Des Artefacts aux Programmes

Gilles Kassel — Pascal Lando — Anne Lapujade — Frédéric Fürst

Laboratoire de Recherche en Informatique d'Amiens
Université de Picardie Jules Verne et CNRS (FRE 2733)
33 rue Saint-Leu, 80039 Amiens Cedex 1
{gilles.kassel, pascal.lando, anne.lapujade, frederic.furst}@u-picardie.fr

RÉSUMÉ. *Le domaine des programmes informatiques fait l'objet depuis plusieurs années d'investigations ontologiques, l'enjeu principal étant de disposer de descriptions formelles des programmes pour mieux maîtriser leur conception et leur utilisation. Dans la lignée de ces efforts, nous développons une ontologie « noyau » des programmes (COPS) rassemblant des concepts et relations centraux de ce domaine. Pour le développement de COPS, notre démarche consiste à introduire de façon intermédiaire des concepts situés à des niveaux d'abstraction plus élevés, comme ceux d'*expression*, de* langage *et de* fonction*, pour aider à définir, par spécialisation, des concepts plus spécifiques comme celui de* programme*. Suivant cette démarche, nous proposons dans cet article une double contribution. Nous définissons tout d'abord les bases d'une ontologie des artefacts apportant un ensemble de primitives conceptuelles pour rendre compte de l'artificialité et de la fonctionnalité d'entités. Nous montrons ensuite l'apport de ces primitives pour rendre compte du concept de programme informatique et aider à structurer une ontologie des programmes.*

ABSTRACT. *For some years, ontology research has investigated the field of computer programs, the main objective being to have at one's disposal formal descriptions of the programs so as to master their design and use. In line with these efforts, we currently develop a "core" ontology of programs (COPS) encompassing main concepts and relations for this field. For the development of COPS, our approach consists in introducing concepts situated at higher levels of abstraction, such as the concepts of* expression, language *and* function*, to help defining, by specialization, more specific concepts such as the concept of* program*. In line with this approach, we propose in this paper a twofold contribution. We first define the basis of an ontology of artefacts providing a set of conceptual primitives that allow to take into account the artificiality and functionality of entities. We then show the contribution of these primitives to render an account of the concept of computer program and help structuring an ontology of programs.*

MOTS-CLÉS : *Ontologies formelles et de haut niveau (foundational ontologies), Ontologies générales de domaines (ontologies noyau), Ontologie des artefacts, Ontologie des programmes informatiques.*

KEYWORDS: *Formal and top-level ontologies (foundational ontologies), General ontologies of domains (core ontologies), Ontology of artefacts, Ontology of computer programs.*

1. Introduction

Depuis une dizaine d'années, le domaine des programmes informatiques fait l'objet d'un nombre croissant d'investigations ontologiques visant, selon les disciplines, différents objectifs : en philosophie de l'informatique, l'objectif est d'accéder à une meilleure connaissance de la nature des programmes (Eden et Turner, 2007) ; en génie logiciel, la description formelle des programmes vise à aider leurs développeurs à les concevoir et les maintenir (Welty, 1995)(Oberle *et al.*, 2006) ou, pour les services sur le Web, à automatiser leur découverte et leur combinaison (Roman *et al.*, 2005).

Dans la lignée de ces efforts, l'équipe Ingénierie des Connaissances du LaRIA conçoit une ontologie générale (ou « noyau » (Gangemi et Borgo, 2004)) du domaine des programmes et des logiciels, rassemblant des concepts et relations centraux de ce domaine (Lando *et al.*, 2007a, 2007b). Cette ontologie, nommée COPS (pour : Core Ontology of Programs and Software), a vocation à aider à conceptualiser des sous-domaines de programmes pour le développement d'ontologies d'application. Ainsi, dans le cadre du projet NeuroLOG[1] visant à développer une plate-forme logicielle distribuée pour aider la communauté des chercheurs en neuro-imagerie à mutualiser des images et des programmes de traitement d'images, l'ontologie COPS est utilisée pour conceptualiser le sous-domaine des outils de traitement d'images (Temal *et al.*, 2006).

Pour concevoir COPS, notre démarche consiste à définir les concepts et relations de l'ontologie par rapport à des primitives conceptuelles plus abstraites. Notamment, la définition commune des programmes informatiques (des expressions d'instructions dans un langage de programmation, dont la fonction est de permettre à un ordinateur de réaliser un traitement de l'information) fait référence aux concepts plus abstraits d'*expression*, de *langage* et de *fonction*. Pour aider à structurer l'ensemble de ces concepts, l'ontologie de haut niveau DOLCE (pour : Descriptive Ontology for Linguistic and Cognitive Engineering) est utilisée. Cette ontologie comporte un ensemble restreint de concepts et de relations, choisis et organisés selon des principes philosophiques clairement exposés et qui sont indépendants de domaines particuliers (Masolo *et al.*, 2003).

Nous constatons toutefois que les entités conceptuelles de DOLCE sont situées à un niveau d'abstraction trop élevé pour permettre de définir directement celles de COPS. DOLCE comporte des concepts très abstraits comme ceux d'*objet*, de *substance*, d'*évènement* et de *qualité*, mais ne propose pas de concepts plus spécifiques comme ceux d'*expression*, de *langage* ou de *fonction*. Ces toutes dernières années, des extensions ont été apportées à DOLCE en vue d'introduire des concepts abstraits de niveau intermédiaire. C'est ainsi que le LaRIA a défini l'ontologie I&DA (pour : Information and Discourse Acts) dans le domaine de la sémiotique, qui introduit notamment le concept d'*expression* (Fortier et Kassel,

[1] http://neurolog.polytech.unice.fr

2004). En revanche, à l'exception d'une contribution très récente de (Borgo et Vieu, 2006), aucune proposition n'a été faite pour les concepts de *fonction* et d'*artefact*, qui se révèlent pourtant centraux pour définir le concept de *programme*.

Le travail présenté dans cet article entend répondre à un tel manque. L'objet de l'article est de proposer une ontologie noyau du domaine des artefacts, utilisant comme cadre de référence les ontologies DOLCE et I&DA, et de montrer l'aide apportée par cette ontologie pour définir le concept de *programme informatique*[2].

L'article suit le plan général suivant : nous menons tout d'abord une analyse informelle de la notion philosophique d'*artefact*, qui nous conduit à distinguer entre des entités *artificielles* et des entités *fonctionnelles* et à définir une notion de *fonction* ; nous présentons ensuite une vue d'ensemble de notre cadre ontologique de référence (DOLCE + I&DA) et montrons comment nous l'avons étendu pour définir une ontologie formelle et multi-composants des artefacts ; nous montrons alors l'apport de cette ontologie pour définir le concept de programme informatique ; avant de conclure, nous comparons notre ontologie des artefacts avec d'autres travaux proches.

2. Vers une notion d'artefact

2.1 Entités artificielles *et* fonctionnelles

Les dictionnaires courants recensent deux notions communes principales pour le terme « artefact » :

- Un objet fait par l'homme (ex : une arme, un ornement)
- Un résultat expérimental étranger au phénomène naturel étudié et qui est dû au cadre expérimental même (ex : une ombre sur une image de poumons se révélant être due à la technique d'imagerie médicale utilisée).

Les deux notions font référence à un produit ou effet de la technique, ce qui correspond à l'étymologie du mot « arte-fact » : quelque chose de fait (*factum*) par art ou adresse (*arte*). Elles se distinguent suivant que l'objet produit correspond, ou non, à un résultat visé.

En philosophie, un artefact est couramment défini comme une *entité intentionnellement faite ou produite pour une certaine raison* (Hilpinen, 2004). La notion philosophique se veut plus précise que la notion commune, en ajoutant que l'entité est « intentionnellement » produite, et qu'elle est produite « pour une certaine raison ». Nous comptons montrer que ces deux propriétés, considérées isolément, conduisent à distinguer deux catégories d'entités, des entités *artificielles*

[2] De fait, l'ajout de cette ontologie conduit à une version plus complète de COPS que celle présentée dans (Lando *et al.*, 2007a, 2007b).

et des entités *fonctionnelles*. Leur prise en compte conjointe aboutit à une double caractérisation des *artefacts*.

Le terme « intentionnel », dans le contexte de cette définition[3], implique que l'artefact corresponde à un résultat visé par son créateur. La notion philosophique d'artefact (que nous adopterons) épouse ainsi la première notion commune d' « artefact-visé ». Comme nous l'avons vu, la seconde notion commune fait référence, a contrario, à une entité créée non intentionnellement, autrement dit qui n'est pas visée. L'existence de telles entités amène à considérer une classe des entités *artificielles* (= entités produites comme résultat d'une activité humaine), qui est plus large que la classe des artefacts.

La *raison* (*purpose*, *goal*) pour laquelle l'objet est produit fait pour sa part référence à l'utilité attendue de l'objet, ou sa *fonction* : un objet est ainsi conçu pour *servir à faire quelque chose* (ex : la fonction d'un marteau est *de servir à frapper fort et à coups redoublés*). Il convient de noter que, formulée en ces termes : « aider à faire quelque chose », la *fonction* de l'artefact est distincte de l'intention de sa création. Du reste, comme l'ont rappelé récemment (Borgo et Vieu, 2006), il est possible d'évoquer la(es) fonction(s) d'une entité, que celle-ci soit artificielle ou naturelle[4] :

> "Consider the agent's intentions underlying the creation of an artefact. There are two aspects: the intention to get an entity for some purpose or use, and the intention to physically modify or process some pre-existing entity to "produce" the artefact. We focus here on the first one only because we take the second aspect, i.e., being an *artificial* entity, not to be necessary for artefacts: a pebble can make a paper-weight when taken to an office, and a fallen trunk a bench."

Si nous définissons une entité *fonctionnelle* comme une entité *à laquelle est attribuée une fonction*, les exemples du galet (pebble) et du tronc d'arbre, servant respectivement de presse-papier et de banc, montrent qu'être une entité artificielle n'est pas nécessaire pour une entité fonctionnelle. Les deux propriétés sont donc largement indépendantes. Pour souligner cette indépendance, précisons qu'une entité artificielle, possédant déjà une fonction, peut se voir attribuer une autre fonction. Une utilisation détournée d'un artefact, par exemple le fait de se servir d'une agrafeuse ou d'un taille-crayons comme presse-papier, illustre ce cas de figure.

Revenons-en à l'artefact. La définition philosophique en fait une entité qui est tout à la fois artificielle et fonctionnelle. Nous venons de voir que ces deux

---

[3] Depuis Searle, les philosophes contemporains distinguent schématiquement deux types d'intention, que (Pacherie, 2000), reprenant la terminologie de Searle, nomme « intention préalable » et « intention en action ». L'*intention préalable* suppose une planification de l'action et une représentation de son but, tandis que l'*intention en action* relève du guidage et du contrôle de l'action tout au long de son exécution (Pacherie, 2000). Dans ce contexte, le terme « intentionnel » est à prendre au sens d'une intention préalable.

[4] Dans l'article, pour éviter tout contresens, nous reproduirons les citations en anglais sans les traduire.

propriétés ne vont pas de pair. Dans le cas de l'artefact, l'objet étant intentionnellement produit, nous pouvons considérer que cette production s'accompagne d'un acte mental d'attribution d'une fonction.

Pour continuer à caractériser les notions en jeu, nous choisissons de nous focaliser sur les notions de *fonction* et d'*attribution d'une fonction à une entité*. Précisons que, notre projet étant à visée ontologique et non épistémologique, nous laissons de côté l'explication du mécanisme par lequel un agent attribue une fonction à une entité, pour nous intéresser à la nature même de ces notions.

2.2. Compétences *et* Fonctions

La formulation de la notion de fonction que nous retenons : « aider à faire quelque chose », invite à inscrire cette notion dans une théorie de l'action. À cette fin, nous convoquons préalablement deux notions fondamentales, celle de *connaissance* (ou plus spécifiquement de *compétence*) et celle d'*agent*, pour lesquelles nous considérons les définitions courantes suivantes :

- Une *connaissance* est la capacité à réaliser une action (pour atteindre un but).
- Un *agent* est une entité ayant la capacité à réaliser une action (pour atteindre un but).

Le terme « capacité » indique un potentiel qui est exploité lors de la réalisation d'une action et rend cette dernière possible. La distinction *capacité* vs *action* rejoint la distinction de Chomsky entre *compétence* et *performance* linguistiques[5]. En tant que potentiel, la compétence existe indépendamment de l'action à laquelle elle se rapporte et indépendamment du fait que cette action réussisse ou échoue, donc indépendamment du fait que le résultat visé existe ou pas[6]. À noter que l'action à laquelle il est fait référence correspond à un type d'action plutôt qu'à une action individuelle, la capacité à réaliser une action supposant la capacité à répéter cette action. La compétence est possédée par une entité, l'*agent*. Toujours du fait qu'il s'agit d'un potentiel, la compétence est attribuée à l'agent (par un agent ou une collection d'agents), ce qui revient à considérer que l'agent « possède » ce potentiel. Ce potentiel pouvant être acquis puis perdu, il doit être distingué d'une *qualité* qui, à l'instar de la masse ou de la couleur d'un objet physique, est inhérente à l'entité.

---

[5] Du reste, le terme « connaissance » étant trop général, car pouvant s'appliquer à une connaissance conceptuelle, nous lui préfèrerons par la suite le terme « compétence » entendu au sens de Chomsky et lié à la réalisation d'actions.

[6] Nous avons placé l'expression « pour atteindre un but » entre parenthèses car, suivant la distinction des deux types d'intention que nous avons rappelée supra, nous considérons qu'une action n'est pas nécessairement précédée d'une intention préalable. Ainsi, par exemple, lors d'un trajet en voiture, un conducteur peut « machinalement » débrayer ou changer de vitesse sans avoir eu au préalable l'intention de réaliser ces actions.

Concernant la notion d'*agent* : le terme « agent » est couramment utilisé selon deux sens principaux, celui d'un rôle temporel joué par une entité lors d'une action individuelle – ce rôle est appelé dans la littérature « rôle de participation » ou « rôle thématique » (Sowa, 2000) -, et celui, que nous venons d'introduire, d'une entité à laquelle quelqu'un attribue temporellement une compétence. Pour distinguer ces deux notions, nous emploierons dorénavant les termes « agent » (pour le rôle) et « agentive » (pour le statut). Ces deux notions sont liées : un *agentive* est une entité à laquelle quelqu'un attribue un statut, la capacité à pouvoir jouer le rôle d'*agent* dans des actions.

Par analogie, nous proposons pour les notions de *fonction* et d'*entité fonctionnelle*, d'adopter les définitions suivantes :

- Une *fonction* est la capacité à permettre de réaliser une action (pour atteindre un but).
- Une *entité fonctionnelle* est une entité ayant la capacité à permettre de réaliser une action (pour atteindre un but).

Le terme « fonction » est largement polysémique et il serait illusoire de vouloir rechercher un concept qui rende compte de l'ensemble de ses acceptions. L'objectif que nous visons est de proposer une notion qui contribue à la caractérisation des artefacts. Nous noterons la proximité de cette notion de fonction avec celle proposée en ingénierie par (Chandrasekaran et Josephson, 2000) de *function as effect* (le terme « effet » faisant référence ici à un « faire » et au changement dans l'environnement extérieur qui en résulte) :

> "All the meanings for the term "function" arise from the idea of a machine, system or a person *doing* something or having a property that is *intended* or *desired* by someone, or deemed as appropriate from someone's point of view. Thus the ontology of function starts with the ontology of behavior, but it is distinguished by the fact that some agent regards it as desirable or intends the behaviour. All the other terms – structure, behaviour, causal models – are neutral with respect to intent […] Thus a central meaning of function is *function as (desired) effect*."

Selon ces auteurs, l'idée même de fonction repose sur le fait qu'une entité fasse quelque chose qui est attendu par une personne, autrement dit qui permette à cette personne de réaliser une action. Une fonction repose ainsi sur un comportement qu'une entité est capable de manifester.

Une telle notion de comportement fonde également notre notion de fonction. Pour le mettre en évidence, nous proposons d'assimiler un comportement à un couple (rôle, action), autrement dit à *une manière de participer à une action*. Défini de la sorte, nous pouvons voir que la distinction entre nos notions de fonction et de compétence repose sur une distinction entre deux comportements : capacité à participer en tant qu'*agent* pour la *compétence* ; capacité à participer en tant qu'*instrument* pour la *fonction*. Le rôle d'instrument, que traduit la présence du verbe « permettre » dans la définition, est à prendre au sens large d'*apporter une aide à un agent pour la réalisation d'une action*.

Concernant la notion d'*entité fonctionnelle* : de même que nous avons distingué supra le rôle temporel de participation d'*agent* et le statut d'*agentive* attribué à une entité, nous retrouvons le rôle d'*instrument* et le statut d'*entité fonctionnelle*. Sur cette distinction également, on peut noter une convergence avec l'analyse de (Chandrasekaran et Josephson, 2000) :

> "Suppose a person finds a ledge to sit on after a tiring hike. In this case, not only is it the case that the ledge plays a *role* of a chair, but that it serves the *function* of a chair, because this role is intended by the person. This definition of function as role applies to devices explicitly designed for a function, for objects that are just used by someone for a certain function (such as the use of a ledge for sitting on), as well as to biological objects. […] *Functions are roles + intention*."

En conclusion, résumons le cadre de modélisation auquel nous parvenons au terme de cette analyse. Suivant la définition philosophique de la notion d'artefact, nous assimilons les artefacts à des entités artificielles intentionnellement produites (au sens de l'intention préalable) et auxquelles une fonction est attribuée. Par contre, nous considérons les propriétés d'artificialité et de fonctionnalité comme indépendantes des artefacts. Ainsi, nous admettons l'existence à la fois d'entités artificielles qui ne sont pas des artefacts (celles qui ne sont pas produites intentionnellement) et d'entités fonctionnelles qui ne sont pas des artefacts (les entités naturelles, notamment)[7]. Dans la suite de l'article, nous allons formaliser ce cadre.

3. Cadre ontologique de référence

3.1. *DOLCE*

DOLCE (Masolo *et al.*, 2003) est une ontologie « fondationnelle », en ce sens qu'elle comporte des concepts abstraits ayant vocation à généraliser l'ensemble des concepts que l'on peut rencontrer dans les différents domaines de connaissances. Suivant des principes philosophiquement fondés, le domaine de DOLCE – les *Particulars* (PT)[8] – est partitionné en quatre sous-domaines (cf. Fig. 1).

- Les *Endurants* (ED) sont des entités « endurant dans le temps » (ex. : le présent article). Parmi les *Endurants* sont distingués les *Physical Objects* (POB) et les *Non-Physical Objects* (NPOB), les premiers étant les seuls à posséder des qualités spatiales directes. Le domaine des *Non-Physical Objects* recouvre le

[7] Contrairement à (Borgo et Vieu, 2006) (cf. citation supra), notre concept d'artefact n'épouse pas celui d'entité fonctionnelle.

[8] Les ontologies présentées dans l'article n'existant qu'en langue anglaise, nous conservons les étiquettes anglaises pour nommer les entités conceptuelles. Ces étiquettes sont écrites en *Italique* (avec une première majuscule), pour les concepts, et dans une *notationALaJAVA*, pour les relations. Nous associons également à l'entité un nom abrégé, qui est utilisé pour la formalisation logique.

domaine des entités sociales (ex. : la communauté francophone des chercheurs en Ingénierie Ontologique) et des entités cognitives (ex. : votre notion d'ontologie).

- Les *Perdurants* (PD) sont des entités se « déroulant dans le temps » (ex. : votre lecture de l'article). Parmi les *Perdurants* sont distingués, suivant un principe de cumulativité, les *Statives* et les *Events*[9]. Parmi ces derniers, suivant qu'ils sont atomiques ou non, les *Achievements* sont distingués des *Accomplishments*. Finalement, au sein des *Accomplishments*, les *Actions* (ACT) « exemplify the intentionality of an agent » : il s'agit ainsi d'*Accomplishments* qui sont contrôlés par un agent[10].
- *Endurants* et *Perdurants* ont des *Qualities*, que nous percevons et/ou mesurons (ex. : le poids de la copie papier de l'article entre vos mains, la durée de votre lecture de l'article).
- Ces *Qualities* prennent une valeur (*Quale*) dans des régions de valeurs qui sont des *Abstracts* (ex. : 25 grammes, 20 minutes).

On notera également la principale relation entre *Endurants* et *Perdurants*, la relation ternaire de participation temporelle (PC) signifiant que : *un Endurant participe à un Perdurant durant un Time Interval*.

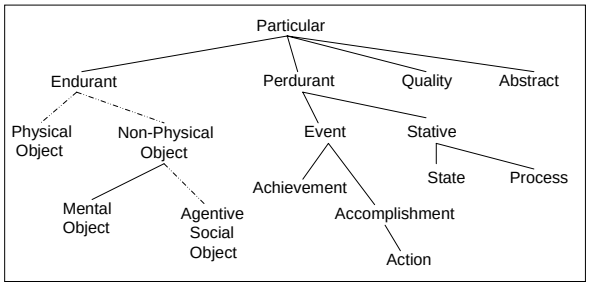

Figure 1. *Extrait de la hiérarchie des concepts de DOLCE*[11]**.**

[9] La somme méréologique de deux *Statives* (ex. : être assis) est un *Stative* du même type, cette propriété n'étant pas valable pour des *Events* (ex : un match de football).

[10] La notion d'*Action* figure en réalité dans DOLCE-Lite+, une extension « brouillon » de DOLCE, sans être formellement définie. Nous introduirons la relation *contrôle* et le rôle *agent* dans la section suivante.

[11] Ces concepts sont définis dans DOLCE au moyen d'une axiomatisation riche, qu'il est impossible de présenter dans l'article, faute de place. Notamment, *Endurants* et *Perdurants* se distinguent par le comportement temporel différent de leurs parties. Le lecteur pourra se référer à (Masolo *et al.*, 2003).

3.2. *I&DA*

I&DA (Fortier et Kassel, 2004) est une ontologie noyau du domaine de la sémiotique, initialement développée pour classer les documents par leur contenu. Comme le montre la figure 2, I&DA étend DOLCE en introduisant trois concepts principaux.

– Les *Inscriptions* sont des formes physiques de connaissances matérialisées par une substance et inscrites sur un support physique (ex. : un texte imprimé matérialisé par de l'encre sur du papier). En outre, ce sont des objets intentionnels, tenant pour d'autres entités : les *Inscriptions* réalisent (*realize*) des *Expressions*.

– Les *Expressions* sont des formes non-physiques de connaissances organisées par (*isOrderedBy*) un *Language*. À leur tour, les *Expressions* tiennent pour d'autres entités, à savoir les contenus que leur assignent des agents : les *Expressions* expriment (*express*) des *Conceptualizations*.

– Les *Conceptualizations* sont finalement les moyens utilisés par des agents pour raisonner sur le monde. Parmi les *Conceptualizations*, une distinction fonctionnelle est faite entre les *Propositions*, qui sont des descriptions de situations, et les *Concepts*, qui servent à classer des entités.

On notera que, pour rendre compte des documents, I&DA choisit de considérer trois entités distinctes plutôt que trois points de vue différents sur une seule entité. Nous verrons plus loin l'impact de ce choix sur la conceptualisation des programmes informatiques. Il est important de noter également, pour la suite de l'article, que la notion de *Concept* de I&DA correspond à celle de *notion*. Ainsi, un *Concept* correspond à un ensemble de propriétés, à l'instar du concept *Action* de DOLCE qui rassemble les propriétés *de se dérouler dans le temps*, *d'admettre des parties de types différents*, *d'être non atomique* et *d'être contrôlé par un agent*.

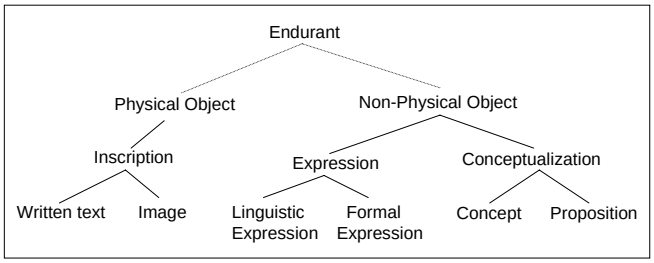

Figure 2. *Haut niveau de la hiérarchie des concepts de I&DA*

4. Notre notion d'artefact

Dans cette section, nous jetons les bases d'une ontologie des artefacts. Suivant le mode originel de spécification de DOLCE et de I&DA, nous élaborons une théorie logique du 1er ordre comportant différents types de formules : des Axiomes (A), des Définitions (D), des Théorèmes (T) et des Faits (F). Les variables apparaissant libres dans les formules doivent être considérées comme universellement quantifiées.

4.1. *Rôles de participation*

Comme nous l'avons vu dans l'analyse informelle que nous avons menée en introduction, la notion d'*artefact* repose sur des notions comme celles d'*agent* et d'*instrument* qui s'apparentent à des rôles de participation, autrement dit à des manières, pour des *Endurants*, de participer temporellement à des *Perdurants*. Pour modéliser de tels rôles, nous introduisons des relations primitives spécialisant la relation *PC* de participation temporelle. Nous définissons ainsi les relations *isAgentOfAt* (A1), *isInstrumentOfAt* (A2) et *isResultOfAt* (A3) signifiant respectivement que : i) *une entité contrôle l'action à laquelle elle participe* ; ii) *une entité est utilisée (par l'entité contrôlant l'action) pour aider à réaliser cette action* ; et iii) *une entité actualise (pour l'entité contrôlant l'action) le but visé par cette action*. On peut noter que ces rôles ne sont définis que vis-à-vis d'*Actions*. L'entité participante est pour sa part nécessairement un *Endurant*[12] (T1)(T2)(T3).

(A1) $isAgentOfAt(x,y,t) \rightarrow PC(x,y,t) \wedge ACT(y)$
(T1) $isAgentOfAt(x,y,t) \rightarrow ED(x)$
(A2) $isInstrumentOfAt(x,y,t) \rightarrow PC(x,y,t) \wedge ACT(y)$
(T2) $isInstrumentOfAt(x,y,t) \rightarrow ED(x)$
(A3) $isResultOfAt(x,y,t) \rightarrow PC(x,y,t) \wedge ACT(y)$
(T3) $isResultOfAt(x,y,t) \rightarrow ED(x)$

Ces relations permettent de définir des classes d'*Endurants* suivant leur manière de participer (D1)(D2)(D3) et de spécialiser à leur tour ces classes suivant le type d'*Action* à laquelle l'*Endurant* participe (D4)(D5)(D6).

(D1) $Agent(x) =_{def} ED(x) \wedge \exists y,t(isAgentOfAt(x,y,t))$
(D2) $Instrument(x) =_{def} ED(x) \wedge \exists y,t(isInstrumentOfAt(x,y,t))$
(D3) $Result(x) =_{def} ED(x) \wedge \exists y,t(isResultOfAt(x,y,t))$
(D4) $AgentOfWritting(x) =_{def} ED(x) \wedge \exists y,t(Writting(y) \wedge isAgentOfAt(x,y,t))$
(T4) $AgentOfWritting(x) \rightarrow Agent(x)$
(D5) $InstrumentOfPaperKeeping(x) =_{def} ED(x) \wedge \exists y,t(PaperKeeping(y) \wedge isInstrumentOfAt(x,y,t))$
(T5) $InstrumentOfPaperKeeping(x) \rightarrow Instrument(x)$
(D6) $ResultOfDiagnosing(x) =_{def} ED(x) \wedge \exists y,t(Diagnosing(y) \wedge isResultOfAt(x,y,t))$

[12] L'axiomatisation de DOLCE (Masolo *et al.*, 2003) conduit à assimiler les notions d'« endurant » et de « participant ». En effet, selon les axiomes de DOLCE : Ad33 ($PC(x,y,t) \rightarrow ED(x) \wedge PD(y) \wedge T(t)$) et Ad35 ($ED(x) \rightarrow \exists y,t(PC(x,y,t))$), seuls les *Endurants* participent à des *Perdurants* et tout *Endurant* participe nécessairement à un *Perdurant*.

(T6) ResultOfDiagnosing(x) → Result(x)

Pour compléter la définition de ces prédicats, nous rappelons quelques unes de leurs propriétés, relatives notamment à leur comportement temporel et modal[13]. Ces prédicats sont ainsi *anti-rigides* (ou dynamiques) : un *Endurant* n'est que contingentement un *Agent*, pouvant perdre cette propriété sans cesser d'exister ; qui plus est, un *Endurant* peut être l'*Agent* de plusieurs *Perdurants*, à des moments différents, voire simultanément. Ces prédicats sont également *dépendants relationnellement* : un *Agent* n'existe (en tant qu'*Agent*) que parce qu'un *Perdurant*, auquel il participe, existe. Ces propriétés (présentées ici informellement) caractérisent une classe de concepts appelés « rôles » par (Guarino et Welty, 2000).

Une telle modélisation conduit à considérer une taxinomie de rôles de participation sous le concept *Endurant* (cf. Fig. 3). Cette figure illustre surtout, avec le rôle *Instrument*, le choix de modélisation retenu pour les instances de ces rôles. Ainsi, pour prendre l'exemple d'un objet physique comme un galet jouant temporellement le rôle d'aider à maintenir des papiers, nous ne considérons dans notre modèle qu'une seule entité sur laquelle portent deux points de vue différents (F1)(F2). Il convient de noter que cette conceptualisation est conforme au paradigme courant de modélisation des rôles (Steimann, 2000)[14].

(F1) Pebble(Pebble#i)
(F2) InstrumentOfPaperKeeping(Pebble#i)

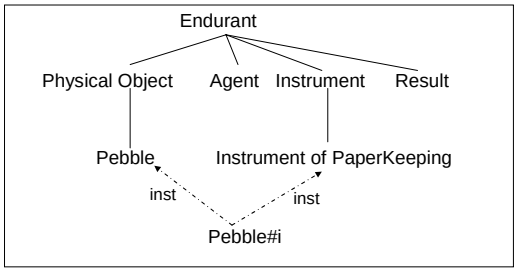

Figure 3. Pebble#i *a pour type* Pebble et *joue le rôle* de InstrumentOfPaperKeeping

[13] Pour une analyse détaillée de ces propriétés, le lecteur peut consulter (Masolo *et al.*, 2004).

[14] Une alternative, étudiée par (Masolo *et al.*, 2005), consisterait à considérer deux entités, d'une part le galet et, d'autre part, le galet en tant qu'instrument d'une action de maintien de papiers, ces deux entités étant des instances de taxinomies disjointes. Les auteurs de (Masolo *et al.*, 2005) étudient plusieurs approches consistant ainsi à introduire une nouvelle entité appelée « qua-individual » mais soulignent en conclusion que des efforts sont encore nécessaires pour approfondir la nature ontologique de ces entités.

4.2. *Compétences et fonctions*

La modélisation des rôles de participation *Agent* et *Instrument* nous permet d'aborder la modélisation des notions de *compétence* et de *fonction*. Nous effectuons cette modélisation en deux temps.

Dans un premier temps, nous introduisons des états de capacité (*CapacityState*) représentant la capacité (potentiel ou aptitude) portée temporellement par une entité (A5). Une telle capacité peut concerner (*hasForThemeAt*) le fait d'être l'*Agent* ou l'*Instrument* d'une *Action*, la distinction de ces deux rôles permettant de distinguer deux états de capacité, des états de compétence (D7) et de fonctionnalité (D8). Le fait (ou l'idée) de jouer un rôle est modélisé comme un *Concept* (au sens de I&DA). Comme il a été dit, un concept classe des entités. La relation *classifies* (A4) signifie qu'*un concept classe une entité* ou, formulé en d'autres termes, qu'*une entité vérifie l'ensemble des propriétés constituant le concept*. À noter, dans la théorie logique, la notation utilisée pour nommer les constantes désignant des concepts : le nom du prédicat est placé entre crochets (ex : [Physical Object]).

(A4) $\text{classifies}(c,x) \rightarrow \text{Concept}(c) \wedge \text{PT}(x)$
(F3) $\text{classifies}([\text{Pebble}],\text{Pebble\#i})$
(A5) $\text{CapacityState}(x) \rightarrow \text{ST}(x)$
(D7) $\text{CompetencyState}(x) =_{\text{def}} \text{CapacityState}(x) \wedge \forall y,t(\text{hasForThemeAt}(x,y,t) \rightarrow (\text{Concept}(y) \wedge \forall z(\text{classifies}(y,z) \rightarrow \text{Agent}(z))))$
(D8) $\text{FunctionalityState}(x) =_{\text{def}} \text{CapacityState}(x) \wedge \forall y,t(\text{hasForThemeAt}(x,y,t) \rightarrow (\text{Concept}(y) \wedge \forall z(\text{classifies}(y,z) \rightarrow \text{Instrument}(z))))$

Dans un second temps, nous pouvons maintenant assimiler une compétence et une fonction au thème d'un état, respectivement de compétence (D9) et de fonctionnalité (D10). La *Competence* et la *Function* les plus abstraites correspondent respectivement aux concepts d'*Agent* et d'*Instrument* (F4)(F6), autrement dit aux idées respectives de contrôler et d'aider à réaliser une *Action*. Une *Competence* plus spécifique est le concept d'*AgentOf Diagnosing* (F5) ; une *Function* plus spécifique est celle d'*InstrumentOfPaper Keeping* (F7). On peut noter que, contrairement aux *Qualities*, une *Competence* ou une *Function* ne dépendent pas de l'entité à laquelle on peut l'attribuer.

(D9) $\text{Competence}(c) =_{\text{def}} \text{Concept}(c) \wedge \exists y,t(\text{isAThemeOf}(c,y,t) \wedge \text{CompetencyState}(y))$
(F4) $\text{Competence}([\text{Agent}])$
(F5) $\text{Competence}([\text{AgentOfDiagnosing}])$
(D10) $\text{Function}(f) =_{\text{def}} \text{Concept}(f) \wedge \exists y,t(\text{isAThemeOf}(f,y,t) \wedge \text{FunctionalityState}(y))$
(F6) $\text{Function}([\text{Instrument}])$
(F7) $\text{Function}([\text{InstrumentOfPaperKeeping}])$

4.3. *Artefacts*

Venons-en à l'attribution d'une capacité à une entité. Rappelons que nous ne visons pas à rendre compte de ce mécanisme, qui est un acte mental, mais à

modéliser son résultat. Nous assimilons le résultat au fait qu'une entité éprouve temporellement (*bearsAt*) pour un agent (ou une collection d'agents) un certain état de capacité, ce dernier pouvant être un état de compétence (D11) ou de fonctionnalité (D12)[15]. Les relations que nous venons d'introduire permettent alors de définir deux catégories d'*Endurants*, des *Agentives* possédant temporellement une *Competence* (D13) et des *FunctionalObjects* possédant temporellement une *Function* (D14).

(D11) hasForCompetence(x,y,t) $=_{def}$ bearsAt(x,z,t) $\wedge$ CompetencyState(z) $\wedge$ hasForThemeAt(z,y,t)
(D12) hasForFunction(x,y,t) $=_{def}$ bearsAt(x,z,t) $\wedge$ FunctionalityState(z) $\wedge$ hasForThemeAt(z,y,t)
(D13) Agentive(x) $=_{def}$ ED(x) $\wedge \exists$y,t(hasForCompetence(x,y,t))
(D14) FunctionalObject(x) $=_{def}$ ED(x) $\wedge \exists$y,t(hasForFunction(x,y,t))

Avant de définir le concept d'*artefact*, il nous reste une étape à franchir : définir la notion d'*artificialité*. Le terme « artificiel » s'oppose couramment au terme « naturel » pour désigner toute entité « découlant » ou « résultant » d'une action humaine. Dans cette phrase, nous avons volontairement accolé les termes « découler » et « résulter » pour souligner leur large synonymie. Sur le plan sémantique, pour notre modélisation, nous faisons toutefois appel à une primitive conceptuelle distincte de notre concept *Result*, reposant sur une notion de *conséquence* mais n'entretenant aucun lien avec le but visé par l'action (contrairement au concept *Result*). Nous introduisons ainsi la relation *isConsequenceOf* (A6) pour pouvoir désigner *une entité dont, soit l'existence, soit des propriétés, découlent ou « sont la conséquence » d'une Action*, autrement dit *une entité produite ou transformée au cours d'une Action*. Plutôt qu'une relation temporelle de participation à une *Action*, à l'instar de la relation *isResultOf*, la relation *isConsequenceOf* introduit une propriété « historique » (le fait d'avoir été créé ou transformé par une *Action*) qui demeure satisfaite par l'entité tant que celle-ci existe, ou tant que les modifications apportées subsistent. Cette propriété historique explique le caractère *rigide* habituellement attribué à l'artefact, comme en témoigne l'analyse de (Dipert, 1993, p. 9) :

> "Artifactuality does not consist in any present physical qualities of a thing […] I suggest that a correct characterization of artifactuality requires a historical definition."

(A6) isConsequenceOf(x,y) $\rightarrow$ ED(x) $\wedge$ ACT(y)

La relation *isConsequenceOf* permet de définir le concept *ArtificialObject* (D15) et ce dernier permet finalement de définir notre concept *Artefact* (D16)(T7) : il

[15] Les relations introduites ne tiennent pas compte de l'« agent ». En effet, à ce stage de l'élaboration de notre ontologie, nous ne disposons pas d'une notion de *collection* qui nous permettrait de distinguer entre des artefacts *individuels* et des artefacts *sociaux*. En revanche, une telle extension est prévue en tirant parti de travaux menés par (Botazzi *et al.*, 2006) concernant l'intentionnalité collective.

s'agit d'un *ArtificialObject* résultat (au sens de la relation *isResultOf*) de l'*Action* l'ayant produit (ce résultat était donc visé) et auquel une *Function* est attribuée (c'est un *FunctionalObject*). Comme conséquence de cette modélisation, deux catégories de *FunctionalObjects* sont considérées (cf. Fig. 4) : des entités qui, à l'instar des presse-papiers (*Paper-Weight*) (D17)(T8), peuvent être indifféremment naturels ou artificiels, et des *Artefacts* qui, à l'instar des agrafeuses (*Stapler*) (D18), sont nécessairement des *ArtificialObjects*.

(D15) ArtificialObject(x) $=_{def}$ ED(x) $\wedge$ $\exists$y(isConsequenceOf(x,y))
(D16) Artefact(x) $=_{def}$ ED(x) $\wedge$ $\exists$y,t(isConsequenceOf(x,y) $\wedge$ isResultOfAt(x,y,t)) $\wedge$ FunctionalObject(x)
(T7) Artefact(x) $\rightarrow$ ArtificialObject(x)
(D17) Paper-Weight(x) = ED(x) $\wedge$ hasForFunction(x,[InstrumentOfPaperKeeping])
(T8) Paper-Weight(x) $\rightarrow$ FunctionalObject(x)
(F8) hasForFunction(Pebble#i, [InstrumentOfPaperKeeping])
(F9) Paper-Weight(Pebble#i)
(D18) Stapler(x) $=_{def}$ Artefact(x) $\wedge$ hasForFunction(x,[InstrumentOfPaperStapling])
(F10) Stapler(Stapler#j)
(F11) Paper-Weight(Stapler#j)

Comme pour les rôles de participation, ce mode de définition des *Artefacts* revient à spécialiser le concept *Endurant* au moyen d'une nouvelle taxinomie et à considérer des entités ayant tout à la fois un type et remplissant une (ou plusieurs) fonctions(s) (cf. Fig. 4). Nous aboutissons à une caractérisation duale des artefacts qui rejoint la conception de (Kroes et Meijers, 2002) pour les artefacts techniques :

> "Technical artifacts thus have a dual nature: They cannot exhaustively be described within the physical conceptualization, since this has no place for their functional features, nor can they be described exhaustively within the intentional conceptualization, since their functionality must be realized in a physical structure that is adequate to it."

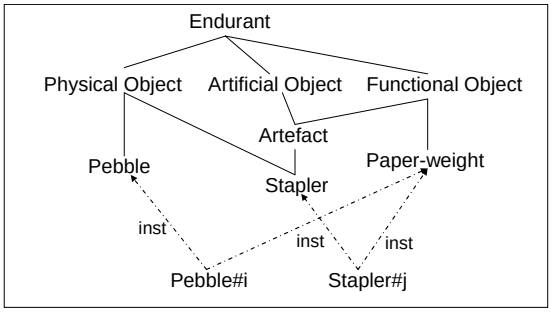

Figure 4. Pebble#i *a pour type* Pebble *et remplit la fonction* de Paper-Weight ; Stappler#j *a pour type* Stappler *et remplit la fonction de* Paper-Weight.

5. De la notion d'artefact à la notion de programme

Dans cette section, nous présentons l'apport du concept d'*Artefact* pour définir le concept de programme informatique et des concepts proches. La conceptualisation présentée est issue de l'ontologie COPS (Core Ontology of Programs and Software) en cours de développement par les auteurs du présent article (Lando *et al.*, 2007a, 2007b). COPS repose sur une modélisation de notions de base, largement consensuelles en informatique, telles qu'on peut les trouver exprimées dans de nombreux ouvrages de cours, dictionnaires et encyclopédies. Dans un premier temps, nous utilisons DOLCE et I&DA pour ancrer les programmes dans le domaine des *Expressions*, puis nous intégrons leur dimension fonctionnelle pour rendre compte finalement de leur nature duale.

5.1. *Les programmes en tant qu'expressions*

En premier lieu, la distinction introduite dans DOLCE entre *Endurants* et *Perdurants* permet de distinguer le programme-*Endurant* de ses exécutions, qui sont autant de *Perdurants*. En second lieu, en se focalisant sur le programme-*Endurant*, les distinctions établies dans I&DA entre *Inscriptions*, *Expressions* et *Conceptualizations* conduisent à distinguer trois catégories d'entités, fréquemment désignées sous le terme « programme » (cf. Fig. 5) :

– Des fichiers (*Files*), qui sont des *Inscriptions* inscrites sur un support informatique (ex. : disque optique, mémoire centrale, bande magnétique). Ces fichiers ne sont du reste qu'un type d'*Inscription* de programme : une impression sur un listing ou une visualisation sur écran sont également des *Inscriptions* de programmes.
– Des expressions, qui sont des formules bien formées (*isAWellFormedFormulaOf*) d'un langage informatique (*Computer Language Expression*). Parmi ces *Expressions* figurent les *Programs*.
– Des *DataTypes* et des *Algorithms*, qui sont des *Conceptualizations* rendant compte de la sémantique des *Programs*. Les *DataTypes* sont des *Concepts* sur lesquels reposent les langages de programmation (ex. : classe, enregistrement), et dont la diversité accompagne la variété des langages de programmation (et, au-delà, des paradigmes de programmation). Les *Algorithms* décrivent des étapes de calcul en termes de ces structures de données (ex. : affecter une constante à la valeur d'une variable, puis…).

Cette conceptualisation revient à assimiler les programmes à des entités syntaxiques. Le concept *Program* de COPS spécialise donc celui d'*Expression*, en intégrant plusieurs restrictions. Une particularité des *Programs* est de pouvoir être, soit directement exécutés par un ordinateur (éventuellement après une compilation), soit pris en charge par un interpréteur[16]. Ce n'est pas le cas, par exemple, d'une

[16] Par exemple, un programme en langage C est composé d'une ou plusieurs fonctions, dont une en particulier est impérativement baptisée « main ». Par contre, des *Expressions* comme

fonction ou d'une *déclaration de variable* qui sont pourtant des *Expressions* spécifiées dans un *Langage de Programmation*. Par ailleurs, nous considérons qu'il existe des *Expressions* exécutables ou interprétables qui ne sont pas des *Programs*. En effet, à l'instar de (Eden & Turner, 2007), nous ne parlons de *Programs* que dans le cas d'*Expressions* en langages Turing-complets (ou *General purpose programming language)*. Ainsi, une requête SQL ou une commande shell, qui sont pourtant interprétables ou exécutables, ne sont pas considérées dans COPS comme des *Programs*.

5.2. *Nature duale des programmes*

L'analyse de la dimension fonctionnelle des *Expressions* apporte un éclairage complémentaire. En effet, on peut noter qu'une requête SQL ou une commande shell sont des *Expressions* permettant à un utilisateur d'*ordonner* à un ordinateur de réaliser quelque chose, respectivement rechercher une information et lancer un programme. Ce « quelque chose » correspond à un traitement particulier de l'information, comme en témoigne le concept *Domain specific programming language* de COPS mais, surtout, il faut considérer que les *Actions* que permettent de réaliser ces *Expressions* ont pour *agent* un utilisateur et qu'il s'agit d'actes de discours : le fait de donner un ordre à un ordinateur. Ces *Expressions* sont dès lors assimilées dans COPS à des *Artefacts of communication* (cf. Fig. 5). Un *Program*, à l'inverse, permet à un ordinateur de réaliser un traitement quelconque de l'information, cette classe d'*Actions* correspondant dans COPS au concept *Computation*. Ainsi, la définition du concept *Program* est-elle complétée en ajoutant qu'il s'agit d'*Artefacts of computation*. Suivant ensuite la nature de la *Computation*, différents programmes peuvent être définis, qu'il s'agisse d'un « programme de calcul de PGCD » ou d'un « programme de traitement d'images ».

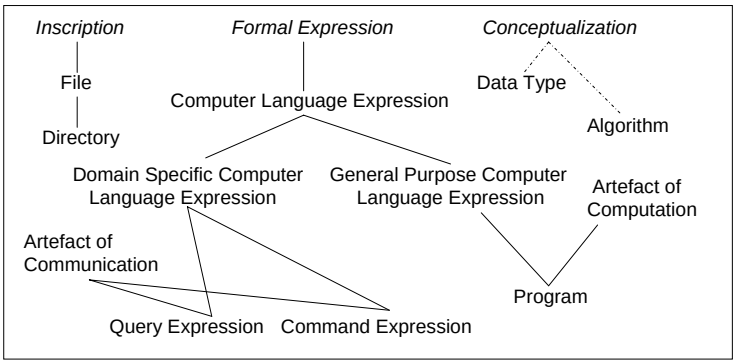

Figure 5. *Extrait de l'ontologie COPS***.**

---

une *fonction* ou une *instruction* ne possèdent pas ce point d'entrée rendant l'*Expression* exécutable ou interprétable.

6. Travaux liés

Comme nous l'avons indiqué en introduction, plusieurs efforts visant à concevoir des ontologies de programmes informatiques sont actuellement en cours (Roman *et al.*, 2005)(Oberlé *et al.*, 2006)(Eden et Turner, 2007). Les travaux de (Oberlé *et al.*, 2005) sont proches des nôtres, aussi bien sur le plan de l'objectif visé – développer une ontologie générale des programmes – que sur le plan de la démarche : l'ontologie développée COS (Core Ontology of Software) étend l'ontologie DOLCE ainsi que l'ontologie DnS (Descriptions and Situations) (Gangemi et Mika, 2003), couvrant le même domaine que I&DA. Toutefois, comme nous l'avons montré dans (Lando *et al.*, 2007b), COS repose sur des choix de modélisation différents[17] et, surtout, dans sa version actuelle, elle ne tient aucun compte de la fonction des programmes. La dimension fonctionnelle est présente dans l'ontologie WSMO (Web Service Modeling Ontology), dédiée aux services Web (Roman *et al.*, 2005). Prenons l'exemple d'un service permettant à un utilisateur de réaliser l'action *commander des billets d'avion ou de train en ligne*. La fonctionnalité du service est modélisée par sa « capacité » décrivant des pré-conditions devant être satisfaites pour que l'action puisse être réalisée (ex : l'utilisateur doit indiquer un trajet avec les lieux de départ et d'arrivée) et des post-conditions correspondant à l'état du monde après la réalisation de l'action, en cas de succès (ex : une commande est confirmée, le compte du client est débité de la somme). Ces connaissances affinent la description de l'action et doivent être considérées comme complémentaires par rapport à notre cadre de modélisation. On peut noter, par contre, que WSMO est une ontologie ad hoc pour les services Web n'intégrant pas de niveau de conceptualisation plus général, contrairement à COS et COPS. Dans le paragraphe suivant, nous examinons des travaux concernant le concept général d'artefact.

Deux ontologies connues de haut niveau, OpenCyc[18] et SUMO[19] disposent d'une caractérisation explicite du concept d'artefact, cependant ces conceptualisations demeurent éloignées des nôtres : dans OpenCyc, la définition philosophique courante sert de référence mais le concept n'est pas intégré dans une ontologie fondationnelle comme DOLCE et le concept de fonction est absent ; dans SUMO, le concept d'artefact et d'autres concepts liés comme celui d'agent sont présents, mais ont été introduits de façon superficielle. Concernant l'analyse ontologique des artefacts, les travaux les plus proches des nôtres, comme nous l'avons vu en introduction, sont ceux de (Borgo et Vieu, 2006). Ces travaux concernent des

[17] Notamment, partant de la constatation que tout programme peut être amené à être une donnée d'un autre programme, le concept *Data* de COS subsume celui de *Program*. Dans COPS, nous disposons d'une ontologie de rôles temporels de participation et le rôle *Data* peut être joué par des entités diverses, notamment des programmes. Par contre le concept *Program* est rigide et ne saurait être subsumé par un concept anti-rigide (dynamique).

[18] http://www.opencyc.org.

[19] http://ontology.tecknowledge.com.

artefacts qui sont des *Endurants* physiques (l'ontologie étend également DOLCE) et il est postulé, en préambule, que ces *Endurants* possèdent une qualité inhérente particulière, une *capacité*, ayant pour valeur, à un instant donné, l'ensemble des *dispositions* ou *comportements* que l'*Endurant* est capable d'exprimer à cet instant (par exemple : *tenir de façon stable et maintenir des papiers sans les abîmer*). L'artefact, par exemple un presse-papier, correspond à un nouvel objet créé mentalement et qui est co-localisé avec l'*Endurant* dont son existence dépend, par exemple un galet. La création de l'artefact s'accompagne de l'attribution à cet artefact d'une *capacité sélectionnée* ayant pour valeur le comportement plus précis visé pour l'artefact. On peut noter plusieurs différences avec notre proposition : le fait qu'il y ait deux objets (et non un seul, auquel une fonction est attribuée en vertu du fait que l'objet possède un état de fonctionnalité), et le fait que la proposition se limite aux *Endurants* physiques (et ne concerne donc pas les programmes qui, selon notre conceptualisation, sont des *Endurants* non physiques). En l'état, cette proposition reste incomplète, ce qui empêche d'établir des comparaisons plus détaillées. Notamment, il n'est pas précisé comment la valeur d'une capacité est modélisée, ou plutôt, comment se fait le lien entre cette valeur et les concepts de *Perdurants* et d'*Endurants* auxquels elle fait référence.

7. Conclusion

Dans cet article, nous avons proposé une double contribution. Nous avons présenté, d'une part, les bases d'une ontologie générale des artefacts et, d'autre part, nous avons montré l'apport de cette ontologie pour conceptualiser le domaine des programmes informatiques.

L'ontologie des artefacts s'appuie sur le concept philosophique courant d'artefact. Son originalité tient au cadre de modélisation proposé, ancré dans une théorie de l'action et reposant sur les deux principaux principes de modélisation suivants : i) la distinction entre des objets artificiels et des objets fonctionnels, rendant compte respectivement du caractère rigide et idéal des artefacts ; ii) la modélisation explicite d'une entité *fonction*, assimilée à la *capacité* (idéale) de pouvoir jouer le rôle d'instrument dans une action. En outre, comme résultat complémentaire, en se fondant sur une notion générale de *comportement* assimilé à un couple (rôle,action) et en l'appliquant au rôle d'agent, nous avons proposé une modélisation de la notion de *compétence*.

Dans le domaine des programmes informatiques, plusieurs notions comportent une dimension fonctionnelle, à commencer par la notion de programme elle-même. On peut également mentionner les notions de *plate-forme* et de *Service Web*. Certaines de ces notions sont du reste principalement fonctionnelles. C'est le cas du concept *plate-forme* désignant une *entité, pouvant être matérielle et/ou logicielle, permettant d'exécuter des programmes*. Pour de telles notions, l'apport d'un concept d'artefact est de permettre de préciser la fonction des entités sans détailler

leur nature. C'est ainsi qu'en imagerie médicale (dans le cadre du projet NeuroLOG) nous exploitons actuellement cette possibilité pour modéliser des *outils de traitement d'images* en définissant le traitement réalisé mais en ne précisant pas la nature logicielle de l'outil, à savoir s'il s'agit d'un programme isolé ou d'un logiciel composé de plusieurs programmes.

Toujours dans le domaine des programmes informatiques, plusieurs notions font plutôt référence à un statut de donnée. C'est le cas des *fichiers*, qui sont des inscriptions *interprétables* par un ordinateur, et des *langages de programmation de haut niveau*, auxquels est attribuée la propriété d'être *compréhensibles* par des êtres humains. Une perspective à court terme de nos travaux consiste à mobiliser la notion de comportement appliquée cette fois au rôle de donnée, autrement dit à considérer l'*idée de pouvoir jouer le rôle de donnée dans une action*, pour rendre compte de telles notions.

Remerciements

Ce travail est en partie financé dans le cadre du projet NeuroLOG (ANR-06-TLOG-024) du programme Technologies Logicielles de l'Agence Nationale de la Recherche : http://neurolog.polytech.unice.fr.

8. Bibliographie

Borgo S., Vieu L., "From Artefacts to Products", *Proceedings of the second Workshop: Formal Ontologies meet Industry (FOMI 2006)*, Trento (Italie), 14-15 décembre 2006.

Bottazzi E., Catenacci C., Gangemi A., Lehmann J., "From Collective Intentionality to intentional Collectives: an Ontological Perspective", *Cognitive System Research – Special Issue on Cognition, Joint Action and Collective Intentionality*, vol. 7, issue 2-3, 2006, p. 192-208.

Chandrasekaran B., Josephson J.R., "Function in device representation", *Engineering with Computers*, vol. 16, n° (3/4), 2000, p. 162-177.

Dipert R.R., *Artifacts, Art Works and Agency*, Temple University Press, 1993.

Eden A.H., Turner R., "Problems in the Ontology of Computer Programs", *Applied Ontology*, vol. 2, n° 1, 2007, p. 13-36.

Fortier J.-Y., Kassel G., "Managing Knowledge at the Information Level: an Ontological Approach", *Proceedings of the ECAI'2004 Workshop on Knowledge Management and Organizational Memories*, Valencia (Spain), 2004, p. 39-45.

Gangemi A., Borgo S. (eds.), Proceedings of the EKAW'04 Workshop on Core Ontologies in Ontology Engineering, Northamptonshire (UK), 2004. *http://ceur-ws.org (Vol-118).*

Gangemi A., Mika P., "Understanding the Semantic Web through Descriptions and Situations", *Proceedings of the International Conference on Ontologies, Databases and*

Applications of Semantics (ODBASE 2003), R. Meersman *et al.* (eds), Catania (Italy), 2003.

Guarino N., Welty C., "A formal Ontology of Properties", *Proceedings of the 12th International Conference on Knowledge Engineering and Knowledge Management: EKAW2000*, Lecture Notes on Computer Science, Springer Verlag, Dieng R. and Corby O. (eds.), 2000, p. 97-112.

Hilpinen R., Artifact, *Stanford Encyclopedia of Philosophy*, 2004. http://plato.stanford.edu/entries/ artifact/.

Kroes P., Meijers A., "The Dual Nature of Technical Artifacts – presentation of a new research programme", *Techné*, vol. 6, n°2, Winter 2002, p. 4-8, http://scholar.lib.vt.edu /ejournals/SPT/v6n2/pdf/kroes_meijers.pdf.

Lando P., Fürst F., Kassel G., Lapujade A., « Premiers pas vers une ontologie générale des programmes informatiques », *Actes des 18èmes Journées Francophones d'Ingénierie des Connaissances : IC 2007*, Grenoble, juillet 2007, p. 25-36.

Lando P., Lapujade A., Kassel G., Fürst F., "Towards a general ontology of computer programs", *Proceedings of the 2nd International Conference on Software and data Technologies: ICSOFT 2007*, Conference area: Knowledge Engineering, Barcelonia (Spain), 25-27 July 2007.

Masolo C., Borgo S., Gangemi A., Guarino N., Oltramari A., Schneider L., The WonderWeb Library of Foundational Ontologies and the DOLCE ontology. *WonderWeb Deliverable D18, Final Report*, vr. 1.0, 31-12-2003.

Masolo C., Guizzardi G., Vieu L., Bottazi E., Ferrario R., "Relational Roles and Qua-Individuals", *Proceedings of the AIII Fall Symposium on Roles, an interdisciplinary perspective*, Hyatt Crystal City, Arlington, Virginia, November 3-6, 2005.

Masolo C., Vieu L., Bottazzi E., Catenacci C., Ferrario R., Gangemi A., Guarino N., "Social Roles and their Descriptions", *Proceedings of the Ninth International Conference on the Principles of Knowledge Representation and Reasoning*, Whistler, D. Dubois, C. Welty and M.-A. Williams (eds.), BC, Canada, 2004, p. 267-277.

Oberle D., Lamparter S., Grimm S., Vrandecic D., Staab S., Gangemi A., "Towards Ontologies for Formalizing Modularization and Communication in Large Software Systems", *Applied Ontology*, vol. 1, n° 2, 2006, p. 163-202.

Pacherie E., The content of intentions, *Mind and Language* 15, 2000, p. 400-432.

Roman D., Keller U., Lausen H., De Bruijn J., Lara R., Stollberg M., Polleres A., Feier C., Bussler C., Fensel D., Web Service Modeling Ontology, *Applied Ontology*, vol. 1, 2005, p. 77-106.

Sowa J.F., *Knowledge Representation: logical, philosophical, and computational foundations*, Brooks/Cole, 2000.

Steimann F., "On the representation of roles in object-oriented and conceptual modelling", *Data and Knowledge Engineering*, 35, 2000, p. 83-106.

Temal T., Lando P., Gibaud B., Dojat M., Kassel G., Lapujade A., "OntoNeuroBase: a multi-layered application ontology in neuroimaging", *Proceedings of the 2nd Workshop: Formal Ontologies Meet Industry: FOMI 2006*, Trento (Italy), 14-15 décembre 2006.

Welty C., An Integrated Representation for Software Development and Discovery. *Ph.D. Thesis*, RPI Computer Science Dept. July, 1995. Disponible à : http://www.cs.vassar.edu/faculty/welty/papers/phd/