Développements opérationnels des outils de modélisation de la qualité de l'eau dans le bassin de la Seine : PROSE à tubes de courant, version 3

Stéphanie EVEN[1], Nicolas FLIPO[1], Michel POULIN[1], Stéphane BONNIEZ[2], Ronan KERYELL[2]

[1] *Centre d'Informatique Géologique, ENSMP, 35 rue Saint-Honoré, 77 305 FONTAINEBLEAU. even@cig.ensmp.fr*

[2] *Laboratoire d'Informatique et Télécommunications, ENST Bretagne, Technopôle Brest Iroise, 29280 PLOUZANE*

1. Introduction

1.1. Généralités sur le logiciel PROSE

Le logiciel PROSE , développé à l'ENSMP dans le cadre du PIREN SEINE , permet une analyse fine du comportement de systèmes aquatiques en réponse à diverses actions anthropiques. Il permet de simuler l'impact de tout type de pollutions, ponctuelles ou diffuses, sur une large gamme de systèmes aquatiques, qu'il s'agisse d'apports permanents, comme par exemple les rejets de station d'épuration, ou transitoires, tels que les rejets urbains de temps de pluie, ponctuelles ou diffuses. Pour répondre à ces objectifs, des choix conceptuels ont été réalisés dans le but de pouvoir représenter les impacts avec une bonne précision, pour des échelles temporelles et spatiales fines, tout en couvrant correctement l'ensemble du champ des pollutions étudiées.

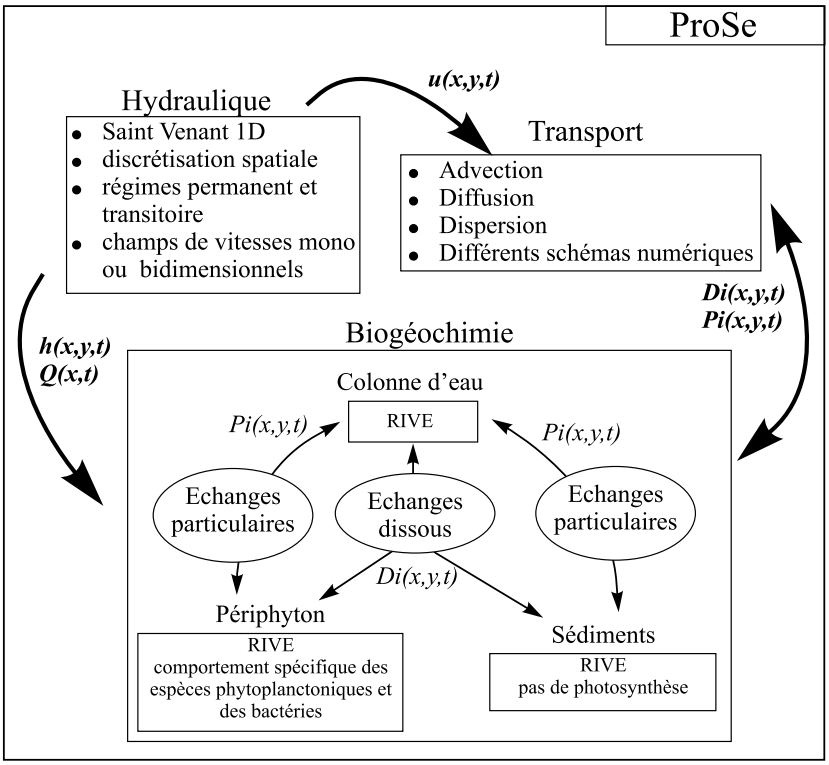

Figure 1: *Schéma conceptuel simplifié de* PROSE *; h est la hauteur d'eau, u la vitesse, Q le débit, Di et Pi représentent les phases dissoutes et particulaires de la variable i.*

Le logiciel comprend (Figure 1) :

- un module hydraulique basé sur la résolution des équations de SAINT-VENANT. Ce calcul

permet de représenter les contraintes physiques (vitesses, hauteur d'eau, section mouillée) en chaque point du système à l'aide d'une information réduite et disponible (débit amont, débits des apports, contraintes aux barrages, bathymétrie) ;

- un module de transport représentant la convection et les processus de dispersion. Plusieurs schémas numériques ont été implantés pour pouvoir traiter différentes problématiques (pollutions accidentelles présentant des fronts de pollution abrupts et nécessitant un traitement numérique fin, pollutions chroniques présentant des variations sur de longues périodes et pouvant être traitées plus simplement) ;
- un module de réactions entre les constituants fondé sur le schéma conceptuel de RIVE. Ce modèle a été adopté dans le cadre d'une concertation avec les autres équipes du PIREN SEINE.

Cependant le logiciel fait l'objet de nombreux développements du point de vue conceptuels et dans le but d'en faciliter l'utilisation.

1.2. PROSE, version 2

Un modèle de transport sédimentaire complet a été couplé au modèle RIVE en remplacement du modèle VENICE existant initialement. Ce dernier ne prenait en compte qu'un flux net de sédimentation, uniforme sur tout le domaine, et supposait une hypothèse de stationarité pour permettre une résolution analytique. Ces hypothèses ne s'appliquaient pas à des situations transitoires, telles que celles traitées dans le cadre des rejets urbains de temps de pluie. Un modèle complet représentant explicitement les flux de sédimentation et d'érosion et permettant de moduler spatialement les zones de déposition/érosion a donc été couplé aux équations de la colonne d'eau (Even et al. 1998).

1.3. PROSE, version 3

1.3.1. Les développements conceptuels

Les derniers développements conceptuels apportés au logiciel PROSE doivent permettre de répondre à de nouveaux objectifs :

- le contexte de l'agglomération parisienne, avec ses nombreux rejets, exige de s'intéresser au champ proche des rejets. Les critères de qualité d'eau imposent des contraintes fortes, surtout dans ce secteur où vivent près de 6 millions d'habitants. Une version pseudo-bidimensionnelle du logiciel PROSE a donc été développée et mise au point (Even et Poulin 2001). L'idée est de ne pas utiliser un modèle bidimensionnel complet qui serait trop lourd à mettre en œuvre et coûteux en temps de calcul, mais de disposer d'une représentation simple du transport bidimensionnel. Cet objectif est réalisé à l'aide d'un modèle à tubes de courant (Thery et al. 1993) ;
- le fait de s'intéresser aux pollutions diffuses exige d'être capable de représenter leur mode de transfert dans les cours d'eau amont, où elles sont essentiellement drainées. Afin d'être à même de traiter correctement les problématiques des petits cours d'eau, les aspects benthiques (périphyton) ont été introduits dans le logiciel PROSE (Flipo et al. 2003; Flipo 2001) ;

Le couplage avec le basin versant et des modèles hydrologiques, à venir, doit permettre de représenter des problématiques telles que les pollutions diffuses (origine, quantification, effet sur les

cours d'eau), la diffusion vers les cours d'eau de pollution venant des nappes, remonter à la source de certains constituants transportés dans les rivières (MES, nitrates, ...).

1.3.2. Les développements informatiques

Le logiciel PROSE est essentiellement développé sous un système de type UNIX et fonctionne également sur des PC sous LINUX. Il est programmé en langage C orienté objet avec l'utilisation de parsers tels que LEX et YACC. Les perspectives de développements informatiques, envisagées afin de faciliter et diversifier son utilisation, sont les suivantes :

- interfaçage des entrées ;
- interfaçage des sorties ;
- couplage avec des bases de données et systèmes d'information géographique ;
- portage sous système d'exploitation WINDOWS.

En effet le point névralgique des modèles est l'utilisation, la manipulation et l'interprétation d'une masse importante de données et c'est sur cet aspect que les trois premiers points mettent l'accent. Les interfaçages, devant faciliter l'utilisabilité du logiciel du point de vue des utilisateurs, ont fait l'objet d'avancées sérieuses cette année.

Le présent rapport fait état de l'avancement des travaux sur les aspects suivants :

1. les développements conceptuels relatifs au périphyton ;
2. l'interfaçage des entrées de PROSE : le projet GABI ;
3. l'interfaçage des sorties de PROSE.

2. Le compartiment périphytique

L'introduction d'un nouveau compartiment biologique dans PROSE nécessite de définir le mode de fonctionnement de ce nouveau compartiment et son mode d'échange avec les autres compartiments déjà présents (figure 1). Dorénavant, la rivière est, d'un point de vue biogéochimique, divisée en trois compartiments : la colonne d'eau, les sédiments et le périphyton. Le jeu de variables est homogène dans tous les compartiments. Les processus modélisés sont ceux représentés dans le modèle RIVE (Figure 2) (Billen et al. 1994; Garnier et al. 1995). Le modèle RIVE ne représentant que les processus de la colonne d'eau, des comportements spécifiques par couche ont été introduits, notamment pour les espèces algales (plus ou moins forte inhibition par la lumière) et pour les espèces particulaires (processus d'arrachage ou de remise en suspension).

2.1. Le modèle périphytique

2.1.1. Les producteurs primaires

La croissance des producteurs primaires suit la relation photosynthèse-irradiance suivante (Platt et al. 1977):

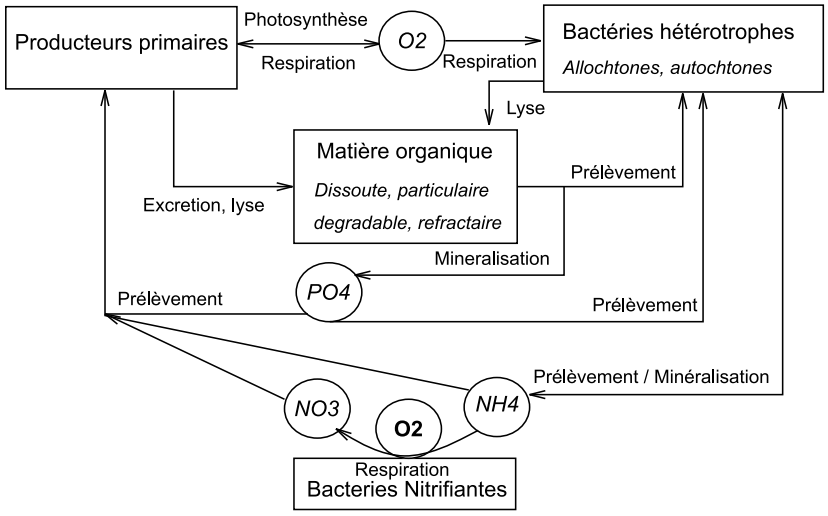

Figure 2: *Schéma conceptuel simplifié de* RIVE .

$P(z) = P_{max}(1 - e^{-\frac{\alpha I(z)}{P_{max}}})e^{-\frac{\beta I(z)}{P_{max}}}$, où P_{max} est la photosynthèse maximale, $P(z)$ and $I(z)$ la photosynthèse et l'irradiance à la cote z, α la pente de la zone limitée par la lumière de la courbe $P-I$, et β le coefficient de photoinhibition. L'irradiance suit la loi de Beer-Lambert : $I(z) = I_0 e^{-\eta z}$, où $\eta = \eta_{const} + \eta_{chla}[chla] + \eta_{sm}[SM]$. Les concentrations en chl *a* et matières en suspension (SM) sont entre crochets. La prise en compte de la chl *a* dans le calcul du coefficient d'extinction permet de représenter l'auto-ombrage. I_0 représente l'irradiance à la surface de l'eau pour les espèces de la colonne d'eau, ou l'irradiance au fond de la rivière pour les espèces périphytiques.

Chaque couche (eau ou périphytique) est distribuée verticalement pour le calcul d'un taux de photosynthèse moyen sur l'épaisseur de la couche $P_{moy} = \frac{1}{N}\sum_n P(z_n)$, où N représente le nombre de sous-couches. On suppose que les producteurs primaires périphytiques ne sont pas photoinhibés lorsqu'ils sont attachés par contre ils le sont lorsqu'ils sont en dérive dans la colonne d'eau (Hill et Boston 1991). Les paramètres des producteurs primaires périphytiques ont été calés sur le Grand Morin (Flipo et al. 2003).

2.1.2. L'activité bactérienne

Dans le biofilm, la croissance des bactéries nitrifiantes est représentée par :

$\mu = \mu_{max}\frac{[O_2]}{K_{O_2}+[O_2]}\frac{[NH_4]}{K_{[NH_4]}+NH_4}\frac{B_{max}}{B_{max}+[BN]}$, où μ_{max} est le taux maximum de croissance, B_{max} la biomasse maximum, et K_i le coefficient de saturation de l'espèce i.

La croissance des bactéries hétérotrophes est représentée par : $\mu = \mu_{max}\frac{[subs]}{K_{subs}+[subs]}\frac{[O_2]}{K_{O_2}+[O_2]}$, où $subs$ représente le substrat organique.

Les bactéries s'adaptent au milieu où elles évoluent. Par exemple, Fischer et Pusch (2001) ont observé que les bactéries sont plus grosses dans les sédiments que dans la colonne d'eau. Par ailleurs, McCarter (1999) a montré que des bactéries organisées en colonnie n'ont pas le même comportement que des bactéries isolées. C'est pourquoi, nous avons distingué les caractéristiques des bactéries en fonction de la couche où elles évoluent (Flipo et al. 2003).

2.2. Echanges de la phase dissoute

Les échanges dissous entre la colonne d'eau et les deux compartiments benthiques se font par diffusion turbulente dans la couche limite : $\beta(C_b - C_w)$ (Boudreau 1997). C_i est la concentration dans le benthos (indice b) et dans l'eau (indice w), β est le coefficient de transfert de masse exprimé par $\beta = u_* A S_c^n$ (Boudreau 1997), où A et n sont des coefficients empiriques, S_c est le nombre de Schmidt (environ 1000 pour les eaux de surface). La formulation $\beta = E_s u_*$ est retenue pour PROSE où u_* est la vitesse de cisaillement au fond. Elle s'exprime par $u_* = \sqrt{(\frac{\tau}{\rho})}$, où τ est contrainte moyenne de cisaillment et ρ, la masse volumique de l'eau.

2.3. Echanges particulaires

Le flux de sédiments provenant de la colonne d'eau vers le benthos est réparti entre les deux compartiments (sédiments et périphyton) au moyen du taux de recouvrement du fond de la rivière par le périphyton. Ce taux a été estimé sur le Grand Morin à 30 %. Ce taux par défaut peut être spécifié maille par maille si nécessaire.

2.3.1. Sédimentation, erosion

Le mode d'échange entre le benthos et la colonne d'eau diffère suivant le compartiment d'origine des particules.

Les échanges entre les sédiments et la colonne d'eau sont gérés par un processus d'érosion qui dépend uniquement de la puissance hydraulique de l'écoulement. Cette capacité érosive de l'écoulement est déterminée à l'aide de la théorie de la capacité de transport (Bagnold 1966) et s'exprime dans PROSE (Even et al. 2003) par :

$$F_{ero} = \zeta \frac{1}{\sum_i \chi_i \frac{\rho_i - \rho_w}{\rho_i}} \frac{P_{hyd}}{g} \tag{1}$$

où F_{ero} est la capacité érosive de l'écoulement , ζ est la part totale de la puissance hydraulique servant à maintenir les particules en suspension , χ_i est la fraction massique de l'espèce i dans le compartiment, ρ_i est la masse volumique de l'espèce i et ρ_w est la masse volumique de l'eau, P_{hyd} est la puissance hydraulique, g la gravité.

2.3.2. Les pertes en régime hydraulique stable

Les échanges entre la colonne d'eau et le périphyton sont représentés, en période de débit stable, à la fois par des pertes permanentes et une érosion similaire à celle exprimée au paragraphe précédent. Les pertes permanentes sont de 0.01 j^{-1} (DeAngelis et al. 1995; Dent et Henry 1999). Le compartiment d'origine des particules (colonne d'eau ou périphyton) est pris en compte pour modéliser les échanges. En effet, les espèces périphytiques (producteurs primaires attachés, espèces bactériennes représentatives du périphyton) ne sont soumises qu'à des pertes permanentes. Par contre pour les espèces originaires de la colonne d'eau (comme le phytoplancton par exemple) un coefficient de retention (ret) est défini. Ce paramètre représente la capacité du périphyton à retenir les particules sédimentées dans sa matrice. Ce paramètre varie entre 0 et 1. En considérant que B_w est la biomasse

périphytique originaire de la colonne d'eau, les pertes permanentes sont alors égales à $ret.B_w$ et le processus d'érosion est appliqué à $(1 - ret).B_w$.

2.3.3. L'arrachage

L'évolution de la biomasse périphytique est très souvent expliquable par l'hydrodynamique. Par exemple, lors d'une crue l'augmentation de débit contribue très fortement à l'arrachage du périphyton, si bien, qu'après l'évènement, la biomasse restée fixée peut être proche de zéro (Uehlinger et al. 1996). Il est donc extrêmement important de modéliser correctement ce processus.

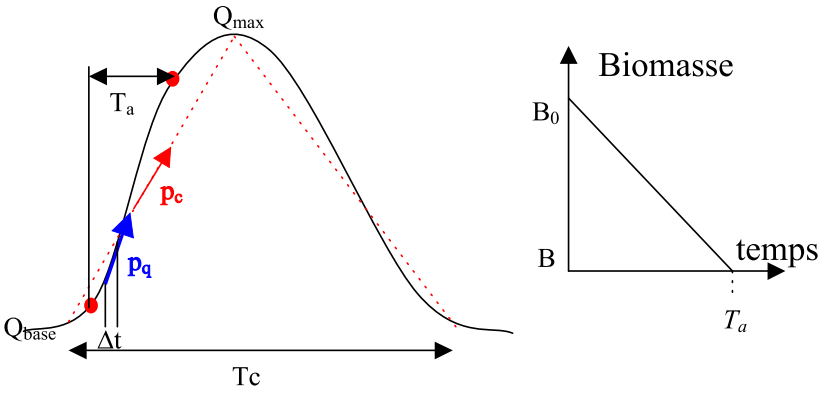

Figure 3: *Utilisation d'un hydrogramme pour caractériser l'arrachage. A gauche, réduction de l'hydrogramme réel (noir) en un hydrogramme idéalisé (pointillés rouges). A droite, schéma idéalisé de réduction de la biomasse lors d'une crue. T_c est le temps de passage de la crue, T_a le temps d'arrachage , p_c la pente critique au dessus de laquelle il y a effectivement arrachage, p_q la pente réelle sur un pas de temps ΔT (Flipo, 2001).*

Biggs et Close (1989) ont déterminé une relation entre l'intensité d'une crue, représentée par le rapport débit de la pointe de crue sur le débit de base ($\frac{Q_{max}}{Q_{base}}$), et le pourcentage total de biomasse arrachée pendant toute la durée de l'évènement extrême. Dans la démarche que nous allons détailler dans la suite, nous considérerons que Q_{base} correspond au débit d'étiage. La pente d'augmentation du débit (p_q) est une variable, pour la modélisation, plus pertinente que l'intensité de la crue (Figure 3). En reliant l'arrachage, à la pente d'augmentation du débit, il est alors possible d'estimer l'arrachage à chaque pas de temps de calcul sans connaître *a priori* l'intensité de la crue. Cette démarche n'est valable que si l'augmentation de la pente de débit est corrélée à l'intensité de la crue. Cette hypothèse semble cohérente avec la problématique des petits bassins versants qui sont *a priori* très réactifs.

La détermination de la loi d'arrachage en fonction de la pente d'augmentation du débit passe par une analyse de plusieurs événements extrêmes. Pour chaque évènement, on détermine un couple arrachage, pente de débit ($a(p)$,p). L'interpolation de tous ces couples donnent ensuite la loi d'arrachage (Figure 4). Pour un événement donné, on définit une pente critique de l'hydrogramme $p_c = 2\frac{Q_{max}-Q_{base}}{T_c}$, où T_c correspond au temps de passage de la crue (Figure 3). Cette pente fût déterminée sur le Grand Morin pour les crues des étiages 2000 et 2001. Ces pentes ont ensuite été reliées aux quantifications de Biggs et Close (1989) en considérant les intensités des évènements.

D'après Biggs et Close (1989), à p_c correspond une réduction totale de la biomasse lors du passage de la crue de $A(p_c)$ %. On estime que, pour des intensités de crue variant entre 0 et 3, $A(p_c)$ est nul, et

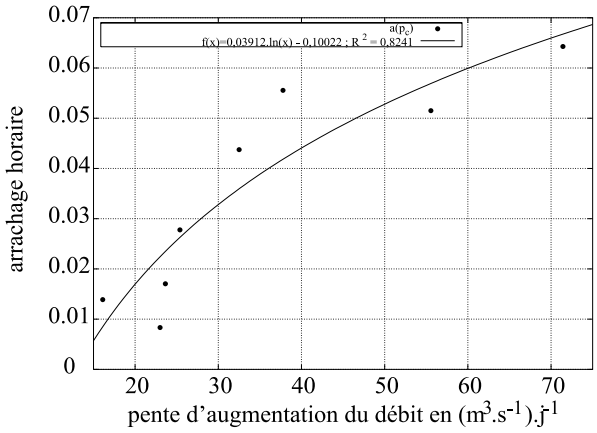

Figure 4: *Loi d'arrachage déterminée sur le Grand Morin.*

que,pour des intensités variant entre 3 et 7, $A(p_c)$ varie linéairement de 0 à 50 %. On admet aussi une variation linéaire de 50 à 95 % pour des intensités variant entre 7 et 25. Par contre au dessus d'une intensité de 25, on considère l'arrachage permanent à 95 % de la biomasse fixée.

Pour la modélisation de l'arrachage lors d'une crue, on admet qu'il n'y a arrachage que si la pente instantanée d'augmentation du débit (p_q) est supérieure à la pente critique de l'hydrogramme. L'analyse de l'hydrogramme de crue (Figure 3) montre que le temps d'arrachage effectif T_a est largement inférieur à $\frac{T_c}{2}$. En effet, en début de crue et à la pointe de crue, la pente instantanée d'augmentation du débit est plus faible que la pente critique de l'hydrogramme. On peut donc admettre que le temps caractéristique d'arrachage effectif est fonction de la pente critique de l'hydrogramme. Le taux maximum d'arrachage peut alors être exprimé à l'aide de la relation $a(p_c) = \frac{A(p_c)}{T_a(p_c)}$. Cette loi d'arrachage a été déterminée sur le Grand Morin (Figure 4).

On admet que touts les éléments composant le périphyton ne sont pas arrachés aussi facilement. On introduit donc un paramètre k_i qui caractérise l'arrachage de la variable i. Ce paramètre varie entre 0 et 1. $a(p_c)$ doit donc être pondéré par k_i afin de représenter l'arrachage relatif à l'espèce i. Pour une espèce donnée, les pertes instantanées par arrachage sont égales à la réduction instantanée de sa biomasse. L'évolution de la biomasse fixée de l'espèce i (B_i) est alors égale à :

$$\frac{\partial B_i}{\partial t} = -k_i a(p_q) B_i \tag{2}$$

3. L'interfaçage des entrées : le projet GABI

Le développement d'une interface pour les entrées de PROSE s'intègre dans le cadre d'un projet plus général qui a pour ambition de concevoir la *génération automatique de composants (beans) d'interfaces* pour des logiciels scientifiques, le projet GABI.

Il s'agit ici de créer des interfaces dont le but est de faciliter la création par l'utilisateur des fichiers de données qui sont nécessaires au fonctionnement d'un modèle. Nous n'aborderons pas les aspects de couplage avec les bases de données, qui est une forme d'interfaçage *via* les gestionnaires des bases.

Les fichiers lus par un logiciel ont des formats conçus par le programmateur du logiciel. Plusieurs

choix de programmation sont envisageables, parmi lesquels :

- couramment la lecture des données est intégrée au programme et est écrite dans le même langage que le code dans son ensemble, qu'il s'agisse d'un langage C/C++/FORTRAN/QUICK BASIC. Les avantages de cette approche sont de ne pas nécessiter l'apprentissage d'un langage supplémentaire. Cependant ces langages sont mal adaptés pour gérer la lecture de données qui s'avère fastidieuse à programmer d'une part et à utiliser ensuite d'autre part ;
- il existe en informatique des langages adaptés à l'analyse d'informations à partir d'une grammaire abstraite décrivant la structure des données : les parsers tels que LEX/YACC, FLEX/BISON (version GNU), JLEX/CUP (version JAVA). Ces outils d'analyse permettent une écriture claire et simple de code ayant pour vocation d'analyser des fichiers (en particulier les compilateurs de langage sont eux-même écrits à l'aide de parsers). Ils permettent ainsi de libérer le programmeur qui peut se recentrer sur le cœur du logiciel développé.

Un des points névralgiques de l'utilisation des logiciels est la manipulation d'un nombre important de données regroupées dans un système de fichiers complexes. Afin de simplifier cette opération, il est classique de procéder à un interfaçage, qui sera communément vu comme un système de *fenêtres* où l'utilisateur n'aura plus qu'à remplir les champs libres et effectuer un certain nombre de choix.

Outre l'aspect pratique, l'interfaçage doit remplir également une fonction de contrôle en obligeant l'utilisateur à valider des données jugées indispensables. Dans l'état actuel des choses, les interfaces sont généralement écrites au cas par cas. Leur conception doit être adaptée dès qu'une modification a lieu dans le logiciel interfacé. Le risque est grand alors, soit de figer complètement les choses et de limiter les perspectives d'évolution d'un logiciel, soit de passer un temps énorme dans la gestion de l'information.

Le projet GABI envisage d'automatiser la conception de l'interface en concevant un outil qui sera capable d'analyser la partie *code de lecture des données d'entrée* d'un logiciel et de générer le code d'interface automatiquement. Pour préciser les choses, nous dirons que l'outil GABI est un générateur d'éditeur de format pour le logiciel scientifique auquel il est appliqué.

3.1. Cahier des charges de l'application

Un certain nombre de choix ont été effectués pour le développements du projet.

3.1.1. Le langage JAVA

L'application est développée en JAVA afin de profiter des facilités des langages objets et des nombreux composants graphiques et de gestion des données fournis en standard. C'est un langage couramment utilisé qui bénéficie de nombreux développements. De plus il présente l'avantage d'être indépendant de la machine. Le concept de JAVA beans, qui permet d'écrire des composants réutilisables et adaptés à la programmation visuelle, a été utilisé.

3.1.2. Le choix de la grammaire

Compte tenu du grand intérêt présenté par les parsers (voir ci-dessous), le choix a été fait de baser le développement du générateur automatique d'interface sur l'analyse des parsers utilisés dans les

programmes. Pour des logiciels qui utiliseraient des parsers de type LEX et YACC ou FLEX et BISON (version GNU), un prétraitement pour une transcription de la grammaire en JAVA est prévu.

3.1.3. Réutilisabilité et extensibilité

Afin que l'outil puisse avoir un large spectre d'application, l'analyse de plusieurs types de grammaires est prévue ainsi que l'importation et l'exportation automatisée avec génération de description des types de données (DTD) ou de schémas.

Un grand soin a été apporté à la conception de l'outil qui doit pouvoir être manipulé facilement. Dans ce but, l'aspect visuel de l'interface doit pouvoir être aisément modifié par le concepteur du logiciel.

3.2. Analyse fonctionnelle de l'application

3.2.1. Le logiciel scientifique

Un programme consiste à interpréter et structurer des données, calculer et générer des résultats (figure 5). Un des principes du projet est de séparer la partie du code assurant la lecture et l'interprétation des données (partie syntaxe) et de la partie calculatoire du modèle, dans le sens où chacune des fonctions sera écrite dans des fichiers distincts. C'est de fait le cas avec l'utilisation des parsers qui font appel à des fichiers de définition spécifiques, la grammaire. Les fichiers de définition de la grammaire sont compilés pour générer du code en C. Le couplage avec le code général se fait *via* l'appel à une fonction `yyparse()`. On arrive alors à la représentation idéalisée d'un programme telle que présentée sur la figure 5.

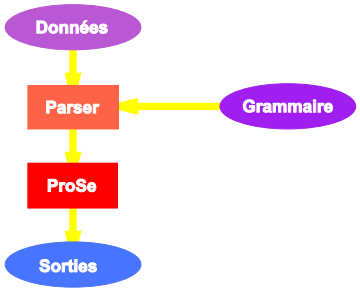

Figure 5: *Représentation schématique des fonctionnalités d'un logiciel de calcul scientifique.*

3.2.2. La grammaire

La grammaire permet de fournir une abstraction des données qui se composent d'un nombre limité d'éléments génériques.

Une grammaire se compose d'un lexique et d'une syntaxe :

- la définition d'un lexique (vocabulaire). Il est écrit dans un fichier spécifique, fichier LEX. Il permet d'associer simplement des chaînes de caractères à des référents qui deviennent les nouveaux objets manipulés. La déclaration peut être très simple :

```
"=" return LEX_EGAL;
```

établit que le référent du caractère = devient LEX_EGAL. Mais les définitions peuvent être plus compliquées :

```
[Pp][Hh][Yy] {yylval.integer = PHY; return LEX_VAR_BIO;}
[Oo]2 {yylval.integer = O2; return LEX_VAR_BIO;}
```

établit que le référent LEX_VAR_BIO peut être soit `phy` soit `o2`, écrits en minuscule ou en majuscule. Le tampon permet de récupérer également une valeur intermédiaire, qui est ici un entier (yylval.integer) et qui est initialisée à PHY ou O2, où PHY et O2 ont par ailleurs été définis comme des valeurs entières correspondant à des indices de tableau. On aura alors la possibilité d'un traitement générique d'une liste d'indices (de variables biologiques en l'occurence), quelle que soit leur ordre de lecture.

L'inclusion de nouveaux fichiers, fonction très appréciable, est commandée par un mot clé (`inclure` par exemple). Ce mot clé enclenche un mode[1] d'inclusion qui gouverne l'ouverture d'un fichier du nom qui suit. La lecture reprend en étant alors pointée vers le nouveau fichier ouvert jusqu'à sa fin ou jusqu'à ouverture d'un nouveau fichier.

Le même principe est utilisé pour gérer les unités : les crochets enclenchent le passage dans un mode unité. Chaque unité est alors interprétée dans ce mode avec association à un facteur de conversion.

- la définition d'une syntaxe. Elle est contenue dans un fichier spécifique YACC. Elle utilise le vocabulaire (les référents) définis dans la partie lexicale.

  Une syntaxe devant lire l'exemple suivant

```
# paramètre
Strickler = 20

# Points singuliers
Suresnes
Chatou

# Bief
Suresnes -> Chatou

# Apport
Colombes :  Suresnes {
      type  =  rejet
      x  =  1500 [m]
      q  =  [j] [m^3/s] inclure fichier_debit
}
```

  ressemble à cela :

[1] Un mode correspond à une analyse contextuelle de certaines chaînes de caractères. Ainsi une chaîne `phy` sera analysée comme un indice de tableau LEX_VAR_BIO en mode normal par défaut et sera analysé comme un nom de fichier en mode conditionnel d'inclusion, `cm` serait analysé comme un nom par défaut et comme une unité associée à la valeur 0.01 en mode conditionnel unité.

```
action : LEX_STRICKLER LEX_EGAL LEX_FLOTTANT
| LEX_NOM {cree_singularite();}
| LEX_NOM LEX_FLECHE LEX_NOM {cree_bief();}
| LEX_NOM LEX_DEUX_POINTS LEX_NOM attributs_apport {cree_apport();}

attribut_apport : LEX_ACCOLADE_OUVRANTE attributs LEX_ACCOLADE_FERMANTE

attributs : LEX_TYPE LEX_EGAL LEX_TYPE
| LEX_X LEX_EGAL distance
| LEX_Q LEX_EGAL unités valeurs

unités : LEX_UNITE LEX_UNITE {unite_jour=$1; unite_q = $2;}
```

Il est convenu que tous les identifiants, déclarés dans le fichier lexical et utilisés ici de manière générique, commencent par `LEX_`. `LEX_STRICKLER` aura par exemple été déclaré comme correspondant à la chaîne de caractères « strickler » ou bien « [Ss][Tt][iI][cC][kK][lL][eE][rR] » si on veut lire indifféremment des lettres majuscules et minuscules. Le sens de la plupart des identifiants est facilement compréhensible. Des identifiants plus généraux ont été déclarés : `LEX_FLOTTANT` et `LEX_NOM`.

Cette écriture permet une grande clareté qui fait que l'on comprend assez rapidement ce qui est lu et ce qui est fait, d'où une grande facilité de modification et d'amélioration.

Une syntaxe est une succession d'actions s'enchaînant dans une série de ET (liste) et de OU (|). Les différentes options possibles quant à l'ordre des données s'écrit alors très simplement. L'ordre qui est écrit ici prévoit que l'on lise SOIT un `strickler` (paramètre) SOIT un `nom` (singularité) SOIT un `nom -> nom` (bief) SOIT un `nom : nom` (apport avec attributs). Chaque enchaînement peut se présenter dans un ordre quelconque (succession de OU). Cette souplesse permet de gérer simplement les informations.

Cette souplesse de formatage (possibilité de lire un bief ou une limite dans n'importe quel ordre) permet de coupler très facilement plusieurs rivières. Il serait possible ici de prendre les fichiers existants relatifs à chaque rivière, tels quels, sans reformatage et ce d'autant plus facilement qu'on dispose de la fonction « `inclure` ».

Chaque sous action commande une fonction qui permet de formater les données dans des objets manipulés par le logiciel.

Les actions se déclinent les unes les autres à partir d'une action mère. La structure des données se représente donc facilement sous la forme d'une arborescence d'actions, se déclinant les unes après les autres jusqu'aux feuilles terminales de l'arbre correspondant aux mots de base du langage (figure 6).

En utilisant les parsers, les messages d'erreur sont très explicites, du type *lu ... au lieu de ..., ligne ... du fichier ....*

Outre les nombreux avantages de programmation que représentent l'utilisation d'une grammaire on retiendra donc qu'elle oblige à sérier l'information, le lexique et la syntaxe étant définis dans des fichiers bien spécifiques.

Remarque Classiquement les données sont traitées en utilisant le langage choisi pour le reste de la programmation du code, langage C/C++/FORTRAN/….

La lecture des données se fait à l'aide de fonctions `read` (FORTRAN) ou `scanf` (C) qui nécessitent la connaisance *a priori* des formats devant être lus et ce de manière précise (un entier ne peut pas être lu à la place d'un nombre flottant ou d'une chaîne de caractère, etc).

En général, le concepteur prévoit quelques options possibles, mais pas trop, du fait de l'explosion combinatoire. Le simple fait de pouvoir rentrer une série de paramètres dans un ordre quelconque devient vite fastidieux. Le concepteur préfèrera prévoir un ordre *a priori*. À la charge de l'utilisateur de se plier à cet ordre.

Devant la difficulté à gérer les conditions, la quantité d'informations à lire doit souvent être prédéfinie : on vous demandera donc combien de biefs vous allez lire, combien de paramètres. Il y a donc surinformation puisque l'information « nombre de biefs » est déjà contenue dans la liste des biefs. Le risque est grand, souvent, de modifier la liste en oubliant de modifier le nombre.

Un type d'information est aussi souvent géré en un endroit unique : vous ne pourrez pas lire un bief, un paramètre, un apport, puis de nouveau un bief. Tous les biefs seront lus ensemble, tous les apports ensemble. Peut être l'intérêt d'un mélange de genre ne saute-il pas aux yeux du lecteur, mais imaginez que vous vouliez simuler une rivière Seine et une rivière Marne. Vous avez déjà simulé chaque rivière séparément et donc les fichiers de définition existent déjà. Sans utiliser de grammaire, il faudrait lire tous les biefs en même temps, puis tous les apports en même temps. Cela impliquerait de fusionner tous les fichiers pour constituer de nouveaux fichiers contenant les informations reclassées. En d'autres termes, cette absence de grammaire peut se traduire par d'importantes pertes de temps du point de vue de l'utilisateur puisque ce dernier doit alors effectuer deux fois un travail déjà effectué sur des rivières séparemment! En conclusion, la lecture des données au sein même du programme scientifique va de paire avec une restriction à des formats rigides pour éviter une complexité exponentielle dans les procédures de lecture. Ces formats sont difficilement prévisibles et engendrent un manque de lisibilité du code.

3.2.3. Un interpréteur de grammaire

Nous avons vu que la partie syntaxique se présentait simplement sous la forme d'une série d'actions s'enchaînant les unes aux autres à partir d'une action mère et que chaque action se composait elle-même d'une sous-action ou bien était définie à l'aide des référents lexicaux. L'analyse des fichiers lexical et syntaxique permet ainsi simplement, grâce à une interface JAVA, de générer une abstraction des données sous la forme d'un *arbre* (figure 6).

Les référents sont quant à eux de 3 types :

identifiant un référent - une chaîne de caractère définie (exemple : `LEX_EGAL` et `=`)

liste un référent - une liste de chaînes de caractères (exemple : `LEX_VAR_BIO` valant `PHY, O2`, ...)

valeur un référent - une chaîne indéfinie (exemple : `LEX_NOM` sera toute chaîne de caractère non définie dans le lexique).

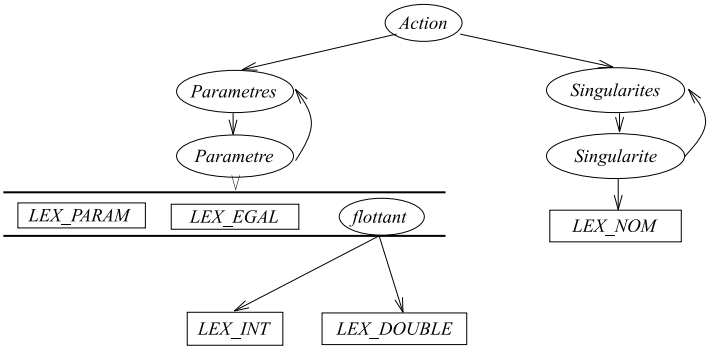

Figure 6: *Représentation structurée de données. Étiquettes rondes : nœud ou action* **composite**, *étiquettes carrées : feuilles (équivalents aux référents lexicaux de type* **valeur**, **liste** *ou* **identifiant**).

On ajoute un quatrième type d'objet qui sera l'objet **composite** correspondant aux nœuds de l'arbre.

La prétraitement permet de générer un arbre des données, géré simplement à l'aide des outils disponibles avec JAVA.

3.2.4. L'interface

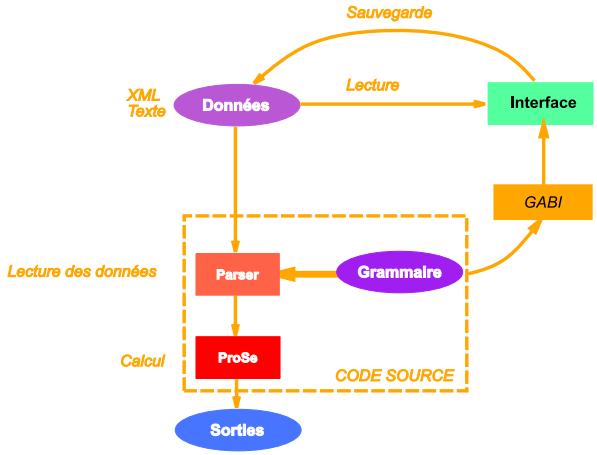

Figure 7: *Schématisation des fonctionnalités de l'interface.*

Les fonctionnalités de l'interface sont les suivantes (figure 7) :

- elle est alimentée d'une part par la structure des données, fournie sous forme d'une arborescence ;
- elle doit gérer l'arborescence et permettre des modifications. Dans un état initial, les données se trouvent dans un état virtuel. L'objectif est que l'utilisateur vienne faire ses choix et remplisse au fur et à mesure les champs restés libres (choix dans une liste, valeur à fournir). Ce faisant,

l'utilisateur valide un nouvel arbre effectif qui correspond aux données qu'il a entrées. Cet arbre, de même que l'arbre virtuel initial, est lisible par l'interface et modifiable à loisir ;

- elle doit permettre la sauvegarde et l'échange en différents formats (texte, arbre, XML, ...).

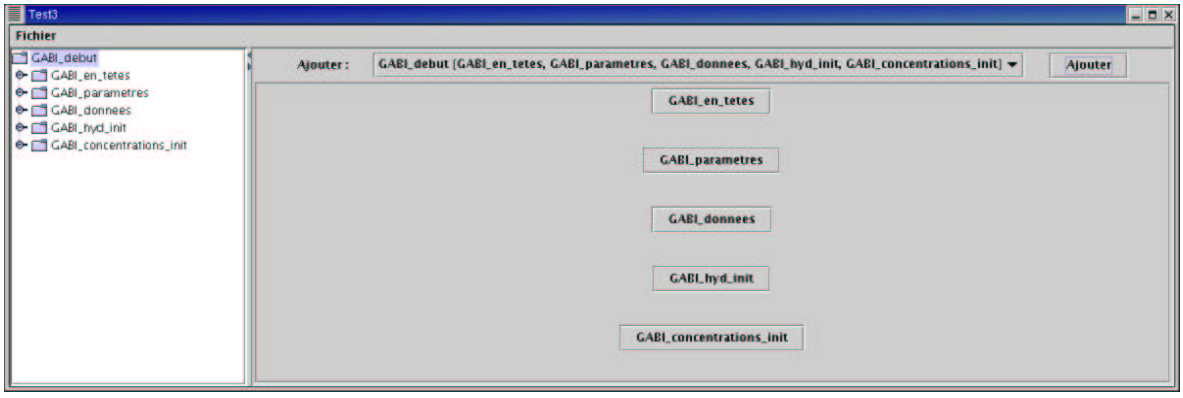

Figure 8: *Fonctionnement de l'interface. A gauche, actualisation de l'arborescence et possibilité de sélectionner une action. A droite en haut, activation de la liste de choix pour l'action sélectionnée à gauche, en bas, apparition des champs correspondant aux actions choisies. C'est le lieu de validation de données.*

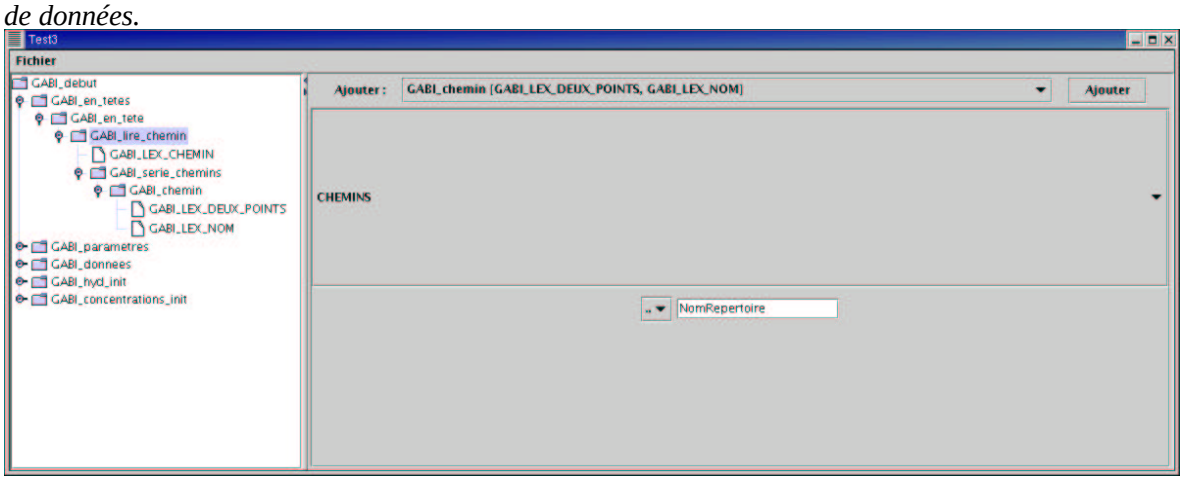

Figure 9: *Fonctionnement de l'interface après avoir choisit l'action* `en_tete`*, puis chemins. À droite en bas, apparition des champs* **identifiants** `CHEMINS` *et « : » et du champ* **valeur** *à remplir par l'utilisateur.*

L'interface se présente donc de la manière suivante (voir les figures 8 et 9) :

- une barre de menu en haut qui permet de gérer les différents prétraitements (conversion du fichier LEX ou YACC, génération de l'arbre des données, conversion de format), de quitter, de sauvegarder ;
- un première fenêtre à gauche où s'affiche l'arbre effectif. Au départ, seule l'action mère est activée. Au fur et à mesure des sélections (dans la fenêtre de droite), l'arbre est mis à jour et permet la sélection des actions successives ;

- une fenêtre en haut à droite où s'affiche la liste des actions possibles à partir d'une action sélectionnée à gauche. Ainsi dans l'exemple de la figure 8, l'action mère `GABI_debut` se décline comme étant une lecture d'en-tête (définition des chemins d'accès aux données) PUIS de paramètres PUIS de données (points singuliers, biefs, apports, ...) PUIS d'hydraulique initiale, ... Une des sous-actions ayant été choisie (`en_tete`), un choix possible découlant de cette action a été sélectionné dans la partie haute de la fenêtre de droite (`lire_chemin`) (Figure 8).

- une fenêtre basse à droite dans laquelle s'affichent les étiquettes de l'action sélectionnée. Dans la figure 8, il ne s'agit que d'étiquettes **composite** correspondant à des nœuds. Dans la figure 9 les choix ont été poussés jusqu'aux feuilles terminales. La partie basse de la fenêtre de droite affiche alors le format pour rentrer un chemin : le mot clé `CHEMINS` et une suite de « `: NOM` ».

3.3. État de l'avancement du projet

La partie prétraitement est actuellement entièrement réalisée : transcription de fichiers LEX et YACC en JLEX et CUP, génération de l'arbre.

Une première version de l'interface existe qui permet de commander le prétraitement et la génération de l'arbre, de charger un arbre, de le modifier et de le sauvegarder.

L'outil a actuellement fonctionné sur deux versions de grammaire de PROSE relativement différentes et ce de manière tout à fait satisfaisante.

La gestion de la fonction inclure n'est pas encore effective. Une solution est envisagée mais non encore implantée.

Le prétraitement et l'interface lui-même sont gérés à partir de deux fenêtres dictinctes qui devront être fusionnées.

Les étapes manquantes sont essentiellement les liens avec les fichiers texte : lire un fichier texte et générer l'arbre correspondant, sauvegarde sous format texte.

L'interface elle-même doit être améliorée. Cet aspect reste facilement contrôlable par l'utilisateur. Ce dernier peut créer lui-même sont propre style de fenêtrage et appeler simplement les commandes de base qui servent à intégrer la génération de l'interface.

Il est prévu d'étendre le champ des grammaires, d'échanger automatiquement des données au format XML, de coupler avec un interfaçage des sorties.

4. L'interfaçage des sorties

Les sorties brutes d'un logiciel de calcul se composent de résultats (nombres). Ces résultats peuvent être interprétés à l'aide de visualisations grahiques ou de traitements statistiques, de calculs d'erreur, ...

Dans la suite du propos nous nous limiterons à considérer les aspects graphiques. Pour l'interfaçage des sorties on s'attend donc à disposer d'un outil qui permettrait simplement à l'utilisateur d'effectuer ses choix (choix de variable, de lieu, de fichier) et de visualiser instantanément les résultats désirés. Cette action comprend elle-même deux parties :

- les aspects graphiques sont liés à l'utilisation de logiciels graphiques, indépendant du logiciel de calcul. À la limite on pourrait demander au logiciel de ne fournir que des fichiers de nombres et chaque utilisateur, en fonction de ses habitudes, serait libre d'utiliser le logiciel graphique qui lui convient ou auquel il est le plus habitué. Chaque logiciel ayant des formats particuliers, il est nécessaires de prévoir des post-traitements, car il n'existe pas un format type unique qui marcherait pour tout logiciel graphique. Il a cependant été prévu que PROSE génère directement des sorties dans des formats requis pour certains logiciels, choisis selon certains critères que nous verrons ci-après. Les fichiers de commande propres à ces logiciels graphiques sont également générés pas PROSE.

- le deuxième aspect consiste en une mise en forme de l'interface, qui ne passerait plus par des commandes en ligne (habituelles sous UNIX/LINUX), mais par une interface « graphique » (fenêtrage) qui permettrait facilement à l'utilisateur de faire ses choix et de commander un type graphique plutôt qu'un autre. Il est au final envisagé que cette interface soit couplée avec l'interfaçage des entrées pour ne disposer que d'un unique « interface PROSE », qui permettrait de lancer et interpréter les simulations.

Seule la première étape a aujourd'hui été réalisée et nous présenterons donc les logiciels utilisés, les couplages réalisés et des exemples de graphes pouvant aujourd'hui être facilement générés en sortie de PROSE.

Deux critères ont guidé nos choix de logiciel :

- des logiciels libres pour la pérénité ;

- la recherche du meilleur compromis simplicité/ergonomie/esthétique.

Le critère « logiciel libre » réduit grandement le champ des investigations. Il nous est cependant apparu d'autant plus important qu'un petit tour d'horizon des logiciels utilisés de-ci de-là montre qu'aucun standard n'existe vraiment en terme de logiciel graphique. Il apparaît aussi que d'excellents outils existent dans le domaine public et qu'il serait dommage de s'en priver.

Si les aspects monodimensionnels sont relativement simples à traiter, le fait de considérer des aspects bidimensionnels complique sensiblement la question des traitements graphique. Il a donc été prévu de disposer d'une part d'un logiciel simple permettant des sorties monodimensionnelles rapides et d'autre part d'un logiciel permettant des sorties bidimensionnelles. Trois logiciels sont utilisés :

- logiciel DOT pour le tracé de graphes (http://www.research.att.com/sw/tools/graphviz/) ; fichiers de commande `.dot` ;

- le logiciel GNUPLOT utilisé uniquement pour les aspects monodimensionnels ; fichiers de commande de type `.exe.gnu`[2] ;

- le logiciel OPENDX (http://www.opendx.org) pour les tracés bidimensionnels ; fichiers de commande `.dx` et `.general`.

[2]Convention interne à PROSE. Dans la suite du texte, tout fichier mentionné avec une extension `.exe.gnu` représente un fichier de commande de GNUPLOT, directement exploitable par le logiciel graphique.

4.1. Interprétation graphique des entrées

Une option de PROSE permet aujourd'hui de générer des sorties graphiques d'un certain nombre d'informations fournies en entrée, dans un but de vérification :

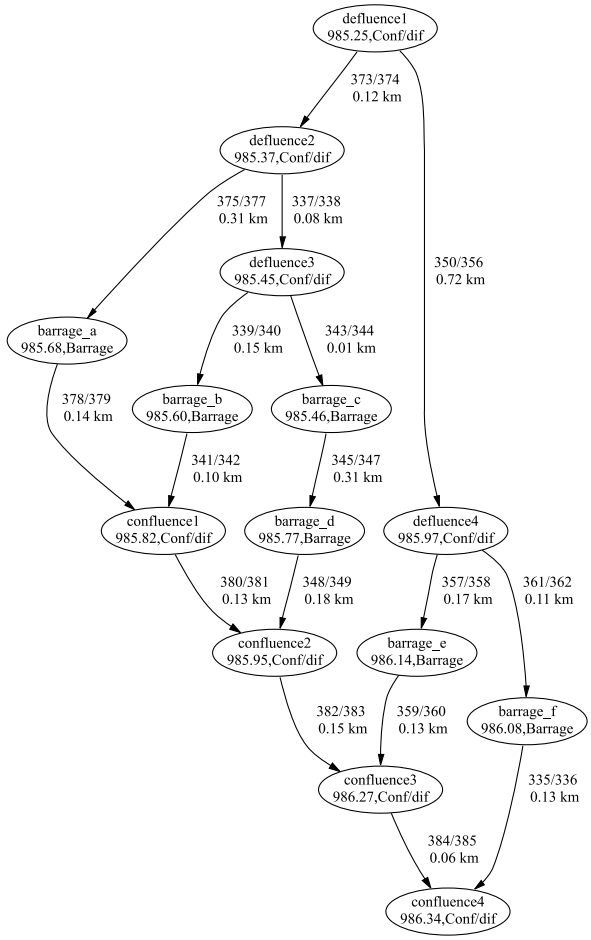

Figure 10: *Graphe d'un réseau de cours d'eau sur le Grand Morin, à Crécy La Chapelle. Résultat du logiciel* PROSE.

- le graphe de la rivière (figure 10, fichier **schema.dot**), permettant de visualiser le nom des singularités avec leur PK et leur type ainsi que les connexions (biefs) avec les numéros des première et dernière mailles, la longueur totale du bief, la liste des apports dans le bief. Le logiciel dot offre la possibilité d'une interface graphique pour le tracé de graphe. Les formats de PROSE ont été adaptés pour être compatibles avec ceux de ce logiciel, ce qui permet de définir la rivière (points singuliers + biefs) à travers cette interface ;
- un tracé des profils transversaux (bathymétrie) utilisée par l'utilisateur ;
- le tracé du profil longitudinal des cotes de fond Deux cotes sont tracées : la cote de fond correspondant à la cote minimale du profil (Z_f) et la cote de fond moyenne, correspondant à

la moyenne des cotes telles que $Z < \frac{S}{L}$, où S est la section mouillée et L la largeur au miroir. Dans le cas de profils simples (carrés, trapèzes) les deux valeurs sont confondues ;

- tracé des cotes de tous les seuils représentant un barrage et comparaison avec les cotes de fond amont et aval du barrage ;
- un tracé des zones de méandre dans le cas d'une information géoréférencée (Figure 11).

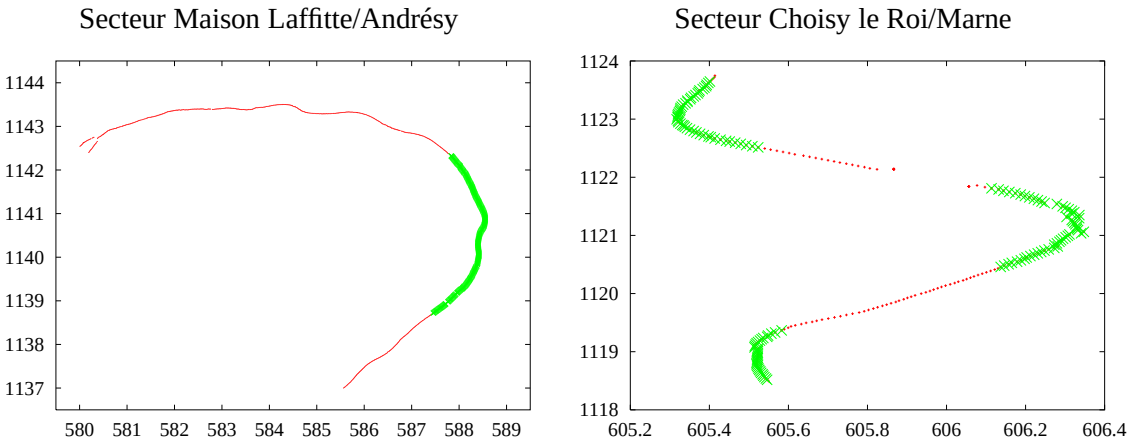

Figure 11: *Zones de méandre (en vert) identifiées par le logiciel* PROSE *dans le secteur Seine Aval-Andrésy et dans le secteur Choisy-le-Roi/confluence Marne.*

Par ailleurs, en sortie standard, un fichier texte **pk** est crée, qui permet de connaître les Pk des points singuliers et apports.

4.2. Les graphes monodimensionnels

Une majorité des sorties de PROSE est interprétable à l'aide d'un logiciel monodimensionnel (2 axes) tel que le logiciel GNUPLOT. Compte tenu des formats relativement simples, un chargement des fichiers de résultats pour affichage sous EXCEL est également réalisable. Le couplage est aujourd'hui direct, à savoir que l'utilisateur n'a plus à sa charge de créer les fichiers de commande de GNUPLOT. Des fichiers type sont crées par le logiciel (fichiers avec une extension .exe.gnu) et permettent une visualisation rapide des résultats.

Outre les exemples présentés ci-dessus, le logiciel permet de présenter :

- des évolutions temporelles de la valeur moyenne sur la section, en point fixe : à un PK donné, évolution d'une variable (débit, vitesses, concentration, ...) en fonction du temps (figure 12) ;
- des évolutions temporelles comparées des valeurs par tubes à un PK donné (figure 12) ;
- des évolutions longitudinales de valeurs moyennes sur la section à différents instants (figure 13) ;
- les profils transversaux en point fixe et à différents instants, (cas d'une simulation bidimensionnelle) ;

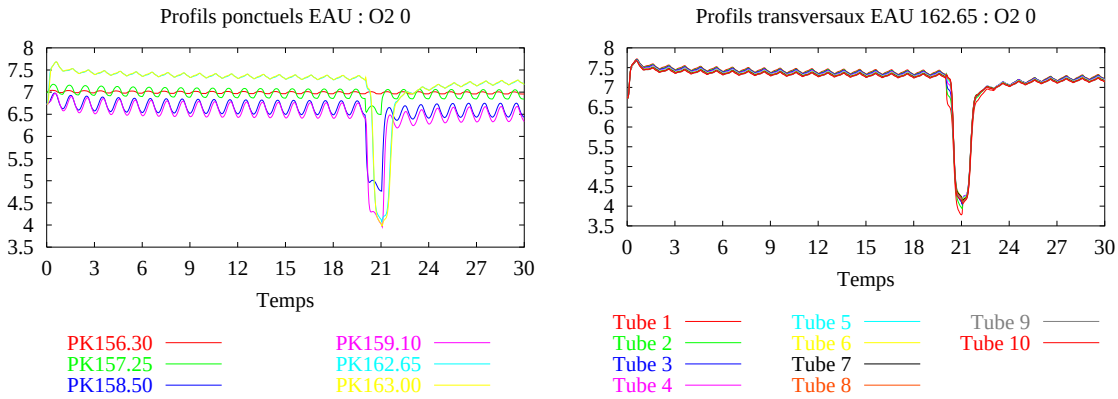

Figure 12: *Résultats en points fixes en fonction du temps, valeurs moyennes en différents PK à gauche (fichier « pp_EAU.moy.exe.gnu ») et valeurs dans chaque tube au droit d'Ivry à droite (fichier « pp162.65_EAU.exe.gnu »).*

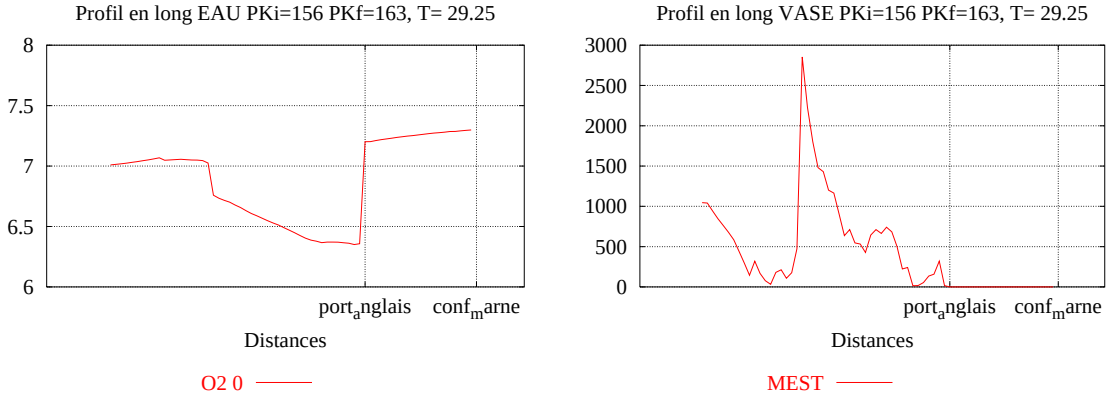

Figure 13: *Évolutions longitudinales entre Choisy le Roi et la Marne de l'oxygène dans l'eau et des MES totales dans le vase. Tracés à l'aide des fichiers* **$RESULT/pl_pk156_pk163_seine_EAU_O2.exe.gnu** *et* **$RESULT/pl_pk156_pk163_seine_EAU_MEStot.exe.gnu**.

- des évolutions temporelles de flux et les bilans de masse associés (figure 14).

Les graphes peuvent être tracés dans les trois compartiments : eau, vase et périphyton.

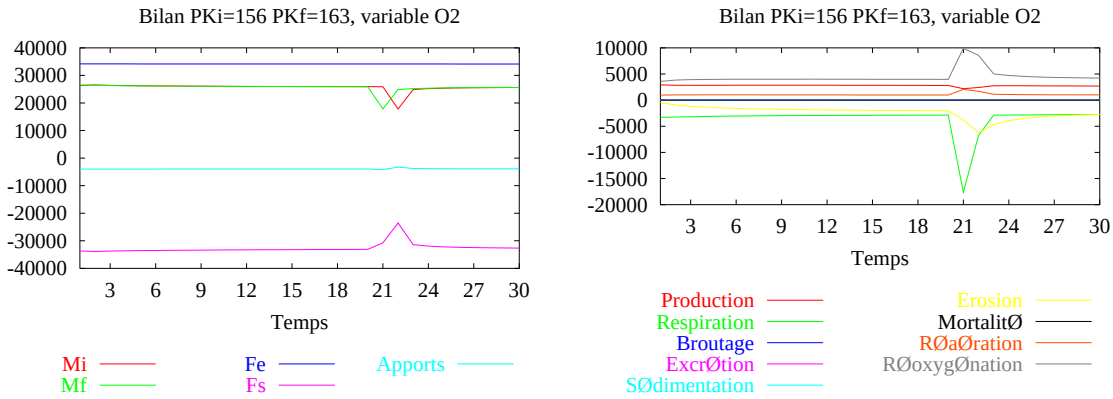

Figure 14: *Termes du bilan de l'oxygène entre Choisy le Roi et la confluence Marne : stocks (masse initiale et finale) et flux (entrant, sortant) et flux échangés par les processus biologiques. Fichier* **bilan_pk156_pk163_seine.exe.gnu**.

4.3. Les graphes bidimensionnels

Les aspects bidimensionnels sont beaucoup plus complexes à gérer. Les développements effectués pour PROSE se s'appuient sur le logiciel OPENDX, logiciel libre dérivé du logiciel Visualization Data Explorer d'IBM. Le logiciel (http://www.opendx.org) est disponible pour un grand nombre de plateformes.

Outre ses nombreuses possibilités, OPENDX est ici essentiellement utilisé pour la visualisation graphique des résultats 2D de PROSE et la possibilité d'effectuer des animations. Il offre également une possibilité d'interfaçage : liste de paramètres, liste de sorties, ...

Moyennant le fait que les résultats soient placés dans un répertoire correct, l'utilisateur doit lancer, sous OPENDX, un programme développé spécialement pour interpréter les sorties de PROSE : PROFIL_LONG.net pour une vue de dessus de la rivière à un instant donné, PROFIL_PONCTUEL.net pour une visualisation en 2D de l'évolution des concentrations transversales en un PK donné.

La commande DX provoque l'affichage d'une fenêtre, présentée sur la figure 15.

L'utilisateur n'a alors plus qu'à lancer l'application à l'aide de la commande RUN VISUAL PROGRAMS. Après sélection du programme PROFIL_LONG.net, les fenêtres de choix (de variables, de fichiers, de commandes d'animation) s'ouvrent ainsi qu'une visualisation des graphiques par défaut : premier instant du premier secteur de profil en long et de la première variable. La fenêtre de choix (figure 16) s'affiche dans le même temps. Elle permet à l'utilisateur de changer de secteur (changer de nom de fichier), de changer de variable (variable trace sur l'exemple), de lancer l'animation, ... La visualisation se présente sous une forme de deux graphes : une vision en vue de dessus permettant une visualisation 2D (repère en X-Y et coloration pour les valeurs de concentrations). En parallèle s'affiche le graphe 1D correspondant aux valeurs longitudinales moyennes (voir figure 17).

Le tracé graphique 2D n'est possible que si les données sont géoréférencées. PROSE autorise deux types de référencement pour la bathymétrie :

- un repérage transversal relatif pour chaque profil (repère quelconque, généralement 0 sur une rive) et chaque point de profil est défini par une abscisse transversale Y. Aucun repérage des

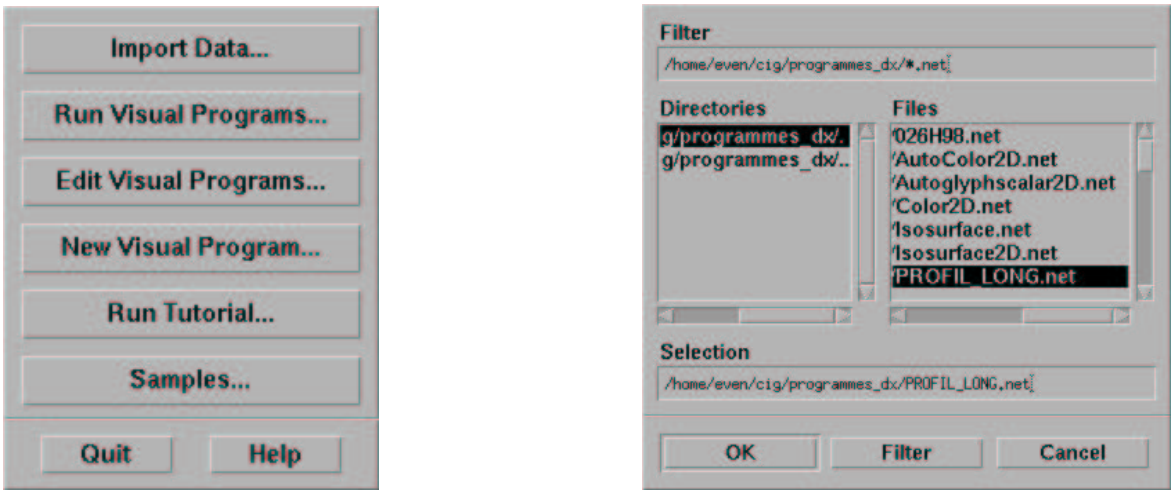

Figure 15: *A gauche, fenêtre apparaissant au lancement de* DX *; à droite, fenêtre de choix d'un programme (PROFIL_LONG.net) après sélection de* RUN VISUAL PROGRAMS *dans la fenêtre de gauche.*

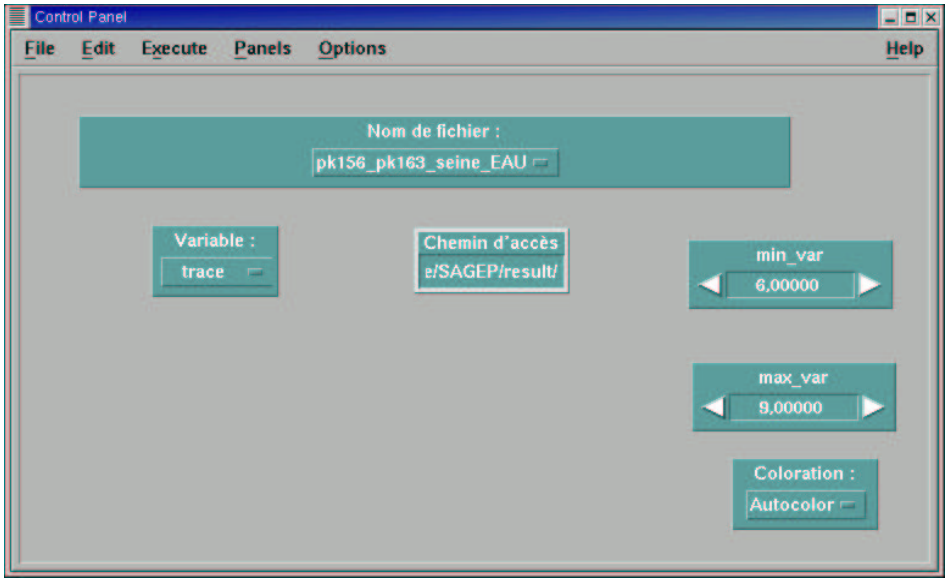

Figure 16: *Fenêtre de choix pour le programme PROFIL_LONG.net.*

profils les uns par rapport aux autres n'est alors possible. Les seuls tracés possibles sont selon une abscisse curviligne longitudinale. Seule les valeurs moyennes transversales sont prises en considération en sortie.

- un repérage géoréférencé (coordonnées Lambert par exemple) et chaque point de profil est positionné en absolu à l'aide d'un couple de point X et Y. Seule cette solution permet des tracés graphiques 2D. Les données bathymétriques acquises par le SNS sont géoréférencées. La bathymétrie de la Seine entre Suresnes et Maison-Laffitte est aujourd'hui à jour dans le système Lambert II, par pas de 20 m.

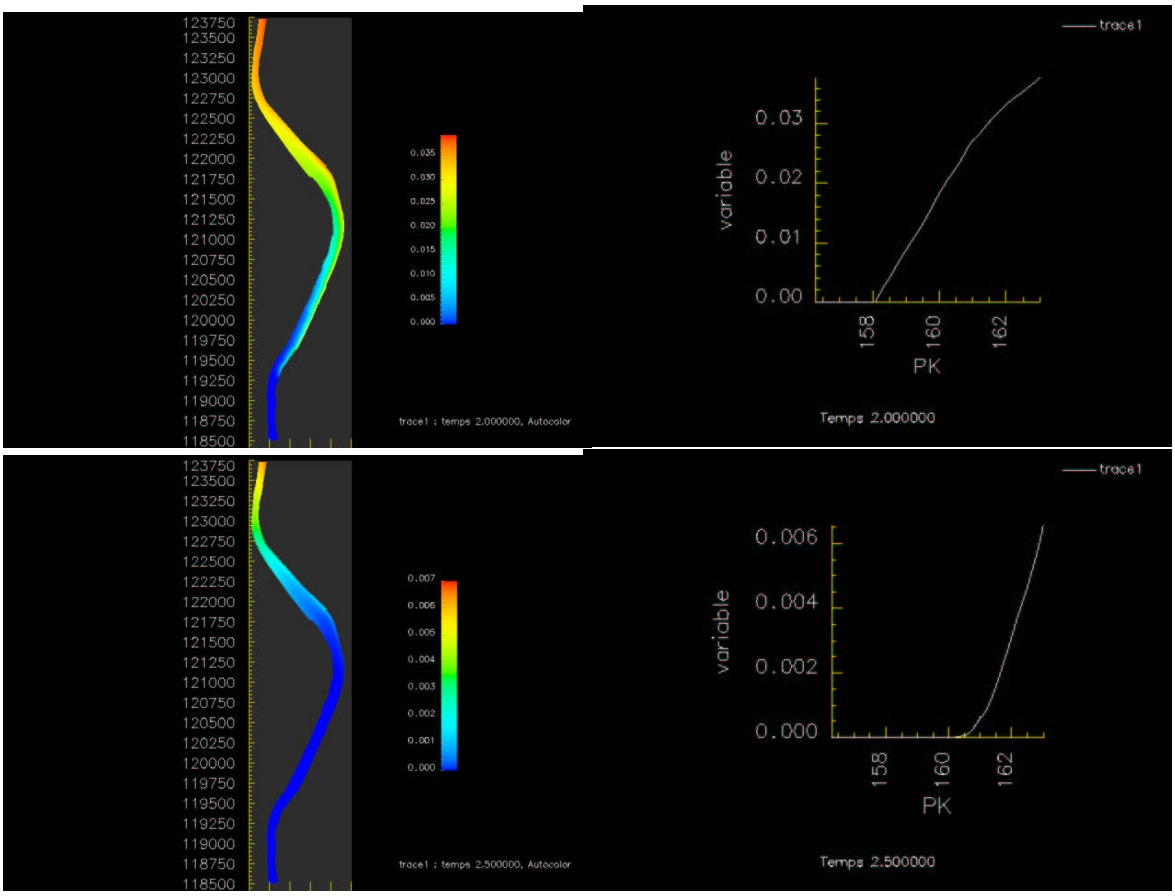

Figure 17: *Vue du traçage du panache de Seine Amont.*

5. Conclusion et perspectives

Le logiciel PROSE est en évolution permanente, tant du point de vue conceptuel que du point de vue des outils informatiques mis en œuvre.

Trois améliorations conceptuelles importantes, le traitement du transport bidimensionnel, le couplage avec un modèle sédimentaire complet et le traitement de la couche périphytique ont été intégrés à la version 3.

Parallèlement à ces développements, l'accent est aujourd'hui mis sur les aspects interfaçage et portabilité. Les tâches suivantes ont été réalisées : participation au projet GABI dont l'objectif est la conception d'un outil capable de générer automatiquement des interfaces pour des logiciels scientifiques et dont une première version a été appliquée au logiciel PROSE ; l'interfaçage des sorties avec la création des post-traitements pour assurer la relation directe avec des logiciels graphiques.

Le travail relatif à la conception d'un outil de création automatique d'interfaces doit être poursuivi : traitement de nouvelles fonctions gérées par les parsers, nouveaux formats traités par l'interface, amélioration et fiabilisation de certaines fonctions, mise en forme (ergonomie).

Un autre aspect en perspective concerne le couplage avec des bases de données et les outils de

gestion de ces bases ainsi que le couplage avec des SIG, travail qui doit être mené en collaboration avec les autres équipes du PIREN SEINE. Outre les aspects de prétraitement permettant d'établir la communication entre ces bases et le logiciel PROSE, il s'agit de faire évoluer les bases pour intégrer les concepts supplémentaires introduits par PROSE. Il s'agit donc de penser une base de données, non plus dédiée à tel ou tel logiciel, mais suffisamment générale pour qu'elle soit indépendante des logiciels. Cette base générale doit alors permettre d'alimenter des bases dérivées, spécifiques à chaque logiciel.

L'objectif final est d'aboutir à une interface globale pour le logiciel PROSE qui sera capable de gérer, ausi bien en entrée (création de fichiers texte ou XML, lien avec les SIG) qu'en sortie, les liens avec les différents outils. Cette interface ne pourra être mise en place que lorsque les étapes citées auront été réalisées.

Bibliographie

Bagnold, R. (1966). An Approach to the Sediment Transport Problem from General Physics. Physiographic and Hydraulic Studies of Rivers. Technical report, Geological Survey Professional paper 422-I.

Biggs, B. and M. Close (1989). Periphyton biomass in gravel bed rivers : the relative effects of flows and nutrients. *Freshwater Biology 22*, 209–231.

Billen, G., J. Garnier, and P. Hanset (1994). Modelling phytoplankton development in whole drainage networks: The RIVERSTRAHLER model applied to the Seine river system. *Hydrobiologia 289*, 119–137.

Boudreau, B. P. (1997). *Diagenetic Models and their Implementation*. Springer.

DeAngelis, D., M. Moreau, D. Neergaard, P. Mulholland, and E. Marzolf (1995). Modelling nutrient-periphyton dynamics in streams : the importance of transient storage zones. *Ecol. Model. 80*, 149–160.

Dent, C. and J. Henry (1999). Modelling nutrient-periphyton dynamics in streams with surface-subsurface exchange. *Ecol. Model. 122*, 97–116.

Even, S., J.-M. Mouchel, and M. Poulin (1998). Améliorations apportées au modèle ProSe relatives au transport de particules. Technical report, PIREN Seine.

Even, S. and M. Poulin (2001). Transport convectif et dispersif dans la seine de choisy-le-roi à ivry-sur-seine à l'aide du logiciel prose. mise en œuvre d'une version à tubes de courant. Technical report, SAGEP - Armines, École des Mines de Paris, Centre d'Informatique Géologique. LHM/RD/2001/05.

Even, S., M. Poulin, J.-M. Mouchel, M. Seidl, and P. Servais (2003). Modeling oxygen deficits in the seine river downstream of combined sewer overflows. *Ecol. Model.*. accepted.

Fischer, H. and M. Pusch (2001). Comparison of bacterial production in sediments and epiphyton and the pelagic zone of a lowland river. *Freshwater Biology 46*, 1335–1348.

Flipo, N. (2001). Site atelier du grand morin : modélisation biogéochimique et étude d'un compartiment benthique. Master's thesis, DEA HHGG, ENSMP.

Flipo, N., S. Even, M. Poulin, M.-H. Tusseau-Vuillemin, T. Améziane, and A. Dauta (2003). A distributed benthic biofilm modelrepl : dynamical biogeochemical functioning at the river scale (the grand morin, france). *Ecol. Model.*. submitted.

Garnier, J., G. Billen, and M. Coste (1995). Seasonal succession of diatoms and chlorophycae in the drainage network of the river Seine: Observations and modelling. *Limnol. Oceanogr. 40*(4), 750–765.

Hill, W. R. and H. L. Boston (1991). Community development alters photosynthesis-irradiance relations in stream periphyton. *Limnol. Oceanogr. 36*(7), 1375–1389.

McCarter, L. (1999). The multiple identities of vibrio parahaemolyticus. *J. Molec. Microbiol. Biotechnol. 1*(1), 51–57.

Platt, T., K. Denman, and A. Jassby (1977). *The sea - Ideas ans Observations on Progress in the Study of the Seas - Marine Modelling*, Volume 6, Chapter Modelling the productivity of phytoplankton, pp. 807–856. John Wiley and Sons.

Thery, L., L. Simon, and M. Poulin (1993, oct). Simulation numérique du transport de substances dissoutes en rivière à l'aide d'un modèle à tubes de courant. rapport final. Technical report, SEDIF-CGE-CERGRENE-CIG.

Uehlinger, U., H. Bührer, and P. Reichert (1996). Periphyton dynamics in a floodprone prealpine river: evaluation of significant processes by modelling. *Freshwater Biology 36*, 249–263.