Un bilan pédagogique de l'Université Pierre et Marie Curie pour reconvertir à l'informatique des enseignants chercheurs

Jean-Pierre Bénéjam, Titou Durand

UN BILAN PÉDAGOGIQUE DE L'EXPÉRIENCE DE L'UNIVERSITÉ PIERRE-ET-MARIE-CURIE POUR RECONVERTIR À L'INFORMATIQUE DES ENSEIGNANTS-CHERCHEURS

Jean-Pierre BÉNÉJAM et Titou DURAND

Résumé : Nous dressons ici un bilan de l'expérience de formation à l'informatique d'enseignants-chercheurs issus de disciplines variées, essentiellement scientifiques. Conçue et mise en œuvre en 1982, à l'Université Pierre-et-Marie-Curie (PARIS 6), avec un encadrement d'informaticiens — dont les auteurs —, cette expérience a été couronnée de succès, et a permis de compenser la pénurie d'enseignants d'informatique, en informatique même, mais aussi dans les autres disciplines. Certains stagiaires ont opéré une conversion thématique complète et travaillent actuellement dans des laboratoires d'informatique. Compte tenu de l'hétérogénéité du public à former, il nous a fallu affiner nos méthodes pédagogiques.

1. CADRE DE L'EXPÉRIENCE

De 1982 à 1989, l'Université Pierre-et-Marie-Curie (UPMC) a proposé aux enseignants-chercheurs de l'UPMC d'abord, et de la (grande) région parisienne ensuite, un stage annuel, dit de *reconversion à l'informatique*. Ce stage avait pour objectif de fournir une formation de base permettant aux stagiaires d'envisager éventuellement une reconversion totale en informatique. Pour cela on voulait les amener au niveau des connaissances nécessaires pour suivre une maîtrise d'informatique l'année suivante.

Cette opération, les résultats obtenus, ainsi que l'expérience de l'un (parmi la centaine) de nos stagiaires, ont été exposés à Besançon [Bénéjam & Poc-Paget] en novembre 1988. Nous nous attacherons ici à en présenter un bilan pédagogique.

Nous indiquons, dans un premier temps, l'origine disciplinaire et les motivations initiales des stagiaires car, outre un intérêt intrinsèque, la modification de ces données a impliqué certains changements dans l'organisation, le contenu et la didactique des stages.

1.1. Origine des stagiaires

L'origine disciplinaire des stagiaires a évolué avec les années : les deux premières années, les stagiaires étaient, pour plus de la moitié d'entre eux, des physiciens, la plupart des autres étant mathématiciens ou chimistes. Les années

suivantes, un grand nombre de stagiaires étaient issus des sciences de la vie et les groupes étaient beaucoup plus hétérogènes quant à l'origine.

Les connaissances initiales en informatique des stagiaires ont aussi changé lors des différentes promotions : la première année, pratiquement tous les stagiaires avaient déjà une certaine pratique de l'informatique (ne serait-ce qu'à travers un traitement de texte) et, au fil des ans, de plus en plus de stagiaires sont arrivés sans aucune connaissance.

1.2. Motivation des stagiaires

On retrouve aussi les deux périodes pour la motivation initiale des stagiaires : les deux premières années les stagiaires visaient, pour plus de la moitié d'entre eux, une reconversion totale à l'informatique alors que cela n'était le cas que pour une faible minorité les années suivantes.

2. PROGRAMME DU STAGE

Le stage annuel de 325 heures, que nous avons mis en place, comprend lors des deux premières années :

- 50 heures de cours, par Jean-Pierre Bénéjam et Titou Durand, dans le module « *Logique et programmation* » de la licence d'informatique,
- 50 heures de cours, par André Bernardy, dans le module « *Microprogrammation et microprocesseurs* » (moitié du certificat d' « *Informatique appliquée* » dont est responsable Françoise Madaule et qui est ouvert aux étudiants de l'UPMC des disciplines scientifiques autres que l'informatique),
- 125 heures de séances de rattrapage, de compléments de cours (graphes et algorithmique, notions sur les systèmes, etc.), et de travaux dirigés,
- heures de travaux pratiques sur des micro-ordinateurs.
- Ces 325 heures d'enseignement ont été complétées par des conférences (par exemple, 10 heures, par Jean-François Perrot, en juin 1983, sur les gros systèmes informatiques) et par la présentation des activités des laboratoires de recherche de l'Institut de Programmation.

Dès la seconde année, afin de rapprocher le langage des enseignants de celui des enseignés, et avec la volonté de favoriser l'inscription de stagiaires issus des sciences de la vie, nous avons demandé à Robert Bellé, biologiste, stagiaire de la première année (et qui ne se destinait pas à une reconversion totale) d'assurer, en binôme avec l'un d'entre nous pour la première année et seul ensuite, les travaux pratiques. Nous pensons que cela a été un facteur primordial de la réussite des stages ultérieurs. Il nous paraît en effet essentiel d'avoir, dans l'équipe enseignante, une personne capable de faire le lien entre les informaticiens et les stagiaires : il ne suffit pas de prendre un ancien stagiaire, encore faut-il qu'il reste dans sa discipline d'origine. En effet, les stagiaires qui ont effectué une reconversion totale, s'ils ont,

bien sûr, conservé les connaissances de leur discipline d'origine et le langage de cette discipline lorsqu'ils discutent avec d'ex-collègues, ont parfaitement intégré le langage informatique dès qu'ils parlent d'informatique : ils ne sont donc plus capables, sans un effort très important, de jouer ce rôle de lien dans un enseignement d'informatique destiné à d'ex-collègues. A contrario, le langage de Robert Bellé est toujours celui d'un biologiste (il est devenu professeur de biologie depuis lors) même s'il est parfaitement capable de comprendre (et de parler à) un informaticien.

A partir de la troisième année, un autre changement important a été le remplacement du cours de la licence d'informatique par un cours propre intégré aux séances de travaux dirigés. D'abord, cela a permis une concentration des horaires (ce qui était nécessaire pour les collègues des universités autres que celles de Jussieu). Mais, surtout, il existait un fossé de plus en plus grand (à cause de l'origine et de la motivation des stagiaires) entre les objectifs, le contenu et le langage du cours de licence et ceux de la partie strictement « stage de reconversion » : on ne peut pas enseigner de la même façon à des étudiants sortant du Deug et se destinant à une profession informatique et à des collègues n'ayant pas fait de mathématiques depuis fort longtemps (et même pour certains étant restés sur un échec dans cette discipline) et qui veulent utiliser l'informatique dans leur discipline ou l'enseigner à des étudiants de leur discipline.

3. OBJECTIFS

Un premier objectif nous est apparu comme un préalable : comme il n'est pas raisonnable, d'un point de vue financier, d'élaborer une formation complète propre aux « recyclés », le programme a été conçu pour amener graduellement les stagiaires au niveau atteint par les étudiants de formation initiale titulaires de la licence d'informatique (sans algorithmique numérique ni probabilités) afin que les stagiaires puissent poursuivre leur formation en maîtrise d'informatique.

Dans le premier cours, il s'agit d'étudier les concepts et les méthodes de la programmation. Nous développons un peu les objectifs et le contenu de ce cours dans le paragraphe suivant.

L'autre cours concerne l'architecture et le fonctionnement d'un ordinateur, la programmation des interfaces pour l'instrumentation, etc. On pourra trouver dans [Bernardy, Auban & Boudin] une étude de cette partie.

4. CONTENUS DU COURS SUR LA MÉTHODOLOGIE DE LA PROGRAMMATION

En concevant le programme de ce cours, nous avions à l'esprit deux points qui nous paraissent fondamentaux :

- la nécessité que les stagiaires aient une pratique, et de partir de cette pratique. Cela explique l'importance des travaux pratiques et aussi la volonté que les travaux dirigés soient réellement des travaux dirigés, c'est-à-dire des périodes où l'enseignant ne donne pas (presque) immédiatement la solution mais guide, le plus individuellement possible, les apprenants.
- la nécessité qu'ils acquièrent les concepts et les méthodes de la programmation. Cela est vrai pour tout apprenant en informatique, mais c'est crucial pour de futurs enseignants d'informatique. Par exemple, nous ne sommes pas persuadés qu'il faille enseigner les invariants de boucle en Deug (nous pensons plutôt que non) ; par contre nous sommes convaincus qu'il faut maîtriser ce concept pour enseigner en Deug où on peut exiger des étudiants qu'ils fournissent un index des variables qu'ils utilisent. Cela veut dire que de futurs enseignants doivent avoir une réflexion poussée sur les concepts et les méthodes puisqu'il ne s'agit pas de les utiliser tels quels mais de les réinvestir dans un processus didactique.

Styles de conception de programmes :

Il nous a semblé intéressant que les stagiaires voient deux styles de conception de programmes : l'impérative et l'applicative [1]. Non seulement ce sont des modes de pensée différents que tout informaticien doit connaître, mais, surtout, cela permet d'avoir plusieurs points de vue sur les différents concepts tout en facilitant leur introduction [2].

Principaux concepts introduits :

Hormis les concepts classiques (on pourra consulter [Arsac], [Derroite & Le Charlier] et [Rogalski] par exemple) de la programmation impérative (variable informatique, alternative, itération...), nous insistons beaucoup sur les concepts suivants :

- **Spécification - implantation** : le principal concept que nous voulions inculquer aux stagiaires est celui de spécification (que l'on ne peut pas faire comprendre sans parler du concept dual d'implantation). C'est un concept qui peut être introduit lors des analyses rigoureuses des problèmes à résoudre, qui peut être explicité en programmation impérative à propos des assertions d'entrée et de sortie, mais l'expérience prouve qu'il est nécessaire de l'étudier avec plusieurs points de vue. C'est une des raisons qui nous ont poussés à étudier aussi la conception applicative des programmes. On peut remarquer que ce concept est un prérequis indispensable à l'étude des concepts suivants.

---

1. Nous parlons plutôt de style de conception de programmes et non de style de programmation car, dans les deux cas, le programme final est écrit en Pascal UCSD.

2. Cela plaiderait aussi pour une introduction de la programmation déclarative. Nous y avons renoncé pour trois raisons : le manque de temps, la programmation déclarative alors enseignée en maîtrise (et pas en licence) et la nécessité d'utiliser, pour la mise en œuvre, un autre langage de programmation que celui que nous utilisions pour les deux autres styles.

- **Procédure - fonction** : alors que la notion de sous-programme est relativement simple, il n'en est pas de même pour les notions de fonction (clairement une vision applicative de la programmation permet d'approfondir cette notion) et de procédure. On peut d'ailleurs remarquer que, dans la plupart des langages, la différence entre fonction et procédure est avant tout une différence d'utilisation et non de concept. Par exemple, en Pascal, on ne peut pas utiliser de fonction lorsque le type du résultat n'est pas « simple » et des paramètres de fonction peuvent être des résultats ; en Ada, le résultat d'une fonction ne peut pas être un produit cartésien et il n'est pas complètement interdit de modifier une variable globale à l'intérieur d'une fonction.
- **Récursivité** : c'est la raison primordiale de l'introduction de la conception applicative. L'expérience a en effet prouvé que cette notion, réputée difficile, est relativement facilement assimilée en conception applicative dès que les apprenants maîtrisent les concepts de spécification et d'implantation d'un problème à l'aide de sous-problèmes.
- **Efficacité** : qui dit programmation dit efficacité. La réalisation de programmes efficaces doit être un souci constant pendant l'apprentissage de la programmation et cela même sans l'étude des techniques d'évaluation de complexité, étude que nous ne faisions pas car elles demandent trop de connaissances en mathématiques.
- **Type de données** : nous retrouvons ici la dualité spécification - implantation. On peut remarquer que cette notion a pris beaucoup d'importance avec la conception orientée objet.

Ordre de l'introduction des styles de conception :

On ne peut pas introduire de façon concomitante les deux styles de conception (nous avons fait l'expérience dans un autre cadre, ce ne fut pas une réussite, les étudiants mélangeant les deux styles de façon anarchique). Nous avons essayé les deux ordres possibles et nous ne savons toujours pas quelle est la meilleure solution. Le problème n'est pas tant de savoir quelle méthode est le plus facilement assimilable en premier (à notre avis les difficultés sont à peu près équivalentes), mais de savoir quel est l'ordre qui permet d'avoir un meilleur résultat global : lorsque l'on commence par la conception itérative, les apprenants ont plus de difficultés à concevoir des algorithmes récursifs car il leur est plus difficile d'abandonner une vision itérative de la récursivité ; lorsque l'on commence par la conception applicative, de nombreux apprenants (en général ceux qui ont le mieux intégré cette vision de la conception de programmes) peuvent éprouver beaucoup de difficultés en conception impérative.

5. CONSOLIDATION DU STAGE

Bien sûr, nous ne considérons pas que 325 heures suffisent pour former des informaticiens. Aussi les stagiaires ont poursuivi leur formation les années

suivantes en consolidant les connaissances acquises lors du stage et en acquérant de nouvelles.

Les stagiaires ont pu bénéficier d'une décharge d'un demi-service d'enseignement pour poursuivre leur formation, généralement en maîtrise d'informatique (directement en troisième cycle pour quelques uns d'entre eux). La plupart des stagiaires qui se sont engagés dans une reconversion totale ont obtenu un DEA, voire un DESS pour trois d'entre eux.

Mais nous voudrions insister ici sur la formation par l'enseignement : afin d'asseoir les connaissances acquises dans cette année de stage, nous avons incité (fortement) les « recyclés » à enseigner ces connaissances (en Deug, et même pour certains en licence). Il n'était pas question de « lâcher dans la nature » des enseignants pas encore entièrement formés : aussi avons-nous cherché des équipes d'enseignants actives, constituées autour d'informaticiens confirmés. De plus, les stagiaires eux-mêmes ont souvent préféré avoir une année de transition avant de prendre en charge un groupe de travaux dirigés : ils n'assuraient que des « doublements » de travaux pratiques (en Deug, les séances de travaux pratiques sont encadrées par deux enseignants, celui qui assure les travaux dirigés et une « doublure »). On doit remarquer que ces enseignants (d'une autre discipline) ont généralement été d'excellents enseignants (d'informatique), le fait d'avoir récemment surmonté eux-mêmes les difficultés les ayant placés dans une position favorable pour mieux comprendre et aider les étudiants [3].

Certains d'entre eux ont pu, à l'issue de ce stage, s'engager dans des activités de recherche en informatique *au sein d'une équipe de recherche dans un laboratoire d'informatique*. Dans le cas d'une reconversion partielle, ils ont conservé leur activité de recherche dans leur discipline, profitant de leur savoir en informatique, et, soit participent aux enseignements d'informatique dans les cursus d'informatique, soit sont maîtres d'œuvre d'un enseignement d'informatique dans leur domaine d'origine, mettant à profit la (deuxième) *compétence pédagogique* ainsi acquise.

6. CONCLUSION

Cette action de formation a connu peu d'échecs, le plus souvent sur abandon. Nos « recyclés » se sont bien adaptés — même ceux qui débutaient en informatique —, en apportant beaucoup de travail personnel. Hormis les reconversions totales qu'elle a entraînées, elle a également favorisé la rénovation des filières traditionnelles et la mise en place de filières à double compétence. Elle a favorisé surtout la rencontre interdisciplinaire d'enseignants-chercheurs, solidaires pour atteindre le même

---

[3]. Loin de nous l'idée que les meilleurs enseignants sont les enseignants récemment formés, mais la conjonction enseignant confirmé d'une autre discipline, nouvel enseignant en informatique et équipe active d'enseignement semble donner d'excellents résultats. Nous ne pensons pas , non plus, que le Deug soit le meilleur cadre pour un premier enseignement, mais c'est, à l'UPMC, le seul lieu où il y a des "doublures" en TP.

objectif, réalisant tous que l'informatique, au-delà de l'outil puissant qu'il constitue, est aussi une science.

Le programme d'un tel stage doit être à la fois théorique et très pratique, dégageant les concepts au fur et à mesure à partir d'exemples simples, mais traités de manière approfondie. Il semble essentiel que son enseignement soit assuré par une équipe constituée d'informaticiens professionnels et d'anciens stagiaires gardant leur ancienne discipline comme activité principale. La formation des stagiaires doit se poursuivre non seulement par la recherche — indispensable même pour les reconvertis partiels — mais également par l'enseignement qu'ils doivent nécessairement pratiquer ensuite, bien intégrés à des équipes pédagogiques : clarifiant leurs connaissances, ils peuvent ainsi totalement assimiler les notions acquises et en faire profiter leurs étudiants.

Jean-Pierre BÉNÉJAM et Titou DURAND,

Laboratoire d'Informatique Fondamentale
Université Pierre-et-Marie-Curie (PARIS 6)
Tour 55, boîte 166
4 Place Jussieu, 75252 PARIS-CEDEX 05

Télécopie : **44 27 62 86**
Adresses électroniques : **durand@frcirp81**
ou **rdu@ccr.jussieu.fr**

7. BIBLIOGRAPHIE.

[ARSAC] *Les bases de la programmation,* Dunod, 1983 ;

[BÉNÉJAM] *L'informatisation des enseignants du supérieur : reconversion ou requalification ?* , Symposium « Technologies nouvelles et formation », Salon EDUCATEC 83, Paris, 14 décembre 1983 ;

[BÉNÉJAM & POC-PAGET] *La reconversion à l'informatique des enseignants-chercheurs du supérieur : l'expérience de l'Université Pierre-et-Marie-Curie*, Journées de la SPECIF, Besançon, 17 et 18 novembre 1988, p.125-133 ;

[BERNARDY, AUBAN & BOUDIN] « Apprentissage par l'ordinateur des concepts informatiques utilisés dans les systèmes à base de microprocesseurs pour la conduite de procédés ou de robots », *Deuxième colloque international sur la robotique pédagogique*, Montréal, 22 - 24 août 1990 ;

[DELOITTE & LE CHARLIER] « Un système d'aide à l'enseignement d'une méthode de programmation », *Premier Colloque Francophone sur la Didactique de l'Informatique*, Paris, EPI, 1988 ;

[ROGALSKI] « Acquisition de structures conditionnelles », *Annales de Didactique et de Sciences Cognitives*, IREM de Strasbourg, 1988.