Vers des traces numériques comme objets informatiques de premier niveau

Pierre-Antoine Champin, Alain Mille, Yannick Prié

Intellectica, 2013/1, **59**, pp.

Vers des traces numériques comme objets informatiques de premier niveau : une approche par les traces modélisées

Pierre-Antoine CHAMPIN[*], Alain MILLE[#], Yannick PRIÉ[♦]

RÉSUMÉ. Toute activité médiée par l'outil informatique laisse des traces numériques qui constituent en puissance des connaissances utiles, pour cette activité ou des activités futures. L'approche proposée dans cet article vise à exploiter effectivement ces connaissances dans des systèmes informatiques dits à base de m-traces. Cette approche consiste à proposer une modélisation *a priori* des traces, et à capturer leurs diverses interprétations possibles dans des transformations reproductibles. Nous proposons un méta-modèle général des systèmes à base de traces modélisées, et illustrons sa mise en œuvre dans deux applications. Nous discutons des défis que pose la conception de tels systèmes, notamment pour la prise en compte pluridisciplinaire de l'activité humaine tracée.

Mots-clés : Traces numériques, ingénierie des connaissances, dynamique de la connaissance, modèle d'interprétation de m-trace, Système de Gestion de Base de Traces modélisées (SGBTm).

ABSTRACT. Towards digital traces as first class digital objects: an approach based on modelled traces. Every computer-mediated activity leaves digital traces that are potentially usable as knowledge, for that activity or future activities. The approach proposed in this paper aims at actually using that knowledge in so-called trace-based computer systems. This approach consists in *a priori* modeling traces, and capturing their various possible interpretations in reproducible transformations. We propose a general meta-model for modeled-trace-based systems, and illustrate its deployment in two applications. Finally, we discuss the challenges raised by the design of such systems, especially in its taking into account the traced human activity, hence its multidisciplinary nature

Keywords: digital traces, knowledge engineering, knowledge dynamics, m-trace interpretation model, modelled Traces Base Management System (mTBMS).

[*] Université de Lyon, CNRS, Université Lyon 1, LIRIS, UMR5205, F-69622.
pierre-antoine.champin<at>liris.cnrs.fr.
[#] Université de Lyon, CNRS, Université Lyon 1, LIRIS, UMR5205, F-69622.
alain.mille<at>liris.cnrs.fr.
[♦] Université de Nantes, CNRS, LINA, UMR6241, École Polytechnique, rue Christian Pauc, BP 50609, F-44306 Nantes Cedex 3, yannick.prie<at>univ-nantes.fr.

1 - INTRODUCTION

Par construction, l'informatique produit et utilise en permanence des traces. Tout processus informatique travaille sur des données qu'il gère dans des mémoires plus ou moins volatiles et produit de nouvelles données qui seront à leur tour stockées dans de telles mémoires. Toute inscription numérique participe donc au traçage des processus informatiques qui ont permis de la produire. Ces inscriptions numériques permettent de rendre compte des processus informatiques eux-mêmes, en tant qu'elles résultent de l'exécution de programmes (Deransart *et al.*, 2002), mais elles rendent également compte de processus interactifs avec l'humain, en tant qu'elles participent à des systèmes applicatifs exploités *in fine* par des humains, dans le cadre d'une activité humaine dite *médiée* par l'environnement informatique (Kaptelinin, 1996).

L'activité médiée par un environnement informatique produit des inscriptions qui sont liées à tous les processus informatiques mobilisés par cette activité. Par exemple une journée de travail classique sur un ordinateur connecté au réseau mondial produira différents types d'inscriptions à la fois sur cet ordinateur et sur les machines participantes. Si l'on s'intéresse aux inscriptions résultantes de l'activité à la fin de la journée, on aura des documents créés et reçus, courriers électroniques, messages instantanés, historiques de navigation pour ce qui concerne les inscriptions manipulées par l'utilisateur ; mais également des fichiers de journal (*logs*) des applications et des serveurs impliqués. Si l'on s'intéresse aux inscriptions associées au déroulement même de l'activité pendant celle-ci, on ajoutera à la liste les différentes structures de données manipulées par ces applications : messages en cours d'écriture, indicateurs de charge réseau, fenêtres à l'écran, occupation de la mémoire, etc.

Les inscriptions dont il est question sont des traces numériques au sens large (Laflaquière *et al.*, 2006), qui peuvent prendre n'importe quelle forme : document numérique, structure de données, *log*, etc. Nous pouvons faire deux remarques à leur sujet. Tout d'abord, le statut de trace n'est que marginalement pris en charge comme tel par le système informatique, et l'interprétation de l'inscription en tant que trace est alors extérieure au système qui l'a produite : ce serait par exemple le cas pour une structure de données telle qu'un contact dans un annuaire, interprétée comme trace de l'activité qui l'a générée. Ce statut de trace liée à une activité est renforcé par la prise en compte de la temporalité intrinsèque du processus ayant produit les inscriptions, comme par exemple avec la date de modification d'un document, ou bien dans celui du marquage temporel des différents événements qui composent un fichier *log*. Ensuite, les traces considérées comme telles dans les systèmes informatiques possèdent leurs propre structures et formats adaptés au domaine applicatif dans lequel ces traces sont mobilisées : formats de fichiers *log* de serveurs webs, d'applications EIAH (George *et al.*, 2012) ou encore d'historiques de navigation Web. Il n'existe pas de modélisation générale de ces traces numériques qui en permette une utilisation par-delà les domaines applicatifs particuliers, et qui soit associée à des processus de manipulation génériques.

Notre objectif – ambitieux – est pourtant d'être à même de construire des traces numériques qui soient des objets informatiques de premier niveau. Il s'agit pour nous de définir un nouvel objet *trace numérique* qui d'une part porte en tant que telle différents caractères – notamment temporels – qui en font une trace explicitement manipulée par le système informatique, et d'autre part soit suffisamment générique pour pouvoir être utilisable dans des contextes applicatifs variés.

Dans cet article nous présentons une approche d'ingénierie des connaissances de la trace développée depuis plusieurs années par l'équipe SILEX du LIRIS, celle des systèmes à base de traces modélisées (SBTm). Notre première contribution concerne donc la notion de trace modélisée explicitement. Une trace modélisée (m-trace) est composée d'éléments temporellement situés, appelé obsels (pour *observed elements*), elle est associée à un modèle de m-trace regroupant des types d'obsels, qui en fournit un guide de construction et de manipulation. Notre seconde contribution est liée à la volonté de pouvoir prendre en compte différents niveaux d'interprétation des traces numériques, ainsi que les transformations permettant de passer d'un niveau à l'autre, transformations considérées comme interprétations automatisées. Un système informatique de gestion de bases de traces modélisées (SGBTm) est alors un outil permettant manipulations et transformations de traces modélisées, explicitant à tous niveaux les décisions prises sous la forme de lignées de traces et constituant un système assistant l'interprétation de celles-ci.

Dans une première section, nous présentons de façon générale ce que signifie une approche des traces numériques à base de modèles et dans quelle mesure il est possible de considérer une transformation de traces dirigée par des modèles comme interprétation automatique de ces traces. La seconde section est consacrée à l'approche des systèmes à base de traces modélisées, et nous y détaillons les concepts les plus importants notamment ceux liés aux bases de traces et à leurs évolutions. La troisième section est l'occasion de présenter deux systèmes à base de traces modélisées concrets, l'un dédié à diverses pratiques réflexives dans le cadre du tutorat en ligne synchrone, l'autre à l'analyse de l'activité du conducteur automobile. La dernière section est une discussion des questions de modélisation de traces et d'ingénierie des systèmes à base de traces, dans laquelle l'accent est mis sur les conséquences du caractère d'objet informatique des traces modélisées.

2 - UNE APPROCHE DE MODÉLISATION ET DE TRANSFORMATION DES TRACES À BASE DE CONNAISSANCES

Le concept de trace échappe à une définition générale, mais quand le terme est employé – dans un contexte ou un autre – il s'agit alors d'un construit cognitif mobilisant les capacités d'interprétation de l'observateur pour le produire. Si ce construit cognitif est inscrit sur un support partageable, il prend forme d'une manière ou d'une autre. Sa nature de trace de quelque chose est alors partageable pour peu que la forme d'inscription le soit, matériellement et symboliquement. Nous avons vu que l'activité médiée par un environnement informatique produit des marques numériques obligatoires dans

l'environnement, la question est donc de savoir comment il est possible de construire des traces inscrites et partageables à partir de ces marques.

2.1 - Usages ad hoc *des traces numériques et l'observation de l'activité*

Observer une activité pour la comprendre consiste à en considérer des observables comme permettant de construire des *éléments* guidant une interprétation. La séquence d'éléments formant une trace peut alors servir de *support*, de *justification* et *d'explication* à l'interprétation. Lorsque l'activité est médiée par un environnement informatique, il est facile d'instrumenter celui-ci pour qu'il collecte des éléments considérés comme potentiellement porteurs de sens et constituer alors des *traces numériques.*

Les traces numériques ont été utilisées d'emblée pour faciliter le débogage de programmes informatiques avec l'idée qu'un observateur averti (le programmeur en général), *analyste* de l'observation faite, sera capable d'interpréter les traces issues de l'exécution du programme pour en comprendre le comportement et le corriger en cas de besoin. C'est ainsi que depuis les premiers environnements informatiques, l'exécution d'un programme peut être accompagnée du *dump*[1] de la mémoire telle qu'elle est observée au moment de l'occurrence d'une *exception*[2]. La trace produite peut être standard (toujours le même type de trace) ou pilotée par le concepteur/analyste qui dispose d'outils de traçage lui permettant de suivre un certain nombre d'événements qu'il considère comme représentatifs du processus observé (Deransart *et al.*, 2002).

A l'instar du programmeur, analyste du comportement d'un processus automatique qu'il a lui-même conçu, il est possible d'observer des comportements de processus médiés par un environnement numérique, dès lors que l'instrumentation de l'environnement permet de laisser des marques suffisamment pérennes pour être observées par un *analyste* (qui peut être un analyste professionnel ou simplement une personne cherchant à comprendre). Il convient alors de lui présenter une représentation interprétable de la trace collectée à partir des marques disponibles. Cette représentation est le résultat d'un calcul qui peut être élémentaire (par exemple affichage en hexadécimal des informations collectées pour le programmeur) ou constituer une présentation spécifique associée à un mode d'interprétation choisi (ou non) par le lecteur humain de la trace (choix de forme, d'ampleur, de niveau de détail, de présentation du temps, etc.).

Les traces numériques sont donc considérées comme des données utilisables par un traitement informatique permettant d'en tirer des

---

[1] Dump mémoire : impression du contenu de l'ensemble de la mémoire utilisée par un programme en cours d'exécution.

[2] Une exception est un cas qu'un système informatique ne peut pas traiter. Par exemple, une division par zéro, l'accès à une zone mémoire non existante, la manipulation d'une valeur supérieure à la capacité de représentation de la machine. Lorsqu'une exception arrive, le système déclenche automatiquement un processus de gestion d'exception fournissant des informations sur le contexte de son occurrence. Le programme est en général enlevé de la liste des programmes en cours d'exécution. Le terme *erreur* est parfois utilisé pour désigner ces exceptions, mais la notion même d'erreur renvoie à une interprétation et même à un jugement.

informations grâce à un algorithme conçu pour cet objectif. En particulier, les éléments collectés sont traités de manière à présenter à l'utilisateur une trace matérialisée pour qu'il puisse l'interpréter. La présentation est le résultat d'un calcul permettant de transformer les données brutes collectées en symboles accessibles dans le registre de lecture de l'analyste.

Un traitement statistique des éléments observés peut être mené par un algorithme guidé par des heuristiques dépendant de l'interprétation qui est recherchée. L'exemple le plus évident est la fouille de traces numériques qui cherche à établir des régularités de forme, à mettre en œuvre des techniques d'analyse de données ou encore à identifier des *modèles* de comportement ou de processus (Song *et al*., 2009 ; Van der Aalst *et al*., 2003 ; Cook *et al*., 1998). Une classe d'usage de ce type, très importante et déjà ancienne, est celle du calcul d'indicateurs d'apprentissage humain utilisés pour la recommandation, la personnalisation des environnements informatiques, etc. De tels usages des traces numériques se sont généralisés, les champs de recherche et les domaines d'application sont notamment particulièrement nombreux sur le web[3] (Sachan *et al*., 2012).

La fouille de traces numériques peut être intégrée dans un processus plus général de *découverte* de connaissances sur les comportements, les activités ou les usages. L'analyste expert y joue le rôle le plus important pour la construction des connaissances retenues. L'usage des traces séquentielles d'interactions a été théorisé par Fischer & Sanderson (1996) qui démontrent que le travail essentiel de l'analyste est de trouver les *transformations* à appliquer aux observations brutes pour y découvrir des descriptions utiles à l'explication du processus observé. En effet, les observations brutes sont parcellaires, atomiques, élémentaires et exprimées dans le vocabulaire des instruments de collecte, tandis que l'expression de la connaissance sur l'activité se fait dans le registre de l'activité. Le processus d'interprétation de traces nécessite donc une transformation, intégrant les compétences et connaissances de l'analyste, pour reformuler les séquences d'éléments bruts collectés (actions et réactions élémentaires) en séquence d'éléments ayant un sens pour l'activité observée. Par ailleurs, comme le montrent de très nombreux travaux sur l'analyse d'activité et sur la découverte de connaissances (Fayyad *et al*., 1996 ;Brachman & Anant, 1996), de tels processus sont itératifs et interactifs.

La littérature et les pratiques (Mille & Marty, 2009 ; Laflaquière, 2009) démontrent que des opérations de collecte, de modélisation, de transformation, de reformulation, d'exploration interactive sont donc toujours nécessaires pour la mise en œuvre d'observations nécessitant des interprétations nombreuses. L'intégration de ces différents processus nous amène à proposer une approche unifiante de structures de représentation riche et accompagnées d'opérateurs spécifiques à la manipulation de traces issues d'observation.

[3] Par exemple, Google Analytics qui offre des moyens d'analyse précis de la fréquentation d'un site web : http://www.google.com/intl/en/analytics/.

2.2 - Modéliser les traces numériques pour associer un modèle d'interprétation aux éléments observés

Si les traces numériques peuvent bénéficier de la normalisation de leurs éléments[4] (Nottingham, M. & Sayre, R., 2005), leur structure ne possède en général pas de sémantique propre et les modèles sous-jacents, c'est-à-dire la sémantique des choix faits par le concepteur de l'instrumentation produisant la trace numérique, ne sont pas explicités au niveau informatique. Typiquement, la description des éléments observés se limite en effet souvent à la documentation textuelle d'un format de *logs*, que le programmeur devra lire et interpréter pour construire un programme qui respecte cette sémantique implicite pour la rendre *opérationnelle*. Lors des processus de découverte de connaissances sur les traces numériques, les connaissances produites, et qui pourraient donc participer à la description de la sémantique opérationnelle des traces observées, ne sont en général pas associées structurellement aux traces numériques concernées car rien n'est prévu pour cela dans la structure informatique des données représentant les traces numériques.

Nous proposons de porter un regard différent sur les traces numériques d'interaction, de les considérer comme des *inscriptions de connaissance* destinées non seulement à porter l'information collectée mais aussi les éléments permettant d'en faciliter l'interprétation humaine et computationnelle. L'exploitation des traces numériques d'interaction devient alors une question d'ingénierie des connaissances, ce qui permet d'élargir considérablement la panoplie des outils permettant d'exploiter, transformer, partager, réutiliser ces traces en profitant d'une sémantique explicite et opérationnelle. Le caractère explicite et opérationnel de la sémantique permet d'une part de déclencher des calculs dans un système d'inférence généraliste ; d'autre part de présenter à l'analyste une représentation symbolique (graphique ou textuelle) du modèle. Une telle présentation vise à *expliquer* la forme symbolique choisie pour représenter la collection *d'éléments observés* : l'utilisateur peut demander à voir une représentation symbolique du modèle explicatif de la sémantique d'un élément observé O ayant la sémantique (modèle) S, et peut donc explorer comment l'élément O a pu être considéré comme *observé*. Par exemple, si on observe un motif d'opérations pour faire un virement sur un compte bancaire, on peut facilement en inférer que l'on est en train d'observer une opération générique de type *virement interne*, *virement international, virement national, etc*.

Nous choisissons d'exprimer la sémantique sous la forme d'un *modèle de trace* associé à une collection d'éléments observés. La notion de modèle est ici considérée du point de vue des techniques d'intelligence artificielle, c'est-à-dire comme *modèle d'interprétation* permettant à un *système d'inférence* d'établir la validité (ou la non validité) de *propositions* concernant le *domaine* décrit par les propositions. Un tel modèle peut être associé à un moteur

[4] Les éléments possibles à collecter peuvent être décrits conformément à une norme, comme par exemple les événements qui circulent pour faire de la supervision d'un réseau, ou encore pour décrire des éléments systématiquement produits par les applications à des fins de contrôle : http://www.rfc-editor.org/.

d'inférence, il est alors possible de produire de nouvelles propositions logiquement déductibles au sein du modèle, autorisant de réaliser des transformations *valides* au sein d'un même groupe de traces. Dans le cas des traces numériques d'interaction, le domaine est constitué par ce qu'il est possible d'observer et ce qui est vrai (ou faux) du point de vue de l'observation en cours : essentiellement des faits, mais également des relations entre les faits qui sont par exemple déclarées par un observateur *averti*[5] à partir de ce qu'il sait déjà. En pratique, il n'est donc pas question de décrire le monde extérieur, mais les *interactions* telles qu'elles sont reconnues par l'observateur.

Nous proposons d'utiliser une représentation *ontologique formelle* des éléments observés associée à un *espace temporel* explicite, précisant la sémantique opérationnelle liée au temps. L'ontologie formelle est un *modèle* du domaine d'observation correspondant à la collection d'éléments observés concernés (Bachimont, B., 2004, p. 160 *sqq*.). C'est *l'inférence logique* qui est le mode d'inférence associé, et c'est la sémantique des relations entre éléments observés qui exprime cette sémantique.

Du statut de *structure de données*, la trace ainsi *modélisée* passe au statut d'*inscription de connaissance,* ajoutant de nombreux moyens de calcul à ceux déjà disponibles, mais surtout permettant de réintégrer dans le domaine d'observation de nouveaux modèles (issus par exemple de ré-interprétation de la trace, d'hypothèses, de calculs...). On peut alors *transformer* une trace *modélisée* en d'autres traces modélisées intégrant ces nouveaux modèles (par exemple une relation de causalité entre éléments observés).

Il n'existe pas de *modèle unique* permettant de décrire une trace, mais le *méta-modèle* générique que nous proposons spécifie comment un modèle de trace peut s'exprimer. C'est ce méta-modèle qui donne l'expressivité des modèles de m-trace0, et chaque trace *modélisée* conformément peut faire l'objet d'une interprétation qu'il convient de réifier par une opération que nous avons proposée d'appeler transformation. La section suivante précise cette relation entre interprétation et transformation.

2.3 - L'interprétation de trace vue comme transformation

Nous considérons l'interprétation d'une trace comme une « reformulation » de cette trace exprimée dans un vocabulaire initial explicite dans une autre trace exprimée dans un autre registre de vocabulaire, qui permet d'intégrer des concepts non représentés dans la trace initiale. Par exemple, il sera facile de reformuler une séquence -*cliquer sur l'icône de toto, lancement de word, chargement de toto, affichage de la fenêtre associée, focus utilisateur sur la fenêtre ouverte*- par -*ouverture d'un document word nommé toto*-. On voit bien sur cet exemple qu'il va falloir choisir comment modéliser -*ouverture d'un document word nommé toto-,* ce qui pourra se faire par un élément observé -*ouverture d'un document*-, et cet élément aura deux propriétés (au moins) :

[5] Observateur *averti :*observateur qui connaît le contexte (au sens large) de l'observation et est donc capable de compléter par ses connaissances ce qui ne serait pas explicitement représenté dans une trace donnée.

type de document (prenant pour valeur « Word ») et nom du document (prenant pour valeur « toto »).

L'opérateur de transformation est donc constitué d'une partie « conditionnelle » consistant à chercher une séquence satisfaisant la spécification « cliquer sur une icône, lancement d'une application associée, affichage d'une fenêtre associée, mise du focus sur la fenêtre associé » et d'une partie « conséquence » consistant à indiquer la construction à faire en cas de succès et dans l'exemple : créer un élément observé nouveau (dans la trace transformée) dont le type est *-ouverture d'un document-* et possédant deux propriétés (type_doc et nom_fichier) qui seront renseignés à partir des valeurs des éléments observés dans la séquence satisfaisant la partie *condition* de la transformation.

Ce mécanisme de transformation relève de l'inférence puisque la transformation permet d'affirmer l'observation de *-ouverture d'un document-* à partir de l'observation d'autres éléments, n'ayant pas de liens *a priori* avec l'observation inférée, mais affirmés dans une trace source existante.

D'un point de vue technique, la transformation est donc une *réécriture* d'une trace *observée selon un modèle A* en une nouvelle trace *observée selon un modèle B.*

La posture d'ingénierie des connaissances prise pour considérer les traces comme des m-traces (traces modélisées) associées à des opérateurs de transformation de m-trace change fondamentalement le statut des traces, initialement simples collections d'éléments observés, en leur fournissant une forme d'inscription computationnelle de connaissances.

3 - SYSTÈMES À BASE DE TRACES MODÉLISÉES

Nous présentons maintenant le méta-modèle proposé par l'équipe SILEX (Settouti et *al.*, 2009 ; Settouti, 2011) pour la représentation des traces modélisées et leur exploitation dans des systèmes à base de connaissances dédiés, nommés Systèmes à Base de Traces modélisées (SBTm).

Exemple : tout au long de cette section, nous illustrerons les notions présentées par l'exemple d'une utilisatrice (Alice) d'un environnement informatique, utilisant le courrier électronique pour communiquer et échanger des documents avec ses collègues.

3.1 - Traces modélisées (m-traces)

Nous définissons tout d'abord la notion centrale de notre méta-modèle, celle de *trace modélisée,* ou *m-trace*. Pour cela, nous devons d'abord définir les notions *d'obsel* et de *modèle de m-trace*.

Toute activité tracée est représentée par une liste d'*éléments observés*, ou obsels[6]. Un obsel est constitué :

[6] Ce néologisme, construit sur le même principe que le mot *pixel* (*picture element*), permet d'alléger le discours lorsqu'il est souvent question d'éléments observés.

d'une date de début et d'une date de fin situant l'obsel dans le temps de l'activité, ces deux dates peuvent éventuellement être égales ;

- d'un sujet (l'agent dont l'activité est observée) ;
- d'un type rattachant cet obsel à une catégorie explicite d'éléments observés ;
- d'un ensemble d'attributs de la forme <type d'attribut, valeur>.

On remarquera que les constituants d'un obsel sont, volontairement, peu spécifiés par le méta-modèle. Ils dépendent en effet fortement de l'activité, qui devra donc être décrite par un *modèle de m-trace*. Ce modèle de m-trace définit :

- la manière de représenter le *temps* ;
- la manière de représenter les *sujets* ;
- les *types d'obsels* permettant de décrire l'activité ;
- pour chaque type d'obsel, les *types d'attributs* possibles, spécifiant en particulier le type de valeurs qu'ils peuvent prendre ;
- un ensemble de *types de relations binaires* que peuvent entretenir les obsels entre eux ;
- un ensemble de *contraintes d'intégrité* que doivent vérifier les obsels pour être conformes à ce modèle.

Exemple : dans l'activité « Courrier électronique », on décide de considérer le temps à la seconde près. On identifiera les sujets des obsels par leur adresse électronique.

On peut identifier trois types d'obsels : la réception d'un message (MsgReçu), l'envoi d'un message (MsgEnvoyé) et la sauvegarde d'une pièce jointe (PjSauvée). Les obsels de type MsgReçu et MsgEnvoyé auront comme attributs semblables le contenu du message et éventuellement de la pièce jointe (pj). Par ailleurs, les obsels MsgReçu auront comme attribut supplémentaire l'adresse e-mail de l'expéditeur, tandis que les obsels MsgEnvoyé auront l'adresse e-mail du ou des destinataires, et éventuellement le nom du fichier d'où provient la pièce jointe (pj_Origine). Enfin, les obsels PjSauvée auront comme attribut le nom de fichier sous lequel la pièce jointe a été sauvée.

Les contraintes d'intégrité de ce modèle imposent aux obsels des trois types d'être instantanés (leur date de début doit être égale à leur date de fin). Par ailleurs, l'obsel destination d'une relation Depuis doit avoir son attribut pj (pièce jointe) renseigné. Enfin, l'attribut pj Origine d'un obsel MsgEnvoyé ne peut être renseigné que si l'attribut pj l'est également.

Un modèle de m-trace permet également d'associer à un type d'obsel un ou plusieurs types d'obsel parents. Ce mécanisme a un intérêt syntaxique et sémantique. Syntaxiquement, il permet l'héritage des attributs des types parents vers le type fils, ce qui favorise la modularité. Sémantiquement, tout obsel du type fils appartient, indirectement, à tous ses types parents (même si,

conformément à la définition ci-dessus, il n'a qu'un seul type direct), ce qui augmente les possibilités de raisonnement (et donc de transformation) sur les m-traces concernées. Les types de relation peuvent également entretenir des liens de parenté, avec le même intérêt sémantique que pour les types d'obsels.

Exemple : Dans notre exemple de modèle de m-trace ci-dessus, les attributs communs à MsgReçu et MsgEnvoyé (contenu et pj) peuvent être factorisés dans un type abstrait[7] *Message. Ce type est ensuite déclaré comme type parent de MsgReçu et MsgEnvoyé. Le modèle de m-trace correspondant est représenté sur la Figure 1.*

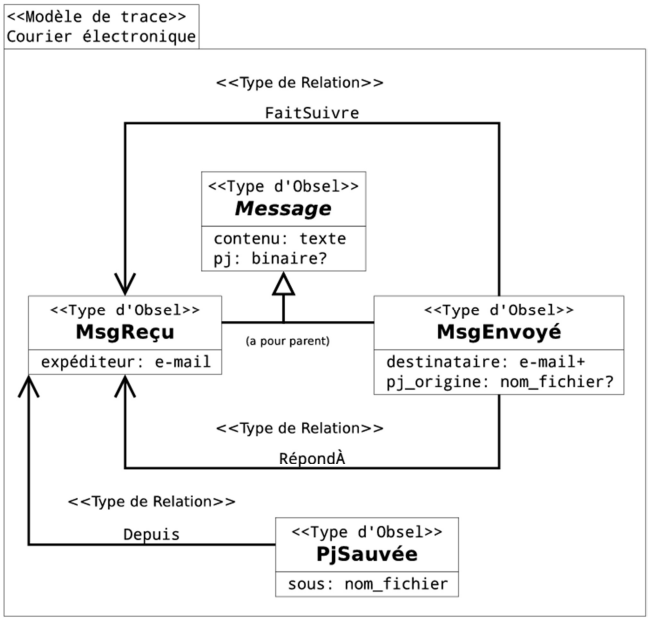

Figure 1

Exemple de modèle de m- trace.

Enfin, un modèle de m-trace peut se composer par importation d'un ou plusieurs autres modèles de m-trace, à condition que tous ces modèles partagent la même représentation du temps et des sujets. Dans ce cas, le modèle importateur comportera tous les types d'obsel, d'attribut et de relation des modèles importés, ainsi que les contraintes d'intégrité associées. Ce mécanisme est intéressant du point de vue de l'ingénierie des connaissances, permettant la réutilisation de modèles déjà définis. On peut ainsi définir le

[7] Il est abstrait au sens où il n'a pas vocation à être utilisé directement comme type d'un obsel. Les obsels pourront avoir comme type MsgReçu ou MsgEnvoyé, et seront donc *indirectement* de type Message.

modèle d'une activité composite en important les modèles des sous-activités plus focalisées. On peut également décrire de manière plus précise une spécialisation d'une activité en important son modèle et en définissant de nouveau types sur la base des types importés et en y ajoutant de nouveaux attributs par exemple. Ceci s'avérera une opération très utile réalisée par transformation par exemple.

Le modèle de m-trace « Courier électronique » défini ci-dessus pourra être importé dans un modèle plus large « Activité bureautique », qui importera également un modèle « Traitement de texte ». On pourrait également tracer l'activité d'utilisation d'une application spécifique de courrier électronique. Le modèle correspondant importera le premier modèle « Courier électronique », en le spécialisant par de nouveaux types (d'obsel, d'attribut, de relation) dérivés ou liés aux types importés.

Il nous est maintenant possible de définir précisément une *trace modélisée*. Elle est constituée :

- d'une *référence à un modèle de m-trace* défini par ailleurs ;
- d'une *origine* (date de début) et d'une *durée* délimitant l'*extension temporelle* de la m-trace ;
- d'une *liste d'obsels* ;
- d'un *ensemble de relations binaires* entre ces obsels.

L'extension temporelle de la m-trace correspond à la période durant laquelle l'activité tracée a été observée. Dans la mesure où le premier (resp. le dernier) obsel ne coïncide pas forcément avec le début (resp. la fin) de l'extension temporelle, cette information peut-être importante pour interpréter l'*absence* d'obsel dans l'intervalle de temps correspondant.

L'origine et la durée de la m-trace doivent être conformes à la manière de représenter le temps telle qu'elle est spécifiée par le modèle. Les types et les attributs des obsels, ainsi que les relations définies entre eux, doivent également être conformes au modèle. Enfin, les obsels doivent évidemment tous être inclus dans l'extension temporelle de la m-trace.

Exemple : la Figure 2 représente une trace modélisée de l'activité d'Alice relative au courrier électronique. Cette m-trace fait référence au modèle de m-trace « Courier électronique » défini plus haut. Son extension temporelle s'étend de lundi 09h à lundi 10h.

Elle comporte 4 obsels. Pour simplifier, nous n'avons pas fait figurer le sujet de chaque obsel (qui est toujours alice@example.org) ni la date de fin (qui est toujours égale à la date de début). À 9h15, Alice reçoit un mail de Bob ; à 9h31, elle enregistre la pièce jointe sous le nom bilan_détaillé.docx, puis répond à Bob à 9h32. À 9h47, elle envoie à Charlie un message en y joignant le fichier bilan_résumé.docx.

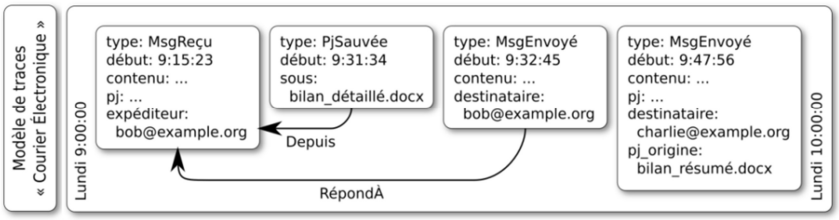

Figure 2

Un exemple de trace.

3.2 - Représentations du temps

La vocation de notre méta-modèle est de permettre de créer des modèles pour des traces d'activités diverses, nécessitant des représentations du temps différentes. Dans notre exemple d'activité bureautique, une résolution d'une seconde semble suffisante ; mais d'autres domaines d'application nécessiteront une représentation du temps plus précise, comme les m-traces de conduite automobile présentées à la section 0 ou les m-traces fournies par un système d'*eye-tracking*. À l'inverse, on pourrait souhaiter modéliser des m-traces à plus gros grain temporel où une résolution d'une heure, voire d'une journée, est suffisante, et où l'on n'a d'ailleurs pas toujours le moyen de dater plus précisément les éléments observés.

Par ailleurs, selon le contexte de collecte des éléments observés de l'activité tracée, il se peut qu'on ne dispose que d'une mesure *relative* du temps dans la m-trace. Par exemple, dans la m-trace représentée sur la *Figure 2* l'extension temporelle est définie entre lundi 9:00:00 et lundi 11:00:00, sans que l'on sache de quel lundi il s'agit. Cette information peut ne pas être disponible pour plusieurs raisons : soit elle a été perdue entre la collecte et le moment où la m-trace modélisée est constituée (certains fichiers *log* n'encodent pas la date complète), soit elle peut avoir été volontairement retirée pour des raisons de confidentialité.

C'est la raison pour laquelle notre méta-modèle permet de préciser comment un modèle de m-trace doit représenter les informations temporelles :

- Chaque modèle de m-trace doit spécifier la représentation du temps par *l'unité temporelle* qui sera utilisée par les m-traces et leurs obsels.
- Pour les m-traces pour lesquelles on dispose *d'une information temporelle absolue, l'origine est une date*, exprimée avec une précision égale à l'unité spécifiée par le modèle. Si on ne dispose que d'une *information temporelle relative, l'origine est un identifiant opaque* ; on ne peut alors pas situer les obsels de manière absolue (sur un calendrier). En revanche, si deux m-traces ont la même origine opaque, on peut comparer les dates de leurs obsels entre eux. Ceci sera important pour comparer des m-traces transformées issues de la même m-trace.

- La *durée* de la m-trace, et les dates de *début* et de *fin* des obsels, sont des *entiers* interprétés à partir de l'origine de la m-trace, dans l'unité spécifiée par le modèle.

La *Figure 2* montre la m-trace de la *Figure 1* en explicitant la représentation du temps comme nous venons de la décrire.

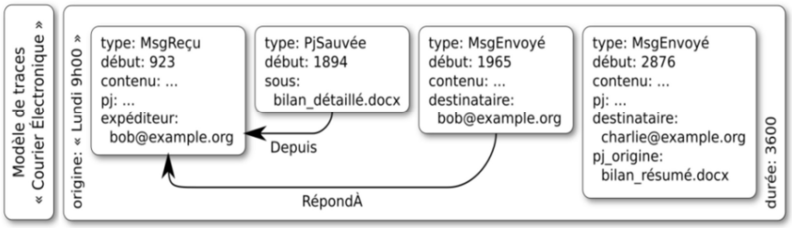

Figure 3

L'exemple de la Figure 2 avec la représentation unifiée du temps.

Enfin, dans certains cas, la m-trace ne sera constituée que d'une séquence ordonnée d'obsels, sans même que l'on puisse savoir le temps qui s'est écoulé entre chaque obsel. Dans ce cas particulier, le modèle utilise l'unité spéciale *séquence* ; les *dates* de début et de fin des obsels sont leur numéro d'ordre dans la m-trace, en commençant à 1 ; *l'origine* de la m-trace est forcément opaque ; sa *durée* est le nombre d'obsels qu'elle contient.

3.3 - Système à base de traces modélisées

Les notions de trace modélisée et de transformation de m-trace ainsi précisées, nous pouvons décrire l'architecture d'un Système à Base de Traces modélisées (SBTm), illustrée par la *Figure 4*.

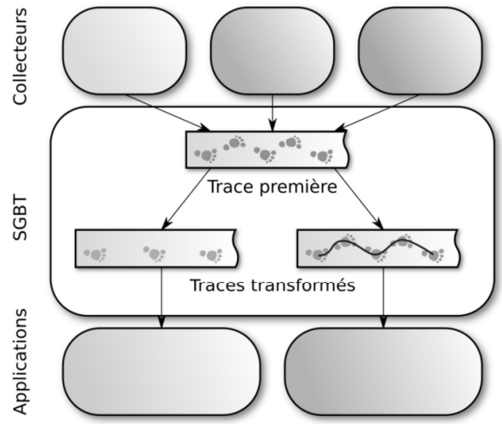

Figure 4

Architecture générale d'un Système à Base de Traces modélisées (SBTm) construit autour d'un Système de Gestion de Base de Traces modélisées (SGBTm).

Le composant central d'un tel système est le *Système de Gestion de Bases de Traces modélisées* (SGBTm). Ce dernier joue le même rôle que le Système de Gestion de Bases de Données dans une application standard, mais gère des objets particuliers, des *traces modélisées*, conformes au méta-modèle présenté plus haut. Il doit être suffisamment flexible pour autoriser la cohabitation de multiples modèles de m-traces (et leurs éventuelles évolutions). Il doit également être capable de gérer efficacement le caractère intrinsèquement dynamique des traces. Il doit enfin être capable de gérer les transformations (cf. ci-dessous).

Le SGBTm est alimenté par un ensemble de *collecteurs*, dont le rôle est de récolter l'information nécessaire à la constitution d'une ou plusieurs traces. Cette information peut être capturée de manière synchrone pendant l'activité, en scrutant les événements du système, ou *a posteriori* en utilisant par exemple les fichiers *log* laissés par le système. Tous ces canaux d'information exploités par les collecteurs sont appelés des *sources de collecte*. Ce sont les modèles de m-trace qui déterminent le sous-ensemble de l'information disponible qui sera enregistré dans ces traces. Toute m-trace constituée par les collecteurs est appelée m-trace *première* (par opposition aux m-traces transformées).

Les m-traces (premières et transformées) sont enfin exploitées par des applications. Certaines sont des applications de visualisation, qui présentent la m-trace à l'utilisateur sous différentes formes, qui peuvent être plus ou moins généralistes (par exemple la liste des obsels sous forme de tableau) ou spécialisées pour un modèle de m-trace donné et/ou pour une tâche particulière de l'utilisateur. D'autres vont appliquer des traitements aux m-traces afin d'altérer leur comportement, comme des systèmes d'assistance réutilisant les expériences passées de l'utilisateur.

3.4 - Transformation de m-trace

Comme nous l'avons dit plus haut (section 0), la notion de transformation est centrale dans un SBTm. Elle permet la reformulation et la composition des m-traces déjà présentes dans le système, qui constituent la forme privilégiée d'interprétation de ces traces.

Dans un SBTm, une m-trace *transformée* est définie par :

- une ou plusieurs m-traces sources (qui peuvent elles-mêmes être des m-traces premières ou transformées) ;
- une méthode de transformation ;
- éventuellement un ou plusieurs paramètres conditionnant l'exécution de la méthode de transformation.

Les propriétés de la m-trace transformée (son modèle, son extension temporelle, ses obsels et leurs relations) sont calculées en appliquant la méthode de transformation (éventuellement paramétrée) aux m-traces sources.

L'éventail de méthodes disponibles dépend de l'implémentation particulière du SGBTm, mais on peut distinguer les trois grandes catégories suivantes :

- Les méthodes de *sélection* visent à ne retenir dans la m-trace transformée qu'un sous-ensemble des obsels d'une unique m-trace source, selon un critère déterminé. Le modèle de la m-trace transformée est en général le même que celui de la m-trace source, et son extension temporelle peut être réduite si le critère de sélection est temporel.

Exemple : avec le modèle de m-trace « Courier électronique », on pourrait imaginer les sélections suivante : ne garder que les obsels ayant eu lieu entre 9h30 et 9h40 (critère temporel) ; ne garder que les obsels de type MsgEnvoyé (critère sur le type) ; ne garder que les obsels ayant une pièce jointe (critère sur les attributs) ; ne garder que les obsels ayant fait l'objet d'une réponse (critère sur les relations).

- Les méthodes de *fusion* rassemblent dans une même m-trace les obsels provenant de plusieurs m-traces sources. Si les sources ont des modèles de m-trace différents, il faudra spécifier pour la m-trace transformée un modèle qui importe tous ces modèles. Cela suppose que ces modèles soient homogènes dans leur manière de représenter le temps.

Exemple : la fusion de m-traces peut servir différents buts. On pourrait par exemple fusionner la m-trace de la Figure 3 avec la m-trace d'Alice pour la période 10h-11h et l'activité « Courrier Électronique », afin d'observer cette activité sur une plus longue durée. On pourrait aussi fusionner cette même m-trace avec la m-trace de Bob pour la même période et l'activité « Courrier Électronique », afin d'observer de manière plus globale la manière dont ils communiquent. Enfin, on pourrait fusionner cette même m-trace avec la m-trace d'Alice sur la même période pour l'activité « Traitement de texte », afin d'observer l'activité bureautique d'Alice dans son ensemble.

- **Les** méthodes **de *réécriture* peuplent la m-trace transformée avec** des obsels construits à partir d'un ou plusieurs obsels d'une *unique* m-trace source. Cette réécriture peut consister à recopier chaque obsel de la m-trace source en l'appauvrissant (*i.e.* en retirant certains attributs ou en altérant leur valeur) ou en l'enrichissant (*i.e.* en ajoutant des attributs/relations en utilisant des connaissances déduites de la m-trace ou externes). Mais la réécriture n'est pas forcément injective ; elle peut également consister à créer un nouvel obsel lorsqu'un *ensemble* d'obsels vérifiant certains critères est identifié dans la m-trace source.

Exemple : une m-trace de « Courrier Électronique » peut-être anonymisée en supprimant les champs expéditeurs et destinataire[8]*. À l'inverse, on peut imaginer de l'enrichir en utilisant un algorithme de détection d'émotion sur le contenu pour ajouter un attribut* émotion *dans chaque obsel (ce qui suppose d'étendre le modèle de m-trace en conséquence). On peut aussi réécrire cette m-trace sous une forme plus synthétique avec un obsel par jour, contenant le nombre de messages reçus et envoyés dans la journée (selon un modèle de m-trace à définir). Enfin, on peut imaginer une réécriture plus complexe où une séquence de messages se répondant les uns aux autres serait réécrite en un obsel de type Conversation, et où un envoi successif du même contenu à des adresses différentes serait réécrit en un obsel de type Diffusion.*

Notons que les transformations ne portent pas seulement sur les attributs et relations des obsels, mais peuvent également porter sur leurs propriétés temporelles ainsi que sur celles de la m-trace. Par exemple, une ré-écriture peut changer la granularité temporelle d'une m-trace (conversion de millisecondes en secondes) afin de la comparer avec une autre. Il peut également être nécessaire de changer l'origine d'une m-trace : cela peut être nécessaire si la date précise des obsels peut faciliter l'identification du sujet d'une m-trace anonymisée. Mais cela peut aussi permettre *d'aligner* deux m-traces s'étant déroulées à des moments différents, mais dont on souhaite comparer le contenu (même tâche effectuée dans des contextes différents).

Notons enfin que l'attribut *sujet* des obsels, et les transformations influant sur cet attribut, permettent d'étudier une *activité collective*. Par exemple, en fusionnant des m-traces individuelles ayant des sujets différents, on obtient une m-trace dont les obsels n'ont pas tous le même sujet ; on parle alors de m-trace *conjointe*. Une transformation de type réécriture peut ensuite créer de nouveaux obsels dont le sujet n'est plus un individu en particulier mais un groupe d'individus (par exemple, un obsel de type Conversation, dont le sujet serait le binôme [Alice et Bob]). On parle dans ce cas de m-trace *de groupe*. Une m-trace de groupe n'est pas forcément conjointe (si tous les obsels ont le même groupe pour sujet). On peut en revanche imaginer des situations plus complexes, comme une m-trace de groupe conjointe décrivant l'activité parallèle de plusieurs groupes, voire des groupes et de certains individus les composant.

3.5 - M-traces transformées, lignées et explications

On a vu ci-dessus qu'une base de m-traces contient des m-traces premières et des m-traces transformées. Ces dernières peuvent avoir, selon la méthode de transformation qu'elles utilisent, une ou plusieurs m-traces sources. Ces m-traces sources peuvent être premières ou transformées, avec comme restriction qu'une m-trace ne peut pas être, directement ou indirectement, sa propre

[8] Cette forme d'anonymisation est cependant assez naïve. Une véritable anonymisation requerrait sans doute également un traitement du corps des messages et des pièces jointes, qui peuvent également contenir des données nominatives.

source. En d'autres termes, l'ensemble des m-traces d'une base constitue un graphe orienté sans cycle. On appelle *lignée* d'une m-trace l'ensemble des m-traces transformées dont elle est, directement ou non, une source.

Cette généalogie des m-traces transformées est importante dans notre perspective de considérer les transformations comme des interprétations de la m-trace. Tout l'abord, une même m-trace peut être la source de plusieurs traces transformées, ce qui rend compte de la diversité des interprétations possibles Ensuite, les relations généalogiques entre m-traces devraient permettre d'*expliquer* une m-trace transformée en remontant aux m-traces sources qui ont permis de la construire. Dans cette optique, une m-trace transformée pourra définir une relation particulière entre ses obsels et les obsels des m-traces sources qui ont servi à les construire. En effet, bien que relativement simple à établir dans le cas d'un filtrage ou d'une fusion, ce lien vers les *obsels sources* peut être beaucoup plus délicat à reconstituer pour les transformations par ré-écriture, d'où l'importance de le conserver au moment où la m-trace transformée est calculée.

De la même manière, il est souhaitable que les obsels d'une m-trace première contiennent un maximum d'information sur la manière dont ils ont été construits à partir des informations disponibles dans les sources de collecte, ce qui constituerait une indication sur le modèle d'interprétation mobilisé pour la collecte. Mais ceci ne peut être codifié dans le méta-modèle, puisque par définition les sources de collectes ne sont pas modélisées. Cette responsabilité incombe donc aux concepteurs de modèles de m-traces et de collecteurs.

3-6 - Dynamique des bases de m-traces

Par nature, la manière de considérer les m-traces d'une activité est dynamique aussi bien au niveau de leur production qu'au niveau des interprétations variées et successives qui peuvent en être faites. Cette dynamique pose quelques questions essentielles à prendre en compte dans la gestion d'un SGBTm.

3.6.1 -Collecte, monotonie, amendement

Il existe différentes manières dont les m-traces premières peuvent évoluer. Nous avons décrit plus haut comment elles sont alimentées par des collecteurs, qui leur ajoutent des obsels. On qualifie la collecte d'évolution *monotone* dans la mesure où le contenu de la m-trace ne fait qu'augmenter. De plus, les obsels le plus souvent sont ajoutés dans l'ordre chronologique d'apparition dans l'environnement informatique ; nous qualifions alors cette collecte de *temporellement monotone*. Cependant, la collecte peut, dans certains cas, ne pas respecter la monotonie temporelle, pour les raisons suivantes :

- dans les cas d'obsels ayant une durée, il n'existe plus de chronologie unique, puisqu'on peut les ordonner par date de début ou par date de fin. Or un obsel peut commencer avant un autre et se terminer après lui, donc leur ordre diffère selon le critère choisi ;
- lorsqu'une m-trace est constituée à partir de plusieurs sources de collectes, ces dernières peuvent ne pas être totalement

synchronisées, et certains obsels peuvent être produits en retard par rapport à d'autres.

Enfin, il est important qu'un SGBTm garantisse à l'utilisateur tracé un droit de regard total sur ses propres traces. L'utilisateur doit donc avoir un moyen de modifier, indépendamment de la collecte, le contenu de ses traces, en modifiant ou même en supprimant certains obsels. Ce type de modification non monotone est appelé amendement d'une m-trace première.

Les m-traces transformées, quant à elles, sont calculées en appliquant une méthode de transformation à leurs m-traces sources. Il est intéressant de noter que, selon la méthode de transformation appliquée, la monotonie, temporelle ou non, n'est pas préservée entre les m-traces sources et la m-trace transformée. L'exemple le plus flagrant serait une transformation de sélection ne gardant que le dernier obsel de la m-trace source : même si cette dernière évolue de manière temporellement monotone, à chaque ajout, la m-trace transformée évolue de manière non monotone puisqu'elle subit la suppression de l'obsel devenu avant-dernier.

Un SGBTm doit donc être capable de gérer les transformations de m-traces quelle que soit la manière dont les sources évoluent.

3.6.2 - Archivage, oubli

Il existe des situations où l'on ne souhaite pas garder le lien entre une m-trace source et sa m-trace transformée. Une raison peut-être que la m-trace source est trop détaillée et volumineuse, et qu'on souhaite l'effacer ou l'archiver en dehors du SGBTm, pour ne garder que l'information plus synthétique de la m-trace transformée. Une autre raison peut être que la m-trace source contient des informations sensibles qui ont été volontairement exclues de la m-trace transformée, qui est donc la seule que l'on souhaite conserver.

Le SGBTm doit donc permettre l'*émancipation* d'une m-trace transformée, c'est à dire son changement de statut en m-trace première. La m-trace transformée ne sera alors plus considérée comme dépendante de son ancienne m-trace source, qui peut sans risque être retirée du SGBTm.

Exemple : Alice crée chaque mois, une m-trace transformée par ré-écriture qui contient un obsel par jour contenant simplement le nombre de courriers électroniques qu'elle a échangé. Le premier jour de chaque mois, elle émancipe la m-trace transformée du mois précédent, et archive sur un disque externe la m-trace première contenant tous les échanges de message. Elle peut ainsi garder dans son SGBTm une vue synthétique de son activité sur plusieurs mois ou plusieurs années, sans saturer son disque dur avec le détail des échanges.

4 – ILLUSTRATIONS

À titre d'illustration concrète des différents concepts proposés ci-dessus, nous présentons deux systèmes à base de traces modélisées, l'un dédié au tutorat en ligne synchrone et asynchrone, l'autre à l'analyse de données séquentielles issues de la conduite automobile.

4.1- VISU : un outil de tutorat en ligne à base de traces modélisées

Contexte applicatif. Différents acteurs interviennent dans le cadre de l'enseignement de Français Langue Étrangère (FLE). Les apprenants sont à distance, ce sont des étudiants qui dans le cadre de leur cours de FLE ont pendant sept semaines de suite une interaction directe (visioconférence) avec un tuteur situé en France. Les tuteurs eux-mêmes ne sont pas des professionnels, mais des étudiants de M1 en Sciences du Langage qui apprennent à devenir enseignants de FLE. On parlera alors d'apprenti-tuteurs qui réalisent à cette occasion leurs premiers tutorats. Le responsable de la formation est un enseignant qui pilote l'Unité d'Enseignement au sein de laquelle se déroule la formation des apprenti-tuteurs. Par ailleurs, dans le cadre de cette formation, les séances sont préparées par les apprenti-tuteurs qui mettent en place chaque semaine un plan de séance composé de quatre activités pédagogiques, lesquelles mobilisent différents documents (images, vidéos), mots-clés et instructions.

Les séances de tutorat peuvent alors avoir lieu, pendant lesquelles apprenti-tuteurs et apprenants mènent l'interaction synchrone pendant 45 minutes. À cette occasion, les apprenti-tuteurs dirigent la séance et fournissent aux apprenants les retours nécessaires (*feedback*). En fin de séance, un rapide résumé de la séance est dressé par l'apprenti-tuteur qui en résume les éléments importants. Après la séance, qui a été enregistrée, il est encore demandé aux apprenti-tuteurs d'analyser réflexivement leurs performances en tant que tuteurs en utilisant cet enregistrement, afin de préparer une séance de débriefing avec les autres apprenti-tuteurs et animée par le responsable de la formation. L'ensemble de ces tâches est répété sept semaines de suite de janvier à mars.

Démarche. Il s'agit de fournir un outil à base de m-traces qui permette de soutenir l'ensemble de ces tâches, aussi bien la gestion de plan de séances que les séances synchrones ou les analyses a posteriori de l'activité. On collectera donc les m-traces pendant la séance synchrone, éventuellement enrichies de marqueurs posés par les utilisateurs, qu'on utilisera immédiatement.

- comme soutien à la réflexivité instantanée par leur présentation aux utilisateurs qui en sont à l'origine,
- comme soutien à *l'awareness* mutuel par leur partage instantané.

Les m-traces serviront par ailleurs

- de support de remémoration pour les tuteurs pour faire le résumé de fin de séance,
- de support de réflexivité individuelle sur la pratique pour les tuteurs qui préparent le *debriefing* après la séance,
- pour la construction de bilans, des tuteurs vers les apprenants (bilans consolidés), des apprenants vers les tuteurs si une telle tâche leur est demandée (par exemple pour poser des questions), ou des tuteurs vers le responsable (bilan de pratique professionnelle par exemple). La construction de bilans peut notamment favoriser les activités réflexives et métacognitives.

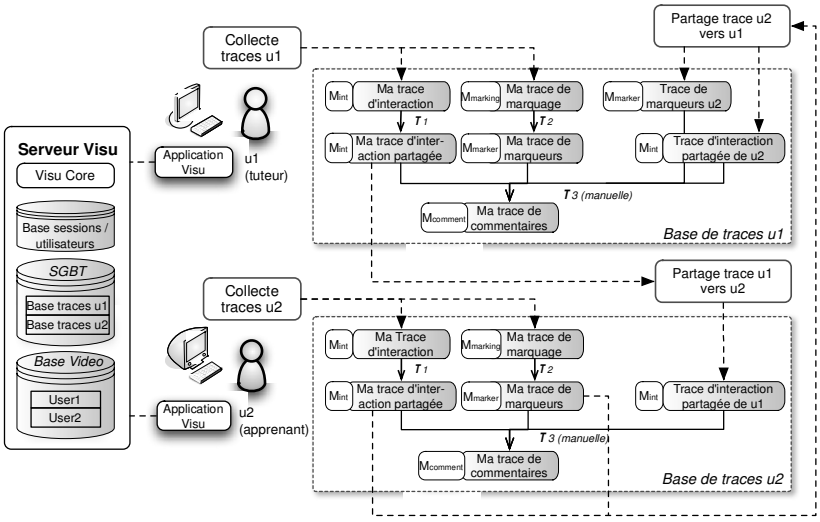

Figure 5

Les différentes m-traces de Visu et leur partage. Un tuteur u1 interagit avec un apprenant u2. La base vidéo stocke les flux enregistrés pendant l'interaction. Chaque utilisateur a sa propre base de traces. La base de m-trace de u1 contient 7 traces, dont 5 proviennent de son activité et 2 ont été partagées par u2 depuis sa propre base. La base de m-trace de u2 contient 6 traces, dont 5 proviennent de son activité et 1 a été partagée par u1 depuis sa propre base. En tant que tuteur, u1 ne partage pas sa m-trace de marqueurs, alors que u2, en tant qu'apprenant, le fait.

L'outil Visu est conçu pour permettre ces usages, il offre la possibilité de créer et de partager des plans de séances, de gérer des utilisateurs, de gérer la planification des séances. Il permet surtout de mener les séances dans le salon synchrone et d'interagir avec les m-traces de celles-ci dans le salon de rétrospection. La *Figure 6* et la *Figure 7* présentent globalement ces deux interfaces.

Traces modélisées considérées. Différents types de m-traces sont considérés dans Visu, illustrés par la *Figure 5*. On distingue ainsi trois types de m-traces modélisées, en plus des enregistrements audio et vidéo de la vidéoconférence (Bétrancourt *et al*., 2011).

Les m-traces d'interaction collectées automatiquement à partir de l'activité de l'utilisateur. Par exemple *Lancer document vidéo*, *Envoyer mot-clé*, *Commencer nouvelle activité*, *Envoyer message chat*, *Agrandir vidéo synchrone*, etc. Sur la *Figure 6*, la m-trace première de l'utilisateur u1 a été filtrée (transformation τ_1) et la m-trace résultante est partagée. Sur la *timeline* horizontale de la *Figure 7*, différents obsels (icône documents) sont présents correspondant à la réception et à l'affichage d'une image par un étudiant, affichés dans la m-trace du tuteur, après leur partage.

Les m-traces de marqueurs contiennent des obsels définis par l'utilisateur similaires aux notes papier prises pendant une présentation orale en vue d'un *feedback*. Le contenu des marqueurs est du texte libre qui peut porter sur n'importe

quel élément de l'activité en cours (par exemple « Utilisation du vocabulaire de la session précédente : bien », « Prononciation », « X ne parle pas beaucoup » ou « Problème microphone »). Les marqueurs peuvent également être posés par les apprenants pour marquer par exemple une incompréhension ou un moment sur lequel ils souhaiteraient revenir. Sur la *Figure 6* la m-trace des marqueurs a été construite à partir de la m-trace de marquage (transformation τ_2), laquelle est une m-trace d'interaction qui décrit l'activité de marquage (créer, effacer, modifier un marqueur). Sur la *timeline* de la *Figure 7*, différents marqueurs déposés par le tuteur sont présents (icône crayon). Dans les choix de conception qui ont été faits, les m-traces de marqueurs de tuteurs sont privées, tandis que celles des étudiants sont partagées.

Les m-traces de commentaires contiennent des obsels commentaires définis par l'utilisateur après l'activité pour décrire celle-ci, par exemple « bon dialogue entre les étudiants », « moment de blanc un peu long » ou « bien dit ». Une m-trace de commentaires est considérée comme une m-trace manuellement transformée, comme une réécriture des m-traces disponibles pour exprimer une interprétation en lien avec les besoins de l'activité en cours (par exemple réflexion sur sa pratique de tuteur en vue d'un débriefing, ou bien description en vue d'un bilan pour l'apprenant). Sur la *timeline* horizontale de la *Figure 8*, la m-trace des commentaires se trouve en haut.

Nous pouvons maintenant présenter les deux interfaces principales de Visu, qui sont le salon d'interaction et le salon de rétrospection.

Salon d'interaction. La *Figure 7* présente le salon au sein duquel se déroule l'interaction synchrone. On peut y voir un gestionnaire de plan de séance (à gauche) l'espace de visioconférence (milieu haut), une fenêtre de chat (droite haut) et une timeline (en bas à droite) qui présente différents obsels, issus soit de la m-trace d'interaction soit de la m-trace des marqueurs sous la forme de crayons, fournissant un support de mémoire et d'*awareness* de l'activité des apprenants aux tuteurs. La *timeline* permet également de poser des marqueurs (bouton *Poser marqueur* sous la vidéo, fenêtre surgissante d'édition du marqueur) afin de prendre des notes sur des moments importants de l'activité afin d'y revenir plus tard, soit en fin de séance, soit dans le salon de rétrospection.

Salon de rétrospection. Après l'interaction, les participants peuvent accéder de façon asynchrone au salon de rétrospection, qui offre un accès à toutes les m-traces collectées durant la séance. La *Figure 8* en donne un aperçu : un lecteur permettant de lire les vidéos de la séance en haut, et une timeline en bas qui permet de contrôler le lecteur. La timeline horizontale présente les obsels issus de la séance groupés par utilisateurs et permet de commenter la m-trace, ce qui revient à ajouter des obsels commentaires dans la m-trace des commentaires (*Comments*), qui au contraire des marqueurs peuvent avoir une durée. L'utilisation du salon de rétrospection est liée à l'interprétation de la m-trace *a posteriori* comme construction d'un ensemble de commentaires. Dans le cas où un tuteur prépare un débriefing, il s'agit ici de favoriser une prise de distance réflexive permettant un développement de la pratique professionnelle. L'interprétation de la m-trace comme structuration peut être poursuivie au-delà des commentaires par la construction d'autres types d'inscriptions, documentaires

cette fois-ci, que nous appelons des bilans dans le cadre d'une activité de documentarisation. Nous ne détaillerons pas ici ces fonctionnalités.

Figure 6

Salon d'interaction de Visu, les m-traces en cours de collecte sont visualisées dans la timeline, il est possible de marquer l'activité.

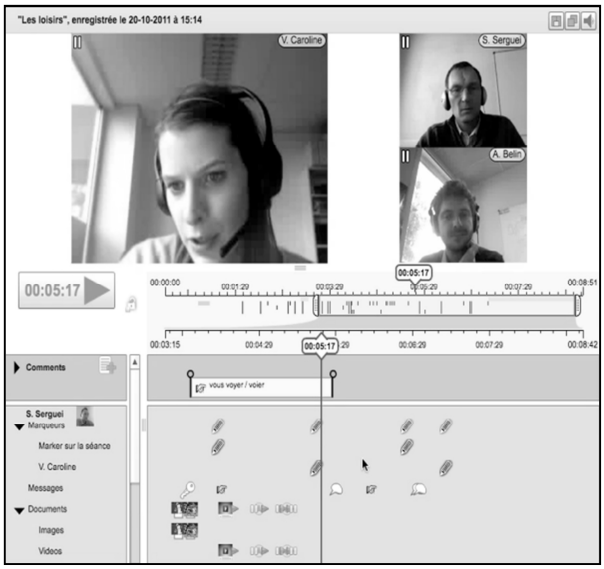

Figure 7

Salon de rétrospection de VISU, les m-traces modélisées sont visualisées dans la timeline.

Le logiciel Visu nous semble intéressant du fait qu'il présente presque toute la gamme des utilisations possibles des m-traces (analyse, réflexivité, utilisation des m-traces pendant et après l'activité, partage de traces, sources de collecte « humaines » et collecte manuelle, etc.). Cependant, les transformations considérées sont relativement simples, ce qui n'est pas le cas dans notre second exemple ABSTRACT.

4.2 - ABSTRACT : un système à base de m-traces pour l'analyse du comportement du conducteur automobile[9]

Les analystes du LESCOT travaillent en équipe pluridisciplinaire pour établir des connaissances sur les comportements de conduite automobile aussi bien en situation écologique qu'en situation de simulation. Les sujets conducteurs sont observés de manière très fine lors des conduites sur les routes ou sur simulateur dans le cadre d'études d'hypothèses ou pour établir les éléments explicatifs de leurs comportements.

Que ce soit dans le cadre de la conduite réelle ou en simulateur, l'environnement de conduite est massivement instrumenté pour récupérer les données liées à l'activité de conduite (capteurs d'actions sur le véhicule, états du véhicule, coordonnées du véhicule, vidéos, eye-tracking, questionnaires, observation directe, données des bus internes du véhicule, etc.). L'activité d'analyse est résumée dans le schéma de la *Figure 8* et le SBTm permet d'accompagner l'ensemble du processus d'analyse comme illustré dans la *Figure 9*. La description du système et les illustrations d'utilisations concrètes sont détaillées dans (Georgeon *et al*., 2006, 2007, 2011)[10].

Les *données* enregistrées pendant la conduite sont *collectées* à l'aide d'un environnement de collecte paramétrable (*BIND)*[11] qui permet de construire les m-traces *premières* à destination du module de gestion de base de m-traces de ABSTRACT.

C'est le SGBTm d'ABSTRACT qui gère les M-traces collectées et propose aux utilisateurs les outils de modélisation interactive permettant de construire des m-traces transformées correspondant à leurs choix d'analyse ou permettant de vérifier plus facilement telle ou telle hypothèse de recherche. Le module SGBTm se charge de gérer les modèles et les instances des m-traces et procure les outils de spécification des transformations. Les m-traces et les modèles sont représentés dans une ontologie formelle tandis que les transformations s'expriment dans le langage de requête associé (SPARQL). La *Figure 10* illustre l'interface permettant au concepteur-analyste de préciser ses règles de transformations avec ces outils.

---

[9] Plateforme d'analyse au Laboratoire Ergonomie et Sciences Cognitives pour les Transports (LESCOT). Le LESCOT est une des équipes de recherche de l'Institut français des sciences et technologies des transports, de l'aménagement et des réseaux (IFSTTAR) http://www.ifsttar.fr/.

[10] Le lecteur intéressé pourra trouver des exemples et illustrations des usages d'Abstract sur http://liris.cnrs.fr/abstract/abstract.html.

[11] *BIND* est un environnement qui permet le prétraitement des données enregistrées au moment de la collecte (traitement du signal, synchronisation des données, videos, tags, etc.), il est la « source de collecte » pour ABSTRACT.

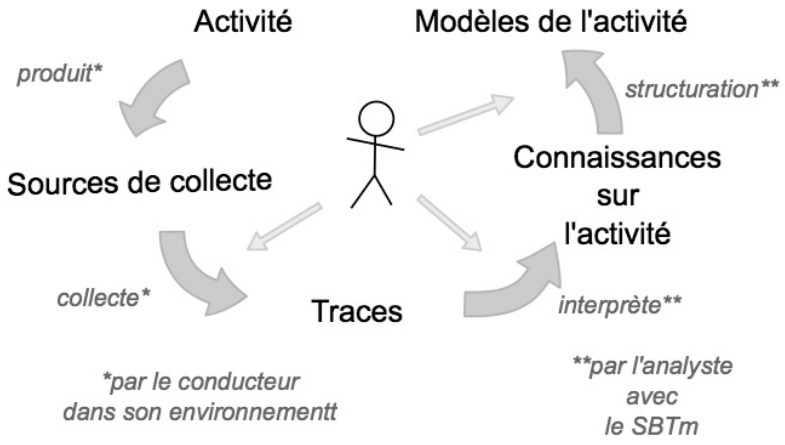

Figure 8

De l'activité aux modèles de l'activité : le processus de l'analyse

Le module de visualisation interactive permet de naviguer dans les M-traces en les synchronisant avec d'autres m-traces possédant les mêmes référentiels temporels (signaux physiques des capteurs, images vidéo, etc.), comme l'illustre la *Figure 10*.

L'analyse des m-traces permet par transformation de sélectionner les séquences d'obsels qui démontrent un type de comportement particulier.

L'activité de construction de connaissances est riche et intense. Le fait d'utiliser un SGBTm permet de capitaliser les choix d'interprétation réalisés en mettant en évidence la *lignée* d'une m-trace première correspondant à une expérimentation et à ses différentes interprétations. Les hypothèses de recherche s'expriment par des branches particulières d'une lignée et peuvent facilement être comparées, discutées et partagées. Le corpus de m-Traces constitué par une lignée est particulièrement utile dans une démarche scientifique puisqu'il intègre explicitement et opérationnellement les transformations et les modèles associés aux différentes interprétations.

L'intégration d'une approche basé sur les m-traces est prometteuse pour la conception d'une plateforme de découverte de connaissances sous la comportementales et une thèse récente (Mathern, 2012) a permis de compléter la plateforme d'analyse par un dispositif de synthèse (synthèse d'automates forme de réseaux de Pétri) sur la base d'un corpus de motifs exprimant les différentes variantes d'un comportement particulier. La synthèse sous forme d'automate permet de vérifier la pertinence de l'hypothèse, en facilite l'explicitation formelle et autorise son intégration dans des simulateurs de conduite.

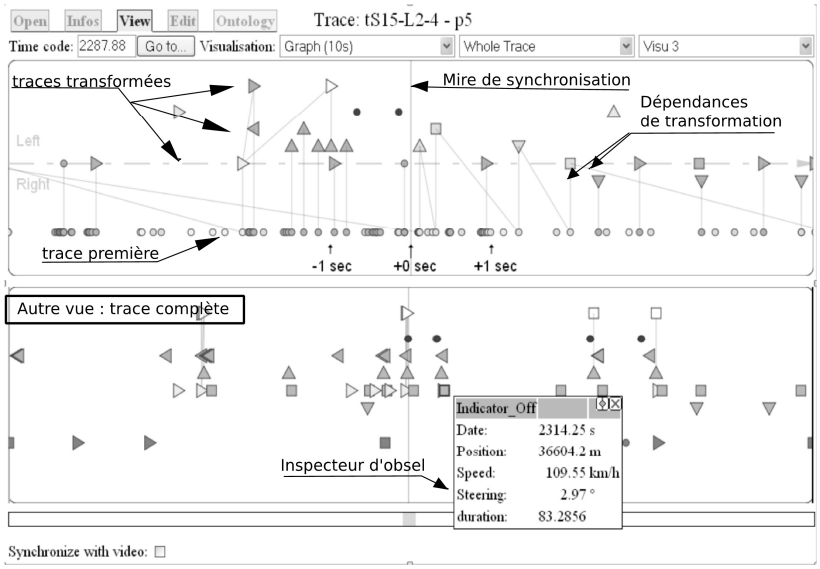

Figure 9

Interface de visualisation des m-traces modélisées sous ABSTRACT : l'interface permet de visualiser plusieurs flux de m-trace et un autre flux temporellement synchrone (vidéo, signaux physiques par exemple). Les échelles de temps ne sont pas obligatoirement les mêmes, la synchronisation se fait sur la mire rouge verticale au centre. La m-trace première figure dans la vue du haut. Les m-traces transformées sont affichées en mettant en relation les obsels calculés avec les obsels source dont ils sont produits par le calcul de transformation. Chaque obsel peut être inspecté, présenté avec l'ensemble de ses attributs.

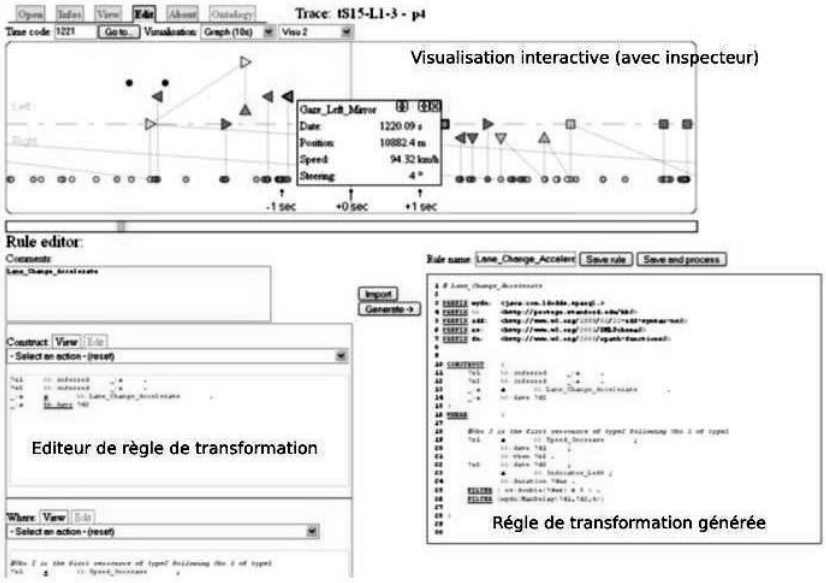

Figure 10

Interface pour gérer les règles de transformation : l'analyste peut adapter ou créer une règle de transformation et relancer la visualisation pour en voir l'effet sur les m-traces transformées.

5 - MODÉLISER DES M-TRACES : UNE PROBLÉMATIQUE PLURIDISCIPLINAIRE ET UN DÉFI INFORMATIQUE

Comme illustré dans la section précédente, nous menons depuis plusieurs années – en parallèle du travail conceptuel et formel autour des systèmes à base de traces modélisées – une exploration de différents systèmes concrets associés à des situations variées. L'objectif de cette exploration est triple : 1) vérifier que les concepts théoriques et leur(s) implantation(s) sont effectivement utiles pour construire ces systèmes, 2) nourrir l'élaboration théorique par les retours issus des applications, mais aussi 3) s'intéresser à la question de la possibilité de définir une *ingénierie des modèles de m-traces et des applications à base de m-traces modélisées*.

5.1 - Modéliser des m-traces pour construire des systèmes à base de m-traces

La modélisation de m-trace est une activité difficile, dont la difficulté apparaît justement parce que les m-traces sont *explicitement* modélisées. Il y a alors lieu d'identifier explicitement, et non incidemment[12], d'une part ce qu'il est nécessaire de conserver de l'activité dans la m-trace première, et d'autre part ce qu'il est utile de construire comme traces(s) transformée(s) en vue de la construction d'un système applicatif particulier. Toute activité de modélisation de m-trace doit *en fait* tenir compte d'un certain nombre de déterminants qu'on peut regrouper en deux grandes catégories.

La première est la *situation tracée et l'application visée*. Que ce soit pour fournir des services d'analyse d'activité à base de traces, d'assistance plus ou moins automatisée ou encore de support à l'activité réflexive, il y a lieu de s'intéresser d'une part à la situation d'activité médiée visée, individuelle ou collective, afin d'en modéliser une partie de la dynamique de réalisation qui sera collectée sous la forme de m-trace ; d'autre part au type d'application ou d'assistance visée. Dans le cadre de l'analyse, on devra par exemple définir précisément l'outil de l'analyste, les différentes possibilités de visualisation et d'interaction avec les m-traces qui lui seront offertes. Dans le cadre de l'assistance ou de la réflexivité, il faudra concevoir une *évolution* de la situation d'activité tracée en imaginant l'impact de la présence d'un assistant logiciel ou d'un support de visualisation réflexif de l'activité (cas où l'activité dont les m-traces sont collectées existe avant toute intervention) ou bien concevoir *à la fois* l'activité principale qui sera tracée et la manière dont celle-ci sera soutenue (cas où le système à base de m-traces est créé *ex nihilo*).

La seconde catégorie concerne les *possibilités techniques de manipulation des traces*. L'activité étant médiée informatiquement, il s'agit de construire un nouveau système technique ou bien de modifier un système existant. De nombreuses contraintes techniques pèsent alors sur les choix de modélisation de traces, telles que la possibilité d'accéder au code source de l'outil (ajout de sondes ou nécessité d'observer depuis l'extérieur) ; la distribution éventuelle

[12] Comme cela peut être fait lorsque la trace n'a pas de statut particulier, n'est qu'une donnée intermédiaire regroupant les informations à conserver en vue de leur utilisation dans une application quelconque.

des machines et logiciels client et serveur qui constituent l'application (contraintes de synchronisation temporelle et de pertes d'information) ; les possibilités de transformation de m-traces (qui mobilisent plus ou moins de puissance de traitement informatique) ; les possibilités liées à la visualisation (puissance de traitement, disponibilité d'affichage) ou encore la nécessité de mener collecte et traitement/transformation de m-traces pour une utilisation immédiate ou différée.

5.2 - De la possibilité d'une ingénierie des traces modélisées

La conception de différents systèmes à base de traces modélisées peut être vue comme une exploration des différents formes que ceux-ci peuvent prendre en fonction des contraintes multiples auxquelles doit faire face le concepteur : contraintes *conceptuelles* liées au modèle théorique (méta-modèle présenté dans la section 0), contraintes *informatiques* liées à l'implantation concrète choisie, contraintes *applicatives* définies par la situation instrumentée et les objectifs d'utilisation des traces. Nous proposons dans la suite de cette section un certain nombre de remarques sur la question de la modélisation de traces au sein d'une ingénierie des traces modélisées et des systèmes à base de traces modélisées.

Dialectique analyse de l'activité / conception d'une application. Construire un SBTm consiste *in fine* à construire une application utilisateur, ce qui signifie qu'une démarche de conception d'application informatique standard peut et doit être appliquée. Une première remarque concerne cependant le fait qu'une telle conception passe nécessairement par une modélisation de m-trace, laquelle consiste en premier lieu en *une analyse explicite de l'activité médiée* par le concepteur. Dans le cadre applicatif d'analyse, ceci se fera dans une démarche de "pré-modélisation" de l'activité pour déterminer ce qui constituera le matériel primaire de l'analyste. Dans le cadre applicatif de l'assistance ou de la réflexivité, la démarche même de conception suppose l'analyse de l'activité à soutenir. On retrouve donc systématiquement une tension entre la conception classique d'une application informatique et la nécessité d'analyser finement une activité et une application existante (cas de l'amélioration d'un système) ou à construire (cas de la conception d'un système à base de m-traces natif). Dans le premier cas, il est frappant de constater que les concepteurs commencent le plus souvent par tracer l'activité en vue de leur propre analyse, avant de pouvoir imaginer en quoi et comment il sera possible de la soutenir. Dans le second cas, on redouble la difficulté de la conception d'application informatique – où il faut définir avec les futurs utilisateurs des scénarios ou des cas d'utilisation du système qui permettront d'en imaginer les fonctionnalités et le fonctionnement – par la nécessité de définir *a priori* la manière dont il sera possible d'assister à l'utilisation de celle-ci, avant toute utilisation réelle.

Analyse de l'activité médiée. Il s'agit donc de mener une analyse de l'activité, c'est-à-dire de la comprendre. Une telle analyse peut se mener de façon *ad hoc* (après tout, les informaticiens se sont peu privés d'imposer leur vision du monde et de l'activité dans leurs applications), ou bien en essayant de tirer parti du vaste corpus interdisciplinaire issu des sciences humaines ou cognitives. Il est alors certain que l'analyse de l'activité médiée est une

problématique pluridisciplinaire qui peut s'appuyer sur des approches théoriques variées. Le *cognitivisme* fournit par exemple une modélisation de l'activité définie en termes de perceptions, traitements et actions. La *théorie de l'activité* vise à thématiser l'utilisation d'artefacts et la composante développementale de l'activité. La *cognition distribuée* s'intéresse à la distribution de traitements entre humain et artefacts numériques, etc. Les conséquences sur la modélisation apparaissent immédiatement. Par exemple une modélisation simple se limitera à considérer des actions de l'utilisateur sur des objets numériques, tous les obsels de la m-trace se révélant au final comme des actions (Yahiaoui *et al.,* 2012). Un autre type de modélisation simple visera à séparer ce qui relève de l'action humain et de l'événement machine, ce qu'on pourra refléter par des règles de nommage des types d'obsel (verbe pour les actions utilisateur, et nom pour les réactions de la machine, *eg.* SauverDocument *vs.* SauvegardeDocument). Une modélisation plus complexe visera à prendre en compte, après avoir soigneusement déterminé un *périmètre de modélisation*, différentes dimensions de l'activité : dimension du *sujet* (sociale), de *l'objet* (inscriptions manipulées) et de *l'outil* (opérations) auxquelles on ajoute la dimension *temporelle* (Laflaquière, 2009). On pourra encore différencier un niveau de *données* (objets informatiques manipulés) d'un niveau de *l'interaction* (interface graphique, boutons) et d'un niveau de *l'intention* utilisateur ou de *l'usage* (Prié, 2011 ; Belin & Prié, 2012).

L'approche de modélisation de l'activité peut être purement *bottom-up* (cas où la m-trace de base est disponible, et qu'il s'agit de tout construire à partir d'elle, ce qui favorise les analyses *ad hoc*), purement *top-down* (cas où l'application d'une théorie analytique définit les éléments considérés comme importants qu'il s'agit de se donner les moyens de détecter automatiquement dans un système donné), ou plus généralement mixte, recherchant un compromis entre les directions données par la théorie et les contraintes techniques particulières du système considéré.

Deux remarques complémentaires peuvent être faites. La première porte sur l'inévitable et nécessaire *oubli* de ce qui n'a pas été tracé, ce qui résulte de contraintes techniques nécessitant de sélectionner parmi les événements systèmes innombrables liés au fonctionnement d'une application ceux qui constitueront la m-trace modélisée dite *première*. Le choix d'écarter des informations doit être assumé par le concepteur, même si celui-ci peut s'offrir éventuellement le luxe de retarder une partie de cet oubli. La seconde remarque concerne le traçage des objets informatiques manipulés par les utilisateurs, qui font partie intégrante de l'application tracée, et sont donc nativement manipulés et enregistrés par celle-ci (qu'on pense par exemple à un document de traitement de texte). Une question importante et difficile pour la modélisation de m-trace est de définir ce que la m-trace doit garder de l'évolution de ces objets eux-mêmes, de l'état de l'environnement informatique tracé aux différents instants considérés. Deux positions extrêmes peuvent être considérées : ne garder que les actions utilisateurs (et être incapable éventuellement d'expliquer une action par l'état de l'objet sur lequel elle portait) ou bien garder tous les états de tous les objets manipulés au cours de l'activité (et se retrouver avec une m-trace potentiellement énorme qui porte les différents états de l'environnement numérique durant l'activité). Il va de soi

qu'une position médiane doit être choisie, qui satisfasse au mieux à la fois les besoins applicatifs et les contraintes techniques de manipulation et de stockage.

Transformations de traces. Comme nous l'avons vu plus haut, une transformation de m-trace est une inférence portant sur les obsels d'une première m-trace qui vise à construire de nouveaux obsels dans une seconde m-trace, et à *automatiser* une interprétation de l'activité. Quelques remarques peuvent ici être d'intérêt. Tout d'abord, on constate pratiquement que les m-traces de bas-niveau sont les plus souvent composées d'événements, c'est-à-dire d'obsels non-duratifs, et que la construction d'obsels duratifs se fait par transformation. Ceci signifie que plusieurs modèles de m-trace de niveaux différents sont le plus souvent conçus *en même temps, dans le même mouvement* que la conception des transformations qui permettent de passer de l'un à l'autre. Ensuite, la visualisation interactive de traces, si elle offre la possibilité de filtrer des obsels, de les réorganiser, d'ajouter des commentaires, peut en elle-même être considérée comme une interprétation temporaire adaptée à la pratique en cours. La conception d'une application à base de m-trace modélisée doit donc prendre en compte cette particularité en ce qui concerne la visualisation interactive comme adaptation ou affinage de l'interprétation. Par ailleurs, une telle possibilité ouvre la voie à une réification de cette interprétation comme transformation automatisée. Enfin, de façon plus générale, les transformations de m-traces peuvent être considérées comme ce qui permet de poursuivre ou de modifier l'interprétation de l'activité telle qu'elle avait été considérée initialement à la conception du système, donc comme un support de re-conception d'application à base de traces, y compris pour le concepteur. Une telle re-conception permet de faire évoluer les modèles de m-traces d'activité considérés sans modifier la collecte, pour adapter l'application à base de m-trace aux usages réels du système (si la modélisation initiale était incomplète), voire à l'évolution même des usages, évolution *induite* par l'introduction des m-traces (Laflaquière, 2009).

Vers une méthodologie ? Au final se pose la question de la possibilité même d'une ingénierie des systèmes à base de traces modélisées, c'est-à-dire d'une ou de plusieurs méthodes permettant de guider la construction de systèmes à base de traces modélisées dans des situations d'activité médiée et pour des objectifs plus ou moins précis. En premier lieu, remarquons que toute conception d'un système informatique résulte d'une application plus ou moins heureuse de méthodes[13], de choix *ad hoc*, de prise en compte des retours des utilisateurs, etc. Il n'y a donc *a priori* pas de raison pour que la conception de systèmes à base de traces modélisées comme systèmes informatiques diffère de ces pratiques. La dimension de modélisation (de données, processus, traitements, objets, etc.) qui est présente dans la majorité les méthodes de conception logicielle est ici complétée par la modélisation de l'activité de l'utilisateur, sous la forme d'une m-trace explicite. Il semble donc qu'il soit possible de définir une méthodologie comme guide d'analyse et d'interprétation de l'activité et de modélisation de celle-ci dans des m-traces et des transformations. Une telle méthodologie viserait *a minima* à attirer

[13] Orienté données et fonctions (MERISE), orienté-objet (Processus Unifié de Développement Logiciel), agile (eXtreme Programming, SCRUM), etc.

l'attention du concepteur tout à la fois sur la dimension analytique (situation, activité, inscriptions manipulées, modification de l'activité due à la présence des traces, etc.), sur la dimension conceptuelle (traces et transformations) et technique (implantation, contraintes de performance, etc.) associée à un SBTm. Au-delà, elle pourrait spécifier différentes étapes normalisées (*workflow*) visant à l'analyse d'une situation, à la spécification d'une application à base de m-trace, à la conception des modèles de m-traces, à la spécification d'une collecte la moins coûteuse possible, à la conception de transformations efficaces, le tout dans une démarche éventuellement itérative. Une telle méthodologie pourrait être accompagnée d'outils dédiés à la modélisation de m-traces et de transformations, ce qui nécessiterait un travail dédié, attentif à la dimension matérielle du support de modélisation et à son influence sur la modélisation elle-même[14]. Par ailleurs, la construction des transformations et des modèles de m-traces résultants peut s'appuyer sur des outils dédiés au cours de phases de fouille (*mining*) dans les m-traces permettant de proposer des interprétations candidates (Cram, 2011 ; Mathern, 2012) comme modélisations de l'activité.

Il va cependant de soi qu'une telle méthodologie devrait être construite à partir de la conception réussie de plusieurs applications à base de traces modélisées, puis validée par son application dans la conception de nouvelles applications. Une telle démarche nécessite alors des applications réelles de grande taille et un support industriel des développements, ce que nous nous attachons à mettre en place dans nos pratiques de projets collaboratifs[15].

6 - CONCLUSION

L'analyse et l'interprétation de tout processus nécessitent une observation *active*, entrelaçant des phases de construction de traces, d'analyse des traces et d'interprétation jusqu'à aboutir à une interprétation satisfaisant l'objectif de l'observation. Dans le cadre d'activités humaines couplées à un environnement informatique, les traces numériques jouent un rôle particulier lié à leur disponibilité à faible coût et au fait que leur nature (numérique) permet de les considérer comme des entités informatiques spécifiques, pouvant faire l'objet d'une part d'interprétation associée à des calculs, et d'autre part d'une gestion explicite (au même titre, par exemple, que les processus ou les fichiers).

Pour accompagner les processus d'interprétation fondés sur des traces numériques, nous proposons donc de considérer un nouveau type d'objet informatique, appelé trace modélisée, associant chaque collection d'éléments observés à un modèle décrivant la sémantique les caractérisant au moment de l'observation. Chaque observation permettant de comprendre d'une manière ou d'une autre le processus observé, nous proposons un mécanisme formel de transformation de trace rendant compte des interprétations faites pour produire une trace transformée bénéficiant de la sémantique issue de l'interprétation.

Un système basé sur l'exploitation de Traces Modélisées est appelé Système à Base de Traces Modélisées (SBTm). De façon à permettre une

[14] Nous avons pu dans notre pratique essayer des outils de modélisation d'ontologies formelles, des traitements de textes ou des tableurs standard pour lesquels nous avons dû inventer des formes de présentation et d'écriture de modèles de traces.

[15] Nous invitons le lecteur à visiter le site http://liris.cnrs.fr/silex et les pages web des auteurs.

gestion aisée et formalisée de ces objets traces modélisées, nous proposons la notion de Système de Gestion de Traces Modélisées (SGBTm), facilitant leur usage pour réaliser les SBTm.

Nous avons illustré cette proposition par deux SBTm différents s'appuyant sur des SGBTm différents. Le premier exemple VISU est un outil de tutorat en ligne à base de traces modélisées et le second exemple ABSTRACT est un système à base de m-traces pour l'analyse du comportement du conducteur automobile. Ces deux exemples de SBTm démontrent l'importance d'une ingénierie de la trace modélisée mobilisée dans une construction dynamique de la connaissance.

La genèse de cette construction de connaissance est rendue partiellement accessible (graphe des transformations de traces) et rend tangible les processus cognitifs à l'œuvre dans la *découverte* des connaissances liée aux processus d'interprétation interactive.

Il va de soi que la proposition d'un nouvel objet informatique nécessite d'importants développements théoriques et le développement de systèmes à base de m-traces modélisées opérationnels démontrant des qualités nouvelles pour accompagner les processus de construction de connaissance.

D'un point de vue théorique, la formalisation des structures de représentation des m-traces modélisées, de la sémantique formelle des transformations (en particulier temporelles), de la nature des objets numériques permanents correspondant à ces objets informatiques constitue un volet de recherche actif et en plein développement.

Du point de vue des applications, la disponibilité de premiers SGBTm en open source[16] et l'amélioration progressive de leurs fonctions et de leurs performances facilite le développement de systèmes à base de traces modélisés. Notre équipe est engagée dans de nombreux projets collaboratifs intégrant l'approche dans des domaines aussi variés que la gestion des connaissances, l'édition collaborative, les EIAH collaboratifs, l'ingénierie collaborative des ontologies, l'assistance à la réutilisation de l'expérience, etc.

Les conditions d'une définition plus précise de la notion d'ingénierie des traces modélisées sont réunies. Il s'agit d'un champ de recherche en soi, important pour permettre la réalisation d'applications de type systèmes à base de traces modélisées, notamment dans le domaine du web sémantique, des réseaux sociaux, de l'apprentissage collaboratif, de l'adaptation et de la personnalisation de l'interaction homme-environnement numérique.

RÉFÉRENCES

Bachimont, B. (2004). *Arts et sciences du numérique : Ingénierie des connaissances et* Bachimont, B. (2004). *Arts et sciences du numérique : Ingénierie des connaissances et critique de la raison computationnelle.* Habilitation à Diriger les Recherches, Université de Technologie de Compiègne

Belin, A. & Prié, Y. (2012). Towards a Model for Describing Appropriation Processes Through the Evolution of Digital Artifacts. *Designing Interactive Systems DIS2012*, 645-654.

[16] http://github.com/ktbs/ktbs.

Bétrancourt, M., Guichon, N. & Prié, Y (2011). Assessing the Use of a Trace-Based Synchronous Tool for Distant Language Tutoring. In *CSCL2011 Conference Proceedings*. Volume I - Long Papers (pp. 478-485). EDITEURS + MAISON EDITION??

Boulicaut, J.-F. & Jeudy, B (2005). *Constraint-Based Data Mining*. Data Mining and Knowledge Discovery Handbook, Springer, (pp. 399-416).

Brachman, R.J. & Anand, T. (1996). The process of Knowledge Discovery in Databases. *Advances in knowledge discovery and data mining* (pp. 37-57). Menlo Park, CA, American Association for Artificial Intelligence.

Cook, J.E., & Wolf, A.L. (1998). Discovering Models of Software Processes from Event-Based Data. *ACM Transactions on Software Engineering and Methodology*, *7(3)*, 215-249. Retrieved from http://dl.acm.org/citation.cfm?id=287001.

Cram, D., Mathern, B. & Mille, A. (2011). A Complete Chronicle Discovery Approach: Application to Activity Analysis. *Expert Systems*, *29(4)*, 321-346. no-no. doi:10.1111/j.1468-0394.2011.00591.x

Deransart, P., Ducassé, M. & Langevine, L. (2002). A Generic Trace Model for Finite Domain Solvers. In B. O'Sullivan (éd.), *Proceedings of the International Workshop on User Interaction in Constraint Satisfaction (UIJCS'02)* (pp. 32-46). Cornell University.

Fayyad, U.M., Piatetsky-Shapiro, G. & Smyth, P. (1996). From Data Mining to Knowledge Discovery: an Overview. In *Advances in knowledge discovery and data mining* (pp. 1-34). Menlo Park, CA, AAAI Press / The MIT Press.

Fisher, C. & Sanderson, P. (1996). Exploratory Sequential Data Analysis: Exploring Continuous Observational Data. *Interactions*, *3(2)*, 25-34.

George, S., Michel, C. & Ollagnier-Beldame, M. Usages réflexifs des traces dans les environnements informatiques pour l'apprentissage humain. *Intellectica*, 59, PP(même numéro)

Georgeon, O., Mille, A. & Bellet, T. (2006). Analyzing Behaviorial Data for Refining Cognitive Models of Operator. In *DEXA '06 Proceedings of the 17th International Conference on Database and Expert Systems Applications* (pp. 588-592). Cracovie, Pologne. EDITEURS + MAISON EDITION??

Georgeon, O., Henning, M. J., Bellet, T. & Mille, A. (2007). Creating Cognitive Models from Activity Analysis: A knowledge Engineering Approach to Car Driver Modeling. In *International Conference on Cognitive Modeling* (pp. 43-48). EDITEURS + MAISON EDITION??

Georgeon, O.L., Mille, A., Bellet, T., Mathern, B & Ritter, F.E. (2012). Supporting Activity Modelling from Activity Rraces. *Expert Systems*, *9(3)*, 261-275.

Kaptelinin, V. (1996). Learning with Artefacts: Integrating Technologies into Activities. In B.A. Nardi (éd.), *Context and Consciousness, Activity Theory and Human Computer Interaction* (pp. 54-68). Cambridge, MA, MIT Press.

Laflaquière, J. (2009) *Conception de système à base de traces numériques pour les environnements informatiques documentaires*. Thèse de doctorat en informatique de l'Université de Technologie de Troyes.

Laflaquière, J., Settouti, L.S., Prié, Y. & Mille, A. (2006). A Trace-Based Framework for Experience Management and Engineering. In B. Gabrys, R.J. Howlett, & L.C. Jain (eds.) *Second International Workshop on Experience Management and Engineering in 10th International Conference on Knowledge-Based Intelligent Information and Engineering Systems 2006* , UK, October 9-11, 2006 (pp. 1171-1178). Berlin-Heidelberg, Springer, ???, vol. 4251.

Mathern, B (2012) *Découverte interactive de connaissances à partir de traces d'activité : Synthèse d'automates pour l'analyse et la modélisation de l'activité de conduite automobile*. Thèse d'informatique. Université Lyon1 Claude Bernard.

Mille, A. & Marty, J.-C. (2009). *Analyse de traces et personnalisation des environnements informatiques pour l'apprentissage humain*. Paris, Hermès Sciences Publications.

Prié, Y. (2011). *Vers une phénoménologie des inscriptions numériques : Dynamique de l'activité et des structures informationnelles dans les systèmes d'interprétation*. Habilitation à Diriger des Recherches en Informatique. Université Lyon1 Claude-Bernard.

Nottingham, M. & Sayre, R. (2005). *The Atom Syndication Format*. RFC 4287, IETF. http://tools.ietf.org/html/rfc4287.

Roussel, N., Tabard, A. & Letondal, C. (2006b). All you Need is Log In. *Proceedings of the WWW2006 Workshop on Logging Traces of Web Activity: The Mechanics of Data Collection*. EDITEURS + MAISON EDITION??

Salembier, P., Theureau, J. & Relieu, M. (2004). *Activités et action/cognition située. @ctivités*, *1(2)*, (pp. 1-216).

Sachan, M., Contractor, D., Faruquie, T.A. & Subramaniam, L.V. (2012). Using Content and Interactions for Discovering Communities in Social Networks. *Proceedings of the 21st International Conference on World Wide Web - WWW '12*, (pp. 331-341). EDITEURS + MAISON EDITION??

Sanderson, P. M. & Fisher, C. (1994). Exploratory Sequential Data Analysis: Foundations. *Hum.-Comput. Interact.*, *9(4)*, 251-317.

Settouti, L.-S., Prié, Y., Champin, P.-A., Marty, J.-C. & Mille, A. (2009). *A Trace-Based Systems Framework : Models, Languages and Semantics*. Retrieved from http://hal.inria.fr/inria-00363260/en/.

Settouti, L.-S. (2011). *Systèmes à base de traces modélisées : modèles et langages pour l'exploitation des traces d'interactions*. PhD Thesis. Computer Science. Université Lyon1.

Song, M., Günther, C.W. & Van der Aalst, W. (2009). Trace Clustering in Process Mining. D. Ardagna, M. Mecella & J. Yang (éd.), *Business Process Management Workshop* (pp. 109-120). Berlin-Heidelberg, Springer, Lecture Notes in Business Information Processing, Vol. 17.

Van der Aalst, W.M.P., Van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G, & Weijters, A.J.M.M. (2003). Workflow Mining: A Survey of Issues and Approaches. *Data & Knowledge Engineering*, *47(2)*, 237-267. doi:10.1016/S0169-023X(03)00066-1.