale

Chapitre I : Principes de la modélisation et de l'évaluation des performances

Chapitre II : Processus stochastiques

Chapitre III : Chaînes de Markov
III.1 : Chaînes de Markov à temps discret
III.2 : Chaînes de Markov à temps continu

Chapitre IV : 3. Files d'attente
IV.1 : Files d'attente mono-serveur
IV.2 : Files d'attente multi-serveurs
IV.3 : Files d'attente à capacité limitée

Chapitre V : Les réseaux de Petri
Chapitre VI : Les réseaux de Petri stochastiques

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

- Randal Douc, Eric Moulines, Pierre Priouret, Philippe Soulier, Markov Chains, Springer, 2018.
- Annie Choquet-Geniet, Les réseaux de Petri - Un outil de modélisation : Cours et exercices corrigés - Licence, Master, écoles d'ingénieur, Editeur Dunod, 2006.
- B. Baynat, Théorie des files d'attente, Hermes 2000.

Semestre : 8

Unité d'enseignement : UET1

Matière : Projet Pluridisciplinaire

Crédits : 4

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs d'enseignement :

Ce cours se déroule durant le semestre 8 et consiste en la conception et la réalisation d'un projet en informatique qui se déroule dans les mêmes conditions que celles qui existent dans une entreprise. Le projet doit être décrit à travers un cahier des charges précis et peut porter sur des thèmes très variés. Il doit couvrir au moins deux disciplines. Il est proposé et encadré par un ou plusieurs enseignants. Le but de cette matière étant l'immersion des étudiants dans le milieu socio-économique il est possible de réaliser le projet dans le cadre d'un stage de 15 jours dans une entreprise économique ou un établissement professionnel pendant les vacances du printemps.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Matières enseignées dans les semestres antérieurs allant du S1 au S8.

Contenu de la matière

Le groupe de projet est composé de 2 étudiants au minimum et de 4 au maximum, selon la taille du projet et sous la responsabilité d'un chef de projet nommé parmi les membres du groupe. Le groupe doit se comporter et opérer comme une véritable équipe de développement.
Outre le contenu technique, qui consistera en l'application des connaissances acquises pour la mise en œuvre du cycle de développement d'un petit logiciel, l'accent sera mis sur l'acquisition et l'application des aspects organisationnels et relationnels entre les membres du groupe et leur encadreur ainsi que l'entreprise d'accueil en cas de stage externe.
Les tâches à réaliser dans le projet doivent couvrir les points suivants :

- Analyse et découpage du travail,
- Répartition des charges de travail entre les membres du groupe par le chef de projet / l'encadreur.
- Circulation de l'information entre les membres du groupe,
- Mise en place d'un planning de travail,
- Exposés périodiques de l'avancement du projet,
- Délivrance des livrables fixés dans la fiche de projet,
- Rédaction d'un rapport de stage (entre 20 et 30 pages).
- Exposé du travail réalisé devant une commission d'examen.

Mode d'évaluation : Soutenance devant une commission d'examen du projet (100%)

Références :

Semestre : ***S9***
Unité d'enseignement : UEF1

Matière : ***Méthodes formelles pour GL***

Crédits : 4

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Les compétences développées dans le cadre de ce cours sont :

- Connaître les langages de description formelle des systèmes logiciels,
- Appliquer les techniques de modélisation aux systèmes critiques,
- Connaître les langages de spécification formelle des propriétés et des exigences,
- Exprimer les propriétés à satisfaire par les systèmes logiciels,
- Apprendre à appliquer les techniques de vérification formelle.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Génie Logiciel, Logique Mathématique, Probabilités & Statistiques

Contenu de la matière

1) **Introduction aux méthodes formelles**
   - Processus de modélisation et de vérification formelle
   - Langages formels de description des systèmes
   - Langages formels de spécification des systèmes
   - Techniques formelles de vérification

2) **Modèles formels à base d'automates**
   - Systèmes de transitions
   - Automates communicants et leurs types
   - Sémantiques

3) **Algèbres de processus**
   - Syntaxe
   - Sémantique Opérationnelle structurée

4) **Réseaux de Petri**
   - Syntaxe
   - Propriétés des graphes de marquages
   - Variantes et Extensions

5) **Logiques temporelles & Technique de Model-Checking**
   - Logique Temporelle à temps linéaire
   - Logique Temporelle à temps arborescent
   - Algorithmes de model-checking

6) **Spécification Comportementale**
   - Relations d'équivalence comportementale
   - Relations basées sur les modèles des traces, failures et raidies
   - Relations de Raffinement

7) **Descriptions Formelles des Données**
   - Types Abstrait de Données : Syntaxe & Sémantique axiomatique
   - Langage Z.

8) **Modèles probabilistes**
   - Chaînes de Markov
   - Evaluation des performances.
   - Model-Checking Probabiliste.

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

Markus Roggenbach, Antonio Cerone, Bernd-Holger Schlingloff, Gerardo Schneider, Siraj Ahmed Shaikh. Formal Methods for Software Engineering Languages, Methods, Application Domains. Springer 2021.

C. A. R. Hoare, Communicating Sequential Processes, Prentice-Hall, Inc., USA, 1985.

Collectif sous la direction de Michel Diaz. Vérification et mise en œuvre des réseaux de Petri. Editions Hermes-Science, 2003.

Wolfgang Reisig. Understanding Petri Nets: Modeling Techniques, Analysis Methods, Case Studies Springer; 2013th edition.

R. Milner (1989): Communication and Concurrency, Prentice Hall

M. Hennessy (1988): Algebraic Theory of Processes, MIT Press J.C.M.

Baeten & W.P. Weijland (1990): Process Algebra, Cambridge University Press

Bill Roscoe (1998, amended 2005): The theory and practice of concurrency, Prentice Hall W.J.

Fokkink (2000): Introduction to Process Algebra, Texts in Theoretical Computer Science, An EATCS Series, Springer.

Peter Ryan, Steve Schneider, Michael Goldsmith, Gavin Lowe and Bill Roscoe, Modelling and Analysis of Security Protocols, Addison Wesley; 01 edition (6 Dec. 2000).

Semestre : ***S9***
Unité d'enseignement : UEF1

Matière : ***Développement de logiciels embarqués***

Crédits : 4

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

- Maîtriser le langage et la terminologie propres au domaine des systèmes embarqués.
- Apprendre les concepts de base des systèmes embarqués : microcontrôleurs et leur interfaçage avec les périphériques ou autres machines pour des tâches d'acquisitions de données, contrôle et monitoring de composants.
- Comprendre le problème d'interfaçage à tous les niveaux : architecture, logique, timing, chargement et protocole.
- Maitriser les logiciels et plateformes pour concevoir et construire des systèmes embarqués.
- Apprendre le cycle de vie et les étapes du développement d'un système embarqué.
- Apprendre la modélisation d'un système embarqué et l'architecture d'une plate-forme pour systèmes embarqués.
- Apprendre la conception et la programmation de la partie logicielle d'un système embarqué.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Programmation & génie logiciel
Architectures des ordinateurs & systèmes d'exploitation.

Contenu de la matière

1. **Introduction aux systèmes embarqués**
   - Définitions & Historique des systèmes embarqués
   - Domaines d'applications
   - Caractéristiques et attributs des systèmes embarqués
   - Différents modèles de la famille ARM
   - Architecture ARM et Concepts de base de Cœurs des systèmes embarqués (ARM), pipelining, interruptions, vecteurs d'interruption, registres statuts…
   - Capteurs et actionneurs
   - Contrôle et transfert de données
   - Interfaces de communication et Interfaçage avec des modules GSM, GPS pour traitement de données et affichage.
2. **Systèmes embarqués sous Linux et systèmes embarqués sous C**

- Aspects systèmes: gestion/allocation mémoire, ordonnancement, timers, process, Threads, Multi-Threading, Sémaphores
- Logiciels embarqués ;
- Exemples d'applications temps-réel
- Contraintes temporelles
- Différentes architectures de systèmes embarqués temps-réels
- Noyaux pour systèmes embarqués temps réel
- Ordonnancement et tolérance aux fautes : ordonnancement monoprocesseurs, ordonnancement multiprocesseurs, tolérance aux fautes.
- Eléments d'architecture matérielle et logicielle : Noyau, Bus et réseaux.

3. Outils de développement des logiciels

- Environnement de programmation : C-POSIX, environnement de compilation.
- Simulateurs ; émulateurs
- moniteur ROM,
- interface JTag,
- programmation et tests directs des ROM/flash

4. Programmation multitâche et les mécanismes de synchronisation entre tâches

- Utilisation d'un exécutif temps réel :
- Décomposition fonctionnelle des systèmes embarqués ;
- Architecture logicielle d'un système de pilotage ;
- Implantation synchrone et asynchrone : rôle de l'exécutif ;
- Structure générale d'un exécutif ;
- Primitives de synchronisation, de gestion des événements et du temps ;
- Exemples d'applications Temps Réel.

5. Programmation Smartphone

- Spécificité de la programmation sur Android
- composant d'une interface graphique
- persistance des données
- programmation concurrente sur Android
- mise en place d'une communication réseau

6. Méthodes de conception des systèmes embarqués

- Modèles de co-design
- Modélisation système (SysML) :
  - Analyse des besoins et spécification du système
  - Modélisation de l'architecture du système
  - Modélisation du comportement
  - Modélisation d'aspects transverses : spécification d'équations et contraintes (Parametric Diagram), allocation du comportement sur l'architecture et gérer la traçabilité des exigences.
- Utilisation des modèles :

- Modélisation et analyse des propriétés temps-réel d'un système embarqué avec le profil normalisé MARTE (Modeling and Analysis of Real Time Embedded systems).
- Simulation des systèmes par exécution des modèles.
- Techniques avancées de l'IDM : Définition d'un langage de modélisation spécialisé (DSML) et définition d'un générateur de code.

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

Emmanuel Grolleau, Jérôme Hugues, Yassine Ouhammou, & Henri Bauer ; Introduction aux systèmes embarqués temps réel - Fondamentaux et études de cas: Conception et mise en œuvre Broché –.Ed. Dunod, 2018.

Modélisation et analyse de systèmes embarqués. Auteurs : KORDON Fabrice, HUGUES Jérôme, CANALS Agusti, DOHET Alain. Editions Lavoisier. 2013

Joseph Yiu, The Definitive Guide to ARM® Cortex®-M3 and Cortex®-M4 Processors, Newnes editor, 2013. ISBN-13 978- 0124080829.

Les systèmes d'exploitation temps réel - J.P. Elloy Y. Trinquet, Techniques de l'Ingénieur, 27 septembre 2010.

Computer organization and design: the hardware/software interface, David A. Patterson, John L. Hennessy, Morgan Kaufmann, 4th edition, 2008.

Computer Architecture: A Quantitative Approach, David A. Patterson, John L. Hennessy, Morgan Kaufmann, 4th edition, 2009.

Architecture des machines et des systèmes informatiques, Alain Cazez, Joëlle Delacroix, 3ème édition, Dunod, 2008.

L'art du développement Android. Mark Murphy. Éditeur : Pearson, 2010

Programmation Android. de la conception au déploiement avec le SDK Google Android.Damien Guignard, Julien Chable, Emmanuel Robles. Editions Eyrolles, 2010.

Sanford Friedenthal, Alan Moore, Rick Steiner, A Practical Guide to SysML, The Systems Modeling Language, MK/OMG Press, 2009, (ISBN 978-0-12-378607-4)

Conception orientée modèle de logiciels embarqués. Application à la communication dans le cadre d'une flotte de drones Auteurs : Jean-Aimé Maxa, Mohamed Slim Ben Mahmoud, Nicolas Larrieu, ISTE Group. 2018.

Semestre : ***S9***

Unité d'enseignement : UEF2

Matière : ***Conception de jeux vidéo : Théorie et Pratique***

Crédits : 4

Coefficient : 3

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

L'objectif de ce cours et de fournir à l'étudiant les principes de base de design et de conception d'un jeu vidéo allant du 2D au 3D. Plusieurs plateformes seront présentées et étudiés en pratique pour offrir un choix de développement.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Algorithmique, Programmation

Contenu de la matière

Chapitre 1. Conception et développement de jeux : Principes de base

1.1 Concepteur de jeux : Définition et tâches
1.2 Notion de jouabilité de jeu
1.3 Besoins du concepteur de jeux
1.4 Exigences des joueurs
1.5 Outils de conception
1.6 Document de conception (GDD)

Chapitre 2. Gestion des Lutins (Sprites) et tuiles (Tiles)

2.1 Génération et visualisation de sprites
2.2 Génération et visualisation de de tuiles
2.3 Atlas de tuiles (spritesheet)
2.4 Défilement de scènes de tuiles (Scrolling tilemaps)
2.5 Tuiles isométriques
2.6 Exemple d'un jeux 2D

Chapitre 3. Détection de collision

3.1 Physique des objets rigides
3.2 Le moteur physique
3.3 Calcul de la collision
3.4 Moteur physique box2dweb

Chapitre 4. Webgl et son intégration dans les jeux 3D

4. 1 Le WEBGL
4. 2 Three.js
4.3 Editeur Three.js
4.4 Intégration de WebGL dans les jeux 3D
4.5 Exemple de jeux 3D basé sur Three.js

Chapitre 5. Applications : Le moteur de jeux Unity

5.1 Principes de base de Unity
5.2 Navigation dans les scènes et physique
5.3 Importation de ressources statiques
5.4 Création de scripts avec C#
5.5 Animation
5.6 Armement et effets spéciaux.

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen pratique (60%)

Références

- Game design : theory and practice; Richard Rousse, 2005
- Beginning game level design, John Feil & Marc scattergood, 2005
- Game Programming, All in One, Jonathan s. Harbour, 2004
- Android game programming for dummies, Derek James, John Wiley & Son Inc. 2013
- Unity for Absolute Beginners, Sue Blackman, 2014, Apress, www.it-ebooks.info

Semestre : ***S9***
Unité d'enseignement : UEF2

Matière : ***Internet of Things (IoT) : Concepts et développement***

Crédits : 5

Coefficient : 4

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

L'objectif de ce cours est de transmettre à l'étudiant les concepts fondamentaux et technologies nécessaires pour comprendre, concevoir et développer une solution IoT (Internet des objets). A travers ce cours l'étudiant va apprendre à concevoir une solution IoT, fabriquer et programmer des objets IoT et l'intégrer dans un système IoT.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Systèmes d'exploitation, Réseaux, Programmation web, Python

Contenu de la matière

Chapitre 1 : Introduction à l'Internet des objets

- 1.1 Définitions
  - 1.1.1 Internet des objets (IoT)
  - 1.1.2 Objets connectés
- 1.2 Caractéristiques d'IoT
- 1.3 IoT vs M2M
- 1.4 Domaines d'application d'IoT
- 1.5 IoT enabling technologies
  - 1.5.1 Réseaux de capteurs sans fil (WSN)
  - 1.5.2 Cloud computing
  - 1.5.3 Big data analytics
  - 1.5.4 Protocoles de communication
  - 1.5.5 Systèmes embarqués
    - 1.5.5.1 Définition et caractéristiques
    - 1.5.5.2 Modèles d'interaction avec ou sans contrainte de temps
    - 1.5.5.3 Approches de conception
    - 1.5.5.4 Co-conception (Codesign)
- 1.6 Challenges

Chapitre 2 : Architecture d'IoT et protocoles

- 2.1 Architecture d'IoT
  - 2.1.1 Architecture Three and Five layers
  - 2.1.2 Architecture basée Cloud and Fog
- 2.2 Couche physique d'IoT
  - 2.2.1 Objets IoT (IoT devices)
  - 2.2.2 Architecture typique d'objet IoT
- 2.3 Protocoles de la couche liaison de données

2.3.1 IEEE 802.3 - Ethernet
2.3.2 IEEE 802.11 - WiFi
2.3.3 IEEE 802.15.1 - Bluetooth
2.3.4 IEEE 802.15.4 - LR-WPAN
2.3.5 IEEE 802.16 - WiMax
2.3.6 2G/3G/4G/5G - Communication mobile
2.3.7 LoRaWAN
2.3.8 Technologies de communication sans contact : RFID, NFC

2.4 Protocoles de la couche Réseaux/Internet
2.4.1 IPv4
2.4.2 IPv6
2.4.3 6LoWPAN

2.5 Protocoles de la couche Transport
2.4.1 TCP
2.4.2 UDP

2.6 Protocoles de la couche application
2.6.1 HTTP
2.6.2 CoAP
2.6.3 WebSocket
2.6.4 MQTT
2.6.5 XMPP
2.6.6 DDS
2.6.7 AMQP

2.7 Conception logique d'IoT
2.7.1 Blocs fonctionnels d'IoT
2.7.2 Modèles de communication d'IoT
2.7.3 APIs de communication d'IoT

Chapitre 3 : Conception et développement d'un système IoT

3.1 Composants d'un système IoT
3.1.1 Objets IoT
3.1.2 Ressources
3.1.3 Contrôleur de service
3.1.4 Base de données
3.1.5 Web service
3.1.6 Composant d'analyse
3.1.7 Applications IoT

3.2 Méthodologie de conception d'un système IoT
3.2.1 Spécification des objectifs et exigences
3.2.2 Spécification du modèle de processus (cas d'utilisation)
3.2.3 Spécification du modèle de domaine (concepts, objets,...)
3.2.4 Spécification du modèle d'information
3.2.5 Spécification de services
3.2.6 Spécification des composants du système IoT
3.2.7 Spécification de la vue fonctionnelle
3.2.8 Spécification de la vue opérationnelle
3.2.9 Intégration des composants et objets
3.2.10 Développement d'application IoT

3.3 Etude de cas d'un système IoT

Chapitre 4 : Développement d'une solutions IoT

4.1 IoT and Python
- 4.1.1 Motivation
- 4.1.2 Packages Python pour IoT

4.2 Objets IoT (IoT physical devices)
- 4.2.1 Exemples d'objets IoT
- 4.2.2 Programmation d'objets avec Python

4.3 Serveurs IoT et cloud
- 4.3.1 Modèles de stockages cloud et APIs de communication
- 4.3.2 Web Application Messaging protocol (WAMP)
- 4.3.3 Xively cloud pour IoT
- 4.3.4 Développement d'applications avec le Framework Django (Python web application framework)
- 4.3.5 Développement des services web REST

4.5 Cas d'étude illustratifs

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

- ArshdeepBahga and Vijay Madisetti. Internet of Things, A Hands on Approach, 2015.
- Pallavi Sethi1 and Smruti R. Sarangi. Internet of Things: Architectures, Protocols, and Applications. Journal of Electrical and Computer Engineering. 2017.
- P.P. Ray. A survey on Internet of Things architectures. King Saud University Journal of King Saud University –Computer and Information Sciences. 2016.
- Sridipta Misra Muthucumaru and Maheswaran Salman Hashmi. Security Challenges and Approaches in Internet of Things. SpringerBriefs in Electrical and Computer Engineering. 2017.
- Colin Dow . Internet of Things Programming Projects: Build modern IoT solutions with the Raspberry Pi 3 and Python. 2018.
- Olivier Hersent , David Boswarthick and Omar Elloumi. L'internet des Objets : Les principaux protocoles M2M et leur évolution vers IP. Edition DUNOD. 2014.

Semestre : ***S9***
Unité d'enseignement : UEM1

Matière : ***DevOps et Cloud Computing***

Crédits : 5

Coefficient : 4

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

- acquérir les fondamentaux liés au concept de la virtualisation et à son application dans le domaine du Cloud computing, à la gestion des infrastructures et au déploiement d'applications scalables.
- Identifier les principes fondamentaux de DevOps et sa fonction dans le développement de logiciels contemporains.
- Découvrir les technologies et les processus DevOps, y compris le contrôle de version, l'infrastructure en tant que code (IaC), l'intégration continue/livraison continue (CI/CD) et la
- conteneurisation.
- Découvrir comment effectuer des tâches d'analyse, de test, de déploiement et de surveillance du code.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Contenu de la matière

Chapitre 1 : Virtualisation (3 Séances)

1- Principes de la virtualisation
2- Objectifs et bénéfices
3- Historique
4- Définitions, concepts et techniques de virtualisation
   - les différentes architectures de la virtualisation de système d'exploitation
   - Virtualisation dans différents domaines
5- Performances et rendement

Chapitre 2 : Environnement Cloud Computing (3 Séances)

1- Rappel sur les systèmes distribués : Concepts et architectures, Concepts d'objet, composant, service, Architecture SOA
2- Définitions liées à l'environnement Cloud Computing
3- Caractéristiques
4- Modèles de déploiement
5- Modèles de services (SAAS, PAAS,IAAS)
6- Architectures du cloud
7- Entreprises face au cloud
8- Fournisseurs /clients de services
9- Environnement multi-cloud
10- Virtualisation et cloud

Chapitre 3 : Applications de Cloud computing (3 Séances)

1- Sécurité
2- Gestion
3- Stockage
4- Allocation de ressources, etc
1- Outils de mise en œuvre : Cloudsim, Openstack

Chapitre 4 : Introduction à DevOps (1 Séance)

- Qu'est-ce que DevOps ?
- Principes et pratiques DevOps.
- Avantages du DevOps : rapidité, qualité, collaboration
- DevOps vs développement de logiciels traditionnel

Chapitre 5 : Contrôle de version avec Git (2 Séances)

- Introduction à Git et aux systèmes de contrôle de versions distribués
- Commandes Git de base : cloner, ajouter, valider, pousser, tirer, créer une branche
- Workflows de collaboration avec Git : stratégies de branchement, pull request
- Utilisation des plateformes d'hébergement Git (par exemple, GitHub, GitLab)

Chapitre 6 : Intégration continue et livraison continue (CI/CD) (2 Séances)

- Comprendre les principes et les avantages du CI/CD
- Créer des pipelines CI/CD avec des outils comme Jenkins, GitLab CI/CD
- Tests automatisés et contrôles de qualité du code
- Stratégies de déploiement : bleu-vert, canari, mises à jour progressives

Chapitre 7 : Infrastructure as Code (IaC) (2 Séances)

- Gérer l'infrastructure avec du code : avantages et outils
- Introduction aux outils IaC : Terraform, Ansible, AWS CloudFormation
- Définir et provisionner les ressources de l'infrastructure
- Gestion et orchestration des configurations

Chapitre 8 : Conteneurisation avec Docker (4 Séances)

- Introduction à la conteneurisation et à Docker
- Créer et exécuter des conteneurs Docker
- Images Docker, registres et référentiels
- Orchestration de conteneurs avec Kubernetes (Concepts de base)

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

1. K. C. Gouda, AnuragPatro, Dines Dwivedi, NagarajBhat, Virtualization Approaches in Cloud Computing, in International Journal of Computer Trends and Technology (IJCTT), vol 12 Issues 4, (Juin 2014).
2. Tonglin Hawk, Ioan Raicu, and Lavanya Ramakrishnan. Scalable state management for scientific applications in the cloud. IEEE International Congress on Big Data, 2014.
3. Ali Belgacem, Kadda Beghdad Bey, Hassina Nacer, Sofiane Bouznad: Efficient dynamic resource allocation method for cloud computing environment. Clust. Comput. 23(4): 2871-2889 (2020)

4. Kadda Beghdad Bey, Sofiane Bouznad, Farid Benhammadi, Hassina Nacer: Improved Virus Optimization algorithm for two-objective tasks scheduling in Cloud Environment. FedCSIS (Communication Papers) 2019: 109-117
5. Linda Ouchaou, Hassina Nacer, Chahrazed Labba. Towards a distributed SaaS management system in a multi-cloud environment**.** Clust. Comput. 25(6): 4051-4071 (2022)
6. Guillaume Plouin "Cloud Computing et SaaS", Dunod,Paris, 1-ere _edition 2009, 2-eme édition 2012
7. Christopher M. Moyer, "Building Applications in the Cloud : Concepts, Patterns, and Projects" Addison-Wesley, 2011, (Pearson 2011, en français) pour les développeurs
8. Eric A. Marks, Bob Lozano "Executive's Guide to Cloud Computing", Wiley, 2010
9. Charles Babcock "Management Strategies for the Cloud Revolution", McGraw-Hill, 2010

Semestre : S9

Unité d'enseignement : UEM1

Matière : *Sécurité Logicielle*

Crédits : 3

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Le module de sécurité logicielle a pour but de comprendre les risques de sécurité dus aux failles du système informatique et des applications d'une part et d'autre part de faire face aux différentes attaques possibles. De plus, proposer de bonnes pratiques de développement logiciel. Mais surtout de comprendre les différentes couches de sécurité d'un système informatique, de l'authentification, du contrôle d'accès et de confidentialité.
Aussi, les innovations technologiques influencent la sécurité des applications et augmentent leurs vulnérabilités. A l'heure actuelle, il est essentiel de concevoir des projtes et travaux en phase avec les besoins d'apprentissage, tout en donnant aux étudiants les moyens de mettre efficacement leurs connaissances en pratique.

Connaissances préalables recommandées : Introduction à la sécurité d'Informatique (ISEC)

Contenu de la matière

Chapitre I : Rappel sur la cryptographie

1- Introduction
2- Objectifs de la sécurité
3- Sécurité moderne

Chapitre II : Authentification

1- Définitions
2- Type d'authentification
3- Authentification centralisée
4- Authentification distribuée
5- Authentification Multi-facteurs
6- Les protocoles d'authentification de connexion

Chapitre III : Contrôle d'accès

1. Définitions
2. Caractéristiques des modèles de contrôle d'accès
3. MAC
4. DAC
5. RBAC
6. Modèles à base de XML
7. Contrôle d'accès et applications

Chapitre IV : Confidentialité

1. Définitions
2. Modèles de confidentialité
3. Partage de la donnée

Chapitre V : Attaques et moyens de défense / Cyber sécurité

1. Définition d'une attaque
2. Les vulnérabilités logicielles
3. Classification des hackers
4. Classification des attaques
5. Scénarios d'attaques
6. Les virus, chevaux de Troie
7. Systèmes de détection ou de prévention d'intrusions

Chapitre V : Impact des nouveaux environnements sur la sécurité logicielle

1. Sécurité des applications Web
2. Sécurité dans les réseaux sociaux
3. Sécurité dans le bigdata (IoT et Cloud computing)
4. Blockchain
5. Vie privée et anonymat

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

1. Salwa Alem, David Espes, Florent De Lamotte, Eric Martin, Laurent Nana. "Hybrid Intrusion Detection System in Industry 4.0 based on ISA95 standard", 16th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 2019), Saoudi Arabia, 2019."
2. CyberArk. URL https://www.cyberark.com/threat-research-blog/anatomy-triton-malware-attack/ (accessed 5.2.18)
3. Ashibani, Y., Mahmoud, Q.H., 2017. Cyber physical systems security: Analysis, challenges and solutions. Computers & Security 68, 81–97. https://doi.org/10.1016/j.cose.2017.04.005

Semestre : ***S9***

Unité d'enseignement : UEM1

Matière : ***Développement mobile***

Crédits : 3

Coefficient : 2

Mode d'enseignement : Présentiel

Objectifs de l'enseignement

Ce cours a pour objectif de :

- Présenter aux étudiants les fondements nécessaires pour prendre en main la programmation mobile (composants d'une application mobile, cycle de vie, évènements…).
- Permettre aux étudiants de concevoir et programmer des applications mobiles sous Android.

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Système d'exploitation mobile, POO, Java, XML

Contenu de la matière

Chapitre I : Environnement de développement Android

I.1 Qu'est-ce qu'Android ?

I.1.1 Historique et versions

I.1.2 Architecture générale d'Android

I.2 Kit de développement Android

I.2.1 Android SDK

I.2.3 SDK Manager

I.2.3 Android Virtual Device (AVD) Manager

I.2.4 Les outils SDK (adb, émulateur,...)

I.2.5 IDE et Plugins

I.2.6 Langages de programmation

1.3 Applications Android

I.3.1 Eléments applicatifs d'une application Android : Activité, services, fournisseur de contenu

I.3.2 Eléments d'interaction d'une application Android : Intents, récepteur d'Intents, Notifications.

I.3.3 Permissions

I.3.4 Le fichier de configuration AndoidManifest.xml

1.4 Structure d'un projet Android

1.5 Création, Exécution et lancement d'une application

1.6 Création d'un paquet installable

Chapitre II : Création d'interfaces utilisateur

II.1 Le concept d'interface

II.2 Structure d'une interface Android

II.3 Les vues
- II.3.1 TextView, ImageView
- II.3.2 EditText
- II.3.3 Button, CheckBox
- II.3.4 Autres vues

II.4 Groupes de vues

II.5 Les gabarits (layouts)
- II.5.1 LinearLayout
- II.5.2 RelativeLayout
- II.5.3 FrameLayout
- II.5.4 TableLayout
- II.5.5 Autres

II.6 Création d'interfaces
- II.6.1 Définir une interface en XML
- II.6.2 Définir une interface par instanciation (associer une interface à une activité)
- II.6.3 Créer une interface sans définition XML

II.7 Gestion d'événements

Chapitre III : Communication entre applications

III.1 Principe de fonctionnement

III.2 Navigation entre écrans au sein d'une application
- III.2.1 Lancement d'une activité
- III.2.3 Lancement d'une activité et obtenir un retours

III.3 Lancement d'activités d'autres applications.
- III.3.1 Déléguer au système le choix de l'application
- III.3.2 Accorder les permissions liées aux actions
- III.3.3 Filtrer les actions
- III.3.4 Exploiter l'objet Intent d'une activité

III.4 Intents
- III.4.1 Démarrer un service Intent
- III.4.2 Transporter des données dans un Intent (Ajout, extraction)
- III.4.3 Transférer un Intent
- III.4.4 Intent en mode déclaratif

III.5 Diffuser et recevoir des intents
- III.5.1 Notion de diffusion
- III.5.2 Diffuser des Intents
- III.5.3 Recevoir et traiter des Intents diffusés
- III.5.4 Créer des répéteurs d'Intents dynamiquement
- III.5.5 Les messages d'informations natifs.

Chapitre IV : Interfaces graphiques avancées

IV.1 Création d'interfaces personnalisées

IV.2 Les adaptateurs pour accéder aux données d'interface

IV.3 Barre d'action et menus
- IV.3.1 Création d'un menu
- IV.3.2 Mise à jour dynamique d'un menu
- IV.3.3 Création des sous menus
- IV.3.4 Menus en cascade

IV.3 Menus contextuels
IV.4 Annonces : toasts
IV.5 Dialogues
- IV.5.1 Dialogue d'alerte
- IV.5.2 Boutons et affichage d'un dialogue d'alerte
- IV.5.3 Dialogues personnalisés
- IV.5.4 Création d'un dialogue
- IV.5.5 Affichage du dialogue

IV.6 Liste d'items
- IV.6.1 ListView, adapterView
- IV.6.2 ArrayList
- IV.6.3 Actions sur les listes

5.4 Fragments et activités

Chapitre V : Persistance et partage des données

V.1 Persistance des données
V.2 Préférences partagées
- V.2.1 Récupérer les préférences partagées
- V.2.2 Enregistrer ou mettre à jours les préférences partagées
- V.2.3 Les permissions des préférences
- V.2.4 Réagir à la modification des préférences avec événements.

V.3 Stockage dans des fichiers
- V.3.1 Opérations sur les fichiers (lire, écrire, supprimer...)
- V.3.2 Partager un fichiers avec d'autres applications
- V.3.3 Intégrer des ressources dans les applications.
- V.3.4 Gérer les fichiers

V.4 Stockage dans une base de données QSLite
- V.4.1 Caractéristique d'une BD SQLite
- V.4.2 Création et mise à jours d'une BD SQLite
- V.4.3 Accéder à une BD SQLite
- V.4.4 Effectuer des requêtes dans une BD SQLite (sélection, insertion, modification,...)

V.5 Partage de données et fournisseurs de contenu
- V.5.1 Accéder à un fournisseur de contenu
- V.5.2 Créer un fournisseur de contenu
- V.5.3 Les gabarits pour exposer les fournisseur de contenu

V.6 Partage de données et les dossiers dynamiques (LiveFolders)

Chapitre VI : Services de localisation et de cartographie

VI.1 Déterminer la position courante

VI.1.1 Obtenir la liste des fournisseurs de position
VI.1.2 Choix d'un fournisseur de position en fonction des critères

VI.2 Les formats des fichiers de position pour l'émulateur (GPX, KML de google earth)
VI.3 Simulateur et fausses positions
VI.4 Alertes de proximité
VI.5 Les API Google
VI.6 Création d'activités utilisant Google Maps
VI.6.1 Étendre la classe MapActivity
VI.6.2 Manipuler la classe MapView
VI.6.3 Manipuler la classe MapController
VI.7 Placer des données sur une carte

Mode d'évaluation : (type d'évaluation et pondération)

Examen de TP (40%), Examen (60%)

Références

Patrick Auxerre. Kotlin : Développement d'applications mobiles. Edition Ellipses, 2020.
Anthony Cosson. Kotlin : Les fondamentaux du développement d'applications Android. Edition Eyrolles, 2018
Sylvain HEBUTERNE. Android : Guide de développement d'applications Java pour Smartphones et Tablettes. Edition ENI, 2016.
Grant Allen. L'art du développement Android. Édition Pearson Education, 2012.
Damien Guignard, Julien Chable, Emmanuel Robles. Programmation Android : De la conception au déploiement avec le SDK Google Android 2. Edition Eyrolles. 2011
Pierre Nerzic. Programmation mobile avec Android. Support de cours, 2023.

Semestre : *S9*

Unité d'enseignement : UET1

Matière : *Aspects juridiques*

Crédits : 2

Coefficient : 1

Mode d'enseignement : Présentiels

Objectifs de l'enseignement

- Connaitre l'aspect juridique et la réglementation de la cybersécurité en Algérie

Connaissances préalables recommandées (*descriptif succinct des connaissances requises pour pouvoir suivre cet enseignement – Maximum 2 lignes).*

Contenu de la matière

- Rappels sur la Cybersécurité
- Géopolitique du Cyber Espace et Cyber Attaques
- Cybersécurité pour les Systèmes d'Information
- Cybersécurité à l'ère des réseaux sociaux
- Souveraineté Numérique et respect de la vie privée
- Politiques de Cybersécurité à travers le Monde et coopération internationale
- Politiques de Cybersécurité à travers le Monde et coopération internationale
- Stratégie de lutte contre Incidents Cybernétiques

Mode d'évaluation : (type d'évaluation et pondération)

Contrôle continu (40%), Examen (60%)

Références

- Compilation des décrets, arrêtés (Réglementations en vigueurs)

IV- Accords / Conventions

(Champ obligatoire)

V – Curriculum Vitae succinct
De l'équipe pédagogique mobilisée pour la spécialité
(Interne et externe)

(selon modèle ci-joint)

VI - Avis et Visas des 0rganes Administratifs et Consultatifs

VII – Avis et Visa de la Conférence Régionale
(Uniquement dans la version définitive transmise au MESRS)

VIII – Avis et Visa du Comité pédagogique National de Domaine
(Uniquement dans la version définitive transmise au MESRS)