zéro un informatique

N° 107 - février 197

Des OBLIGATIONS respectives de l'utilisateur et du constructeur

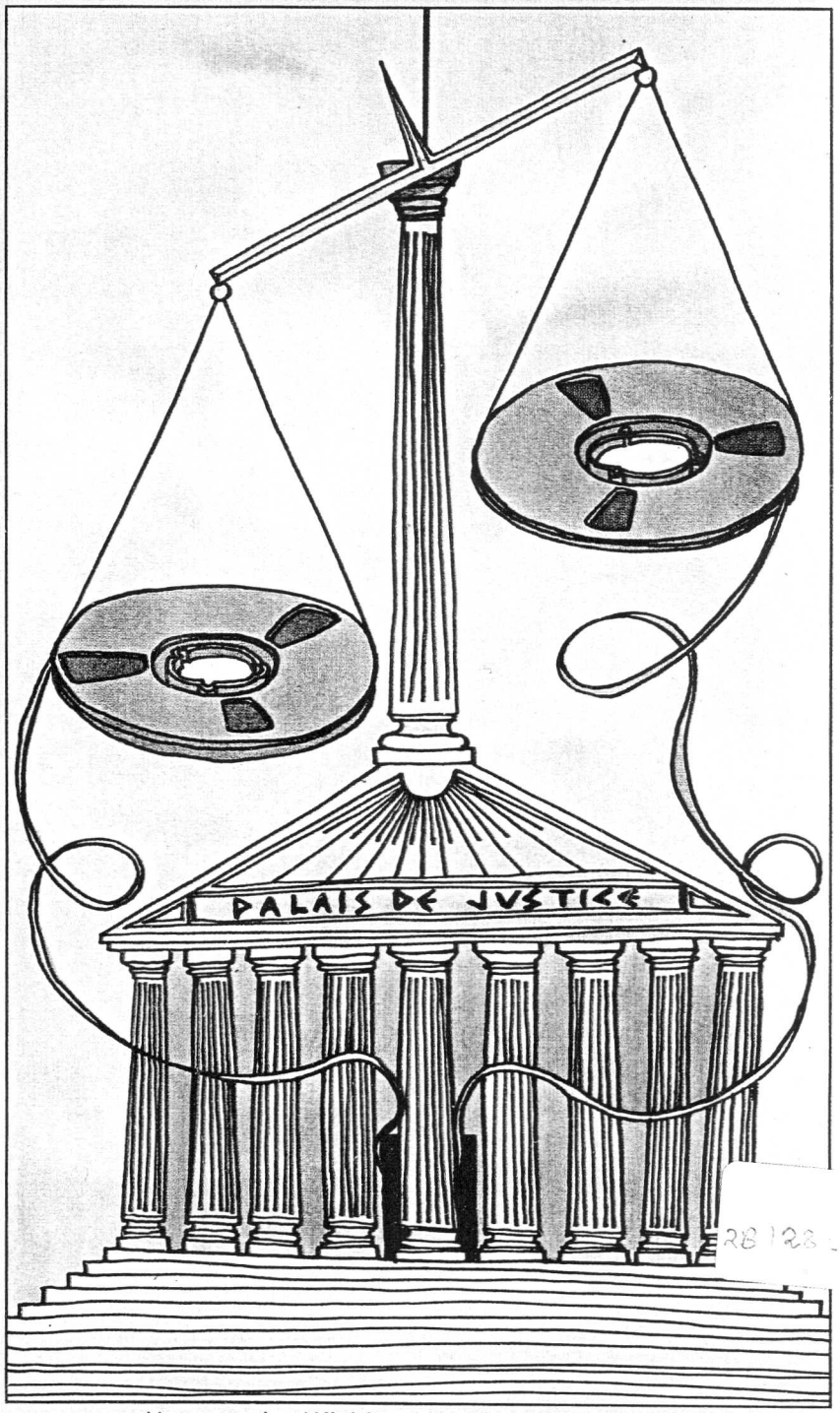
Programmation structurée : un langage sans GO TO

Les MINIS-GESTION : méthodes de choix et prix

Les fibres optiques en transmission de données

Contre les risques d'altération, un système de CERTIFICATION des informations

Comment ENSEIGNER la programmation



Une mauvaise définition de la relation contractuelle liant utilisateur et constructeur peut les conduire devant les tribunaux. Les magistrats s'appuient maintenant sur une certaine jurisprudence

Quand on doit recruter * UN

INFORMATICIEN

on insère une Petite Annonce dans le journal

LU

par les informaticiens

* Si vous ne vous préoccupez pas personnellement du recrutement des informaticiens de votre entreprise, veuillez vous assurer que les responsables intéressés connaissent bien 01

Pour transmettre votre texte

PARIS

- **le télex :** EDITEST 230 589 F
- **le télécopieur** 202 29 10 (Rank Xerox 400)
- **le courrier :** 41, rue de la Grange-aux-Belles, 75010 Paris

AVANT LE MERCREDI SOIR

BRUXELLES

- **le téléphone :** 425 56 58
- **le courrier :** 110, av. Broustin, 1080 Bruxelles

AVANT LE MARDI SOIR

LAUSANNE

- **le téléphone :** — (021) 32 61 77
- **le courrier :** — 40, chemin de Pernessy, 1052 le Mont-sur-Lausanne

AVANT LE MARDI SOIR

zéro·un·informatique 01 hebdo

Comment enseigner la programmation

Claude Kaiser, Bertrand Meyer, Etienne Pichat

L'article qui suit commente l'enseignement à l'Institut d'Informatique d'Entreprise de l'informatique (de base) et notamment de la programmation. Il ne traite pas de l'informatique d' organisation, des méthodes quantitatives, de l'économie et gestion, enseignées cependant à tous les élèves. L'horaire de l'enseignement d'informatique est environ le quart de l'horaire global des enseignements, comme l'indique le tableau des différents enseignements.

L'INSTITUT D'INFORMATIQUE D'ENTREPRISE

L'Institut d'Informatique d'Entreprise a pour mission de former des ingénieurs spécialistes de l'informatique appliquée à l'entreprise. Créé dans le cadre du Conservatoire National des Arts et Métiers, l'IIE profite de sa pluridisciplinarité et de son corps enseignant. Les liens de l'IIE avec l'industrie sont étroits grâce à sa Commission Technique, ses enseignants, ses anciens élèves regroupés dans l'Association des Ingénieurs IIE, les mémoires de 3e année, les stages et les travaux pratiques.

Les élèves (50 par promotion) ont au départ une solide formation scientifique. En effet 35 élèves de mathématiques spéciales à programme M ou P' sont recrutés sur le concours commun Ecole Centrale des Arts et Manufactures — Ecole Supérieure d'Electricité, 5 titulaires du Deug mention Sciences sont recrutés sur le concours commun national d'admission aux ENSI, enfin une dizaine de titulaires du DUT option informatique sont recrutés sur concours spécial.

L'enseignement se déroule sur 3 ans avec une dernière année consacrée, 75 % du temps, au travail personnel dans le cadre d'équipes de recherche-développement de l'IIE, ou d'organismes extérieurs (entreprises, administrations) ; il s'articule autour de trois grands thèmes : informatique, méthodes quantitatives, économie et gestion ; il se réalise dans l'enseignement carrefour qu'est l'informatique d'organisation. En mettant en avant la finalité de l'ensemble des études, on peut dire : d'une part, elles doivent faire acquérir aux élèves une grande maîtrise des disciplines qui participent à la conception et à la mise en œuvre d'un système informatique de gestion (informatique et notamment programmation et systèmes informatiques, informatique de gestion organisation, technique d'aide à la décision, psychosociologie de l'équipe et de l'entreprise) ; d'autre part, elles doivent familiariser les élèves avec les domaines d'application afin qu'ils puissent dialoguer avec les utilisateurs et construire des systèmes d'information adaptés à leurs besoins (techniques financières et comptables, gestion financière, choix d'investissement, économétrie, marketing, gestion du personnel, de la production...).

FINALITE PROFESSIONNELLE DE L'ENSEIGNEMENT

L'IIE a comme rôle de former des ingénieurs qui soient capables d'introduire les techniques modernes de l'informatique dans la gestion de l'entreprise, tant pour l'automatisation de ses fonctions que pour le traitement des informations qui servent à la diriger. Cette activité professionnelle nécessite de savoir utiliser et choisir des systèmes informatiques, réaliser des programmes longs et complexes. Dans l'état actuel de la technique ces systèmes font appel aux composants les plus divers (macro, mini, micro, péri, téléinformatique...) et pré- ▷

▷sentent des organisations variées (centralisées, décentralisées, réparties, distribuées...).

L'enseignement de l'IIE vise à donner une formation pratique solide, étayée par des bases théoriques destinées à fournir les mécanismes conceptuels nécessaires pour dominer la complexité des systèmes informatiques. On peut schématiser l'orientation de l'enseignement en précisant que la formation pratique doit procurer aux étudiants un bagage utilisable pendant les premières années (5 à 10 ans) de leur activité professionnelle et que la formation théorique doit leur permettre de s'adapter rapidement aux techniques telles qu'elles seront découvertes ou développées pendant ce temps. Enfin, comme il s'agit d'une formation d'ingénieur, on insiste plus particulièrement sur les aspects quantitatifs au niveau de l'utilisation et du contrôle des techniques.

Une autre idée directrice de cet enseignement est qu'il est préférable de le consacrer à l'étude approfondie d'un petit nombre de domaines bien choisis plutôt que de fournir un vernis superficiel sur un grand nombre de méthodes. En effet, la connaissance technique générale peut s'obtenir rapidement mieux et avec de meilleures motivations qu'à l'école, pendant les premières années professionnelles alors que la réflexion et l'acquisition des concepts de base y sont plus difficiles.

LES THEMES DE L'ENSEIGNEMENT

Les grands thèmes de l'enseignement de l'informatique découlent de la finalité professionnelle de l'IIE et sont la programmation et les systèmes informatiques.

• *L'enseignement de la programmation* vise à ce que les ingénieurs IIE soient d'excellents programmeurs.

Par programmation, nous entendons le processus complet qui permet la résolution de problèmes sur ordinateurs, c'est-à-dire la spécification, l'écriture, la certification, la mise au point, la documentation, la maintenance, l'évaluation et l'utilisation d'un programme.

Cet enseignement commence par l'apprentissage de l'algorithmique et aboutit à dégager de véritables méthodes de travail après avoir passé en revue les algorithmes fondamentaux liés à la gestion des informations. Il indique comment se servir d'une machine, comment elle est constituée et comment elle fonctionne. Il doit donner les bases théoriques qui permettent de répondre très clairement aux questions :

— *Qu'est-ce qu'un algorithme ?*
— *Qu'est-ce qu'une machine ?*
— *Qu'est-ce qu'un langage de programmation ?*

• *L'enseignement des systèmes informatiques* vise à permettre aux ingénieurs IIE de comprendre le fonctionnement des systèmes, de les utiliser, de les choisir et d'en réaliser.

Il commence par l'apprentissage des techniques informatiques de base, par une description des fonctions remplies par les systèmes informatiques et par une classification fonctionnelle de ceux-ci. Cette initiation se termine par une description anatomique simplifiée du système du laboratoire de calcul et par son utilisation.

Puis les principes de conception des systèmes d'exploitation sont dégagés sous diverses rubriques : gestion des processus et communications entre processus, gestion des ressources, gestion des informations, de leur dénomination et des relations entre objets.

On présente également les problèmes généraux de la réalisation et de la mise en œuvre des systèmes informatiques : fiabilité, évaluation, évolution, mise en œuvre, documentation, gestion de l'exploitation d'un ordinateur.

Ces principes et ces problèmes recouvrent des aspects rencontrés plus ou moins dans tous les systèmes. Ils sont illustrés par des études détaillées de cas bien choisis et couvrant :

— les systèmes centralisés (OS/VM et HB 64) ;
— l'informatique répartie et les réseaux d'ordinateurs ;
— la gestion en temps réel ;
— les bases de données.

ORGANISATION DES ETUDES

Comme les élèves ignorent tout de l'informatique en arrivant à l'IIE, il nous est apparu nécessaire de leur faire suivre le rythme pédagogique suivant :

• *En première année,* acquisition d'une expérience pratique au cours de laquelle ils sont confrontés sur le terrain des faits avec les problèmes et les méthodes de l'informatique : au cours de cette phase, ils reçoivent leurs motivations pour l'informatique et acquièrent les bases expérimentales nécessaires pour développer leur intuition et pour comprendre les concepts présentés par la suite. C'est une période d'accumulation de connaissances : l'enseignement des problèmes et des techniques se fait d'une manière descriptive, l'enseignement des méthodes passe par l'apprentissage d'un comportement.

• *En deuxième année,* mise en évidence des concepts et développements théoriques associés : il s'agit de dégager des classes de problèmes et d'étudier leurs structures communes ; c'est une phase d'analyse critique, de réflexion et de dépassement de la connaissance expérimentale.

L'enseignement, nécessairement plus normatif, utilise parfois les connaissances acquises dans les cours de mathématiques (particulièrement mathématiques pour l'informatique) et réclame surtout des élèves l'habitude de manier l'abstraction.

• *En troisième année,* études de problèmes spécialisés en liaison avec l'orientation de l'IIE et le mémoire choisi par l'élève. C'est l'élargissement des connaissances théoriques et pratiques des élèves par des exposés synthétiques utilisant autant que possible les concepts mis en évidence au cours de l'enseignement théorique et s'appuyant sur l'expérience personnelle des élèves : réseaux d'ordinateurs, système de gestion en temps réel, langages et compilation, modèles de performance des systèmes informatiques, ergonomie informatique...

DISCIPLINES	MATHEMATIQUES ET METHODES QUANTITATIVES	INFORMATIQUE DE BASE	INFORMATIQUE D'ORGANISATION	ECONOMIE ET GESTION	COMMUNICATION
1ERE ANNEE	algèbre linéaire et analyse mathématique pour les seuls élèves issus des IUT : 120 h analyse fonctionnelle calcul des probabilités analyse numérique théorie des graphes	initiation à l'algorithmique et à la programmation : 80 h informatique générale : 80 h structures d'informations : 40 h assembleur IBM/360 : 60 h fonctions et utilisation d'un système d'exploitation : 20 h	Cobol et analyse organique Organisation	macroéconomie microéconomie comptabilité générale droit	communications anglais
2EME ANNEE	analyse convexe et applications recherche opérationnelle statistique mathématiques pour l'informatique	conception des systèmes d'exploitation : 60 h organisation et mise en œuvre des systèmes d'exploitation : 40 h	analyse et conception des systèmes informatiques de gestion banque de données théories et systèmes d'organisation	analyse des coûts et gestion de l'entreprise gestion prévisionnelle et contrôle de gestion gestion financière choix d'investissement économétrie et prévision droit des affaires	sociologie anglais
3EME ANNEE		langages et compilation : 20 h un grand système de gestion en temps réel : 30 h	modélisation d'entreprise modèles de gestion financière	stratégie marketing stratégie et politique financières	
		MEMOIRE (700 H) ET CONFERENCES (50 H)			
Total d'heures par disciplines	400	430	300	400	170

MOTIVATION

Tout discours pédagogique n'est véritablement perçu que si les élèves sont motivés ; une fois la motivation acquise, les techniques d'enseignement deviennent secondaires.

La motivation est développée par divers facteurs :

— *défi intellectuel* posé par la résolution de problèmes, conçue comme un jeu ou une curiosité intellectuelle qui trouve sa victoire ou sa satisfaction dans l'exécution correcte du programme par l'ordinateur ;

— *connaissance du milieu professionnel futur*, grâce à des stages dans les entreprises et des visites, des conférences techniques ou professionnelles;

— *assurance d'un débouché professionnel* dans une activité qui est en pleine évolution et où les élèves peuvent développer leur personnalité ;

— *participation active et intervention des élèves* dans tous les actes fondamentaux de l'enseignement : orientation de l'enseignement, définition des programmes, choix des enseignants, méthodes pédagogiques, organisation de l'enseignement ;

— *qualité de la communication* entre enseignants et enseignés.

Quant aux méthodes d'enseignement, elles varient selon les matières, les enseignants et le niveau de maturité des élèves. Il n'y a pas de méthode générale, ni de recette, mais on doit en permanence rechercher celle qui convient le mieux « hic et nunc ». On trouvera :

— des cours magistraux pour les synthèses et les exposés véritablement magistraux ;
— des groupes de travail associés plus ou moins étroitement à des enseignants ;
— des travaux dirigés ;
— des travaux pratiques et des projets ;
— des enseignements isolés ou des équipes pédagogiques ;
— un projet collectif réalisé par toute une promotion avec répartition des tâches entre divers groupes ;
— etc.

ENSEIGNER LA PROGRAMMATION

Faut-il enseigner la programmation ? Cette question n'est pas aussi naïve qu'elle peut le paraître : combien de fois entend-on encore répéter que la programmation n'est pas une science, ni même une discipline autonome, mais simplement une collection de « trucs » et de recettes que l'on acquiert peu à peu par l'expérience ! Dans ces conditions, le seul « enseignement » qu'on peut dispenser consiste à introduire quelques-unes de ces recettes dans un stage d'initiation consacré pour l'essentiel à l'apprentissage détaillé d'un langage de programmation — Fortran, Cobol, PL/1, ou Algol en milieu universitaire, voire un langage d'assemblage — on insiste alors de préférence sur les caractéristiques les plus baroques.

Le résultat d'une telle attitude est probant : les programmes qui existent dans l'industrie sont le plus souvent lourds, de fiabilité incertaine, difficiles à comprendre, inefficaces, intransportables. Les programmes sont liés à un langage, parfois à un système ; il leur manque une méthode globale pour décomposer les problèmes. L'enjeu économique est énorme : qui n'a jamais eu affaire à l'un de ces « monstres », de▷

▷ces programmes gigantesques auxquels personne n'ose toucher de peur de provoquer un effondrement, et qu'il faut réécrire tous les 3 ans ?

En réaction contre cette attitude, l'enseignement dispensé à l'IIE cherche à insister dès le début de la première année sur l'originalité de la programmation considérée comme une discipline scientifique, possédant des méthodes et des techniques qui lui sont propres et (trait peut-être plus important encore) requérant tout un mode de pensée particulier. Cet enseignement, qu'il faut distinguer du pôle « langages de programmation/compilation » même si certains aspects sont évidemment communs, comprend :

- *un stage d'initiation de deux semaines,* dispensé « à chaud » lors de l'entrée des élèves en 1re année,
- *un enseignement de programmation,* réparti sur toute la 1re année et prolongeant ce stage,
- *une influence sur l'ensemble de l'enseignement en* 1re *année,* en particulier sur le projet traditionnellement appelé d'« assembleur ».

Ces différents cours seront présentés successivement ci-après. Ils sont encore pour la plupart dans une phase de mise au point, et l'on ne peut donc juger définitivement pour l'instant des résultats. Un point à noter est qu'une équipe d'enseignants guidés par les mêmes idées de base se dégage peu à peu ; elle comprend des professeurs et des assistants de l'IIE ainsi que des chargés de cours extérieurs et devrait contribuer à une certaine homogénéité dans l'enseignement des trois années.

INITIATION A LA PROGRAMMATION

Le stage inaugural de 15 jours prend les élèves à l'entrée à l'école, vierges de toute connaissance informatique (*).Il les initie aux fondements de la programmation, selon une méthode progressive. Notons que cette méthode, pour ne leur être pas identique, se rapproche de propositions faites ailleurs et maintenant bien connues : cours de Floyd et de Dijkstra, articles de Lecarme, expériences menées à Grenoble, à Toulouse, à l'EDF, etc. (*Voir bibliographie*).

Le but premier du stage est de montrer aux élèves, à l'occasion d'un certain nombre d'exemples concrets, la nécessité d'une démarche rigoureuse dans la décomposition des problèmes. L'enseignement est dispensé en petits groupes, à l'exception d'une séance introductive développant les concepts de base de l'informatique, et de quelques cours magistraux de synthèse. Au fur et à mesure que les exercices traités se compliquent, on introduit les principaux concepts : structures de contrôle (boucles, choix, enchaînement), procédures et modularisation des programmes, utilisation des variables, structures de données d'abord élémentaires puis complexes, etc. Dès les premières séances, l'accent est mis sur un certain nombre de principes méthodologiques : conception descendante, spécifications externes, utilisation d'assertions et de démonstrations de validité au moins informelles.

Dès les premiers jours de stage, les élèves « programment » effectivement et font « passer » des programmes, par l'intermédiaire des enseignants qui effectuent un contrôle préalable. En l'absence d'une telle expérience concrète, tout enseignement méthodologique reste en effet lettre morte. Par contre, il est important de persuader les étudiants, dès le début, qu'on ne « lance » pas un programme sans être convaincu de sa validité ; en particulier le temps de restitution ne doit pas être trop rapide.

Un élément à ne pas négliger lors d'une initiation est le langage de programmation choisi. Ce langage doit être seulement un support concret pour la réalisation, et non pas un cadre rigide qui gêne la pensée ; il est important que son utilisation n'exige pas des contorsions intellectuelles constantes.

Aussi l'emploi de langages truffés de scories conceptuelles, comme Fortran ou Cobol, ou de langages d'une complexité tératologique comme PL/1, est-il une lourde hypothèque pour qui veut « faire passer » un petit nombre de notions simples et fondamentales. Parmi les langages acceptables, on peut citer Pascal, Algol W, Algol 68, Simula 67, ou tout langage moderne équivalent, voire Lisp. Nous avons choisi Algol W, le premier en date des langages « structurés », qui présente l'avantage d'avoir été conçu spécifiquement pour l'enseignement, et dont le compilateur pour les IBM 360/370 possède d'exceptionnelles facilités d'aide à la mise au point. Ceci dit, nous sommes impuissants devant l'appellation « cours Algol W » qui s'est répandue dans l'Ecole pour caractériser le cours d'initiation, quelqu'attentifs que nous ayons pu être à insister sur le fait que le langage de programmation n'est pour nous qu'un outil.

(*) *Mis à part une dizaine d'élèves provenant d'IUT d'informatique. Il est capital pour la suite de l'enseignement qu'ils suivent aussi le stage.*

BIBLIOGRAPHIE

— *Dijkstra, E.W.: A Short Introduction to the Art of Programming, rapport EWD* 316, *The Eindoven* 1972.

— *Floyd, R.W. : Introduction to Programming and the ALGOL W Language, Stanford University* 1970.

— *Lecarme, O. : Introduction à la « bonne » Programmation : Bilan de quelques expériences ; Revue Informatique/Enseignement (AFCET),* 1976.

— *Lucas, Mossière et Scholl: Initiation à la Programmation - Réflexions et propositions - Revue bleue de l'AFCET, mars* 1975, *pp.* 5-25.

— *Meyer, B. : Initiation à la Programmation en Milieu Industriel, EDF-DER Note HI* 2011/01, 1976.

Le cours d'initiation a été dispensé sous cette forme pour la première fois en octobre 1975, sur un laps de temps écourté. Il est donc difficile de tirer des conclusions complètes. Les points suivants méritent cependant d'être notés :

— il semble que les élèves programment mieux que les années précédentes et sachent mieux aborder un problème nouveau :

— l'initiation à Fortran dispensé en deux jours, trois mois après le stage, n'a posé aucun problème (Fortran est utilisé pour des travaux pratiques d'analyse numérique et pour les stages en fin d'année) ;

— la connaissance d'Algol W par les élèves facilite la tâche des enseignants chargés des cours d'informatique générale, structures de données, systèmes informatiques, langages ;

— la répartition des élèves en petits groupes favorise l'initiative individuelle. Elle entraîne cependant des problèmes de coordination entre enseignants ; une meilleure répartition entre « petites classes » et conférences de synthèse doit être recherchée.

Les exercices ont dans l'ensemble intéressé les élèves ; nous avons cherché à éviter les sempiternels calculs de e et des racines du trinôme du second degré, et avons pu grâce à Algol W privilégier les manipulations de textes, tracés de dessin etc. Deux idées nous paraissent devoir être développées :

— *privilégier l'aspect « jeu »*, à la fois en faisant programmer la simulation de jeux simples, et en confiant à un groupe d'élèves la préparation des données pour un programme écrit par d'autres, et dont ils connaissent seulement la spécification externe ;

— *systématiser la fabrication d'« outils »*, ou de « mécanos » : par exemple programmes de dessin de figures de base, utilisés ensuite par des programmes de tracé de lettres, qui interviennent eux-mêmes dans des tracés de textes, etc.

Enfin, nous avons sans doute été trop timides jusqu'ici en introduisant la notion de spécification formelle, qui ne devrait pas présenter de difficulté pour des « taupins », et qu'il faut imposer dès le début de la scolarité si l'on veut avoir une chance de succès.

SUITE DES ENSEIGNEMENTS EN 1re ANNEE

Les élèves suivent en première année toute une série de cours faisant intervenir la programmation : structures de données, théorie des graphes, analyse numérique, programmation en langage d'assemblage, en Fortran et en Cobol. Ils n'avaient pas jusqu'ici de cours consacré à la programmation pour elle-même.

Plutôt que d'alourdir l'emploi du temps par un cours de plus, sans lien avec les autres, l'idée retenue et appliquée à partir de l'année 76-77 est de distribuer sur toute la première année, à raison d'une ou deux▷

Cet article a été présenté au groupe de travail ENSEIGNEMENT DE L'INFORMATIQUE DE L'AFCET

Le groupe Enseignement de l'Informatique de l'AFCET est un lieu de rencontre où sont évoqués et discutés les problèmes communs aux enseignants de notre discipline : secondaire, université, MIAG, IUT, écoles d'ingénieurs, formations continue et permanente.

De nombreux efforts sont réalisés en de nombreux endroits pour enseigner les différentes facettes de notre discipline. Les expériences réalisées ici et là, échecs et succès, n'ont qu'une diffusion locale et les contacts sont souvent difficiles à établir de façon régulière. La vocation de l'AFCET est de fournir ce cadre.

L'existence du groupe est donc liée aux activités qu'il supporte. La liste des centres d'intérêts, fournie ci-dessous, est délibérément large, de manière à susciter, à l'intérieur même du groupe, la formation de sous-groupes rassemblant les membres spécialement intéressés par certains thèmes particuliers, dont le Syllabus 75 fournit une liste non exhaustive évidemment. Formation et disparition de sous-groupes sont dictées par l'évolution de la composition du groupe, l'achèvement de travaux...

Dans son ensemble, le groupe se réunit une ou deux fois par an pendant 2 à 5 jours, les sous-groupes ont évidemment des fréquences de réunion laissées à leur libre choix.

Les activités du groupe se concrétisent sous forme de :
— réunions de travail ;
— publication d'un bulletin, organe d'expression du groupe.

Pourquoi un tel journal ?

Une première motivation est d'apporter aux enseignants un moyen d'expression. Beaucoup d'efforts sont consacrés à l'enseignement de l'Informatique, efforts trop obscurs et qui n'ont autre diffusion que celle des relations personnelles.

Une autre motivation est la diversité des problèmes ; enseignements de masse (Deugs par exemple), enseignements de gestion, enseignements au niveau secondaire, enseignements spécifiques (système, compilation, etc.). Chaque centre éducatif a développé une solution qui lui est propre.

Enfin s'il est sain que les enseignements possèdent une certaine évolution relativement aux pratiques industrielles, il est essentiel de faciliter l'adaptation des étudiants au monde du travail, donc de maintenir le dialogue avec les employeurs.
— organisation de conférences plus larges ;
— organisation de concours réservés aux étudiants ;
— etc.

▷séances par type d'application, un séminaire de programmation ou aucun problème nouveau n'est introduit, mais où les problèmes soumis aux élèves en analyse numérique, théorie des graphes, etc., sont étudiés et discutés du point de vue de la méthodologie de la programmation : décomposition des problèmes, méthodes d'affinage, comparaison entre les différents algorithmes possibles et leur complexité, structures de contrôle et structures de données adéquates, récursion, problèmes d'expression, lisibilité, démonstrations de validité...

Nous attendons beaucoup de cet enseignement pour conserver et enrichir l'acquis du stage d'initiation, attirer l'attention des élèves sur certains grands problèmes et les initier à quelques techniques de base de l'informatique tout en rapprochant l'enseignement de la programmation de celui des disciplines d'application.

LE PROJET DE « MINI COMPILATEUR »

A la suite du cours d'initiation aux langages d'assemblage, étudiant plus particulièrement celui de l'IBM 360/370, les élèves doivent réaliser un projet d'écriture d'un mini-assembleur. Il a été remplacé en 1975-1976 par un projet d'écriture de compilateur pour un mini-langage dont la sémantique est très simple (instructions d'affectation, de branchement conditionnel, d'impression et d'arrêt ; objets de type entier et chaîne de caractères) mais dont la syntaxe est semblable à celle d'un langage de haut niveau. Le compilateur devrait être écrit en Algol W ; le projet était divisé en 15 « sujets », chaque groupe d'élèves devant choisir deux sujets, et chaque sujet étant pris par au moins deux groupes afin d'assurer l'aboutissement de l'ensemble. Les relations entre les « sujets » étaient données sous la forme de notes polycopiées, comprenant en particulier des programmes Algol W faisant ressortir la hiérarchie des différentes procédures, l'organisation de leurs appels mutuels et la structure des données partagées.

Ce projet a été un échec : si tous les groupes ont rendu un rapport convenable, voir brillant pour quelques-uns, et si un petit nombre ont abouti à des procédures Algol W correctes, le résultat est très loin d'un compilateur utilisable.

Cet échec est dû pour une part, sans aucun doute, aux enseignants : il y a eu un défaut d'organisation initial, et quelques imprécisions dans les spécifications, certes mineures, mais graves dans un projet de cette ampleur. Nous avons en outre commis l'erreur de ne pas prévoir dès l'origine un ou plusieurs « sujets » à part entière consistant à construire des environnements de test pour le reste des sujets.

Ceci dit, le projet a fait aussi ressortir un certain nombre de points intéressants quant à l'attitude des élèves. Chaque « sujet » conduisait à un module de programmation fort simple, de l'ordre de moins d'une page de programme. La difficulté essentielle, la seule en vérité une fois le sujet compris, était l'« interface », la communication avec les autres groupes. Des blocages psychologiques étonnants se sont révélés dans ce domaine : des étudiants venaient demander aux enseignants de leur fournir les spécifications mises au point par leurs camarades ; pressés de s'adresser directement aux intéressés, ils nous avouaient leur impossibilité totale de communiquer, qu'ils attribuaient généralement aux années de taupe, à la mentalité « bien française » etc. Nous avons là encore commis une erreur en suggérant, et en n'imposant pas, une « infrastructure de communication » comprenant un panneau d'affichage, des réunions régulières, etc.

Si nous insistons sur cet enseignement plus ou moins manqué, ce n'est pas pour le plaisir de relater une expérience béhavioriste : c'est que les problèmes qu'elle met en lumière sont au cœur même de la méthodologie de la programmation en milieu industriel, où les difficultés sont pour une grande part des difficultés de communication. Nous recommencerons cette expérience, en visant cette fois le succès grâce à une organisation plus méticuleuse, mais c'est sans aucun doute à tous les niveaux de l'enseignement de l'informatique que devraient être systématiquement développées les méthodes de programmation (ou plutôt de « multiprogrammation ») en équipe.

Claude Kaiser
Bertrand Meyer
Etienne Pichat

Claude Kaiser, ingénieur de l'Ecole Polytechnique (1957) *et de l'ENSGM* (1968), *docteur es sciences, ingénieur en Chef du Génie Maritime (CR) est né le* 5/02/1938.
Ingénieur au Service Technique des Constructions et Armes Navales (STCAN) de 1962 *à* 1968 *et à la DRME de* 69 *à* 70, *ingénieur de Recherche à l'IRIA de* 1971 *à* 1974, *il est maître de conférences au CNAM et à l'IIE depuis* 1974. *Avec une formation et une activité professionnelle tant en automatique qu'en informatique, il s'intéresse aux systèmes informatiques et plus particulièrement aux systèmes d'exploitation et aux systèmes en temps réel.*

Bertrand Meyer, 21-11-1950, *ingénieur de l'Ecole Polytechnique* (1972) *et de l'ENST* (1974). *Master of Sciences de l'université de Stanford (Californie), ingénieur-chercheur à EDF (Direction des Etudes et Recherches) depuis octobre* 74, *est chargé de cours à l'IIE.*

Photo Roger-Viollet

Etienne Pichat, ingénieur des Arts et Manufactures, docteur es sciences et licencié es sciences économiques, professeur au CNAM et maître de conférences à l'Ecole Polytechnique, dirige l'Institut d'Informatique d'Entreprise. Spécialiste en algorithme non numérique, il s'intéresse à la méthodologie d'analyse et de conception des systèmes informatiques, notamment des banques de données.