Proposition de programme de formation pour les enseignants chargés de la spécialité *Informatique et sciences du numérique* en terminale S

La formation des enseignants chargés de la spécialité *Informatique et sciences du numérique* en terminale S à la rentrée 2012 constitue un défi, qui peut être relevé, si ces enseignants consacrent une journée par semaine à se former, pendant les deux années académiques 2010-2011 et 2011-2012, soit $2 \times 36 \times 2 = 144$ demi-journées (432 heures). Nous proposons de répartir ces 144 demi-journées en 12 cours de 12 demi-journées chacun (36 heures), par exemple en organisant, chaque trimestre, deux cours en parallèle. Chaque cours se conclut naturellement par un examen qui peut prendre plusieurs formes : sur table, oral, validation de projet, ...

Chaque demi-journée est typiquement constituée d'un cours magistral d'une heure et demie et d'une séance de travaux dirigés ou de travaux pratiques d'une heure et demie également. Certains cours demandent une séance de travaux dirigés, mais la plupart demandent plutôt une séance de travaux pratiques, devant un ordinateur. Ces séances de travaux pratiques permettent aux enseignants en formation à la fois de mettre en œuvres les notions vues en cours et de consolider leur savoir-faire en programmation. À partir d'un certain moment dans la formation, ces séances de travaux pratiques s'enchaînent naturellement en mini-projets.

Il est important de veiller à mettre en valeur les nombreuses articulations entre les différent concepts théoriques et pratiques présentés dans les différents cours. Ces articulations se placent à la fois au niveau conceptuel (la programmation fait appel aux concepts de langage et d'algorithme, la compilation aux concepts de langage et de machine, ...) et au niveau technique (ces concepts étant utilisés conjointement dans la conception des applications modernes et des objets numériques qui nous entourent). Une application suffisamment riche (un système de modélisation / simulation d'un processus physique, un jeu, un smartphone, etc.) peut servir de fil rouge à l'ensemble de la formation, chaque cours détaillant la manière dont les notions introduites dans ce cours sont utilisées dans la conception de cette application.

Certaines parties du cours pourront aussi faire appel à des conférences vidéo ou des webinars (séminaires sur le web).

En plus de cette formation aux concepts fondamentaux de l'informatique, cet enseignement doit aussi aborder des questions de didactique de la discipline,

typiquement sous forme de conférences, organisées tout au long de la formation. Nous donnons dans ce document une liste de questions qui sont au centre des réflexions sur la pédagogie de l'informatique.

1 Les cours

L'informatique est une science structurée autour de quatre concepts, ceux d'information, de langage, d'algorithme et de machine. Une règle importante dans l'organisation de tout enseignement d'informatique, qu'il soit destiné aux lycéens ou à leurs professeurs, est de veiller à un équilibre entre ces quatre notions. Cela nous a amené à organiser cette proposition de programme en quatre groupes de trois cours.

1.1 Information

1.1.1 Représentation numérique de l'information (36h, année I)

Codage numérique du texte (toutes langues), de l'image, du son, de la vidéo, de la géométrie, des forces, etc. Exemples d'opérations numériques : inversion vidéo d'une image, augmentation du contraste.

Interfaces, capteurs et actionneurs, orientés hommes (clavier, souris, écran) ou physique (CCDs, accéléromètres, etc.). Valeurs numériques et événements abstraits associés.

Densité d'information. Compression générique et spécifique. Compression maximale, complexité de Kolmogorov. Code préfixe, méthode de Huffmann. Compression d'une image par arbre quaternaire. Codage cryptographique.

1.1.2 Structuration et contrôle de l'information (36h, année I)

Structures de données de base : types numériques, enregistrements, etc. Vision abstraite et indépendance de l'implémentation par l'introduction d'API (Application Programming Interface). Persistance de données : linéarisation et sauvegarde en fichiers.

Structures universelles du web : les langages HTML et XML, les URL et DNS, les pages web statiques et dynamiques (PHP-MySQL).

Protection de données. Où les informations sont-elles stockées ? Contrôle des accès et pare-feux. Licences. Propriété intellectuelle. Respect de la vie privée. Droit à l'oubli.

1.1.3 Bases de données et systèmes d'information (36h, année II)

Algèbre relationnelle, calcul relationnel, théorème d'équivalence. SQL, requêtes et mises-à-jour. Structure d'accès, table de hachage et arbre B. Optimisation de requêtes. Conception de schéma, UML, dépendances fonctionnelles et multivaluées. Concurrence et distribution.

Architecture fonctionnelle des systèmes d'information (décomposition descendante, modularité, service, processus). Modélisation et architecture des données (UML, distribution, transaction). Fiabilité du système d'information (disponibilité, durée de reprise, ordres de grandeur, analyse d'ensemble). Performance (débit, latence, files d'attente).

1.2 Langage

1.2.1 Langage de programmation (36h, année I)

Le noyau impératif (affectation, séquence, test, boucle, déclaration). La notion d'état et de transformation d'état. Fonction. Récursivité. Allocation. Type de données dynamique. Partage, copie. Objet, module.

Méthodes et outils de développement (méthodes de debugging, éléments de génie logiciel).

1.2.2 Automate et grammaire (36h, année II)

Automate d'états finis, application (interface homme-machine, contrôle discret, protocole, etc.). Expression régulière. Grammaire. Langue naturelle et langage formel (classification de Chomsky) Principes d'un analyseur syntaxique simple. Algorithmique du texte (recherche d'une sous-chaîne, appartenance à un dictionnaire, algorithmes sur le génome).

1.2.3 Compilation et vérification (36h, année II)

Sémantique opérationnelle à petits et grands pas. Réalisation d'un interpréteur pour un langage simple. Machine abstraite, transformation de l'interpréteur en compilateur. Principes d'optimisation.

Test, vérification, preuve.

1.3 Algorithme

1.3.1 Algorithmes classiques (36h, année I)

Tri, recherche en table. Parcours d'arbres en profondeur et en largeur d'abord. Arbres de recherche. Méthode min/max. Parcours de graphe en profondeur et en largeur d'abord. Algorithme RSA. Notion de protocole cryptographique. Exemples d'algorithmes géométriques (enveloppe convexe, algorithme de Bresenham, ...).

1.3.2 Calculabilité et complexité (36h, année II)

Complexité en temps et en espace. Complexité en moyenne et dans le pire cas. Complexité des algorithmes de tri et borne inférieure. Classes de complexité P, EXPTIME, PSPACE. Définition de la notion de fonction calculable. Réécriture, machines de Turing, Turing-complétude. Calculabilité de l'interpréteur.

Indécidabilité du problème de l'arrêt. Machine de Turing non déterministe. La classe NP et ses centaines d'algorithmes pratiques. Le problème P $\neq$ NP.

1.3.3 Algorithmes à grande échelle (36h, année II)

Moteur de recherches. Diffusion pair à pair. Cloud computing. Évaluations probabilistes de complexité.

1.4 Machine

1.4.1 Architecture de machine (36h, année I)

Circuits Booléens : calcul d'une fonction booléenne à l'aide de portes logiques. Circuits associés aux automates d'états finis.

Composants synchrones : processeurs, DSPs, mémoires, gestionnaires mémoires, interfaces (USB, GSM, etc.). Circuits programmables (FPGAs). Composants analogiques : convertisseurs analogique / digital, radios, capteurs. Machines de Moore et de Mealy.

Structure d'un processeur : modèle de Von Neumann (mémoire, unité arithmétique et logique, contrôleur, entrée/sorties). Langage machine.

Composition de composants dans les Systèmes sur puces : mémoires distribuées, bus, files d'attentes multi-horloges. Streaming pour les signaux audio et vidéo.

1.4.2 Réseaux (36h, année I)

Notion de paquet et de protocole. Transmission d'information point à point, détection et correction d'erreurs de transmission. Réseaux local, réseau global, détection et correction d'erreurs réseau. Routage, nommage, TCP / IP. ADSL.

Décomposition en couches. Équipement (hub, switch, routeur, ...). Qualité de service. Outils de développement distribué.

1.4.3 Systèmes d'exploitation (36h, année II)

Différents types de systèmes d'exploitation (interactif, batch, embarqué dédié, temps réel). Principes de gestion de ressources (abstraction, sécurité, virtualisation). Interface programmes/noyau (appel système, abstraction de périphérique). Interfaces noyau/matériel (interruptions, projection d'entrées-sorties en mémoire). Composants du noyau (gestion mémoire, ordonnancement, système de fichiers, pilotes de périphériques). Exclusion mutuelle, algorithme de Peterson. Interface homme-machine. Système multi-fenêtre. Éléments de gestion des multiprocesseurs.

2 Questions pédagogiques pour les futurs professeurs

- Comment relier les concepts scientifiques du cours avec le monde numérique environnant ?
- Quelles parties du cours faire en direct, quelles parties faire à l'aide de supports externes (vidéos, applications Web, etc.) ?
- Comment verbaliser ce qui se passe quand on exécute un programme ?
- Comment mener une séance de travaux pratiques ? Avec quel fil rouge ? Que distribuer aux élèves ? Jusqu'à quel point faut-il les laisser chercher ? Comment partager son temps entre les différents élèves lors d'une séance de Travaux pratiques ? Comment favoriser l'aide des élèves par les élèves ?
- Quelle place donner aux enseignements magistraux au lycée ? Comment ne pas être redondant avec une séance de travaux pratiques ? Comment montrer en cours magistral des exemples de l'activité de programmation ?
- Comment équilibrer son enseignement entre les quatre concepts de langage, d'algorithme, de machine et d'information ?
- Dans quel ordre traiter les différentes parties du programme ?
- Comment utiliser l'activité de programmation pour donner son unité à l'ensemble du cours ?
- Comment transmettre l'idée que la rigueur syntaxique exigée dans un programme est une condition nécessaire pour que ce programme fonctionne ?
- À quel moment de l'année faire commencer un projet informatique par les élèves ?
- Quel langage choisir ?
- Comment intégrer la perspective de l'épreuve du bac et de ses modalités ?
- Quelle place donner aux questions transdisciplinaires ? Quelle place donner aux question sociétales liées au développement de l'informatique (respect de la vie privée, évolution de la notion de propriété, ...) ? Quelle place donner aux exposés faits par les élèves ?