Préface

La grande majorité des codes scientifiques a été développée en langage Fortran il y a en moyenne une vingtaine d'années. Néanmoins, il faut analyser l'historique de l'informatique scientifique pour comprendre que la démarche de ce livre se situe tout naturellement dans la continuité de ces développements informatiques. Elle traduit une synergie permanente entre l'évolution des logiciels et l'évolution des moyens informatiques. On pourrait distinguer cinq périodes distinctes : les balbutiements, la prise de contrôle par les scientifiques, le dialogue homme-machine, la maturité et pour finir la maîtrise des architectures, bases de données et interfaces.

Les balbutiements (1960 – 1970)

Au commencement, est la règle à calcul ; c'est l'outil de base indispensable pour l'ingénieur. Le plus gros des conceptions se fait sur la base de calculs dits « coin de table » consistant à résoudre des systèmes d'équations à la main. Cela se passe évidemment avant le développement des calculs de RDM (Résistance des Matériaux). Jusqu'en 1975, les élèves ingénieurs maîtrisent la règle à calcul et s'en servent pour l'obtention de leur diplôme. L'informatique est entrée à l'université en France entre 1965 et 1970 avec l'arrivée de l'IBM 1130. Pour nous, étudiants, cela reste une curiosité car cela ne fait pas partie de notre cursus. Les plus curieux commencent à programmer des systèmes de résolution de n équations à n inconnues, la nuit, car le jour l'ordinateur est à la disposition des scientifiques. Le folklore des nuits blanches, le bruit des perforatrices, les piles de cartes perforées, les clignotements des voyants qui signalent le plantage de l'ordinateur. F6, F8, F12 clignotent… c'est un « overflow ». On redémarre l'ordinateur. Il faut dire que la mémoire est vite saturée ! Une matrice 20 x 20 est utilisable à condition de jongler avec la taille mémoire. Entre 1960 et 1970 sont donc apparus les premiers « logiciels » FORTRAN de calculs matriciels, principalement dans l'aéronautique.

La prise de contrôle par les scientifiques (1970 – 1976)

Les ordinateurs progressent vite. En 1974, Control Data et IBM proposent des matériels autorisant des tailles mémoires suffisantes pour faire tourner de gros calculs aux éléments finis : Thermo-mécanique, neutronique, thermo-hydraulique. Dans le domaine du

nucléaire et de l'aéronautique, des logiciels aux noms plus farfelus les uns que les autres permettent de remplacer les calculs de « coin de table ». Les jeunes ingénieurs, pour les calculs de dimensionnement, sont obligés de se mettre journellement devant la perforatrice ou « puncheuse » pour préparer les entrées des calculs en lecture formatée. Une erreur de colonne et c'est le désastre. Un coursier vient prendre vos paquets de cartes et les emmène à l'informatique, salle à part climatisée à l'ambiance feutrée. Les jeux de cartes sont ingérées par le lecteur de carte à grand débit, l'ordinateur cliquette et vous recrache le précieux listing. Le coursier vous ramène le listing et les cartes après une demi-journée. Vous pouvez enfin analyser vos résultats. Ceci est le meilleur des cas. On suppose à ce stade que le lecteur haut débit ne vous a pas mangé une partie de vos cartes perforées, que le coursier n'a pas glissé dans une flaque d'eau, que l'imprimante n'a pas eu ses vapeurs entre l'encre qui s'épuise et les rouleaux qui déchiquettent le papier…. Concernant les cartes dans la flaque d'eau, nous vous conseillons de procéder à un séchage sur le radiateur. Vous obtenez alors, lorsque c'est à point, une carte en forme de tôle ondulée que le lecteur, très susceptible, ne voudra pas relire sans une indigestion. On peut mettre de la bande adhésive en cas de déchirement mais en faisant bien attention à ne pas recouvrir les trous. Vous passez alors le précieux carton reconstitué dans la perforatrice en mode « reproduction ». Si tout se passe bien, vous obtenez un clone de la carte initiale. Une relecture est néanmoins conseillée, à condition que vous sachiez lire le langage « perforé » ou le braille…

Maintenant, vous pouvez soumettre à nouveau votre paquet (ou jeu) de cartes au lecteur qui l'ingurgite gloutonnement. Les calculs commencent, tout va bien … jusqu'à ce que le calcul « plante ». Nouvel exercice : la recherche de « bug » dans la carte de « debuggage » qu'on récupère sur le listing, bande de papier de 37,9 cm de large et perforée sur les côtés (je dis cela pour ceux qui n'en n'ont jamais vu). La « map » de « debuggage » consiste en une série de chiffres qui vous donnent le contenu des 100 adresses encadrant celle du plantage, et tout cela en système octal et non décimal où 10 est égal à 8. Le jeu consiste à retrouver la dernière instruction qu'a exécutée le programme, et dans quel sous-programme. Ensuite on remonte de proche en proche pour reconstituer le cheminement de la bête. Il faut se mettre à raisonner comme l'ordinateur et surtout ne pas lui prêter une intelligence qu'il n'a pas. Combien de fois n'a-t-on pas entendu « il a fait ceci ou il a fait cela » en parlant du calculateur ? Ne pas oublier que l'ordinateur ne fait que ce qu'on le programme à faire, et se contente de nous mettre face à nos propres erreurs : Veiller à l'humilité avant tout. Précisons que le développement de logiciel est déconseillé aux personnes sujettes à la dépression. Lorsque tout fonctionne c'est l'euphorie, mais le problème c'est qu'il y a toujours un « bug » tapi dans l'ombre des dizaines ou centaines de milliers d'instructions dont vous avez la responsabilité. Celui-ci surgira toujours au moment où tout semble vous réussir et il faudra des heures, voire parfois des jours pour le débusquer. Mais, revenons à nos moutons.

Entre la soumission des calculs et le retour des résultats, les jeunes ingénieurs se font la main en développant des petits utilitaires, parfois farfelus mais aussi très utiles par la suite. Cela nous a permis d'acquérir une forte maîtrise du développement informatique. Nous assurions à cette époque près de 95 % de l'expertise informatique pour nos autres collègues. Avec l'évolution des tailles mémoires, nous avons très rapidement eu la capacité de sauvegarder des versions de nos logiciels en format « UPDATE ». Cet utilitaire était déjà très performant et permettait de ne gérer les évolutions de version qu'avec un jeu de cartes perforées de modification. Nos logiciels de l'époque représentaient environ 4000 à 6000 cartes perforées, c'est à dire quatre tiroirs.

En 1976, nous voyons apparaître les premiers terminaux à écran. Non ils ne sont pas dans les bureaux, ils sont placés dans des placards en bout des couloirs. Ils permettent néanmoins aux plus acharnés de suivre les pérégrinations de leur « job » sur l'ordinateur, mais par contre pas d'intervenir dans les séquences. Avec trois consoles de ce type, l'ordinateur commence à ramer. Malgré tout cela, nous continuons à faire évoluer nos logiciels de simulation pour y mettre toujours plus de physique, en FORTRAN bien sûr. Chacun a sa version du logiciel. La tentation est forte de « bidouiller » pour avoir de meilleurs résultats, mais avec toute l'honnêteté scientifique dont nous savons faire preuve, bien évidemment. C'est la grande époque des logiciels à « tournevis ». Une autre curiosité est l'informaticien de profession qui gère principalement le matériel et parle une sorte de patois local et touristique, mélange entre le français et l'anglais prononcé à la bretonne. C'est en général un gars très sympathique au demeurant qui, lorsqu'il vous a à la bonne, vous donne quelques tuyaux pour résoudre votre problème. Dans l'ensemble, c'est à vous de vous débrouiller. Les logiciels disposent en général de peu de documentation, quelques informations sur les équations, et un manuel d'utilisation. L'Assurance de la Qualité nous oblige à garder mémoire de nos calculs et surtout maintenir la capacité de refaire les calculs dix, vingt, voire quarante ans après en retrouvant les mêmes résultats. Heureusement il y a la bande magnétique, sur laquelle on sauvegarde la version « update » du logiciel en carte perforée, la version binaire absolue et les entrées des calculs. Grand bien nous en a pris. A ce jour, j'aimerais savoir combien de bandes magnétiques peuvent être relues ? Les supports magnétiques ne sont alors pas assez fiables et les matériels ont aujourd'hui trop évolué pour les reprendre. Pour ma part, j'ai tenu bon avec mes cartes perforées jusqu'en 1985 jusqu'à l'arrivée de supports et de procédures de sauvegarde plus fiables. La taille mémoire et les temps calculs restent à toute époque un souci permanent. Les meilleurs développeurs maîtrisent la gestion des tailles mémoire, quelques centaines de kilos octets dans le meilleur des cas, avec l'utilisation des options OVERLAY, les segmentations, les premières gestions dynamiques. Les ordinateurs commencent à être multitâches, capables de gérer plusieurs logiciels en exécution simultanée.

Le dialogue homme-machine (1978-1985)

La puissance des ordinateurs continue à s'accroître, et surtout les volumes mémoire. Ceci permet progressivement de gérer par le biais des terminaux qui se multiplient, les logiciels et les soumissions de calcul à partir de ceux-ci. Nos fichiers peuvent maintenant être gérés avec des mots de passe. Pour des raisons d'Assurance de la Qualité, on prend conscience qu'il faut gérer des versions de référence des logiciels, et surtout les documenter. C'est un travail de titan car il faut reconstituer les organigrammes à partir du fortran, avec ciseaux et colle. Les plus malins développent des logiciels spécialement pour cela : pour exemple le logiciel ORGIE reconstitue l'organigramme à partir du listing Fortran, l'utilitaire JERZY que j'ai moi-même développé, reconstitue la traçabilité des variables transmises par « COMMON » par balayage des « MAP » de compilation. De plus, on développe des systèmes de lecture sans format plus performants les uns que les autres : DIP (un des plus ancien), LSD (génération 68 oblige !) développé par des neutroniciens et capable de gérer des chaînes de caractères… A ce stade, une partie des logiciels scientifiques développés dans le cadre de l'AQ a un gestionnaire pour la version de référence, des utilisateurs et des développeurs. On commence à se soucier de la portabilité des logiciels, principalement entre les ordinateurs gérant à 8 digits (IBM) et ceux à 10 digits (CDC). En effet, chaque constructeur a un compilateur avec des options non standard du FORTRAN77. Il faut alors mettre l'ensemble des logiciels en compatibilité avec le FORTRAN77 standard. La documentation des logiciels se résume au manuel utilisateur et un descriptif reconstitué du logiciel. On commence à définir le principe de l'Assurance Qualité des logiciels vers 1985. Dans la même période, les possibilités de tracés graphiques se multiplient et chacun développe des logiciels pour exploiter ses propres résultats graphiquement : plume levée, plume baissée … et maintenant la légende…

La maturité (1985-1998)

Les capacités et performances des ordinateurs évoluent toujours plus vite. Les possibilités se multiplient. Les consoles « intelligentes » apparaissent sur lesquelles on peut stocker les informations, gérer les entrées-sorties d'un ordinateur central ou même plus tard faire les calculs directement sur le terminal lui-même. Le calcul vectoriel proposé par la génération CRAY permet de réduire les temps de calcul et donc d'accroître les capacités des logiciels. Certains scientifiques commencent à s'intéresser aux nouveaux langages. C'est le cas d'EDF qui tente le développement de COCAINE pour remplacer son logiciel de thermomécanique aux éléments finis ALI BABA. Malheureusement, ce projet a dû arriver trop tôt et n'est pas allé à terme. On commence à parler des règles de développement en V où on écrit au préalable ce qu'on va faire et comment on va valider. Côté combustible nucléaire, nous prenons la décision de réécrire complètement un logiciel pour remplacer l'ancien. Le client écrit un cahier des charges. Nous écrivons ensuite les spécifications de développement pour répondre à ce cahier des charges. C'est une nouvelle démarche pour les développeurs et il y a de la résistance ! On veille à la modularité des logiciels. Chaque

modèle physique fait l'objet d'un module indépendant avec une possibilité de l'activer et de le tester séparément. On multiplie les cartes commentaires avec le souci de la maintenance ultérieure. Chaque nouveau développement (option, modèle) fait l'objet d'un document de spécification. Les « savoir-faire » métier sont décrits le plus possible. Pour chaque version du logiciel il y a une mise à jour du manuel de réalisation (descriptif), du manuel utilisateur, du manuel de validation et d'un dossier des tests unitaires. Les pré et post-processeurs sont clairement identifiés et séparés du logiciel de calcul. C'est la réconciliation entre les scientifiques et les architectes informaticiens. Les versions de référence sont clairement identifiées ainsi que les niveaux d'accès pour les utilisateurs. On cherche à minimiser les tentations « tournevis » citées précédemment. La réécriture permet de reconsidérer l'architecture globale du logiciel avec des professionnels. Des coupleurs de logiciels « maison » commencent à devenir performants.

Maîtrise des architectures, bases de données et interfaces (de 1998 à nos jours)

Les composants des logiciels sont de plus en plus complexes. L'accroissement de rapidité des calculateurs permet le développement de modèles de plus en plus descriptifs et donc plus prédictifs, mais gourmands en temps calcul. Il devient une nécessité entre scientifiques d'échanger des modules entre les logiciels. La programmation objets développée dans les années 1980 semble être une nouvelle voie de développement pour aller vers des plates-formes multi-métiers. C'est enfin le total retour en grâce des informaticiens auprès des scientifiques qui n'ont pas la maîtrise de ces architectures objets et des langages qui vont avec (C, C++, PYTHON …).Le logiciel scientifique devient alors une application à l'intérieur d'une plate-forme, dont le programme principal définit en langage objets le chaînage des calculs. Les modèles « métier » sont des briques élémentaires de fortran encapsulées. En effet, il n'est pas dans un premier temps réaliste de reprogrammer l'ensemble de nos logiciels qui représentent des dizaines de milliers d'instructions. C'est l'occasion de mettre à jour la documentation des logiciels les plus anciens qui n'ont pas franchi la précédente étape.

Évidemment, cet historique est la vérité de l'auteur qui en assume complètement la responsabilité et a cherché à montrer plusieurs choses : Il ne faut pas sous-estimer les personnes qui nous ont précédés dans le développement des logiciels. Ils ont su pour la plupart suivre au mieux l'évolution effrénée des matériels. L'évolution vers une architecture objets est une suite logique qui permet de faire communiquer les logiciels, échanger les outils métiers, gérer des ensembles complexes. Pour les plus pessimistes, nous avons néanmoins gardé nos règles à calcul ; on ne sait jamais ce que nous réserve l'avenir…

Daniel Baron
Physicien Expert Senior à EDF R&D