Proposition de référentiel de formation des enseignants en Informatique et Sciences du Numérique

Groupe d'experts

2 septembre 2011

Formation des enseignants chargés de l'enseignement de l'Informatique et des Sciences du Numérique

Le présent document propose un schéma général destiné à maintenir une cohérence entre les diverses formations qui seront proposées par les académies aux professeurs en vue de l'enseignement de spécialité ISN ainsi que celui de préparation au BTS SIO. Il revient aux formateurs d'adapter ce cadre en fonction des stagiaires et des moyens attribués, sachant que les volumes horaires indiqués constituent un minimum ne permettant qu'une présentation simplifiée des notions essentielles.

L'informatique est une science structurée autour de quatre concepts, ceux d'information numérique, de langage, d'algorithme et de machine. C'est pourquoi le cadre de formation proposé est articulé autour de ces notions et est réparti suivant deux niveaux, dont chacun représente environ une soixantaine d'heures de formation. Chaque niveau est découpé en « chapitres » puis en « unités » logiques dont le volume est purement indicatif.

A / Plan

Niveau I		Niveau II	
Information numérique			
– Représentation numérique de l'information	10h	– Contrôle de l'information	2h
– Structuration et contrôle de l'information	10h	– Bases de données	8h
Langages			
– Langage de programmation	10h	– Automates et grammaires	8h
– D'autres langages	5h	– Compilation et vérification	8h
Algorithmes			
– Algorithmes classiques	7h	– Complexité	8h
– Quelques algorithmes plus avancés	6h	– Algorithmes à grande échelle	4h
Machines numériques			
– Architecture des machines	6h	– Réseaux	8h
– Initiation aux réseaux	4h	– Systèmes d'exploitation	8h
Mini projet		**Projet**	
– Initiation à la pédagogie de projet, exemple	2h	– Conduite de projet, réalisation de projet	6h

Les chapitres ne sont pas conçus comme successifs ; leur traitement devrait plutôt se faire de manière simultanée (en une progression « spiralée »).

Chaque unité est présentée comme un « cours » qui doit être accompagné d'un travail pratique visant à développer le savoir-faire des professeurs, ainsi que la connaissance des contextualisations possibles sans lesquelles l'enseignement de l'informatique devient vite aride. Aussi souvent que possible, les notions présentées seront confrontées aux questionnements didactiques (« comment enseigne-t-on ceci? ») ainsi qu'aux problèmes sociétaux posés par la généralisation des outils et

systèmes numériques.

La formation de niveau I s'achève avec la réalisation d'un projet personnel simple dont le but est essentiellement de faire percevoir aux professeurs les difficultés que rencontrent les élèves face à une telle activité. Corrélativement, les problématiques liées au développement de projets sont esquissées sans toutefois entrer dans des schémas trop rigides.

La formation de niveau II va de pair avec le développement d'un projet plus ambitieux dont le but est de faire percevoir les enjeux et contraintes du développement des systèmes informatiques.

Il est important de veiller à mettre en valeur les nombreuses articulations entre les différents 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).

B / Formation de niveau I

1. Représentation numérique de l'information

Principe de la représentation binaire, opérations arithmétiques en binaire, structuration en octets et en mots.

Numérisation et échantillonnage ; codage numérique des textes, de l'image ponctuelle, du son.

Opérations (ou transformations) numériques sur les images ou les sons (sous forme de tableaux de valeurs numériques) : extraction (ou recadrage), mise à la taille (ou rééchantillonnage), modification du contraste ou de l'amplitude (luminance, volume), filtrages réversibles ou non.

Principe de la compression (avec ou sans perte), compression générique et spécifique. Omniprésence des systèmes de compression dans les objets numériques (photo, audio, vidéo…).

Questions didactiques :

- Élaboration du lien entre la théorie et la pratique manipulatoire des objets numériques ;
- usage de références historiques, notamment dans le domaine de la cryptologie ;
- élaboration de liens transdisciplinaires, notamment avec la physique à propos du son et de l'image.

Les compétences suivantes doivent être développées :

- *Manipuler bits, octets et mots par des opérations arithmétiques, logiques ou de masquage ;*
- *exprimer des formules (ou opérations) logiques simples ou complexes par combinaison d'opérateurs de base ;*
- *coder, recoder, transcoder un texte, une image, un son au moyen d'un algorithme programmé, d'un logiciel dédié ;*
- *compresser une source d'information selon sa nature.*

2. Structuration et contrôle de l'information

Structures de données de base : types numériques, enregistrements composés d'une suite de champs hétérogènes et de tailles variables (textes, nombres, images...).

Données persistantes : sauvegarde en fichiers.

Organisation des informations sous forme d'une arborescence, d'un hypertexte, d'un graphe. Différentes manières de représenter un graphe, degrés, chemins, circuits. Graphes valués,

chemins associés.

Codage cryptographique : principe, exemples simples (XOR), repérage des dangers.

Codes correcteurs d'erreurs : principe, exemples simples (parité, somme).

Protection des données et persistance de l'information.

Non-rivalité de l'information, licences. Éléments juridiques sur la propriété intellectuelle et le respect de la vie privée. Questions déontologiques et éthiques (par exemple, la revendication du « droit à l'oubli »).

Questions didactiques :

- Élaboration de démarches interdisciplinaires autour des questions sociétales ;
- conduite de recherches documentaires autour des questions juridiques ;
- conduite de débats en classe autour des questions éthiques.

Les compétences suivantes doivent être développées :

- *Structurer des informations, choisir des types de données appropriées ;*
- *organiser des informations suivant une arborescence, un graphe ;*
- *connaître et savoir expliquer de manière générale les éléments du droit relatif aux supports numériques (droit d'auteur, LCEN, DADVSI, HADOPI) ;*
- *choisir une licence et un modèle de diffusion pour une création numérique ou logicielle ;*
- *choisir un format de document approprié par rapport à un usage (RGI) ;*
- *choisir et manipuler un système de chiffrement à clé publique ;*
- *vérifier la validité d'un téléchargement en recourant aux mécanismes dédiés (SHA, MD5) ;*
- *analyser un problème de sécurité, un problème d'archivage de données.*

3. Algorithmes classiques

Algorithmes de base :

- Rechercher un élément dans un tableau déjà trié par une méthode dichotomique ;
- rechercher un élément dans un tableau ;
- additionner, soustraire, multiplier deux entiers exprimés en binaire.

Algorithmes plus avancés :

- Parcours d'arbres en profondeur et en largeur d'abord, arbres de recherche ;
- recherche d'un chemin dans un graphe par un parcours en profondeur (Tarjan, DFS) ;
- recherche d'un plus court chemin par un parcours en largeur (Dijsktra, BFS) ;
- tri par fusion et principe « diviser pour régner » ;
- introduction aux problèmes et méthodes d'ordonnancement (MPM, PERT) ;
- exemples d'algorithmes géométriques (enveloppe convexe, algorithme de Bresenham, ...).

Questions didactiques :

- Présentation, à partir des divers enseignements suivis par les élèves, de situations de nature algorithmique et de situations non-algorithmiques ;
- sensibilisation aux questions de performance et complexité ;
- sensibilisation à la nécessité de justification (ou preuve) d'un algorithme.

Les compétences suivantes doivent être développées :

- *Comprendre un algorithme et expliquer ce qu'il fait ;*
- *concevoir une solution algorithmique à un problème, savoir l'implanter (programmer) et*

savoir l'analyser ;

- *modifier un algorithme existant pour obtenir un résultat différent ;*
- *s'interroger sur l'efficacité d'un algorithme.*

4. Langages

Langage de programmation :

Noyau impératif : affectation, séquence, test, boucle, déclaration.
Opérateurs numériques, booléens, de chaînes de caractères (concaténation).
Notion d'état et de transformation d'état.
Fonctions, récursivité (terminale, non terminale).
Allocation mémoire (locale, globale) et types de données dynamiques.
Données en partage, copie. Effets de bord.
Méthodes et outils de développement (méthodes de débogage, sources d'erreur usuelles).

Questions didactiques :

- Accompagnement des élèves dans la prise en main des activités de programmation de sorte que ce ne soit ni rebutant ni limité à des situations trop simples ;
- recherche d'un « style » de programmation correct : exigences de clarté, exactitude, transférabilité, etc.
- intérêt et limites des systèmes de visualisation d'algorithmes (organigrammes, UML, etc.) ;
- valorisation de la lisibilité des productions des élèves (commentaires, nommage, modularité, documentation, réutilisation de composants) ;
- développement de l'autonomie des élèves face à un projet comportant une part de programmation ;
- conduite de projet.

Les compétences suivantes doivent être développées :

- *Programmer un algorithme décrit en langue naturelle ;*
- *comprendre un programme et exprimer en langue naturelle l'algorithme sous-jacent ;*
- *choisir un type de donnée en fonction d'un problème à résoudre ;*
- *concevoir l'entête (ou l'interface) d'une fonction, puis la fonction elle-même ;*
- *documenter une fonction, un programme ;*
- *mettre un programme au point en le testant, l'instrumentant ;*
- *utiliser un outil de mise au point.*

D'autres langages :

Structures universelles du web : les langages HTML et CSS.
Initiation aux expressions régulières.

Questions didactiques :

- Développement de l'autonomie des élèves face à un projet impliquant une web-publication ;
- valorisation des standards : W3C, RFC, RGI, etc ;
- séparation des notions de « langage formel » et « programmation ».

Les compétences suivantes doivent être développées :

- *Créer une page web en dissociant le contenu de la mise en forme ;*
- *utiliser un validateur en ligne ;*

- *créer un site dynamique simple, éventuellement à partir d'une base téléchargée.*

5. Architecture de machine

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

Structure d'un processeur : modèle RAM (Random Access Machine, antérieurement de Von Neumann) : mémoire, unité arithmétique et logique, contrôleur, entrée/sorties.

Interfaces, capteurs et actionneurs, orientés hommes (clavier, souris, écran) ou physique (CCDs, microphones, accéléromètres, ...).

Questions didactiques :

- Expliquer aussi simplement que possible comment fonctionne un ordinateur (avec quels constituants), dans une continuité entre matériel et logiciel, sans entrer dans des détails très techniques ;
- repérer et faire repérer la diversité des machines numériques : téléphones, clés USB, appareils photo numériques, baladeurs mp3, capteurs « intelligents », etc.

Les compétences suivantes doivent être développées :

- *Identifier le rôle des constituants d'un ordinateur ;*
- *« dérouler » l'exécution d'une séquence d'instructions simples de type langage machine ;*
- *acquérir et afficher un signal à l'aide d'un logiciel d'acquisition et de visualisation.*

6. Réseaux

Notions de paquet et de protocole.

Transmission d'information point à point, (liaison sérielle), détection et correction d'erreurs de transmission.

Réseaux local, réseau global, différents types d'adressage.

Routage, nommage, TCP / IP.

Équipement réseau (hub, switch, routeur, ...).

Questions didactiques :

- Observation et simulation d'un réseau en fonctionnement au moyen de logiciels dédiés ;
- analyse des problèmes posés par la supranationalité des réseaux ;
- Questions juridiques et philosophiques liées à la circulation de l'information.

Les compétences suivantes doivent être développées :

- *Établir une communication sérielle entre deux machines ;*
- *analyser le trafic sur une liaison sérielle ;*
- *décrire une situation d'adressage sur un type de réseau particulier ;*
- *analyser les entêtes de messages électroniques, pour décrire le chemin suivi par l'information ;*
- *piloter un système de lecture et d'écriture de données numériques.*

C / Formation de niveau II

Le descriptif de cette composante de la formation est volontairement plus sommaire.

1. Logique et circuits

Calcul propositionnel, quantificateurs, algèbre booléenne.

Circuits booléens : calcul d'une fonction booléenne à l'aide de portes logiques.

Circuits associés aux automates d'états finis.

2. Contrôle de l'information

Codage cryptographique : principe et pratique des cryptosystèmes à clé secrète, à clé publique.

Codes correcteurs d'erreurs : principe et pratique des systèmes de hachage (MD5, SHA1).

Protection des données et persistance de l'information.

Contrôle des accès et pare-feux. Politiques de sécurité.

3. Bases de données

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, UML2, dépendances fonctionnelles et multivaluées. Concurrence et distribution.

4. Automates et grammaire

Automate d'états finis, applications (interface homme-machine, contrôle discret, protocole, ...).

Expressions régulières. Grammaire. Principes d'un analyseur syntaxique simple. Algorithmique du texte (recherche d'une sous-chaîne, appartenance à un dictionnaire, algorithmes sur le génome).

5. Compilation et vérification

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.

6. Complexité

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.

Le problème $P \neq NP$.

7. Algorithmes à grande échelle

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

8. Systèmes d'exploitation

Le traitement de cette partie doit être accompagné de mises en pratique variées.

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 multifenêtres. Éléments de gestion des multiprocesseurs.