35 ans d'actions méthodologiques dans les S.I.

La pertinence et la durabilité des S.I. au centre des enjeux des T.I.C.

par Pierre Fischof
Secrétaire d'ADELI

Les technologies numériques de l'information et de la communication ont toujours été le terrain d'un débat passionné entre tenants du plus pur pragmatisme (tant technique que fonctionnel) et tenants d'une approche plus philosophiquement conceptualisée du développement de systèmes d'information.
L'enjeu s'est souvent trouvé, pour caricaturer, dans une volonté ou non, dans une capacité ou non à pouvoir « maîtriser », « piloter » et « conduire », d'une part, les processus d'élaboration, et d'autre part, le produit lui-même, à savoir les Applicatifs et les Systèmes d'Information, donc leur exploitabilité, leur maintenabilité et leur évolutivité durable. Oui, le développement durable des S.I. a toujours été fortement un enjeu, donc l'écologie des S.I., des développements applicatifs et de leurs rapports humains, avant même que la mode soit officiellement à l'écologie !

Améliorer les développements d'applications et de programmes

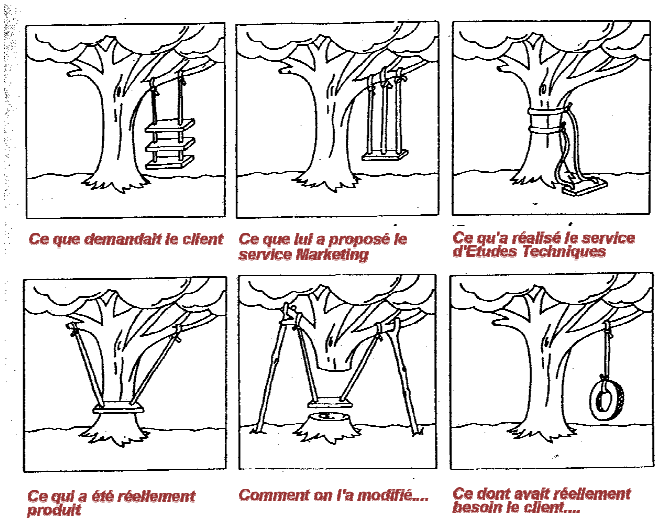
Pour des raisons de simple bon sens, la guerre a toujours existé contre les programmes « spaghettis », dénués de surcroît de toute documentation ou auto-documentation valides. Un programme n'est en général pas destiné à servir une seule fois ni à être jeté à la poubelle après son premier usage, ce qui serait trop coûteux. Son exploitation doit pouvoir évoluer, être revue et complétée, cela en fonction des évolutions de l'environnement et des besoins. Le partage de l'information et le jeu collectif doivent être de bonne règle, dans un sens comme dans l'autre, et horizontalement comme verticalement.

Pour l'anecdote, combien de programmes n'avons-nous pas vu dotés de noms de traitements et de variables dénotant la poésie personnelle de leur auteur mais moins leur souci d'être compris par tous ? Comme si celui-ci souhaitait se rendre définitivement reconnu et indispensable… Ce qui n'est pas forcément la meilleure des stratégies pour cela : encore fallait-il que l'auteur de l'application fût encore présent et soit capable de se relire et de se comprendre après plusieurs mois, ce qui était loin d'être toujours gagné…

Une série de dessins humoristiques - assez connue dans les milieux informatiques mais non toujours bien acceptée - avait efficacement illustré les premiers obstacles que devait contribuer à surmonter une méthode : une application livrée rendue excessivement complexe de façon inutile et qui n'avait plus grand rapport avec les besoins de l'utilisateur ou du client.

Cette application était symbolisée sous la forme d'une balançoire sous un arbre. Caricature éloignée de la réalité ?

Non : chacune et chacun sait que, au dire les cabinets d'analyse qui font internationalement référence, un pourcentage trop élevé de projets sont malheureusement encore trop souvent voués au placard, faute d'être adaptés au besoin et de pouvoir être utilisés…



Source : La qualité avec le sourire – René Droin - Dunod 1991

Parmi les méthodes proposées destinées à rationaliser la production des applications et des programmes, on trouva donc en particulier, par exemple, dans les années 1970 les méthodes **CORIG**, d'un côté, et **LCP-Warnier**, de l'autre.

Jean-Dominique Warnier
(source : davehigginsconsulting.com)

La première, fondée sur la rationalisation des traitements, était promue par la CGI auprès des organisations et des personnes intéressées, et la seconde, promue par Bull notamment par l'intermédiaire du service de formation et du « SCAT »[1], et reposait sur la structuration hiérarchique des résultats demandés et des données ; une première approche objet en somme. La méthode CORIG permit de donner le jour aux outils d'aide à la programmation et générateurs de code **PAC** (Programmation Automatique Corig), qui deviendront par la suite Pacbase.
De son côté, IBM proposa aussi sa méthode **Hypo IPT**, fondée elle aussi sur une analyse des données.

« L'ADELI » était alors née, en 1978, au départ d'une volonté de structurer le rapprochement entre des praticiens et enseignants de LCP-Warnier en vue d'une meilleure diffusion et compréhension de cette méthode et approche de construction de programmes.

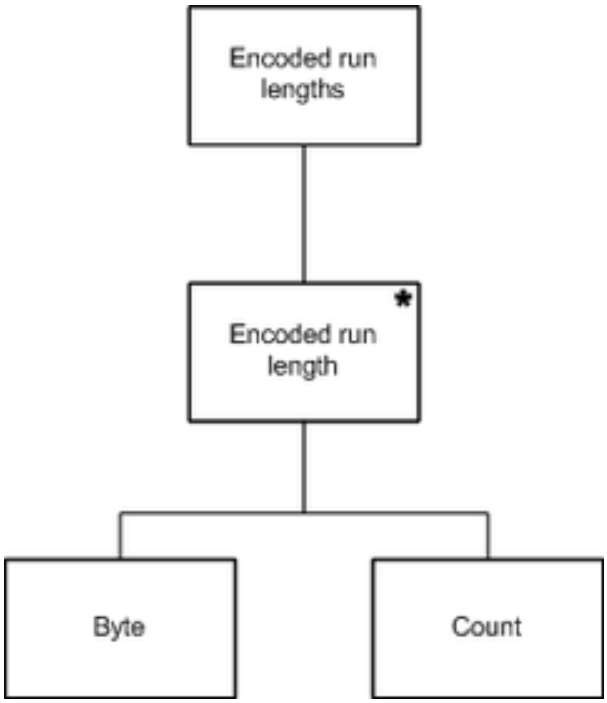
Celle-ci, simple dans ses bases mais réclamant un effort de discipline, n'était, en effet, pas pour autant simple à bien comprendre, mettre en œuvre et à faire partager en France, le pays d'Astérix et de Descartes, sinon dans les écoles, contrairement aux pays d'Afrique, d'Europe de l'Est et d'Amérique du Sud comme du Nord qui semblaient l'adopter plus facilement (nul n'est prophète en son pays !).
La volonté de la naissance d'ADELI était aussi de privilégier une véritable approche scientifique et philosophique de l'informatique (sa « logique ») et non plus sa seule approche « techniciste », trop souvent privilégiée (la structure des machines).

L'enseignement de l'algorithmique à l'entreprise et à l'école n'était pas toujours privilégié.

D'abord humaniste et soucieuse d'alléger les difficultés des professionnels, la logique de LCP s'appuyait sur l'algèbre des ensembles et sur un respect d'un juste dialogue entre les utilisateurs et les développeurs.

Une variante britannique de la méthode promue par Michael A. **Jackson**, vit le jour outre-manche sous le nom de ce dernier.

Tandis que la première privilégiait graphiquement l'utilisation d'une hiérarchie horizontale d'accolades pour outil de conceptualisation, la seconde privilégiait l'utilisation verticale de rectangles reliés entre eux par des traits mais signifiant la même chose. On voit, comme souvent, combien la différence était fondamentale !



Exemple de diagramme Jackson (Source : Wikipédia)

Parmi les quelques méthodes enseignées à la fin des années soixante-dix figurait aussi la méthode de « **l'arbre programmatique** » (promue par Marie-Thérèse Bertini et Yves Tallineau) ou dite de la « programmation sans panne » (**PSP**).

[1] *Service Clientèle d'Assistance Technique du groupe Bull.*

Finalement assez proche de LCP, mais de façon moins détaillée, et de la méthode Jackson, celle-ci consistait à rechercher et à décomposer hiérarchiquement, en répétitives et d'alternatives, graphiquement de haut en bas sous la forme de rectangles reliés entre eux par des traits, les dits « objets de gestion » (fonctionnels ou métiers) résumant :

- premièrement, les données à obtenir en « sortie » ;
- deuxièmement les données à fournir en « entrée » ;
- troisièmement, enfin, les traitements permettant de fournir les sorties en fonction des entrées.

Il suffisait alors d'ajouter à chaque niveau hiérarchique des pavés de début, de fin et de séparation des répétitives et des alternatives.

On pouvait alors obtenir des paragraphes susceptibles d'être directement codés simplement en COBOL.

Ces paragraphes étaient systématiquement appelés par des « Perform », ou appels de sous-programmes internes, ce qui pouvait rendre la lecture du programme, il est vrai, parfois un peu fastidieuse.
Paradoxalement, c'est en pratiquant avec efficacité cette méthode dans l'écriture de programme Cobol et Assembleur, à l'université puis en entreprise, que j'ai pu mieux comprendre et apprécier la méthode LCP de Warnier. Celle-ci formait en effet une démarche de base similaire, mais expliquée de façon plus détaillée et modélisée différemment.

Notons aussi que la méthode anglo-saxonne **SADT (**Structured Analysis and Design Technique) à l'origine utilisée dans les processus industriels, puis étendue à la gestion, était fondée sur un découpage hiérarchique vertical en modules successifs, débouchant graphiquement sur une succession de pavés ressemblant à ceux de la méthode précédente, tout en privilégiant la décomposition des traitements.

Paradoxalement, l'utilisation des méthodes en France sur le terrain semblait pourtant inversement proportionnelle à leur multiplicité, tant elle était sujette à dénigrements - était-ce par paresse intellectuelle ? - sous le prétexte d'une perte possible d'efficacité et de rapidité…

Les utilisateurs de méthodes faisaient parfois figure soit de précurseurs audacieux et courageux, dans le meilleur des cas, soit, dans le pire des cas, de « dangereux illuminés », d'ailleurs déconnectés de toute réalité…

La survenance des générateurs informatisés de programmes, peu à peu, rendit en revanche indispensable et obligatoire le respect d'une méthode de développement, là où ces outils étaient présents et bien utilisés.
En somme, une méthode était bien utilisée là où il était quasiment impossible, faute de sanction, de se passer de l'utiliser !

Plus tard, la **programmation structurée** prit son essor, ainsi que la **programmation objet**, parfois si mal comprise et recouvrant des pratiques parfois tellement différentes, celle-ci alliée à un développement systématiquement modulaire des applications et la revendication proclamée de la réutilisabilité des composants.

Il convient ici de mentionner ici l'importance prise par la **programmation événementielle**, tout d'abord destinée à piloter les interfaces hommes-machines (IHM) supportés par des moniteurs transactionnels, tels CICS, et les traitements face à chacune des actions interactives que pouvait décider l'utilisateur.

Plus tard, cette même logique événementielle fut utilisée pour piloter les IHM en environnements graphiques à base de fenêtres, icônes et listes déroulantes, puis plus tard pour les IHM graphiques fondées sur le Web.

Parmi ces courants de méthodes, souvent entremêlés et sujets à des modes parfois cycliques, il serait souvent par trop illusoire de vouloir dresser une chronologie historique linéaire, tout comme des technologies qui en étaient le soubassement, tant l'histoire, en informatique, peut partiellement parfois se répéter.

Parmi ces différents courants de méthodes de développement d'applications et programmes, notons, enfin, les courants qui pourraient apparaître aujourd'hui comme les plus modernes, pouvant reposer sur des **maquettages et prototypages itératifs et incrémentaux** des produits à obtenir, associant tantôt utilisateurs finals et développeurs au sein d'une même équipe, de façon **agile** – pour utiliser un terme contemporain - tantôt associant de façon très étroite les développeurs eux-mêmes entre eux.

Sans doute pouvons-nous dire aujourd'hui que les formes de développement d'application les plus coopératives et les plus incrémentales semblent ce qui peut émerger de plus moderne dans nos façons de développer.
Maintes remarques exprimées ici sur les programmes pourraient être reconduites de façon assez similaire à propos de la construction des systèmes d'information, construction que nous passerons brièvement en revue à présent.

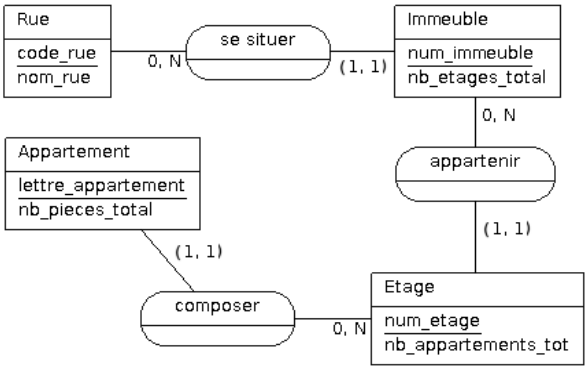
Améliorer la construction des systèmes d'information

Vint la période de l'apparition des premières méthodes soucieuses de maîtriser les domaines fonctionnels de l'entreprise, correspondant souvent à des départements ou divisions, ainsi qu'ensuite leur système d'information intégral, ceci donc de plus en plus globalement et avec de plus en plus d'ambition.

C'est souvent les méthodes qui s'étaient précédemment attaquées à la rationalisation des programmes qui cherchèrent peu à peu à s'étendre à la rationalisation de la construction des systèmes d'information.
Chez Warnier, et au sein de Bull, LCP connut de petits frères en **LDR**, logique de description des résultats attendus, et **LCS**, logique de construction des systèmes.
Ceci en s'appuyant toujours sur l'algèbre ensembliste, l'étude des résultats demandés et des données, mais beaucoup plus cette fois sur l'approche systémique des organisations et de leurs systèmes d'information, telle que le formalise Jean-Louis Lemoigne.

Le début des années quatre-vingt connut le développement et la popularisation naissante des **approches objet**.
Celles-ci ambitionnent l'étude systématique des objets de gestion nécessaires au fonctionnement d'une organisation ainsi que de leurs différentes interrelations, ceci pour bâtir son système d'information. Notamment au travers de la démarche **individus-relations** (ou entités-associations).

Cette approche mondialement reconnue par les milieux universitaires vit, par la suite en France, son encapsulation, quasi-gouvernementale car planifiée par le ministère de l'industrie, au sein de la méthode **Merise**.
Celle-ci fut le fruit d'une collaboration entre universités, industriels et SSII sous l'égide de l'État.



Modèle Conceptuel de Données Merise
(Source Développez.com)

Séparant hiérarchiquement les niveaux de préoccupations en trois niveaux d'abstraction, la représentation entités-associations des données utilisée au niveau conceptuel donnait lieu au niveau logique à une représentation sous forme « Codasyl » de type réseau, plus avancée que la représentation hiérarchique, en attendant de pouvoir s'abstraire technologiquement de ces contraintes pour pouvoir utiliser une représentation logique de type relationnel.

Côté traitements, Merise intégra de même une méthode de formalisation des processus de l'entreprise, au niveau conceptuel, par le biais d'une représentation **événements-synchronisations-opérations**, les événements pouvant être déclencheurs, en amont, ou résultats de l'opération, en aval, mais sans aucune préoccupation à ce stade du « qui fait quoi », ni du « quand », du « où », ni du « comment ».

Au niveau organisationnel, la représentation utilisée s'insérait dans un formalisme plus traditionnel en forme de diagrammes de circulation de l'information entre les différents services, les opérations se transformant alors en différentes phases ou tâches.

Au niveau physique des données et opérationnel des traitements, nul nouveau modèle n'avait besoin d'être créé, les représentations informatiques traditionnelles existantes et courantes se suffisant généralement.

Cette méthode a permis de formaliser efficacement peu à peu la compréhension de maintes organisations complexes et de leur système d'information. Ceci malgré parfois certaines lourdeurs et lenteurs exagérées qui pouvaient résulter d'une mise en œuvre par trop rigide de celle-ci, freinant alors le développement rapide nécessaire du système d'information.

Pour la maîtrise des systèmes d'information, les Anglo-Saxons comme les germains préférèrent, quant à eux, mettre souvent en place des catalogues de procédures écrites, de bonnes pratiques à utiliser et d'étapes à franchir et valider, le tout inséré dans des classeurs, comme ce fut le cas au travers de la méthode de conduite de projets très diffusée **SDM (**System Development Management),

Tandis que les francophones semblaient alors privilégier la modélisation graphique, les Anglo-Saxons semblaient alors privilégier des chronologies de procédures écrites à respecter.

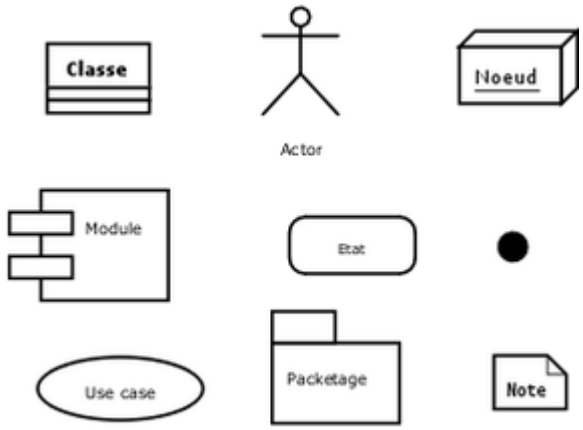
Tout ceci bien qu'une majorité de praticiens et d'organisations, au moins en France, semblât souvent préférer n'utiliser aucune de ces méthodes, à moins d'y être contraints (ce qui fut pour la qualité, heureusement il faut le dire, souvent le cas…).

Peu à peu à cette époque se multiplièrent les outils de conception, AGL supportant les méthodes et facilitant et structurant en principe leur mise en œuvre.
En France, Merise fut, par exemple, supportée par un module **Pacbase**, ainsi que par l'AGL **MEGA** permettant le maquettage, ainsi que par quelques autres outils.

On peut citer également la méthode IDA (Interactive Design Approach) qui supportait et structurait un atelier de génie logiciel (AGL) du même nom, pour la conception, mettait en œuvre, pour modéliser les traitements, des formalismes propres plus proches des approches anglo-saxonnes.

Succédant à ces différents formalismes, **UML**, fruit d'une concertation et d'un consensus international très méritoires, il faut le souligner, quelques années plus tard, permit une formalisation objet se voulant plus précise des programmes, des organisations, et des systèmes de données et des traitements, entre autres choses, qui mit peu à peu tout le monde d'accord sur le principe sur les modèles utilisables.

Utilisée d'abord pour modéliser la programmation objet, UML fut utilisé ensuite et étendu à la description des organisations, à leurs systèmes informatiques et d'information.



Éléments de modélisation UML
(source Wikipédia)

Par la suite virent le jour une multitude de tentatives pour bâtir des méthodologies européennes, puis mondiales, pour succéder à Merise et à UML.
Ces tentatives, chacune intrinsèquement méritoire, ne rencontrèrent guère un succès aussi universel que leurs aînées.
Ceci sans doute parce qu'elles en contenaient encore partiellement une part de limitations similaires en rigidité et souplesse, cette faiblesse contemporaine réveillant plutôt la nostalgie pour les méthodes traditionnelles connues, plutôt que de faire l'effort de s'en approprier une nouvelle.
Citons ici en particulier **Eurométhode**, à côté de bien d'autres excellentes méthodes dont le succès « commercial » n'a pas été au rendez-vous.

Aujourd'hui, l'exigence accrue et critique croissante en termes d'adaptabilité et de souplesse des organisations comme de leurs systèmes d'information conduit à exiger des méthodologies mettant en œuvre de plus en plus de souplesse et d'agilité. À tel point que le terme « d'agilité » est presque devenu la pierre angulaire de toute méthode qui se respecte et de toute présentation commerciale et marketing parfaitement bien accepté.

Ces nouvelles **approches méthodologiques agiles**, riches et diverses, méritent de faire l'objet en elles-mêmes d'articles axés sur ce seul sujet ; certains d'entre eux ont déjà été écrits et publiés, parmi différents endroits, dans les colonnes de la Lettre d'ADELI, d'autres sont encore à écrire.

Améliorer la maîtrise des organisations

Pour mémoire, citons enfin une troisième catégorie de méthodologies prometteuses dont la volonté, plus large encore, est la maîtrise des organisations et de leurs processus dans tous leurs aspects, qu'ils soient humains, matériels, financiers ou informationnels…

Ces méthodologies sont multiples et leur naissance ne date pas d'hier, mais leur caractéristique est d'intégrer de mieux en mieux et de façon de plus en plus souple tous ces différents aspects inhérents au bon fonctionnement d'une organisation et de leur combinaison.
Observons que certaines de ces méthodologies peuvent être heureusement assistées ou supportées par des automates informatiques, facilitant ainsi leur modélisation et leur mise en œuvre, comme ADELI en a fourni l'illustration par le passé.

Beaucoup d'entre elles ont fait l'objet d'une présentation, notamment lors des conférences mensuelles organisées par ADELI.

Vous trouverez ainsi dans la présente Lettre 93 le compte rendu de la dernière et très riche présentation-débat animée par Henri Chelli sur l'organisation et l'informatisation.

D'autres présentations sont disponibles sur le site www.adeli.org, dont par exemple dans la Lettre 71 le compte rendu d'un débat animé conjointement par Éric Pasteyer et Arnaud Trouvé sur la **BPM** (Business Process Management) et le management relationnel, autour des outils et méthodes **LCMD** (Linear Color Modeling Design) et **LCO** (Logique de Conception des Organisations).

En guise de conclusion

Pour aujourd'hui et dans ce cadre-ci, limitons-nous à ce bref panorama de ces trois catégories de méthodes. Celles-ci loin de recouvrir la totalité du champ des domaines méthodologiques permettent d'améliorer le pilotage, sinon la « maîtrise » des systèmes d'information.

Comme toujours, dans le domaine des systèmes d'information (S.I.) et des technologies de l'information et de la communication (T.I.C.), il convient de départager. Départager :

- d'un côté, la progression réelle et concrète de l'état de l'art quotidien permis par les nouvelles approches incrémentales, coopératives et agiles de développement des organisations et systèmes d'information
- et, de l'autre côté, la nécessité d'associer à ces nouvelles approches certaines anciennes développées avec grand succès précédemment, celles-ci pas toujours suffisamment mises à profit, comme par exemple toutes les approches objet.

Continuer à innover, donc, de façon de plus en plus coopérative, dans nos façons d'agir et de travailler, en réutilisant en même temps ce qu'il y a de meilleur dans toutes les méthodologies développées aujourd'hui, demain et hier depuis la naissance de la science informatique.

Car, au fond, tout se combine aujourd'hui de façon de plus en plus complexe, savante et rapide, nécessitant l'acquisition progressive et constante de nouvelles manières de faire. Néanmoins, les fondements pris isolément de l'informatique, des S.I. et de la science des organisations restent fondés finalement sur les mêmes réalités qu'à la naissance de ces sciences,

Ne pas intégrer simultanément au présent ces deux bouts de la chaîne diamétralement opposés de la réalité (l'aspect plus ancré dans le futur et celui plus ancré dans le passé) c'est risquer de se priver de la meilleure combinaison d'outils méthodologiques disponibles. Ceci pour tendre à toujours mieux piloter, conduire et gérer la présente réalité.▲

pierre.fischof@yahoo.fr