Claude Pair (1934 – 2024)

Quelques éléments sur l’apport de Claude Pair à la création de la science Informatique

29 décembre 2024

Ces documents sont rassemblés dans l’urgence à l’intention de celles et ceux qui voudraient écrire à propos de l’œuvre scientifique de Claude Pair. Ils ne couvrent donc qu’une partie de cette personnalité hors du commun.

Jean-Pierre Finance (06 85 11 02 79), Brigitte et Jacques Jaray (06 62 62 13 35), Pierre Lescanne (06 85 70 94 31), Alain Quéré (06 01 88 42 49)[1], anciens élèves de Claude Pair, restent à votre disposition pour tout complément.

1 Les numéros de téléphone permettent de nous joindre.

Publié le : 11/10/2007 Par : Isabelle Bellin **Niveau** ○○○

Claude Pair : un mathématicien qui rêvait de programmation

HISTOIRE DU NUMÉRIQUE

LANGAGES, PROGRAMMATION & LOGICIEL

Précurseur français de la recherche en informatique, Claude Pair a toujours eu une vision très pragmatique de cette discipline, qu'il concevait pour soulager le travail, affranchir des contraintes spatiales et temporelles.

Entretien avec Claude Pair, mené par Isabelle Bellin.

Vous avez lancé les premiers travaux de recherche en informatique à Nancy. Comment êtes-vous passé de l'enseignement des mathématiques à la recherche en programmation ?

J'ai eu beaucoup de chance. J'avais presque trente ans lorsque cette nouvelle discipline est née. Le mot n'existait d'ailleurs pas encore, il date de 1962 ; avant cela, on parlait de « calcul automatique ». C'était tout simplement l'âge idéal pour partir à l'aventure d'une science émergente.

Après mon agrégation de mathématiques à l'École normale supérieure en 1956, j'ai fait une partie de mon service militaire au Commissariat à l'énergie atomique (CEA) dans un service de calcul numérique. Il

y a là un ordinateur, le Bull Gamma AET, avec une unité centrale de 8 mots de mémoire. Aussi rudimentaire, encombrant, fragile et coûteux qu'il soit, cet ordinateur fait alors figure d'outil révolutionnaire. Il permet déjà de faire des calculs qu'on ne peut pas faire à la main, comme des résolutions d'équations aux dérivées partielles pour mesurer la propagation de phénomènes... Même si nous avions du mal à évaluer les temps de calcul — il m'est arrivé d'abandonner un calcul, en réalisant qu'il monopoliserait 15 jours de machine —, nous mesurions déjà les gains de temps et de fiabilité. Nous avons aussi vite réalisé que notre point faible était l'écriture des programmes. Quel langage employer ? À l'époque, on utilise uniquement un langage machine codé en binaire (suite de 0 et de 1 ou bits) ne faisant appel qu'aux opérations câblées dans la machine : un programme est une suite d'instructions dont chacune ordonne à la machine d'exécuter une de ces opérations.

Néanmoins, on commence à parler ici et là de langages plus proches de la langue usuelle. En 1958, se tient au CEA une conférence sur une « programmation automatique », une merveille à nos yeux ! On y évoque un « assembleur » capable de traduire automatiquement dans le langage de la machine les mêmes opérations câblées mais codées par des mots (additionner, déplacer, imprimer...). Nous ignorions que **John Backus** venait de créer au laboratoire d'IBM à New-York le **Fortran** : un langage inspiré de celui des mathématiques, avec des opérateurs , x..., des variables et des mots du langage courant (if, go to, do...) : « un langage avec des si et des do » disait **Jean Legras**, le pionnier du calcul automatique à Nancy, mon directeur de thèse.

Pourtant, après mon service militaire, en 1959, c'est d'abord à l'enseignement en classe de mathématiques spéciales que je me suis consacré, plutôt qu'à la recherche en mathématiques qui m'attirait peu : à l'époque, l'école dominante — celle de **Bourbaki**, un groupe de mathématiciens français — privilégie l'abstraction, la généralisation des concepts, le formalisme. Si j'appréciais l'activité mathématique, je souhaitais qu'elle soit plus proche des besoins des gens. Je me suis néanmoins vite rendu compte que la préparation aux grandes écoles était réservée à une élite ; en outre, son côté répétitif ne me convenait guère.

En 1962, j'ai repensé à ma rencontre de 1958 avec l'ordinateur. J'ai contacté l'université de Nancy et je me suis inscrit au cours de troisième cycle « Analyse et calcul numérique » créé par Jean Legras quelques années auparavant en même temps qu'un centre de calcul. C'est à ce moment là que j'ai réellement bifurqué. Jean Legras m'a obtenu pour l'année 1963-64 un poste au CNRS qui commençait timidement à s'intéresser ce type de recherche.

À cette époque, un nouveau langage voit le jour, Algol 60, qui laisse espérer une vraie recherche en programmation. Ce sera votre premier sujet de travail ?

Effectivement, **Algol 60** a été une révélation pour nous; nous avons pris conscience que l'informatique pouvait devenir une science. Le rapport final définitif sur ce langage créé par un groupe international, date de 1963. Algol est un langage révolutionnaire car il commence à ressembler à la langue naturelle : un programme n'est plus une suite d'instructions apparaissant au même niveau mais, de même qu'en français une proposition peut contenir d'autres propositions, une instruction peut contenir d'autres

instructions, qui elles-mêmes en contiennent d'autres... et ceci sans limite. Cette application de la notion mathématique de **récursivité** tenait une place importante dans Algol 60. Il restait néanmoins à trouver l'algorithme qui permette de traduire automatiquement ce langage complexe en langage machine. Si cette « compilation », comme on dit, paraît aujourd'hui plutôt banale, à l'époque, nous n'étions pas sûrs d'y parvenir. C'est d'ailleurs une des raisons qui a finalement causé l'échec d'Algol 60.


Une machine IBM 1620.
Source : Wikipédia

Malgré cela, ou plutôt grâce à cela, nous étions convaincus de l'intérêt de développer de nouvelles techniques, non seulement pour construire un **compilateur**, mais aussi pour programmer dans un tel langage efficacement et sans trop se tromper (en limitant les bugs ou bogues comme on dit). J'ai donc mis en place une petite équipe pour aborder ces problèmes. C'était une des premières en France (avec Grenoble, Paris, Toulouse). Entre 1963 et 1966, nous avons écrit un compilateur Algol 60 pour un IBM 1620. C'était un exploit : nous ne disposions pas de la machine, elle était chez le constructeur, à Metz, accessible seulement certaines nuits. Ce compilateur n'a permis que de soutenir quelques thèses. Il n'était pas assez abouti pour être utilisé par d'autres que nous et l'université n'ayant pas acheté la machine, l'aventure n'alla pas plus loin. Malgré tout, cela nous a permis d'apprendre beaucoup sur la compilation et sur la programmation en général.

Après ces débuts à la fois prometteurs et décevants sur la programmation, comment se sont orientées vos recherches ?

Nous avons continué à travailler sur la compilation, du point de vue théorique, avec l'idée de construire un métacompilateur, autrement dit un programme engendrant un compilateur à partir de la définition du langage à traduire. Nous n'étions pas très armés du côté de la sémantique (le sens d'un texte). Dans un premier temps, nous avons affronté le problème en travaillant sur l'analyse syntaxique des langages : il s'agit de mettre en évidence la structure du texte sous la forme d'un arbre ; dans le cas des langues naturelles, une telle analyse montre la décomposition d'une phrase en groupe de mots ou l'inclusion de propositions subordonnées dans une proposition principale.

Manuel rédigé sous la direction de Pierre Bacchus, Jacques André et Claude Pair, publié en

Plus tard, les travaux autour d'Algol 68, qui a succédé à Algol 60 pour étendre considérablement son domaine d'application, ont profondément influencé nos recherches. J'ai accepté en 1967, à la demande de Louis Bolliet, un des fondateurs de l'informatique grenobloise, de prendre la responsabilité du groupe français Algol, dans le cadre de l'AFCET (Association française pour la **cybernétique** économique et technique), prédécesseur de l'actuelle Association française des sciences et technologies de l'information et des systèmes. Ce fut l'occasion de se retrouver entre informaticiens universitaires français : les occasions étaient rares à l'époque. Nous avons aussi créé une école d'été francophone en informatique, dont la première session s'est tenue à Alès en 1971. Nancy devenait une référence en informatique.

Algol 68 a le mérite d'avoir initié une réflexion approfondie : la programmation structurée, les travaux sur les types de données, puis les langages à objets sont

issus de cette réflexion. Mais on avait voulu y mettre tous les concepts les plus en pointe. C'était une belle recherche mais aboutissant à un outil trop complexe, de définition trop formelle. Il sera encore moins utilisé que son prédécesseur Algol 60. C'est d'ailleurs d'un schisme du groupe international qui l'avait conçu qu'est né le langage Pascal, beaucoup plus simple et qui a rapidement percé. 1975.

En 1973, vous fondez le Centre de recherche en informatique de Nancy, le CRIN qui, en s'associant à l'INRIA, deviendra en 1997 le LORIA (Laboratoire lorrain de recherche en informatique et ses applications). Racontez-nous.

En 1963, nous étions quatre ou cinq. Notre activité était centrée sur les langages de programmation et leur compilation. Puis très vite, elle s'est étendue aux méthodes de conception des programmes et à la structure des informations qu'ils manipulent dès qu'ils sortent du domaine numérique : une recherche fondamentale tournée vers le logiciel, faisant appel à nos compétences en mathématiques. Nous avions aussi une forte activité d'enseignement, passant du troisième cycle aux écoles d'ingénieurs puis à l'IUT et à la maîtrise. En 1973, nous avons été reconnus comme équipe associée au CNRS. La plupart des chercheurs et enseignants du CRIN étaient d'anciens élèves, les premiers thésards en informatique. En 1975, nous étions déjà 70 personnes, en majorité des enseignants, et nous avions plusieurs centaines d'étudiants.

Mais, comme les autres laboratoires français, nous étions écrasés de charges d'enseignement et nous ne disposions d'aucun personnel technique. En outre, il a fallu encore près de dix ans pour obtenir un matériel à la hauteur de nos ambitions, comparable à celui des laboratoires étrangers. L'ouverture nationale et internationale s'amorçait pourtant. En particulier, nous avions noué des liens avec l'IRIA, ancêtre de l'INRIA : en 1972, lors de la création du LABORIA (Laboratoire de recherche d'informatique et d'automatique de l'IRIA), **Jacques-Louis Lions** m'avait demandé d'en être conseiller scientifique. Aujourd'hui, le LORIA est un laboratoire de 450 personnes et on entend parler toutes les langues dans les couloirs. Les études fondamentales sur la construction des logiciels en sont restées un point fort, même si d'autres axes ont été introduits, par exemple sur la reconnaissance de la parole et des images.

Vous avez abandonné la recherche en informatique en 1981 pour vous consacrer aux systèmes éducatifs en tant que directeur des lycées au ministère de l'éducation nationale puis de recteur d'académie. Quel est aujourd'hui votre point de vue sur l'informatique ?

C'est celui d'un utilisateur d'informatique domestique comme les autres, même moins que beaucoup d'autres ! Nous sommes bien loin des seules applications de l'informatique au calcul numérique comme c'était le cas au départ. Et si je confronte mes ambitions de l'époque à la réalité actuelle, j'observe que l'informatique et son couplage avec les télécommunications via Internet ont bien affranchi l'être humain de la plupart des opérations répétitives dans le travail et des contraintes temporelles et spatiales. Désormais, on échange à distance, des chirurgiens peuvent même opérer de loin, toute la connaissance humaine est accessible partout. Mais cela donne-t-il plus de capacité de réfléchir et de comprendre ? Assurément non. L'informatique à la maison consomme énormément de temps et conduit à un manque de disponibilité croissant, en particulier pour les relations de proximité. L'ordinateur permet de faire plusieurs choses à la fois mais l'Homme est un être inséré localement et il ne peut pas bien faire plusieurs choses à la fois.

Je suis inquiet des conséquences sur le développement personnel des générations actuelles et futures. La tendance à prendre sur Internet les informations tous azimuts exige le développement de l'esprit critique. En outre, ces informations n'ont que peu de valeur si elles ne sont pas intégrées à un savoir ; or, nous ne savons pas grand chose sur la manière dont se constitue et s'enrichit le savoir d'un individu.

Je suis aussi très préoccupé par le fossé que l'informatique crée avec les plus faibles, les plus pauvres, ceux qui ont du mal à se mouvoir dans l'abstrait. L'école devient d'ailleurs de plus en plus exigeante sur ces capacités d'abstraction, d'où un risque d'échec supplémentaire pour ces mêmes personnes.

Finalement, les technologies de l'information et de la communication sont la meilleure et la pire des choses. Il est urgent de se préoccuper de la manière dont est utilisé l'ordinateur par le grand public, de ses conséquences sur la psychologie des individus et donc sur la société et la civilisation.

A tout CRIN : de la naissance à la maturité (1963-1976)

Histoire et mémoire de l'informatique universitaire à Nancy, 14 janvier 2019

Claude Pair[1]

Le mot « informatique » a été forgé en 1962 pour nommer une société de services, à un moment clé : le début du passage d'un outil – l'ordinateur – à une science qui sera un peu plus tard désignée en français par ce même nom. Pour moi, 1962 est l'année où je prends contact avec Jean Legras qui m'ouvre son Centre de Calcul et m'invite à suivre le cours de programmation qu'il y a confié à deux assistantes, Jacqueline Giannesini et Marion Créhange : Jacqueline enseigne le « langage-machine » de l'ordinateur IBM 650 et Marion son « code de programmation » [13] qui facilite quelque peu cet exercice.

Par rapport à un *outil* – un ordinateur programmé dans un dialecte qui lui est propre – une *science* fait preuve de généralité : à cette époque, il s'agit d'introduire des langages de programmation « évolués », plus proches du langage humain et indépendants de la machine utilisée. 1962 est ainsi l'année où apparaît FORTRAN IV, un langage destiné à tous les ordinateurs… pourvu qu'ils soient des IBM, une firme alors toute-puissante ! Mais depuis plusieurs années, un comité international travaille à définir ALGOL 60 [5] indépendant de tout constructeur et plus novateur que FORTRAN : un programme n'est plus une suite d'instructions, toutes au même niveau, mais structuré comme un texte en langue naturelle ; la récursivité apparaît à deux niveaux, pour accroître la capacité d'expression des programmes et pour décrire la syntaxe dans un métalangage dû à John Backus[2] et proche des grammaires du linguiste Chomsky [12].

ALGOL 60 sera fondamental dans la constitution d'une science informatique, par les notions qu'il introduit et les problèmes que pose sa « compilation », c'est-à-dire sa traduction dans le langage-machine de l'ordinateur dont on dispose. Je le découvre à travers une collection des Communications de l'ACM[3] trouvée dans une armoire du Centre de Calcul, puis grâce à d'autres articles que je demande au centre de documentation du CNRS[4]. Sa définition finale est publiée en 1963 [39] et une rencontre se tient à Grenoble pour faire connaître sa version française ; Jean Legras en revient enthousiasmé.

Nous décidons donc de l'adopter comme support d'enseignement de la programmation, et d'en construire un « compilateur ». Jean Legras incite quelques étudiants à participer à l'aventure. Je me souviens de la réunion inaugurale à la rentrée 1963, où je trace les perspectives et organise une équipe de trois personnes pour construire ce compilateur, sur un ordinateur 1620 prêté par IBM dans l'attente qu'il puisse être acheté pour remplacer 650. Mais cette machine n'est accessible que certaines nuits et à Metz : ce n'est guère commode ! En 1965 pourtant, le compilateur est presque achevé, le groupe européen des utilisateurs de 1620 nous demande de le lui présenter à Mannheim [42], et l'université de Saint-Andrews (Ecosse) de le lui envoyer sur cartes perforées, l'outil de l'époque pour entrer des informations dans un ordinateur. Mais, patatras, plutôt que le 1620, le Centre de Calcul choisit, dans le cadre du Plan Calcul naissant, une machine française CAE 510 qui possède déjà un compilateur ALGOL. Notre motivation disparaît et IBM arrête son prêt, avant que le compilateur soit réellement utilisable par d'autres que nous.

---

[1] Voir [55, 15] et l'ouvrage de P-É Mounier-Kuhn sur l'émergence de la science informatique en France, PUPS.

[2] Auteur du langage FORTRAN, détaché par IBM auprès du Comité définissant Algol.
Docteur honoris causa de l'université Henri Poincaré en 1989.

[3] Association for Computing Machinery.

[4] Ancêtre de l'INIST, aujourd'hui installé à Vandœuvre.

Mais l'expérience ne sera pas perdue. Des notions ont été dégagées, comme celle d'analyse syntaxique : pour traduire un texte, il faut d'abord en faire une analyse grammaticale, sous la forme d'un arbre ; en 1964, j'avais publié un article [41] introduisant la méthode d'analyse « ascendante » utilisée dans notre compilateur ALGOL[5]. Des étudiants ont été formés, en particulier à la théorie des langages, et cinq thèses de troisième cycle ont été soutenues [16, 67, 1, 6, 24, 27] dont trois sur ce compilateur. J'ai moi-même présenté fin 1965 une thèse d'Etat[6] [43] qui récapitule les méthodes d'analyse syntaxique des langages de Chomsky. Deux sociétés de service nous ont demandé de les conseiller sur la compilation[7]. Surtout, une équipe de recherche en informatique a été créée, individualisée à l'intérieur du Centre de Calcul. La plupart des artisans de notre compilateur ont pris un emploi à l'extérieur, mais de jeunes étudiants la rejoignent : sa taille augmente rapidement – plus de 50 chercheurs en 1975 – grâce à la création de postes pour faire face à l'explosion et à la diversification de l'enseignement de l'informatique dans les facultés, écoles, et à l'IUT créé en 1966. En 1971, nous accueillerons un des quatre centres créés par le ministère de l'éducation nationale pour former des professeurs de lycée. Pendant longtemps, les chercheurs seront presque tous des enseignants et nous ne disposerons d'aucun technicien.

Cette pauvreté s'étend au matériel auquel nous avons accès, ce qui nous conduit à une recherche plutôt théorique. D'ailleurs, faire de l'informatique une science, c'est comprendre, formaliser, généraliser, ce que permet l'origine mathématique de la plupart de nos chercheurs. Mais nous nous refusons à faire des maths pour les maths. Depuis 1963, j'avais écrit plusieurs articles et communications dans le cadre de l'AFCALTI[8]. En les relisant aujourd'hui, je constate que leur caractère formel et pas toujours explicite les rend difficiles à lire et donc à utiliser pour fonder un programme. C'est encore vrai pour ma thèse, administrativement rangée dans la spécialité « mathématiques » où elle est la première en France traitant d'informatique, ce qui ne manque pas de susciter l'étonnement et le mépris de certains mathématiciens. L'année précédente, Jean-Claude Boussard avait soutenu à Grenoble une thèse d'Etat [8] sur la réalisation d'un compilateur ALGOL, qui, elle, avait dû être qualifiée de « science appliquée » en fonction de son contenu.

Il nous faudra un peu de temps pour inventer un style d'écriture à la fois rigoureux et moins uniquement mathématique, adapté à cette nouvelle science qu'est l'informatique [7] et à ses concepts fondamentaux : algorithmes [59] comme pour l'analyse syntaxique ; langages comme ALGOL, structures de données (ou d'information) [47, 50] comme les textes, les arbres, les piles (premier entré, dernier sorti). A partir de 1966, nos sujets de recherche – algorithmes de cheminement dans les graphes [44, 19, 20] dont les arbres sont un cas particulier, analyse syntaxique de figures à deux dimensions [38], puis langages d'arbres et définition des structures de données – dépassent la compilation, tout en réutilisant un certain nombre de notions qu'elle nous a fait rencontrer. En outre, nous voulons ouvrir la science informatique sur des disciplines variées (histoire [28], linguistique [45, 69, 48, 11], médecine [30], géologie [18]), comme vient de l'expliquer Marion Créhange. Nous nous construisons ainsi une place spécifique dans la recherche informatique française, moins appliquée qu'à Grenoble et Toulouse, moins théorique qu'à Paris. Nos relations nationales sont concrétisées

[5] Cette méthode sera réinventée en 1966 par N. Wirth et H. Weber sous le nom d'analyse par précédence.

[6] Au jury, présidé par Jean Legras, participaient Jacques Arsac (directeur de l'Institut de programmation à la faculté des sciences de Paris) et Jean-Claude Herz (IBM France et professeur associé à la faculté des sciences de Lille), ainsi que deux mathématiciens de Nancy.

[7] pour PL/1 (lettre de la CERCI, 18/07/1966) et pour SNOBOL [31].

[8] Association française de calcul et de traitement de l'information, qui a succédé à l'association française de calcul créée en 1956 et qui prend ensuite successivement les noms d'association française d'informatique et de recherche opérationnelle (AFIRO) et enfin de cybernétique économique et technique (AFCET, 1968-1994).

par l'école d'été d'informatique que nous avons fondée en 1971 sous l'égide de l'AFCET, des liens étroits avec l'IRIA à partir de 1972, mon élection en 1973 comme président de la sous-section d'informatique créée au Comité consultatif des universités, et la même année notre reconnaissance par le CNRS comme équipe associée puis en 1976 comme laboratoire associé.

Nous obtenons aussi petit à petit une reconnaissance internationale, d'abord grâce à ma participation au groupe de travail de l'IFIP [9] « Formalization of programming concepts », dès sa création en 1966. Elle me conduira à co-diriger le Ph. D. d'un étudiant de l'université de Pennsylvanie [2, 3] qui traduira en outre ma thèse en anglais. Alain Quéré et moi publions en 1968 dans la revue américaine Information and Control un article [57] qui introduit les « bilangages », langages d'arbres généralisant les langages linéaires et permettant de traiter de l'analyse syntaxique des langages de Chomsky [61, 52, 49]. Cet article recueille un écho international notable, en particulier une correspondance avec Rózsa Péter, célèbre logicienne hongroise (1905-1977) avec qui nous présentons deux communications croisées à une université hongroise et à l'Académie des Sciences de Budapest [60, 58]. Quant à l'école d'été, dès sa création elle attire des collègues Suisses et Espagnols, puis se tient dans divers pays francophones, en Europe, Afrique du nord, Amérique : Montréal en 1977.

A partir des bilangages, la voie est ouverte pour introduire et étudier des algèbres plus générales [32]. Elles conduiront à une formalisation des types de données comme systèmes formels à axiomes équationnels [65, 50, 51], dans le cadre de la première Action thématique du CNRS en 1972-74, proches de ce qui sera nommé plus tard « types abstraits algébriques ». Ceux-ci seront utilisés pour définir la sémantique des langages de programmation [34, 53, 54, 33, 10], une question qui nous fera revenir à la compilation[10], ou plutôt la méta-compilation : décrire les compilateurs pour pouvoir les engendrer automatiquement [6, 37, 22].

Dans la même veine est proposée une formalisation de la sémantique d'Algol 68 [25], un nouveau langage [68] qui sera l'occasion de réinvestir beaucoup de nos réflexions. Comme son prédécesseur Algol 60, il fait avancer la science informatique. Le groupe Algol franco-belge, que nos collègues de Grenoble m'ont proposé d'animer, se fixe pour objectif de le faire connaître, notamment à travers plusieurs publications [9, 68, 4]. Un des progrès importants apporté par ce langage est que les types de données n'y sont plus prédéfinis comme en Algol 60 (nombres entiers, réels, valeurs booléennes, tableaux), mais peuvent être introduits en fonction des besoins. Le langage permet donc de définir, non seulement des programmes et des fonctions, mais aussi des types de données, et ceci de manière récursive. Trente ans après, j'ai eu la surprise de voir reconnue, grâce à Michel Sintzoff et Pierre Lescanne, mon antériorité sur la décidabilité de l'égalité de deux types récursifs, établie par un article [46] publié en 1970 à propos d'Algol 68.

Cependant, Algol 68 ne sera guère utilisé pour programmer. Ce n'était pas d'un langage à tout faire, donc complexe, qu'on ressentait alors le besoin pour améliorer l'efficacité de la programmation, mais plutôt d'une réflexion en profondeur sur cette activité. Le temps était venu de la programmation structurée, prônée dans les années 1972-76 par des leaders de la communauté informatique internationale [17]. A partir de 1973, nous introduisons et enseignons une voie différente : une méthode déclarative et modulaire, dite « déductive », qui, au lieu de décrire des calculs, définit des objets en partant du résultat à atteindre [56, 23]. L'idée est proche de ce qui sera nommé plus tard programmation fonctionnelle. En même

[9] International Federation for Information Processing.
[10] Ce sujet avait été effleuré dès 1964 à propos de la compilation [40].

temps est développé le projet Civa [21] adapté au développement de logiciels de taille importante, se référant à cinq bras d'un seul dieu pour cinq temps de la vie d'un programme.

Après 1968, la réorganisation universitaire créant à Nancy trois établissements d'enseignement supérieur, avec de l'informatique dans chacun d'eux, et le caractère « sans murs de notre labo, auraient pu mener à une dispersion des informaticiens. Leur volonté et l'existence d'un séminaire commun permettront de l'éviter. Au contraire, la transformation en 1971 du Centre de Calcul en service commun de ressources, le séparant de l'enseignement et de la recherche, conduit à l'indépendance du laboratoire, qui sera confortée par son association au CNRS. Il prend le nom de CRIN en 1975. Les thèmes de recherche se sont diversifiés. Jean-Paul Haton est venu du laboratoire d'automatique pour créer un pôle d'intelligence artificielle avec des chercheurs déjà présents. Deux thèses d'Etat ont été soutenues [21, 14] et d'autres s'annoncent [33, 26, 35, 66], permettant à leurs auteurs de diriger des recherches. Le directeur, élu président de l'Institut polytechnique et de la nouvelle section informatique-automatique du CNRS, est moins disponible. Il est temps de revoir l'organisation. Lorsqu'en 1976 le CRIN devient laboratoire associé, on y distingue cinq équipes : informatique théorique et théorie des langages ; outils et méthodes de développement de logiciels ; reconnaissance des formes et intelligence artificielle ; informatique d'organisation et bases de données ; informatique pour l'éducation et la formation [62, 63], en lien avec la formation des professeurs de lycée dirigée alors par Maryse Quéré.

Quelques années plus tard, en 1981, nommé directeur au Ministère de l'Education nationale, je passe le flambeau à Jean-Claude Derniame. Puis viendront Jean-Pierre Finance et Jean-Marie Pierrel, avant que le CRIN se transforme en LORIA en 1997 : il y a 20 ans, mais après 40 ans d'informatique universitaire à Nancy.

Références

1. André J. 1965. *Contribution à la construction d'un compilateur Algol pour IBM 1620*, Thèse 3e cycle[11], Nancy.
2. Berger, J. 1977. *A study of inference for regular bilanguages*, Ph.D., University of Pennsylvania.
3. Berger, J. et Pair, C. 1978. Inference for Regular Bilanguages, *Journal of Computer and System Sciences* 16.
4. Bacchus, P., J. André, C. Pair (éds) 1975, *Manuel du langage algorithmique Algol 68*, Actualités scientifiques et industrielles, Hermann.
5. Backus, J. W. (éd.). 1960. Report on the algorithmic language Algol 60, *Nümerische Matematik.* 2 (Le comité de définition comprend un Français, B. Vauquois, professeur à Grenoble).
6. Bellegarde, F. 1972. *Face, langage d'écriture de compilateurs : définition et implémentation*, Thèse 3e cycle, Nancy.
7. Borillo, M. et Pair, C. 1979, L'informatique une dimension nouvelle de la science contemporaine, in Mathieu, V. et Rossi, P. *La culture scientifique dans le monde contemporain,* Scientia, Milan.
8. Boussard, J. C. 1964. *Étude et réalisation d'un compilateur Algol 60 sur calculatrice du type IBM 7090/7094 et 7040/44*, Thèse d'État Science appliquée, Grenoble.
9. Boussard, J. C. et Duby J. J. (éds) 1971, Rapport d'évaluation Algol 68, *Revue française d'informatique et de recherche opérationnelle* R-1.
10. Broy, M., M. Wirsing, C. Pair. 1984. A systematic study of models of abstract data types, *Theoretical Computer Science* 33.
11. Carbonell, N. 1972. *Rôle des fonctions récursives primitives de ramifications dans la définition d'une langue naturelle : application à la syntaxe française*, Thèse 3e cycle, Nancy.
12. Chomsky, N. 1959. On certain properties of grammars, *Information and Control* 2.
13. Créhange, M. 1961. *Structure du code de programmation*, Thèse 3e cycle, Nancy.
14. Créhange, M. 1971. *Description formelle, représentation, interrogation des informations complexes : système Pivoines*, Thèse d'État, Nancy.
15. Créhange, M. et Haton M. C. 2014. L'informatique universitaire à Nancy : un demi-siècle de développement, *1024, Bulletin de la société informatique de France*, 3.
16. Cusey, M. 1964. *Construction d'un compilateur Algol pour IBM 1620*, Thèse 3e cycle, Nancy.
17. Dahl O. J., E. W. Dijkstra, C. A. H. Hoare 1972. *Structured Programming*, Acad. Press.
18. De Bary, C. 1974. *Conception et réalisation d'un logiciel de gestion et d'interrogation d'une base de données géologiques*, Thèse 3e cycle, Nancy.
19. Derniame, J. C. 1966. *Étude d'algorithmes pour les problèmes de cheminement dans les graphes finis*, Thèse 3e cycle, Nancy.
20. Derniame, J. C. et Pair, C. 1971. *Problèmes de cheminement dans les graphes*, préface de Jean Kuntzmann, Monographies d'informatique, Dunod.
21. Derniame, J. C. 1974. *Le projet CIVA, un système de programmation modulaire*, Thèse d'État, Nancy.
22. Deschamp, P. 1980. *Production de compilateurs à partir d'une description sémantique des langages de programmation : le système Perluette,* Thèse docteur-ingénieur, Nancy.
23. Ducrin Amédée. 1984. (ouvrage collectif : M. Créhange, J. P. Finance, J. Guyard, N. Hertschuh, C. Pair, M. Quéré, J. Souquières), *Programmation*, tome 1, Dunod Informatique.

[11] A l'époque concernée, les universités françaises délivraient trois sortes de thèses : deux courtes, thèse de troisième cycle et thèse d'ingénieur pour les impétrants possédant ce titre ; une longue, la thèse d'État, aujourd'hui remplacée par l'habilitation à diriger des recherches.

24. Émond, A. 1965. *Application de la notion de pile à des problèmes portant sur les chemins des graphes*, Thèse 3e cycle, Nancy.
25. Finance J. P. 1974. *Contribution à la formalisation de la sémantique d'un langage de programmation* : *application à Algol 68*, Thèse 3e cycle, Nancy.
26. Finance J. P. 1979. *Étude de la construction des programmes : méthodes et langages de spécification et de réalisation de programmes,* Thèse d'État, Nancy.
27. Floc'h, A. 1966, *Achèvement d'un compilateur Algol, traitement des procédures*, Thèse 3e cycle, Nancy.
28. Fossier, L. et Créhange, M. 1970. Un essai de traitement sur ordinateur des documents diplomatiques du Moyen Age, *Les Annales* 25.
29. Gaudel, M. C. 1980. *Génération et preuve de compilateurs basées sur une sémantique formelle des langages de programmation,* Thèse d'État, Nancy.
30. Germain, P. 1972. *Système de gestion et d'exploitation documentaire d'un corpus de dossiers médicaux*, Thèse 3e cycle, Nancy.
31. Jaray, J. 1975. *Le langage SNOBOL4, ses applications, son implémentation*, Thèse 3e cycle, Nancy.
32. Lescanne, P. 1971, *Étude de quelques théories des langages et généralisation du théorème de Kleene,* Thèse 3e cycle, Nancy.
33. Lescanne, P. 1979, *Étude algébrique et relationnelle des types abstraits et de leurs représentations*, Thèse d'État, Nancy.
34. Livercy, 1978. (nom collectif : J.-P. Finance, M. Grandbastien, P. Lescanne, P. Marchand, R. Mohr, A. Quéré, J.-L. Rémy), préface de C. Pair, *Théorie des programmes : schémas, preuves, sémantique*, Dunod Informatique ; d'après un cours de l'école d'été francophone d'informatique, Tarbes, 1974.
35. Marchand, P. 1981. *Langages d'arbres, langages dans les algèbres libres*, Thèse d'État, Nancy.
36. Marin Navarro, J. 1979. *Un método de programación : presentación, implematación, transporte*, Universitad Complutense, Madrid.
37. Maroldt, J. 1972. *Définition de Face, langage pour l'écriture des compilateurs, implémentation d'un sous-ensemble*, Thèse docteur-ingénieur, Nancy.
38. Mohr, R. 1973. *Modèle algébrique pour l'analyse syntaxique de figures*, Thèse 3e cycle, Nancy.
39. Naur, P. (éd.). 1963. Revised Report on the algorithmic language Algol 60, *Comm. ACM* .
40. Pair, C. 1964. Essai de description de la sémantique des langages de programmation, *Séminaire AFCALTI sur les langages symboliques* (animateur François Genuys).
41. Pair, C. 1964. Arbres, piles et compilation, *Revue française de traitement de l'information.*
42. Pair, C. 1965. Description d'un compilateur Algol, *European Region 1620 Users Group,* Mannheim.
43. Pair, C. 1965. *Étude de la notion de pile, application à l'analyse syntaxique,* Thèse d'État, Nancy.
44. Pair, C. 1966. Sur les algorithmes pour les problèmes de cheminement dans les graphes finis, *Théorie des graphes,* Centre international de Calcul, Rome, et Dunod, Gordon-Breach, 1967.
45. Pair, C. 1967. *La formalisation des grammaires*, Centre de Recherches et d'Applications Linguistiques, Faculté des lettres et sciences humaines, Nancy.
46. Pair, C. 1970. Concerning the Syntax of Algol 68, *Algol Bulletin* 31.
47. Pair, C. 1971 *Les structures de données et leur représentation en mémoire*, École d'été francophone d'informatique, publié avec la collaboration de M.-C. Gaudel, IRIA, 1977.

48. Pair, C. 1971. La formalisation des langages de programmation (introduction d'André Lentin), *Mathématiques et Sciences humaines* 34, École Pratique des Hautes Études.
49. Pair, C. 1971. Application de la théorie des ramifications au problème de l'équivalence structurale de deux C-grammaires, *Revue française d'informatique et de recherche opérationnelle* R-2.
50. Pair, C. 1974. *Formalization of the Notions of Data, Information and Information Structure*, IFIP Working Conference Data Base Management, North Holland.
51. Pair, C. (éd.) 1974. *Rapport de l'ATP Informatique 1972 : Informatique théorique, programmes et données,* Equipe associée 364, Université de Nancy II.
52. Pair, C. 1976. Les arbres en théorie des langages, *Colloque Les arbres en algèbre et en programmation*, Lille.
53. Pair, C. 1978. Some Theoretical Aspects of Program Construction, in F. L. Bauer and M. Broy (éds), *Program Construction,* International Summer School Marktoberdorf, Springer.
54. Pair, C. 1982. Abstract data types and algebraic semantics of programming languages, *Theoretical Computer Science* 18.
55. Pair, C. 1988. A tout CRIN : histoire d'un laboratoire, *Colloque sur l'Histoire de l'Informatique en France*, Grenoble. CRIN : The History of a Laboratory, *Annals of the History of Computing 12* (1990).
56. Pair, C. et Maroldt, J. 1975. Introduction à une méthode de programmation déductive, in *L'enseignement de la programmation*, IRIA.
57. Pair, C. et Quéré, A. 1968. Définition et étude des bilangages réguliers, *Information and Control* 13.
58. Pair, C. et Quéré A. 1970. Sur les fonctions récursives primitives de ramifications, *Acta Mathematica Academiae Scientiarum Hungaricae* 21.
59. Pair, C., R. Mohr, R. Schott. 1988. *Construire les algorithmes*, Dunod Informatique.
60. Péter R. 1968. *Die Pairschen freien Binoïden als Speziallfälle des angeordneten freien holomorphen Mengen*, Eötvös Lorand Universität Budapest.
61. Quéré, A. 1969. *Étude des ramifications et des bilangages*, Thèse 3e cycle, Nancy.
62. Quéré, M. 1975. Démarches algorithmique et déductive en enseignement assisté, rapports avec l'intelligence artificielle, *Colloque Analyse de la didactique des mathémat.,* Bordeaux.
63. Quéré, M. 1975. Modèle mathématique d'un système d'enseignement, *Computers in education,* North Holland.
64. Quéré, M. 1980. *Contribution à l'amélioration des processus d'enseignement, d'apprentissage et d'organisation de l'éducation : l'ordinateur outil et objet d'enseignement, application au projet SATIRE,* Thèse d'État, Nancy.
65. Rémy, J. L. 1974. *Structures d'information, formalisation des notions d'accès et de modification d'une donnée*, Thèse 3e cycle, Nancy.
66. Rémy, J. L. 1982. *Étude des systèmes de réécriture conditionnels et application aux types abstraits algébriques*, Thèse d'État, Nancy.
67. Romac, R. 1964, *Étude des méthodes de tri*, Thèse 3e cycle, Nancy.
68. Van Wijngaarden, A., B. J. Mailloux, J. E. L. Peck, C. H. A. Koster (éds) 1969. *Report on the Algorithmic Language Algol 68*, Springer. Traduit par le groupe franco-belge Algol (P.Arnal, J. Buffet, A. Quéré, éds), Actualités scientifiques et industrielles, Hermann, 1972.
69. Villard, J. 1969. *Emploi de PL/1 pour les problèmes de linguistique*, Thèse 3e cycle, Nancy.

Claude Pair (1934 – 2024)

Quelques éléments sur l’apport de Claude Pair à la création de la science Informatique

29 décembre 2024

Ces documents sont rassemblés dans l’urgence à l’intention de celles et ceux qui voudraient écrire à propos de l’œuvre scientifique de Claude Pair. Ils ne couvrent donc qu’une partie de cette personnalité hors du commun.

Jean-Pierre Finance (06 85 11 02 79), Brigitte et Jacques Jaray (06 62 62 13 35), Pierre Lescanne (06 85 70 94 31), Alain Quéré (06 01 88 42 49)[1] restent à votre disposition pour tout complément.

1 Les numéros de téléphone permettent de nous joindre.

L'informatique de Claude Pair

L'apport de Claude Pair à la création de la science informatique (années 1963 – 1981)

Marion Créhange, Pierre Lescanne et Alain Quéré [1]

Introduction

Sur une suggestion amicale de Claude Pair, nous avons accepté, 50 ans après, de présenter son activité scientifique en tant que l'un des fondateurs de l'informatique universitaire en France. Cet article couvre la période 1963–1981 : en effet après avoir été professeur en classe préparatoire, Claude rejoint l'enseignement supérieur en 1963. Dix-huit ans plus tard, en 1981, il s'oriente vers la haute administration en devenant directeur des lycées au ministère de l'Éducation nationale.

Avant d'aborder la rédaction, nous avons proposé à Claude l'organisation d'un colloque. Cette formule a permis de multiplier les témoignages et de mieux faire collectivement le bilan de ses recherches. Ce colloque a eu lieu au LORIA, à Nancy, le 14 juin 2019 [2]. Il a réuni 150 participants avec 17 intervenants dont Antoine Petit, PDG du CNRS. Près de deux ans plus tard, nous avons assemblé nos notes, puisé dans les documents à notre disposition et activé notre mémoire.

1. marion.crehange@sfr.fr, pierre.lescanne@ens-lyon.fr, apmf.quere@laposte.net. Article également disponible sur HAL, https://hal.archives-ouvertes.fr/hal-03193950v2.

2. http://claudepair.fr/.

DOI:10.48556/SIF.1024.18.91 **1024** – Bulletin de la SiF, numéro 18, Novembre 2021, pp. 91–116

Précisons d'entrée que le lecteur ne trouvera pas ici des démonstrations rigoureuses ni des formalisations complètes. Il lui faudra pour cela consulter les nombreuses publications de Claude ou les 48 thèses qu'il a dirigées [3].

Nous avons choisi d'ajouter à des propos scientifiques des parties plus anecdotiques signalées en italique. Nous espérons ainsi aérer la lecture et évoquer l'ambiance du travail d'équipe que Claude a instauré, équipe qui au départ se comptait sur les doigts d'une main. Équipe interuniversitaire qui, en grandissant, sera associée au CNRS en 1973 grâce à Claude, puis constituera le Centre de recherche en informatique de Nancy (CRIN) en 1975. La croissance s'est poursuivie : 70 personnes en 1977, une centaine en 1984 avec la venue de l'INRIA [4].

Il faut se souvenir qu'au début des années 60, la science informatique est dans la petite enfance. La plupart des domaines que nous connaissons aujourd'hui sont alors à créer, le plus souvent dans l'indifférence — ou même contre l'avis — des mathématiciens qui ne voient alors aucune raison de s'intéresser aux ordinateurs. Il faut aussi rappeler que les moyens de communication actuels n'existent pas. Il n'y a pas encore de terminologie propre à la recherche informatique [5] et beaucoup de chercheurs français publient dans leur langue maternelle, ce qui contribue à les isoler dans le monde. C'est dans ce cadre que Claude Pair imagine des concepts qui ont été redécouverts par la suite. Comme nous le montrerons, Claude est un bel exemple d'application de la loi de Stigler [6] qui affirme qu'une découverte scientifique ne porte jamais le nom de son premier auteur. Nous souhaitons donc que le lecteur retienne de ce texte le caractère extrêmement novateur de ses recherches. Le recul du temps permet en effet d'affirmer qu'il a été un authentique visionnaire.

Les débuts : la compilation

Vers l'informatique

Comme sa formation initiale l'y conduit [7], Claude devient professeur de « taupe » à Metz avant son service militaire qui lui offre l'opportunité de faire ses premiers pas en « calcul automatique » au CEA en 1958-59. Il y programme en langage machine

3. Pour être précis, 47 thèses sur la période considérée ici puis, quelques années plus tard, celle de Radhia COUSOT (1985).

4. En 1997, l'ensemble est devenu le LORIA, unité mixte associant CNRS, INRIA et les universités. En 2017, il comptait 300 permanents de 48 nationalités (rapport HCERES). Sur l'histoire de l'informatique à Nancy, voir aussi [9, 25].

5. Un seul exemple : pour traduire l'inélégant *garbage collector*, Claude a introduit le terme « ramasse miettes », voir Wikipedia.

6. Il se dit que cette loi s'applique aussi à son auteur, voir « loi de Stigler » sur Wikipédia.

7. Né à Blâmont (Meurthe-et-Moselle) en 1934, Claude devient un élève brillant au collège à Lunéville. Il est admis en 1951 au lycée Louis le Grand à Paris, puis en 1953 à l'École normale supérieure. Il est agrégé en 1956, et nommé professeur de mathématiques spéciales à Metz (1956-1957).

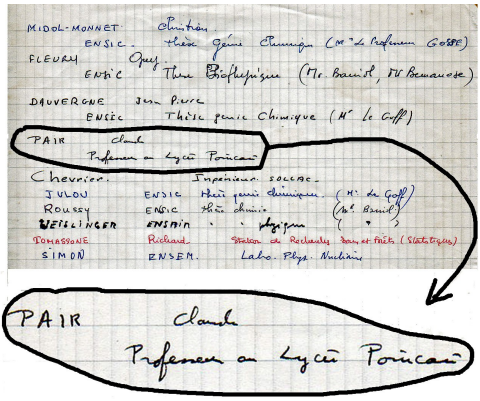

FIGURE 1. Émargement au cours du soir d'informatique en 1963.

sur Bull Gamma « à extension tambour », concurrent de l'IBM 650. Il est ensuite nommé à Nancy, au lycée Henri Poincaré.

> *À Nancy, l'informatique universitaire a été introduite par Jean Legras*[8] *dès 1957 avec une machine à programme câblé, l'IBM 604, puis un ordinateur*[9] *IBM 650 dès 1958*[10]*. En 1962, Claude prend contact avec Jean Legras qui lui donne accès au « Centre de calcul » qu'il a créé, et l'invite à suivre à la rentrée le cours de programmation sur 650 destiné à des adultes (cf. figure 1). Marion Créhange, chargée de la seconde partie, ne se doutait pas que cet élève, qui lui paraissait doué, allait devenir le père de la recherche en informatique à Nancy et serait son patron de thèse d'état !*

Claude prend vite conscience que, derrière des machines différentes, se cache une réalité commune qu'il préfère explorer, plutôt que de s'orienter vers l'analyse numérique, spécialité de Jean Legras. Au Centre de calcul, il consulte la collection des Communications de l'ACM et d'autres revues. C'est ainsi qu'il découvre un nouveau langage de programmation, Algol [3], défini par un groupe international indépendant de tout constructeur, donc attirant pour des universitaires, et beaucoup plus

8. http://www.professeurs-medecine-nancy.fr/livre_jean_legras.htm.

9. Le mot « ordinateur » a été inventé en 1955 par le philologue Jacques Perret, en réponse à une demande d'IBM, qui voulait un nom plus parlant que « calculateur » pour son IBM 650.

10. L'Est Républicain du 6 novembre 1958 annonce la création d'un institut de calcul automatique dont la direction sera confiée à Jean Legras : « *Doté d'un matériel ultra moderne, cet institut est destiné à être commun à toutes les facultés de la région de l'Est ; les cours seront notamment suivis par les élèves de 3ème cycle de mathématiques appliquées. On pense qu'il pourrait ouvrir ses portes dans un délai de deux mois : au 1er janvier très probablement* ».

novateur que le Fortran d'IBM : la récursivité de la définition des programmes et des fonctions fait rêver. Un programme n'est plus une suite d'instructions apparaissant au même niveau, mais commence à ressembler à un texte en langue naturelle ; sa syntaxe est d'ailleurs décrite par des règles proches des grammaires formelles dites *context-free* [11], introduites quelques années plus tôt par le linguiste Noam Chomsky. Compiler ce langage demeure un problème sérieux. Claude, devenu pour un an attaché de recherche au CNRS, constitue une équipe dans ce but et, dès 1963, il signe une première publication aux Comptes-rendus de l'Académie des sciences [P1], puis, en 1964, il publie l'article intitulé « Arbres, piles et compilation » dans la revue française de traitement de l'information [P2].

Cours de DEA, création de l'équipe

> *En 1963, Jean Legras assiste, à Grenoble, à un séminaire dont l'objet est la présentation de la version française de la définition d'Algol 60. Claude se souvient de son retour enthousiaste : « un langage avec des SI et des DO ! ». Jacques André* [12][T3] *raconte : « toute affaire cessante, ils ont organisé dans le cadre du DEA un cours sur Algol 60 : ça a été pour moi une révélation car c'était quand même plus passionnant que le PASO de la 650 ! Quand Claude a parlé au début de l'année scolaire suivante de prendre des étudiants en 3e cycle pour faire un compilo d'Algol 60, j'ai sauté sur l'occasion ».*

À cette époque, Claude met en place des réunions régulières, premier « séminaire informatique nancéien », associant théorie et applications. Il signe plusieurs publications internes au Centre de calcul.

> *Jacques André raconte : « Claude Pair nous donnait des conférences sur les langages et nous devions chacun rédiger un chapitre. Je me souviens qu'il m'avait rendu ma prose couverte de rouge. Voyant cela, je m'étais dit que ce n'était pas la peine que je continue... Mais en réalité il était content de ma rédaction et visait seulement à en améliorer le fond ».*

Compilateur Algol 60 (1963-1965) et thèses à Nancy

Puisqu'il n'existe pas encore de cursus informatique, la plupart des étudiants qui rejoignent cette équipe jusqu'en 1968 viennent des « mathématiques pures ». C'est le cas de Michel Cusey [T1], premier thésard de Claude et pilote de la petite équipe.

> *Jacques André se souvient : « On a commencé le compilo sur IBM 650, j'y ai travaillé la lecture des nombres, et assez vite, bonne nouvelle, on allait avoir accès à un 1620 chez IBM (cf. figure 2). Mais cette machine était à Metz et n'était disponible qu'après la fermeture des bureaux. Tous les quatre* [13]*, on*

11. Que Claude, adepte de la francophonie, rebaptisera « langages à contextes libres » dans son cours.
12. `http://jacques-andre.fr/`.
13. Jacques André, Michel Cusey, Alain Floc'h, Jean-Marie Laporte.

FIGURE 2. IBM 1620, source Wikipédia.

y allait avec la voiture de Michel, départ vers 17 heures et retour... quand nos essais étaient finis, le plus souvent à deux heures du matin. La bécane était en vitrine et on avait l'impression d'être des ours en cage quand les passants nous regardaient avec curiosité. On « sandwichait » autour de la machine jusqu'au jour où le directeur du centre IBM nous a convoqués pour nous signifier assez rudement que les bières sur le pupitre, ce n'était pas vraiment dans l'esprit de l'entreprise. Du coup, on est allé souvent prendre une choucroute à la fin du travail à la brasserie de la gare de Metz, et je n'en ai jamais mangé d'aussi bonnes depuis... ».

Avant d'envisager la programmation du compilateur, Michel Cusey travaille sur son ossature générale : l'analyse syntaxique, les déclarations de variables, la gestion des piles et de l'ensemble de la mémoire, la construction du programme objet. Il le fait en respectant scrupuleusement les idées de Claude qui souhaitait vérifier sur un cas concret la pertinence de son approche mathématique. Claude est ravi que sa théorie « marche » et il lui en est reconnaissant. Jacques André est en charge des entrées-sorties et des procédures de service. La définition du langage Algol 60 ne définissant rien sur les entrées-sorties, il adopte les propositions de Donald Knuth [21]. *Jean-Marie Laporte s'occupe des tableaux. Plus tard, Marion Créhange traite des procédures, en particulier de la récursivité, avec la collaboration d'Alain Floc'h qui sera ensuite chargé de la mise au point finale du compilateur.*

Citons Claude [14] *: « En 1965, le compilateur est presque achevé. Le groupe européen des utilisateurs de 1620 me propose de le présenter à sa réunion de Mannheim* [P5]. *À partir de là, l'université de Saint-Andrews (Écosse) me demande de le lui envoyer sur cartes perforées, l'outil de l'époque pour entrer des informations dans un ordinateur. Mais, patatras, pour remplacer le 650 vieillissant, Legras choisit, dans le cadre du Plan calcul naissant, plutôt que le*

14. A tout CRIN : de la naissance à la maturité (1963-1976), Archives Poincaré, Nancy, janvier 2019.

1620, une machine française CAE 510 qui possède déjà un compilateur Algol. Notre motivation disparaît et IBM arrête son prêt, avant que le compilateur soit réellement utilisable par d'autres que nous ».

Dans les mois qui suivent, l'équipe commence à se disperser. Mais l'expérience ne sera pas perdue. Des notions ont été dégagées, comme celles d'analyse syntaxique et de pile, des étudiants ont été formés, en particulier à la théorie des langages, et cinq thèses de troisième cycle ont été soutenues, dont trois à propos du compilateur [T1, T2, T3, T4, T5].

Fin 1965, Claude soutient sa thèse d'État, « Étude de la notion de pile, application à l'analyse syntaxique ». Elle récapitule les méthodes d'analyse syntaxique des langages de Chomsky, mettant en évidence trois familles d'algorithmes de construction d'arbres syntaxiques, une ascendante et deux descendantes « en profondeur d'abord » et « en largeur d'abord », dira-t-on plus tard. D'un style formel, rangée dans la spécialité « mathématiques », elle suscitera quelque incompréhension des mathématiciens « purs ». Par la suite, Claude pensera d'ailleurs que sa présentation était trop mathématique : « *Il faudra un peu de temps pour inventer un style d'écriture rigoureux mais moins uniquement mathématique, adapté à cette nouvelle science et tenant compte de ses composantes fondamentales : algorithmes* [P66]*, types de données* [P24] *(comme les piles ou les arbres syntaxiques d'une grammaire de Chomsky), langages de programmation* ». Quoi qu'il en soit, il s'agit de la seconde thèse d'État en informatique, après celle que Jean-Claude Boussard avait soutenue l'année précédente à Grenoble, décrivant un compilateur Algol qu'il avait écrit pour un IBM 704 que son université avait la chance de posséder. Cette thèse, elle, était qualifiée « de science appliquée » car l'informatique n'était pas encore reconnue comme une discipline à part entière à l'époque dans les universités françaises.

Le chercheur et sa prescience

L'analyse syntaxique de précédence

L'analyse syntaxique est la première étape de la compilation : elle consiste à calculer, d'après la chaîne de caractères du programme à traduire, sa structure grammaticale sous la forme d'un arbre. Cette opération doit être faite de façon déterministe si on veut éviter tout retour en arrière. Claude montre que c'est possible pour le langage Algol, en mettant au point un algorithme (cf. figure 3) qui le permet, fondé sur une analyse ascendante, qui sera celui du compilateur nancéien et qu'il décrit dans un article publié en 1964.

Cet algorithme, redécouvert en 1966 par Niklaus Wirth et Helmuth Weber [36]*, est nommé méthode de « précédence » (ordre de priorité sur les caractères au cours de l'analyse). Dans leur livre sur l'analyse syntaxique* [1]*, Aho et Ullman parlent du concept de précédence de Wirth-Weber (p.404), mais,*

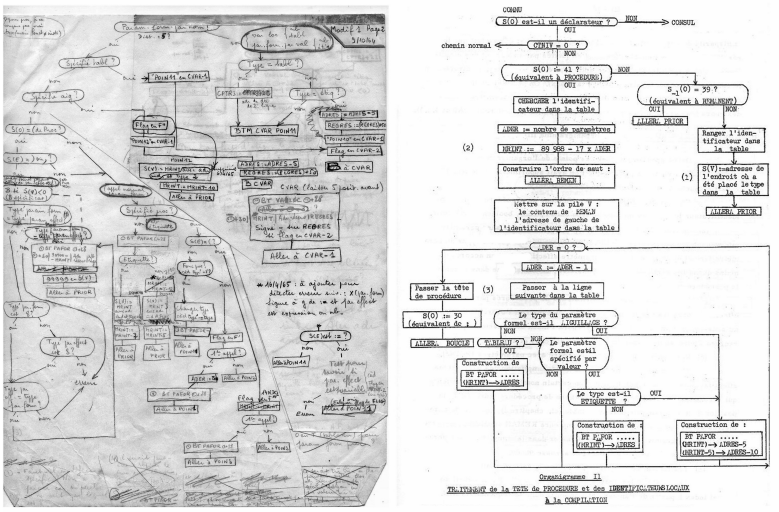

FIGURE 3. La compilation, document de l'époque : les idées et leur mise en forme.

surtout, de façon étonnante, ils leur attribuent aussi l'algorithme, en écrivant « les grammaires simples de précédence ont été définies par Wirth et Weber (1966) et indépendamment par Pair (1964) » (p. 428). Une belle application de la loi de Stigler !

Les problèmes de cheminement dans les graphes

Après avoir soutenu sa thèse et constaté l'arrivée d'étudiants visant une thèse de troisième cycle, Claude ouvre de nouveaux domaines de recherche, à partir des notions rencontrées pour la compilation. Il s'agit en particulier des graphes dont les arbres sont un cas particulier, et de leur exploration à l'aide d'une pile pour résoudre les problèmes de cheminement, par exemple l'existence d'un chemin d'un point à un autre ou la recherche de plus courts chemins. Constatant le foisonnement de différents articles indépendants sur le sujet et la similitude des algorithmes pour des problèmes divers, il remarque qu'en adoptant un point de vue algébrique, il est possible de retrouver ces algorithmes, ainsi que d'autres, par simple définition d'homomorphismes adaptés. Cette idée élégante sera présentée plus tard dans différentes publications [T6, P16]. Elle est malheureusement restée méconnue (ou plagiée ?). Le lecteur intéressé est invité à consulter Jean-Claude Derniame [11] qui raconte cette « histoire chaotique ».

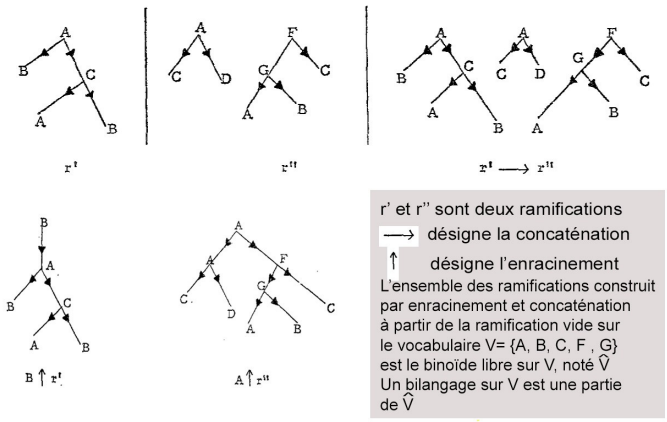

FIGURE 4. Ramifications et bilangages (*Information and Control* [P11]).

Langages et bilangages

Claude est donc féru d'algèbre, et c'est pourquoi la théorie des langages le comble. Un langage est un ensemble de phrases, éléments unidimensionnels ; mais l'analyse syntaxique construit des arbres, à deux dimensions, et Claude nomme « bilangage » un ensemble d'arbres (cf. figure 4). De même qu'un langage a une structure algébrique de monoïde (une seule opération), les bilangages conduisent à deux opérations, pour engendrer les arbres, et à la structure de « binoïde ».

> *L'histoire commence en mai 1968, par la publication de l'article de Claude Pair et Alain Quéré* [T8] *« Définition et Étude des Bilangages Réguliers » dans la revue Information and Control* [P11]. *Le mot « régulier », utilisé pour les langages reconnus par un automate fini, est étendu ici aux bilangages : l'ensemble des arbres syntaxiques d'un langage context-free est un bilangage régulier.*
>
> *Après cette publication, Claude reçoit de nombreuses demandes de tirés à part. L'une vient de Rózsa Péter une chercheuse hongroise, pionnière de la calculabilité et qui a écrit le tout premier livre sur les fonctions récursives* [29] *et publié sur elles des contributions remarquables.*

En particulier, en utilisant la récurrence structurelle, Rózsa Péter a eu l'idée d'étendre la récursivité qui, à l'origine, se limitait aux entiers naturels, à ce que nous appellerions aujourd'hui des structures de données bien fondées. Elle les nomme *angeordneten freien holomorphen Mengen* (ensembles libres holomorphes structurés).

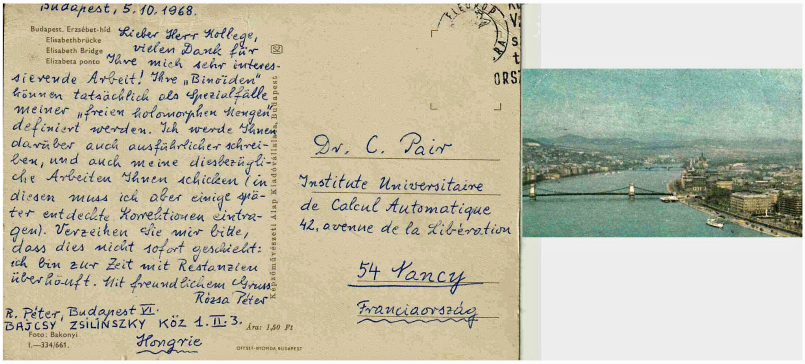

FIGURE 5. Rózsa Péter accuse réception du tiré à part et annonce sa réponse sur le fond.

On peut comprendre son excitation quand elle a vu qu'à partir de besoins de formalisation en informatique, Claude proposait une telle structure pour les arbres syntaxiques. Claude envoie donc aussitôt l'article qu'elle demande et le remerciement arrive par une jolie carte postale du Pont Élisabeth à Budapest (cf. figure 5).

Elle se plonge donc avec passion dans les écrits de Claude, y passant des nuits, lisant et réfléchissant jusqu'à 4–5 heures du matin, comme elle l'écrit avec enthousiasme et fraîcheur, dans sa lettre du 28 octobre 1968. On a l'impression qu'à travers le rideau de fer qui divise alors l'Europe, elle parle à un ami qu'elle connaît depuis toujours, bien qu'elle ne l'ait jamais rencontré. Elle annonce dans cette lettre l'article qu'elle va publier dans Acta Mathemica dont elle envoie, à son tour, un tiré à part à Claude, article intitulé : « Die Pairschen freien Binoïden als Spezialfälle der angeordneten freien holomorphen Mengen ». On notera la création, en allemand, du nouvel adjectif « Pairschen » !

En réponse, Claude et son élève Alain Quéré publient dans la même revue, à l'invitation de leur collègue hongroise, un complément intitulé « Sur les fonctions récursives primitives de ramifications » qui montre que l'égalité est primitive récursive dans le binoïde libre qu'on peut comparer aux ensembles holomorphes libres de Rózsa Péter.

Dans la ligne de recherche ainsi créée, en 1971, Pierre Lescanne [T16] a emprunté à la théorie des catégories ses concepts pour généraliser les langages réguliers, reconnaissables par automates et solutions d'équations spécifiques. Il montre ainsi que, même généralisées, ces trois familles de langages coïncident.

En 1974, Pierre Marchand étudie les « bigrammaires » et les systèmes transformationnels [T26]. Plus tard, en utilisant des algèbres libres (objets mathématiques à plusieurs opérations, dont les chaînes de caractères ne sont qu'un cas très particulier), il généralise, lui aussi, les notions de langages reconnaissables et de langages algébriques, mais dans un cadre plus général où des relations existent entre les opérateurs. Il montre la construction algorithmique des algèbres minimales de reconnaissance pour les parties reconnaissables, et, par ailleurs il explique, par l'adjonction d'une loi à une algèbre, la hiérarchie de Chomsky, qui devient alors infinie [T45].

L'équivalence des types infinis

Le groupe Algol francophone, que les collègues grenoblois de Claude lui ont proposé d'animer, se fixe pour objectif de faire connaître le nouveau langage Algol 68, notamment à travers plusieurs publications comme la traduction du rapport de définition et un copieux manuel d'utilisation. Algol 68 fait avancer la science informatique : le principal progrès est que les types de données n'y sont plus prédéfinis comme en Algol 60 (nombres entiers, réels, valeurs booléennes, tableaux), mais peuvent être introduits en fonction des besoins de chaque programme. Il permet donc de définir des types de données, et de le faire éventuellement de manière récursive.

La question se pose alors de savoir si deux définitions distinctes sont équivalentes, c'est-à-dire conduisent au même type. En 1970, Claude publie un algorithme de décision dans la revue Algol Bulletin, alors destinée aux spécialistes d'Algol 68, sous le titre un peu imprécis *Concerning the Syntax of Algol 68* [P19].

> *Cette décidabilité est redécouverte en 1993 dans un cadre plus général par Roberto Amadio et Luca Cardelli* [2]. *La démonstration repose sur un concept mis en évidence autour de 1980 : la coinduction. Cardelli* [2] *a indiqué que, cherchant une antériorité, il avait pensé à Simula, mais n'avait pas regardé du côté d'Algol 68. L'aurait-il fait, qu'il n'aurait sans doute pas trouvé l'article de Claude, bien caché dans la littérature scientifique !*

Aujourd'hui, nous dirions que la démonstration de la décidabilité d'un problème se fait soit par réduction à un autre problème décidable, soit par la description d'un algorithme de décision, et c'est ce que propose Claude dans la section 4 de l'article cité [P19]. Il propose une méthode de raisonnement sur des objets infinis qui, comme plus tard la coinduction, fait appel à la résolution d'une équation à point fixe [15]. Une équation à point fixe peut avoir plusieurs solutions. Alors que l'induction s'appuie sur un raisonnement fondé sur le plus petit point fixe, la coinduction s'appuie sur le plus grand point fixe. Claude ne peut pas connaître ces différences, qui ne seront mises en évidence qu'une décennie plus tard, mais il doit en avoir l'intuition, car

15. Une équation à point fixe est une équation de la forme $X = F(X)$. Si l'ensemble dans lequel on cherche les solutions est ordonné, par exemple par inclusion d'ensembles, il y a une plus grande et une plus petite solution (on dit aussi un plus grand et un plus petit point fixe).

sa démonstration fonctionne en deux temps. Premier temps, il démontre l'unicité du point fixe, le plus petit point fixe est aussi le plus grand. Deuxième temps, il démontre que sur cet unique point fixe, on peut mettre en œuvre un processus de décision.

> *Et Benjamin Pierce peut donc écrire dans son livre de référence* [31] *qu'il s'agit bien de la première démonstration de la décidabilité de l'égalité de deux types (équi)-récursifs alors qu'il lui faut 648 pages pour présenter aux étudiants en master la notion que Claude avait introduite, à savoir les types* [16] *dans les langages de programmation.*

Les structures de données

En juillet 1971 à Alès, lors de la première école d'été de l'AFCET [17], Claude donne l'un des trois cours, « les structures de données et leur représentation en mémoire ». Ce cours, destiné à la programmation, sera revu par Marie-Claude Gaudel et publié par l'IRIA en 1977 [P24].

Une donnée est un objet informatique (liste, pile, arbre, table...) susceptible d'être transformé par des opérations. Une structure de données [P33] indique ces opérations et leurs propriétés, de la même façon qu'en mathématiques une structure algébrique (groupe, anneau, corps...) comprend une ou plusieurs opérations et leurs propriétés sous la forme d'axiomes.

En 1972, Claude obtient du CNRS un contrat d'Action thématique programmée (ATP) sous le titre « Informatique théorique, programmes et données ». Il le confie à Jean-Luc Rémy qui précise en particulier que les axiomes sur les opérations seront des égalités (axiomes équationnels). Le contrat dure deux ans et fait l'objet en 1974 d'un rapport de Claude au nom de l'équipe de recherche associée 364 au CNRS et de l'université de Nancy II.

Dans les années qui suivent, Claude commence à s'effacer devant ses chercheurs qui prennent leur envol, échangent entre eux et publient ensemble. Jean-Pierre Finance et Jean-Luc Rémy font une communication à l'école d'été de Grenade (1973, [13]), à la suite de quoi Jean-Pierre Finance reprend avec Claude la rédaction d'un document de travail qui donnera lieu à un article dans une des premières conférences IFIP [P30]. Deux thèses en sont issues, soutenues toutes les deux le 25 juin 1974 : celle de Jean-Luc Rémy [T24] étudie les aspects les plus formels, par une utilisation de la logique mathématique ; la thèse de Jean-Pierre Finance [T25] utilise les structures de données pour décrire la sémantique d'un langage de programmation. Plus tard, celle de Marie-Claude Gaudel [T43] emploiera la sémantique pour engendrer des compilateurs et prouver leur correction. En 1979, dans sa thèse [T39], Pierre Lescanne revisite les structures d'information dans une approche d'algèbres, préfigurant les contributions de Yuri Gurevich et de ses collaborateurs.

16. Qu'il appelle « mode » à la mode de l'époque !
17. La société savante d'informatique à l'époque.

Indépendamment de l'équipe nancéienne, des développements très similaires ont émergé à partir de 1993, notamment grâce à Yuri Gurevich, sous le nom d'Evolving Algebras [18]*, puis de Sequential abstract-state machines, nom plus attractif* [19]*, quand l'industriel Microsoft s'y est intéressé.*

Ces recherches sur les données ont stimulé une activité de nature didactique sur la théorie des programmes et l'informatique théorique en général, qui sera le support d'un cours à l'école d'été d'informatique de Tarbes en 1974 puis conduira à l'écriture d'un livre publié en 1978 chez Dunod [26].

Entre eux, les sept auteurs Jean-Pierre Finance, Monique Grandbastien, Pierre Lescanne, Pierre Marchand, Roger Mohr, Alain Quéré et Jean-Luc Rémy, s'appelaient, dans un premier temps, Blanche Neige, nom inapproprié, puisqu'ils n'étaient que sept nains, sous le charme d'une princesse. Le nom de Livercy (pour Liverdun et Commercy, deux villes lorraines où eurent lieu les séances d'édition) fut finalement retenu. Ce livre, l'un des premiers à traiter d'informatique théorique, toutes langues confondues, fut utilisé, par exemple, par Jean-Louis Lassez comme base de ses cours à l'Université de Melbourne en Australie ou par Olivier Danvy, professeur à l'Université d'Aarhus au Danemark, pour s'initier à la théorie des continuations.

Autres approches et nouvelles pousses

Claude aime aussi étendre son domaine de compétence et il ne se limite pas à un seul axe en matière de recherche. Nous évoquons ici la diversité de ses intérêts. Cependant, nous sommes loin d'un inventaire à la Prévert, car, chez Claude, ses intérêts sont tous ancrés sur une exigence de rigueur scientifique. Parcourons brièvement ces autres approches qui sont devenues aussi de nouvelles directions de recherche au CRIN.

En informatique, dans la lignée des premières recherches, au début de la décennie 1970, les compilateurs de compilateurs se multiplient, le plus connu étant le couple *lex* et *yacc*. Le nom de yacc, « encore un compilateur de compilateur » montre combien le sujet est chaud à cette époque. Sa version finale est publiée en 1975 [20]. À Nancy, Claude aiguille Françoise Bellegarde [T17] et Jean Maroldt [T18] sur ce sujet, ils soutiennent une thèse en 1972. Notez l'antériorité du yacc nancéien, utilisé aussi comme outil d'enseignement. Il s'appellera Fabrication automatique de compilateurs efficaces (Face) et mis à profit pour la mise au point de compilateurs (Anne-Marie Rasser [T30]).

Dès les débuts, Claude avait conscience que l'informatique, science transversale, pouvait servir en dehors de son champ propre et donner lieu à des recherches mixtes, au-delà de simples utilisations. La liste des thèses dirigées par Claude et

explorant des nouvelles branches donne un aperçu de cet esprit d'ouverture. Indiquons quelques exemples qui montrent l'étendue des domaines abordés : linguistique [T9, T21, T22], médecine [T11, T19], enseignement [T42], représentation des connaissances [T46].

> *Le plus souvent ce sont les thésards eux-mêmes qui ont choisi des orientations correspondant à leur compétence. Ils auraient pu être découragés par un patron trop directif. Au contraire, Claude les a encouragés dans le sens de leurs goûts. Donnons quelques exemples.*
>
> *Jean-Claude Derniame s'est intéressé à la prise en compte dans un même environnement des différentes étapes de construction du logiciel : spécification, analyse, programmation, exécution, maintenance. En référence au dieu hindou auquel la tradition attribue cinq bras, il nomme CIVA son projet qui traite de définition modulaire et de classes d'objets, préfigurant les langages à objets si populaires aujourd'hui* [T23].
>
> *Pour Marion Créhange, c'est un stage au laboratoire d'Analyse documentaire pour l'archéologie (CNRS, Jean-Claude Gardin à Marseille) qui la détermine à s'orienter vers les questions de recherche d'informations. Ensuite, elle collabore avec la médiéviste Lucie Fossier qui l'avait sollicitée pour traiter des documents diplomatiques du Moyen Âge au Centre de recherche et applications linguistiques (CRAL)* [15]*; ce qui lui permet d'acquérir une expérience en recherche documentaire. Plus tard, ce sera la conception de PIVOINES, langage d'interrogation de bases de données, très « déclaratif » et indépendant des choix de représentation* [T29] *dans la ligne des études théoriques sur les structures de données. Voilà un exemple représentatif des allers-retours entre théorie et application qui se fécondent l'une l'autre, une démarche qui intéressait particulièrement Claude* [18].
>
> *Autre exemple : Maryse Quéré* [T42] *avait débuté sa recherche en mathématiques pures, puis s'était consacrée à la formation d'adultes au CUCES* [19]*. Elle avait ensuite participé à celle de professeurs de lycées à l'informatique, puis créé le Centre lorrain d'enseignement par ordinateur (CLEO), d'où son attrait pour l'usage innovant de l'informatique dans l'enseignement. Pour Roger Mohr* [20]*, la proposition de Claude d'imaginer les figures comme des objets à deux dimensions engendrés par une grammaire* [T20] *a été un premier départ vers la science des images, à laquelle il s'est ensuite consacré en*

18. Marion écrit « *j'ai pu apprécier la qualité extrême de Claude comme directeur de thèse, ferme et incitatif, mais gentil et promouvant, préoccupé par la réussite de ses élèves.* »

19. Autre institution nancéienne, le Centre universitaire de coopération économique et sociale et l'un des tous premiers organismes de formation des adultes, créé par le recteur Capelle et André Grandpierre de la société Pont-à-Mousson, puis dirigé par Bertrand Schwartz.

20. Cf. *In memorian Roger Mohr*, revue 1024 n° 11, septembre 2017.


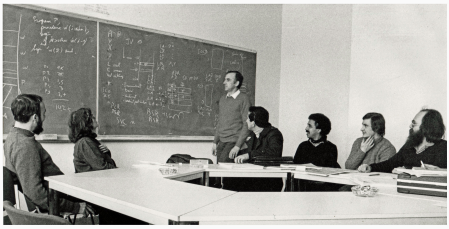

FIGURE 6. Bibliothèque et salle de réunion au CRIN en 1982.

créant au CRIN, avec Jean-Paul Haton, l'équipe Reconnaissance des formes et intelligence artificielle (RFIA).

Recherche et enseignement

Didactique de la programmation : la méthode déductive

Pour Claude, l'informatique naissante n'est pas seulement un objet de recherche, c'est aussi une discipline à enseigner. Dans les années 60, on sait laborieusement expliquer un programme, mais la démarche qui consiste à trouver un algorithme pour résoudre un problème reste difficile, voir impossible, à expliciter. Le plus souvent l'enseignant, tel un magicien, dessine un organigramme qu'il transpose ensuite dans un langage de programmation. Si la seconde partie du travail est facile, l'invention de l'algorithme est certes immédiatement saisie par les étudiants les plus brillants, mais reste incompréhensible pour la plupart. La conséquence de cette situation est l'erreur de programmation, le plus souvent découverte lors d'une exécution sur machine et rectifiée par des essais successifs sans s'appuyer sur des preuves sérieuses. Depuis 1968, la programmation structurée [12], [10] a mis en évidence une construction raisonnée des algorithmes et il existe quelques méthodes, le plus souvent sous forme de recettes standard pour résoudre certaines familles de problèmes (Méthode Warnier [34]).

D. E. Knuth propose « la programmation littéraire » [22] qu'on peut rapprocher de la méthode que Claude introduit à partir de 1973. Cette méthode sépare par étapes les activités de définition du problème, d'écriture de l'algorithme et enfin du programme. L'idée de départ est aussi simple que nouvelle : résoudre un problème, c'est partir du but à atteindre, « le résultat », et remonter le chemin jusqu'aux entités connues, appelées « données ». Autrement dit, construire un programme, c'est raisonner à partir de la définition du résultat pour revenir aux données en utilisant à chaque étape une définition puisée dans une liste prédéfinie de primitives usuelles de l'algorithmique : formules, conditionnelles, itérations — considérées ici comme

la définition de suites —, fonctions, dont la définition apparaît comme un nouveau problème à résoudre. Chaque étape de la construction est une déduction, d'où le nom de « méthode déductive ».

Pour illustrer la démarche contentons-nous d'un exemple simplissime, à savoir la recherche du maximum de 3 nombres. Commençons donc par écrire la caractérisation informelle du résultat :

$$M : \text{est le maximum de } a, b, c$$

Il s'agit alors de choisir, parmi les définitions algorithmiques référencées dans la méthode, l'une d'entre elles qui nous fera progresser vers la solution en introduisant au besoin de nouvelles entités. Ce sera, ici, une définition conditionnelle :

$$M = \text{ si } a > mbc \text{ alors } a \text{ sinon } mbc$$

avec introduction de deux intermédiaires à définir :

$$a = \text{ donnée,}$$

$$mbc = \text{maximum de } b \text{ et } c$$

et l'introduction de deux intermédiaires à définir, b et c qui sont des données. Une nouvelle conditionnelle porte sur des données qui sont connues :

$$mbc = \text{ si } b > c \text{ alors } b \text{ sinon } c$$

Ce qui résout le problème. La méthode organise ce travail dans une table à deux colonnes, les définitions informelles et définitions formelles. Ici :

Définitions informelles	Définitions formelles (algorithmiques)
M : Maximum de a, b, c	$M =$ si $a > mbc$ alors a sinon mbc
mbc : maximum de b et c	$mbc =$ si $b > c$ alors b sinon c
a, b, c : données	a, b, c = lire

Pour faire de ces définitions un algorithme séquentiel, il reste à les organiser, c'est à dire à les ordonner pour la relation « dépend de » (tri topologique) facile à déterminer depuis la colonne de droite. Une troisième colonne peut servir à établir ce tri, tâche qui peut être automatisée qu'on peut rapprocher de la méthode [P47] que Claude introduit à partir de 1973.

Définitions informelles	Ordre	Définitions formelles (algorithmiques)
M : Maximum de a, b, c	3	$M =$ si $a > mbc$ alors a sinon mbc
mbc : maximum de b et c	2	$mbc =$ si $b > c$ alors b sinon c
a, b, c : données	1	a, b, c = lire

Cette méthode a été largement employée à Nancy (université, IUT, écoles, IREM et formation des professeurs du second degré), notamment pour l'initiation avec un avantage pédagogique notoire : montrer aux étudiants que la programmation, loin

de se ramener à quelques recettes, peut s'appuyer sur le raisonnement et que ce raisonnement s'enseigne.

Cependant, sa pratique ne s'est pas répandue et elle est donc restée une méthode purement nancéienne [21]. Son implantation informatique (Bernard Huc, [T34]), était complète, mais la faiblesse des techniques ergonomiques de l'époque l'ont desservie. Encore une innovation arrivée trop tôt, sans doute !

> *Jacques Jaray évoque cette période : « à l'IUT, avec des élèves débutants en informatique, nous avions constaté qu'en mal d'inspiration pour construire l'algorithme solution d'un problème, ils commençaient par écrire* `lire(données)` *et s'ensuivait une période de grande perplexité.*
>
> *Je situe la naissance de la Méthode déductive, abrégée en MÉDÉE [22], un peu après 1973. Claude était professeur au département informatique de l'IUT, il enseignait la programmation aux élèves de première année. Il avait, entre autres comme assistants : Françoise Bellegarde, Hubert Pistré, Bernard Huc et moi-même... La méthode déductive s'est éteinte au milieu des années 80, lorsque les langages objets ont été enseignés et parce qu'elle était peu naturellement adaptée pour concevoir des systèmes réactifs ».*

Autres aventures : école d'été, IFIP, Algol 68, IREM...

Nous ne pouvons pas décrire toutes les facettes de l'activité de Claude. Il nous a plus d'une fois étonnés par sa manière d'accumuler les responsabilités... et de les assumer tranquillement. Nulle boulimie, mais simplement quelques facilités, une grande capacité de travail, de l'imagination, l'envie de servir et le soutien indéfectible de son épouse Monique. Évoquons ici quelques autres aspects de ses actions en lien avec la recherche.

Tout de suite après sa thèse, Claude entre en 1966 au groupe de travail WG 2.2 de l'IFIP *Formalisation of programming concepts* qui vient d'être créé. Nous avons indiqué son rôle d'animateur du groupe franco-belge sur Algol 68 (Manuel Algol 68, [P37]), groupe qui subsistera sous plusieurs formes jusqu'en 1993 avec des participants de Grenoble, Rennes, Nancy, Paris, Montpellier [16]. Du côté belge, le groupe bénéficiera notamment de la contribution déterminante de Michel Sintzoff [23], professeur à l'université libre de Bruxelles [17].

---

21. Mais la méthode était encore étudiée avec un oeil mathématique en 2010, bulletin N°48 de l'APMEP, juillet-août 2010.

22. Dénomination proposée par Pierre Lescanne et déclinée en Amédée Ducrin, nom d'auteur du livre de programmation associé à la méthode [P63].

23. C'est par Algol68 que Claude et ses élèves ont connu Michel Sintzoff lorsqu'il était chercheur au MBLE (laboratoire de Philips). Michel est resté en contact avec le CRIN. Lors d'une année sabbatique passée à Nancy, il y a donné un séminaire remarquable sur les fondements de la programmation.

Jacques André : « J'avais gardé quelques contacts ténus avec le Centre de calcul, et c'est ainsi que Claude Pair m'a invité à assister à une réunion de ce qui allait devenir le Groupe Algol 68, que j'ai suivi régulièrement par la suite, et « donc » le GROPLAN de l'AFCET, ce qui m'a laissé en contact avec Pair, et le monde universitaire... ».

Le premier institut de recherche sur l'enseignement des mathématiques est apparu en 1969 à Paris. C'est en 1971 que l'IREM de Nancy est créé sous l'impulsion de Claude. Il en assure la co-direction avec Jean-Louis Ovaert [24].

Dès les débuts de l'informatique universitaire, il faut aussi l'enseigner, performance inédite car, en l'absence de toute formation à une matière qui n'existe pas, les étudiants recrutés comme enseignants chercheurs, ont suivi le plus souvent des études de mathématiques. Ils ignorent donc tout de cette nouvelle discipline. En 1971, Claude crée, sous l'égide de l'AFCET, une université d'été francophone [25] (cf. figure 7), forme originale d'auto-formation du nouveau « milieu informatique ». Elle permet d'acquérir à la fois la matière à enseigner et la pédagogie pour la présenter.

1971 Alès	1972 Neufchâtel	1973 Grenade	1974 Tarbes	1975 Rabat
1976 Lannion	1977 Montréal	1978 Namur	1979 Monastir	1980 Aix en Provence
1981 Thiès	1982 Namur	1983 Sfax	...	

FIGURE 7. École d'été d'informatique de 1971 à 1983.

C'est aussi en 1971 que sont lancés des stages de « formation approfondie » à l'informatique pour quelques dizaines de professeurs du second degré. Ces cours ont lieu sur une année scolaire dans les universités de Grenoble, Nancy, Toulouse et à l'école normale supérieure de Saint-Cloud. Claude prend en charge avec une petite équipe (Noëlle Carbonell et Brigitte Jaray) la formation de Nancy ; celle-ci s'appuie à partir de 1973 sur la méthode déductive, avec programmation en LSE [26]. L'opération est stoppée par le ministère en 1976 et elle reprendra en 1982 à la suite d'un rapport sur l'introduction de l'informatique dans l'enseignement scolaire, confié en 1981 par le ministre Alain Savary à deux anciens présidents d'université, Yves Le Corre et... Claude Pair.

24. Jean-Louis Ovaert était un ami de Claude depuis leur scolarité au lycée Louis le Grand. Mathématicien hors pair et excellent pédagogue il a enseigné à la faculté des sciences de Nancy et plusieurs de ses étudiants ont rejoint l'équipe de Claude à la fin des années soixante.

25. Directeurs : Claude Pair de 1971 à 1974, Jean-Claude Derniame en 1975, Georges Stamon de 1976 à 1983.

26. Langage symbolique d'enseignement implémenté sur des ordinateurs Mitra 15 installés dans 58 lycées.

Conclusion

La pratique universitaire courante consiste à procéder à des évaluations au travers d'innombrables rapports établis par des comités d'experts. Ici, le lecteur l'aura compris, nous sommes dans une toute autre démarche puisque ce sont des élèves qui tentent d'évoquer les années de travail lointaines qu'ils ont vécues avec leur « maître », et grâce à lui. À l'époque, ils n'avaient pas bien conscience qu'ils vivaient une période aussi enrichissante pour eux que pour la science informatique. Mais ils avaient compris qu'ils bénéficiaient, auprès d'eux, d'une immense compétence, toujours disponible, parfois critique et avant tout visionnaire, comme ils ne l'ont réalisé que bien plus tard.

Critique, oui il pouvait être critique et parfois vivement mais toujours de manière argumentée et si habilement présentée qu'à la fin d'une réunion où nous avions tous largement divergé, nous entendions Claude conclure par sa synthèse si pertinente que nous le quittions avec le plan qu'il avait dessiné et pleinement convaincus que c'était notre plan [27] !

Disponible, devenu président de l'INPL (1976), Claude nous recevait encore longuement. Sur son bureau, symbole du personnage, rien d'autre qu'une feuille de papier, gratifiée de quelques lignes griffonnées [28] : son plan du jour.

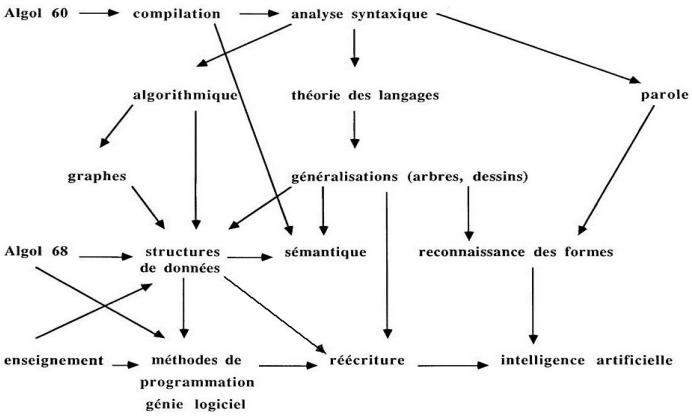

FIGURE 8. Le graphe de développement du CRIN (1963-1981) selon Claude Pair.

Encourageant, toujours à l'écoute, faisant des observations pointues, bienveillantes et constructives, et promouvant nos idées, Claude nous a incités à raisonner de façon rigoureuse et sobre.

27. Socrate parlait de maïeutique.
28. On ne peut pas dire que Claude a une belle écriture !

Fédérateur, Claude a tenu de manière constante à regrouper sans exclusion tous les enseignants chercheurs en informatique de Nancy sans tenir compte des clivages institutionnels.

Visionnaire, on peut constater que certaines publications de Claude sont encore citées plus de trente ans après leur parution [29].

Conclure, c'est aussi laisser une impression synthétique au lecteur. Alors, résumons : la motivation de Claude Pair chercheur, c'était de faire de l'informatique une science. Il y a réussi. Achevons par un graphe dont il est l'auteur (cf. figure 8).

Publications de Claude Pair

[P1] *Justification théorique de l'utilisation des piles en compilation*, CR AS 257, 3278- 3281, 1963.

[P2] *Arbres, piles et compilation*, Revue française de traitement de l'information - Chiffres 7, n°3, 199-216, 1964.

[P3] *Sur la détermination de la correspondance paramètre formel - paramètre effectif des procédures Algol*, congrès AFIRO, 157-162, 1964.

[P4] *Essai de description de la sémantique des langages de programmation*, séminaire AFIRO et rapport interne, 1964.

[P5] *Description d'un compilateur Algol*, European Région 1620 Users Group, Mannheim, 1965.

[P6] *Étude d'un algorithme d'analyse syntaxique,* manuscrit consultable aux Archives Henri-Poincaré, `https://poincare.univ-lorraine.fr/`, Nancy, 1965.

[P7] *Étude de la notion de pile, application à l'analyse syntaxique*, thèse d'État, Nancy, 1965.

[P8] *Analyse syntaxique des langages de Chomsky*, exposé à un séminaire à Grenoble, 1966.

[P9] *Sur des algorithmes pour les problèmes de cheminement dans les graphes,* Théorie des graphes, Journées internationales d'étude, Rome, Dunod - Gordon and Breach, 271-300, 1966.

[P10] *La formalisation des grammaires*, Centre de Recherches et d'Applications Linguistiques, Faculté des lettres et sciences humaines, Nancy, 1967.

[P11] *Définition et étude des bilangages réguliers*, avec A. Quéré, Inf. and Control 13, 565-593, mai 1968.

[P12] *Introduction à Algol 68*, RIRO Informatique 3, n°3, 17-52, 1969.

[P13] *Algol 60*, Techniques de l'Ingénieur H 2160, 1970.

[P14] *Sur des notions algébriques liées à l'analyse syntaxique*, RIRO Math.4 n° 3, 3-29, 1970.

[P15] *Sur les fonctions primitives récursives de ramifications*, avec A. Quéré, Acta Math. Acad. Sc. Hung. 21, 437- 444, 1970.

[P16] *Mille et un algorithmes pour les problèmes de cheminement dans les graphes*, RIRO Informatique 4 n° 3, 125-143, 1970.

[P17] *Survol de la théorie des langages*, journées AFCET, Grenoble, 1970.

[P18] *FACE, langage pour l'écriture des compilateurs*, avec F. Bellegarde et J. Maroldt, congrès AFCET, broch. 2, 5-21, 1970.

29. Par exemple, selon Google Scholar, pour l'article sur les bilangages paru en 1968 [P11], on trouve encore 37 citations postérieures à l'an 2000. Le lecteur pourra aussi lire l'introduction prémonitoire du livre de Livercy [26].

[P19] *Concerning the syntax of Algol 68*, Algol Bulletin 31, 16-27, 1970.

[P20] *Rapport d'évaluation Algol 68*, avec coauteurs, RIRO Informatique 5, n° 1, 15-106, 1971.

[P21] *Application de la théorie des ramifications au problème de l'équivalence structurale de deux C-grammaires*, RIRO Math. 5, n° 2, 130-136, 1971.

[P22] *L'algorithme d'analyse syntaxique de P. Broize*, RIRO Informatique, n° 2, 111-123, 1971.

[P23] *Problèmes de cheminement dans les graphes*, avec J.C. Derniame, Monographies d'informatique, Dunod, 1971.

[P24] *Les structures de données et leur représentation en mémoire*, cours à l'Ecole d'Eté de l'AFCET, 1971, puis, avec Marie-Claude Gaudel, IRIA Roquencourt, 270 p., 1977.

[P25] *La formalisation des langages de programmation*, Informatique et Sciences Humaines 9, n° 34, 71-86, 1971.

[P26] *Définition du langage algorithmique Algol 68*, avec le groupe Algol de l'AFCET, Act. Sc. et Ind. 1354, Hermann, 1972.

[P27] *Compilation*, cours à l'école d'été de l'AFCET, 1972.

[P28] *Analyse syntaxique*, cours à l'école d'été CEA-EDF-IRIA, 1973.

[P29] *On description languages for algorithms and programs*, rapport interne, 1973.

[P30] *Formalization of the notions of data, information and information structure*, in IFIP Working Conference Data Base Management Systems, Klimbie-Koffemann éd., North-Holland 149-168, `https://dblp.uni-trier.de/db/conf/ds/dbm74.html#Pair74`, et 1974 [30]

[P31] *Toute grammaire LL(k) est LR(k)*, RAIRO Informatique 8 n° 2, 59-62, 1974.

[P32] *Aspects de la théorie de la programmation*, avec coauteurs, cours à l'école d'été de l'AFCET, 1974.

[P33] *Rapport de l'ATP Informatique 1972 : Informatique théorique, programmes et données, Équipe associée 364,* Université de Nancy II. Éd, 1974.

[P34] *Réflexions sur la programmation*, journée SESORI sur la programmation, 13-18, 1974.

[P35] *Calculs, ensembles de calculs, équivalence de programmes*, Conv. Informatica Teorica 1973, Symposia Matematica 15, Academic Press, 35-54, 1975.

[P36] *Introduction à une méthode de programmation déductive*, avec J. Maroldt in L'enseignement de la programmation, IRIA, 1975.

[P37] *Manuel du langage algorithmique Algol 68*, P. Bacchus, J. André, C. Pair (éds), groupe Algol de l'AFCET, Act. Sc. et Ind. 1373, Hermann), 1975.

[P38] *Algorithms for checking consistency of attribute grammars*, in Proving and Improving programs, avec B. Lorho, IRIA Int. Coll., Arc et Senans, 29-54, 1975.

[P39] *Some proposals for a very high level language on a variable universe*, in New Directions in Algorithmic Languages (S. Schuman, ed.), W.G. 2.1. Conf., IRIA, 53- 70, 1975.

[P40] *Du problème à sa solution*, Colloque SESORI, Logique et Programmation, Le Bischenberg, 1975.

[P41] *Initiation à la programmation*, avec N. Carbonnel, CNTE, 1976.

[P42] *Les arbres en théorie des langages*, Colloque « Les arbres en algèbre et en programmation », Lille, 196-216, 1976.

[P43] *Inference for regular bilanguages*, in Formal Languages and programming (R. Aguilar, éd.), North-Holland, 15-30, 1976.

30. On peut aussi consulter la page dédiée à Claude Pair sur le site du *Digital Bibliography & Library Project,* `https://dblp.uni-trier.de/pid/63/4413.html`..

[P44] *Objectifs, réalisations et expériences de l'introduction de l'informatique dans l'enseignement secondaire en France*, Colloque sur Informatique et Enseignement, Neuchatel, 1977.

[P45] *La construction des programmes*, Journées informatiques de Nice et Ecole d'été de Montréal, 1977.

[P46] *Inference for regular bilanguages*, avec J. Berger, Journ. Comp. System Sciences 16, 100-122, 1978.

[P47] *MEDEE, a type of language for constructing programs*, avec coauteurs, Workshop on reliable software, Bonn 1978.

[P48] *La programmation : de l'énoncé au programme*, conférence invitée au congrès de l'AFCET, 1978.

[P49] *Correctnes proofs of syntax-directed processing description by attributes*, avec M. Amirchahy et D. Néel Journ. Comp. System Sciences 18 , 1-17, 1979.

[P50] *La construction des programmes*, RAIRO Informatique 13, 113-137, 1979.

[P51] *Construction de compilateurs basée sur une sémantique formelle*, avec M.C. Gaudel, Journées francophones, Genève, 83-101, 1979.

[P52] *The use of formal semantics to produce and prove compilers*, avec M.C. Gaudel, Workshop on semantics of programming languages, Bad Hohnef, 1979.

[P53] *Spécifications et langages de spécification*, Panorama des langages d'aujourd'hui, séminaire GROPLAN, Congrès, 1979.

[P54] *Computer Science, a new dimension of contemporary science*, avec M. Borillo in Scientific Culture in the Contemporary World, Scientia, 343-363, 1979.

[P55] *L'algorithmique, ou comment l'informatique amène à faire des mathématiques*, Conférence invitée au 4e séminaire, coll. de la Comm. Int. Enst Math., Luxembourg, 117-134, 1979.

[P56] *Some theorical aspects of program construction*, in Program Construction, International Summer School, Springer Lecture Notes in Comp. Sc. 69, 617-651, 1979.

[P57] *La production assistée du logiciel, introduction et perspectives*, Journées Francophones, Genève, 1-13, 1980.

[P58] *Types abstraits et sémantique algébrique des langages de programmation*, CRIN 80-R-011, 1980.

[P59] *Application of abstract data types to the définition of the semantics of programming languages*, conférence invitée à « Formalization of Programming Concepts », Peniscola, 1981.

[P60] *Claude Pair et Yves Le Corre, L'introduction de l'informatique dans l'éducation nationale,* Rapport au Ministre, `http://www.epi.asso.fr/revue/histo/h81_Pair-Le-Corre.htm`, octobre 1981.

[P61] *Abstract data types and algebraic semantics of programming languages*, Theor. Comp. Sc.18, 1 – 31, 1982.

[P62] *A systematic study of models of abstract data types*, avec M. Broy et M. Wirsing Theor comp. Sc.33, 139-174, 1984.

[P63] *Programmation*, par Amédée Ducrin (ouvrage collectif : M. Créhange, J.-P. Finance, J. Guyard, N. Hertschuh, C. Pair, M. Quéré, J. Souquières), tomes 1 et 2, Dunod Informatique, 1984.

[P64] *Informatique et enseignement : hier, aujourd'hui et demain*, Bulletin de l'association Enseignement public et Informatique, septembre 1987.

[P65] *Informatique et lutte contre l'échec scolaire,* in J.-M. Hoc, P. Mendelsohn, Les langages informatiques dans l'enseignement, Psychologie française 32, 1987.

[P66] *Construire les algorithmes*, avec R. Mohr et R. Schott, Dunod Informatique, 1988.

[P67] *Can computer help combat school failure* ?, C. Pair et coll., Computers in education, Elsevier, 1988.

[P68] *Programmation, langages et méthodes de programmation*, Le Travail humain 51, no 4, 1988.

[P69] *A tout CRIN : histoire d'un laboratoire*, Colloque sur l'Histoire de l'Informatique en France, Grenoble, 1988. in The History of a Laboratory, Annals of the History of Computing 12, 1990.

[P70] *Programming, programming languages and programming methods*, Psychology of Programming, Academic Press, 1990.

Thèses dirigées par Claude Pair

[T1] M. CUSEY. Construction d'un compilateur Algol pour IBM 1620, 3e cycle (1964).

[T2] R. ROMAC. Étude des méthodes de tri, 3e cycle (1964).

[T3] J. ANDRÉ. Contribution à la construction d'un compilateur Algol pour IBM 1620, 3e cycle (1965).

[T4] A. ÉMOND. Application de la notion de pile à des problèmes portant sur les chemins des graphes, 3e cycle (1965).

[T5] A. FLOC'H. Achèvement d'un compilateur Algol, traitement des procédures, 3e cycle (1966).

[T6] J.C. DERNIAME. Étude d'algorithmes pour les problèmes de cheminement dans les graphes finis, 3e cycle (1966).

[T7] H. SCHIVI. Étude de Cogent, langage de traitement de structures arborescentes, 3e cycle (1968).

[T8] A. QUÉRÉ. Étude des ramifications et des bilangages, 3e cycle (1969).

[T9] J. VILLARD. Emploi de PL/1 pour les problèmes de linguistique, 3e cycle (1969).

[T10] J.M. DIRAND. Les langages ATF-LMU, application aux problèmes linguistiques, mise en œuvre sur CII 10070, 3e cycle (1969).

[T11] J.M. MARTIN. Un mode d'exploitation du dossier médical, 3e cycle (1969).

[T12] J. CHABRIER. Etude d'une télétransmission par terminal, 3e cycle (1970).

[T13] J.M. LECLAIRE. Définition de la syntaxe des langages de programmation, 3e cycle (1970).

[T14] R. KHALIL. Essai de formalisation de la description des compilateurs, docteur- ingénieur (1970).

[T15] C. LEMAIRE. Système général d'analyse syntaxique, 3e cycle (1971).

[T16] P. LESCANNE. Étude de quelques théories des langages et généralisation du théorème de Kleene, 3e cycle (1971).

[T17] F. BELLEGARDE. Face, langage d'écriture de compilateurs, définition et implémentation, 3e cycle (1972).

[T18] J. MAROLDT. Définition de Face, langage pour l'écriture des compilateurs, implémentation d'un sous-ensemble, docteur-ingénieur (1972).

[T19] P. GERMAIN. Système de gestion et d'exploitation documentaire d'un corpus de dossiers médicaux, 3e cycle (1972).

[T20] R. MOHR. Modèle algébrique pour l'analyse syntaxique de figures, 3e cycle (1973).

[T21] N. CARBONELL. Rôle des fonctions récursives de ramifications dans la définition d'une langue naturelle, application à la syntaxe française, 3e cycle (1972).

[T22] J.B. JOUIN. Reconnaissance des fonctions primaires de la phrase anglaise, 3e cycle (1973).

[T23] J.C. DERNIAME. Le projet CIVA, un système de programmation modulaire, État (1974).

[T24] J.L. RÉMY. Structures d'information, formalisation des notions d'accès et de modification d'une donnée, 3e cycle (1974).

[T25] J.P. FINANCE. Contribution à la formalisation de la sémantique d'un langage de programmation, application à Algol 68, 3e cycle (1974).

[T26] P. MARCHAND. Étude et classification des bigrammaires, applications à l'étude des systèmes transfomationnels, 3e cycle (1974).

[T27] H. PISTRÉ. L'analyse syntaxique dans FACE, langage pour l'écriture des compilateurs, 3e cycle (1975).

[T28] J. JARAY. Le langage SNOBOL4, ses applications, son implémentation, 3e cycle (1975).

[T29] M. CRÉHANGE. Description formelle, représentation, interrogation des informations complexes : système Pivoines, État (1975).

[T30] A.M. RASSER. Outils d'aide à la mise au point d'un compilateur écrit dans le langage FACE, 3e cycle (1976).

[T31] M. MAZAUD. Système d'aide à la production de traducteurs, 3e cycle (1976).

[T32] J.J. GIRARDOT, F. MIREAUX. Réalisation d'un interprète complet du langage APL sur un mini-ordinateur, docteur-ingénieur (1978).

[T33] F. PRUSKER. Aspects théoriques et pratiques du tri par fusion sur disque, État (1977).

[T34] B. HUC. Mise en œuvre de la méthode de programmation déductive, docteur- ingénieur (1977).

[T35] J. BERGER. A study of inference for regular bilanguages, PH.D, University of Pensylvania (1977).

[T36] A. TISSERANT. Compilateur du langage Pascal sur mini-ordinateurs, réalisation sur Solar 16, docteur-ingénieur (1977).

[T37] A. COCHET-MUCHY. La production de programmes dans le projet SYGARE, docteur-ingénieur (1978).

[T38] P. NONN. Le système d'exploitation dans le projet SYGARE, docteur-ingénieur (1978).

[T39] P. LESCANNE. Étude algébrique et relationnelle des types abstraits et de leurs représentations, État (1979).

[T40] J.P. FINANCE. Étude de la construction des programmes : méthodes et langages de spécification et de réalisation de problèmes, État (1979).

[T41] J. MARIN-NAVARRO. Un método de programacion .- presentacion, implementacion, transporte, Universitad Complutense, Madrid (1979).

[T42] M. QUÉRÉ. Contribution à l'amélioration des processus d'enseignement, d'apprentissage et d'organisation de l'éducation : l'ordinateur outil et objet d'enseignement, application au projet SATIRE, État (1980).

[T43] M.C. GAUDEL. Génération et preuve de compilateurs basées sur une sémantique formelle des langages de programmation, État (1980).

[T44] P. DESCHAMP. Production de compilateurs à partir d'une description sémantique des langages de programmation : le système PERLUETTE, docteur-ingénieur (1980).

[T45] P. MARCHAND. Langages d'arbres, langages dans les algèbres libres, État (1981).

[T46] E. CHOURAQUI. Contribution à l'étude théorique de la représentation des connaissances. Le système symbolique Arches, État (1981).

[T47] J.L. RÉMY. Etude des systèmes de réécriture conditionnels et applications aux types abstraits algébriques, État (1982).

[T48] Radhia COUSOT. Fondements des méthodes de preuve d'invariance et de fatalité de programmes parallèles, État (1985).

Références

[1] A. V. Aho and J. D. Ullman, The Theory of Parsing, Translation, and Compiling, Vol. 1, Parsing. Prentice Hall, 1972.

[2] R. M. Amadio, Luca Cardelli, *Subtyping Recursive Types,* ACM Trans. Program. Lang. Syst. 15(4), 575-631, DOI :10.1145/155183.155231, 1993.

[3] J.-W. Backus et coll[31], *Report on the algorithmic language Algol 60*, Nümerische Mathematik 2, 1960.

[4] J. C. Boussard, *Étude et réalisation d'un compilateur Algol 60 sur calculatrice du type IBM 7090/7094 et 7040/44*, Thèse d'État Science appliquée, Grenoble, https://hal.archives-ouvertes.fr/tel-00009449/, 1964.

[5] J. C. Boussard, J. J. Duby (éds), *Rapport d'évaluation Algol 68*, Revue française d'informatique et de recherche opérationnelle R-1, 1971.

[6] J. Buffet, P. Arnal, A. Quéré *Définition du langage algorithmique Algol 68,* présent. et trad. française du Report on the algorithmic language Algol 68 éd. Hermann (Actualités scientifiques et industrielles), 1972.

[7] N. Chomsky, *On certain properties of grammars*, Information and Control 2, 1959.

[8] M. Créhange, *Structure du code de programmation*, Thèse 3e cycle, Nancy, 1961[32]. Et *Code de programmation*, Cahier n° 1 du groupement des utilisateurs scientifiques des ordinateurs IBM 650, octobre 1960.

[9] M. Créhange, M.-C. Haton, *L'informatique universitaire à Nancy : un demi-siècle de développement*, revue de la SIF 1024 no 3, pp. 59-74. 2014.

[10] O. J. Dahl, E.W. Dijkstra, C. A. H. Hoare, *Structured Programming*, Acad. Press, 1972.

[11] J.C. Derniame, *A propos du cheminement dans les graphes*, revue de la SIF 1024, N° 16, 2020.

[12] E. W. Dijkstra, *Go To Statement Considered Harmful*, CACM, Communications of the ACM 11, 3. 147-148, 1968.

[13] J.-P. Finance, J.-L. Rémy, *Structure d'information et sémantique d'un langage de programmation,* école d'été de l'AFCET GRENADE 1973.

[14] J.-P. Finance, P. Lescanne, P. Marchand, R. Mohr, C. Pair, A. Quéré, J-L. Rémy, *Aspect de la Théorie de la Programmation*, Cours de l'école d'été d'informatique, Tarbes, (imprimé à l'IREM de Nancy), 1974.

[15] L. Fossier, M. Créhange, *Un essai de traitement sur ordinateur des documents diplomatiques du Moyen Age*, ANNALES Economies, Sociétés, Civilisations) 1, 1970.

[16] A. Gerbier, Mes premières Constructions de programmes, LNCS 55, Springer- Verlag, 1977.

[17] Anna Gram, *Raisonner pour programmer*, Dunod informatique, 1993.

[18] Y. Gurevich, Evolving algebras, Lipari guide. Specification and validation methods, 9-36, https://dblp.uni-trier.de/db/conf/asm/eapl993.html#Gurevich93, 1993.

[19] Y. Gurevich : Sequential abstract-state machines capture sequential algorithms. ACM Trans. Comput. Log. 1(1) : 77-111, https://dblp.uni-trier.de/db/journals/tocl/tocl1.html#Gurevich00, 2000.

31. Le comité de définition comprend un français, B. Vauquois, professeur à Grenoble.

32. Dans l'article du journal du CNRS du 23.09.2016 par Martin Koppe, Pierre Mounier-Kuhn signale qu'il s'agit « de la première thèse de France en informatique ».

[20] S. C. Johnson, Yacc : Yet Another Compiler-Compiler, AT&T Bell Laboratories Technical Reports. AT&T Bell Laboratories Murray Hill, New Jersey 07974 (32), http://dinosaur.compilertools.net/yacc/, 1975, Retrieved 31 October 2014.

[21] D. E. Knuth. A proposal for input-output conventions in Algol 60. Commun. ACM 7, 5, 273-283, May 1964.

[22] D. E. Knuth, *Literate Programming*, Californie, Stanford University Center for the Study of Language and Information, (ISBN 978-0937073803), 1992.

[23] D. E. Knuth, The Art of Computer Programming, Addison-Wesley, https://www-cs-faculty.stanford.edu/%7Eknuth/taocp.html, à partir de 1968.

[24] E. Lazard, P. Mounier-Kuhn, *Histoire illustrée de l'informatique*, 2e édition, EDP Sciences, 2019.

[25] P. Lescanne, *« La science informatique »*, dans Encyclopédie Illustrée de la Lorraine, Histoire des Sciences et des Techniques, vol. Sciences exactes, Éditions Serpenoises, p. 105-116, 1996.

[26] C. Livercy (nom collectif : J.-P. Finance, M. Grandbastien, P. Lescanne, P. Marchand, R. Mohr, A. Quéré, J.-L. Rémy), *Théorie des programmes : schémas, preuves, sémantique*, Dunod Informatique 1978 [33].

[27] R. Mohr, *Généralisation de la notion de langage à contexte libre. Application à l'analyse syntaxique de figures*, RAIRO Informatique théorique, tome 9, no R2, p. 55-88, 1975.

[28] P. Naur.(éd.), *Revised Report on the algorithmic language Algol 60*, Comm. ACM, 1963.

[29] R. Péter, *Recursive Functions*. Traduit par István Földes. New York : Academic Press, 1967 [34].

[30] R. Péter, *Die Pair-schen freien Binoïden als Speziallfälle des angeordneten freien holomorphen Mengen*, Eötvös Lorand Universität Budapest, 1968, publié dans Acta Mathematica Academiae Scientiarum Hungaricae Tomus 21 (3—4), pp. 29--313, 1970.

[31] B. C. Pierce, *Types and Programming Languages*, The MIT Press Cambridge, Massachusetts, ISBN 0-262-16209-1, 2002.

[32] M. Quéré, *Modèle mathématique d'un système d'enseignement*, Computers in education, North Holland, 1975.

[33] D. Sangiorgi, *On the origins of bisimulation and coinduction* [35], ACM Trans. Program. Lang. Syst. 31(4), 15 :1-15 :41, https://dblp.uni-trier.de/db/journals/toplas/toplas31.html#Sangiorgi09, 2009.

[34] J-D Warnier, B. M. Flanagan, Entraînement à la construction des programmes d'informatique : Principes et exercices pratiques, Les Éditions d'organisation, 1970.

[35] A. Van Wijngaarden, B. J. Mailloux, J. E. L. Peck, C. H. A. Koster (éds), *Report on the Algorithmic Language Algol 68*, Springer, 1969 et Traduction française par le groupe franco-belge Algol P. Arnal, J. Buffet, A. Quéré, éds.), Actualités scientifiques et industrielles, Hermann, 1972.

[36] N. Wirth, Helmut Weber, *EULER : a generalization of Algol and its formal definition*, Part 1. Commun. ACM 9(1), 13-25 (1966) et Part II. Commun. ACM 9(2), 89-99, 1966.

33. http://denif.ens-lyon.fr/data/programmation_enslyon/2007_sem2/biblio/Livercy.pdf.

34. Édition originale en 1951.

35. Une intéressante étude historique qui ignore cependant la contribution de Claude Pair.

Travaux, publications et thèses encadrées par Claude Pair

Travaux de C. PAIR

L'objet de mes recherches est la programmation. Leur approche consiste en général à partir de problèmes concrets, notamment la compilation [3,5,T1,T3,T5,T30,T35], l'écriture des programmes [32,44], l'emploi des structures de données [21], pour construire des théories aidant à les formuler et à les résoudre. Les méthodes sont algébriques et logiques.

Les aspects syntaxiques de la compilation des langages de programmation m'ont amené, de 1963 à 1965 surtout, à étudier l'analyse syntaxique [1,6,T15]. J'ai mis notamment en évidence les méthodes de précédence [2], ultérieurement envisagées par WIRTH et WEBER. Puis, pour formuler et étudier plus précisément ce problème de l'analyse syntaxique, j'ai, de 1966 à 1968 notamment, étudié les langages d'arbres, souvent réétudiés depuis, en introduisant les notions de ramification et bilangage [8,T8,11,12], et en utilisant pour cela des méthodes algébriques.

L'application des méthodes syntaxiques à la compilation [T14] a conduit à la définition d'un langage d'écriture de compilateurs, FACE [15,T17,T18,T26,T29]. D'autre part, en liaison avec la théorie des bilangages, elle a amené à étudier la méthode des attributs de KNUTH de deux points de vue : celui des algorithmes de détection de la circularité [34] et celui des preuves de définition d'attributs [45] (premier travail du genre).

Outre leur application à la description de la syntaxe des langages naturels [T2l] ou de programmation [T13], et des démonstrations élégantes de résultats de théorie des langages [18,19,25,28,38,T44], les bilangages ont permis d'introduire de nouvelles méthodes d'inférence grammaticale [39,42,T34]. Ils ont eu ensuite deux sortes de prolongements : l'un dans l'étude de la sémantique algébrique des langages ; l'autre par l'extension de leur théorie à des algèbres plus générales [T16] : une application est l'introduction des méthodes syntaxiques en reconnaissance des formes ([T20] et travaux de l'équipe de J.P. HATON).

Cette extension à des algèbres plus générales est proche de ce qu'on nomme aujourd'hui les types abstraits algébriques [T38], qui formalisent les structures de données. L'étude des structures de données a d'ailleurs été une autre direction de mes travaux : structure de pile [6], cheminement dans les graphes [T4,7,13,20,T6] . Ce dernier travail, employant également des méthodes algébriques, se rattache à l'algorithmique (et à la complexité des algorithmes) : il introduisait le mode d'exploration des graphes dit aujourd'hui "en profondeur d'abord" puisqu'il utilise une pile pour l'exploration.

Des travaux précurseurs aux types abstraits algébriques ont été effectués d'une autre manière, dont la motivation a été la construction des programmes et la définition des bases de données [Tll,T19,T28]. En effet, la recherche d'une formalisation pour les structures de données a amené, autour de 1974, à

considérer des systèmes formels du premier ordre où le seul prédicat est l'égalité. Cette théorie des "structures d'information" [27,T24] s'accompagnait de celle de leurs modifications, c'est-à-dire du passage d'une information à une autre de même structure.

Elle a donné lieu à deux genres d'applications bien liées : une méthode de description de la sémantique des langages de programmation [T25] , une méthode et un outil de description formelle de compilateurs [47,48,T42,T43] .

Il est cependant apparu que l'emploi de deux niveaux de description, l'un pour les aspects instantanés, l'autre pour leurs modifications dans le temps, nuisait à l'unité de la théorie. Aussi, dans le cadre des types abstraits, une nouvelle description de la sémantique a été proposée [54,T56], qui paraît conduire à des descriptions plus simples et plus formelles que la sémantique dénotationnelle. Il s'agit d'une "sémantique algébrique" car elle est faite d'un type abstrait dont les modèles sont des algèbres. Cette orientation vers la sémantique ne date d'ailleurs pas d'hier, puisque des travaux dans ce domaine avaient été faits dès 1965-66 [4], et d'autres, introduisant des méthodes issues de la théorie des langages, en 1973 [30] .

Mais l'étude de la programmation ne se réduit pas à celle des aspects syntaxiques ou sémantiques des langages. Outre l'étude de langages particuliers [10,T7,T9,Tl0,T27], et celle d'Algol 68 m'a occupé plusieurs années [9,16,17,23,33], on s'aperçoit de plus en plus que les méthodes de programmation sont essentielles. Il convient de séparer divers aspects, à traiter dans des étapes différentes, et plus ou moins susceptibles d'automatisation ; il convient de limiter le domaine des notions dynamiques, sources d'erreurs et de difficultés dans la construction et la validation. Ces idées sont à la base de la méthode de programmation déductive introduite en [26,31,41,43,46,T33,T40], et depuis lors élargie vers la spécification [35,36,49,T39] et employée dans l'enseignement [37]. Des travaux plus appliqués ont d'ailleurs été orientés vers l'introduction de l'informatique dans l'enseignement [40,51,T4l].

PUBLICATIONS

1. *Justification théorique de l'utilisation des piles en compilation*, CR AS 257 (1963), 3278-3281.
2. *Arbres, piles et compilation*, RFTI - Chiffres 7 (1964), n°3, 199-216.
3. *Sur la détermination de la correspondance paramètre formel - paramètre effectif des procédures Algol*, congrès AFIRO (1964), 157-162.
4. *Essai de description de la sémantique des langages de programmation*, séminaire AFIRO et rapport interne (1964) .
5. *Description d'un compilateur Algol*, European Région 1620 Users Group, Mannheim (1965).
6. *Etude de la notion de pile, application à l'analyse syntaxique*, thèse, Nancy (1965).
7. *Sur des algorithmes pour les problèmes de cheminement dans les graphes, Théorie des graphes*, Dunod - Gordon - Breach (1966), 271-300.
8. *Définition et étude des bilangages réguliers*, Inf. and Control 13 (1968) 565-593, avec A. QUERE.
9. *Introduction à Algol 68*, RIRO Informatique 3 (1969) n°3, 17-52.
10. *Algol 60*, Techniques de l'Ingénieur H 2160 (1970).
11. *Sur des notions algébriques liées à l'analyse syntaxique*, RIRO Math.4 (1970) n° 3, 3-29.
12. *Sur les fonctions primitives récursives de ramifications*, Acta Math. Acad. Sc. Hung. 21 (1970), 437-444, avec A. QUERE.
13. *Mille et un algorithmes pour les problèmes de cheminement dans les graphes*, RIRO Informatique 4 (1970) n° 3, 125-143.
14. *Survol de la théorie des langages*, journées AFCET, Grenoble (1970).
15. *FACE, langage pour l'écriture des compilateurs*, congrès AFCET (1970) broch. 2, 5-21, avec F. BELLEGARDE et J. MAROLDT.
16. *Concerning the syntax of Algol 68*, Algol Bulletin 31 (1970), 16-27.
17. *Rapport d'évaluation Algol 68*, RIRO Informatique 5 (1971), n° 1, 15-106, avec coauteurs.
18. *Application de la théorie des ramifications au problème de l'équivalence structurale de deux C-grammaires*, RIRO Math. 5 (1971) n° 2, 130-136.
19. *L'algorithme d'analyse syntaxique de P. BROISE*, RIRO Informatique, (1971) n° 2, 111-123.
20. *Problèmes de cheminement dans les graphes*, Monographies d'informatique, Dunod (1971), avec J.C. DERNIAME.

21. *Les structures de données et leur représentation en mémoire*, cours à l'Ecole d'Eté de 1'AFCET (1971), publié par IRIA (1977) avec la collaboration de MC. GAUDEL.

22. *La formalisation des langages de programmation*, Informatique et Sciences Humaines 9 (1971), n° 34, 71-86.

23. *Définition du langage algorithmique Algol 68*, Act. Sc. et Ind. 1354, Hermann (1972), avec le groupe Algol de 1'AFCET.

24. *Compilation*, cours à l'Ecole d'Eté de 1'AFCET (1972).

25. *Analyse syntaxique*, cours à l'Ecole d'Eté CEA-EDF-IRIA (1973).

26. *On description languages for algorithms and programs*, rapport interne (1973) .

27. *Formalization of the notions of data, information and information structure*, in_ Data Base Management Systems (KLIMBIE-KOFFEMANN éd.), North-Holland (1974) 149-168.

28. *Toute grammaire LL(k) est LR(k)*, RAIRO Informatique 8 (1974) n° 2, 59-62.

29. *Aspects de la théorie de la programmation*, cours Ecole d'Eté AFCET (1974), avec coauteurs.

30. *Calculs, ensembles de calculs, équivalence de programmes*, Conv. Informatica Teorica 1973, Symposia Matematica 15, Academic Press, (1975), 35-54.

31. *Introduction à une méthode de programmation déductive*, rapport interne (1975) avec J. MAROLDT.

32. *Réflexions sur la programmation*, journée SESORI sur la programmation (1974), 13-18.

33. *Manuel du langage algorithmique Algol 68*, Act. Sc. et Ind. 1373, Hermann (1975) avec le groupe Algol de 1'AFCET.

34. *Algorithms for checking consistency of attribute grammars*, in Proving and Improving programs, IRIA Int. Coll. Arc et Senans (1975), 29-54, avec B. LORHO.

35. *Some proposals for a very high level language on a variable universe*, in New Directions in Algorithmic Languages (S. SCHUMAN, ed.) W.G. 2.1. Conf., IRIA (1975), 53-70.

36. *Du problème à sa solution*, Colloque SESORI, Logique et Programmation, Le Bischenberg (1975).

37. *Initiation à la programmation*, CNTE (1976), avec N. CARBONNEL.

38. *Les arbres en théorie des langages*, Colloque "Les arbres en algèbre et en programmation", Lille (1976), 196-216.

39. *Inference for regular bilanguages*, in Formal Languages and programming (R. AGUILAR, éd.), North-Holland (1976), 15-30.

40. *Objectifs, réalisations et expériences de l'introduction de l'informatique dans l'enseignement secondaire en France*, Colloque sur Informatique et Enseignement, Neuchatel (1977).

41. *La construction des programmes*, Journées informatiques de Nice (1977) et Ecole d'été de Montréal (1977) .

42. *Inference for regular bilanguages*, Journ. Comp. System Sciences 16 (1978), 100-122, avec J. BERGER.

43. *MEDEE, a type of language for constructing programs*, Workshop on reliable software, Bonn (1978), avec coauteurs.

44. *La programmation : de l'énoncé au programme*, conférence invitée au congrès de 1'AFCET (1978).

45. *Correctnes proofs of syntax-directed processing description by attributes*, Journ. Comp. System Sciences 18 (1979), 1-17, avec M. AMIRCHAHY et D. NEEL.

46. *La construction des programmes*, RAIRO Informatique 13 (1979), 113-137.

47. *Construction de compilateurs basée sur une sémantique formelle*, Journées francophones, Genève (1979), 83-101, avec M.C. GAUDEL.

48. *The use of formal semantics to produce and prove compilers*, Workshop on semantics of programming languages, Bad Hohnef (1979), avec M.C. GAUDEL.

49. *Spécifications et langages de spécification*, Panorama des langages d'aujourd'hui, séminaire GROPLAN, Congrès (1979).

50. *Computer Science, a new dimension of contemporary science*, in Scientific Culture in the Contemporary World, Scientia (1979), 343-363, avec M. BORILLO.

51. *L'algorithmique, ou comment 1'informatique amène à faire des mathématiques*, Conférence invitée au 4e séminaire, coll. de la Comm. Int. Enst Math., Luxembourg (1979), 117-134.

52. *Some theorical aspects of program construction*, in_ Program Construction, International Summer School, Springer Lecture Notes in Comp. Sc. 69 (1979), 617-651.

53. *La production assistée du logiciel, introduction et perspectives*, Journées Francophones, Genève (1980), 1-13.

54. *Types abstraits et sémantique algébrique des langages de programmation*, CRIN 80-R-011 (1980).

55. *Application of abstract data types to the définition of the semantics of programming languages*, conférence invitée à "Formalization of Programming Concepts", Peniscola (1981).

56. *Abstract data types and algebraic semantics of programming languages*, Theor. Comp. Sc.18 (1982), 1 - 31.

57. *A systematic study of models of abstract data types*, Theor comp. Sc.33 (1984) 139-174, avec M. BR0Y et M. WIRSING.

THESES SOUTENUES SOUS MA DIRECTION

T 1. M. CUSEY *Construction d'un compilateur Algol pour IBM 1620*, 3e cycle (1964);

T 2. R. ROMAC *Etude des méthodes de tri*, 3e cycle (1964).

T 3. J. ANDRE *Contribution à la construction d'un compilateur Algol pour IBM 1620*, 3e cycle (1 965).

T 4. A. EMOND *Application de la notion de pile à des problèmes portant sur les chemins des graphes*, 3e cycle (1 965) .

T 5. A. FLOCH *Achèvement d'un compilateur Algol, traitement des procédures*, 3e cycle (1966).

T 6. J.C. DERNIAME *Etude d'algorithmes pour les problèmes de cheminement dans les graphes finis*, 3e cycle (1 966) .

T 7. H. SCHIVI *Etude de Cogent, langage de traitement de structures arborescentes*, 3e cycle (1968),

T 8. A. QUERE *Etude des ramifications et des bilangages*, 3e cycle (1969).

T 9. J. VILLARD *Emploi de PL/1 pour les problèmes de linguistique*, 3e cycle (1969).

T10. J.M. DIRAND *Les langages ATF-LMU, application aux problèmes linguistiques, mise en oeuvre sur CII 10070*, 3e cycle (1969).

Tll. J.M. MARTIN *Un mode d'exploitation du dossier médical*, 3e cycle (1969).

T12. J. CHABRIER *Etude d'une télétransmission par terminal*, 3e cycle (197 0).

T13. J.M. LECLAIRE *Définition de la syntaxe des langages de programmation*, 3e cycle (1970).

T14. R. KHALIL *Essai de formalisation de la description des compilateurs*, doct.ingénieur (1970).

T15. C. LEMAIRE *Système général d'analyse syntaxique*, 3e cycle (1971) .

T16. P. LESCANNE *Etude de quelques théories des langages et généralisation du théorème de Kleene*, 3e cycle (1971)

Tll. F. BELLEGARDE *Face, langage d'écriture de compilateurs, définition et implémentation*, 3e cycle (1972).

T18. J. MAROLDT *Définition de Face, langage pour l'écriture des compilateurs, implémentation d'un sous-ensemble*, docteur-ingénieur (1972).

T19. P. GERMAIN *Système de gestion et d'exploitation documentaire d'un corpus de dossiers médicaux*, 3e cycle (1972).

T20. R. MOHR *Modèle algébrique pour l'analyse syntaxique de figures*, 3e cycle (1973).

T21. N. CARBONELL *Rôle des fonctions récursives de ramifications dans la définition d'une langue naturelle, application à la syntaxe française*, 3e cycle (1 972) .

T22. J.B. JOUIN *Reconnaissance des fonctions primaires de la phrase anglaise*, 3e cycle (1973).

T23. J.C. DERNIAME *Le projet CIVA, un système de programmation modulaire*, Etat (1974).

T24. J.L. REMY *Structures d'information, formalisation des notions d'accès et de modification d'une donnée*, 3e cycle (1974).

T25. J.P. FINANCE *Contribution à la formalisation de la sémantique d'un langage de programmation, application à Algol 68*, 3e cycle (1974).

T26. H. PISTRE *L'analyse syntaxique dans FACE, langage pour l'écriture des compilateurs*, 3e cycle (1975).

T27. J. JARAY *Le langage SNOBOL4, ses applications, son implémentation*, 3e cycle (1975).

T28. M. CREHANGE *Description formelle, représentation, interrogation des informations complexes : système Pivoines*, Etat (1975).

T29. A.M. RASSER *Outils d'aide à la mise au point d'un compilateur écrit dans le langage FACE*, 3e cycle (1 976).

T30. M. MAZAUD *Système d'aide à la production de traducteurs*, 3e cycle (1 976) .

T31. J.J. GIRARDOT, F. MIREAUX *Réalisation d'un interprète complet du langage APL sur un mini-ordinateur*, docteur-ingénieur (1978).

T32. F. PRUSKER *Aspects théoriques et pratiques du tri par fusion sur disque*, Etat (1977).

T33. B. HUC *Mise en œuvre de la méthode de programmation déductive*, docteur-ingénieur (1977).

T34. J. BERGER *A study of inference for regular bilanguages*, PH.D, University of Pensylvania (1977).

T35. A. TISSERANT *Compilateur du langage Pascal sur mini-ordinateurs, réalisation sur Solar 16*, docteur-ingénieur (1977).

T36. A. COCHET-MUCHY *La production de programmes dans le projet SYGARE*, docteur-ingénieur (1978).

T37. P. NONN *Le système d'exploitation dans le projet SYGARE*, docteur-ingénieur (1978).

T38. P. LESCANNE *Etude algébrique et relationnelle des types abstraits et de leurs représentations*, Etat (1979).

T39. J.P. FINANCE *Etude de la construction des programmes : méthodes et langages de spécification et de réalisation de problèmes*, Etat (1979).

T40. J. MARIN-NAVARRO *Un método de programacion .- presentacion, implementacion, transporte*, Universitad Complutense, Madrid (1979).

T41. M. QUERE *Contribution à l'amélioration des processus d'enseignement, d'apprentissage et d'organisation de l'éducation : l'ordinateur outil et objet d'enseignement, application au projet SATIRE*, Etat (1980) .

T42. M.C. GAUDEL *Génération et preuve de compilateurs basées sur une sémantique formelle des langages de programmation*, Etat (1980).

T43. P. DESCHAMP *Production de compilateurs à partir d'une description sémantique des langages de programmation : le système PERLUETTE*, docteur-ingénieur (1980).

T44. P. MARCHAND *Langages d'arbres, langages dans les algèbres libres*, Etat (1981).

T45. E. CHOURAQUI *Contribution à l'étude théorique de la représentation des connaissances. Le système symbolique Arches*, Etat (1981)

T46. J.L. REMY *Etude des systèmes de réécriture conditionnels et applications aux types abstraits algébriques*, Etat (1982).

T47. Radhia COUSOT Thèse d'état 1985 UHP Nancy.