Mémoires d'un programmeur du bâtiment 8 de Rocquencourt dans les années 1970-1980

Gérard Huet
Inria Paris Center

Je suis arrivé au "Bat 8" en septembre 1972. Je venais de finir mon PhD dans une Université américaine dotée d'un PDP-10 relié au réseau Arpanet. Cet ordinateur haut de gamme de la société DIGITAL équipait les départements d'informatique et centres de recherche en intelligence artificielle des principales universités américaines. Même le centre de recherches PARC de XEROX de Palo Alto en avait un - ils avaient contourné le refus de leur administration d'en faire l'acquisition en construisant un clone à partir de composants. Ces machines étaient équipées d'un OS permettant le temps partagé, et le réseau Arpanet avait un protocole de transferts de fichiers, ancêtre de ftp, qui permettait de télécharger notamment Stanford LISP depuis une plate-forme libre DECUS des clients de DIGITAL. C'est ainsi que pour mon MS j'avais implémenté en LISP un système de démonstration automatique PETER proposant toutes les variantes du principe de résolution en logique de premier ordre avec égalité.

Les moyens de calcul de l'IRIA de l'époque consistaient essentiellement en un 10070 de la société CII. La CII et l'IRIA étaient les deux mamelles du Plan Calcul. Les politiques ne comprenaient pas le principe de la recherche préparant l'innovation au sein d'un même organisme, et avaient créé une paire de canards boiteux, l'un privé, l'autre public, en fait en concurrence. Ils essayaient un bootstrap improbable, en dotant l'institut de recherches des machines du constructeur. Le 10070 était en fait la Sigma 7 de la société XEROX construite sous license. Comme le dit Wikipedia: "Le Sigma 7 est un des premiers ordinateurs de 3e génération, d'architecture élégante mais sans logiciel".

En fait, le premier projet d'envergure des premiers informaticiens de l'IRIA avait été le projet ESOP de doter le 10070 d'un OS multitâches. Mais la CII tarvaillait en concurrence sur cet objectif, et a exigé que le projet ESOP soit arrêté. En 1972, quand je suis arrivé à l'IRIA, les chercheurs ayant travaillé d'arrache-pied pendant 4 ans, désabusés, poussaient néanmoins

la conscience professionnelle jusqu'à rédiger la documentation d'une suite de logiciels destinés à la benne.

C'est ainsi qu'en 1972 l'ordinateur 10070 de Rocquencourt fonctionnait en batch sur des crénaux horaires qui devaient être réservés. A l'heure dite, il fallait convaincre l'équipe du créneau précédant de vous laisser la place au lecteur de cartes. A l'inverse, il fallait vite récupérer son listing à l'imprimante avant que l'équipe du créneau suivant n'arrive. J'ai assisté à des prises de contrôle d'autorité violentes.

Pour ma part, je mettais au net l'article sur mon algorithme d'unification dans le lambda-calcul typé, qui sera publié dans le premier numéro de Theoretical Computer Science en 1975. A cette fin, je voulais implémenter l'algorithme, et l'expérimenter sur les différents exemples que je présente dans le papier. J'avais donc besoin d'un ordinateur muni d'un OS et d'un compilateur d'un langage se prêtant au calcul symbolique. J'ai trouvé cet ordinateur à Grenoble, sous la forme d'un IBM 360-91. Une ligne à 150 bauds y était reliée depuis un cagibi sans fenêtre du bâtiment 8. Un autre groupe avait obtenu cette liaison pour faire de l'APL. Une machine IBM à boule servait de console. C'est mon collègue Gilles Kahn qui m'indiqua la manip. Il suffisait de remplacer la boule APL par une boule ASCII ordinaire, de taper un mot de passe et on pouvait travailler en temps partagé - à 150 bauds.

Cette machine avait notamment un compilateur pour le langage ALGOL-W de Niklaus Wirth. Ce langage combinait Algol 60 avec des structures de données gérées dynamiquement, comme LISP, par un garbage collector. Pour mon projet, ce langage était idéal. Son défaut principal était l'absence de modules, et le compilateur s'étouffait si votre programme excédait 5000 lignes. Mon programme final de Septembre 1974 comporte 3130 lignes. J'ai gardé une copie du mince papier du télétype IBM, et 50 ans plus tard, avant que l'encre ne s'estompe définitivement, j'ai réussi à le scanner sous une forme encore lisible, dont le pdf est disponible à l'URL: UNIF J'ai donné ses conditions d'utilisation, et l'exemple d'une exécution, à l'URL README.

En 1974 on installa un IRIS 80, le successeur du 10070, à Rocquencourt, équipé d'un systême multi-tâches. La mémoire était de 4MB de mots de 32 bits. Gilles Kahn sous-traita à un ingénieur Toulousain l'écriture d'un compilateur Pascal. On décida alors, Gilles et moi, de démarrer un projet d'éditeur structuré qui sera appelé Mentor.

Pendant plusieurs mois, Gilles et moi avons discuté de la conception du système. Mais nous proscratinions à écrire du code. Un jour, en arrivant au bureau, je vois Gilles accroupi par terre, avec de grands ciseaux, en train de découper un gros listing. Il m'explique que c'est le listing du compilateur

Pascal de Wirth, écrit en lui-même en Pascal. Il était en train d'enlever à coups de ciseaux tous les endroits où le compilateur émettait du code, pour les remplacer par un générateur d'arbres de syntaxe abstraite. Une fois le découpage effectué, on a juste recopié le code, en ajoutant les instructions de construction d'arbre. En quelques heures, on a obtenu un programme exécutable, capable de construire une forme structurée d'un programme Pascal arbitraire. Je dois dire que j'ai été soufflé par ce procédé brutal mais efficace. Gilles a conceptualisé le procédé, qu'il appelait la programmation par crassussage.

Bernard Lang et Véronique Donzeau-Gouge complétaient l'équipe, et même Jean-Jacques Lévy s'est joint à nous quelques temps. La mémoire de l'IRIS 80 étant trop petite pour charger tout l'exécutable, on a dû recourir à l'overlay pour compenser ce handicap. On est ainsi arrivé à un prototype d'éditeur structuré spécialisé au langage Pascal, similaire à l'éditeur Interlisp développé alors par Xerox PARC. Tout cet effort s'arrêta net quand les équipes de maintenance de l'IRIS 80 firent une fausse manip en montant les ventilateurs de refroidissement à l'envers, qui fit se déposer sur les circuits la poussière accumulée depuis le démarrage de la machine. Il fallut tout nettoyer au coton-tige, et tout effort de programmation a dû s'arrêter pendant près d'un an. Pour la petite histoire, les équipes de la CII à Louveciennes, de l'autre côté de l'autoroute, diagnostiquaient l'IRIS 80 à Rocquencourt par une liaison téléphonique établie sur les anciens cables de l'OTAN, reliant le Shape Headquarters de Louveciennes à la base militaire américaine de Rocquencourt. Ces tunnels passaient dans un tunnel secret sous l'autoroute, dans une autre circonscription téléphonique, incognito des PTT, en toute illégalité!

Sur la photo en Figure 1 (©David MacQueen, 1976), je suis au clavier éditant le source de Mentor. Gilles fait le kibbitzing. C'est le seul programme dans ma carrière que j'ai écrit en tandem. Nous nous entendions très bien, Gilles et moi, et travailler avec lui était un vrai plaisir. Simplement, il fallait veiller à ses affaires, genre stylo, que Gilles aimait "chibrer", c'est-à-dire tordre entre ses mains.

J'ai gardé un listing de Mentor Version 3.3 du 14 décembre 1978 (38*20*4 cm).

Mentor fut suivi de Centaur, développé en commun par les centres de Rocquencourt et de Sophia-Antipolis au sein du projet CROAP (Conception et Réalisation d'Outils d'Aide à la Programmation).

Lors du décrassage de l'IRIS 80, j'ai décroché progressivement de l'effort Mentor pour me consacrer à la réécriture. A l'époque, les mieux équipés parmi les chercheurs informaticiens français étaient ceux de l'équipe d'intelli-

Figure 1: Gérard Huet et Gilles Kahn

gence artificielle de Patrick Greussay à l'Université Paris-Vincennes. Ils squattaient le PDP 10 de l'IRCAM. Boulez avait exigé à la création de l'IRCAM de s'équiper en toute indépendance des chimères du plan calcul, et avait ainsi réussi à obtenir ce PDP-10 alors que les chercheurs de l'IRIA en étaient réduits à regarder un IRIS 80 obsolète à l'arrêt. Seuls trois mandarins ont réussi à obtenir un PDP-10 à l'époque, à l'IRCAM, à l'hôpital de la Pitié-Salpêtrière, et à l'Ecole des Mines de St Etienne.

Un soir, j'ai décidé d'inspecter le repaire de Greussay, et je suis allé de nuit visiter les sous-sols de l'IRCAM. Dans une grande salle machine à la lumière blafarde, des anarchistes hirsutes pioanotaient sur leur clavier. Je regarde par dessus l'épaule d'un jeune barbu, et je découvre sur son écran la liste de son espace de travail courant, avec deux fichiers LISP intitulés "HUET_76.L" et "HUET_78.L". Intrigué, je lui demande ce qu'il fait. Il m'apprend qu'il est étudiant de mon cours de DEA d'Orsay, et qu'il étudie la différence entre les structures de données de termes de premier ordre que j'enseigne en cours, et celles définies dans ma thèse d'Etat. Cet étudiant s'appelait Jean-Marie Hullot, et j'allais faire avec lui le système KB de réécriture algébrique, développant des variantes de la méthode de Knuth et Bendix.

Jean-Marie Hullot était le disciple de Jérôme Chailloux, auteur de VLISP-10, l'implémentation du LISP de l'Université de Vincennes sur PDP-10. Jérôme était un hacker de rang international - il avait accès à tous les centres de calcul. Lorsque je suis parti en sabbatique au Stanford Research Institute (SRI) en 1980, emmenant Jean-Marie avec moi, Jérôme vint en personne installer VLISP sur les PDP-10 du SRI et de l'AI Lab de Stanford. Jean-Marie embarqua chez lui un terminal et un modem, et développa KB à son aise, il ne se montra plus au labo. Le matin, il préemptait un "login slot" (il n'y en avait que 10 pour tout le labo) qu'il ne relâchait que le soir. Les chercheurs du labo, qui se disputaient les accès pour remplir leurs obligations contractuelles à temps, ne comprenaient pas qu'un jmh qu'ils n'avaient jamais vu était incrusté à perpétuité.

Jean-Marie Hullot est l'un des programmeurs virtuoses les plus talentueux que j'aie jamais rencontrés. Ses programmes étaient élégants, corrects, et efficaces. Le tour de force fut d'implémenter les structures de termes dans les algèbres munies d'opérateurs commutatifs et associatifs. Le chemin avait été ouvert par Mark Stickel, avec l'invention d'un algorithme d'unification de termes modulant des congruences. L'unificateur principal des termes libres était remplacé par un ensemble fini d'unificateurs modulo, calculés à partir d'une base d'équations diophantiennes linéaires. Pour réduire l'espace de recherche de cette base, j'ai inventé une borne origi-

nale au problème, qui a été reconnue comme résultat arithmétique nouveau. Jean-Marie Hullot a fait une implémentation fine du calcul des unificateurs, utilisant la structure d'arbres binomiaux de Jean Vuillemin. Nous étions la seule équipe au monde à posséder un logiciel de réécriture muni de telles fonctionnalités. Pour sa thèse, Jean-Marie Hullot établit le système équationnel correspondant aux mouvements du robot de Poppelstone de l'Université d'Edimbourg, et construisit par KB les formes canoniques de ses déplacements.

Le résultat donnant une borne au calcul des bases d'équations diophantiennes fut présenté comme une note établissant la correction d'un programme Pascal BASIS. Cette note parut dans Information Processing Letters en 1978.

A notre retour au bat 8 en 1981, Jean-Marie Hullot, ayant terminé KB et sa thèse, fut embauché par l'IRIA dans l'équipe naissante de Jean Vuillemin, au projet VLSI. Une nouvelle époque s'ouvrait pour le bat 8, avec le recrutement de Jérôme Chailloux à l'IRIA, avec ses collègues de Vincennes Yves Devillers, ingénieur système spécialiste d'UNIX, et Louis Audoire, électronicien. Le bâtiment 8 allait s'affranchir des contraintes de calcul sur matériel français en s'équipant d'un Exormacs Motorola, 1er ordinateur basé sur le chip 68000, puis avec un VAX 780 de DIGITAL, qui remplaçait le PDP-10 dans les Universités Américaines. On évacua le bat 8 pour le reconstruire selon des plans dessinés par Louis Audoire et Didier Ronsain, avec en partie centrale un centre de calcul autonome et un laboratoire de micro-électronique. Le jour de l'arrivée du VAX, une fois la recette de l'OS effectuée, les techniciens de DIGITAL à peine repartis, Patrick Sinz de l'équipe Vincennoise arriva avec une disquette de boot d'UNIX Berkeley tombée du ciel. UNIX démarré, il se produisit une scène surréaliste. On faisait tous la queue pour ajouter notre nom au fichier des utilisateurs etc/passwd, qui commençait par Ken Thomson et Dennis Richie.

Plus tard, Yves Devillers installa Internet, en reprenant la première liaison du CNAM, ce faisant ignorant l'impératif officiel d'utiiliser le protocole réseau X400 des P&T. Cet accès français unique à Internet fit vite boule de neige. Tous les courriers électroniques depuis ou vers la France transitaient par le VAX du bâtiment 8 de Rocquencourt ! Yves était de fait le gestionnaire de FNET, le premier fournisseur d'accès à Internet en France. On fut vite débordé, et FNET fut externalisé dans une structure privée EUNET, tandis que l'association AFNIC gérait les noms de domaines.

Entre temps, Jean-Marie Hullot s'était désintéressé de KB, et avait programmé une extension objet CEYX de Le_Lisp (évolué de VLISP), puis avait conçu Interface Builder, qu'il présenta à Steve Jobs à l'occasion d'un

congrès. Steve Job l'embaucha pour diriger l'architecture de la machine NEXT, qui deviendra MAC OS-X. Avant de lui confier la mission secrète qui donna naissance à l'iPhone...

J'avais gardé le système KB opérationnel, et de nouveaux étudiants ont travaillé à son extension: François Fages pour l'extension au calcul des prédicats, et Philippe Le Chenadec pour investiguer les structures algébriques munies de formes canoniques, et notamment les groupes finiment engendrés. J'ai gardé un listing de KB à la date du 16 avril 1984, avec quelques exemples d'utilisation (38*20*4 cm).

Après un intermède de près d'un an où j'ai fait l'ingénieur pour connecter une photo-composeuse Mergenthaler au Multics qui était alors l'ordinateur principal du Centre, je suis revenu à la recherche avec le plan ambitieux de mettre en chantier un démonstrateur interactif de théorèmes, combinant ordre supérieur, récurrence et réécriture. L'aspect interactif était essentiel, utilisant l'expertise du mathématicien utilisateur pour guider le recherche, tout en profitant d'automatisation de certaines preuves, lorsque l'énoncé était décidable facilement. Le modèle récent était le système LCF récemment conçu à l'Université d'Edimbourg, où ces preuves étaient programmées dans un méta-language fonctionnel de tactiques.

Pour moi, il s'agissait d'un projet à long terme, et mon premier souci était de choisir le langage de programmation pour le développer. Ce langage se devait d'être sûr, et donc basé sur une sémantique bien comprise. Je demandai conseil à mon voisin de bureau Jean-Jacques Lévy, qui pour moi faisait autorité en programmation. Et il me dit: "Pourquoi pas ML ?" J'étais très surpris de sa réponse. ML était le méta-langage des tactiques du système LCF, interprété, plutôt un langage de commandes spécialisé, a priori inadapté à serveur de cadre de programmation d'un logiciel conséquent. Néanmoins, il avait pour avantage d'avoir une sémantique très claire, puisqu'il avait été conçu par des spécialistes du λ-calcul. De plus, l'idée que le langage de tactiques soit le même que celui du démonstrateur ouvrait la possibilités de bootstaps intéressants.

Je suis allé voir Gilles Kahn pour avoir son opinion sur la suggestion de Jean-Jacques. Sur le moment, il ne fit pas de réaction tranchée. Pourtant, une semaine plus tard, il vint me voit avec une bande magnétique contenant les derniers sources de LCF, comme si elle tombait du ciel. Je me mis avidement au travail. C'était un gros système LISP, mais soigneusement programmé. Je lisais le source, la nuit, de mon domicile, connecté par un modem à notre Multics, qui avait un compilateur Maclisp avec tous les outils afférents comme Emacs.

Un module m'intéressait particulièrement, c'était le programme LISP in-

terpréteur de ML. Surprise: c'était juste EVAL, l'interpréteur de S-expressions LISP. Le programme intéressant était en fit le macro-générateur de cette S-expression: un compilateut de ML en LISP. C'est Lockwood Morris qui l'avait écrit, un vrai bijou. Je passais des heures, la nuit, à lire ce code. Puis il me vint une idée saugrenue. Je remplace l'appel à EVAL par l'appel du compilateur LISP. Instantanément je gagne un facteur 10 dans le temps d'exécution. Un exemple flagrant d'une évidence qui sautait aux yeux: "it was hiding in plain sight".

Cela changea la donne. La base de code de traitement de ML dans LCF devenait l'environnement de développement du future système de preuves. Guy Cousineau, professeur de Paris 7 en sabbatique à l'IRIA, se joignit à l'effort ML et implémenta le traitement des types concrets, une amélioration considérable pour l'utilisation du langage pour le traitement des données symboliques. De nombreux étudiants se joignirent à l'effort. Michel Mauny sut adapter les combinateurs catégoriques de Pierre-Louis Curien pour faire un nouveau modèle d'exécution du langage, et bientôt toute une équipe développait un puissant langage fonctionnel à partir du noyau ML. En parallèle, un effort international mené par Robin Milner, professeur à Edimbourg et architecte de LCF, s'efforçait de définir un standard pour le langage ML, unifiant les différentes implémentations. Notre équipe participa à cet effort, mais in fine ne se plia pas à ce standard, pour un nombre de raisons, la première étant que nous voulions garder notre liberté de chercheurs à faire évoluer le langage au gré des innovations. L'avenir nous donnera raison.

Robin me demanda de cesser d'appeler ML le langage développé dans notre équipe pour ne pas faire de confusion avec Standard ML en cours de définition. Nous optâment pour CAML, acronyme de Categorical Abstract Model Language. CAML eut un certain succès, notamment en France dans l'enseignement. L'IRIA conclut une convention avec ILOG pour assurer sa distribution.

A un certain point, on sépara la base LCF du noyau ML. Mais, aux premiers temps, j'avais join mes efforts à ceux de Larry Paulson à Cambridge qui venait d'étendre LCF avec un système de théories. Il me vint une idée bizarre: fondre le système KB augmenté avec Cambridge LCF, sous le principe simpliste que "tout ça, c'est du LISP". D'ailleurs j'avais fait un fichier de configutation qui permettait à notre système d'être porté sur toutes les implémentations de LISP disponibles à l'époque! L'idée était d'augmenter la fonctionnalité de la logique sous-tendant LCF par des théories équationnelles, et de lier les tactiques de simplifications de LCF à un système de réécriture canonique produit par KB à partir du système d'équations. Le problème était que les structures de représentation des termes de LCF

n'avaient rien à voir avec celles utilisées par KB. J'ai dû faire un transducteur entre les deux représentations, dans les deux sens, pour pouvoir lancer KB sur des équations LCF, et à la fin du calcul régurgiter les règles de réécriture du système canonique en tant que simplifications LCF. J'avais interconnecté deux machines à gaz avec des tuyaux ad-hoc par la plomberie LISP.

Notre équipe s'appelait le projet Formel. Il y avait jeu de mot sur Formel signifiant la conception d'un environnement "For ML". J'avais même fait déposer par le service juridique de l'IRIA la marque déposée "For_ML". Nous participions à un effort national piloté par le CNRS appelé le GRECO de programmation. Pour le colloque inaugural, à Bordeaux en 1984, je préparai une démonstration du système Formel, nom éphémère de l'éléphant blanc LCF+KB. Le centre de Roquencourt venait de se doter d'un service audiovisuel, avec un banc de montage vidéo. Je leur explique mon plan, et on me répond que c'est faisable, avec une semaine de tournage et une semaine de montage. Je décide qu'on ferait un jour de tournage et un jour de montage.

Le tournage n'était pas une mince affaire. Pour faire un clip un peu intéressant, il fallait montrer autre chose que des équations défilant sur un écran alphanumérique noir et blanc. J'ai ajouté à cette fin quelques gadjets audiovisuels. Par exemple, on pouvait faire imprimer les axiomes d'une théorie logique sur des transparents écrits par un robot traceur, par l'intermédiaire du langage de description topologique FLIP de Gilles Kahn. On pouvait aussi la visualiser sur l'écran bitmap couleur Colorix de Louis Audoire. Le montage commençait par un générique de plusieurs minutes montrant un clip psychédélique des Vincennois, sur fond du Messie de Handel, terminant sur la page titre, inspirée de l'affiche de campagne de Mitterand, réalisée par Jean-Jacques Lévy sur Colorix, synchronisée sur l'Alleluia. Après, la démo se déroulait au rythme apaisant de la sonate de Franck. Tous les morceaux sonores étaient repiqués de ma collection de 33 tours, au grand dam du service audio-visuel qui prétendait que c'était illégal, alors que je soutenais qu'il s'agirait juste d'une projection privée. Du coup, cette vidéo n'a pas été disponible par la suite. Mais j'avais gardé un master, qui a pû être repiqué sur un DVD, exemplaire unique.

Cette démonstration au GRECO a été la dernière exécution du logiciel union de KB et LCF. Le développement de CAML se poursuivit en tant qu'implémentation du langage de programmation, de ses bibliothèques et de sa documentation. J'ai conservé un exemplaire du source de CAML, en trois volumes, daté de mai 1989 (21*27*4 cm).

En 1983 Thierry Coquand rejoignit l'équipe comme thèsard. Ce brillant mathématicien m'aida à concevoir le système logique du futur assistant

de preuves en combinant un calcul des prédicats d'ordre supérieur avec le système de types du lambda-calcul $F\omega$ de Jean-Yves Girard. Il établit la cohérence du système dans sa thèse, pendant que j'entreprenais la programmation d'un vérificateur de typage du langage, appelé Calcul des Constructions. Le noyau du vérificateur était une machine virtuelle appelée "The constructive engine" et publiée en 1989 comme le code CAML commenté du noyau de l'assistant de preuves CONSTR. Je poussais ainsi au bout le point de vue de Donald Knuth sur "Literate programming": un programme doit être suffisamment clair pour servir de publication aus idées mathématiques sous-jacentes. Toute ma carrière, je me suis efforcé de publier les idées innovantes de mes logiciels sous la forme des programmes source commentés.

C'est ainsi qu'un 1985 j'ai enseigné à l'Université Carnegie-Mellon le cours "Formal Structures for Computation and Deduction" entièrement sous ce style.

Ceci conclue mes mémoires de programmeur des années 70 et 80. Dans les années 90, Xavier Leroy et Damien Doligez utilisèrent CAML pour bootstrapper une implémentation native du langage appelée Caml Light. Le langage fut muni de modules, et devint Caml Special Light, puis finalement Ocaml. Du côté preuves, Christine Paulin-Mohring avait travaillé avec Thierry Coquand à l'extension du Calcul des Constructions avec des types inductifs à la Martin-Löf, et une nouvelle équipe entrepris la généralisation du vérificateur CONSTR en un assistant de preuves puissant appelé Coq, et implémenté bien sûr en Ocaml.