ELOGE DE LA METHODE POUR DEVELOPPER LES LOGICIELS

T.LEBEL

L'hydrologie, science de l'environnement qui a du mal à définir ses perspectives scientifiques, se tourne de manière croissante vers l'informatique pour l'aider à résoudre ses problèmes de programmation. Ce faisant, elle est contrainte de sortir de l'ère préhistorique où chacun développait son logiciel pour résoudre des problèmes spécifiques. Des hydrologues, considérant l'extrême particularité (selon eux) de leurs problèmes numériques, statistiques et de leurs techniques de modélisation, ont longtemps regardé des hauteurs de leurs certitudes, les efforts des géologues ou autres géographes développant des logiciels d'utilisation très générale et allant jusqu'à les commercialiser. Lorsque la télédétection a fait irruption sur la scène, il a également été clair d'entrée de jeux qu'elle ne pouvait être utile à l'hydrologie que de manière marginale. Justifiée ou non sur le plan scientifique, cette attitude nous a éloigné encore un peu plus du savoir-faire, (impératif pour exploiter correctement la masse de données fournie par les satellites) qui s'améliorait sans cesse dans le domaine de la conception des logiciels. Aujourd'hui l'heure des systèmes experts a sonné et les regards se tournent plein d'espoir vers ces nouveaux sauveurs de l'hydrologie en danger. Quiconque fait preuve d'un minimum de clairvoyance peut anticiper le développement de tels systèmes dans les sciences de l'environnement de la décennie à venir. La question dès lors est de savoir si nous devons attendre qu'on nous vende des systèmes, plus ou moins bien adaptés de systèmes mis au point pour d'autres propos, ou bien si nous devons faire l'effort d'acquérir le savoir-faire nécessaire à la réalisation de tels systèmes à partir d'un cahier des charges soigneusement réfléchi au préalable. Mais avant même que de songer à réaliser des systèmes sophistiqués, interrogeons-nous sur notre capacité du moment à développer des logiciels classiques, utilisant au mieux les moyens techniques des ordinateurs actuels dans le domaine de la représentation graphique et des réseaux notamment, des logiciels portables, évolutifs, interconnectables.

Cette capacité est d'une importance particulière à l'ORSTOM du fait de la diversité de nos missions : opérationnelles, de recherche et d'enseignement. Contrairement à la plupart des laboratoires universitaires ou du CNRS, nous n'avons pas d'autre choix que de placer nos activités de recherche dans la perspective de nos missions opérationnelles Outre-Mer. Cette exigence est contraignante car elle nous oblige à développer nos logiciels de manière à ce qu'ils soient utilisables aisément dans des contextes très variés et par du personnel pas nécessairement formé aux manipulations informatiques. C'est en même temps un atout car nos efforts pour améliorer la

qualité de nos logiciels ont des effets multipliés par leur nombre d'implantations. Nous avons la chance d'être obligé de bien faire ce travail si nous voulons qu'il soit efficace. Est-ce le cas ?

Dans l'univers de pénurie relative propre à la Recherche, il est surprenant de constater le peu de moyens consacrés à rentabiliser nos investissements en matériel informatique. Nous sommes prêts à nous battre pour acheter du matériel si possible récent donc relativement coûteux, mais ensuite sur quels éléments rationnels nous appuyons-nous pour l'utiliser au mieux ? Mettre au point une méthodologie permettant de définir nos besoins et de développer le(s) logiciel(s) qui y satisferont semble au delà de nos forces : trop lourd, effort disproportionné avec les résultats escomptés. Il est vrai que ce type de travail est long, qu'il ne peut être démarré puis arrêté puis redémarré au gré des humeurs et des disponibilités en personnel.. L'investissement est refusé parce que ses effets sont masqués, semblables en cela à ceux qui résultent d'une meilleure organisation ou d'une meilleure circulation de l'information. Quelles seraient les grandes lignes d'un projet permettant d'améliorer l'efficacité de ce type d'investissement ?

On peut réfléchir à :

1. Définir des niveaux de logiciel en fonction de certains critères :

+ fiabilité
+ documentation
+ degré de diffusion
+ rentabilité

2. Cataloguer les langages à utiliser en fonction :

+ des objectifs du logiciel
+ du niveau recherché

3. Etablir pour chaque niveau un manuel traitant du développement des logiciels et de leur documentation.

4. Regrouper les logiciels en bibliothèques remises périodiquement à niveau, sous la responsabilité d'une personne bien identifiée.

Ce type de travail requiert une organisation qui ne sera pas discutée ici. Tout au plus peut-on indiquer que l'organisation de stages et de séminaires ainsi qu'une amélioration de nos circuits de communications en sont à coup sûr les prémices indispensables.

Une des tâches les plus importantes à laquelle nous devons faire face aujourd'hui est d'identifier les domaines dans lesquels le développement de logiciels performants (interactifs, modulaires, évolutifs, ...) est devenu prioritaire, soit parce que le retard s'y est accumulé, soit parce que la progression de nos connaissances ou de nos outils est bloquée en l'absence de tels logiciels. C'est ce à quoi nous nous sommes attachés en développant une chaîne de traitement de données spatialisées. Ce travail est très caractéristique de notre mode artisanal de fonctionnement actuel et de ces deux points de vue au moins :

a) il répondait à un besoin non exprimé au sein de l'ORSTOM mais néanmoins facile à détecter vu que les objectifs d'ensemble étaient clairement définis (cartographie; estimation de valeurs ponctuelles ou spatiales) et il s'est déroulé comme une succession plus ou moins planifiée de quasi-improvisations.

Certains éléments positifs sont à retenir :

+ définition de formats standards pour des fichiers de travail servant de points d'entrée obligatoire dans la chaine de traitement à différents niveaux.

+ mise au point d'une documentation concernant les fichiers et les logiciels développés.

+ essais de standardisation du dialogue entre l'utilisateur et le logiciel.

+ regroupement des outils développés à cette occasion en bibliothèques rassemblant des utilitaires, des outils graphiques, des fonctions statistiques etc.

+ et pour finir, logiciels fiables et d'une utilisation relativement aisée.

L'inventaire des points négatifs n'est pas négligeable pour autant.

+ difficulté à tenir les délais de réalisation.

+ dérapages par rapport aux objectifs initiaux, entrainant des conceptions erronées et des lourdeurs inutiles.

+ notices utilisateurs souvent perçus commes inutilement compliquées.
+ évolution, portabilité et maintenance des produits difficile à réaliser.

Etape	Fonctionnalité du logiciel	Spécifications du logiciel	Analyse détaillée et Implantation	Test	Mise en place	Maintenance
Objectifs	- fonctions du logiciel - niveau de développement souhaité - interfaces avec autres logiciels - interfaces avec banques de données - interaction avec l'utilisateur	- enchaînement des fonctions - données nécessaires et accès - développements ultérieurs envisagés - choix d'un langage - choix d'un matériel - définition du niveau de portabilité	- structure interne répondant aux standards - documentation interne - interfaces	- Le logiciel remplit-il les spécifications ? - Est-il en accord avec les standards ? - Comportement des différents modules face à erreurs utilisateur ou entrées - Le degré de fiabilité général souhaité est-il atteint ? - Test par utilisateurs non avertis.	- construction de bibliothèques - sauvegardes - documentation : fichiers, manuel utilisateur, notice d'emploi rapide.	- Répertorier les mauvais fonctionnements. - Faire les corrections nécessaires. - Actualiser la documentation et les bibliothèques.
Document support	Manuel destiné au la documentation	développement et à	Manuel du programmeur pour langage choisi et type de développement souhaité		Notice système. Catalogues des bibliothèques.	
UR, groupe ou personne responsable	Définir une collectivité concernée.	Groupe de travail - Chercheur responsable - Programmeur	Chercheur responsable et programmeur	Chercheur responsable et programmeur	Programmeur et Ingénieur système	Ingénieur système

Tableau 1 : Planification en vue du développement d'un logiciel

Cette première expérience a montré que pour développer ce type de logiciel dans de bonnes conditions, une certaine planification préalable était nécessaire, du type de celle esquissée dans le tableau ci-joint. La mise en oeuvre d'une telle planification ne peut avoir un sens qu'à travers la mise sur pied de véritables projets informatiques dont on soit capable de définir les spécifications au préalable et de contrôler la manière dont elles sont progressivement atteintes.

Le développement d'une banque de modèles hydrologiques pourrait être l'occasion de tester la pertinence des concepts définis ci-dessus compte tenu des contraintes de notre environnement de travail.

La modélisation hydrologique destinée à la reconstitution ou à la prévision des débits écoulés à un exutoire quelconque est encore largement considérée comme l'apanage d'une poignée de spécialistes, qu'ils soient simples utilisateurs ou à plus forte raison concepteurs de modèles. Il est vrai que la conception de ces modèles reste un art autant qu 'une science, et qu 'aucun d'eux ne s'est jamais imposé comme supérieur aux autres au point de le faire adopter comme un étalon standard auquel on puisse référer les résultats obtenus à l'aide de tout modèle appartenant à la même catégorie. Rien dans l'évolution actuelle de nos connaissances hydrologiques ne peut nous faire supposer que cet état de fait changera soudainement et il est dès lors logique de croire que seule une approche comparative fondée sur l'utilisation conjointe de plusieurs modèles est à même de fournir les réponses les plus complètes aux problèmes traités. Or la manière dont sont conçus les logiciels de modélisation hydrologique est le plus souvent incompatible avec une telle utilisation.

Un modèle hydrologique de simulation d'apports n'est en général qu'une succession d'algorithmes, mathématiquement simples mais enchevêtrés de façon tellement inextricable au niveau du code informatique que seul celui qui l'a écrit (ou quelques pairs initiés) peut le comprendre et l'améliorer le cas échéant. Considérant que ces modèles ont tous un objectif semblable, qu'ils sont basés sur des concepts suffisamment voisins pour être interchangeables, que leurs techniques de calcul sont analogues (par exemple l'optimisation), et que les critères d'appréciation de la qualité des résultats leur sont pareillement applicables, aucune raison inhérente à la nature de ces modèles ne peut expliquer leur complexité formelle.

Une écriture plus lisible, mieux structurée et obéissant à des règles communes à tout logiciel d'une catégorie donnée améliorerait nos capacités de développement et serait d'un bénéfice immédiat pour les utilisateurs :

- en évitant de réécrire plusieurs fois des algorithmes équivalents sous des formes qui les rendent inidentifiables;

- en facilitant le partage des tâches de développement entre plusieurs chercheurs ou informaticiens travaillant simultanément ou séquentiellement et dispersés géographiquement;
- en autorisant des échanges aisés de modules remplissant des fonctions comparables;
- en permettant ainsi de comparer rapidement les performances de différentes combinaisons de ces modules pour chaque cas particulier.
- en offrant à l'utilisateur un dialogue standard servant de point de repère d'un modèle à l'autre.

On voit que le concept central de cette approche est celui de la programmation modulaire et structurée. Ce type de programmation permettrait de créer de véritables boîtes à outils, accessibles au programmeur sous forme de code source et à l'utilisateur sous forme de logiciels interactifs d'activation de ces modules. Dans l'état actuel de nos connaissances et de nos capacités à modéliser les mécanismes hydrologiques pertinents, cette démarche du type génie logiciel est une des seules qui soit susceptible de faire progresser l'état de l'art en matière de modèles de simulation d'apports. Ceci ne sera possible que grâce à un véritable projet informatique fondé sur une réflexion adéquate. Cependant dans un organisme de recherche où le soutien logistique informatique est limité, une telle démarche signifie que chaque chercheur impliqué y participe dans la mesure de ses moyens. Plus précisément, on est en droit d'exiger des chercheurs développant des modules intégrables dans cette banque de logiciels en cours de constitution, qu'ils respectent certaines règles qui rendront leurs outils facilement utilisables.

Il s'agit donc à ce stade de définir l'organigramme type d'un modele hydrologique, d'identifier ses différentes couches et d'établir pour chacune d'elle un formalisme. Dans le schema de la figure 1, on a esquissé la stratification d'un modèle conceptuel global. On remarquera que l'opérateur de transformation pluie-débit constituant le noyau de l'ensemble fonctionne sur un pas de temps, le temps étant supposé discrétisé puisque ce type de modèle ne résout analytiquement aucune équation différentielle. Le temps n'apparait donc pas en tant que tel au niveau du noyau et l'enchainement des pas de temps est géré par la couche supérieure .

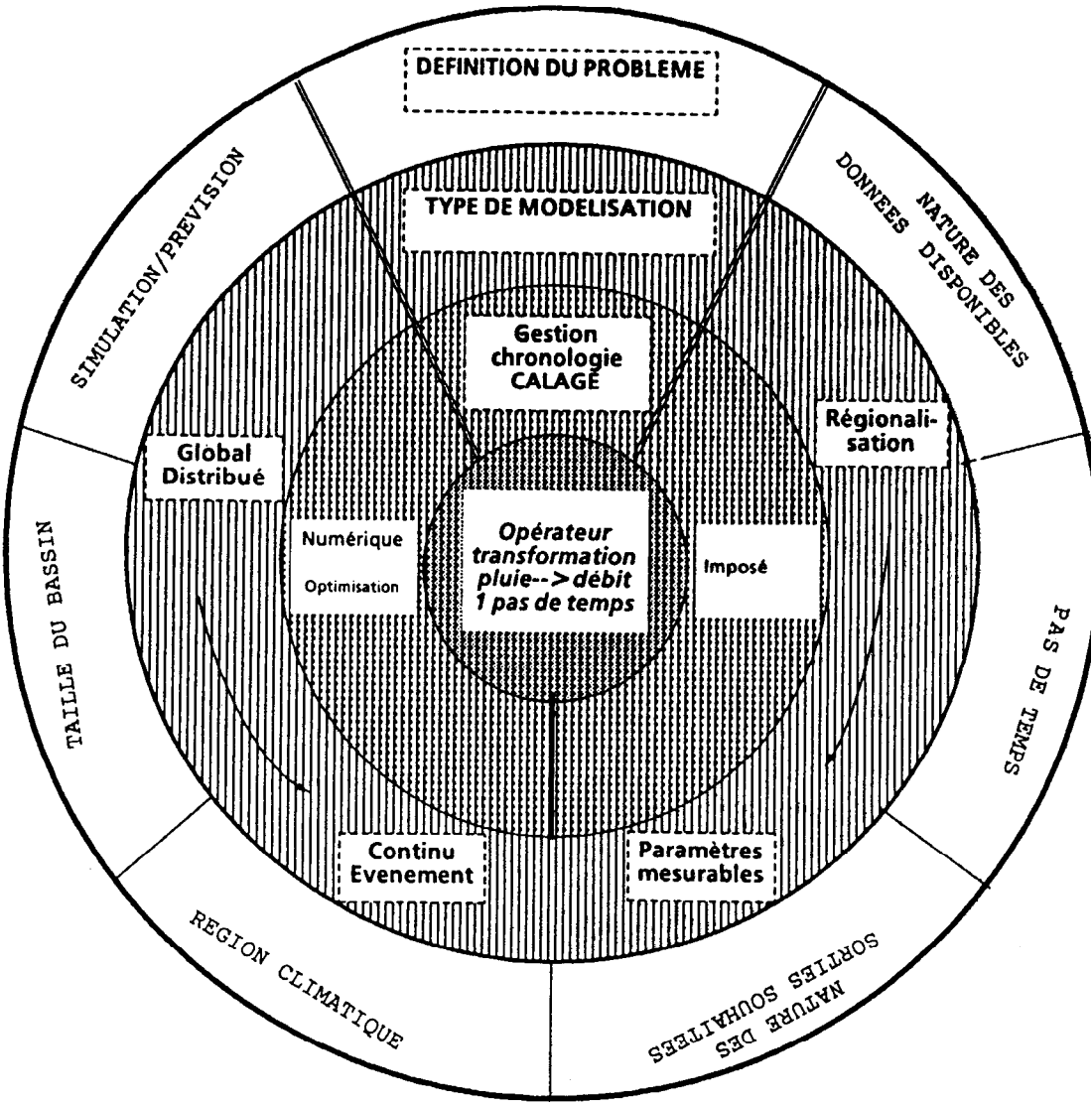

Figure 1 : Décomposition d'un modèle hydrologique d'apports en vu de la constitution d'une banque de modules permettant de construire des modeles "sur mesure". On définit des couches qui se superposent à un noyau contenant l'opérateur principal du modèle. Plus une couche est extérieur, plus ses éléments constitutifs feront appel aux spécificités du calculateur sur lequel le logiciel est implanté..

Pour chaque couche on cherchera à bâtir une boîte à outils constituée de modules interchangeables : ce sont les éléments constitutifs de l'ensemble-couche. On peut alors définir pour chaque élément d'une couche donnée un certain nombre de règles s'adressant selon les cas au programmeur ou à l'utilisateur (c'est à dire à celui qui veut incorporer un de ces modules dans un logiciel en cours d'élaboration). Dans la mesure où l'ensemble concerné est suffisamment restreint et bien caractérisé, ces règles peuvents être très strictes, permettant une définition aussi rigoureuse que possible des éléments.

Exemple d'application:

Couche : NOYAU

Opérateurs de transformation pluie-----> débit.

Elément identificateur : HYDROGRAMME UNITAIRE **HU** **N1**
Langage de programmation : FORTRAN 77
Bibliothèque
Eléments constitutifs
Dernière mise à jour
par
Implantations

On définira par ailleurs une documentation type de l'élément donnant les indications sur son mode opératoire, les variables d'entrée, de sortie, d'état et les paramètres du type de celle déja utilisée au Laboratoire d'Hydrologie pour documenter les routines de certaines bibliothèques et dont un exemple est donné ci-dessous.

Nous sommes entrés dans une nouvelle étape de l'utilisation de l'informatique dans les sciences de l'environnement. Pour profiter au mieux des possibilités très vastes qui nous sont offertes,un effort de structuration dans la conception et le développement de logiciels spécifiques à nos disciplines et introuvables sur le marché est nécessaire. Cet effort peut commencer à des niveaux élémentaires à condition qu'une analyse préalable ait permis de définir ces niveaux sans figer les évolutions éventuelles. Dans un organisme dispersé géographiquement et pauvre en soutien informatique spécifique, une telle démarche doit s'appuyer sur le volontarisme d'un nombre de chercheurs suffisants et sur une bonne coordination de leurs travaux. C'est une entreprise contraignante mais qui pourrait déboucher sur la conception d'une nouvelle génération de logiciels hydrologiques préparant le terrain à l'irruption des produits issus de l'intelligence artificielle.

ANNEXE : Documentation type d'un sous-programme catalogué dans une bibliothèque répertoriée

NUMBER

1- FONCTION : Ecriture d'un nombre
Repère DESSIN (xmicm, ymicm)

2- APPEL : CALL NUMBER (NB,X, Y, HX, HY, LDEC)

3- ARGUMENTS :

ENTREE :

. NB	:	Nombre réel à écrire
. X, Y	:	Coordonnées du point de départ d'écriture exprimées en cm, repère dessin
. HXN, HY	:	Taille des caractères en X, Y en cm
. LDEC	:	Nombre de décimales souhaitées

4- DECLARATION DE VARIABLES DANS LE MAIN :

```
REAL *4 NB
INTEGRER*4 LDEC
```

5- ROUTINE AMONT :

GPRINIT

6- REMARQUE :

Le repère dessin est un repère défini en cm par rapport à l'origine absolue du traceur ou de l'écran graphique.