7e Conférence Internationale de MOdélisation et SIMulation - MOSIM'08 - du 31 mars au 2 avril 2008 – Paris- France
« Modélisation, Optimisation et Simulation des Systèmes : Communication, Coopération et Coordination.».

AUTOUR DE LA CONFRONTATION DE MODELES

P.E. Portier

LIRIS, INSA, Université de Lyon
20, rue Albert Einstein
69621 Villeurbanne Cedex
Pierre-edouard.portier@insa-lyon.fr

G. Caplat

Département Informatique INSA de Lyon
20, rue Albert Einstein
69621 Villeurbanne Cedex
Guy.caplat@insa-lyon.fr

RESUME : *Nous cherchons à nous ouvrir un accès en direction de la problématique de confrontation de modèles. En demandant aux modèles de s'exprimer dans des langages différents, nous espérons offrir un tour original à notre démarche. Cela nous conduit en effet à questionner la notion même de modèle en l'ouvrant selon une synthèse, qu'il faudrait qualifier de disjonctive, de points de vue. Point de vue strictement formel sur le modèle en tant qu'il est exprimé dans un langage. Point de vue sur le modèle comme s'inscrivant toujours dans une situation de communication et revêtant ainsi toutes les caractéristiques du message. Point de vue sur le modèle comme objet d'étude d'un interprétant. Point de vue sur le modèle comme processus actif qui dessine les frontières de son sujet d'étude. Etc. Nous proposons un protocole de confrontation qui offre à l'interprétant de se positionner au cœur de la dynamique de cette multiplicité de points de vue sans avoir à se fixer définitivement sur l'un d'entres eux. Entre autres, nous trouvons dans l'analyse de la trace de construction d'un modèle le centre opérant de notre protocole.*

MOTS-CLES : *Modèle, langage, point de vue, sujet d'étude, confrontation, trace*

1. INTRODUCTION

Notre problème s'énonce simplement : confronter deux modèles. Cette introduction vise à motiver l'intérêt d'une recherche qui s'applique à résoudre ce problème. Elle s'attache aussi à faire de cette formulation simple une problématique à part entière en questionnant les termes de son énoncé. Nous ne pouvons faire ici l'économie d'une anticipation peut-être un peu brutale, mais dont l'absence nous semblerait apporter encore plus de désillusion : une fois notre travail terminé nous n'aurons accompli rien de plus qu'un éclaircissement de la question mais nous aurons le droit d'affirmer : ce qui paraissait simple était en fait complexe.

Supposons, pour fixer les idées, un premier modèle d'un processus quelconque, par exemple industriel. Il sera l'expression dans, disons, le langage des réseaux de Pétri du point de vue d'un premier modélisateur sur ce processus. Sa construction s'inscrit dans un contexte déterminé, qui appartient à l'histoire de l'entreprise, et avec des motivations particulières au modélisateur, au contexte, etc. Supposons ensuite un second modèle du même processus, mais qui s'inscrit dans un nouveau contexte, avec un nouveau modélisateur, de nouvelles motivations, un nouveau langage, par exemple un diagramme d'états UML, etc. Supposons enfin que soit engagé un projet de confrontation des deux modèles dont on espère tirer profit. Il apparaît rapidement qu'une comparaison purement formelle offrira certainement l'assurance d'un résultat mais occultera dans le même mouvement une multiplicité de dimensions qui ne peuvent qu'échapper à une réflexion algébrique. Par exemple, comment considérer des choix de modélisation qui proviennent indirectement de contraintes imposées par le langage d'expression du premier modèle, mais qui n'ont plus lieu d'être lorsqu'il s'agit du second modèle ? La question sera de savoir comment opérationnaliser un travail de confrontation qui ne se réduirait pas à des considérations seulement formelles.

Pour un informaticien, la définition canonique du modèle peut s'énoncer en ces termes : le modèle est l'expression dans un langage (code, formalisme,..) d'un(e) point de vue (théorie) à propos d'un sujet d'étude.

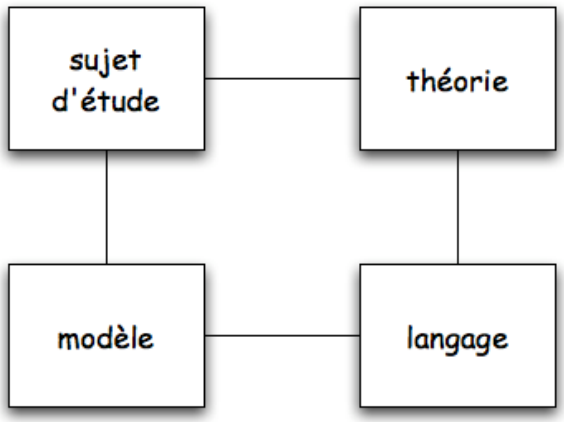

Figure 1. (méta)modèle ?

Se poser la question de la confrontation de deux modèles exprimés dans des langages différents revient ainsi à couper dans le champ exploratoire d'un domaine plus vaste qui s'intitulerait : « la confrontation de modèles ». Cette coupure suit deux axes : elle réduit d'abord le nombre de modèles à deux, elle impose ensuite une variation des langages. Elle maintient cependant un grand nombre de degrés de liberté, n'évoquant la nature ni des acteurs ni du sujet d'étude. Dans la suite, nous tenterons de remplir ces blancs, mais ce faisant nous nous rendrons compte des limites d'une démarche définitionnelle lexicographique (modèle := … ; langage := … ; acteur := … ; etc.) et nous devrons mener un questionnement sur la signification, le signe, le symbole, etc. Nous serons ainsi portés à mettre l'accent sur ce qui dans un modèle ne peut se réduire à un point de vue purement formel. Pourquoi se limiter d'entrée de jeu à la confrontation de deux modèles exprimés dans des langages distincts ? Premièrement parce que le problème de la confrontation de modèles exprimés dans le même langage a déjà fait l'objet de beaucoup de réflexion, de recherches. … Ensuite, mais il est délicat de l'exposer, sous une forme a priori, dans une introduction, parce qu'imposer des modèles exprimés dans des langages différents commandera un questionnement au tour peu commun ; or, comme nous l'apprenait Shannon (Shannon C.E., 1948) , plus l'événement est rare, plus il véhicule d'information !

Nous commencerons par une ouverture de la définition nominale du modèle. Puis nous montrerons pourquoi nous répondons uniquement à une partie de la problématique initiale et comment cette restriction peut se justifier. Nous continuerons par questionner le concept de signe puis nous accompagnerons cette réflexion du développement d'une métaphore qui devrait éclairer notre propos en le répétant sur un autre plan sémantique. Nous exposerons comment nous croyons être parvenus à dégager les moyens d'une opérationnalisation des conséquences des idées que nous aurons développées. Nous détaillerons un protocole qui fixe cette tentative d'opérationnalisation. Nous évoquerons une réalisation logicielle qui implémente le protocole. Nous jouerons de la proximité de notre problématique avec celles des modèles de "undo/redo". Nous proposerons l'idée d'un moment de l'informatique et de sa communication avec d'autres domaines. Enfin nous listerons quelques idées d'applications qui nous furent inspirées par ce travail.

2. OUVERTURE DE LA DÉFINITION DE MODÈLE

Il s'agira, à partir de la définition nominale de modèle qui était rappelée dans l'introduction (soit l'expression par un acteur, dans un langage, d'un point de vue sur un sujet d'étude) d'extraire tout ce que l'on peut trouver d'effectif. La démarche consiste à proposer plusieurs points de vue sur la notion de modèle. Remarquons que ces points de vue en tant que nous les exprimons dans un langage – en l'occurrence un langage naturel, le Français, pour qu'ils soient reçus par d'éventuels lecteurs –, sont à leur tour des modèles.

Un modèle en tant qu'il est l'expression d'un point de vue dans un langage répond à toutes les caractéristiques d'une expression langagière. La partie nominale « langage » de la définition de modèle est en fait très effective, surtout pour un informaticien qui possède en général un bagage conséquent sur la notion de langage. Ce premier point de vue couvre une large partie de l'aspect purement formel de la notion de modèle. Nous pouvons citer dans ce contexte la majorité des travaux de l'ingénierie dirigée par les modèles (transformations de modèles, évolution des modèles, intégration de la dimension temporelle, …) (Mukerji J. et Miller J., 2001). Mêmes des travaux qui pouvaient, dans un premier temps, nous faire penser adopter un point de vue qui ne soit pas seulement formel s'inscrivaient en fait encore dans des jeux de pure forme (Abd-Ali, J. et K. El Guemhioui, 2005 ; Deissenboeck F. et Ratiu D., 2006 ; Madhavan J., and al, 2002).Nous sommes au niveau du symbole, des algèbres, ... L'avantage c'est la possibilité de concevoir les diverses manières d'opérer sur cette forme. L'inconvénient c'est la réduction du sens dans une conception absolument duale signifiant - signifié. C'est l'origine du symbole dont le sens propre provient de la Grèce antique où il était un tesson de poterie cassé aux morceaux partagés entre deux contractants. C'est donc l'univocité du couple signifiant - signifié.

Modifions notre point de vue et considérons le modèle en tant qu'il met nécessairement en jeu des acteurs. Il faut distinguer les différents rôles que peuvent tenir ces acteurs. Une différentiation immédiate sera de classer ces derniers selon qu'ils reçoivent ou émettent le modèle. Nous remarquons que pour ce point de vue qui met l'accent sur les acteurs dans la définition nominale, le modèle se comporte comme un message. Voilà qui devrait résonner aux oreilles de l'informaticien ! lui qui connaît les messages, la quantité d'information, les codages, la transmission, les réseaux, etc. Mais cela ne va pas sans dégager de nouvelles difficultés, jusqu'alors effacées. Entre autres mais surtout, le modèle comme message apparaît au sein d'un contexte de communication. Le sens du message devient plus complexe et la position stricte signifiant ~ signifié délicate à tenir. En effet c'est là le domaine de la pragmatique, mais qui reste malgré tout modélisable par un ensemble de règles adjointes à celles du langage. Si nous disons que la relation signifiant ~ signifié, c'est-à-dire l'univocité du symbole et du sens, s'en trouve mis à mal, c'est qu'entre autre le non-respect des règles peut devenir porteur de sens (reste à savoir, s'il est le résultat d'une action volontaire ou simplement une erreur).

Considérons encore le point de vue obtenu lorsque l'accent est mis sur la notion de sujet d'étude dans la définition nominale de modèle. Il est alors nécessaire de distinguer deux catégories de modèles. Les modèles

descriptifs qui sont l'expression d'un point de vue sur un sujet d'étude qui leur préexiste et les modèles dont l'objet est justement de définir leur sujet d'étude (modèle de spécification, etc.). Mais ces deux catégories de modèles partagent une notion opératoire commune : le dessin d'une frontière. Pour la première catégorie, il s'agira de découper les limites d'une structure, de sélectionner les éléments pertinents et donc de distinguer le sujet d'étude de son environnement. Pour la seconde, il s'agira de spécifier les contraintes pesant sur un système à construire, le 'reste' constituant le monde ouvert du libre et du non-dit. La notion de frontière nous semble ici essentielle, pas du tout évidente. Quels sont les éléments du sujet d'étude qui définissent la frontière avec l'environnement ? C'est une question délicate pour laquelle nous allons exposer immédiatement une réponse qui pourra sembler étonnante mais que le reste du texte devrait justifier. Il n'y a pas d'éléments du sujet d'étude qui ne soient frontaliers. Une conséquence directe : les dichotomies binaires du genre intérieur – extérieur ne pourront plus jouer efficacement.

Nous avons essayé d'ouvrir la définition nominale du modèle par le jeu de plusieurs points de vue. Nous avons fait correspondre à chacun de ces divers points de vue une partition différente dans la stricte linéarité de leur succession. Il faudrait imaginer ce que pourrait donner une composition contrapuntique de tous les points de vue …

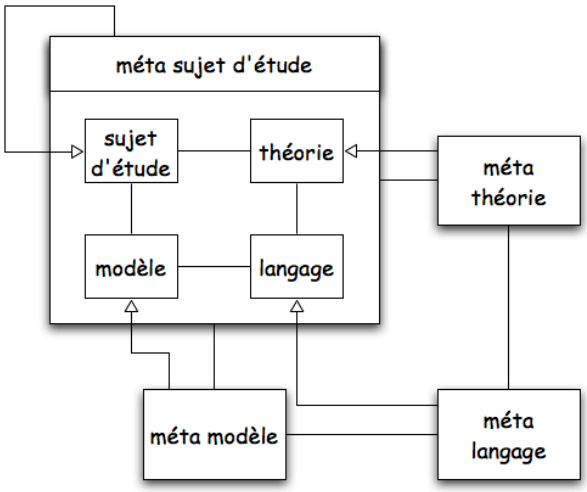

Figure 2. méta((méta)modèle) !

3. UNE SIMPLIFICATION

Rappelons que notre problème initial était la comparaison de modèles lorsqu'ils sont exprimés dans des langages différents. Mais maintenant, de quel type de différence parle-t-on ? Autrement dit, combien les langages peuvent-ils différer pour que la comparaison des modèles conserve malgré tout du sens ? Nous pensons que la possibilité de modéliser les langages L1 et L2 (par exemple Java, C++) dans un même métalangage est une condition suffisante pour pouvoir légitimement comparer des modèles exprimés dans ces langages. Nous pouvons imaginer confronter deux partitions musicales écrites dans des systèmes de notation différents, mais confronter une partition et un programme informatique n'a pas beaucoup de sens et ne nous semble pas relever de notre problématique. Malgré tout, cela reste une intuition, que nous pensons en tous points raisonnable, mais que nous ne saurions mieux justifier. Cette première constatation nous permet d'enchaîner sur une seconde : pour comparer deux modèles exprimés dans des langages différents il est nécessaire de comparer des modèles de ces langages. Seulement, pour ne pas boucler infiniment, il faut supposer pouvoir exprimer les modèles des langages dans un même langage. Nous nous occuperons par la suite uniquement de cette dernière comparaison. Cela peut sembler un grand pas en arrière, mais c'est en vérité un détour qui nous offre une perspective nouvelle sur la confrontation de modèles, refusant de considérer seulement le modèle comme pure forme mais en l'appréhendant également comme un instrument, un métaoutil de communication (dans la mesure où un modèle est un outil de communication, le métamodèle d'un langage d'expression de modèles est un métaoutil !)

4. OUVERTURE DE LA NOTION DE SIGNE

Nous expliquions plus tôt qu'adopter un point de vue très formel sur le modèle, c'est-à-dire mettre l'accent sur le modèle en tant qu'expression d'un point de vue dans un langage, c'est défendre une conception duale symbole ~ sens. Nous montrions aussi comment il pourrait être utile d'assouplir cette univocité au risque de rendre toujours plus délicates les actions sur le plan opérationnel. Nous détaillons maintenant un moyen d'actualiser cet assouplissement. Il s'agit de questionner le concept de signe et de montrer comment il a pu subir un bouleversement important.

Le premier concept du signe est celui de l'univocité d'un signifiant et d'un signifié. C'est le signe tel qu'il est développé par Ferdinand de Saussure (De Saussure F., 1913). Nous citons son nom parce que cela nous semble nécessaire. Mais nous ne voudrions en aucun cas lui associer de force un concept que nous présentons d'une façon beaucoup trop simpliste. Ce que nous voulons faire ici c'est exagérer deux points de vue sur le concept de signe. Cette exagération nous interdit d'être justes avec les signataires de ces concepts. Peut-être ferions-nous mieux alors de taire leurs noms ? Sans doute pas car il est aussi essentiel de noter comment l'évolution d'un concept dans des domaines étrangers à l'informatique peut avantageusement nourrir une réflexion « en informatique », et l'évocation de quelques noms propres marque (dans le temps, dans l'espace, dans le découpage régional des « sciences », …) et situe notre propre discours.

Le second concept du signe est celui de chaîne de renvois signifiants. Il est signé Charles S. Peirce, mais aussi Jacques Derrida (Derrida J., 1967), Umberto Eco (Eco U., 1990), Gilles Deleuze (Deleuze G., 1980 ; Deleuze G., 1988), … qui sont autant d'acteurs de son

évolution, de ses transformations. Il s'agit principalement de refuser la notion de signifié. Il n'y a plus de signifié mais seulement du signifiant qui ne cesse de renvoyer à du signifiant. Il est alors impossible d'accepter l'univocité et l'automatisme de la production du sens. Un processus apparaît : l'interprétation, et avec lui, un nouvel acteur ou personnage conceptuel : l'interprétant. C'est assez étonnant car nous employons facilement et couramment ces deux notions d'interprétation et d'interprétant mais nous ne les identifions pas souvent à un concept du signe qui a supprimé le signifié. Or il nous semble que ces trois concepts sont en fait inséparables.

Pour conclure ce paragraphe, nous aimerions réduire un peu l'exagération que nous avons fait subir à ces deux concepts du signe pour mieux les opposer, en rappelant qu'une idée fondamentale de de Saussure était de concevoir le langage comme un système clos de signes. Ceci impliquait pour tout signe d'être défini par rapport aux autres, par pure différence. Nous voyons bien que l'opposition des deux concepts du signe n'est pas si manichéenne ...

5. UNE MÉTAPHORE

Pour proposer un nouveau point de vue sur ce concept étonnant du signe comme chaîne de renvois signifiants, nous développons maintenant une métaphore de la signification, en l'occurrence, une métaphore « chimique » de la signification : la signification serait le précipité apparaissant quelque fois au fond du tube à essai dans lequel auraient été dissous des éléments de sens. Notons que si c'est bien un nouveau point de vue, la métaphore n'en est pas pour autant un modèle puisqu'elle consiste en une substitution du sujet d'étude pour mieux montrer une similarité structurelle.

Le modèle-message est une solution de liaisons signifiantes. Le sens est un agencement de ces liaisons obtenu par précipitation. L'intérêt de cette métaphore est de montrer les différentes modalités de la précipitation. Certaines précipitations peuvent être brutales, absolument certaines et leur résultat toujours assuré identique, pensons à la réaction acide base qui produit systématiquement un sel et de l'eau. C'est dans cet esprit que les informaticiens conçoivent une ontologie de domaine en tant que système de références communes et recueil d'agréments à partager. D'autres précipitations peuvent être plus incertaines aussi bien dans leur déclenchement que dans leurs résultats. C'est le cas général, lorsque la structure n'est pas contrainte au point d'imposer une univocité du sens. Enfin, la précipitation peut n'avoir jamais lieu. Ce sera par exemple la situation de l'interprétant qui réceptionne un modèle exprimé dans un langage qui lui est inconnu.

6. VERS UNE SOLUTION OPÉRATIONNELLE

Quelle serait une solution opérationnelle à la comparaison de modèles lorsque l'on se place dans cette conception plurivoque, étendue, ouverte du modèle ? Une réflexion de l'utilisateur sur la trace de son propre travail de comparaison nous semble une voie envisageable. Mais nous allons maintenant retracer en quelques mots l'historique de cette idée pour tenter de la légitimer (tout en sachant qu'elle devra ensuite faire ses preuves en pratique).

Tout débute par une réflexion sur les façons de comparer deux modèles. Nous décidons d'adopter un point de vue qui est un peu original : comparer deux modèles c'est isoler ce qui des deux modèles diffère, c'est isoler les différences. Nous sommes amenés à étudier un certain nombre de travaux parmi lesquels nous auront surtout influencés ceux de Gregory Bateson (Bateson, G., 1979) qui est l'auteur de la formule célèbre : « l'information est une différence qui fait une différence ». Une métaphore pour illustrer cette formule serait un gros point de craie sur un tableau noir, si l'on pose le doigt sur la trace, on ne détecte pas le point, mais si on met notre doigt en mouvement nous détecterons les contours de la trace. Ou bien encore une voiture qui bondit sur un dos d'âne, la différence de niveau nécessite l'énergie apportée par la voiture pour devenir effective.

Une fois la différence isolée, nous devons la mettre en mouvement, et c'est alors que nous pensons à proposer un travail de l'interprétant sur la trace de sa propre activité de modélisation.

7. PROPOSITION D'UN PROTOCOLE

Pour isoler la différence entre les deux modèles à comparer nous proposons simplement de factoriser ce qui est identique, c'est-à-dire de factoriser le même. Cette factorisation peut s'effectuer par la construction d'un troisième modèle. Le but de cette construction est d'obtenir un modèle qui exprime tout ce que les deux modèles originaux exprimaient en factorisant le plus possible ce qui est identique entre les deux modèles. Ce but ne sera pas souvent réalisé ni réalisable, soit que l'interprétant ne parvienne pas à factoriser, soit que les deux modèles demeurent intrinsèquement irréductibles. Mais ce n'est pas très important, ce qui compte est la tentative de factorisation.

La trace de construction du modèle tiers est ensuite présentée à l'utilisateur qui a la possibilité de travailler dessus en lui appliquant des patterns qu'il aura construits ou retirés d'une bibliothèque.

C'est l'analyse de ce travail sur la trace qui, en mettant en mouvement la différence, sera la réalisation effective de la comparaison des deux modèles.

Nous remarquons a posteriori que les deux étapes du protocole sont suffisamment indépendantes pour que nous puissions considérer la seconde (l'analyse du

travail sur la trace de construction d'un modèle) comme une solution éventuelle à l'opérationnalisation du concept très ouvert de modèle que nous développions plus haut.

8. PROTOTYPE LOGICIEL

Nous avons implémenté un dispositif qui réfléchit les étapes de notre protocole.

L'interprétant/confrontateur visualise les deux modèles à comparer. Il en construit un troisième avec l'intention de factoriser ce qui est identique aux deux modèles. Cette première étape est très peu contrainte, l'interprétant peut « faire des erreurs », quoiqu'il en soit le modèle construit risque d'être incohérent. Ce ne devra pas être considéré comme une faute. Seul le mouvement de l'interprétant dans sa tentative de factorisation importe.

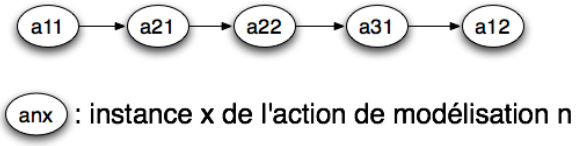

Figure 3. trace linéaire

L'interprétant décide de « terminer » sa comparaison et il est alors confronté à une représentation du modèle de la trace de son activité de modélisation – factorisation (figure 3). Il peut exécuter la trace pour rejouer toutes ses actions de modélisation. Il peut décider d'ouvrir la trace en son cœur et d'entamer de nouvelles actions de modélisations qui seront à leur tour enregistrées et viendront compléter la trace initiale en lui ajoutant « une branche » (figure 4). La trace qui était linéaire prend maintenant une structure d'arbre. Il peut aussi factoriser au sein de la trace une série d'actions de modélisations (car il aura par exemple remarqué qu'elles se répètent régulièrement) en les modélisant par un pattern. Le pattern deviendra lui-même un objet exécutable de la trace.

Nous allons montrer en quoi notre modèle de la trace n'est pas sans rapport avec les modèles de undo-redo.

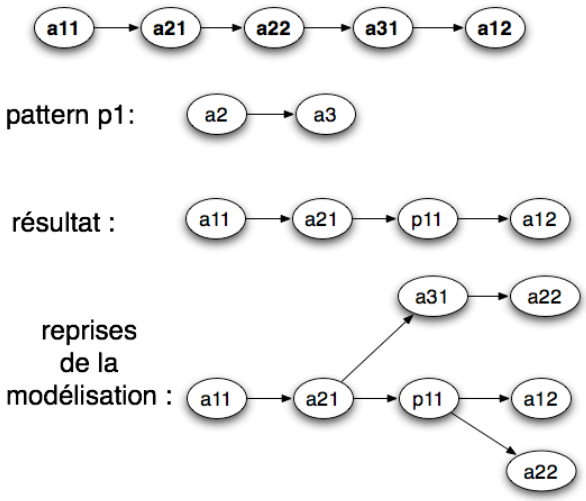

Figure 4. trace arborescente

9. PROXIMITÉ DE LA PROBLÉMATIQUE DU UNDO-REDO

Nous identifions une proximité thématique entre notre proposition de travail de l'interprétant sur la trace de ses actions de modélisation et la problématique du undo-redo. Que nous ayons joué de cette proximité pour questionner, éprouver notre démarche, c'est établi, et nous montrerons ici en quoi ce changement de perspective était riche d'enseignements. Qu'à notre tour, nous offrions une perspective neuve depuis laquelle (ré)entreprendre, (re)questionner la problématique du undo-redo, nous aurions tendance à le croire, et nous essaierons de le montrer.

La problématique du undo-redo est étudiée depuis de nombreuses années. Il s'agissait d'abord d'identifier le besoin. Il fallait ensuite proposer des solutions pragmatiques pour y répondre. Dans le même temps, on cherchait à modéliser le problème pour définir ses frontières et l'extraire de contextes toujours particuliers (traitements de textes, éditeurs graphiques, ...) (Mancini R., et al, 1996). Beaucoup de modèles existent, mais on convient en général de les classer en deux grandes catégories selon qu'ils sont ou non linéaires. Dans un modèle linéaire, la fonction du undo est d'annuler l'effet de l'action précédente. Cette première perspective est extrêmement riche et ouvre sur de nombreux champs exploratoires.

Demandons-nous, par exemple, si annuler l'effet d'une action d'annulation appartient à la fonction de l'undo ? Autrement dit, qu'en est-il de la réflexivité du undo ? Il s'agit de déterminer s'il convient de dissocier des catégories d'actions : celles qui sont spécifiques à un domaine (simulation de construction de structures tubulaires, traitement d'image, ...), celles qui sont réflexives, c'est-à-dire qui ont pour domaine, des actions. La tendance sera de ne pas rendre cette dissociation explicite pour l'utilisateur, même si elle existe structurellement pour les besoins du logiciel.

Les modèles non linéaires de undo apparaissent le plus souvent dans des contextes multiutilisateurs mais dans tous les cas, ils ajoutent des fonctionnalités qui profiteront aussi aux contextes mono-utilisateur. Il s'agit de rompre avec la linéarité des précédents modèles en permettant à l'utilisateur d'annuler une action même lorsqu'elle n'est pas la dernière effectuée. Dans un contexte multiutilisateur, quand un espace de modélisation est partagé, vouloir annuler *sa* dernière action n'est peut-être plus annuler *la* dernière action. Souvent, l'idée est celle du "selective undo" (Berlage, T. 1994): annuler l'effet de l'action b dans la série d'actions a1, a2, ..., an, b, c1, c2, ..., cn revient à exécuter la série d'actions a1, a2, ..., an, c1, c2, cn. Dans les faits, se posent les problèmes de dépendances entre l'action b et les actions ci. Il faut souvent modifier les ci pour que la série reste cohérente (conserve du sens, soit exécutable, ... la difficulté à fixer le vocabulaire reflète à la fois, la complexité de la problématique du undo-redo et les simplifications que nous lui imposons dans la nécessité que nous sommes de la tenir en retrait, au second plan, comme pour mettre en danger, déséquilibrer, provoquer des questions, etc.).

Rompre avec la linéarité du mécanisme de undo-redo demande de penser une représentation de la trace des actions de l'utilisateur, pour lui permettre de rapidement sélectionner l'action à annuler. Une liste textuelle des actions ne conviendrait pas. Il serait trop délicat de distinguer deux actions du même type (par exemple, "changer couleur", "redimensionner", ...). La solution proposée par Masuda H., et Imamiya, A. 2004 permet à l'utilisateur de visualiser l'effet de ses actions "image par image", comme dans un dessin animé, en manipulant une barre déroulante (voir la figure 5).

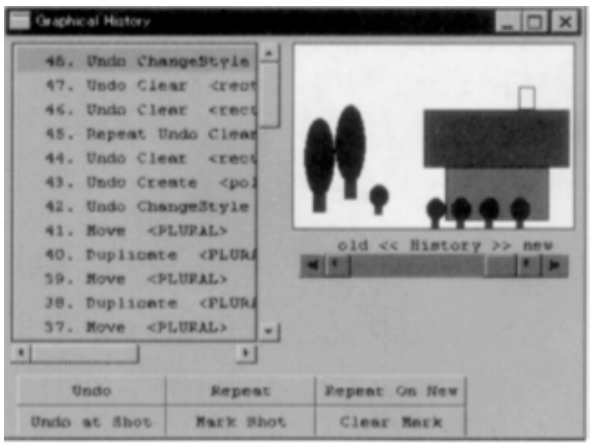

Figure 5. représentation de la trace des actions de l'utilisateur

Nous proposons d'associer à chaque type d'action un symbole. Ainsi, la série ordonnée des actions de l'utilisateur possède pour représentation une série ordonnée de symboles. Nous avons besoin de cette représentation pour permettre de factoriser des séries d'actions contiguës au sein de la trace.

Ce qu'il y a de commun entre la problématique du undo/redo et notre proposition d'une opérationnalisation de la confrontation de modèles quand on ne réduit pas ce dernier concept au seul point de vue formel, c'est principalement la nécessité de représenter la trace de l'activité de l'utilisateur et d'autoriser certaines opérations sur cette trace. Ce qui distingue les deux problématiques, c'est la manière dont joue pour chacune le couple formel / formalisable.
Dans le cas du undo / redo, on suppose que l'utilisateur possède en mémoire une action qu'il veut annuler. Le processus est ensuite très formel, il s'agit de proposer à l'utilisateur un moyen de sélectionner l'action, puis d'effectuer une suite d'opérations "algébriques" qui annulent l'effet de l'action sélectionnée. La manière dont est construite cette suite d'opérations relève d'un travail du programmateur en amont, travail qui demande de l'intelligence, de l'imagination, ... et qui n'est assurément pas une activité formelle, bien que le résultat de cette activité se doive d'être formalisable.
Dans le cas de notre proposition, l'activité de factorisation de l'interprétant sur sa propre trace n'est certainement pas formelle, mais au contraire incertaine, créatrice, ... bien que le résultat de cette activité soit formalisable, et de fait formalisé par un ensemble de "patterns".

On peut entendre ce paragraphe comme une illustration de la méthode proposée, à savoir confronter en isolant les différences et construire à partir d'elles du neuf (ici, distinguer le couple formel / formalisable).

10. POINT DE VUE ÉPISTÉMOLOGIQUE

Toute une suite de travaux récents, l'ensemble des travaux sur les traces (Champin P.A., et al, 2002), les travaux sur les techniques de conception qui interrogent directement Peirce (Morand B., 2004), les travaux sur les documents multistructurés (Abascal R. et al, 2003), … pourraient être les indices d'un moment de l'informatique qui aurait besoin de « solutionner » (au sens de « mettre en (dis)solution ») les notions d'acteurs, de langage et de sujet d'étude qui jouent au sein de la définition canonique de modèle.

Si c'est le cas, et comme nous avons montré que ce mouvement trouve sans doute ses origines dans des domaines extérieurs à l'informatique, il pourrait se révéler intéressant de chercher les modalités selon lesquelles ces domaines différents pourraient communiquer, au moins dans le cadre restreint de cette problématique précise.

11. EXEMPLES D'APPLICATIONS

Nous listons en vrac quelques idées d'applications qui nous sont apparues au cours de notre travail.

Lorsque l'on fait de l'ingénierie dirigée par les modèles (Mukerji J. et Miller J., 2001 ; Site Internet MDA), il faudrait normalement pouvoir séparer absolument le modèle indépendant de la plate-forme d'implémentation du modèle qui en dépend. Malgré tout il n'est pas rare que la construction du PIM (Plateform Independant Model) soit l'occasion de quelques choix qui tiennent compte de la plate-forme, c'est-à-dire du PSM (Plateform Specific Model). Une fois la trace de construction présentée, il sera possible d'isoler par application de patterns les zones litigieuses du PIM et de permettre ainsi leur modification, plus tard, pour « assainir » le modèle.

L'application de patterns sur la trace de modélisation permet aussi de reconnaître des habitudes de modélisation et d'établir ainsi la signature d'un modélisateur. Par extension, il est possible de vérifier l'application de bonnes pratiques de modélisation dans une optique par exemple pédagogique.

Enfin nous pouvons mesurer la part qui revient à l'outil dans l'obtention du modèle final. C'est la problématique de l'outil qui à la fois soutient et sous-tend l'activité de son utilisateur.

12. CONCLUSIONS

Partis d'un problème dont l'énoncé pouvait nous paraître simple (confronter deux modèles), nous avons été contraints de nous rendre à l'évidence de sa grande complexité après avoir interrogé les différentes modalités que recouvrait la définition canonique d'un modèle. Nous avons cherché à proposer les moyens d'opérer avec cette complexité sans pour autant trop la réduire. Notre solution encore neuve demande sans doute d'être davantage contrainte, mais elle garde au moins pour elle de se maintenir dans une posture opérationnelle et créatrice face à une problématique qui justement ne cesse d'apparaître comme opposée à toute opérationnalisation.

RÉFÉRENCES

Abascal R. and Beigbeder M. and Bénel A. and Calabretto S. and Chabbat B. and Champin P.A. and Chatti N. and Jouve D. and Prié Y. and Rumpler B. and Thivant E., 2003 *Modélisation de la structure multiple des documents.* Rapport de Recherche LIRIS Lyon. Mars 2003. 10 p.

Abd-Ali, J. and K. El Guemhioui, 2005. *An approach to models comparison based on their semantics.* University of Quebec in Outaouais (UQO).

Bateson, G., 1979. *La Nature et la Pensée*, pages 74,100 107,116–121. Editions du Seuil, 1979.

Berlage, T., 1994 *A Selective Undo Mechanism for Graphical User Interfaces Based On Command Objects.* ACM Transactions on Computer-Human Interaction (TOCHI) Volume 1, Issue 3 Pages: 269-294

Caplat G. 2008 *Modèles et métamodèles*, PPUR Editions (à paraitre 2008)

Champin P.A., Prié Y. and Mille A., 2002 *Musette : un modèle pour réutiliser l'expérience sur le web sémantique.* Journées "Web Sémantique" Action Spécifique STIC CNRS, oct. 2002, Paris.

Deissenboeck F. and Ratiu D., 2006 *A unified meta model for concept-based reverse engineering.* Proceedings of the 3rd International Workshop on Metamodels, Schemas, Grammars and Ontologies.

Deleuze G., 1980 *Mille Plateaux* , chapter 5, pages 140–184. Les Editions de Minuit.

Deleuze G., 1988 *Le Pli*, chapter 2, pages 20–37. Les Editions de Minuit.

Derrida J., 1967 *De la Grammatologie*, pages 21–41,95 105. Les Editions de Minuit.

Eco U., 1990 *Les limites de l'interprétation*, pages 299 306,369–383. Edition Le Livre De Poche biblio Essais 4192.

Madhavan J., and Bernstein P., and Domingos P., and Halevy A., 2002 *Representing and reasoning about mappings between domain models*. Eighteenth National Conference on Artificial Intelligence (AAAI'2002), Edmonton, Canada.

Mancini R., and Dix A., and Levialdi S., 1996 *Reflections on Undo*. Technical report, University of Hudders eld.

Masuda H., and Imamiya A., *2004 Design of a Graphical History Browser with Undo Facility, and Visual Search Analysis*. Systems and Computers in Japan, Vol. 35, No. 12

Morand B., 2004 *Logique de la conception. Figures de sémiotique générale d'après Charles S. Peirce*. L'Harmattan

Mukerji J. and Miller J., 2001 *Technical Guide to Model Driven Architecture* Overview and guide to OMG's architecture

De Saussure F., (1913)1995 *Cours de Linguistique Générale*. Editions Payot.

Shannon C.E., 1948 *A Mathematical Theory of Communication*. Bell System Technical Journal, vol. 27, pages 379-423, 623-656.

Site Internet MDA, www.omg.org/mda/