Diriger la réutilisation de composants à l'aide d'ontologies

C. Reynaud[1] F. Tort[2]

[1] LRI, Université Paris-Sud, URA CNRS 410
[2] ENS, 94235 Cachan.

Bât. 490, 91 405 Orsay cedex
e-mail : cr@lri.lri.fr

Résumé

Cet article porte sur la construction automatique de modèles de résolution de problèmes à partir d'ontologies du domaine spécifiées formellement. Il est plus particulièrement centré sur les aspects réutilisation. L'objectif est de montrer que l'approche adoptée est différente des approches de réutilisation habituelles qui reposent sur la réutilisation de modèles génériques à affiner et à adapter. Au contraire, nous proposons de construire un modèle de raisonnement d'une application par assemblage de composants élémentaires génériques pré-définis. Une organisation de bibliothèques de tels composants est décrite. Cette approche a été implémentée dans ASTREE. Elle est basée sur une mise en correspondance de rôles de connaissances spécifiés dans des tâches et des procédures avec des connaissances d'une ontologie du domaine. Ces mises en correspondance sont facilitées par la syntaxe précise et la sémantique non ambiguë du langage de modélisation de l'ontologie.

Mots clefs

Méthode de résolution de problèmes, réutilisation, ontologie.

Abstract

This paper deals with the automated construction of problem solving methods from formally specified domain ontologies. Only reuse aspects are addressed. The aim is to show that the approach differs from usual reuse based on generic methods which are then refined and adapted. We propose to design a model of reasoning by configuring elementary generic predefined components. An organisation of libraries of such components is described. This approach has been implemented in ASTREE. The selection of adequate components is based on a mapping between knowledge roles in tasks and procedures and pieces of knowledge in domain ontologies. In the approach the precise syntax and the non ambiguous semantics of the modeling language of the domain ontology are crucial because they help to determine which components must be reused and the way to reuse them.

Key Words

Problem-Solving Method, Reuse, Ontology.

Introduction

Depuis quelques années, des chercheurs proposent des méthodes d'acquisition des connaissances basées sur des modèles (SCHREIBER, WIELINGA, DE Hoog, AKKERMANS, & VAN DE VELDE, 1994), (STEELS, 1990), (PUERTA, EGAR, TU & MUSEN, 1992). Ces recherches ont montré que la construction des modèles d'une application est un processus difficile, qu'il doit être facilité par l'emploi d'outils et de méthodes adaptés.

L'une des techniques actuellement étudiées en ingénierie de la connaissance pour faciliter ce processus est la réutilisation de spécifications d'éléments du domaine ou du raisonnement (cf. CommonKADS (SCHREIBER *et al.*, 1994), PROTEGE (PUERTA *et al.*, 1992), VITAL (O'HARA, MOTTA & SHADBOLD, 1994)).

Appliquée aux éléments du domaine, cette technique repose sur la définition d'ontologies décrivant explicitement les éléments d'un domaine. Appliquée aux éléments du raisonnement, elle repose sur l'identification de descriptions abstraites de méthodes de résolution de problèmes. Ces descriptions précisent les spécifications fonctionnelles et les besoins en connaissances du domaine de méthodes. Elles sont génériques et «réutilisables» pour la spécification d'une résolution d'une classe de problèmes, ou tâches. La technique de spécification par réutilisation est un processus de spécialisation. Ainsi, le modèle de raisonnement de l'application est une instance de modèles de résolution de problèmes génériques sélectionnés dans une bibliothèque, affinés, adaptés (FENSEL, 1997) ou encore assemblés (MOLINA & SHAHAR, 1996).

Toutes ces approches supposent qu'en phase d'analyse, le développeur identifie le modèle pré-défini et générique dont le modèle de l'application est une instance ou à partir duquel le modèle de l'application pourra être défini. Nous considérons qu'il s'agit d'une hypothèse forte. Pour certaines applications, cette hypothèse est même irréaliste, notamment lorsque l'expertise disponible décrit des objets du domaine plutôt que des protocoles à suivre pour résoudre un problème. Dans ce cas, il est difficile, voire impossible, de faire émerger une description des buts poursuivis et des méthodes de résolution adaptées dès une première analyse du recueil de l'expertise. A fortiori, il est quasi impossible de commencer la modélisation par le choix d'un modèle générique de résolution de problèmes, ni même par la sélection de composants de modèles à assembler. La modélisation la plus immédiate semble être la modélisation du domaine. Nos travaux de recherche s'inscrivent dans ce cadre. Ils ont conduit à l'élaboration d'une approche et au développement d'un outil, ASTREE, automatisant la construction de modèles de raisonnement dans de telles situations.

Pour ASTREE, construire un modèle de résolution de problèmes signifie identifier des méthodes permettant de réaliser des tâches spécifiées par un utilisateur-développeur. Le processus d'identification que nous avons retenu est basé sur une approche ontologique. L'ontologie de l'expertise, modèle des connaissances sur le domaine, est acquise en premier. Sa représentation n'est pas établie en fonction de méthodes de raisonnement qui ne sont alors pas encore connues. Néanmoins, nous supposons que le contenu et la structure d'une telle ontologie dépendent de la manière dont les connaissances acquises seront utilisées pour raisonner car les experts délivrent des connaissances *adaptées* à leur raisonnement. Ainsi, l'idée de l'approche consiste à exploiter la structure de l'ontologie de l'expertise acquise pour trouver les méthodes adaptées à la réalisation de la tâche de l'application. Le processus d'identification de méthodes pour réaliser des tâches spécifiées par un utilisateur est basée sur une mise en correspondance d'éléments spécifiés dans la tâche à réaliser et d'éléments représentés dans l'ontologie de l'expertise.

L'objectif de ce papier est de montrer que l'approche adoptée dans ASTREE, bien que s'opposant à des approches de réutilisation de méthodes génériques pré-définies, n'exclut pas tout recours à la réutilisation. Au contraire, notre approche contribue de façon originale aux

travaux effectués en réutilisation (i) en proposant un cadre à la réutilisation, (ii) en proposant de réutiliser non pas des méthodes mais des composants qui sont ensuite assemblés, (iii) en proposant une organisation de bibliothèques de composants réutilisables adaptées à notre approche.

Nous adopterons le plan suivant. En section 2, nous présenterons ASTREE, sous deux angles : ses fonctionnalités et son architecture. Nous décrirons ensuite l'ontologie de l'expertise. La section 4 sera consacrée à la réutilisation dans ASTREE. En section 5, nous discutons les résultats et tirons les conclusions de ce travail.

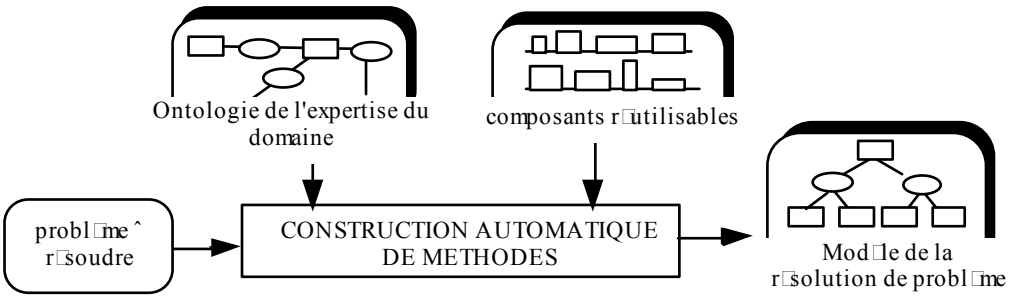

Figure 1 - *Fonctionnalités d'ASTREE*

ASTREE : fonctionnalité et architecture

ASTREE (TORT, 1996) (REYNAUD & TORT , 1997) est un outil d'ingénierie des connaissances, utilisable quelle que soit la méthode de résolution étudiée. Il permet de construire un modèle de résolution pour un problème spécifié en entrée, à partir d'une modélisation des connaissances d'un expert du domaine concerné (cf. fig. 1).

Fonctionnalité

ASTREE fournit un modèle *abstrait* de résolution de problèmes *d'une application*, c'est à dire qu'il réfère directement aux concepts (ou aux classes d'objets) utilisés par les experts du domaine, sans nommer les rôles que ces concepts pourraient jouer dans la résolution.

Une méthode de résolution de problèmes est un modèle abstrait qui décrit comment résoudre un problème. Il est décrit à l'aide des concepts de tâches, sous-tâches et méthodes.

Une tâche est un problème, se définissant par les connaissances de sortie à obtenir à partir de connaissances d'entrée. Une tâche correspond à un problème réel qui se pose à un moment donné au sein d'une organisation.

La méthode est le moyen de transformer une connaissance d'entrée en connaissance de sortie. Elle peut être vue soit comme une procédure pour transformer des ensembles de concepts selon des règles de transformation lorsqu'il s'agit d'une méthode primitive non décomposable en sous-tâches. Elle est alors définie par les spécifications des entrées-sorties d'une tâche et par les définitions des transformations des connaissances du domaine qu'elle suppose. Elle peut aussi être vue comme une décomposition en sous-tâches. Elle est alors définie par l'ensemble des sous-tâches qui la composent, ceci est représenté graphiquement sous forme d'un arbre Tâche-Méthode.

ASTREE n'est pas une boite noire à laquelle on donnerait en entrée une description du problème à résoudre et qui proposerait en sortie, directement, la totalité d'un modèle de résolution de problèmes adéquat. Il est possible de déclencher ASTREE sur un sous-problème du problème général. D'autre part, lorsque le modèle de résolution de problèmes construit par ASTREE consiste en une décomposition en sous-tâches, cette décomposition est présentée progressivement, afin que le développeur puisse l'interrompre ou la modifier en

cours de construction. Ainsi, ASTREE est conçu de façon à travailler en interaction avec le cogniticien.

Architecture

ASTREE comporte deux types d'aide : une aide automatique à la construction du modèle de résolution de problèmes, et une aide pour l'écriture et la manipulation des représentations graphiques et textuelles des connaissances. Ces aides ont été implémentées au sein de différents outils.

Un éditeur d'ontologies facilite la création et la maintenance d'ontologies. Cet outil fournit une interface graphique pour le développeur et l'expert du domaine afin de créer et éditer une ontologie de façon naturelle et agréable.

Un éditeur de tâches permet de définir des tâches. Les données d'entrée-sortie d'une tâche, respectivement nommées contexte et but dans ASTREE, sont des connaissances sur les objets du domaine. Elles sont spécifiées, dans l'éditeur d'ASTREE, à l'aide des concepts de l'ontologie de l'expertise, de déterminants indiquant les structures de données (un, un ensemble de, une liste de, des ensembles de, des listes de), et de contraintes sur les propriétés des objets du domaine. Ces deux dernières informations ne sont pas obligatoires, mais permettent d'obtenir des définitions plus précises. Des facilités de saisie, et des contrôles de saisie sont fournis par l'éditeur.

Sur la figure 2 est présenté un exemple de tâche saisie dans l'éditeur d'ASTREE, issu d'une application d'analyse financière d'entreprise (TORT, 1996). Cette tâche consiste à trouver des événements de la vie de l'entreprise expliquant les agrégats comptables et financiers d'entrée, agrégats ayant tous un niveau jugé significatif[1]. AGREGAT et EVENEMENT sont des concepts de l'ontologie de l'expertise, «niveau» est une propriété de AGREGAT.

nom tâche : diagnostic-financier
contexte : **des** AGREGAT : a
contrainte/contexte : niveau(a) <> moyen
but : **des** EVENEMENT : e
contrainte/but : EXPLIQUE (a,e)

Figure 2 - *Exemple de tâche saisie dans ASTREE*

L'identification de méthodes est réalisée par trois modules dont les fonctionnalités sont les suivantes :

- identifier les parties de l'ontologie de l'expertise concernée par une tâche ;
- identifier différentes transformations de connaissances permettant la réalisation d'une tâche;
- définir une méthode de raisonnement correspondant à une tâche.

Nous décrivons le processus d'identification des méthodes dans le paragraphe qui suit.

L'identification des méthodes

Une méthode est définie par des *entrées* (paramètres) et *sorties* (résultats), correspondant aux entrées-sorties de la tâche qu'elle implémente, des *conditions de déclenchement*, précisant les caractéristiques des objets du domaine sur lesquels elle peut s'appliquer, et un *corps* décrivant le traitement réalisé : soit des sous-tâches et leur contrôle (fig. 3a), soit une procédure primitive (fig. 3b).

[1] La propriété *niveau* de l'agrégat représente ce jugement et peut prendre les valeurs "faible, moyen, fort". Un niveau significatif est un niveau faible ou fort (différent de moyen).

ASTREE utilise un ensemble de mots clé pour désigner les opérateurs de contrôle sur les sous-tâches (puis, et, pour tout). Il connaît un ensemble de procédures primitives décrites par des phrase-type, qu'il instancie à l'aide des concepts de l'ontologie de l'expertise de la tâche étudiée. Dans l'exemple de la figure 3, la phrase type utilisée est : calculer <nom_attribut_calculé> tel que <condition>, où condition est de la forme attribut-opérateur-valeur.

Afin d'identifier une méthode pour une tâche donnée, ASTREE procède :

- soit par la recherche des transformations de connaissances permises par les connaissances du domaine, décrites dans l'ontologie de l'expertise ;
- soit par réutilisation de composants génériques disponibles dans une bibliothèque, en les sélectionnant et les instanciant pour l'application étudiée à l'aide de l'ontologie de l'expertise.

Le premier mode de fonctionnement est décrit dans (REYNAUD & TORT, 1997). Nous nous contentons ci-après d'une description intuitive de ce fonctionnement, ce papier étant consacré au second mode de fonctionnement : la réutilisation de composants (§4).

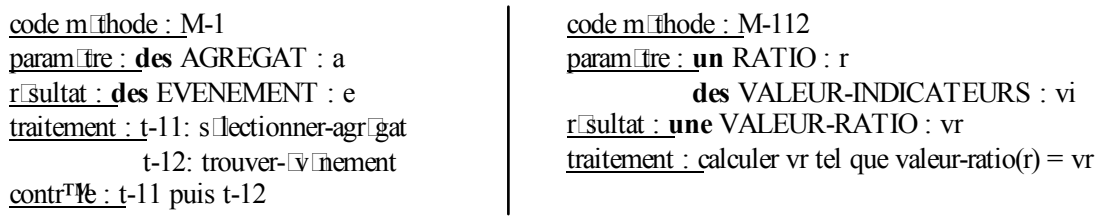

Figure 3 - *Exemples de méthodes proposées dans ASTREE*

ASTREE recherche, dans l'ontologie de l'expertise, quelles *relations* éventuelles existent entre les concepts sur lesquelles portent les connaissances d'entrée et de sortie de la tâche. Ces *relations* correspondent à un ensemble de caractéristiques sur ces concepts et les liens entre eux. Dans la mesure où l'ontologie de l'expertise est décrite à l'aide d'un langage formel, ces relations ont une sémantique précise. ASTREE l'interprète et la traduit en une suite de transformations de connaissances, qu'il présente comme une méthode pour réaliser la tâche.

La relation entre concepts trouvée dans l'ontologie peut être une association binaire entre concepts, ou entre propriétés des concepts, ou encore une composition de telles associations. Lorsqu'il s'agit d'une composition, ASTREE la traduit par autant de sous-tâches que d'associations de la composition. Lorsque c'est une association binaire, selon ses propriétés, définies dans l'ontologie de l'expertise, il propose une procédure primitive. Ce peut être une simple procédure d'«accès» aux données de la base de connaissances ou de la base de faits, une procédure de calcul numérique ou de calcul logique, ou enfin, une procédure appliquant des heuristiques. Selon le cas, ASTREE utilise les informations décrites dans l'ontologie pour construire une description précise de ces procédures (à l'aide de phrases types comme vu précédemment).

Ce mode de fonctionnement n'est pas prioritairement déclenché par ASTREE. En effet, lorsqu'il identifie la tâche étudiée comme une tâche pour laquelle il peut réutiliser des composants génériques, ASTREE privilégie le mécanisme de réutilisation.

L'ontologie de l'expertise

L'ontologie de l'expertise utilisée dans ASTREE est une spécification de la connaissance du domaine. L'expression «ontologie de l'expertise» signifie qu'elle correspond à une conceptualisation de la connaissance du domaine délivrée par un *expert* durant la phase d'élicitation des connaissances.

L'ontologie de l'expertise est une représentation explicite de la structure d'une expertise pour une application particulière. Elle est indépendante de toute méthode de raisonnement mais spécifique à un domaine d'application.

Le langage de représentation des connaissances utilisé est une extension du langage entité-association, un langage simple et bien connu, muni d'une sémantique formelle. Ce langage, emprunté au domaine des systèmes d'information pour modéliser des connaissances, a nécessité quelques adaptations, respectant toutefois sa sémantique. L'utilisation du langage entité-association fournit un cadre pour identifier les principaux concepts à modéliser, pour acquérir des connaissances supplémentaires, pour structurer et organiser les connaissances acquises (REYNAUD, AUSSENAC-GILLES & TORT, 1997). Ce langage est bien connu des développeurs et des experts car il est très utilisé en industrie dans le cadre des développements logiciels classiques. Facilitant la communication, l'ontologie de l'expertise peut ainsi être plus facilement discutée et validée par les experts du domaine. Par ailleurs, le modèle entité-association est aussi reconnu pour la sémantique associée à ses différentes structures qui facilite la modélisation.

L'ontologie de l'expertise est un modèle composé d'un petit nombre de primitives de base : entités, associations, attributs. Ce modèle définit et caractérise des classes d'objets (entités) et identifie les relations autorisées entre les objets, représentée par des relations entre classes (associations). Les objets appartenant à une classe sont appelés instances, ils ne sont pas explicitement représentés. Le modèle identifie également les caractéristiques des instances d'une entité, ou des instances liées par une association, à l'aide d'attributs. Chaque entité et chaque association a un attribut ou un ensemble d'attributs qui l'identifient. A chaque association sont rattachées des cardinalités décrivant le nombre minimum et maximum d'instances d'une des entités associées avec une instance de l'autre entité liée par cette association.

Dans notre approche, les objets et les relations sont ceux utilisés par un expert du domaine lorsqu'il parle de la tâche que doit réaliser l'application. La figure 4 présente une ontologie de l'expertise, dans la représentation graphique fournie par l'éditeur d'ASTREE. Des informations qualifiant les attributs et les associations complètent ce modèle, elles n'apparaissent pas graphiquement, et sont décrites dans des éditeurs textuels. Nous ne les présentons pas ici (Cf. (TORT, 1996)).

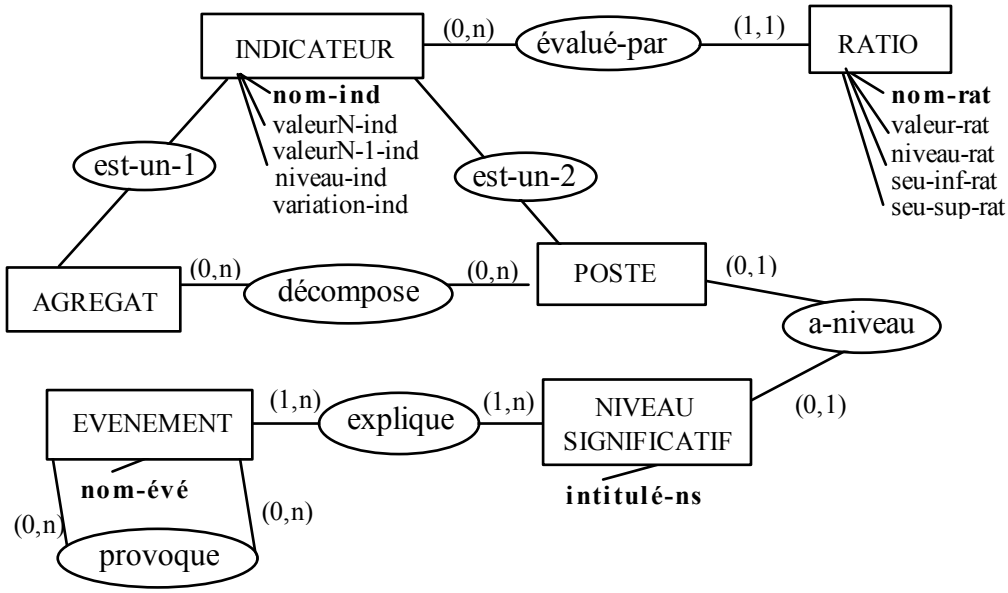

Figure 4 - *Extrait d'une ontologie de l'expertise (application en analyse financière d'entreprise).*

La réutilisation dans ASTREE

Dans ASTREE, la réutilisation est fondée sur la distinction entre les concepts de tâche et de méthode. Il y a deux types de composants réutilisables : des spécifications de tâches et des composants, appelés procédures, qui, combinés et assemblés, permettent de construire des spécifications de méthodes. L'approche que nous proposons exploite donc une bibliothèque de ces deux types de composants, des tâches et des procédures. Nous décrivons tout d'abord le cadre au sein duquel nous avons recours à la réutilisation, puis l'organisation de la bibliothèque et, dans une dernière partie, la façon dont la bibliothèque est utilisée.

Cadre de la réutilisation

Dans notre approche, les techniques de réutilisation sont considérées comme des techniques complémentaires à celles reposant sur l'interprétation de l'ontologie de l'expertise. Elles permettent de pallier certaines «insuffisances», en particulier leur impossibilité à générer des tâches correspondant à des traitements ensemblistes : sélection d'un ou de plusieurs élément(s) dans un ensemble, tri, regroupement. Ce sont des tâches qui peuvent être facilement définies au travers de la structure de leurs entrées/sorties et pour lesquelles il est également possible de décrire des procédures, chaque procédure correspondant à la réalisation d'un type de tâche par application d'un certain critère (critère de sélection, de tri ou de regroupement). En effet, ce sont des tâches de manipulation de données sur lesquelles le sens des données manipulées n'influe pas. Il est donc tout à fait possible et intéressant d'en donner une définition générique et d'en faire des spécifications réutilisables.

Le recours aux techniques de réutilisation se fait principalement dans deux cas de figure. D'une part, lorsque la tâche cible appartient à l'une des classes de tâches pré-définies, les techniques de réutilisation sont exploitées en priorité afin de construire une méthode de résolution de problème adaptée. D'autre part, ces techniques sont utiles lorsque la méthode construite par interprétation des connaissances représentées dans l'ontologie de l'expertise ne permet pas d'atteindre précisément le but de la tâche cible (par exemple, un nombre de valeurs d'attributs obtenu inadapté). Dans ce cas, elles permettent de compléter la méthode de façon à ce que la transformation nécessaire des données soit effectuée.

L'organisation de la bibliothèque de procédures génériques

La bibliothèque de composants réutilisables a été construite en s'inspirant de la classification de schémas d'inférence dressée par Manfred Aben (ABEN, 1994), plus exactement des schémas d'inférence de la catégorie «manipulation d'ensembles». A la différence de M. Aben, nous avons cependant distingué la notion de tâche de la notion de procédure. Enfin, nous avons défini quelles étaient les contraintes de structure portant sur les connaissances du domaine nécessaires à chaque procédure et nous les avons traduites dans le formalisme de l'ontologie de l'expertise, c'est-à-dire en utilisant les primitives du schéma E/A.

Les procédures sont classées selon le type de tâches qu'elles permettent de réaliser. La bibliothèque est donc organisée en deux niveaux, l'un décrivant des classes de tâches, l'autre décrivant des procédures adaptées à la réalisation de tâches appartenant à une certaine classe.

Dans le premier niveau — dans la version actuellement implémentée — quatre classes de tâches sont répertoriées : sélection d'un élément dans un ensemble, sélection d'un sous-ensemble d'éléments dans un ensemble, tri des éléments d'un ensemble, regroupement des éléments d'un ensemble en sous-ensembles. Chaque classe de tâches est définie par un nom et une description syntaxique du contexte et du but des tâches de la classe. Ces classes de tâches correspondant toutes à des opérations ensemblistes, la description précise que les arguments des tâches de la classe peuvent être quelconques mais sont toujours les mêmes en contexte et

en but. Elle spécifie également les indicateurs de structure que les tâches de la classe doivent avoir en contexte et en but.

Exemple : classe TRIER

Les tâches de cette classe consistent à trier les éléments d'un ensemble donné. Ces tâches ont toutes en contexte et en but le même argument. Elles ont un ensemble quelconque d'éléments en contexte et un ensemble ordonné d'éléments en but.

élément : une entité ou une association
IND : «des» ou «des-ensembles-de»
données d'entrée (ou « contexte ») :: IND élément
données de sortie (ou « but ») :: une-liste-de élément

Un second niveau comporte, pour chacune des classes de tâches, une description des différentes procédures génériques permettant de réaliser les opérations ensemblistes auxquelles les classes de tâches correspondent. Chaque procédure correspond à l'application d'un critère particulier. Pour chaque procédure sont définies des contraintes que doivent vérifier les connaissances du domaine manipulées (cf. fig. n°5).

L'exploitation de la bibliothèque

La bibliothèque contenant deux types de composants réutilisables, nous décrivons les modes de réutilisation associés à chacun d'eux.

La réutilisation de la spécification d'une tâche générique

La spécification de tâches fait partie du processus de définition de méthodes de résolution de problèmes complexes. Ces méthodes sont toujours construites par interprétation des connaissances représentées dans l'ontologie de l'expertise. En effet, selon notre approche, les méthodes construites via des mécanismes de réutilisation sont toutes des méthodes primitives, non décomposables en sous-tâches.

La recherche de méthodes adaptées à la réalisation d'une tâche par interprétation des connaissances de l'ontologie de l'expertise, peut ne pas aboutir. Ainsi, lorsque les connaissances de l'ontologie n'autorisent aucune relation entre les différentes données exprimées en entrées-sorties de la tâche, aucune méthode ne sera proposée. Les connaissances de l'ontologie peuvent aussi ne fournir qu'une solution partielle, c'est-à-dire proposer des transformations de connaissances liant les données spécifiées en entrées-sorties de la tâche mais ne permettant pas de réaliser parfaitement la tâche cible (indicateur de structure de la partie but de la tâche non respecté). Dans ce cas, une méthode existe. Les mécanismes d'interprétation de l'ontologie l'ont partiellement spécifiée. La spécification complète nécessite l'ajout d'une sous-tâche effectuant une transformation de connaissances qu'il est alors facile d'identifier. Il ne s'agit pas d'ajouter une transformation reposant sur la sémantique des connaissances en jeu, car sinon l'ontologie de l'expertise aurait été compétente. Il s'agit de l'ajout d'une tâche de manipulation de données indépendante du sens des données manipulées. Des définitions génériques de ces tâches sont stockées en bibliothèque. C'est donc via des techniques de réutilisation de tâches génériques que la méthode en cours de construction sera complètement spécifiée.

Procédures génériques	Caractéristiques des connaissances
P1 *Tri des instances dans l'ordre croissant des valeurs qu'elles prennent pour un attribut donné* Trier les e_i dans l'ordre croissant des valeurs $Att_i(e_i)$	**C1** Il existe un attribut Att_i de l'entité
P4 *Tri des instances dans l'ordre décroissant du nombre d'instances auxquelles chacune est liée par une association donnée* Trier les e_i dans l'ordre décroissant des nombres $\forall$ e_k nb*[$ASS_{ik}(e_i,e_k)$]	**C4** Il existe une association ASS_{ik} de l'entité avec elle-même ou une association ASS_{ik} avec une autre entité _____ **nb exprime le nombre de couples de l'extension de l'association à laquelle appartient l'instance ek*

Figure 5- *Extrait des procédures de tri répertoriées dans la bibliothèque pour la classe TRIER, lorsque l'argument de la tâche est une entité*.

Le processus de réutilisation comporte deux phases : (1) la sélection de la classe de tâches réalisant la manipulation de connaissances adéquate dans la bibliothèque, (2) la spécification de la sous-tâche qui viendra compléter la méthode en cours de construction, par instanciation de la spécification générique réutilisée sur l'application. La sélection est basée sur une mise en correspondance de la transformation structurelle désirée avec celles définies pour chaque tâche générique. L'instanciation consiste à préciser la transformation structurelle (via les indicateurs de structure en partie contexte et but de la nouvelle sous-tâche) et à nommer l'élément sur lequel porte la transformation (attribut spécifié en partie contexte et but de la nouvelle tâche).

Ainsi, la réutilisation de tâches génériques est un mécanisme, parmi d'autres, pour construire des spécifications de méthodes de raisonnement. Les résultats obtenus par interprétation de l'ontologie de l'expertise constituent le contexte au sein duquel les mécanismes de réutilisation sont mis en oeuvre. Ils précisent s'il est pertinent ou non de recourir à la réutilisation de tâches, orientent la sélection dans la bibliothèque de tâches et aident à fournir des définitions de tâches, adaptées à l'application en cours d'analyse, à partir de définitions génériques.

La réutilisation et l'assemblage de procédures génériques

De par sa structure, la bibliothèque de composants offre des procédures élémentaires génériques décrivant des modes de résolution de problèmes, pour chaque classe de tâches. Ces procédures peuvent être instanciées et assemblées pour construire une méthode de raisonnement permettant de réaliser une tâche, instance de la classe de tâches à laquelle les procédures sont liées.

Chaque procédure est définie par les contraintes structurelles que doivent vérifier les connaissances du domaine qu'elle manipule. La sélection d'une procédure consiste alors à étudier l'existence de connaissances dans l'ontologie de l'expertise vérifiant ces contraintes. Cette recherche n'est pas effectuée en aveugle. Les contraintes sont des connaissances liées à l'élément qui est l'objet de la transformation (exemple : existence d'un attribut ou d'une association lorsque les éléments à trier sont des instances d'entités cf. fig. 5). Une telle sélection repose sur l'analyse des connaissances décrites dans l'ontologie de l'expertise. Le

langage de représentation des connaissances dans l'ontologie rend le processus de sélection totalement automatisable. Il conduit à déterminer l'ensemble des procédures applicables, instances d'une ou de plusieurs procédures génériques.

Les procédures applicables sont sélectionnées sur la base de critères syntaxiques car les contraintes associées sont syntaxiques (cf. fig. 5). Toutefois, la sélection effectuée est très précise. Seules les procédures permettant de réaliser une tâche de l'application donnée sont retenues et la définition de la tâche à réaliser précise les éléments de l'application qui sont l'objet de la transformation. Cela conduit à rejeter des procédures «normalement» applicables compte tenu du type des éléments qui sont l'objet de la transformation, lorsque les contraintes associées ne sont justement pas vérifiées pour les éléments considérés (exemple : la procédure P4 décrite figure 6 sera rejetée s'il n'existe pas d'association de l'entité, à laquelle appartiennent les instances considérées, avec elle-même ou avec une autre entité). Ainsi, l'ensemble des procédures applicables, déterminé automatiquement par le système, correspond à l'ensemble des procédures qui ont un sens, étant donnée une tâche précise à réaliser.

Une procédure applicable n'est pas forcément pertinente dans le cadre de l'application. Ainsi, toute procédure applicable devra être validée par le développeur, conjointement avec l'expert. Les procédures validées, qui correspondent en fait à l'application de critères de sélection, de tri ou de regroupement, sont ensuite combinées. C'est la combinaison de l'ensemble de ces procédures qui constitue le corps de la méthode que l'on cherche à construire. Le processus de construction de méthodes est, dans ce cas, un processus coopératif, non totalement automatique. Il comporte trois étapes : (1) la recherche de l'ensemble des procédures (instanciées) applicables, (2) la validation sémantique de ces procédures (et éventuellement le recueil d'informations complémentaires), (3) la configuration de la méthode retenue pour réaliser la tâche, par assemblage ou combinaison de procédures. Les étapes (1) et (3) sont totalement automatisées.

Travaux proches et discussion

Un consensus existe aujourd'hui sur la terminologie concernant les méthodes de résolution de problèmes. Trois composants principaux les caractérisent : leur fonctionnalité appelée aussi compétence, leur description opérationnelle et les contraintes ou hypothèses qui portent sur les connaissances manipulées. Tout le monde s'accorde aussi à reconnaître le besoin de représenter les méthodes de résolution explicitement et indépendamment de leur implémentation. Néanmoins, cette représentation est difficile. Ces dernières années, beaucoup de travaux ont porté sur l'élaboration de techniques ou d'outils permettant de faciliter ce processus. Parmi les techniques étudiées, la réutilisation apparaît comme un des aspects prometteurs des recherches récentes effectuées dans le domaine. Elle conduit d'une part, à étudier le processus de développement de méthodes pouvant être réutilisées, d'autre part, à étudier leur adaptation à une application donnée. Ce papier ne porte que sur le problème de l'adaptation des méthodes.

L'adaptation d'une méthode générique à une application n'est pas simple dans la mesure où la tâche à réaliser dans l'application étudiée et la base de connaissances du domaine de l'application sont tous deux exprimés dans les termes d'un domaine d'application alors que les méthodes réutilisables, issues de bibliothèques, sont en général basées sur des ontologies de méthodes. Ces dernières regroupent des définitions de termes nécessairement indépendantes de tout domaine d'application. Ainsi, adapter une méthode générique à une application est d'abord un problème de mise en correspondance de termes.

Les solutions proposées à ce problème sont variées. Certains travaux proposent des procédures manuelles de mise en correspondance. Dans (BEYS, BENJAMINS & VAN HEIJST, 1996) une ontologie de méthodes définit la structure des rôles manipulés par les méthodes

pouvant être réutilisées. Cette ontologie est utilisée en tant qu'interface pour aider l'ingénieur cogniticien à évaluer les mises en correspondances possibles entre les rôles manipulés par les méthodes et les connaissances du domaine de l'application étudiée.

Adapter une méthode générique à une application peut ne pas être seulement un problème de mise en correspondance de termes. En effet, la méthode de raisonnement de l'application étudiée peut ne pas correspondre à une méthode générique instanciée sur le domaine mais à une *adaptation* d'une telle méthode. Dans (FENSEL, SCHNÖNEGGE, GROENBOOM & WIELINGA, 1996) des adaptateurs font correspondre les termes des modèles de tâches, des modèles de résolution de problèmes ou méthodes et des modèles du domaine. Ils précisent également les hypothèses à faire pour que la fonctionnalité d'une méthode réutilisée corresponde à celle définie dans le modèle de la tâche de l'application étudiée. Certaines de ces hypothèses sont identifiables grâce à l'utilisation de techniques de génie logiciel pour vérifier la consistance des spécifications du SBC en cours de développement (analyse de l'échec de preuves).

D'autres travaux s'intéressent au problème de mise en correspondance pour des méthodes qui sont décrites indépendamment de la tâche qu'elles permettent de réaliser (COEHLO & LAPALME, 1996). De telles méthodes sont a priori davantage réutilisables mais leur réutilisation est plus difficile. Ainsi, ces travaux proposent d'automatiser en partie leur mise en correspondance avec des connaissances d'un domaine donné utilisables. L'approche proposée est entièrement syntaxique et nécessite l'intervention d'un expert pour choisir entre plusieurs mises en correspondances potentielles.

Enfin, si de nombreux travaux semblent considérer que la réutilisation est un problème de mise en correspondance d'un domaine d'application avec des méthodes de résolution, d'autres soulignent que cette mise en correspondance doit aussi être faite entre les méthodes et les sous-tâches qui les composent (STUDER, ERIKSSON, GENNARI, TU, FENSEL & MUSEN, 1996). En effet, la construction d'une méthode de résolution de problèmes d'une application peut être vue comme un problème de configuration de méthodes. Il s'agit alors de sélectionner les sous-méthodes permettant de résoudre les sous-tâches composant la méthode à configurer. Ces mises en correspondance entre méthodes et sous-tâches sont basées sur l'étude des concepts et des relations définissant les rôles des connaissances respectivement manipulés.

Le travail que nous avons présenté aborde la réutilisation d'une manière originale dans la mesure où l'approche évite à l'ingénieur cogniticien d'avoir à sélectionner lui-même une ou des méthodes réutilisables. La construction automatique d'un modèle de résolution de problèmes est par ailleurs un aspect fort intéressant que peu de travaux ont abordée. Elle est basée sur une mise en correspondance des définitions des rôles des connaissances exprimées au sein des tâches et des procédures avec les descriptions des connaissances du domaine contenues dans l'ontologie de l'expertise. Ainsi, l'approche suppose l'existence d'une ontologie du domaine bien définie et formalisée. Ces mises en correspondance pourraient être rapprochées de celles évoquées précédemment à propos des ontologies de méthodes. Toutefois, la réutilisation dans ASTREE est particulière dans la mesure où il ne s'agit pas de réutiliser des méthodes génériques pré-définies ni d'adapter des méthodes sélectionnées mais de combiner des composants génériques plus élémentaires sélectionnés au sein de bibliothèques. Ces composants, appelés procédures, sont adaptés à l'application par un processus de mise en correspondance des rôles manipulés via leur définition. Cette mise en correspondance syntaxique est automatisable alors qu'un processus d'instanciation ne le serait pas.

Les limites de cette approche tiennent à l'importance donnée à l'ontologie du domaine. ASTREE est capable d'apporter une aide utile : (i) si l'ontologie du domaine comporte les connaissances pertinentes pour la tâche étudiée, (ii) si la description d'une tâche se réfère bien aux concepts de l'ontologie et n'adopte pas une vue plus générique se référant à des

rôles des connaissances dans le raisonnement. Ces conditions sont plus faciles à réunir lorsque l'expertise étudiée est plutôt riche en description du domaine. Notre approche suppose que le cogniticien s'astreigne à construire la modélisation à partir des termes et concepts de l'application.

D'autres travaux ont porté sur la construction de méthodes de résolution de problèmes à partir de composants plus élémentaires. Ainsi, Spark-Burn-Firefighter (KLINKER, BHOLA, DALLEMAGNE, MARQUES & MC DERMOTT, 1991) (YOST, KLINKER, LINSTER, MARQUES & MC DERMOTT, 1994) est un atelier permettant l'analyse et l'automatisation d'activités du monde réel. Les auteurs soulignent l'importance des caractéristiques spécifiques à chaque situation de travail qui la rendent unique et qui ne facilitent pas son interprétation en un modèle générique abstrait. Dans ce cadre, l'approche de modélisation préconisée exploite une description du problème réel réalisée par l'utilisateur afin de l'aider à sélectionner dans une bibliothèque les mécanismes permettant au mieux de le réaliser automatiquement. L'intérêt de réutiliser les mécanismes de la bibliothèque de SPARK est de réutiliser, en fait, les morceaux de programmes auxquels ils sont associés. Ces travaux privilégient un indexage dynamique des mécanismes, assuré par l'Active Glossary, tenant compte du vocabulaire du développeur. Les processus de mise en correspondance sont ici purement terminologiques.

Conclusion

L'approche adoptée dans ASTREE vise la construction automatisée de méthodes de résolution de problèmes à partir d'ontologies du domaine spécifiées formellement. Elle offre ainsi un éclairage nouveau sur le processus de construction des méthodes de résolution de problèmes d'une application. Le papier, centré sur les aspects réutilisation, décrit un aspect de ce processus. La méthode intègre par ailleurs des mécanismes d'assemblage de composants élémentaires génériques stockés dans des bibliothèques. Elle apporte ainsi des réponses à deux types de problèmes qui se posent en réutilisation : (i) le problème de l'indexation ou comment retrouver les composants réutilisables, (ii) le problème de la configuration ou comment adapter les composants réutilisés. Ces deux problèmes sont traités via la mise en correspondance des définitions des rôles des connaissances au sein des tâches et des procédures avec les définitions des connaissances du domaine représentées au sein de l'ontologie de l'expertise. Ces mises en correspondance sont facilitées par la syntaxe précise et la sémantique non ambiguë du langage de modélisation de l'ontologie de l'expertise.

L'approche a été testée sur une application de taille réelle, en analyse financière. Ce travail a montré que l'approche fournit un bon guide pour l'analyse d'une expertise, l'extraction et la modélisation des connaissances du domaine et des connaissances de résolution.

Références

ABEN M. (1994) A library of Inference Schemata, Appendix B. In J. BREUKER & W. VAN DE VELDE, Eds. *CommonKADS library for expertise modeling*.

BEYS P., BENJAMINS V.R. & VAN HEIJST G. (1996) Remedying the reusability-usability tradeoff for problem-solving methods. In GAINES B.R. & MUSEN M. A., Eds. *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*. Banff, Alberta, Canada. 2-1, 2-20.

COEHLO E.& LAPALME G. (1996) Describing Reusable Problem-Solving Methods With a Method Ontology. In GAINES B.R. & MUSEN M. A., Eds. *Proceedings of the 10th Banff*

Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Alberta, Canada. 3-1, 3-20.

FENSEL D. (1997) The Tower-of-Adapter Method for Developing and Reusing Problem-Solving Methods. *In Proceedings of the 10th European Workshop EKAW'97*. Sant Feliu de Guixols, Catalonia, Spain. Springer-Verlag. 97-112.

FENSEL D., SCHNÖNEGGE A, GROENBOOM R. & WIELINGA B. (1996) Specification and Verification of Knowledge-Based Systems. In GAINES, B.R. & MUSEN, M. A., Eds. *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*. Banff, Alberta, Canada.

KLINKER G., BHOLA C., DALLEMAGNE G., MARQUES D. & MC DERMOTT J. (1991) Usable and reusable programming constructs. *Knowledge Acquisition*. 3, 117-136.

MOLINA M. & SHAHAR Y. (1996) Problem-solving Method Reuse and Assembly: From Clinical Monitoring to traffic Control. In GAINES B.R. & MUSEN M. A., Eds. *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Alberta, Canada.

O'HARA K., MOTTA E. & SHADBOLD N. (1994) Grounding GDMs: A structured case study. *International Journal of Human-Computer Studies*, 40, 315-347.

PUERTA A., EGAR J., TU S. & MUSEN M. (1992) A multiple-method shell for the automatic generation of knowledge acquisition tools. *Knowledge Acquisition*, 4, 315-347.

REYNAUD C. & TORT F. (1997) Using explicit ontologies to create problem-solving methods. *International Journal of Human-Computer Studies*, 46, 339-364.

REYNAUD C., AUSSENAC-GILLES N. & TORT F. (1997) A support to domain knowledge modelling: a case study. *Proceedings of the 7th European-Japanese Conference on Information Modelling and Knowledge Base*. Toulouse. 27-30.

SCHREIBER A. T., WIELINGA B. J., De HOOG R., AKKERMANS J.M. & VAN DE VELDE W.(1994) CommonKADS: A Comprehensive methodology for KBS Development. *IEEE Expert*, 9(6), 28-37.

STEELS L. (1990) Components of Expertise. *AI Magazine*. 11(2), 28-49.

STUDER R., ERIKSSON H., GENNARI J., TU S., FENSEL D., & MUSEN M. (1996) Ontologies and the configuration of problem-solving methods. In GAINES, B.R. & MUSEN, M. A., Eds. *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*. Banff, Alberta, Canada. 11-1, 11-20.

TORT F. (1996) ASTREE : automatiser l'aide à la construction d'un modèle de résolution de problèmes. *Thèse de l'université Paris-Sud*. N°4593.

YOST G.R., KLINKER G., LINSTER M., MARQUES D. & MC DERMOTT J. (1994) The SBF Framework, 1989-1994: From applications to Workplaces. *Proceedings of EKAW'94*. Springer-Verlag. 318-339.