MATAPLI, 1987, 11, 5-19

SYSTEMES EXPERTS - APPLICATION A LA MODELISATION EN BIOLOGIE.

Alain Pavé
Laboratoire de Biométrie
et de Biologie des Populations
U.A. CNRS 243
Université Claude Bernard
69622 VILLEURBANNE Cedex

Bien que relativement ancienne, cette démarche fut longtemps assez largement ignorée par beaucoup de biologistes. En fait ils n'en voyaient pas toujours la pertinence pour aider à résoudre les problèmes qu'ils se posaient, et souvent avec juste raison. On peut expliquer ceci par la difficulté de formaliser ces problèmes, et même si cette étape est franchie par la complexité d'exploitation des modèles obtenus. Actuellement, certaines de ces difficultés ont été suffisamment bien résolues pour que la modélisation soit devenue une préoccupation de beaucoup de biologistes, agronomes et médecins.

Plus précisément, deux points sont à souligner :

- le choix ou la construction d'un modèle dépend évidemment du système biologique, ou biotechnique en question, mais aussi et surtout des objectifs de la modélisation: s'agit-il d'analyser le système, de comprendre son fonctionnement (aspect souvent appelé cognitif), ou de prévoir son devenir (prévision), éventuellement en le simulant sous diverses hypothèses alternatives (analyse de scénarios), ou de contrôler son fonctionnement afin de l'optimiser, de l'automatiser (aspects normatifs), ou encore de chercher à estimer une variable inaccessible à la mesure directe (aspect instrumental), ou enfin de le décrire le plus simplement possible (aspects descriptifs: résumé de l'information expérimentale). Pour un même système, suivant la question posée, on n'aboutira pas nécessairement au même formalisme, on n'utilisera pas toujours les mêmes techniques. On est loin des modèles "à tout faire" de la physique du 19ème siècle;

- quel que soit l'objectif, mener à bien une tâche de modélisation demande de mettre en œuvre des méthodes et des techniques, et plus généralement des connaissances, de domaines très différents: connaissance de l'objet et de la problématique biologique, des méthodes de formalisation (étape souvent délicate), de développements formels et d'étude qualitative (analyse mathématique, recherche des propriétés de l'objet mathématique), de simulation pour compléter cette étude (informatique et analyse numérique), de gestion des ensembles de données, d'identification (problème d'optimisation, d'ajustement, de régression, dont les développements les plus significatifs sont l'objet de chapitres importants de la statistique ou de l'automatique), de validation (test statistique)... sans aborder les problèmes de retour à l'expérience guidée par le modèle (optimisation de protocole, qui peut être aussi un objectif en soi), et d'utilisation, voire de fabrication, d'outils ou de méthodes particulières de représentation graphique.

On remarque immédiatement que les deux points précédemment cités se décomposent en une multitude d'items faisant appel à des compétences variées dans des domaines très différents : de la biologie à des mathématiques quelquefois délicates..., réunir une telle somme de compétences est difficile pour un individu voire même au sein d'une équipe. Ce qui fait que

souvent les spécialistes de chaque discipline voient d'un œil sceptique, souvent sévère, une activité essentiellement interdisciplinaire plus axée sur la méthode que sur des développements pointus, mais inversement très génératrice de problèmes et ouverte sur la réalité. On comprendra également que former des modélisateurs est une tâche longue et délicate. Ce sont des experts aux sens de la largeur du champ et de la diversité des connaissances à acquérir et à maîtriser.

Depuis ses débuts l'informatique, les méthodes et disciplines annexes, ont été des aides importantes de la modélisation : calculs numériques, graphique, gestion des bases de données... Cependant beaucoup de ces outils sont dispersés, et malgré des efforts pour proposer des logiciels intégrés, leur utilisation demande, pour la plupart, une bonne connaissance des méthodes et de leurs performances ou tout simplement la façon de les mettre en œuvre, connaissances réservées aux spécialistes, c'est-à-dire aux **experts**. Enfin ces produits sont difficiles à modifier, à étendre, à maintenir: ils sont relativement figés.

Ainsi les approches classiques de l'informatique, si elles ont eu un rôle important voire essentiel, se sont heurtées à de multiples problèmes, les uns relevant du génie logiciel (concevoir un système intégré aisément modifiable), les autres de l'utilisation par des non spécialistes: dialogue avec le système, activation de traitements appropriés au problème posé, gestion du travail (carnet de laboratoire), gestion des données et des modèles... Il devenait donc nécessaire d'envisager des approches nouvelles notamment :

- celles relevant de l'intelligence artificielle, et plus particulièrement d'une de ses branches les plus dynamiques relative aux systèmes experts, ou plus généralement aux systèmes à base de connaissances,
- celles relatives au dialogue entre l'utilisateur et le système. Ce dialogue devant s'établir à travers des objets familiers à l'utilisateur, de façon que ce dernier consacre son temps à l'objet de sa réflexion, à sa question, plutôt qu'à résoudre des problèmes techniques dont il ne pourrait trouver la réponse que dans un manuel d'utilisation aussi épais et indigeste que le système a été voulu "performant".

Nous examinerons successivement ces deux aspects du problème, je prendrai comme référence un projet développé par l'INRIA, l'INRA et des UA du CNRS. Il s'agit du projet Edora (*), dont l'objectif était, dans un premier temps, de démontrer la faisabilité d'un tel système et, si oui, et dans un deuxième temps, de réaliser une version opérationnelle sous la forme d'un prototype. Aujourd'hui on peut considérer que la première étape a été franchie.

Systèmes Experts et Représentations des Connaissances.

La description et le stockage de grandes quantités de connaissances, l'aspect évolutif nécessaire des bases associées, conduisent en premier lieu à choisir un formalisme de représentation répondant aux contraintes énoncées, et en particulier adapté à la gestion de connaissances déclaratives et procédurales. Le formalisme informatique classique ne répond que partiellement à la question, il conduit surtout à la conception de programmes gros et complexes difficiles à mettre au point et à maintenir.

C'est en fait pour ces raisons de "génie logiciel" que les chercheurs en intelligence artificielle développent depuis quelques années des modes de représentation adaptés à la conception et à la réalisation de systèmes manipulant de grandes quantités de connaissances. Les plus connus de ces systèmes sont dits experts car leur ambition est de reproduire le raisonnement d'un expert spécialiste d'un domaine technologique ou scientifique donné (par exemple, le diagnostic en pathologie humaine, animale, ou végétale).

Actuellement le mode de représentation des connaissances le plus répandu est sans nul doute basé sur les **règles de production** (particulièrement prisé dans les systèmes de diagnostic), mais on parle aussi des **réseaux sémantiques** et surtout des **représentations centrées-objet**. Pour l'aide à la modélisation les règles de production ont été retenues dans un premier temps (de Swaans Arons, 1983), mais à présent les efforts des chercheurs se

portent principalement sur les représentations centrées-objet qui apparaissent mieux adaptées (Rechenmann, 1985[a], 1985[b], Pavé et Rechenmann, 1986, Zeigler et De Wael, 1986,...).

Les règles de production.

Rappelons brièvement que les connaissances sont représentées par des règles supposées indépendantes de la forme: si condition alors action (ou nouveau fait), ces règles sont stockées dans une base (base de règles, ou plus généralement base de connaissances). La partie condition contient une ou plusieurs prémisses portant sur des faits élémentaires dont la véracité est, soit connue *a priori* (hypothèse), soit inférée à l'aide d'autres règles dans la base de connaissances. L'action la plus fréquente, en partie droite de la règle, consiste à ajouter un fait à une base de travail du système dite base de faits. La partie "active" du système est un moteur d'inférence qui explore alternativement la base de connaissances et la base de faits, et modifie cette dernière en fonction des règles activées. Enfin il existe plusieurs modes d'exploitation (chaînage avant, chaînage arrière, fonctionnement mixte) que nous ne détaillerons pas ici.

Les avantages de cette représentation tournent autour de la notion de modularité: une règle n'incorpore qu'une petite quantité de connaissances, elle est en théorie indépendante des autres, son activation ne devrait dépendre que de la vérification de la partie prémisse et non de la position de cette règle dans la base. Cependant, dans la majorité des cas, ce n'est pas vrai: si plusieurs règles portent en prémisses les mêmes conditions alors il y a "conflit" alors ou bien la première règle rencontrée est activée ou bien le concepteur pécise l'ordre d'activation. Enfin les règles ne communiquent entre elles qu'à travers la base de faits. Ces propriétés rendent théoriquement le système facile à développer, à maintenir et à modifier.

Au passage, on remarquera l'avantage de ce type de programmation, dite déclarative, où le programmeur n'a pas à se soucier du contrôle du déroulement d'un programme, contrôle qui devient la principale source de difficulté dans les programmes écrit avec des langages classiques et qui incluent de grandes quantités de connaissances.

Pour l'aide à la modélisation, l'utilisation des règles de production a été principalement illustrée par de Swaans Arons (*op. cit.*). L'exemple qu'il a proposé concerne la définition du modèle mathématique d'un système physique élémentaire: une masse accrochée à un ressort. Il propose de tenir compte d'hypothèses plus ou moins simplificatrices sur ce système (existence ou non d'une force de friction, prise en compte ou non de la masse du ressort...). Il aboutit ainsi à formuler une vingtaine de règles de la forme :

Règle 13 : si le ressort ne satisfait pas à la loi de Hooke
le ressort est de masse nulle
il n'y a pas de force de friction
il n'y a pas de force extérieure
alors
le modèle est : $x'' + f(x) / M = 0$

dont douze portent en conclusion, comme dans cet exemple, la forme du modèle retenu.

Ces règles définissent en réalité une **hiérarchie** des modèles possibles du système masse-ressort: un modèle d'un niveau traduit des hypothèses communes aux modèles plus spécifiques (par exemple les modèles dérivés du précédent suivant la forme de f(x)). Or l'utilisation des règles, dont l'une des propriétés essentielles est l'indépendance entre elles, fait disparaître par dispersion cette idée de hiérarchie et plus généralement de relation entre les objets que sont les modèles (on trouvera en annexe un autre exemple: celui de modèles élémentaires de croissance en biologie).

En fait on voit rapidement que cette représentation des connaissances, fort bien adaptées à d'autres problèmes, convient assez mal aux besoins d'un système d'aide à la modélisation car:
- on ne peut pas représenter commodément les objets manipulés (modèles, équations, données),

- l'utilisation de méthodes de traitement (calcul numérique ou symbolique) doit être possible, le moteur d'inférence doit pouvoir les choisir suivant les renseignements dont il dispose puis les activer. Il doit donc exister une description explicite des méthodes disponibles: leurs conditions d'utilisation et le code informatique correspondant.

La solution consisterait à définir trois bases séparées (connaissances, objets et méthodes). Mais alors se poseraient d'importants problèmes quant à leur maintenance, en particulier au maintien de leur cohérence mutuelle.

C'est principalement pour ces raisons que dans le cadre du projet Edora une représentation centrée-objet a été retenue.

Représentations centrées-objet.

Ce type de représentation dérive des travaux de Minsky sur les "frames" (1975) et de recherches sur les réseaux sémantiques. L'article original de Minsky présente les concepts essentiels mais ne comporte aucune proposition pour une réalisation effective. Ceci explique les nombreuses interprétations actuelles de la notion de frame, dont la plupart ne reprennent d'ailleurs que les aspects les plus élémentaires. Pour le projet Edora une adaptation originale et particulièrement complète de ces notions a été développée par F. Rechenmann (Rechenmann 1985, Pavé et Rechenmann, *op. cit.*, Rousseau *et al.*,1986) :

- l'entité de description est appelée **schéma**. Un schéma décrit aussi bien une classe d'objets qu'un objet particulier, représentant d'une classe (le terme d'objet doit être pris dans son acception la plus large: ce peut être un objet physique ou conceptuel, une situation ou un contexte, une méthode algorithmique...). Un **schéma** est défini par un nom et la liste de ses **attributs**.
- Un **attribut** est défini par un nom et une liste de **facettes**. Cet attribut peut représenter une propriété ou un lien avec une autre classe.
- les **facettes** permettent de définir le type de l'attribut, c'est à dire le domaine général de ses **valeurs**, ou de décrire les moyens d'obtenir la valeur inconnue d'un attribut d'un représentant de la classe décrite par le schéma. Il est ainsi possible de définir une valeur fixe, de la déterminer en l'extrayant d'un objet dont une définition partielle est spécifiée, de calculer cette valeur en faisant appel à une procédure ou encore de retenir une valeur par défaut quand les moyens précédents ont échoué.

Ainsi, quand un schéma décrit une méthode algorithmique, ses attributs définissent les paramètres d'entrée et de sortie: leurs types, les conditions que leurs valeurs doivent vérifier pour que l'appel puisse avoir lieu, ainsi que les moyens de déterminer leurs valeurs si la forme d'appel ne les fournit pas.

Les schémas permettent donc la description, au sein d'une seule base, de classes d'objets manipulés dans un système d'aide à la modélisation ainsi que la description des méthodes algorithmiques. En effet, un modèle particulier y sera un représentant de la classe modèle. Ses attributs seront par exemple la forme mathématique, différentielle et intégrée, la position de ses asymptotes et la valeur de ses paramètres. Un autre attribut permettra de faire le lien avec des modèles voisins. Toute la connaissance attachée à ce modèle pourra être trouvée dans le schéma de sa classe. Ainsi on pourra chercher un modèle, ou un ensemble de modèles, satisfaisant une propriété, par un mécanisme de **filtrage** (par exemple rechercher dans l'ensemble des modèles de croissance ceux ayant une asymptote positive et un point d'inflexion dont l'ordonnée est inférieure à la moitié de celle de l'asymptote...), ou encore rechercher et activer la méthode disponible la mieux adaptée pour trouver la valeur d'un attribut (par exemple les valeurs numériques des paramètres par une méthode d'identification), il s'agit de **l'attachement procédural**.

Un objet particulier, un "spécimen", un individu, dérivé d'un schéma de classe (par exemple la réalisation particulière d'un modèle une fois connues les valeurs de ses paramètres) peut être conservé, on l'appelle **instance** du schéma de classe correspondant, le mécanisme de création de cet objet particulier est l'**instanciation**.

La conception de cette base est grandement facilitée par sa structure de treillis, sur laquelle un mécanisme d'**héritage** permet à un schéma de classe donné d'hériter des éléments de description des schémas plus généraux qui le dominent. Par l'héritage les modifications effectuées sur une classe donnée sont de fait propagées à toutes ses sous-classes. De plus, à partir d'une certaine taille, l'adjonction dans la base d'un nouveau schéma de classe se ramène à la définition d'une variante d'un ou de plusieurs schémas existants. Ce treillis peut être employé à des fins de **classification**. Un objet, créé dans une classe donnée, peut être placé convenablement dans les classes inférieures s'il satisfait les conditions qui y sont attachées.

Actuellement le moteur d'inférence (SHIRKA) travaillant sur cette représentation centrée-objet est opérationnel, il est écrit en LeLisp, divers utilitaires permettent une utilisation aisée (éditeur de schémas, mécanismes d'explication des inférences). C'est un produit INRIA.

Interaction homme/machine.

Les systèmes informatiques classiques ont permis de libérer l'homme de nombreuses tâches mécaniques. Cependant le dialogue entre le système d'exploitation, ou la plupart des logiciels d'applications scientifiques, et l'utilisateur reflète encore une certaine rigidité. Depuis seulement quelques années une réflexion et des réalisations montrent que le problème a été compris. Ainsi les concepts de l'interaction graphique tels qu'ils sont présentés sur certains postes de travail ou micro-ordinateurs peuvent constituer le point de départ d'une recherche sur la conception d'interfaces facilitant le dialogue.

Il s'agit de réduire au minimum le nombre d'intermédiaires entre les modèles mentaux de l'utilisateur et les entrées/sorties des programmes de façon que cet utilisateur soit plus concerné par le problème relevant de son domaine d'activité scientifique ou technologique que par la mise en oeuvre informatique (ainsi le manque de souplesse de la plupart des logiciels scientifiques fait que les utilisateurs ont perdu la flexibilité du mode de travail "papier - crayon"). Une interface-utilisateur doit s'appuyer sur des modèles conceptuels à partir desquels un environnement fictif est élaboré, environnement dans lequel l'utilisateur est capable d'agir à l'aide d'objets familiers. On peut citer l'environnement de bureau du MacIntosh, et ceux étudiés sur cette machine par de nombreux développeurs. Dans le secteur technologique les logiciels de CAO tenaient compte de ce problème d'interface, plus récemment des systèmes intégrant une expertise ont été étudiés et conçu avec des facilités de dialogue (Fjellheim, 1986, Fujiwara et Sakagushi, 1986). Plus généralement il faut retenir les efforts des concepteurs de systèmes experts qui essayent, entre autre, de proposer des interfaces en langues naturelles.

Ainsi il est clair que la réalisation d'un système d'aide à la modélisation passe par l'étude des moyens de dialogue appropriés. Par exemple, la représentation conceptuelle du fonctionnement d'un algorithme et le choix des représentations graphiques de ses résultats permettent de s'assurer de son "bon fonctionnement" et de la justesse de ce que perçoit l'utilisateur, ce qui constitue une forme d'intégration de l'expertise dans le logiciel. Dans l'optique du projet Edora, on peut citer deux réalisations qui préfigurent les interfaces du logiciel à venir: DYNAMAC et CROISSANCE (on trouvera en annexe une présentation succinte de ces programmes). Enfin un effort important est fait pour faciliter le choix, l'interprétation et la construction de modèles, en particulier par l'intégration de langages de description plus "parlants" et plus concis que le modèle mathématique associé (Pavé et Rechenmann, *op. cit.*, Pavé, 1986).

Conclusion

Les approches liées à l'intelligence artificielle, plus particulièrement aux notions de systèmes experts ou de systèmes à bases de connaissances (Knowledge Base Systems) vont sans nul doute conduire à une nouvelle génération de logiciels pour la modélisation . Il faut cependant se garder d'un optimisme béat: si on a démontré la faisabilité de tels systèmes le chemin à parcourir est encore long, il faut maîtriser ces nouveaux outils quand ils existent, constituer des bases de connaissances, concevoir des interfaces efficaces. Inversement, il est à présent certain

que ces nouvelles approches sont concevables et d'un grand avenir, il s'agit d'une nouvelle façon de faire de l'informatique conduisant à envisager des applications qui le sont difficilement par des approches plus classiques. On peut également attendre que la nécessité de constitution de bases de connaissance et de leur structuration aura une influence dans le champ d'application lui-même. Sous ces deux points de vue, le cas de l'aide à la modélisation des systèmes biologiques et biotechniques envisagé dans le projet Edora est un bon exemple.

(*) **Edora**: Equations Différentielles Ordinaires et Récurrentes Appliquées. On devine que ce projet est focalisé sur les modèles déterministes en biologie. Le choix de ce sigle, qui ne résume en fait que l'aspect mathématique du projet, vient en fait d'une proposition initiale de C. Lobry d'axer en premier lieu la réflexion sur les aspects mathématiques (c'est normal, il met en avant sa discipline!), le sigle etait alors Edoa (pas de récurrence). Comme le domaine d'application envisagé en premier lieu était la biologie j'ai alors proposé d'ajouter "Récurrentes" ce qui permettait d'introduire une classe importante de modèles en biologie (la biologie c'est plutôt mon affaire), en outre le sigle me semblait sonner plus agréablement à l'oreille. Cette proposition a été retenue par les acteurs du projet, en premier lieu par C. Lobry et P. Bernhard qui en furent les initiateurs.

En fait, ce sigle recouvre actuellement deux choses: d'une part le Club Edora (Club INRIA) et d'autre part le projet lui-même propre à cet Institut. Cette aventure, tout à fait risquée au départ, ne se terminera sans doute pas dans une impasse: la faisabilité d'un tel système peut à présent être considérée comme démontrée (une première maquette est opérationnelle depuis le 1/07/86, une version améliorée sera présentée en Juillet 87), en outre les retombées de ce projet, scientifiques et techniques sont d'ores et déjà importantes (c'est d'ailleurs peut-être là la première qualité d'un "bon" projet), citons notamment le moteur d'inférence travaillant sur les schémas développé par F. Rechenmann, et les divers travaux et réflexions menés par les autres membres du Club et du Projet en particulier sur la modélisation et les modèles en Biologie (notamment en biologie des populations).

N.B. : Sous une forme voisine, ce texte a fait l'objet d'une communication au colloque "Modélisation et optimisation de processus biologiques" organisé en Septembre 1986 à PARIS, par MEDIMAT et l'APRIA.

Bibliographie

On trouvera des éléments de base, et des exemples de réalisations dans l'ouvrage:

AI applied to simulation, Edité par E.J.H Kerckhoffs, G.C. Vansteenkiste et B.P. Zeigler, Simulation Series Vol. 18, n° 1. *The Society for Computer Simulation Publications*, San Diego Californie, 1986.

d'où sont tirées les références suivantes :

Zeigler B.P. et De Wael L. - Towards a knowledge-based implementation of multifaceted modeling methodology. 42-51.

Pavé A. et Rechenmann F. - Computer aided modelling in biology : an artificial intelligence approach. 52-66.

Muetzelfeldt R., Bundy A., Uschold M., Robertson D. - ECO: an Intelligent Front End for Ecological Modelling. 67-70.

Fjellheim J. - A knowledge based interface to process simulation.97-102

Fujiwara R. et Sakaguchi T. - An expert system for power system planning. 174-177.

Autres références:

de Swaans Arons H. - Expert system in the simulation domain. *Mathematics and Computers in Simulation*, 25, 1983, 10-16.

Minsky M. - A Framework for Representing Knowledge". *In "The Psychology of Computer Vision"*, P.H. Winston Ed., McGrawHill, 1975.

Rechenmann F. - SHIRKA : mécanismes d'inférence sur une base centrée-objet. Actes du 5ème Congrès AFCET-ADI-INRIA *"Reconnaissance des formes et intelligence artificielle"*, Grenoble, 1985.

Rechenmann F. - Représentation des connaissances dans les logiciels de calcul scientifique. In *"Informatique et Calcul, Computers and Computing"*, Chenin P., DiCrescenzo C., Robert F. Masson et Wiley, 1986.

Rousseau B. et Rechenmann F. - Le projet Edora, vers un poste de travail informatique pour l'aide à la modélisation des systèmes dynamiques en biologie. *Les Cahiers d'Edora*, INRIA, 1987 (sous presse).

Pavé A. - Schémas fonctionnels et modélisation, étude de modèles de la dynamique des populations. *Actes du coll. Biométrie-Econométrie, Sophia Antipolis,* 1986 (sous presse).

Rousseau B., Pavé A., Rechenmann F. et Landau M. - Edora project : Artificial Intelligence Approach and Work Station Concept to aid Dynamic Modelling in Biology and Ecology. *Proceed. of the Summer Conference of the Society for Computer Simulation*, Reno Nevada, 1986.

Annexe

Dans cette partie sont illustrés certains concepts développés dans le texte. En premier lieu un exemple simplifié de base de connaissance (très incomplète...) est présenté sur des modèles de croissance utilisés en biologie; pour représenter la croissance d'organismes ou de populations les deux principaux formalismes sont présentés: règles de production et représentation centrée objet. En deuxième lieu des figures obtenues à partir des programmes DYNAMAC et CROISSANCE, illustrent les aspects interactions graphiques : ces deux programmes, outre leurs intérêts propres, permettent de tester les idées devant conduire à définir les principes essentiels sur lesquels se fonderont les relations système-utilisateur.

Représentation des connaissances

modèles de croissance

Ces modèles sont fréquemment utilisés en biologie pour décrire la croissance d'organismes ou de populations, outre la caractérisation par leur forme mathématique on peut leur associer une interprétation biologique fondée sur les conditions de la croissance, les bases de cette interprétation peuvent être trouvées dans **Pavé** et **Rechenmann** (1986) ou **Pavé** (1987). Ici on considère une situation très simplifiée où la croissance est supposée dépendre d'un facteur de croissance qui peut être limitant ou non (ce facteur peut être interprété en termes de ressources nutritives, de facteur de type catalytique comme une hormone de croissance pour un organisme, d'espace disponible pour les individus d'une population...etc). Ce facteur peut également être lié à un phénomène de saturation: la biomasse ayant des capacités limitées d'assimilation, la vitesse du processus n'est pas proportionnelle à la quantité ou à la concentration de ce facteur, mais tend vers une valeur maximale. Ainsi suivant les hypothèses qu'on peut faire sur le phénomène observé on est conduit au choix d'un modèle particulier (exponentiel, logistique, Monod, Gompertz, dans notre exemple).

forme différentielle générale : $x' = a(x)\,x$

$a(x) = r$ (constante) : modèle exponentiel

ce modèle représente la croissance d'une population dans un milieu non limitant

$a(x) = r\,(1 - x/K)$: modèle logistique

croissance limitée par un facteur , ce facteur est consommé par la biomasse et n'a pas d'effet

saturant sur la vitesse de croissance.

$a(x) = r (1 - \ln(x) / \ln(K))$: modèle de Gompertz

croissance limitée par un facteur, ce facteur est dégradé indépendamment de la biomasse, il n'a pas d'effet saturant sur la vitesse de croissance.

$a(x) = r (K - x) / (K + C - x)$: modèle de Monod

croissance limitée par un facteur (substrat), il est consommé par la biomasse, il a un effet saturant sur la vitesse de croissance.

Représentation sous forme de règles de production (analogue à l'exemple proposé par de Swaans Arons pour le modèle "ressort", cf. texte).

Type général : **si** condition **alors** conclusion (ou fait ou action).

si a(x) est constante **alors** modèle_exponentiel

si a(x) est $r (1 - x/K)$ **alors** modèle_logistique

si a(x) est $r (1 - \ln(x) / \ln(K))$ **alors** modèle_de_Gompertz

si a(x) est $r (K - x) / (K + C - x)$ **alors** modèle_de_Monod

si modèle_exponentiel **alors** solution : $x = x_0 e^{- r t}$

si modèle_logistique **alors** solution : $x = K / (1 + (K - x_0) / x_0 \ e^{- r t})$

si modèle_de_Gompertz **alors** solution : $x = K \exp (\ln (x_0 / K) \ e^{- r t})$

si modèle_exponentiel **alors** asymptote supérieure = nil

si modèle_logistique **alors** asymptote supérieure = K

si modèle_de_Gompertz **alors** asymptote supérieure = K

si modèle_de_Monod **alors** asymptote supérieure = K

si conditions non limitantes **alors** modèle_exponentiel

si facteur limitant
disparition du facteur proportionnelle à la biomasse
pas de saturation par excès du facteur **alors** modèle_logistique

si facteur limitant
disparition du facteur proportionnelle à la biomasse
saturation par excès du facteur **alors** modèle_logistique

si facteur limitant
disparition du facteur indépendante de la biomasse
disparition exponentielle_décroissante **alors** modèle_de_Gompertz

...

On remarquera que la notion de modèle est diffuse, dispersée dans la base, un regroupement des règles serait arbitraire et artificiel et ne rendrait compte qu'en partie des relations entre "objets".

Représentation centrée objet : modèles élémentaires de croissance et exemple d'utilisation du moteur d'inférence SHIRKA développé pour le projet Edora.

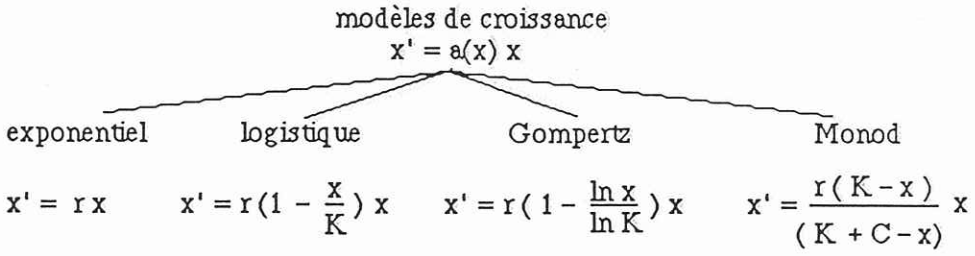
Type général :

L'élement de base de cette représentation est le schéma (proche de la notion de frame introduite par MINSKY (1975)), c'est une formalisation du concept d'objet qui peut se noter de la façon suivante:

```
(nom-de-schéma
    (attribut-1
        ($facette-11 ...
        ($facette-12 ...
        ...
        ($facette-1n ...))
    ...
    (attribut-p
        ($facette-11 ...
        ($facette-12 ...
        ...
        ($facette-1n ...)))
```

Les objets existent en tant que tels dans la base (comme les enregistrements logiques dans une base de données), leurs propriétés sont spécifiées par la liste des attributs, leurs valeurs ou la façon d'obtenir cette valeur sont précisées au niveau des facettes. Les objets peuvent être de natures fort différentes dans leur interprétation et leurs fonctionnalités, ici il s'agit de modèles, ce pourraient être des descriptions de méthodes algorithmiques, d'objets biologiques... Enfin, il est possible de spécifier des relations entre objets et de proposer ainsi une véritable classification. Pour les domaines de connaissances structurées (ou structurables), cette représentation semble actuellement la mieux adaptée (la structuration de la connaissance pour constituer une base peut d'ailleurs être un sujet de recherche en lui-même).

Très schématiquement on peut considérer que les modèles de croissances proposés dérivent d'un modèle plus général, au moins sur le plan formel, les relations avec cette forme peuvent être représentées dans une structure hiérarchique simple :



Les attributs communs à tous ces modèles , ainsi que leurs domaines de valeurs, pourront être spécifiés dans l'objet modèle de croissance, éventuellement des valeurs par défaut pourront être précisées. Ainsi on verra ci-dessous qu'on peut déclarer par exemple à ce niveau l'attribut exp_anal (i.e. expression analytique x = f(t)) et spécifier que par défaut on ne connait pas cette expression (c'est le cas, par exemple, pour le modèle de Monod).

exemple d'implantation

SHIRKA a été développé en Le_Lisp par F. Rechenmann, aussi l'exemple proposé emprunte la syntaxe correspondante (la fonction def-sh permet de construire les schémas). On montre la

possibilité de choix d'un objet, en l'occurrence un modèle de croissance, suivant certaines caractéristiques, ainsi que le processus d'héritage (i.e. comment un objet hérite, au moment de l'instanciation, des attributs d'un schéma de classe placé plus haut dans la hiérarchie). Enfin, pour des raisons qu'on peut voir dans un premier temps comme techniques, la représentation des connaissances est localisée sur trois niveaux (cf. figure-1). En effet, fondamentalement le processus de filtrage se déroule sur des instances de schémas de classe, ainsi si on veut filtrer sur ces schémas eux-mêmes (ici les modèles) on doit les considérer comme des instances de schémas placés à un plus haut niveau ce qui explique dans l'exemple ci-dessous la distinction entre méta-schémas et niveau objet (i.e. schémas de classe représentant la connaissance qu'on souhaite manipuler). L'exemple a été éxécuté (ce n'est pas un exemple de principe), la liste présentée a juste été modifiée pour des raisons de présentation. La machine utilisée est un Macintosh Plus d'Apple, l'implantation de LeLisp a été assurée par la société ACT Informatique.

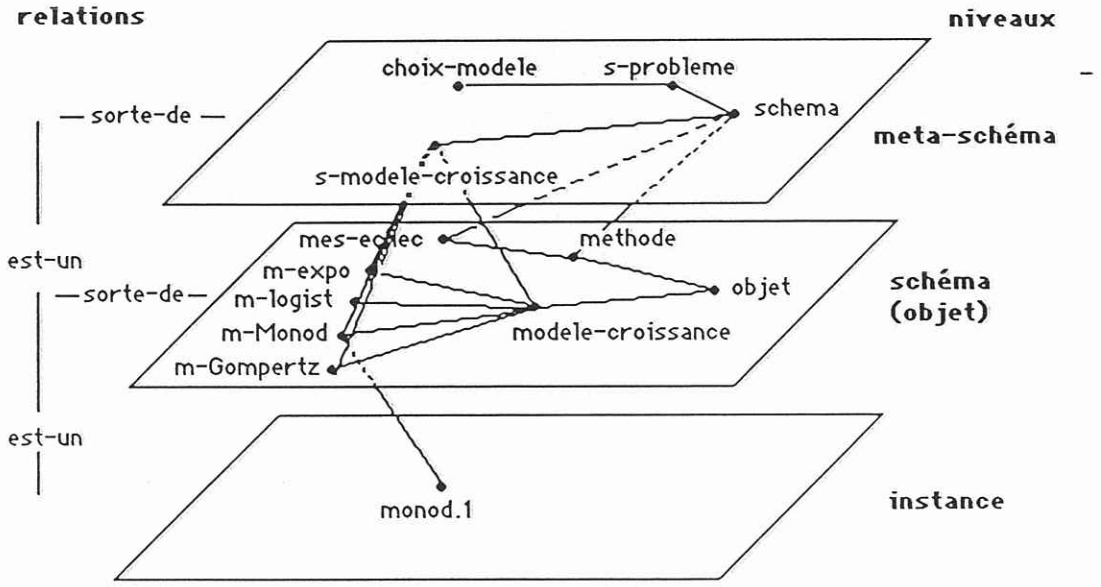


figure-1- Organisation de la connaissance dans SHIRKA, exemple de modèles élémentaires de croissance. La connaissance peut être organisée à trois niveaux, les schémas du niveau objet (schémas de classe) peuvent être considérés comme des instances de schémas écrits au niveau meta-schéma et donc manipulables comme des instances "ordinaires" (i.e. des réalisations de schémas de classes), en particulier pour le mécanisme de filtrage. Les relations verticales sont du type est-un alors que les relations horizontales sont du type sorte-de.

liste de l'exemple

```
;Edora: exemple de recherche d'un modèle suivant son interpretation
;(1)la croissance peut être (ou non) limitée par un facteur, si oui
;(2) ce facteur peut être consommé (ou non) par le système biologique,
;(3) l'excès de ce facteur peut être saturant (ou non) pour la vitesse de
;croissance.
;fact-cr est un attribut de type simple prenant les valeurs fff,vff...
;il permet de spécifier les conditions de croissance suivant les
;trois caractéristiques choisies (v: condition vérifiée, f: condition
;non vérifiée).Par exemple: pour le modèle exponentiel représentant
;une croissance non limitée par un facteur de croissance, l'attribut
;fact-expo prendra la valeur fff, pour les autres modèles on a:
;logistique: vvf
;Monod: vvv
;Gompertz: vff
;les modèles sont des spécialisations de modele-croissance.
;def-sh est une fonction SHIRKA permettant de définir les schémas
```

```
;-------niveau meta-schema

(def-sh '(s-probleme (sorte-de (= schema))))
(def-sh '(s-modele-croissance
            (sorte-de (= schema))
            (fact-cr ($un symbole))))
(def-sh '(choix-modele
            (sorte-de (= s-probleme))
            (fact-cr ($un symbole)
               ($domaine fff vff fvf ffv vvf vfv fvv vvv)
               ($var-nom v-fact))
            (nom-modele ($liste-de s-modele-croissance)
               ($sib-filtre (s-modele-croissance
                        (lui-meme ($var-> nom-modele))
                        (fact-cr ($var<- v-fact))))
               ($si-echec (mes-echec)))))

;-------niveau objet

(def-sh '(modele-croissance
            (sorte-de (= objet))
            (est-un (= s-modele-croissance))
            (remarque ($un chaine)
                ($valeur "ceci est un modele de croissance"))
            (exp_diff ($un symbole)
                ($defaut dx/dt=r{x}x))
            (exp_anal ($un symbole)
                ($defaut ¿-inconnue-?))
            (param ($liste-de reel))))
(def-sh '(m-expo
            (sorte-de (= modele-croissance))
            (est-un (= s-modele-croissance))
            (exp_diff ($valeur dx/dt=rx))
            (exp_anal ($valeur x=x0*exp{rt}))
            (fact-cr (= fff))))
(def-sh '(m-logist
            (sorte-de (= modele-croissance))
            (est-un (= s-modele-croissance))
            (exp_diff ($valeur dx/dt=r{1-x/K}x))
            (exp_anal ($valeur x=K/{1+{K-x0}/x0.exp{-rt}))
            (fact-cr (= vvf))))
(def-sh '(m-Monod
            (sorte-de (= modele-croissance))
            (est-un (= s-modele-croissance))
            (exp_diff ($valeur dx/dt=rx.{K-x}/{K+C-x}))
            (fact-cr (= vvv))))
(def-sh '(m-Gompertz
            (sorte-de (= modele-croissance))
            (est-un (= s-modele-croissance))
            (exp_diff ($valeur dx/dt=rx.ln{K/x}))
            (exp_anal ($valeur x=K.exp{ln{x0/K}.exp{-rt}}))
            (fact-cr (= vff))))

;definition de la fonction d'impression du message d'échec
;(attachement procedural)

(def-sh '(mes-echec
```

```
            (est-un (= schema))
            (sorte-de (= methode))
            (nom-fct ($valeur mes-echec))))

;fonction Lisp correspondante

(de mes-echec (inst)
            (let ((nom-inst (nom-sch inst)))
              (prin "je ne connais pas de modèle adapté à ces conditions..."))
            (terpri))

;---fin de l'exemple---
```

Les instances sont caractérisées par la présence de l'affectation "=" à la valeur des attributs valués (ils ne le sont pas tous forcément). L'apparition de ce type d'affectation au niveau objet signifie que l'attribut correspondant a été défini dans un méta-schéma et qu'il existe une relation est-un avec ce méta-schéma. Ainsi on voit, dans l'exemple proposé, qu'on définit la valeur de l'attribut fact-cr par ce moyen, puisque chacun des schémas de classe définissant les modèles est lié au méta-schéma modele-croissance contenant la définition de l'attribut fact-cr.

exemples d'interrogation et d'utilisation (figure -2)

(a)

Shirka: *cr-inst*
Classe: *choix-modele*
Nom de l'instance: *choix-1*
fact-cr ? *fff*
nom-modele ? -
->choix-1
Shirka: *val?*
instance? *choix-1*
attribut? *nom-modele*
-> m-expo

(b)

Shirka: *cr-inst*
Classe: *choix-modele*
Nom de l'instance: *choix-2*
fact-cr ? *ffv*
nom-modele ? -
->choix-2
Shirka: *val?*
instance? *choix-2*
attribut? *nom-modele*
je ne connais pas de modèle adapté
à ces conditions...
->echec

(c)

Shirka: *cr-inst*
Classe: *m-Monod*
Nom de l'instance: *monod-1*
exp-anal? -
param? *0.1*
param? *65.0*
param? *3.0*
param? -
->monod-1
Shirka: *vi monod-1*
monod-1
"ceci est un modèle de
croissance"
dx/dt = rx.{k-x}/{k+c-x}
¿-inconnue-?
(1.0 65.0 3.0)

figure-2- exemple de dialogue avec SHIRKA, les *réponses de l'utilisateur* sont en italiques. Ce dialogue correspond aux fonctionnalités de base de ce système, il est évidemment possible de programmer des interfaces plus sophistiquées, ce qui sort du cadre de l'exemple proposé.

(a) est un exemple de choix d'un modèle, l'attribut fact-cr prend le valeur "fff", l'utilisateur a la possibilité de forcer la valeur de nom-modele lors du dialogue. Si celle-ci n'est pas affectée (réponse "-"), alors SHIRKA déterminera la valeur de l'attribut par **filtrage** lorsque la question lui sera posée, ici il répond évidemment "m-expo";

(b) illustre l'utilisation de la facette "$si-echec". Cette facette utilise l'**attachement procédural** : le schéma mes-echec est une spécialisation du schéma de classe prédéfini "methode" lié à cette spécificité de SHIRKA, ici l'attchement procédural est simple, il s'agit d'activer la fonction Lisp d'impression du message: "je ne connais pas de modèle adapté à ces conditions...";

(c) montre le mécanisme de création d'une instance d'un schéma de classe (mécanisme d'**instanciation**), ainsi que les processus d'**héritage** et d'affectation de valeurs par **défaut**. Il s'agit de la création d'une instance du schéma de classe "m-Monod", du schéma "modele-croissance" il y a héritage d'attributs, de leurs domaines respectifs de validité et de la façon d'obtenir la valeur : affectée a priori (pour "remarque"), par défaut ("exp_anal") ou en posant la question à l'utilisateur ("exp_anal" ou "param"). D'autres voies sont évidemment possibles, comme on l'a vu précédemment, comme l'attachement procédural ou le filtrage.

Dialogue avec le système, autour du concept de poste de travail.

Pour une application donnée, les concepts actuels d'interface-homme machine peuvent être testés aisément sur des équipements de faible coût (par exemple le MacIntosh d'Apple), c'est ce qui a été fait dans le cadre du projet Edora, par B. Rousseau, avec deux réalisations: DYNAMAC et CROISSANCE. Plus généralement, on peut penser que les futurs logiciels scientifiques seront essentiellement utilisés sur des postes de travail (style SUN, APOLLO, GPX de DEC, SPS 7 de BULL...), machines autonomes offrant une puissance de calcul acceptable et de bonnes possibilités d'interaction graphique.

DYNAMAC

Ce programme permet l'étude graphique interactive des systèmes différentiels ou récurrents sur MacIntosh. L'utilisateur commence par décrire ses équations à l'aide d'un mini-langage type Pascal (cf figure-3-a). L'étude du système repose ensuite sur le tracé graphique de ses solutions dans un plan de l'espace de phase.

Le mécanisme de base consiste à choisir des conditions initiales en se positionnant dans le plan choisi à l'aide de la souris, puis à cliquer pour lancer l'intégration (figure-3-b). De nombreuses options complémentaires permettent le choix d'une méthode d'intégration, le tracé du champ de vecteurs et le tracé des isoclines (dans le cas de systèmes différentiels plans), la représentation des courbes d'évolution en fonction du temps (figure-3-c) ainsi que la localisation et la détermination des points d'équilibre. Enfin des fonctionnalités d'édition permettent une présentation des résultats et la créations de fichiers compatibles avec un logiciel de dessin (figure-3-d).

CROISSANCE

Ce dernier est conçu pour l'identification des modèles de croissance à une variable d'état. Outre les calculs classiques (estimation initiale, puis estimation par la méthode de Gauss-Marquardt), il permet une "identification à main levé" qui illustre bien la notion de travail "papier-crayon". Chacune des courbes de croissance est caractérisée par des points de contrôle dépendant des paramètres et dont les emplacements dans le plan variable-temps définissent entièrement la forme de la courbe. Par exemple, pour le modèle logistique, ces points de contrôle sont le point d'inflexion, l'ordonnée à l'origine et les asymptotes inférieure et supérieure (Figure 4). Pour ajuster la courbe sur les points expérimentaux, il suffit de déplacer l'un des points de contrôle à l'aide de la souris. La courbe se déforme alors, et la modification de forme est immédiatement répercutée sur les valeurs des paramètres. Cette fonctionnalité permet aussi d'examiner les répercussions d'une modification d'un de ces points de contrôle sur la forme de la courbe.

Plusieurs modèles peuvent être choisis (logistique, Gompertz, logistique généralisé, monomoléculaire...), enfin des informations peuvent être obtenues sur la qualité des estimations des paramètres et de l'ajustement.

Structure du système Edora.

La structure générale du système Edora donne une assez bonne vue de l'architecture des futurs systèmes de simulation ou d'aide à la modélisation (cf. figure 5). Il est organisé autour d'un noyau central: le moteur d'inférence SHIRKA travaillant sur une représentation des connaissances centrée-objet (cf. texte) et ses utilitaires : éditeur de schémas, module d'explication (non mentionnés sur la figure). Les différentes interfaces assurent la mise en forme, l'activation de procédures ad-hoc pour échanger des informations entre les différents éléments du système est sous le contrôle de SHIRKA.

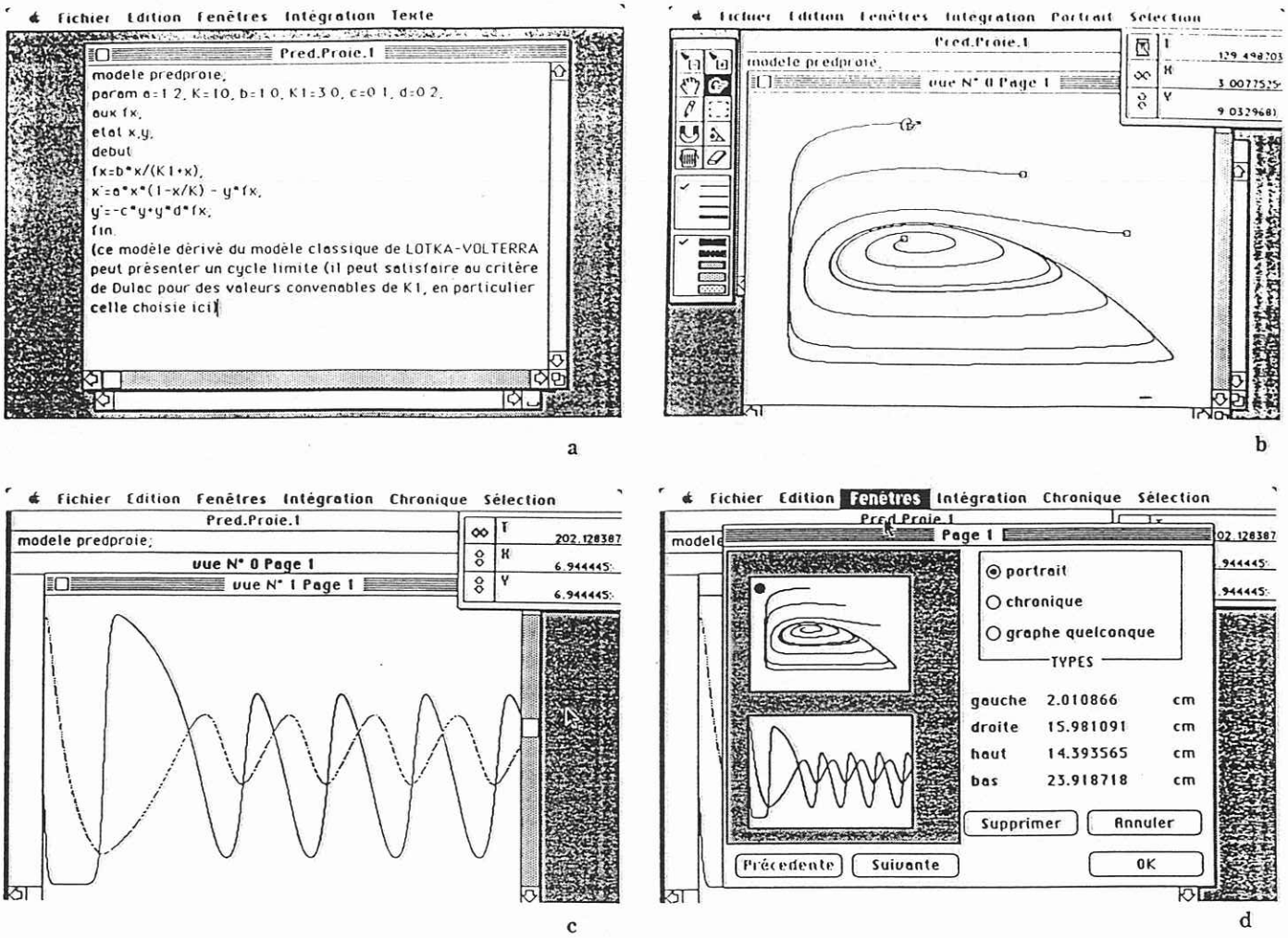


figure-3- Exemple d'utilisation de DYNAMAC pour l'étude d'un système différentiel.

a - édition du modèle utilisant un mini-langage d'entrée du type Pascal;

b - tracé de trajectoires dans le plan de phase, les conditions initiales sont repérées par des petits carrés, on voit nettement l'attracteur qui est un cycle limite;

c - tracé de la chronique : évolution des variables d'état en fonction du temps (x=f(t) en trait continu, y=g(t) en pointillés);

d - exemple de fonctionnalités d'édition des résultats.

Le système différentiel présenté est un modèle de relation prédateur-proie:

$$x' = a\,x\,(1 - x/K) - b\,x\,y\,/(K1 + x)$$
$$y' = -c\,y + d\,b\,x\,y\,/(K1 + x)$$

x (resp. y) représente une mesure de la taille de la population de proies (resp. y pour les prédateurs). La croissance de la population de proies est supposée logistique (terme: a x (1 – x/K)), cette proie est consommée par un prédateur, l'interaction entre les deux populations est modélisée par le terme : x y /(K1 + x), qui présente un phénomène de saturation (i.e. l'importance de ce terme est limitée, à y constant, par la quantité de proies disponibles : le prédateur a une capacité d'assimilation limitée), et cela contrairement au modèle classique de LOTKA-VOLTERRA qui ne fait pas une telle hypothèse (le terme correspondant est simplement le produit x.y). Enfin le prédateur est supposé être soumis à un processus de mortalité de type exponentiel (terme: –cy). Suivant la valeur de K1, le système admet un point fixe ou un cycle limite, notons que dans l'exemple proposé cette dernière solution, celle recherchée, a été trouvée en trois essais sans calculs préalables.

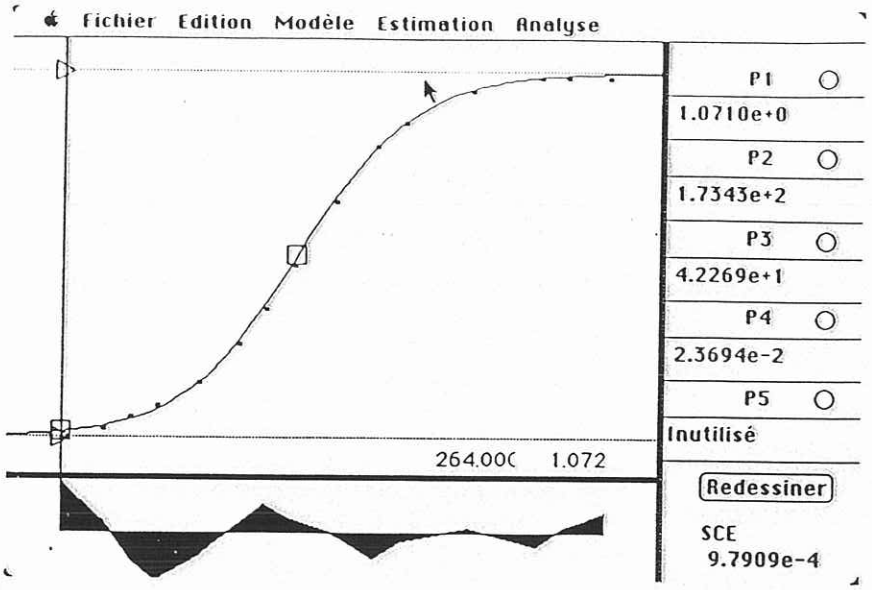


figure-4- Exemple d'utilisation du programme croissance : ajustement du modèle logistique à la croissance d'une population bactérienne (E. Coli en milieu complexe). Les estimations des paramètres sont donnés, la paramétrisation du modèle est la suivante : $y = p4 + p1 / [1 + \exp\{(p2 - p3)\, t / p3\}]$. La courbe peut être déformée en agissant sur les points de contrôle : ainsi on peut procéder à un ajustement "à vue", mais aussi, et surtout, on peut examiner rapidement la déformation de la courbe induite par des variations des points de contrôle (par exemple, la condition initiale).

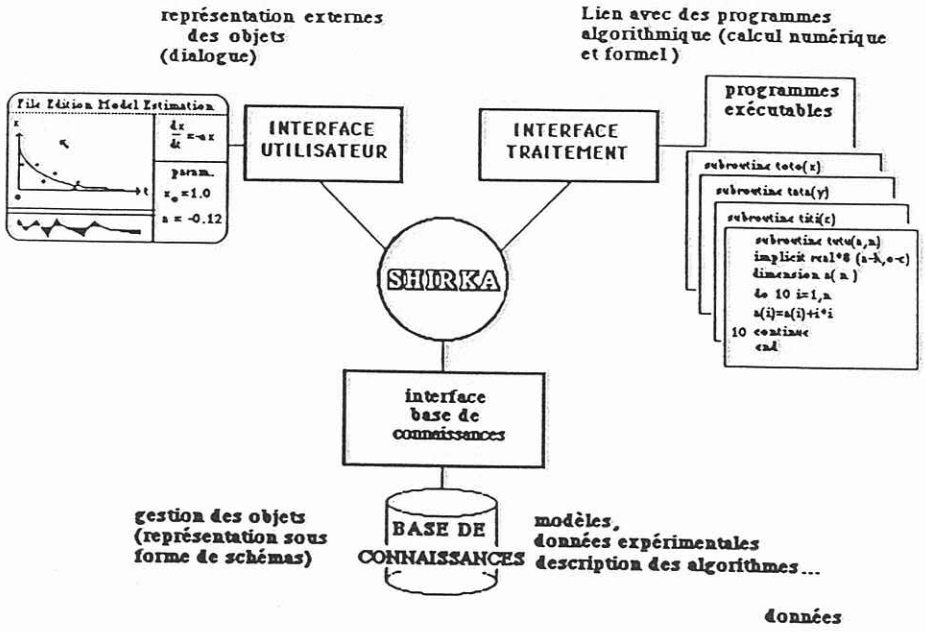


figure-5- Structure générale du système **Edora.** Cette structure préfigure sans doute les futures architectures des systèmes de simulation, d'aide à la modélisation et plus généralement de calcul scientifique: un moteur d'inférence (ici SHIRKA) pilote le fonctionnement du système en fonction du contenu de la base de connaissances associée.