REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MENTOURI DE CONSTANTINE
FACULTE DES SCIENCES DE L'INGENIEUR
DEPARTEMENT D'INFORMATIQUE

N° d'ordre : ……
Série : ……

THESE

Présentée Pour obtenir le Diplôme de Doctorat en Sciences

Interopérabilité des Modèles de Workflow

Soutenue à Constantine le ... /… / ….

Présentée par :

Mr. Salah HAMRI

Dirigée par :

Pr. Mahmoud BOUFAIDA
Pr. Nacer BOUDJLIDA

Devant le jury :

Président	**: Mr. Mohamed Benmohammed**	**Professeur, UMC de Constantine**
Rapporteurs	**: Mr. Mahmoud Boufaida**	**Professeur, UMC de Constantine**
	: Mr. Nacer Boudjlida	**Professeur, UHP de Nancy 1 (France)**
Examinateurs	**: Mr. Ahmed Nacer Mohamed**	**Professeur, USTHB d'Alger**
	Mr. Nacereddine Zarour	**Professeur, UMC de Constantine**
	Mr. Abdelkamel Tari	**Maître de Conférence, Université de Bejaia**

Remerciements

D'abord, louange à DIEU, le tout puissant, que je remercie beaucoup de m'avoir donné la force et le courage de terminer cette thèse Sans lui, rien ne pourrait être réalisé.

Je tiens tout d'abord à remercier Monsieur Mahmoud Boufaida., Professeur à l'université Mentouri de Constantine, pour m'avoir guidé, m'avoir soutenu, aidé et conseillé au cours de cette thèse. Son aide, sa disponibilité, ses conseils, et son soutien durant toute cette période, m'ont toujours redonné confiance et volonté. Je le remercie particulièrement pour ses encouragements incessants qui m'ont été d'un apport considérable dans la motivation et le stimulus de mes capacités. Qu'il trouve ici, l'expression de ma sincère reconnaissance.

Je remercie aussi Mr.Boudjlida Nacer, Professeur à l'université Henri Poincaré de Nancy1 (France), co-encadreur de ce présent travail, pour ses directives et ses critiques constructives durant mes stages à Nancy. Ses remarques pertinentes et ses conseils m'ont été d'une grande aide, sans oublier son souci de rigueur et de clarté qui m'ont beaucoup aidé à voir les choses autrement. Je le remercie également pour la lecture du manuscrit et ses corrections minutieuses. Qu'il trouve ici, le témoignage de ma profonde gratitude.

Je tiens aussi à remercier Monsieur Mr. Mohamed Benmohammed, Professeur à l'université Mentouri de Constantine de m'avoir fait l'honneur d'accepter de présider le jury de ma soutenance de thèse, ainsi que :

- Monsieur Mr. Ahmed Nacer Mohamed, Professeur à l'université USTHB d'Alger,
- Monsieur Mr. Nacereddine Zarour, Professeur à l'université Mentouri de Constantine,
- Monsieur Mr. Abdelkamel Tari, Maître de Conférence à l'université de Bejaia**.**

Pour l'honneur d'avoir accepté de faire partie de ce jury en tant qu'examinateurs.

J'adresse mes remerciements également à ma famille pour m'avoir encouragé et soutenu en toutes circonstances, et en particulier ma petite famille à laquelle je dédie cette thèse pour sa patience et son soutien moral pendant toutes ses années de thèse. Merci d'avoir cru en moi, sans elle, rien de tout ceci n'aurait pu arriver.

Enfin, je n'omettrais pas de remercier vivement les membres de l'équipe du Laboratoire d'Informatique LIRE de Constantine, et en particulier les membres de l'équipe SIBC (Systèmes d'Information & bases de Données), doctorants et magisters avec lesquels j'ai pu échangé des idées, discuter, et enfin partagé et vivre des moments agréables parmi eux.

A ma petite famille

ملخص

لمواجهة عولمة الاقتصاد، القيود المفروضة من طرف الشراكة ومقتضيات الزبائن، تتجه الشركات لتركيز اهتماماتها حول دمج وقابلية التحام نماذجها للمعالجات التجارية. قابلية الالتحام أضحت لكثير من الشركات، خاصة منها الشركات الصناعية كبيرة الحجم، ضرورة قصوى لمواجهة المنافسة في السوق والتأقلم للتطورات والمستجدات المرتقبة. حتى تتمكن من فرض نفسها في السوق، تقوم هذه الشركات بخلق شراكة يتم من خلالها التواصل بين ممثلين من طبيعة مختلفة (الممون، الزبون،الخ...) وهذا لتمكين نماذجها للمعالجات التجارية بالتعاون والتلاحم فيما بينها. من اجل تجسيد واقعي لقابلية التلاحم، تم اقتراح مجموعة من الحلول، الطرق، الوسائل والتقنيات التي تتوجه إلى تبني القياسات. وهذا يعني أن مفهوم المدلول لا يعالج بصفة تصويرية في أغلب الأحيان.

عمل هته المذكرة يندرج تحت إطار تصوري لحل إشكالية قابلية التحام نماذج المعالجات التجارية ويركز خاصة على مشاكل عدم تجانس مدلول هته النماذج هذا العمل يسمح أيضا بتحقيق الأهداف المسطرة من طرف الشراكة بصفة تعاون مرن ومفتوح.

من خلال الاستقصاء وتحليل أدبيات ومصادر قابلية الالتحام للشركات، استطعنا اكتشاف مسارين بحث مقتبسين من تيارين بحث مختلفين. هذان المساران أديا بنا لاقتراح منهجيتين للتصور، حيث تمت المقارنة بينهما وهذا وفق معايير معرفة.

المسار الأول يندرج تحت إطار تصوري ويستخدم تقنيات النمذ جة العليا وتحويل النماذج. المنهجية المقترحة مبنية أساسا على MDA ، على الانتولوجيا (ontologies) ونموذج خدمات الواب (services Web).هذا المسار يوفر مستوى عالي من التجريد لنمذ جة الانتولوجيات من اجل وصف مدلول للنماذج وكذا لنمذ جة بنية تحتية تقنية ما حتى يسمح للتعاون بين ممثلي الشركات. هته المنهجية يطلق عليها اسم " منهجية قياسات" لأنها تستعمل تقنية الواب (Web) مع كل القياسات المتعلقة به وذلك بطريقة غير موجهه مباشرة للصناعة.

المسار الثاني يندرج أيضا تحت إطار تصوري، لكنه يستخدم تقنيات تعليم مدلول النماذج العليا للمعالجات، للنماذج المرفقة وأخيرا للمفاهيم المختلفة القاعدية التي ترتبط جوهريا بنماذج المعالجات المعلوماتية، السلوكية، الوظيفية...الخ. هته التعليمات المد لولية ترجع إلى مجموعة مشتركة من الانتولوجيات. هته المنهجية يطلق عليها اسم" منهجية التوحيد" لأنها تستند إلى نموذج أعلى مشترك موحد، مقبول من جانب الشراكة. هته المنهجية تتميز بكونها تتجاهل كل سياق صناعي وكذلك كل تقنية الواب مع كل القياسات المتعلقة به.

كلمات مفتاحيه : نموذج عمليات المعالجة، قابلية الالتحام، مدلول، نما ذج عليا، (Architecture Model Driven) ،نتولوجيات (ontologies)، خدمات الواب (Web services) ، تعليمات مدلوليه.

Résumé

Face à la mondialisation de l'économie, aux contraintes imposées par le partenariat et aux exigences des clients, les entreprises ont accentué leur besoin en matière d'intégration et d'interopérabilité de leurs modèles de Workflow (ou business process). Cette interopérabilité est devenue pour beaucoup d'entreprises, notamment les grandes entreprises industrielles, une nécessité incontournable pour pouvoir exister dans un marché très concurrentiel et s'adapter à un environnement en perpétuelle évolution. Pour supporter cette interopérabilité, différentes solutions, approches et technologies ont été proposées et convergent directement vers l'adoption de standards, et par conséquent l'aspect sémantique n'est toujours pas correctement traité de manière conceptuelle. En effet, la prise en compte de cet aspect de manière conceptuelle peut promouvoir l'interopérabilité et lui apporter plus de flexibilité.

Notre travail de thèse s'inscrit dans une problématique de recherche d'une solution conceptuelle permettant de résoudre principalement les problèmes d'hétérogénéité sémantique de ces modèles et de répondre aux objectifs fixés par le partenariat en termes de besoins d'une coopération flexible et ouverte entre des participants hétérogènes.

Cependant, l'analyse de la littérature sur l'interopérabilité des entreprises et l'espace bibliographique que nous avons exploré pour résoudre cette problématique, nous ont permis de détecter deux pistes de recherche empruntées par deux courants de recherche différents. Ces deux pistes nous ont amenés à proposer deux approches conceptuelles que nous avons comparées selon certains critères définis et qui sont liés prinicipalement à la problématique posée et aux objectifs visés.

La première piste utilise les techniques de méta-modélisation et de transformation de modèles. L'approche préconisée utilise conjointement MDA (Model Driven Architecture), les ontologies et les Web services. L'approche MDA est utilisée pour capturer l'aspect conceptuel, les ontologies pour la description sémantique et les Web services pour représenter la plate-forme d'interopérabilité. C'est une approche dite "approche de standardisation" car elle se base sur les concepts de MDA. Elle est flexible et ouverte dans la mesure où elle peut s'inscrire dans un cadre d'utilisation des standards industriels.

La deuxième piste utilise cette fois ci, les techniques d'annotation sémantique pour annoter les méta-modèles de processus, leurs modèles correspondants, leurs buts, leurs profils et enfin les différents aspects de base qui leur sont intrinsèquement liés (de type informationnel, comportemental, organisationnel, etc.). Ces annotations sémantiques se réfèrent à un ensemble commun d'ontologies. L'approche préconisée est basée sur un méta-modèle commun d'annotation sémantique des modèles de processus. Elle est dite "agnostique" dans le sens où elle ignore tout contexte industriel, ainsi que toute technologie Web avec les standards qui lui sont associés. Elle est flexible et ouverte dans la mesure où le mécanisme d'annotation sémantique est totalement indépendant de tout langage de représentation d'ontologies permettant ainsi, à la communauté des développeurs de choisir leur langage ontologique préféré tel que OWL ou OWL-S pour représenter la sémantique de leurs modèles.

MOTS-CLES : **Modèle de Workflow, Interopérabilité, Sémantique, Model Driven Architecture, Méta-modèle Commun, Ontologies, Web services, Annotations sémantiques.**

Abstract

In front of the economy globalization, the imposed constraints by the partnership and customers' requirements, companies have stressed their needs of integration and interoperability of their Workflow models (or business process). This interoperability is for many companies, in particular the big industrial ones, a necessity for their existence in the very competitive market and to adapt themselves to an environment which is in perpetual evolution. To support the interoperability, various solutions, approaches and technologies were proposed which converge directly to the adoption of standards, and consequently, the semantic aspect is not correctly addressed by today's interoperability solutions that focus mainly on the syntactical and technical interoperability. Thus, addressing the semantic aspect will promote the interoperability in a conceptual manner by providing it more flexibility.

Our work of thesis treats the research problematic. It concentrates on the conceptual aspect and aims at solving the problem of the interoperability of Workflow models and focuses mainly on the problems of semantic heterogeneousness of these models and allows us to satisfy fixed objectives by the partnerships in terms of flexible and opened cooperation.

However, the analysis of literature on the interoperability of enterprises and bibliographical space that we investigated to resolve this problem, allowed us to find out two different research tracks. This leads us to propose two conceptual approaches which we compared according to certain defined criteria that are related on the problematic and the assigned objectives.

The first track uses the techniques of meta-modeling and transformation of models. The proposed approach combines MDA (Model Driven Architecture), ontologies and Web services. MDA is uses to capture the conceptual aspect, ontologies for the semantics of Workflow models and Web services for the technical plate-form of interoperability. It is a called "approach of standardization" because it is based on the MDA concepts. The proposed approach is flexible and opened since it is based on standards associated to the Web technology.

The second track uses the semantic annotation techniques to annotate semantically the meta-models of processes, their corresponding models, their goals, their profiles and finally the various basic aspects which are intrinsically attached to the Workflow models that are informational, behavioral, and organizationnal types, etc. These semantic annotations refer to a common set of ontologies. This approach is based on a common semantic annotation méta-model of the process models. The proposed is agnostic in the sense that it ignores any industrial context, as well as any Web technology and all associated standards. By keeping the semantic annotation mechanism separate from the representation of the semantic descriptions, the approach offers flexibility and openess to developer community to select their favorite semantic representation language (such as OWL or OWL-S).

KEYWORDS : **Workflow Model, Interoperability, Semantics, Model Driven Architecture Common Meta-Model, Ontologies, Web services, Semantic Annotations.**

LISTE DES FIGURES

LISTE DES TABLEAUX

Sommaire

Sommaire

Chapitre II. Interopérabilité - Formes – Approches et Techniques mises en oeuvre 39

PARTIE 2 – APPROCHES CONCEPTUELLES POUR L'INTEROPERABILITE DES MODELES DE WORKFLOW

Chapitre III. Approche basée sur les Techniques de méta-modélisation et de Transformation de modèles 78

Introduction

Introduction

1. Contexte de la thèse

Aujourd'hui, les entreprises vivent des transformations majeures au plan Humain, Organisation et Technologique (HOT) qui altèrent l'organisation et lui imposent un changement comme une condition de survie. Ce changement provient principalement de différents facteurs tels que :

- La propagation des Nouvelles Technologies de l'Information (NTIC) qui ne cesse de s'accélérer (Internet, technologie objet, Intranet, architecture client/serveur, etc.) ;
- L'évolution économique et industrielle (mondialisation des marchés, restructurations rapides des entreprises, accentuation de la concurrence, etc.) ;
- L'évolution financière (développement des marchés financiers, internationalisation, accentuation des contraintes induites par les manoeuvres financières, etc.) ;
- L'émergence de nouveaux concepts organisationnels et de management pour la gestion des processus métiers tels que : le BPR (Business Process Reegineering) [1] pour une reconception radicale des processus d'entreprise et le CPI (Continuous Process Improvement) [2] pour plutôt, une amélioration incrémentale des processus d'entreprise.

Face à ces différents facteurs, les activités des entreprises ont été perturbées et n'arrivent plus s'adapter à un environnement compétitif et éprouvent de plus en plus de difficultés à réagir aux sollicitations auxquelles elles sont soumises. Ce constat nécessite donc, une adaptation constante à un environnement en perpétuelle évolution, et pour pouvoir exister dans un marché très concurrentiel, ces entreprises doivent s'organiser en partenariat au sein duquel interagissent différents acteurs (fournisseurs, clients, sous-traitants, etc.).

Dans le domaine du Workflow[1] (ou business process[1]), pour faire face à la mondialisation de l'économie, et aux contraintes imposés par le partenariat et aux exigences des clients, les entreprises ont accentué leur besoin en matière d'intégration et d'interopérabilité de leurs processus d'affaires[1] dans un contexte B2B (Business to Business).

Cette interopérabilité que nous appelons aussi, coopération est vue comme étant la capacité des entreprises à présenter leurs modèles de processus afin d'être en mesure de les échanger ou de les partager. Elle est devenue pour beaucoup d'entreprises, notamment les grandes entreprises industrielles, une nécessité incontournable pour assurer la pérennité économique des entreprises, et s'inscrire dans une dynamique d'intégration.

[1] Workflow : traduit littéralement par 'flux des travaux', est souvent confondu dans la littérature avec les termes : 'business process' (traduit en français par 'processus métier'), 'processus d'affaires' dans un contexte B2B ou encore par 'modèle de processus'.

Cependant, pour faire coopérer des modèles de processus qui sont hétérogènes, autonomes et répartis sur des sites différents, différents niveaux d'hétérogénéité interviennent et rendent complexe l'interopérabilité. Il s'agit principalement : d'hétérogénéité syntaxique (qui concerne la diversité des langages de modélisation utilisés, encodages de données, etc.), d'hétérogénéité sémantique (qui concerne la variété des concepts manipulés), d'hétérogénéité technique (qui concerne la multiplicité des plates formes d'exécution, de protocoles de communication, de systèmes d'exploitation, etc.), et d'hétérogénéité au niveau métier (qui concerne l'organisation, les responsabilités et les objectifs métiers, etc.)[3].

Dans notre thèse, nous nous intéressons particulièrement à l'hétérogénéité sémantique des modèles de processus, qui constitue l'handicap majeur pour la mise en œuvre de l'interopérabilité. En effet, la compréhension de la sémantique de ces modèles permet aux partenaires d'interpréter de manière commune et non ambiguë des fragments de modèles telle qu'une activité, par exemple, lors de la coopération. Notre préoccupation principale dans cette thèse, est donc de traiter cette hétérogénéité sémantique au niveau conceptuel de la problématique de l'interopérabilité que nous présentons comme suit.

2. Problématique et objectifs

Notre travail s'inscrit dans une problématique de recherche, et concerne le domaine de l'interopérabilité des modèles de Workflow où il s'agit précisément de trouver une solution conceptuelle permettant de résoudre cette problématique indépendamment des standards industriels, d'utilisation des technologies de Web sémantique et de toute plate-forme technique d'exécution.

Actuellement, dans un contexte B2B (Business to Business), les entreprises utilisent le Web comme moyen de communication pour établir l'interopérabilité entre leurs processus métiers, appelés processus d'affaires, faciliter les échanges commerciaux. Ces entreprises veulent permettre à leurs partenaires d'affaires d'accéder directement à leurs processus métiers via le réseau Internet. Elles veulent ainsi que les différents logiciels de gestion de processus tels que les ERP[1] (Enterprise Resource Planning), SCM[2] (Supply Chain Management) ou CRM[3] (Customer Relationship Management) et autres, communiquent entre eux.

Cependant, pour communiquer et surtout pour coopérer au sein d'un partenariat afin de comprendre de manière commune et partagée les informations métiers échangées, différents problèmes apparaissent et qui devront être pris en compte. Ces problèmes sont dûs à différents niveaux d'hétérogénéité à savoir :

- Hétérogénéité *syntaxique* qui concerne la diversité des langages de spécification Workflow utilisés tels que XPDL (XML Process Definition Language) [4], XLANG

[1] ERP ou en français PGI (Progiciel de Gestion Intégré) : Applications dont le but de coordonner l'ensemble des activités d'une entreprise telles que la production, les approvisionnements, les ventes, les ressources humaines.

[2] CRM ou en français GRC (Gestion de la Relation Client) : Applications qui regroupent toutes les fonctions permettant d'intégrer les clients dans le système d'information de l'entreprise.

[3] SCM ou en français GCL (Gestion de la Chaîne Logistique) : Applications qui regroupent toutes les fonctions permettant d'intégrer les fournisseurs et la logistique au système d'information de l'entreprise.

[5], WSFL (Web Services Flow Language)[6], WSCI (Web Services Choreography Interface) [7], WSCL (Web Services Conversation Language) [8], BPEL4WS (Business Process Execution Language for Web Services) [9], [10], ebXML [11].

- Hétérogénéité *sémantique* qui est dûe au sens et à la signification des éléments métiers. Elle concerne précisément la variété des concepts utilisés pour modéliser un processus métier ;

- Hétérogénéité *technique* qui concerne la multiplicité des plates-formes technologiques tels que les moteurs de Workflows, les protocoles de communication, les systèmes d'exploitation.

Un autre niveau d'hétérogénéité qui concerne le niveau métier auquel nous ne nous intéresserons pas dans notre travail, qui peut-être éventuellement pris en considération dans des cas réels dans des entreprises industrielles. Il s'agit principalement du niveau métier qui est défini comme suit : « il regroupe les responsabilités, autorisations, confiances, aspects légaux, propriétés intellectuelles et structures organisationnelles nécessaires à l'acceptation des échanges d'information, … » [3].

Un autre point concernant la problématique de l'interopérabilité des Workflows est celui du degré de visibilité de ces processus. Certains auteurs [12] expliquent que « pour des raisons stratégiques liées à la compétitivité de l'entreprise, la plupart des entreprises partenaires d'un réseau collaboratif ne dévoilent pas directement leurs savoir-faire et connaissances liés à leurs processus métiers internes ».

Dans un contexte de collaboration inter-organisationnelle, les entreprises doivent d'abord, gérer le degré d'ouverture de leurs processus métiers vis-à-vis de leurs partenaires, suivant un contrat de collaboration pré-établi. Ce degré d'ouverture présente généralement deux parties distinctes : une partie publique (approche de type boîte blanche), qui présente une interface accessible pour tous les partenaires et une partie privée (approche de type boîte noire), qui reste cependant protégée. Dans certains cas, cette dernière partie sera semi privée et sera accessible uniquement à des partenaires autorisés et identifiés dans un contrat de coopération pré-établi

Ce degré de visibilité «public, privé ou semi privé » a certainement un impact dans un contexte d'interopérabilité sur la conception des systèmes d'information et en particulier sur la modélisation des processus métiers et par conséquent, toute solution aux problèmes d'interopérabilité doit le prendre en considération.

Dans la littérature, plusieurs approches d'intégration et d'interopérabilité ont été proposées par différents auteurs [13], [14], [15], [16], [17], [18], [3] pour résoudre la problématique de l'interopérabilité des systèmes d'information, en général, et en particulier, celle des modèles de Workflow [19], [20], [24], [25], [26], [27]. La plupart de ces approches utilisent principalement des techniques basées sur des formats standards, des intergiciels de communication et de distribution (middlewares), des outils EAI (Enterprise Application Integration), des outils BPM (Business Process Management), des architectures orientées services (SOA - Service Oriented Architecture), et enfin des techniques d'ingénierie de modèles (MDA – Model Driven Architecture).

Dans le domaine du Workflow, nous supposons que les entreprises ont modélisé leurs processus métiers distribués de manière à les rendre interopérables quels que soient leurs

plates-formes ou leurs langages. Pour collaborer et accentuer les échanges commerciaux inter-entreprises via le Web, elles adoptent une technologie appropriée : il s'agit de la technologie des Web services.

Cependant, pour coopérer dans un contexte de Web services, ces entreprises fragmentent alors leurs processus métiers en plusieurs activités appelés " services métiers " (ou business services) de sorte que chacun effectue une tâche distincte. Ces services seront publiés en tant que des Web services facilitant ainsi, l'interopérabilité entre les partenaires collaboratifs à travers le Web, en utilisant des langages de composition de Web services tels que BPEL4WS, WSFL, XLANG, ou WSCI.

Bien que cette solution semble être la plus adaptée pour faciliter aux entreprises industrielles de coopérer via Internet dans un contexte d'échanges B2B (Business to Business) et apporte beaucoup d'avantages tels que ouverture, flexibilité, néanmoins, elle ignore complètement l'aspect sémantique des éléments métiers et se focalise principalement sur l'aspect syntaxique des Web services en utilisant le standard XML.

Cependant, pour décrire la sémantique de ces Web services ou les annoter sémantiquement avec des ontologies, certains auteurs ont proposé des langages de représentation d'ontologies basés Web [21], [22], [23] afin de favoriser l'interopérabilité sémantique. Aussi, Il n'en demeure pas moins que ces initiatives sont en cours de développement et/ou manquent de maturité. De plus, la description ou l'annotation sémantique de ces Web services est basée directement sur l'utilisation technique de ces langages de représentation d'ontologies et leur compréhension. Ce qui nécessite donc, beaucoup d'efforts d'apprentissage de la part des utilisateurs pour leur mise en pratique.

Par conséquent, l'interopérabilité sémantique constitue une problématique toujours ouverte dans un cadre conceptuel, indépendamment de toute plate-forme et surtout de tout langage de description ou d'annotation sémantique.

Outre cet aspect conceptuel de la problématique de l'interopérabilité auquel nous nous intéressons dans notre travail, d'autres aspects qui nous paraissent évidents et qui sont liés de manière spontanée à toute coopération dans un environnement compétitif, ouvert. Ces aspects sont en réalité des besoins de coopération qui sont dictés par tout partenariat désirant être flexible dans un contexte hétérogène.

Dans notre travail, nous exprimons ces besoins de coopération en termes d'objectifs que nous citons comme suit:

- **Autonomie :** Elle est définie comme étant la capacité d'un modèle de Workflow de s'exécuter indépendamment de la coopération. Il s'agit en réalité, d'autonomie de modélisation (ou de conception) et d'exécution. Cette autonomie concerne aussi les systèmes de gestion de Workflow qui les gèrent.

- **Flexibilité :** Elle est définie comme étant la capacité d'un modèle de Workflow (ou d'un système de Workflow) de s'adapter facilement ou de s'intégrer de manière faiblement couplé avec d'autres modèles de Workflow (ou d'autres systèmes de gestion de Workflow) dans un environnement coopératif hétérogène.

- **Ouverture :** Elle est définie comme étant la facilité d'adopter la technologie Web, ainsi que les standards qui lui sont associés dans une éventualité d'élargir un partenariat où les entreprises expriment un grand besoin d'ouverture et de coopération à l'échelle mondiale et souhaitent bénéficier des Nouvelles Technologies de l'Information et la Communication (NTIC).

- **Préservation du savoir-faire :** Dans un marché très concurrentiel, les partenaires d'une alliance se méfient toujours de leurs partenaires dans le même secteur et veillent à protéger leurs processus Workflows (ou business process) contre toute intrusion ou espionnage. En effet, si un partenaire arrive à déduire ou à acquérir le processus d'un autre partenaire, il lui est possible de déduire les points forts et les points faibles de son partenaire et peut donc effectuer des améliorations sur son propre processus pour offrir de meilleurs services et gagner des marchés. Par conséquent, les entreprises n'acceptent pas de divulguer leur savoir-faire par mesure de sécurité et de protection de leur expérience et technique de travail. Cependant, pour des partenaires autorisés et confiants, un certain degré d'intervisibilité est permis pour pouvoir coopérer.

Cependant, l'analyse des approches existantes que nous avons faite pour l'interopérabilité des Workflows inter-entreprises [24], [25], [26], [27], nous a révélé que ces dernières ne préservent pas l'autonomie des modèles de Workflow pré-établis des partenaires (ou de leurs systèmes de gestion de Workflow), ne sont pas flexibles ou peu flexibles puisque tous les participants doivent se conformer à des interfaces bien définies.

Notre problématique d'interopérabilité est orientée vers la recherche d'une solution conceptuelle pour résoudre l'interopérabilité des modèles de Workflow (ou business process), et se focalise principalement sur l'hétérogénéité sémantique de ces modèles pour permettre une compréhension commune et un partage entre les partenaires coopératifs. Elle permet aussi, de répondre aux besoins de la coopération exprimés par le partenariat à savoir: l'autonomie des Workflows existants (ou des systèmes de gestion de Wokflow), la flexibilité, le respect du savoir-faire des partenaires et enfin l'ouverture pour les entreprises qui souhaitent élargir la coopération et travailler dans un contexte standard.

3. Contributions

L'étude de l'état de l'art sur les différentes approches d'interopérabilité et les techniques mises en oeuvre, nous a permis de recenser quelques caractéristiques importantes liées principalement aux aspects d'abstraction, de flexibilité et d'ouverture. Ces caractéristiques sont pour nous, des critères importants pour le choix d'une solution d'interopérabilité. De même que l'analyse de la littérature sur l'interopérabilité des entreprises, issue des travaux du réseau d'excellence européen INTEROP [121], du projet européen ATHENA [122] et de plusieurs travaux [3], [16], [17], montre qu'il existe principalement deux approches pour traiter la problématique de l'interopérabilité :

1. **Une approche de standardisation :** Dans cette approche, les entreprises se mettent d'accord pour standardiser la manière de présenter leurs données, processus et applications, mais aussi leurs outils, méthodes et stratégies dans un contexte standard orienté industriel. Ce qui leur permet de bénéficier des nouvelles technologies de l'information, et en particulier de la technologie Web, ainsi que les standards qui lui associés (langages, plates-formes, protocoles de communication, etc.) pour faciliter la

mise en ouvre de l'interopérabilité. Comme exemple, ils se mettent d'accord sur un modèle commun standard pour l'échange de modèles de processus basé selon les concepts du MDA [72], [129], [131].

2. **Une approche d'unification :** Dans cette approche, les entreprises se mettent d'accord pour unifier la manière de présenter leurs données, processus et applications, mais aussi leurs outils, méthodes et stratégies pour faciliter l'interopérabilité. Comme exemple, ils se mettent d'accord sur un langage commun tel que UEML (Unified Enterprise Modeling Language) [3] [18] qui est un langage unifié de modélisation d'entreprise et offre un moyen d'entente aux partenaires. Il représente alors un modèle commun (pivot) pour l'échange des modèles différents d'entreprises.

Partant de ce constat, nous avons exploré deux pistes de recherche différentes pour traiter cette problématique. Ce qui nous a conduit alors à proposer deux approches conceptuelles que nous avons comparées selon certains critères définis et qui sont liés principalement à la problématique posée et aux objectifs visés. Plus précisément, notre travail de thèse est guidé par deux motivations principales s'appuyant sur deux convictions personnelles :

1. La première motivation s'appuie sur la conviction suivante *:*

   " Pour interopérer, d'abord modéliser, puis formaliser pour raisonner ".

Partant de cette conviction, nous proposons une approche dite "standard" permettant de nous fournir un haut niveau d'abstraction et un cadre conceptuel général pour favoriser l'interopérabilité. Elle est orientée dans une perspective d'élargir la coopération à l'échelle mondiale et destinée pour les entreprises qui souhaitent coopérer via Internet afin de bénéficier de la technologie Web, et en particulier des standards qui lui sont associés. Cette approche est dite "conceptuelle" car elle permet de capturer l'aspect conceptuel de toute solution d'interopérabilité et de favoriser la portabilité des modèles. Elle est basée sur l'ingénierie des modèles, et en particulier sur l'approche MDA (Model Driven Architecture) [29].

Notre première contribution porte sur l'utilisation des techniques de méta-modélisation et de transformation de modèles pour permettre :

- *La modélisation des ontologies* pour décrire la sémantique des modèles de processus (ou des fragments de modèles) qui sont des activités ou des tâches de manière indépendante de tout langage de représentation d'ontologies basé Web ou de tout langage d'annotation sémantique tels que OWL (Ontology Web Language) [28], OWL-S (Ontology Web Language for Services) [21] ou WSDL-S (Web Service Description Language - Semantics) [22].

  Dans cette approche, chaque fragment de modèle est considéré comme un service métier que nous dotons d'une sémantique, puis nous le modélisons comme un Web service sémantique afin de favoriser l'interopérabilité des modèles de processus via ces Web services sémantiques.

- *La modélisation de la plate-forme technique* qui sera choisie par les partenaires coopératifs, et qui répond surtout aux besoins de la coopération telles que flexibilité et ouverture. Pour notre travail, le modèle SOA (Service Oriented Architecture) est choisi, et en particulier la plate-forme des Web services comme plate-forme cible. En

effet, le modèle SOA présente beaucoup d'avantages (intégration à faible couplage, à faible coût) par rapport aux autres middlewares. Le choix d'une architecture d'interopérabilité est fortement lié aux besoins d'une coopération flexible.

Cette approche préconise alors de combiner à la fois l'approche MDA (Model Driven Architecture), les ontologies et les Web services. En effet, l'association de ces trois approches nous permet effectivement ***i)*** *de nous offrir un niveau d'abstraction élevé, et cela grâce aux principes de MDA* ***ii)*** *de prendre en compte l'aspect sémantique des modèles (ou fragments de modèles) par l'introduction du concept d'ontologies* ***iii)*** *de nous fournir un cadre architectural partagé et flexible dans un contexte Web grâce aux concepts de SOA.*

Cette contribution se résume principalement à créer une passerelle entre les deux technologies: le Web sémantique et le Web service sémantique afin de faciliter l'adoption des langages standards pour la description sémantique des Web services dans un contexte MDA et permettre ainsi, aux utilisateurs qui sont familiers avec les outils MDA (ou les outils UML) d'intégrer facilement leurs outils pour décrire la sémantique ou annoter sémantiquement leurs modèles en faisant abstraction de tout langage de représentation d'ontologies basé Web tels que OWL, OWL-S ou WSDL-S.

En effet, ces langages nécessitent beaucoup d'efforts d'apprentissage pour les utilisateurs et leur compréhension est parfois difficile. Cependant, pour faciliter l'adoption de ces langages de description sémantique des Web services par ces utilisateurs, nous proposons une méthodologie basée MDA pour une génération semi-automatique d'une ontologie de services décrite en OWL-S, étant donné les avantages que procure OWL-S (découverte dynamique, composition, etc.) [21]. Quant à sa validation, nous utilisons des outils ontologiques open source tel que Protégé 2000 [191].

En plus de l'aspect conceptuel traité, notre approche permet aussi de répondre aux besoins exprimés par le partenariat que nous considérons comme des objectifs de la thèse qui sont : la flexibilité, l'ouverture, le respect savoir-faire des partenaires et l'autonomie des Workflows prè-établis) grâce à l'utilisation conjointe des concepts de MDA, ontologies et Web services.

2. La deuxième approche s'appuie sur la conviction suivante *:*

" Interopérer, c'est modéliser, annoter, puis formaliser pour raisonner ".

Partant de cette deuxième conviction, nous proposons une autre approche conceptuelle en nous plaçant à un niveau élevé d'abstraction. Contrairement à l'approche précédente, cette approche est dite "agnostique" dans le sens où elle ignore complètement tout contexte industriel, la technologie Web et les standards qui lui sont associés (y compris les langages de représentation d'ontologies). L'aspect modélisation que nous traitons ici, est présenté dans un cadre unifié et accepté par le partenariat.

Notre deuxième contribution porte ainsi, sur l'utilisation des techniques d'annotation sémantiques pour permettre :

- De résoudre l'hétérogénéité sémantique des modèles de processus rencontrée à différents niveaux afin de favoriser l'interopérabilité. Elle consiste donc, à explorer le concept d'annotation sémantique déjà utilisé pour les documents, les articles, les pages

Web, etc. pour l'étendre vers les modèles de processus. Ces annotations sémantiques se réfèrent à un ensemble commun d'ontologies et se font à différents niveaux :

1. Au niveau des méta-modèles : pour traiter l'hétérogénéité des concepts des différents langages de modélisation de processus tels que XPDL, BPEL4WS, WSFL.

2. Au niveau des modèles : pour traiter l'hétérogénéité des concepts de modèles (au niveau des instances).

3. Au niveau des différents aspects de base des modèles de processus. En effet, ces aspects sont intrinsèquement liés à tout modèle de processus lors de son exécution et qui sont de type informationnel, comportemental ou fonctionnel.

Dans cette approche, la problématique conceptuelle est traitée aussi bien, dans un environnement coopératif homogène qu'hétérogène. Plusieurs types d'annotations ont été proposés :

1. *Annotations structurelles :* elles sont utilisées dans un environnement homogène où la sémantique est considérée comme implicite pour la compréhension des modèles de Workflow entre les acteurs coopératifs. Dans cet environnement, l'utilisation de la notation UML (Unified Modeling Language) est suffisante pour une bonne interprétation de ces modèles.

2. *Annotations sémantiques et lexicales :* elles sont utilisées dans un environnement hétérogène. Elles concernent le domaine d'intérêt commun utilisé par les acteurs coopératifs pour partager les connaissances du domaine d'application. Les annotations sémantiques réfèrent à un ensemble commun d'ontologies telles que ontologie du domaine de référence, ontologie de buts et ontologie des profils. Les annotations lexicales réfèrent à un glossaire comme celui de la WfMC [36] qui sera utilisé dans un contexte homogène et à une ontologie de type thésaurus qui sera employé dans un contexte hétérogène. Ces annotations lexicales permettent en effet, aux acteurs coopératifs d'éviter des ambiguïtés d'interprétation des termes utilisés et avoir une bonne compréhension des modèles lors de l'échange ou de la coopération.

3. *Annotations sémantiques de type informationnel ou comportemental :* Elles concernent les différents aspects d'un modèle de processus. Elle réfèrent aux différentes ontologies liées à chaque aspect de type informationnel, organisationnel, comportemental, etc. Ce type d'annotations est parfois utile dans un contexte d'interopérabilité où les partenaires autorisés sont liés par un contrat de partenariat et qui ont besoin d'un certain degré d'inter-visibilité d'un aspect particulier afin de pouvoir coopérer.

Pour gérer l'hétérogénéité sémantique de ces modèles, l'approche que nous avons préconisée s'article principalement autour de l'élaboration d'un Méta-modèle Commun d'Annotation Sémantique des Modèles de Processus (MCASMP) dont les concepts provient de :

- L'Ontologie Commune des Processus Workflow (OCPW) qui permet d'annoter les différents méta-modèles de processus (ou les langages de modélisation de processus).

L'ontologie OCPW contient les concepts communs qui sont partagés par la plupart des modèles de processus. Elle est considérée comme un médiateur sémantique pour l'annotation sémantique des concepts de ces méta-modèles. Les concepts des méta-modèles sont alors annotés par les concepts de OCPW : *c'est l'annotation des méta-modèles.*

- L'ontologie commune du domaine qui contient les concepts acceptés et partagés du domaine de référence. Elle sert pour annoter la sémantique des contenus des modèles*: c'est l'annotation des modèles*. Dans notre travail, nous utilisons l'ontologie SCOR (Supply-Chain Operation Reference-model) [226] comme une ontologie du domaine de référence.

- L'ontologie commune des buts des modèles de processus qui contient les concepts des buts qui sont communs et partagés par les partenaires coopératifs. Elle nous sert à annoter les objectifs (ou les buts) que devraient atteindre les processus. Son rôle est d'enrichir la sémantique de ces modèles en termes d'objectifs visés par les processus et de permettre ainsi, leur découverte sémantique. Cette découverte est basée sur des objectifs métiers. Dans notre travail, nous utilisons aussi l'ontologie SCOR comme ontologie des buts.

- L'ontologie commune des profils des modèles de processus qui nous sert à annoter les concepts communs des profils des modèles partagés. Ces profils sont exprimés en termes de fonctionnalités des processus. Ce type d'ontologie permet d'enrichir la sémantique de ces modèles en termes de profils qui sont publiés par les partenaires coopératifs pour permettre leur découverte sémantique qui est basée sur ces profils.

- Différentes ontologies communes qui sont liées aux aspects de base des modèles de processus. Ces ontologies nous permettent de mieux gérer la sémantique hétérogène des modèles par une bonne compréhension des informations. Elles sont généralement utilisées pour permettre un certain degré d'inter-visibilité à des partenaires strictement autorisés et liés par un contrat de partenariat.

Cette approche basée annotation sémantique est purement conceptuelle, elle est dite "agnostique" dans le sens où elle ignore complètement toute contexte Web et tous les standards associés. Elle est surtout flexible dans la mesure où le mécanisme d'annotation sémantique est totalement séparé des descriptions sémantiques et indépendant de tout langage de représentation d'ontologies, permettant ainsi à la communauté des développeurs de choisir leur langage ontologique souhaité et également à des acteurs coopératifs d'annoter leurs modèles selon le langage d'annotation sémantique préféré. En outre, elle permet de répondre aux besoins de la coopération que nous avons exprimés en termes d'objectifs fixés par le partenariat (flexibilité, ouverture, savoir-faire et autonomie).

4- Organisation du document

Cette thèse est structurée en deux grandes parties. La première partie est consacrée à un état de l'art du domaine du Workflow avec les différentes formes d'interopérabilité qui existent, les approches proposées et les techniques utilisées pour la mise en œuvre de l'interopérabilité. Elle comporte deux chapitres.

Le chapitre I dresse un état de l'art de notre domaine d'application qui concerne le Workflow et le business process. Le but est donc, d'introduire le lecteur dans notre domaine, en mettant l'accent sur le concept de Workflow et l'expression « processus métier », puis il expose les différents points de vue de ce concept selon les quatre organisations internationales à savoir : la WfMC (Workflow Management Coalition) [30], l'OMG (Object Management Group) [99], le BPMI.org (Business Process Management Initiative) [100] et le W3C (World Wide Web Consortium) [101]. Ensuite, il présente les principaux formalismes de modélisation, ainsi que les différents langages de spécification de Workflow inter-organisationnel avec une étude comparative de ces langages pour terminer avec une synthèse.

Le chapitre II expose les différentes formes trouvées dans la littérature et les approches existantes pour l'interconnexion et la coopération de Workflows inter-entreprises. Nous y analysons les avantages et les inconvénients de chacune de ces approches par rapport à la problématique posée, et aux objectifs fixés dans la thèse. De même, nous présentons les différentes techniques syntaxiques et sémantiques qui sont mises en œuvre pour favoriser de manière générale, l'interopérabilité dans les systèmes d'information. Nous dégageons leurs limites surtout par rapport à l'aspect conceptuel de la solution que nous proposons, ainsi que par rapport aux objectifs fixés dans la thèse.

La deuxième partie est plutôt consacrée à la proposition de deux approches conceptuelles pour l'interopérabilité des modèles de Workflow. Elle comporte deux chapitres. Le chapitre III porte d'abord, sur la forme d'interopérabilité qui présente le plus de flexibilité par rapport aux autres formes, ensuite sur la première approche conceptuelle qui préconise de combiner l'ingénierie des modèles, l'approche basée ontologie, et enfin le concept de Web services. Cette combinaison permet de nous orienter vers le méta-modèle cible qui présente le plus d'avantages par rapport aux autres méta-modèles, et qui est considéré comme le plus approprié pour la représentation sémantique des modèles de processus (ou fragments de modèles), et qui est dans notre travail, le méta-modèle OWL-S. De même, pour permettre un partage sémantique de ces modèles, nous avons choisi le méta-modèle de la plate-forme de coopération le plus adapté, qui présente le plus de flexibilité et qui est pour nous, le modèle SOA et en particulier, la plate-forme des Web services. Ce chapitre se focalise principalement sur la génération semi-automatique d'une ontologie de services en une description OWL-S afin de faciliter son utilisation dans un contexte MDA.

Le chapitre IV présente une deuxième approche conceptuelle, qui cette fois-ci, est basée sur les techniques d'annotation sémantique. L'objectif est de résoudre l'hétérogénéité sémantique des modèles de Workflow (ou modèles de processus) à différents niveaux : au niveau des méta-modèles de processus, au niveau des modèles, au niveau des profils, au niveau des buts (ou objectifs) des modèles et enfin au niveau de leurs aspects qui sont considérés comme des aspects de base qui sont intrinsèquement liés à tout modèle de processus en exécution tels que aspect informationnel, comportemental, organisationnel, fonctionnel et de gestion des ressources. Nous terminons cette deuxième partie de la thèse par un bilan comparatif de ces deux approches selon certains critères que nous avons fixés et qui sont liés principalement à la problématique posée et aux objectifs visés.

Enfin, nous clôturons notre mémoire par un bilan du travail effectué durant cette thèse en donnant les conclusions et quelques perspectives envisagées pour la poursuite de ce travail.

PARTIE 1

ETAT DE L'ART

Chapitre I

Workflow – Présentation, Définitions et Concepts

1. Introduction

Dans ce chapitre, nous essayons de dresser un état de l'art du domaine du Workflow. Ce dernier est très riche et complexe car il chevauche plusieurs domaines de discours impliquant différents champs d'application tels que l'amélioration des processus logiciels [31], [32], la modélisation d'entreprise [33], [34], le génie logiciel [35], les systèmes de Workflows [36], [37], la composition des Web services [30], [10].

Dans la littérature, le concept de Workflow est désigné sous plusieurs termes tels que "modèle de Workflow", "processus métier" (traduit en anglophonie par "business process"), "modèle de processus" ou encore "processus Workflow". Pour clarifier ce concept, nous présentons d'abord, les principales organisations à savoir : la WfMC (Workflow Mangement Coalition) [30], le BPMI (Business Process Mangement Initiative) [100] et l'OMG (Object Management Group) [64] qui ont investi leurs efforts dans ce domaine afin de promouvoir l'utilisation du Workflow grâce à la définition de standards et la connectivité entre les produits Workflows, ainsi que l'utilisation des outils de gestion de processus métiers. Ensuite, nous citons quelques formalismes de modélisation de Workflow, ainsi que les principaux langages de spécification de Workflow dont les concepts sont différents et qui freinent les mécanismes d'interopérabilité sémantique des modèles de Workflow.

Nous terminons enfin, cet état de l'art par une synthèse portant sur une étude comparative de ces langages pour dégager un modèle de spécification de Workflow inter-organisationel, adapté à un travail collaboratif. Dans notre contexte d'interopérabilité, l'étude des différents langages avec la variété de leurs concepts nous semble bien appropriée pour permettre d'extraire un noyau de concepts de base qui sont communs, acceptés et partagés par la plupart des modèles de Workflow.

2. Présentation du Workflow

La technologie du Workflow est née du besoin qu'ont les entreprises d'améliorer la performance de leurs processus métiers, notamment grâce à une maîtrise de l'information, une meilleure coopération des acteurs et une optimisation des processus métiers. Selon plusieurs spécialistes dont [38], [39], le Workflow est orienté vers la gestion et l'automatisation des processus de l'entreprise. L'important est de retenir que le Workflow est la technologie qui propose des outils de coordination, de gestion et d'automatisation des activités des entreprises.

2.1 Origine du Workflow

L'origine du Workflow se situe au niveau des travaux de recherche sur l'optimisation et la rationalisation des processus de l'entreprise, quelles que soient leurs natures et leurs objectifs. La technologie Workflow s'est fortement développée depuis les années 90, en parallèle avec l'émergence les méthodes de management tels que le CPI (Continuous Process Improvement) [2] pour l'amélioration des processus et le BPR (Business Process Reegineering) [1] pour la reconception des processus, d'une part, l'automatisation des processus d'entreprise et l'amélioration de leurs performances, d'autre part. Parmi les avantages qu'elle procure, nous pouvons citer quelques uns :

- ✓ Fiabilité, transparence et rapidité des accès aux données et documents (tout support d'informations),

- ✓ Résolution des problèmes organisationnels et de synchronisation des participants aux processus,
- ✓ Suivi et traçabilité des travaux effectués.

De ce fait, le Workflow est considéré, pour les organisateurs et les gestionnaires, comme un outil de reengineering. Certains auteurs tel que [40] considèrent que l'origine du Workflow provient de l'évolution de la Gestion Electronique des Documents (GED) pour des objectifs d'organisation globale des processus de l'entreprise. D'autres spécialistes par contre, restreignent l'origine du Workflow au domaine de l'automatisation et l'optimisation des processus de '' bureau'' (office automation). Ce domaine est plus général puisqu'il regroupe tout travail administratif, gestion des transactions et GED [41], [42]. Pour ces spécialistes, l'origine du Workflow provient de la gestion électronique des documents (GED) qui a évolué vers des objectifs organisationnels des processus de l'entreprise.

2.2 Définitions

Selon l'encyclopédie Wikipédia en ligne [43]**,** un Workflow (terme anglais) est un flux d'informations au sein d'une organisation. Le terme officiellement recommandé par la Commission générale de terminologie et de néologie est « flux des travaux », ou « automatisation des processus ». Un Workflow peut donc être défini comme un modèle informatique pour représenter un processus de travail. Il permet généralement un suivi et identifie les acteurs en précisant leurs rôles et la manière d'exécuter les activités qui sont assignées à ces acteurs.

Plusieurs définitions ont été apportées dans la littérature au concept de Workflow. Parmi ces définitions, nous en citons uniquement trois :

- On appelle "*WorkFlow*" (traduisez littéralement par "flux des travaux") : *"la modélisation et la gestion informatique de l'ensemble des tâches à accomplir par différents acteurs impliqués dans la réalisation d'un processus métier (aussi appelé processus opérationnel)"*. Le terme "Workflow" pourrait donc être traduit en français par *la gestion électronique des processus métier* [44].

- Un Workflow est " *l'automatisation de tout ou une partie d'un processus d'entreprise au cours duquel l'information circule d'une activité à l'autre, c'est-à-dire, d'un participant (ou d'un groupe de participants) à l'autre, en fonction d'un ensemble de règles de gestion pré-définies "* [36], [45].

- Le Workflow est "*une technologie qui propose des outils de coordination des activités, de gestion et d'automatisation des processus d'entreprises"* [46]. Le Workflow est défini aussi, comme étant un type d'application de Groupware [45].

De ces définitions, nous pouvons retenir que le Workflow est orienté vers la gestion et l'automatisation de tout ou une partie des processus de l'entreprise (processus organisationnels). Généralement, lorsqu'on parle de Workflow, on fait référence à un outil informatique dédié à la gestion des procédures.

Aussi, il peut-être défini comme étant l'automatisation du circuit de l'information ou encore la logistique d'enchaînement des tâches. Il est donc, orienté vers l'organisation d'un travail de

groupe, impliquant la présence de plusieurs personnes interagissant de façon asynchrone, unies autour de la réalisation d'un objectif commun.

En ce qui concerne le rapport entre les méthodes de mangement telles que le BPR (Business Process Reegineering), le CPI (Continuous Process Improvement) et le Workflow, nous pouvons dire simplement que le Workflow propose des outils méthodologiques ou informatiques, qui permettent d'analyser, d'évaluer les processus métiers et de leur fournir un support d'exécution automatisé.

2.3 Workflow Versus processus métier

Dans une même entreprise, un processus d'entreprise peut-être vu sous plusieurs angles. Le qualiticien, le contrôleur de gestion, l'organisateur, le chef de projet et l'informaticien ont leur propre point de vue sur les processus. A chacun son domaine de compétence particulier et donc à chacun son méta-modèle spécifique. Le contrôleur de gestion sera plus intéressé par les notions de coûts et de performance, le qualiticien par la définition des procédures, le chef de projet par l'affectation des ressources et la planification. Quant à l'informaticien, il sera plus intéressé par l'automatisation et la connexion avec des applications ou des composants logiciels. Evidemment, ces points de vue ne sont pas complètements disjoints mais représentent différents aspects d'un même système. Donc, le concept de ″ processus ″ recouvre plusieurs significations selon les acteurs.

ISO (International Organization for Standardization) [47] a défini une typologie des processus d'entreprise qui inclut tous les types de processus d'entreprise tels que les processus d'affaires, les processus de gestion d'investissements, de ressources, de gestion de projet de planification ou de gestion des risques, ainsi que les processus techniques qui recouvrent l'ensemble des processus qui sont directement impliqués dans la production d'un système depuis l'analyse des besoins, en passant par son intégration jusqu'à son retrait.

Selon [48], nous distinguons principalement trois types de processus :

- *les processus matériels* qui se caractérisent par la manipulation, l'assemblage, la livraison, la transformation, la mesure et le stockage d'objets physiques. La mise en ouvre de ce type de processus doit nécessairement aboutir à la production d'objets physiques.

- *les processus informationnels* qui fournissent de l'information. L'infrastructure de base de ce type de processus est fournie par les systèmes d'information de l'entreprise, tels que les systèmes de gestion de base de données, les systèmes de gestion de transactions.

- *les processus métiers* se situent à un niveau conceptuel plus élevé que les deux types précédents de processus de part leur orientation économique [38]. Par suite, un processus métier peut mettre en œuvre d'autres processus, de type informationnel et/ou matériel, si leur réalisation permet d'atteindre l'objectif qui est associé au processus métier. L'expression de "processus métier" que l'on rencontre généralement dans la littérature francophone est la traduction de "business process". D'autres traductions possibles tels que "processus opérationnel" [49], " processus d'entreprise" que l'on retrouve dans le lexique de la Worklow Mangement Coalition [36] ou encore "processus

stratégique"**.** Pour [49], la notion de processus opérationnel adopte un sens plus global et concerne tout processus d'intérêt pour l'entreprise.

Dans littérature, plusieurs définitions ont été données à l'expression de ″processus métier″. Nous en citons quelques unes :

- Selon [50] et [51], un processus métier est *" une collection d'activités consommant des entrées (matériels, finances, données, etc.) et délivrant un ou plusieurs résultats à orientation économique ("busines result") ou à forte valeur ajoutée pour l'entreprise".*

- Une autre définition, qui s'intéresse plus à la structure d'un processus : « *un processus représente l'organisation d'un ensemble finalisé d'activités effectuées par des acteurs et mettant en jeu des entités »* [52]. Pour [53] un processus inter-organisationnel est un processus collaboratif où « les partenaires ont une maîtrise partielle sur ce dernier ».

- Une définition essentielle : *« un processus est un ensemble partiellement ordonné d'étapes exécutées en vue de réaliser au moins un objectif* » [49].

Toutefois, cette expression de ″processus métier″ indique que le processus en question est directement lié au métier de l'entreprise qui le met en oeuvre. Par métier, on comprend l'ensemble des compétences d'une entreprise, son domaine d'activité qui se traduit par l'offre d'un ensemble de ses services, produits ou artefacts dont la consommation permet de se pérenniser et de se développer. Un processus métier peut être interne ou mettre en jeu des entreprises partenaires.

A la limite, si l'on souhaite apporter une nuance entre le Workflow et le processus métier, on pourrait dire que le Workflow contribue à la performance de tous les types de processus qui sont "actionnés" dans l'entreprise et particulièrement aux processus métiers qui sont les candidats potentiels au Workflow.

Pour [54], un Workflow n'est qu'une représentation d'un processus métier dans un format interprétable par la machine. Il est constitué d'un réseau d'activités et de dépendances entre elles, des critères pour spécifier le démarrage et la terminaison d'un processus et des informations sur les activités individuelles (participants, applications, données associées).

Pour mieux comprendre la différence entre les termes ‘‘Workflow’’ et ‘‘ business process’’, nous pouvons dire que le premier s'intéresse plutôt à l'organisation d'un travail (métier ou autre) représenté par la coordination automatisée des tâches. Il correspond à la représentation schématique ou à la modélisation d'un processus métier, alors que le deuxième (business process) représente le métier proprement dit de l'entreprise en termes d'activités, de rôles et de règles métiers.

En ce qui nous concerne, et en en se référant aux différentes définitions qui ont été proposées, nous pouvons explicitement définir un processus Workflow comme étant un ensemble d'étapes. Chaque étape peut-être une activité, une tâche (activité atomique) ou peut faire référence à un sous-processus. Les activités sont exécutées via les conditions de transition. Une activité peut-être manuelle ou automatisée. Elle est réalisée par un rôle qui décrit les compétences et le savoir faire qu'un acteur (utilisateur du Workflow ou participant) doit avoir au sein d'une organisation. L'exécution de ces activités demande l'invocation de ressources (applications

externes, outils informatiques, etc.), la manipulation de données, ainsi que l'accès aux artefacts tels que documents, formulaires électroniques ou feuilles de styles.

Pour des raisons de simplicité d'écriture, nous utiliserons dans la suite du document, les termes : "modèle de Workflow", "modèle de processus", "business process", " processus métier ", " processus d'affaires "ou encore " processus Workflow" comme des termes synonymes. Ces modèles sont par nature, dynamiques et intègrent plusieurs aspects de type informationnel, organisationnel, de qualité, de sécurité, de délai, etc. Certains aspects sont dits "aspects de base" qui sont intrinsèquement liés à tout modèle de processus et auxquels nous nous intéressons particulièrement dans notre travail. Ces aspects sont décrits dans la section suivante.

2.4 Aspects de base d'un Workflow

Dans une entreprise, un modèle ou processus Workflow regroupe plusieurs aspects de base qui peuvent être modélisés et traités indépendamment (figure 1.1) :

- Aspect Organisationnel : il décrit la structure organisationnelle de l'entreprise (département, service, etc.) qui est invoquée pour la réalisation du processus Workflow à travers les concepts suivants : de rôle, de groupe de rôles, d'acteur (ou d'agent), d'équipe, de compétences de chaque membre de l'équipe, de position dans l'équipe ou de responsabilité dans la structure.

- Aspect Ressources : il décrit l'ensemble des moyens mis œuvre pour réaliser l'activité ou le processus (temps prévu, temps réalisé, outil, machine, etc.).

- Aspect Informationnel : il décrit les informations (données, documents, etc.) qui sont manipulées par l'utilisateur (ou acteur) du Workflow pour exécuter l'activité ou le processus.

- Aspect Comportemental : Il correspond à la modélisation de la dynamique du processus Workflow, c'est-à-dire, la façon dont les activités sont chronologiquement exécutées.

.

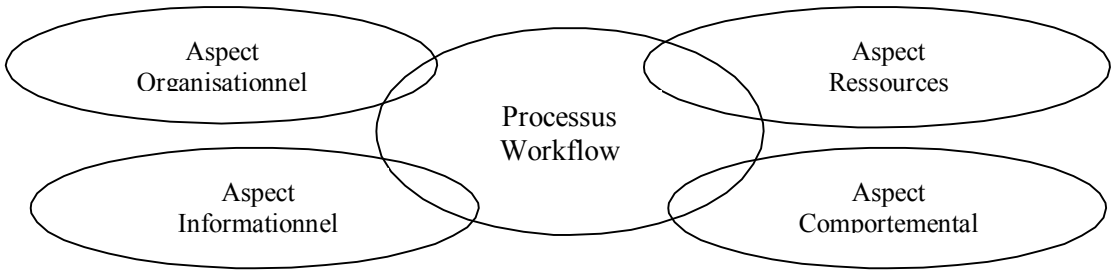

Figure 1.1 : Aspects de base d'un processus Workflow.

Les quatre aspects mentionnés plus haut sont généralement les plus acceptés par la communauté Workflow. Mais, selon le contexte et l'objectif du modèle, d'autres aspects [55] peuvent être intégrés tels que aspect sécurité, aspect transaction, aspect qualité de service. En se référant à la figure 1.1, nous pouvons dire qu'un processus Workflow n'est autre qu'un vecteur d'intégration entre les différentes dimensions ou composantes de l'entreprise (figure 1.2).

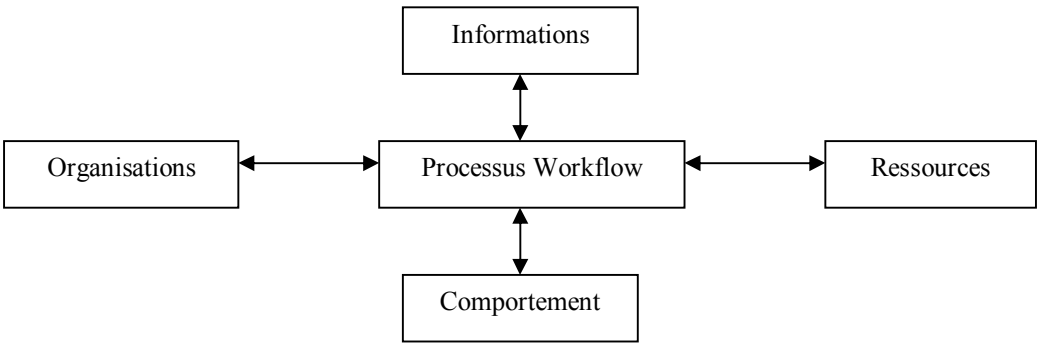

Figure 1.2 : Intégration des composantes de l'entreprise par le processus Workflow.

Etant donnée cette diversité de concepts de "Workflow et de "processus" plusieurs organisations internationales ont investi le domaine du Workflow afin de clarifier tous les concepts qui lui sont rattachés pour proposer ainsi, un ensemble de normes. Il s'agit principalement de la WfMC Workflow Management Coalition, du BPMI (Business Process Management Initiative**)** et de l'OMG (Object Management Group) dont nous décrivons brièvement les efforts.

2.5 Le Workflow vu par la WfMC

L'effort majeur pour standardiser le Workflow a été fait par la Workflow Management Coalition (WfMC) [30]. C'est une organisation internationale de normalisation dans le domaine du Workflow créée en 1993 regroupant des chercheurs, des industriels de l'informatique, des utilisateurs et des experts du domaine. Sa mission est de promouvoir l'utilisation du Workflow grâce à la définition de standards portant sur la terminologie Workflow, l'interopérabilité et la connectivité entre les produits Workflows.

Plusieurs documents ont été publiés par la WfMC. Les définitions suivantes proviennent d'un glossaire [36] comportant tout le vocabulaire de base utilisé dans ce domaine. Les standards définis par la WfMC pour le développement d'applications Workflows, servent notamment à résoudre les problèmes d'interopérabilité entre des systèmes Workflows, mais également à définir les caractéristiques de ces systèmes. Les documents publiés par la WfMC couvrent plusieurs aspects et peuvent être considérés comme des références en la matière. Nous basons d'ailleurs nos définitions sur ces travaux pour définir quelques concepts de base afin de familiariser le lecteur avec notre domaine.

2.5.1 Concepts de base rattachés

Le Workflow s'accompagne de concepts fondamentaux qu'il est nécessaire de présenter. Pour mieux comprendre, il est intéressant de les définir. Les définitions que nous donnons ci-après, proviennent du glossaire établi par la WfMC [36] **:**

- *Système de gestion de Workflow* (SGWF) : C'est un système complet qui sert à définir, gérer et exécuter des procédures grâce à des programmes dont l'ordre d'exécution est pré-défini dans une représentation informatique de la logique de ces procédures. Donc, c'est un système informatique dédié à la gestion des processus.

- *Moteur de Workflow* : C'est un service logiciel qui fournit tout ou une partie de l'environnement d'exécution d'un Workflow.

- *Activité* : c'est une étape d'un processus (manuelle ou automatisée). C'est la description d'un travail représentant une étape logique d'un processus. Elle fait appel à un ensemble de tâches. Comme exemple nous pouvons citer : ouverture de courrier comme une activité manuelle, et le remplissage d'un formulaire électronique comme activité automatisée.

- *Rôle* : Groupe d'utilisateurs (rôle organisationnel). C'est la liste d'attributs de compétences, de savoir-faire qu'un acteur (participant) possède et met en pratique. Ce rôle définit la position de l'acteur dans l'organisation. Il peut-être tenu par un ou plusieurs acteurs du Workflow. Comme exemple, nous pouvons citer : rôle de superviseur, rôle d'administrateur. Donc, un acteur doit assurer un rôle pour accéder aux tâches mises à sa disposition, et les exécuter.

- *Acteur* : C'est une entité (personne ou matérielle) chargée de réaliser l'activité via le rôle qui lui est attribué. Donc, un acteur est identifié par le rôle qui lui est affecté. Parfois, il est désigné par les termes d'intervenant, agent ou utilisateur.

- *Tâche :* C'est la représentation d'un travail à effectuer par un acteur de Workflow dans le cas d'une instance d'activité. Elle correspond à une activité atomique ou une unité de travail, appelée parfois bon de travail.

- *Données :* une donnée est objet créé ou modifié pendant l'exécution d'une activité ou un processus. On parle parfois d'artefact qui correspond à des documents, formulaires électroniques ou codes ayant une structure bien identifiée.

- *Conditions de transition :* Une condition de transition correspond à une condition de routage d'une activité. Elle définit le critère régissant la progression ou le changement d'état d'une activité ou bien le passage à l'activité suivante lors de l'exécution du processus. Une condition peut s'exprimer sous la forme d'une expression logique en utilisant les opérateurs de contrôle de flux des activités tels que And Join, And Split, Or Join, Or Split, XOR Join, XOR Split.

- *Applications Externes :* Une application externe est une application informatique dont l'invocation est nécessaire à la réalisation de l'activité. Elle peut-être une application bureautique qui peut être un traitement de texte, un tableur ou un service groupware complémentaire (courrier électronique, forum, agenda de groupe, etc.) Parfois, on parle de ressources, si l'application invoquée n'est pas uniquement informatique.

2.5.2 Le Modèle de référence du Workflow

Parmi les principales préoccupations de la WfMC, il s'agit de citer l'effort consenti pour la standardisation et l'interconnexion des systèmes Workflows. En particulier, il est important de souligner l'établissement d'un modèle de référence du Workflow [56], c'est-à-dire, une référence dans l'univers Workflow. Ce modèle est basé sur l'hypothèse suivante : tous les systèmes de gestion de Workflow reposent sur les mêmes composantes génériques qui interagissent selon diverses modalités.

Pour établir l'interopérabilité entre plusieurs produits Workflows, la WfMC a établi d'abord un glossaire [36] contenant tout le vocabulaire de base et la terminologie commune utilisée dans ce domaine, puis a défini un modèle de référence contenant des standards d'interface et

d'échange de données. Ce modèle définit les spécifications de l'intéropérabilité entre des moteurs de Workflow hétérogènes (interface 4 de la figure 1.3) [57] permettant à des systèmes Workflow hétérogènes de s'échanger des modèles de Workflow et par suite répartir l'exécution des Workflow sur différents sites et systèmes.

Mais, comme un système Workflow est constitué d'un regroupement de plusieurs technologies, la coalition a définit cinq composantes faisant elles-mêmes l'objet de standards, et cinq interfaces (figure 1.3) :

- L'interface 1 définit un méta-modèle de définition de processus associé à un langage qui est le XPDL (XML-Process Definition Language) [4], et ensemble d'APIs pour l'échange de modèles de Workflow. Elle est désignée sous le terme d'interface d'import/export de définition de processus qui devrait fournir un format d'échange commun aux différents types d'informations.
- L'interface 2 définit une API pour accéder aux fonctions du moteur Workflow (démarrer un processus, terminer une activité, etc.). Des modalités de communication sont donc, ouvertes entre le moteur Workflow et le client Workflow.
- L'interface 3 définit les communications avec les applications externes invoquées par le système de gestion de Workflow pour automatiser une activité, en totalité ou en partie, ou pour assister un acteur du Workflow dans la réalisation d'une tâche. Par exemple: appel d'un service de messagerie, appel d'un outil bureautique.
- L'interface 4 définit une spécification pour l'intéropérabilité entre des moteurs de Workflow pour permettre à différents systèmes de gestion de Workflow de travailler ensemble sur les mêmes activités.
- L'interface 5 définit les évènements historiques au cours de l'exécution d'un processus Workflow. Cela permettra de suivre le déroulement des procédures Workflow et d'obtenir une vision complète de l'état d'un Workflow cheminant à travers une organisation, indépendamment des systèmes Workflows mis en œuvre.

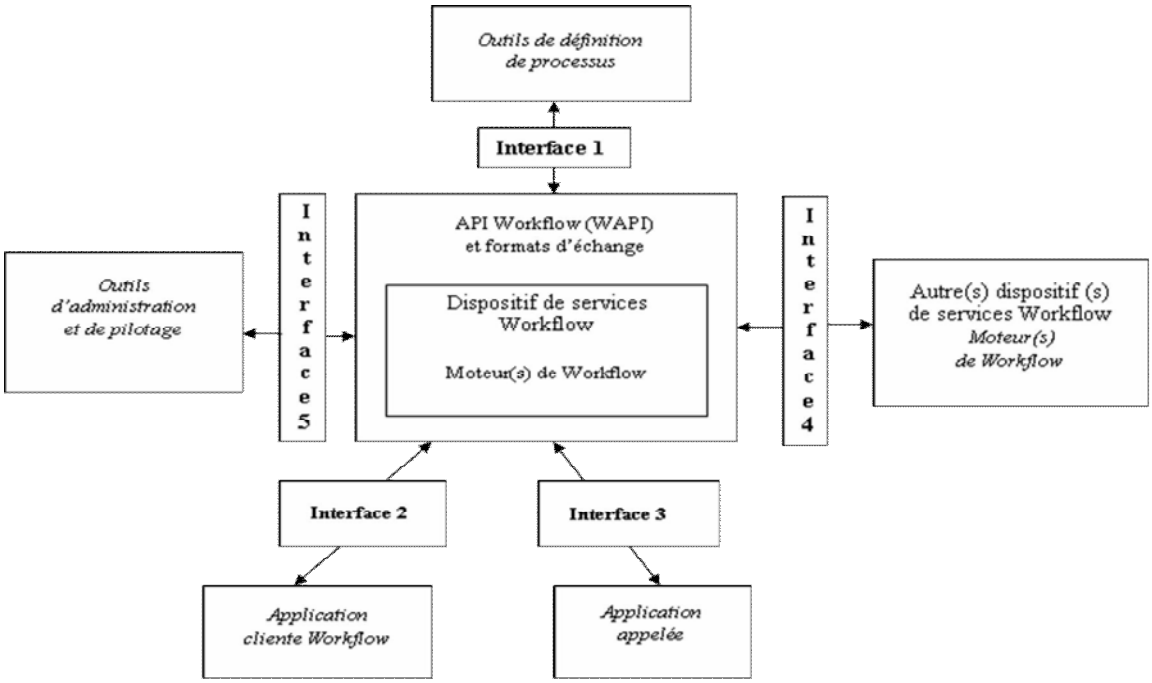

Figure 1.3: Modèle de référence du Workflow de la WfMC [56].

2.5.3 Architecture générale d'un système de gestion de Workflow

La partie concernant la définition du Workflow ne saurait être complète sans un aperçu du principe de fonctionnement des systèmes de gestion de Workflow (SGWFs). Globalement, ce type de système est composé de trois parties complémentaires [30] :

1. Le "build time" qui concerne la modélisation des Workflows. Cette partie est essentiellement composée d'un outil permettant la modélisation des Workflows, sous une notation existante ou propriétaire, le plus généralement sous forme graphique. Un deuxième composant est chargé de transcrire les modèles obtenus en entités compréhensibles par la partie chargée de l'exécution : le *''run time ''*. Selon les systèmes, le '' build time'' peu également être composé d'un '' traducteur'', permettant d'importer des modèles réalisés sous forme des applications existantes (systèmes de modélisation et d'analyse, par exemple), et de les traduire au format compris par le système.

2. Le ''run time '' ou ''environnement d'exécution'' qui est chargé de la gestion complète des modèles de Workflow établis dans la première partie. Cette gestion comprend l'exécution des Workflows, la distribution des tâches aux rôles appropriés, la mise à disposition de l'ensemble des données et des outils nécessaires, la superposition et le contrôle de la cohérence, etc. Cette partie se décompose en fait en plusieurs autres composants, parmi lesquels, il convient de citer :

    - Le moteur Workflow qui est véritablement chargé de la gestion des processus métiers.
    - Le gestionnaire des listes de tâches, chargé de répartir les activités en fonction de leur attribution aux rôles.
    - Les listes de tâches (''Worklist'') qui sont les récipients attribués à chaque rôle dans lesquels le moteur place les tâches à réaliser, par ordre de priorité et en leur associant les données et les outils appropriés.
    - Les outils d'administration et de contrôle.

3. Enfin, la troisième partie concerne des outils et des fonctions d'APIs qui permettent au système de '' s'ouvrir '' vers des applications extérieures, en particulier vers d'autres systèmes Workflows. Les fonctions d'APIs permettent également de customiser le système et ses services en fonction de leurs besoins spécifiques.

Les caractéristiques des systèmes de gestion de Workflow (SGWFs) sont données par la figure suivante :

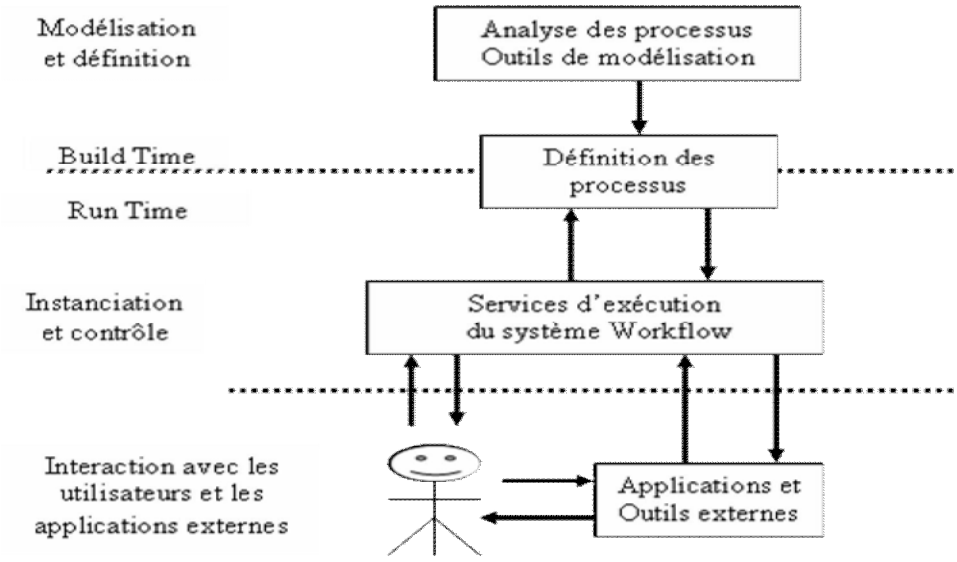

Figure 1.4 : Caractéristiques des systèmes de gestion de Workflow [30].

Au fait, il existe plusieurs types de systèmes, chacun étant conçu selon ses concepts spécifiques, en plus des concepts de base qui sont le plus souvent invariants.

2.6 Le Workflow vu par le BPMI

Partant du constat que l'amélioration de la performance de l'entreprise passe par la maîtrise de ses processus métiers, un consortium d'entreprises a été crée par Intalio regroupant plusieurs leaders du marché tels que IBM, BEA, Rational, Mega International, Vitria Technology. Il s'agit de l'organisation BPMI.org (Business Process Management Initiative**)** [59]. Son objectif est de promouvoir le développement de systèmes orientés processus en complétant l'offre des spécifications favorisant l'interopérabilité d'applications existantes et celles permettant la collaboration inter-entreprises. Aussi, il est focalisé sur la définition des standards dans le domaine du Workflow pour la conception, le déploiement, l'exécution, la maintenance et l'optimisation des processus métiers d'une entreprise.

Selon le BPMI, un processus métier est une chorégraphie d'activités incluant une interaction entre participants sous la forme d'échange d'informations. Les participants peuvent être : des applications, services du système d'information, des acteurs humains ou d'autres processus métiers.

Pour les partisans de BPMI, le concept de Workflow est équivalent au terme de ‘‘business process ’’ ou ‘‘ processus métier ’’. Pour y parvenir, le BPMI.org a défini des standards basés sur la formulation XML pour la gestion de ces processus. Pour pouvoir exécuter ses processus, le BPMI a proposé l'utilisation de BPM qui signifie «Business Process Management» comme solution technologique permettant d'implanter les processus métiers d'une entreprise.

Pour le BPMI, le BPM est plus large que celui des SGWFs. En particulier, il permet de modéliser la collaboration entre de multiples processus métiers, gérés de façon indépendante. Le participant à un processus peut être non seulement un utilisateur mais aussi une application ou un autre processus métier. Enfin, le BPM est centré sur des services, et non pas sur des tâches ou sur des documents, tous deux considérés comme des détails d'implantation d'un processus métier.

Une des similitudes entre le BPM et le Workflow est que les deux domaines permettent la mise en place de processus métiers complexes, pratiquement sans avoir recours à la programmation. Cependant, pour les partisans du BPMI, le BPM va bien au-delà des fonctionnalités traditionnelles de Workflow. Il couvre plus largement la problématique de l'automatisation des processus métiers. Il s'appuie pour cela sur les nouvelles technologies tels que les Web services, ainsi que les différentes normes et langages liés à la définition, la modélisation et l'exécution de processus. L'avantage du BPM est donc, de pouvoir mélanger les concepts : Workflow et intégration.

- **Architecture d'un BPMS**

Le BPMS (Business Process Management System) [60] est un système de gestion des processus métiers. Il s'agit d'une plate-forme intégrant des outils couvrant toutes les étapes du cycle de vie des processus. L'architecture d'un BPMS comprend généralement les éléments suivants :

- Un outil de modélisation de processus, permettant de modéliser à l'aide d'une interface graphique les processus métiers de l'entreprise. en utilisant une interface utilisateur facile qui apporte une grande assistance dans la définition des processus complexes. Une fois définis, les processus sont déployés sur le système d'information.
- Un moteur d'exécution (moteur de Workflow) chargé d'instancier les processus et de stocker le contexte et leur état dans une base de données relationnelle.
- Un outil de contrôle de processus permettant la surveillance et le suivi en temps réel des processus afin de permettre une amélioration facilitée des processus et une meilleure communication entre les acteurs. Des outils de pilotage basés sur des indicateurs précis et pertinents offrant des tableaux de bord pour permettre de prendre rapidement les bonnes décisions. On parle ainsi de BAM (Business Activity Monitoring) pour désigner la notion de contrôle du déroulement des processus de l'entreprise
- Un outil d'analyse et de création de rapport sur les processus : Les analystes d'affaires ont besoin d'indicateurs complets sur les différentes instances des processus qu'ils suivent (en cours ou terminées). Cela leur permet de comparer les durées d'exécution des activités, et d'améliorer leurs processus de manière continue.

Dans un objectif de standardisation, l'organisation BPMI a proposé, un ensemble de normes dont notamment [32], [62], [10] :

- BPMN (Business Process Modeling Notation) : elle définit une notation graphique permettant aux analystes métiers de modéliser de façon standard les processus de l'entreprise. Son objectif est de faciliter la compréhension et l'échange de modèles de processus métiers entre tous les acteurs impliqués. Cette notation a été conçue en s'inspirant des meilleures pratiques de modélisation des processus connues, tels que les diagrammes de flux ou les diagrammes de collaboration.

- BPML (Business Process Modeling Language) **:** le BPML est un méta-langage basé sur XML (Extensible Markup Language) permettant de définir des processus métiers intra et inter-entreprises. La spécification BPML s'inspire des efforts de WSFL [6], WSCL [8] et XLANG [5]. Elle est donc, plus générale en prenant en compte en particulier, les notions de transactions et de contrôle implicite de la succession des activités au moyen des données. Il comprend des spécifications pour gérer : les transactions locales ou distribuées (de façon synchrone ou asynchrone), les flux de données, les messages, les

événements programmés, l'exécution concurrente, les règles de gestion métiers, la sécurité et enfin les exceptions. Ainsi BPML fournit un modèle d'exécution abstrait servant à l'élaboration de processus métiers collaboratifs, transactionnels et sécurisés. Il est indépendant de tout environnement d'exécution.

Dans le cas d'applications et/ou de processus inter-entreprises (fournisseurs, partenaires, clients, etc.), BPML permet la définition d'une interface publique commune aux entreprises partenaires, décrite selon des standards du marché comme ebXML, RosettaNet ou encore BizTalk. Cette interface commune sert de passerelle entre l'implantation privée des processus de chacune de ces entreprises. Toutefois, BPML est remplacé par le langage BPEL4WS (Business Process Execution Language for Web Services, appelé aussi, BPEL Business Process Execution Langage) qui est une spécification de standards pour l'orchestration des Web services.

2.7 Le Workflow vu par l'OMG

Pour l'OMG (Object Management Group) [63], [64], un Workflow est un objet CORBA (Common Object Request Broker Architecture) [65], [66] et l'interopérabilité des Workflows est considérée comme une intégration d'objets CORBA. Les spécifications CORBA du Workflow sont basées sur les standards définis par la WfMC et fournissent une base stable d'intégration de la technologie Workflow à CORBA.

CORBA fournit, entre autres, une terminologie commune dans le domaine des objets répartis, un modèle de référence standard, des interfaces et des protocoles communs. En effet, l'architecture CORBA présente les objets répartis selon quatre types :

1. les objets applicatifs sont des objets CORBA propres à l'application et non concernés par la standardisation de l'OMG ;
2. les services communs sont des objets indépendants de l'application : nommage, vendeur, transactions, événement, etc. ;
3. les interfaces de domaines sont des interfaces spécifiques à un domaine d'application (par exemple, la médecine, la finance, les télécommunications) ;
4. les utilitaires communs sont des services de plus haut niveau fournissant des fonctionnalités utiles dans de nombreuses applications (par exemple, les interfaces utilisateurs, la gestion de données, l'administration des systèmes, etc.).

Chacune des trois organisations internationales (WfMC, BPMI, W3C) a un point de vue différent sur le concept de Workflow. Toutefois, des efforts ont été investis afin de rapprocher et de réconcilier ces trois points de vue. Ainsi, le BPMI a établi en 2002 un joint-venture avec la WfMC et en juin 2005, le BPMI. et l'OMG ont annoncé la fusion de leurs activités de gestion de processus métier pour fournir des standards industriels. Le groupe combiné s'est baptisé " the Business Modeling & Integration (BMI) Domain Task Force (DTF) ".

2.8 Le Workflow vu par le W3C

Le W3C (World Wide Web Consortium) [43] est un organisme de standardisation à but non-lucratif, chargé de promouvoir la compatibilité des technologies du Web. Il est impliqué dans le développement de la technologie des Web services et recommande l'utilisation des standards à travers le Web dans un contexte industriel.

Pour cet organisme, la modélisation des processus métier (ou business process) correspond à l'évolution naturelle des Web services. *Un processus métier est considéré alors comme un assemblage de Web services techniques et métier.* Dans le contexte du Web, il s'agit en d'autres termes d'organiser les Web services entre eux, de les composer, les orchestrer ou les enchaîner. On parle de « composition » de Web services, ''d'orchestration'' d'activités et parfois de ''chorégraphie'' de services. Pour le W3C, le concept de Workflow décrit uniquement l'enchaînement ou représente la composition des Web services.

Afin de faciliter les échanges B2B (Business to Businees) entre des partenaires commerciaux à travers le Web, le W3C a défini différentes couches composant un Web service telles que couche de découverte de Web services, couche de description de Web services La plus haute couche, appelée couche métier est représentée par ebXML, BPML ou BizTalk. Elle permet de mutualiser les fonctions utiles à certains métiers de l'entreprise.

Pour exécuter des processus métiers, le W3C recommande des langages de composition de Web services permettant leur définition, puis leur exécution grâce à des outils ou à des moteurs d'exécution faisant partie d'un BPM (Business Management Process) tel que Biztalk Server de Microsoft.

3. Classification des Workflows

La diversité des applications Workflows et en particulier la variété des processus métiers ont incité à diviser le marché des logiciels Workflows en plusieurs secteurs dépendants sur la valeur du processus qui doit être géré et sur le processus s'il est répétitif ou non. Cette diversité nous amène à répertorier quatre types de Workflows. Chaque type de Workflow est décrit selon l'objectif fixé et ses besoins spécifiques.

Plusieurs types de classification ont été proposés [44], [46] : par domaine d'application, par objectifs, par degré de structuration du Workflow, etc. Seulement la classification par domaine d'application est la plus répandue. Elle est reprise par beaucoup d'auteurs. Elle propose en effet, de distinguer quatre catégories de Workflows [45] :

- Les Workflows Administratifs,
- Les Workflows de Production,
- Les Workflows Ad-hoc,
- Les Workflows Collaboratifs.

3.1 Les Workflows Administratifs

Ce sont des Workflows orientés vers la gestion des processus de type administratif dont le but d'alléger les tâches de bureau, où les erreurs sont souvent humaines. Ces systèmes Workflows permettent de lier à une tâche administrative les documents et les informations nécessaires à la réalisation de la tâche par un acteur humain. Ils prennent en charge le routage des documents et le remplissage des formulaires, par exemple, le remboursement des notes de frais.

3.2 Les Workflows de Production

Ces Workflows s'appliquent à des processus opérationnels répétitifs et critiques pour la performance globale de l'entreprise ou de l'unité organisationnelle qui est responsable. La

principale différence réside dans la complexité de la réussite des processus et des tâches, ainsi que dans l'enjeu que représente leur réussite. Enfin, la réalisation du Workflow correspond directement aux objectifs et au travail de l'entreprise. Ce type de Workflow crée de la valeur ajoutée pour l'entreprise. On le trouve généralement dans les organismes financiers ou dans des compagnies d'assurances, par exemple, le traitement des réclamations déposées par des clients d'une société d'assurance.

3.3 Les Workflows Ad-hoc

Ces Workflows automatisent des procédures d'exception, donc occasionnelles, voire uniques. Ils correspondent à des processus peu ou non structurés dont l'ordre et le temps exact de réalisation des tâches ne sont pas établies au préalable. A la limite les tâches elles-mêmes de ce type de processus ne sont pas connues au départ. Le choix de routage des tâches, et la nature des tâches sont décidées au fur et à mesure de l'exécution, et il est donc très difficile d'automatiser de tels processus.

Donc, La réalisation de ce type de processus peu impliquer à chaque fois l'exécution d'un nouvel enchaînement des tâches, voire de nouvelles tâches. Le système intervient essentiellement pour gérer les échanges entre des rôles (qui font référence uniquement à des acteurs humains), gérer l'accès aux sources d'informations et fournir l'historique du Workflow. Ce sont des Workflows très complexes, indéterminés pour lesquels ils n'existent pas de modélisation pré-définie et sont généralement évolutifs. Comme exemples, nous pouvons citer : rédaction collective d'un rapport d'expertise, recrutement d'une compétence particulière.

3.4 Les Workflows Collaboratifs

Ces Workflows appelés aussi coopératifs, sont dédiés au support de travail de groupe tels que la conception, la gestion de projet ou la résolution des problèmes faisant appel à différents niveaux d'expertise. Ces Workflows permettent de réunir un certain nombre d'intervenants dans le but d'atteindre un objectif commun. Les clients du processus sont souvent y associés. Les tâches gérées sont généralement complexes et les acteurs impliqués dans le processus doivent être très compétents (dans la spécialité qui est la leur). Ces Workflows sont faiblement structurés et peu répétitifs. Par contre, ils sont à forte valeur ajoutée par rapport à l'entreprise qui les met en place.

Certains spécialistes classent les Workflows Ad-hoc et les Workflows Collaboratifs dans la même catégorie, par contre, d'autres experts les distinguent dans la complexité des tâches à réaliser et dans la valeur ajoutée : peu de complexité et faible valeur ajoutée pour les premiers, grande complexité et forte valeur ajoutée pour les deuxièmes. Comme exemple, nous pouvons citer la gestion d'un projet.

4. Modélisation d'un Workflow

La modélisation est un élément de base de la technologie Workflow, en particulier dans le cas des systèmes basés sur une modélisation des processus ou sur l'enchaînement des activités. Cinq éléments à prendre lors de la modélisation d'un Workflow :

- L'activité qui symbolise une étape d'un processus ;
- Le rôle qui accomplit l'activité ;

- Les acteurs affectés à ce rôle ;
- La route (ou chemin) représentant les transitions entre les activités ;
- L'objet qui transite par les activités et les chemins.

Concernant l'implémentation d'un Workflow, il faut définir trois variables : la métaphore des « 3R » : Routes, Règles, Rôles.

- Des *Rôles* pour accomplir les activités (indépendamment des personnes);
- Des *Règles* pour formaliser la coordination entre les activités et les rôles nécessaires à l'accomplissement du processus dans l'organisation ;
- Des *Routes* pour organiser la dynamique des processus. Elles déterminent les itinéraires d'un Workflow, c'est-à-dire, les chemins que prennent les différents résultats d'une activité à l'autre, d'un rôle à l'autre et donc, d'un participant à l'autre. Les différents types de routage utilisés sont : séquentiels, parallèles, conditionnels, boucles. Les routes d'un Workflow sont liées aux règles de coordination implémentées dans une application de Workflow.

Plusieurs formalismes de modélisation et langages de spécification ont été proposés dans la littérature [67]. [45], [46]. [72]. Chacun définit ses concepts spécifiques au domaine ciblé et selon les objectifs définis par leurs concepteurs [68] tels que :

- Faciliter la compréhension humaine.
- Assister l'optimisation.
- Assister la gestion du processus.
- Automatiser l'assistance à l'exécution.
- Automatiser le contrôle de l'exécution.

4.1 Formalismes de modélisation

La plupart des systèmes commerciaux proposent leurs propres formalismes en utilisant une notation graphique et leurs propres méthodes qui, toutefois, se basent sur des concepts connus. Donc, plusieurs produits Workflows différents ont été, ainsi, développés par différentes entreprises en adoptant leurs propres formalismes de modélisation. Ces formalismes ont été élaborés et développés dans les années 70 et 80 (ou même avant) dans des domaines variés telles que les bases de données, le développement du génie logiciel. Aussi, certains d'entre-eux ont été repris ou réadaptés aux besoins et objectifs du Workflow (simplicité d'utilisation, facilité de manipulation, compréhension aisée). Par contre, d'autres ont créés spécifiquement pour ce domaine. Chaque formalisme a donc, ses avantages et ses inconvénients.

Parmi ces formalismes**,** nous citerons brièvement quelques uns dans ce document parce que leurs descriptions sont trop volumineuses :

- Les réseaux de pétri qui s'expriment sous la forme de graphes orientés, composés de places, d'arcs et de transitions. En général, les places correspondent aux activités du processus à modéliser, et les arcs sont associés à des conditions de transitions ou à des flux d'informations. Les transitions représentent les évènements ou les conditions à vérifier pour avancer dans le processus. Fortement utilisé dans le domaine industriel, il sert à traiter les problèmes de synchronisation d'activités. Les notions graphiques sont donc : les places (noeuds), les arcs (flèches) et les transitions (contrôles).

- La méthode SADT (Structured Analysis Design, Technique) qui utilise deux types de diagrammes duaux : les actigrammes et les datagrammes. SADT est basé sur le concept de « boîte », qui peut représenter une activité (actigramme) ou une donnée (datagramme). Dans le premier cas, la boîte permet de présenter les données en entrée et en sortie, les ressources et les données de contrôle.

- La modélisation à base de déclencheurs (TMW : Trigger Modeling Workflow) qui se base sur une modélisation événementielle des Workflows, c'est-à-dire, que l'occurrence d'un ou de plusieurs événements est un élément déclencheur d'activité. En d'autres termes, un événement survient toujours suite à la réalisation d'une activité et y est intimement lié. Le méta-modèle proposé par les concepteurs de la méthode est inspiré de celui proposé par la WfMC.

- La méthode Communication/Action qui se base sur un concept bien précis : la communication. Elle vise un objectif bien déterminé : la satisfaction du client. Pour cette méthode, un modèle de Workflow correspond à une transaction entre un client et un fournisseur. La transaction ne s'arrête que si le client est satisfait du produit livré par le fournisseur avec lequel il est en relation.

- La méthode OSSAD (Office Support System Analysis and Design) qui se base sur deux types de modèles ossadiens : le modèle abstrait (une seule représentation graphique) représentant les objectifs de l'organisation, c'est-à-dire, ce qui devrait se faire. La fonction est le concept de base de ce modèle, et le modèle descriptif représente les moyens humains et technologiques mis en œuvre pour réaliser les activités. Le rôle est le concept de base du modèle descriptif. Cette méthode utilise son propre formalisme graphique.

- La méthode STRIM (System Technique for Role and Interaction Modeling) qui est une méthode centrée sur les rôles et leurs interactions, est basée sur une technique de notation appelée RAD (Role Activity Diagram). Ce diagramme montre les rôles et les interactions entre les activités.

- La notation unifiée objet UML qui présente beaucoup d'avantages par rapport aux autres formalismes en intégrant plusieurs types de diagrammes : de classes, de cas d'utilisation, d'activités, d'états de transition, de séquence, etc. Dans le contexte Workflow, nous utilisons particulièrement trois types de diagrammes : les diagrammes de cas d'utilisation, qui permettent de distinguer rapidement les principaux sous-processus et rôles du Workflow. Les diagrammes d'activités qui sont les mieux adaptés pour la modélisation de l'ensemble des éléments du modèle tels que les rôles, les évènements, les contrôles de flux et les flux de données. Les diagrammes de classes qui permettent de représenter les entités du domaine modélisé, leurs propriétés et leurs relations qui les unissent.

Pour mieux concrétiser les choses, nous présentons un exemple simple d'un Workflow concernant un processus de commande d'achat qui est représenté par un réseau de pétri (figure 1.5). Il s'agit au fait, d'une commande d'article passée par un client à une entreprise donnée. Le Workflow se compose de plusieurs activités :

- La vérification de la disponibilité de la commande dans le stock ;

- Si l'article n'est pas disponible, l'activité suivante consiste uniquement à envoyer une lettre d'excuse au client;
- Si l'article est disponible, il s'agit alors d'envoyer une facture au client et l'ordre de préparation du colis;
- Si la facture est payée dans le délai prévu, l'ordre est envoyé aux convoyeurs pour la livraison du colis;
- Dans le cas contraire, il faut annuler la commande.

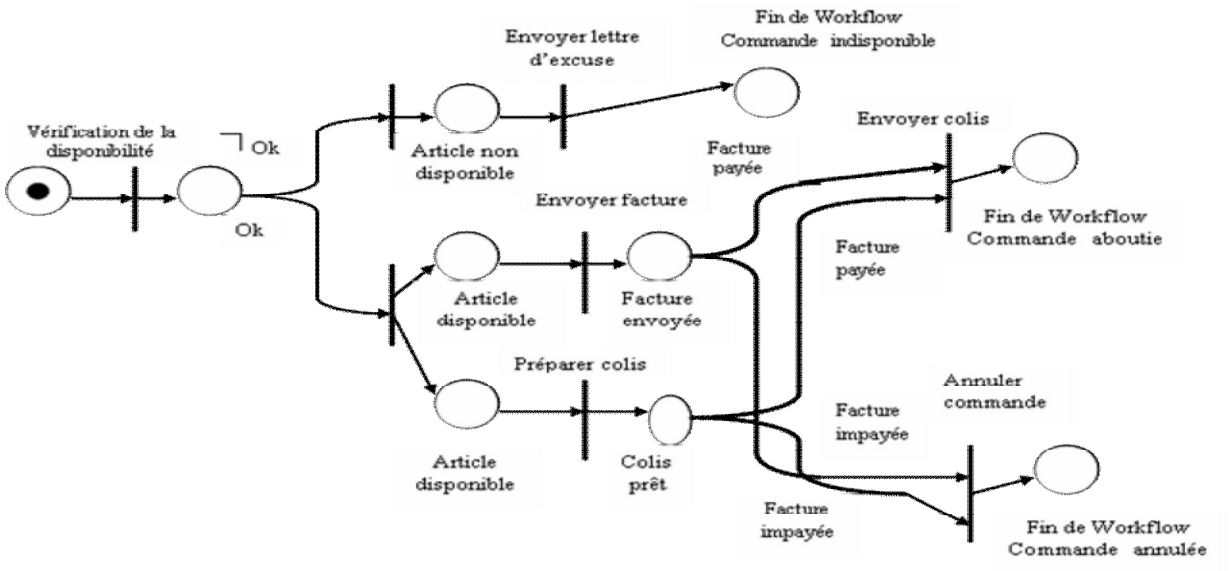

Figure 1.5 : Modélisation d'une commande d'achat à l'aide d'un réseau de pétri.

4.2 Langages de spécification de Workflow

Beaucoup de langages de spécification de processus métiers sont apparus ces dernières années (XPDL, WSFL, XLANG, etc.). Chacun adopte sa terminologie particulière et définit ses concepts spécifiques en fonction des objectifs visés par le méta-modèle de Workflow, du domaine d'intérêt ou du choix de représentation. Ces spécifications sont techniques et sont basées sur le langage XML.

4.2.1 XPDL

XPDL (XML Process Definition Language) [4] est un langage proposé par la WfMC qui permet de définir des processus et utilise le langage XML comme support. XPDL est un méta-modèle de définition de processus (figure 1.6). Ce dernier se positionne comme un format d'échange entre des outils de modélisation et des moteurs d'exécution de Workflows. C'est un langage suffisamment large pour qu'il puisse être adopté par différents vendeurs de systèmes de gestion de Workflow. La figure suivante montre le méta-modèle de définition de processus proposé par la WfMC.

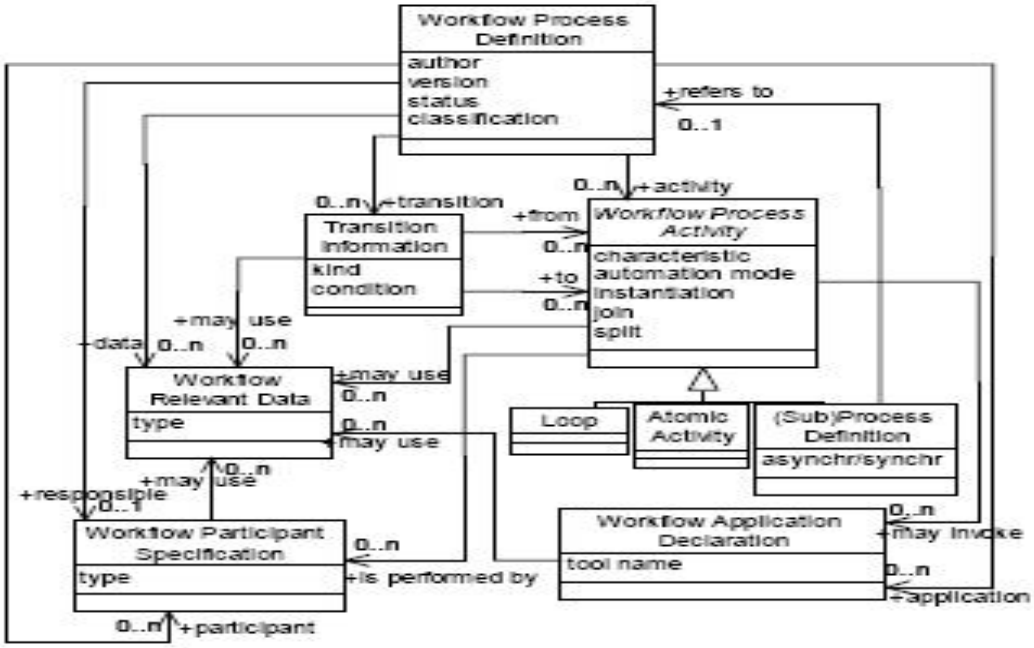

Figure 1.6 : Le méta-modèle de définition de processus de la WfMC [4].

Le *Workflow Process* représente le processus complet. Il est divisé en plusieurs activités (*Workflow Process activity*) qui peuvent être atomiques (tâches élémentaires) ou être des appels vers des sous-processus. Ces activités sont ordonnées grâce aux informations de transition (*transition Information*) qui peuvent porter sur les données pertinentes pour le Workflow (*Workflow Relevant Data)*. Ces données sont typées. Un ensemble de types de base ayant été défini (chaîne de caractères, entier réel, date). Les activités sont réalisées par des ressources humaines ou des entités organisationnelles définies par l'intermédiaire des participants (*Workflow Application Specification*). Elles peuvent invoquer des applications externes via des déclarations d'application (*Workflow Application Declaration)*.

Pour pallier au manque de richesse lié à la définition de la structure organisationnelle, la spécification prévoit la possibilité d'une mise en relation avec un méta-modèle organisationnel. Cela permettrait de formaliser des règles bien plus complexes que ce permet le modèle de base.

4.2.2 WSFL

WSFL (Web Services Flow Language) [6] est un dialecte XML développé par IBM, destiné à la composition des Web services Web. Il s'appuie sur le langage WSDL et offre deux types de composition :

- Modèle de flux *(flow model)* : Ce modèle correspond au processus, c'est-à-dire, à la description explicite de la succession des étapes et de l'enchaînement des appels aux opérations des Web Services. Il représente le processus métier proprement dit, et se rapproche dans ce sens du concept deWorkflow.

- Modèle global *(global model)* : Ce modèle correspond à un modèle d'interactions de Web services pris deux à deux. Il décrit une collection de liens entre opérations duales de Web services, sans indiquer de structure de contrôle explicite. Il peut s'apparenter `a un contrat liant deux parties dans l'univers commercial. Ces liens (plug links)

permettent de définir une interface publique afin que les acteurs, non identifiés, puissent adhérer au contrat, dialoguer avec le Web service et incorporer les processus métier de l'entreprise.

4.2.3 XLANG

XLANG [5] est un langage XML proposé par Microsoft pour l'orchestration d'activités qui constituent un processus métier. Il est utilisé au sein de la plate-forme BizTalk. Tout comme WSFL, XLANG s'appuie sur WSDL en réutilisant un certain nombre des concepts définis dans cette norme. Il reprend, totalement la description d'un service en terme de groupe de ports et de liaisons à des protocoles de transport.

Un service XLANG est donc, un service WSDL qui spécifie un comportement *(Behavior)*.Un comportement est constitué d'un processus et d'un ensemble de corrélations. Il y a huit processus identifiés :

- *All* : indique une exécution parallèle des sous-processus.
- *Compensate* : invoque la compensation d'une transaction.
- *Context* : un contexte forme un cadre pour des déclarations locales, la gestion des transactions et la gestion des exceptions.
- *Empty* : un processus vide.
- *Pick* : attend l'arrivée d'évènements (Event) pour déclencher des processus (un processus par évènement). Cet évènement peut être la levée d'un signal ou la fin d'une action.
- *Sequence* : une séquence de processus et d'actions.
- *Switch* : définit des branches conditionnelles, chacune reliée à un processus.
- *While* : définit une boucle sur un processus.

Il y a quatre types d'action définis en XLANG :

- *Action* : invoque une opération d'un service Web.
- *DelayUnti*l et *DelayFo*r : suspendent l'exécution du processus.
- *Raise* : lève un signal.

4.2.4 WSCL

WSCL (Web Services Conversational Language) [8] est proposé pour d´écrire à l'aide de documents XML les Web services en se basant sur leurs conversations. Les définitions WSDL peuvent être manipulées par WSCL pour d´écrire les opérations possibles ainsi que leur chorégraphie. En retour, WSDL fournit les concrétisations vers des définitions de messages et les détails techniques pour les éléments manipulés par WSDL.

Les éléments d'une spécification WSCL sont les suivants :

1. Description des types de documents : elle référence les types de documents qui peuvent être échangés ou reçus. Les documents sont référencés par leurs URI respectifs.

2. Interactions : elles modélisent les actions des conversations sous la forme d'échanges de documents. Il existe 5 types d'interactions :

(a) *Send* : le service envoie un document.
(b) *Receive* : le service reçoit un document.
(c) *SendReceive* : le service envoie un document et en attend un au retour.
(d) *ReceiveSend* : le service reçoit un document et en renvoie un au retour.
(e) *Empty* : aucun document n'est échangé, le rôle de ce type d'interaction est de modéliser le début et la fin d'une conversation.

3. Transitions : elles spécifient la relation d'ordre entre les opérations. Pour cela, une transition possède une interaction source et une interaction de destination. Il est également possible (mais non requis) de spécifier un type de document pour la transition source, ainsi qu'une condition sur la transition.

4. Conversations : elles listent toutes les interactions et les transitions qui forment une conversation. Elles définissent également des propriétés supplémentaires tels que le nom de la conversation et les interactions de début et de fin.

4.2.5 WSCI

WSCI (Web Services Choregraphy Interfaces) [7] est un langage XML de description d'interface qui décrit le flux de messages échangés par un Web service qui interagit de manière ordonnée avec d'autres services (la chorégraphie de messages). Il permet ainsi, de décrire le comportement externe observable du service et d'exprimer les dépendances logiques et temporelles entre les messages échangés à l'aide de contrôles de séquences, corrélation, gestion de fautes et transactions. Ici encore on remarque que WSDL et ses définitions abstraites sont réutilisées afin de pouvoir également d'écrire par la suite les modalités de concrétisation des éléments manipulés pour modéliser un service.

WSCI définit les concepts suivants :

1. Les interfaces : elles décrivent les scénarios offerts aux clients. Il est ainsi possible d'offrir un scénario différent en fonction du client qui interroge le service. Une interface propose donc, le comportement externe observable du service considéré.

2. Les activités et leurs chorégraphies. Il existe deux types d'activités : atomiques ou complexes. Une activité complexe est en réalité une activité composite où le niveau le plus fin est l'activité atomique. Une activité complexe décrit la chorégraphie des activités qui la composent. Les chorégraphies supportées sont les exécutions séquentielles et parallèles, les boucles et branchements conditionnels.

3. Les processus : il s'agit de l'unité de réutilisation de WSCI. Un processus peut être instancié par la réception d'un ou plusieurs messages déclenchant (triggers). L'instanciation peut également se faire par appel ou dédoublement du processus. L'appel est synchrone alors que le dédoublement correspond au lancement du processus dans un nouveau fil d'exécution (thread).

4. Les propriétés : il s'agit là, d'un détour destiné à implémenter l'équivalent de variables dans les langages de programmation. Elles peuvent en particulier être employées pour référencer une définition de message WSDL afin d'éviter de le faire explicitement, ce qui peut parfois alourdir le document WSCI.

5. Les contextes : ils décrivent les environnements dans lesquels sont exécutés les activités. Un contexte contient :
   (a) les déclarations disponibles pour les activités.
   (b) les exceptions qui peuvent surgir à l'exécution des activités.
   (c) les propriétés transactionnelles des activités.

6. Les corrélations de messages : WSCI définit les conversations comme des échanges de messages. La corrélation de messages vise à associer à chaque message reçu la conversation qui lui est destinée. Pour cela, chaque message échangé fait référence à une instance de corrélation qui permet de le replacer dans sa conversation.

7. Les comportements exceptionnels : lorsqu'une exception survient, WSCI permet d'associer un ensemble d'actions à exécuter en réponse. Cette gestion des exceptions est systématiquement placée dans un point précis de la conversation.

8. Les comportements transactionnels : un contexte peut être associé à une transaction. Une transaction est soit atomique soit ouverte incluse, c'est-à-dire, composée d'autres transactions.

9. Le modèle global : il est décrit par la collection d'interfaces des services participants ainsi que par les liens avec les descriptions statiques effectuées `a l'aide de WSDL.

4.2.6 ebXML

ebXML (electronic business XML) [11] a été créé sous l'impulsion de deux organismes internationaux : OASIS (Organization for the Advancement of Structured Information Standards) et UN/CEFACT (United Nations Centre for Trade Facilitation and Electronic Business). Le but est de proposer une nouvelle infrastructure d'échanges électroniques globale, souple, économique et accessible aux utilisateurs du secteur public et du secteur privé. ebXML est basé sur l'utilisation des technologies avancées de l'Internet et plus particulièrement du langage XML. ebXML vise à la définition des collaborations entre des partenaires au cours des processus d'affaires à travers des transactions et d'échanges de documents. Les actions de plus bas niveau sont des échanges de documents entre partenaires.

Les processus ebXML sont vus comme des collaborations multipartites *(MultiPartyCollaboration)*. Une collaboration multipartite implique différents partenaires *(BusinessProcessPartenerRole*). Ceux-ci interagissent au travers de collaborations bipartites *(BinaryCollaboration*), dans lesquelles deux partenaires vont jouer un rôle parmi les rôles autorisés (*AuthorizedRole)*. Une collaboration multipartite est donc ramenée à un ensemble de collaborations bipartites. Une collaboration bipartite passe par un certain nombre d'états (State) ordonnancés par des transitions (Transition).

Ces transitions sont soit définies par la collaboration, soit par un partenaire. Cela permet à tout partenaire de personnaliser ses processus. Ainsi, dans une collaboration de type achat/vente entre un acheteur et un vendeur, les interactions entre les deux parties peuvent être similaires, alors que le processus côté vendeur peut être différent selon que celui-ci soit producteur ou simple revendeur de l'article.

Parmi les états, nous identifions le point de départ de la collaboration *(Start),* les points finaux *(CompletionState)*, les cas de succès *(Success)* ou d'échec *(Failure)*, les points de parallélisme

(Fork) et de synchronisation *(Join)*. Enfin ces états peuvent également être des activités d'affaires *(BusinessActivity)* qui définissent : une sous-collaboration *(CollaborationActivity)* ou une activité transactionnelle *(BusinessTransactionActivity)* entre deux rôles qui sont l'initiateur *(from)* et le répondeur *(to)*. L'activité transactionnelle est atomique. Elle décrit un échange de document entre deux rôles. Cet échange est réalisé au sein d'une transaction *(BusinessTransaction)* et décomposé en deux parties : la requête (Requesting*BusinessActivity*) et la réponse *(RespondingBusinessActivity)*. La requête est obligatoire, par contre la réponse est optionnelle (il peut arriver que la requête définisse seulement un transfert de documents sans attente de réponse).

Un flux de document est représenté comme une enveloppe *(DocumentEnvelope)* contenant un document principal *(BusinessDocument)*. En plus de celui-ci, un certain nombre de documents optionnels peuvent être attachés *(Attachment)*. Des niveaux de sécurité qui sont définis dans *DocumentSecurity* peuvent être affectés à l'enveloppe, ainsi qu'à chacun des documents attachés.

4.2.7 BPML/ BPEL4WS

BPML [10] définit deux types de définition de processus : *Abstract* et *Process*. Un *Abstract* est un processus partiel, et donc non exécutable. Un *Process* définit un processus exécutable. Il peut implémenter plusieurs *Abstracts.*

Une définition de processus peut définir des messages *(Message),* des participants *(Participant)* et des ensembles de règles *(RuleSet)*. Les messages définissent les informations qui transitent entre activités. Ils peuvent être de type *data, request* ou *response*. Les participants recouvrent l'ensemble des entités qui interagissent avec le processus (Web service, application, rôle organisationnel, etc.).

Enfin, un processus définit une activité de plus haut niveau. Cette activité est soit une activité simple *(SimpleActivit*y), soit une activité complexe *(complexActivity)*. Elle peut avoir des contraintes de temps concernant son démarrage (schedule) ou sa fin *(completeBy)*. Une activité complexe peut définir une transaction *(Transaction)*. Dans le cas où une transaction est abandonnée, des activités de compensation *(compensate)* peuvent être lancées pour chacune des sous-activités de l'activité complexe. Enfin les activités complexes peuvent générer des exceptions *(Exception)*. Une exception peut être adressée à un participant. Une activité peut être désignée pour traiter cette exception quand elle survient au cours de l'exécution d'une activité complexe *(OnException.target)*.

BPML définit trois types d'activités : complexes, simples et procédurales. Les activités complexes sont les suivantes :

- *All* : une activité qui est terminée lorsque toutes ses sous-activités sont terminées. Ces sous activités sont effectuées en parallèle.
- *Choice* : une activité qui est terminée dés-que l'une de ses sous-activités est terminée.
- *Foreach* : une activité qui répète une séquence d'activités sur un ensemble de valeur.
- *Sequence* : une activité qui est terminée lorsque toutes ses sous-activités sont terminées. Ces sous-activités sont effectuées en séquence.
- *Switch* : une activité qui est terminée lorsque toutes ses sous-activités candidates à l'exécution sont terminées. Si aucune activité n'est candidate, et si un lien 'otherwise' est défini, l'activité reliée par ce lien est exécutée.

Les activités simples identifiées sont les suivantes :

- *Empty* : ne fait rien.
- *ExceptionActivity* : génère une exception.
- *Consume :* attend et consomme un message d'un participant.
- *Opération* : invocation d'une opération sur un participant *(InvokingOperation)*, c'est-à-dire, envoi du message sortant au participant et attente du message entrant, ou implémentation d'une opération *(ImplementingOperation)*, c'est-à-dire, consommation du message entrant et production du message sortant.
- *Produce :* produit un message et le délivre à un participant.

Enfin, le dernier type d'activité concernant les activités procédurales et qui sont:

- *Assign* : exécute une affectation.
- *Complete* : termine le processus.
- *Join* : attend la fin de l'exécution d'un sous processus.
- *Release* : libère une donnée de la valeur qui lui a été affectée.
- *Repeat* : répète l'activité parente.
- *Spawn* : démarre un sous-processus.

Pour des raisons de marketing, le langage BPML été adapté et renommé BPEL (Business Process Execution Language) par les co-auteurs (IBM, Microsoft et Intalio) de la dernière spécification. Il correspond en réalité, à la fusion des langages WSFL d'IBM et XLANG de Microsoft. Le nom exact de la spécification BPEL est BPEL4WS (Business Process Execution Language for Web Services) [9]. Un processus BPEL ne se limite pas à l'orchestration de Web services : il est aussi possible d'inclure des procédures stockées SQL [62]. Pour cela BPEL repose sur les capacités d'extension de WSDL. BPEL4WS permet de définir des interactions intra et inter-entreprises et un de ces points forts est la possibilité d'avoir une représentation indépendante des interactions entre les partenaires,

L'élément atomique d'un processus BPEL4WS est une « activité », qui peut être l'envoi d'un message, la réception d'un message, l'appel d'une opération (envoi d'un message, attente d'une réponse), ou une transformation de données. BPEL offre les fonctionnalités suivantes :

- Organisation des activités : activités séquentielles, parallèles, switch, conditionnel, etc.
- Gestion des erreurs : il est possible de définir des gestionnaires d'erreurs pour des erreurs spécifiques – métier ou techniques. Des erreurs peuvent être déclenchées, traitées ou ignorées ;
- Gestion des transactions : il existe deux modes de transactions : les transactions courtes et les transactions longues.
- Organisation du processus en sous processus.

4.2.8. Synthèse

L'analyse de l'état de l'art sur le domaine du Workflow nous a permis en effet, de clarifier ces deux concepts : "Workflow" et "business process". Le terme "Workflow", de part son origine, n'est autre que l'ordonnancement des flots des travaux, impliquant une coordination des tâches entre différents acteurs. Contrairement à la notion de business process qui fait référence

directement à la notion métier de l'entreprise ainsi qu'aux règles métiers qui spécifient l'exécution du processus.

Cependant, avec l'explosion des nouvelles technologies de l'information et de la communication (NTIC), et surtout avec l'émergence de la technologie des Web services, le concept a pris une autre dimension dans les entreprises pour se rapprocher et être équivalent au concept de business process et s'intégrer dans un contexte inter-entreprises.

Beaucoup de formalismes de modélisation graphique de Workflow sont utilisés. Chacun a ses avantages et ses inconvénients. Quant au choix d'un formalisme bien adapté aux besoins, certains auteurs [69], [70] et [71] proposent un ensemble des critères, organisés en différentes catégories tels que :

- Représentation de la granularité : un processus peut se décomposer en sous processus.
- Description de l'enchaînement d'activités : un processus métier doit être décrit en termes d'enchaînement d'activités.
- Traitement de conditions : possibilité de définir les conditions d'exécution d'une activité en fonction de variables internes au processus (coûts, délais, etc.).
- Gestion des évènements : prise en compte des évènements externes et des exceptions dans le processus.
- Gestion documentaire : possibilité d'assigner des documents à une activité métier.
- Gestion de la performance : mettre en oeuvre un suivi de performance d'un processus. Il est important d'attacher à chaque activité des critères de mesure de performance (coûts, délais, qualité, etc.).
- Description des ressources : une ressource doit pouvoir être typée (système, homme, machine) et décrite (capacité, fiabilité, localisation…).
- Intégration des vues : une intégration d'autres vues d'entreprise (informationnelle, organisationnelle, décisionnelle, de ressources) doit être possible ou facilitée.
- Contrôle avancé de flux de processus : possibilité d'inclure des symboles de contrôle avancés telle qu'une horloge pour décrire une période de temps passée.
- Lien avec une définition exécutable : existence d'un lien direct avec une définition exécutable de processus : BPEL (Business Process Execution Language), etc.

Cependant, l'existence des différents langages de spécification de Workflow inter-organisationnel tels que XPDL, WSFL, BPML avec la variété de leurs concepts, ainsi que l'absence d'un standard déclaré, augmentent le fossé pour résoudre l'interopérabilité des modèles de Workflow.

Dans ce contexte d'interopérabilité, une étude comparative entre ces différents langages a été faite par certains auteurs [73] pour permettre de dégager un modèle général pour la spécification d'un langage de Workflow inter-organisationnel permettant la coopération inter-entreprises. Ce modèle doit satisfaire sept perspectives à savoir :

- *Fonctionnelle :* elle spécifie la fonctionnalité du Workflow, qui inclue une description de son but, ses entrées, ses sorties, ses contraintes éventuelles et sa décomposition en sous-Workflows ou en activités composées (ou atomiques). Cette perspective doit comprendre :

  - ✓ Le type de Workflow inter-organisationnel, c'est-à-dire, quel fragment de Workflow doit être exécuté par le (ou les) partenaire(s) collaboratif(s).

- ✓ Un mécanisme souple permettant d'exposer la partie publique du Workflow qui sera exposée aux partenaires pour la coopération et la partie privée qui doit être interne et cachée afin de préserver le savoir-faire des partenaires.
- ✓ La sémantique des activités au moyen des pré-condtions et des post-conditions.

- *Opérationnelle* : elle décrit comment les activités sont implémentées. Un langage de spécification doit permette de spécifier les implémentations d'activités, comme étant un pré-requis pour des spécifications exécutables. Ce qui simplifie considérablement le développement des systèmes Workflows et d'exclure l'implémentation de la spécification à partir de la partie publique, préservant ainsi, l'autonomie de conception des partenaires.

- *Comportementale* qui concerne :

  - ✓ les structures de contrôle basées sur les conditions de transition pour supporter les flux séquentiels, parallèles, etc.
  - ✓ les contraintes de temps d'exécution qui doivent être respectés sur la durée d'exécution des activités (critiques ou autres) et surtout concernant le temps global d'exécution du Workflow inter-organisationnel.
  - ✓ Les types d'exceptions qui devront surgir en cas d'échecs d'exécution d'activités ou de fragments de Workflow inter-organisationnel et comment intervenir pour les manipuler.

- *Informationnelle* : elle porte sur les types de données, la réutilisation de ces types de données et le flot des données qui sont manipulés dans la coopération, ainsi que des formats d'échange dans le cas d'hétérogénéité de ces données.

- *Interactionnelle* : elle correspond aux interdépendances entre des fragments de Workflows et qui demande des interactions entre les participants pour coordonner l'exécution de ces fragments. Ces interactions peuvent simples ou primitives, de type requête/réponse ou complète en les attachant à une représentation concrète des données vers le vocabulaire XML et le protocole de transport tel que HTTP. Aussi, cette perspective doit inclure l'indépendance d'implémentation de ces interactions pour augmenter la réutilisabilité.

- *Organisationnelle* : elle fait référence aux rôles, aux profils, ainsi qu'aux accords (agreements) définis par les partenaires collaboratifs pour le Workflow inter-organisationnel. Cette perspective peut comprendre aussi, la participation dynamique d'une quelconque organisation pour l'exécution du Workflow. Ce qui implique une décision flexible pour le choix d'une organisation pour se joindre au partenariat et cela au cours même de l'exécution du Workflow.

- *Transactionnelle* : elle décrit les parties du Workflow inter-organisationnel qui présentent des propriétés transactionnelles. Ces transactions distribuées ont évoquées par le terme de '' transactions intéropérables'' dans [74]. Elles différent de celles présentées dans un contexte intra-organisationnel qui utilisent les propriétés ACID (Atomicité, Cohérence, Isolation, Durabilité).

Ces différentes perspectives sont présentées dans le tableau suivant (table 1.1). La notation utilisée dans ce tableau est +, × et ~, qui signifie que le critère correspondant à la perspective est respectivement, totalement satisfait, partiellement satisfait, ou n'est pas satisfait du tout.

Perspectives		Langages de spécification de Workflow					
		WSFL	XLANG	BPML	ebXML	WSCL	XPDL
Fonctionnelle	Type Interorg.Workflow	+	+	×	+	+	~
	Partie Publique/Privée	~	~	+	~	~	~
	Sémantique - Activité	~	~	~	+	~	~
Opérationnelle	Implémentation -Activité	+	+	+	×	×	+
Comportementale	Structure de contrôle	+	~	+	+	~	+
	Contraintes de temps	×	~	+	+	×	+
	Manipulation -Exceptions	~	+	+	+	×	×
Informationnelle	Types de données	+	+	+	+	+	+
	Réutilisation - Types de données	~	~	~	+	~	~
	Flots de données	+	~	+	~	~	~
Interaction	Interactions Simples	+	+	+	+	+	×
	Implémentation d'interactions	+	+	×	+	×	×
	Indépendances d'implémentation	+	×	+	+	+	+
Organisationnelle	Rôles	+	×	+	+	+	+
	Profils	~	~	×	+	×	×
	Accords (Agreements)	×	~	×	+	×	×
	Participation dynamique	+	×	+	×	×	×
Transactionnelle	Transactions Intra-org	×	+	+	×	×	×
	Transactions Inter-org	×	×	~	+	×	×

Table 1.1 : Etude comparative des langages de spécification Workflow [73].

5. Conclusion

Ce chapitre nous a permis de mieux comprendre notre domaine d'application, et en particulier de clarifier le concept de Workflow. Cependant, l'étude comparative des langages de spécification de Workflow, nous a permis de mettre en évidence l'hétérogénéité syntaxique des modèles de Workflow. Cette hétérogénéité ne représente en fait, qu'un niveau parmi d'autres niveaux d'hétérogénéité qui apparaissent lors de la mise en œuvre de l'interopérabilité.

Par conséquent, pour mettre en œuvre des mécanismes d'interopérabilité, il est fondamental de prendre en considération d'autres aspects liés à plusieurs niveaux tels que le niveau métier (qui concerne l'aspect organisationnel, les objectifs métiers), le niveau technique (qui concerne les moyens technologiques mis en œuvre) et le niveau sémantique (qui concerne la signification des informations).

Notre travail de thèse se concentre principalement sur le dernier niveau où l'aspect sémantique constitue un défi majeur pour les entreprises qui désirent coopérer. Pour le résoudre, nous allons d'abord, explorer les différentes approches existantes qui sont proposées, ainsi que sur les différentes techniques qui sont utilisées pour la mise en oeuvre de l'interopérabilité. Ensuite, nous les étudions et les analysons par rapport à la problématique posée et aux objectifs fixés tels que la flexibilité, le respect du savoir-faire des entreprises, l'autonomie des Workflows existants (ou systèmes de gestion de Workflow) et l'ouverture. Ce qui permet de nous guider vers une méthodologie de recherche d'une solution conceptuelle pour résoudre l'interopérabilité des modèles de Workflow : C'est l'objet des chapitres suivants.

Chapitre II

Interopérabilité - Formes - Approches et Techniques mises en oeuvre

1. Introduction

Les Workflows que nous appelons processus métiers, ont pris une place prépondérante dans les entreprises durant ces dernières années grâce à leurs bénéfices remarquables. Face à la concurrence et dans le but d'améliorer leur productivité, les entreprises expriment un grand besoin d'ouverture et de coopération à l'échelle mondiale. Elles ont besoin de s'allier à d'autres entreprises de compétences complémentaires afin de coopérer avec, et de réaliser des projets qui ne sont pas à la portée d'une seule entreprise, par exemple, externalisations intensives de processus et de services métiers.

Cependant, pour coopérer, les entreprises doivent adopter de nouveaux schémas de comportement, modifier éventuellement leur organisation et à s'ouvrir davantage à leur environnement et s'organiser en réseaux, au sein desquels interagissent différents acteurs (fournisseurs, sous-traitants, etc.) afin de pouvoir répondre rapidement aux exigences et aux évolutions du marché.

Pour faire face à ces nouvelles exigences, les entreprises ont souvent recours au concept d'intégration et parfois, de façon plus spécifique, au concept d'interopérabilité.

Dans le domaine du Workflow, l'interopérabilité est désormais incontournable pour les entreprises si elles souhaitent s'inscrire dans une dynamique d'intégration et assurer la pérennité économique. Dans ce qui suit, nous allons définir le concept d'intégration et celui de l'interopérabilité en mettant en évidence la différence qui existe entre ces deux concepts.

2. Concepts d'intégration et d'interopérabilité

Les problèmes d'intégration et d'interopérabilité ne sont pas nouveaux et ils constituent des domaines de recherche auxquels s'intéressent plusieurs communautés dont celle des systèmes d'information, celle des bases de données, celle du génie logiciel, celle des systèmes Workflows, etc. Plusieurs approches, standards et/ou technologies ont été proposés pour résoudre ces problèmes.

2.1 Concept d'intégration

Plusieurs définitions ont été proposées dans la littérature. Selon le dictionnaire, l'intégration désigne l'action d'intégrer qui signifie "rendre entier", faire entrer dans un ensemble. Pour [75], l'intégration d'entreprise définit comme étant le processus qui "consiste à faire interopérer fortement les personnes, les machines et les applications afin d'accroître la synergie au sein de l'entreprise". Selon [76] l'intégration de façon générale, consiste à combiner des composants de telle façon à former un nouvel ensemble constituant un tout pour créer de la synergie.

Pour certains auteurs [77], l'intégration d'entreprise est définie comme étant "la coordination de tous les éléments incluant les processus métiers, les ressources humaines et la technologie d'une entreprise, qui fonctionnent ensemble dans le but d'atteindre la réalisation optimale de la mission de cette entreprise telle que définie dans la stratégie de l'entreprise".

Par contre [13] définit le concept d'intégration comme étant "la capacité de communication (par partage de données, par envoi de messages ou d'évènements) entre des outils ou des

applications éventuellement hétérogènes et distribuées. Lorsque celle-ci est augmentée de la capacité de coopération, on parlera, dans ce cas, d'une interopérabilité.

2-2 Concept d'interopérabilité

Plusieurs définitions ont été proposées dans la littérature pour définir le terme "interopérabilité". Nous n'en citer que quelques unes :

- L'interopérabilité est la capacité des systèmes informatiques et des processus de supporter l'échange de données et de permettre le partage d'information et de connaissances [78] ;
- L'interopérabilité est la capacité que possèdent deux ou plusieurs systèmes ou composants à échanger des informations puis à exploiter les informations venant d'être échangées [79] ;
- L'interopérabilité est la capacité de communiquer avec d'autres systèmes et d'accéder à leurs fonctionnalités [75] ;
- Pour [80], l'interopérabilité est définie comme étant : "Interoperability is the ability of a collection of communicating entities to (a) share specified information and (b) operate on that information according to an agreed operational semantics" ;
- Dans le domaine du Workflow, la WfMC a défini l'interopérabilité des modèles de Workflow comme étant la possibilité qu'ont plusieurs moteurs Workflow de communiquer pour exécuter de manière coordonnée des instances de procédures sur l'ensemble de ces différents moteurs [36].

Dans notre travail, le concept d'interopérabilité des modèles de Workflow définit la notion d'échange de modèles et de coopération entre des acteurs collaboratifs malgré les différences dans les langages de spécification de Workflow et les environnements d'exécution. Par définition, la coopération signifie qu'un ensemble de partenaires ont la volonté de réunir leurs compétences et travailler ensemble pour atteindre un objectif commun dans un contexte organisationnel.

3. Problématique de l'interopérabilité des modèles de processus

Les problèmes d'interopérabilité entre des modèles de processus apparaissent lorsqu'on cherche à faire intéropérer un ensemble de modèles qui sont développés indépendamment les uns des autres et qu'il est difficile de les faire fonctionner ensemble à cause des différents niveaux d'hétérogénéité qui interviennent et rendent complexe l'interopérabilité en général.

Concernant l'hétérogénéité de ces modèles, plusieurs niveaux d'hétérogénéité ont été détectés [3], [81**,** [17], et sont regroupés principalement en quatre niveaux :

- ***L'hétérogénéité syntaxique :*** Elle concerne la diversité des formalismes et des langages de spécification de Workflow utilisées qui existent pour décrire l'aspect du Workflow. Le choix d'un formalisme par rapport à un autre peut avoir un impact considérable sur la compréhension du modèle et de la capture du domaine. Ce niveau concerne le format

des données (encodage des données), le niveau structurel (différentes représentations (types) des données au niveau schéma), et le niveau modèle/ représentation (différents modèles sous-jacents tels que relationnel et objet).

- ***L'hétérogénéité sémantique*** **:** Elle est dûe à la diversité des concepts et des vocabulaires utilisés par les modèles de processus. Chaque modèle offre des concepts différents. Ce niveau concerne la signification et la compréhension des éléments métiers telles que les données, les opérations, les fonctions ou les activités collaboratives.

- ***L'hétérogénéité technique*** **:** Elle concerne les plates formes d'exécution des modèles de processus qui sont autonomes et différentes, la variété des systèmes d'exploitation, des protocoles de communication, etc.

- ***L'hétérogénéité organisationnelle/métier*** : Elle définit les responsabilités, autorisations, confiances, aspects légaux, propriétés intellectuelles et structures organisationnelles nécessaires à l'acceptation des échanges d'information entre applications par les différents acteurs. Ce niveau d'hétérogénéité est particulièrement mis en avant dans le cadre de l'interopérabilité de l'administration électronique et des gouvernements [82].

Cependant, le niveau d'hétérogénéité organisationnel est parfois confondu avec le niveau métier. Par exemple, le niveau organisationnel défini par le cadre EIF (*European Interoperability Framework*) [78] et le niveau métier (*business*) défini par le cadre IDEAS *(Interoperability Development for Enterprise Applications and Software)* [81] présentent une forte similitude et concernent tous les deux : le contexte métier des entreprises en termes d'objectifs, de stratégies, de modélisation des interactions entre partenaires, etc.

Cependant, ces quatre niveaux sont généralement croisés en trois niveaux : syntaxique, technique et sémantique dont le dernier constitue un problème crucial pour les entreprises désirant interopérer et auquel nous nous intéresserons particulièrement dans notre travail.

4. Formes d'interopérabilité existantes

Plusieurs formes d'interopérabilité existent dans la littérature pour l'interconnexion de Workflows inter-entreprises. Pour chaque forme, nous citons ses caractéristiques, puis nous l'analysons par rapport aux objectifs fixés dans la thèse.

4.1 Partage de capacités

Dans cette approche [83], un ensemble de partenaires se mettent d'accord, dès la phase de conception, sur un Workflow global mettant en service leurs capacités et savoir-faire mutuels. Le contrôle quant à lui est centralisé et le routage de Workflows est géré par un seul système de gestion de Workflow alors que l'exécution des activités est distribuée sur les différents participants. Le Workflow est géré par un système de gestion de Workflow central et fait appel aux ressources des partenaires pour exécuter les activités qui leur sont associées. Le partage de capacités est simple et utilise une infrastructure commune ainsi qu'une gestion centralisée d'un Workflow unifié.

4.2 Exécution chaînée

L'exécution chaînée [83], [84] consiste à modéliser un Workflow global en plusieurs sous Workflows disjoints exécutés séquentiellement par les différents partenaires. En effet, chaque partenaire se charge d'une partie du Workflow. Une fois cette partie exécutée, le partenaire transfère le flot au partenaire suivant. Cette forme d'interopérabilité ne requiert pas d'exécution parallèle et se base sur un contrôle distribué de Workflow.

4.3 Sous-traitance

La sous-traitante [83] est une forme d'interopérabilité permettant à un partenaire principal de déléguer l'exécution et la coordination d'une partie de son Workflow à d'autres partenaires. Le contrôle des Workflows étant hiérarchique, le partenaire principal voit les sous Workflows sous traités comme atomiques alors que ceux-ci peuvent avoir des structures complexes au niveau des partenaires les exécutant. La sous-traitance se base sur un seul scénario de coopération hiérarchique : un partenaire principal cède une partie de son Workflow au sous-traitant. Celui-ci exécute les activités et rend le contrôle au niveau supérieur après avoir fini l'exécution.

4.4 Transfert de cas

Dans cette forme d'interopérabilité [85], les différents partenaires impliqués dans l'interopérabilité partagent une description commune de la définition de Workflow. La spécification de Workflow est répliquée au niveau de chaque partenaire. Chaque participant possède alors une copie du Workflow complet. Seules les instances de ce schéma (Workflow cases) sont transférées entre les participants afin d'équilibrer le partage de charges ou parce qu'une activité doit être exécutée par un participant précis.

4.5 Transfert de cas étendu

Le transfert de cas étendu [85] est conçu pour pallier au problème précédent et permet aux partenaires impliqués dans le Workflow inter-entreprises de personnaliser leurs Workflows et entrer quelques modifications locales dans leurs structures internes (ajout et/ou la suppression des activités, par exemple), selon leurs besoins avec la contrainte que le comportement des Workflows personnalisés soit identique à celui du Workflow global.

4.6 Couplage souple

Cette forme d'interopérabilité dite aussi "faiblement couplé (loosely coupled)" [85], [84], [86], est basée sur la décomposition d'un Workflow global en sous-Workflows disjoints. Chaque partenaire dispose de son propre Workflow local privé et doit être augmenté d'une structure d'interaction afin de permettre la communication entre les différents partenaires et l'exécution correcte des instances. L'interaction est réalisée grâce à une communication asynchrone et se base sur la circulation de messages entre les Workflows locaux des partenaires.

Dans cette forme d'interopérabilité, les partenaires commencent d'abord, par se fixer un protocole d'interaction grâce à un diagramme de séquence [87], [88] [89] spécifiant les différents messages circulant entre les partenaires et ordonnant les événements qui leurs sont associés. Ensuite, chaque partenaire conçoit un Workflow local privé conforme avec le protocole. Enfin, le Workflow faiblement couplé est exécuté par les différents participants. Ce

dernier est constitué des parties d'un schéma d'un Workflow global qui sont distribuées entre les participants pour exécution. Pour coopérer et échanger des messages, les partenaires disposent de gestionnaires de messages permettant d'interpréter les messages reçus et de les acheminer vers les activités appropriées au sein des systèmes de gestion de Workflow. L'ordre de ces messages est spécifié dans la charte de séquence de messages sur laquelle les participants se sont mis d'accord.

4.7 Public-à-Privé

Dans cette forme [90], [91], les entreprises impliquées dans la coopération se mettent d'accord sur un Workflow public commun servant de contrat entre elles. Cette forme est basée sur la notion de l'héritage. Chaque partenaire crée son propre Workflow à partir de ce Workflow public. Le Workflow privé créé doit être une sous-classe d'une partie du Workflow public

L'approche Public-à-Privé est basée sur trois phases que nous résumons comme suit :

1. ***Conception du Workflow public*** *:* Cette phase consiste à concevoir un Workflow public commun qui peut être vu comme un contrat entre les partenaires. Ce Workflow ne montre nécessairement pas la façon dont les activités sont exécutées mais montre seulement les activités intéressant toutes les partenaires.

2. ***Partitionnement du Workflow public*** **:** cette phase consiste à partitionner le Workflow public selon les partenaires et à relier les parties publiques entre elles pour former un Workflow inter-entreprises. Celui-ci est formé d'un ensemble de Workflows, de canaux pour acheminer les informations, de méthodes représentant la fonction invoquée lors de l'exécution de l'activité, ainsi que des relations de flots de canaux spécifiant la relation entre les participants.

3. ***Conception de Workflows privés*** **:** Après avoir partitionné le Workflow public, chaque partenaire peut réaliser la partie publique du Workflow inter-entreprises correspondant comme il le veut, à condition que le Workflow privé constitue une sous classe de sa partie publique. La relation de sous classe est basée sur la notion d'héritage**.**

4.8 Discussions

Après avoir présenté les différentes formes d'interopérabilité, nous allons les analyser vis-à-vis des objectifs fixés tels que respect du savoir-faire, flexibilité et autonomie.

- **Le partage de capacités** : Cette forme d'interopérabilité se base sur un seul scénario de coopération que fixent les différents partenaires dès la phase de conception ce qui rend le système rigide. Le partage de capacité suppose l'utilisation d'une infrastructure commune. Ceci représente une contrainte rigide pour la coopération de Workflows puisque les partenaires sont obligés d'adopter cette infrastructure commune. Elle ne préserve pas le savoir-faire des partenaires puisque le Workflow global est connu et partagé et par les différents participants.

- **L'exécution chaînée** : Elle définie par un seul scénario de coopération, spécifié dès le début, exigeant une exécution séquentielle du Workflow par les différents partenaires. Contrairement aux systèmes utilisant le partage de capacités, l'exécution et le contrôle

sont ainsi distribués. De même, elle ne préserve pas le savoir-faire des partenaires puisque le Workflow global est connu et partagé par les différents participants.

- **La sous-traitance** : Bien que cette forme d'interopérabilité préserve le savoir-faire des partenaires en sous-traitant les activités du Workflow principal, elle ne supporte pas des coopérations plus complexes avec différents types d'interactions entre les partenaires.

- **Le transfert de cas** : Il définit un seul scénario de coopération accepté par les différents partenaires dès la phase de conception du Workflow global. Il ne préserve pas le savoir-faire des partenaires puisque la même description du Workflow global est répliquée et accessible à tous les participants. En effet, les partenaires doivent se conformer au Workflow global. Enfin, le transfert de cas n'intègre pas les systèmes de gestion de Workflows des partenaires. Ces derniers doivent, en effet, être augmentés de la capacité de réception/envoi des instances ainsi que des données d'états.

- **Le transfert de cas étendu** : Cette forme d'interopérabilité ne préserve pas la totalité du savoir-faire des entreprises puisque les participants partagent le Workflow global.

- **Le couplage souple** : Cette forme d'interopérabilité ne préserve pas les Workflows pré-établis. En effet, la conception du Workflow local d'un partenaire dépend fortement du protocole de communication, et tout changement de partenariat doit être conforme au protocole de communication pré-défini. De plus, les partenaires impliqués sont amenés à étendre leurs systèmes de gestion de Workflow avec des mécanismes permettant d'interpréter les messages reçus et d'exécuter les opérations correspondantes

- La forme dite **"Public-à-privé" :** Cette forme ne préserve pas les Workflows pré-établis des partenaires puisque les partenaires doivent déterminer l'ensemble des règles à appliquer, ainsi que leur nombre et ordre, afin de correspondre leurs Workflows à la partie publique résultant du partitionnement du Workflow global.

Après avoir analysé les différentes formes d'interopérabilité, nous allons aborder dans la section suivante, les différentes approches qui sont proposées pour résoudre l'interopérabilité de Workflows inter-entreprises et nous allons les décrire et les analyser vis-à-vis de notre problématique d'interopérabilité conceptuelle et des objectifs fixés.

5. Approches d'interopérabilité de Workflows existantes

Plusieurs approches existent pour l'interopérabilité de Workflows inter-entreprises. Certaines sont proposées par des organisations internationales telles que la WfMC, l'OMG et le BPMI, et d'autres sont préconisées par des auteurs dans le domaine de la coopération de Workflows inter-organisationnels.

5.1 La WfMC et l'interopérabilité des Workflows

Pour réaliser l'interopérabilité des Workflows entre des entreprises, la WfMC a établi d'abord un glossaire [36] contenant tout le vocabulaire de base et la terminologie commune utilisée dans ce domaine, puis a défini un modèle de référence [56] qui représente un cadre de référence dans l'univers Workflow. Ce modèle contient les spécifications de l'interopérabilité entre des moteurs de Workflow hétérogènes (*interface 4*) permettant à des systèmes de Workflow

hétérogènes de s'échanger des modèles de processus et par la suite de répartir l'exécution de ces modèles sur différents systèmes.

Pour la WfMC, l'interopérabilité des Workflows est assurée par l'interface 4 appelée "WAPI4" (Workflow Application Programming Interface/Interchange 4) qui est une spécification abstraite définissant les fonctionnalités nécessaires pour supporter l'interopérabilité entre les différents moteurs de Workflows [57]. De même, elle a défini un protocole de communication Wf-XML [58], basé sur le langage XML qui sera utilisé comme base pour les implémentations des fonctionnalités de l'interface 4 de la WfMC, et un modèle canonique XPDL (XML-Process Definition Language) [4] (interface 1).

Pour la WfMC, plusieurs scénarios d'intéropérabilité existent, définissant différentes modalités d'exécution partagée d'instances de processus [45] :

- Intéropérabilité hiérarchique : sous-processus à l'intérieur d'un processus général.
- Intéropérabilité chaînée : enchaînement de processus distincts mais complémentaires.
- Intéropérabilité point à point : routage d'une activité à l'autre, chacune pouvant être exécutée par un moteur de Workflow différent.
- Intéropérabilité paralléle : synchronisation de Workflows exécutés en parallèle sur des moteurs de Workflow distincts.

5.2 L'OMG et l'interopérabilité des Workflows

L'OMG présente un Workflow comme un objet CORBA et considère l'interopérabilité des Workflows comme une intégration d'objets CORBA. Elle traite la spécification du Workflow comme un utilitaire CORBA (Common Object Request Broker Architecture) de gestion des activités [92], [93], [64]. La spécification OMG de l'utilitaire de Workflow définit les outils CORBA nécessaires au fonctionnement des gestionnaires de Workflows en se basant directement sur l'interface OMG-IDL (Interface Definition Language) [94] et par les interfaces spécifiées par la WfMC qui sont l'interface 2 (interface client Workflow) [59] et l'interface 4 (interface d'interopérabilité) [57] (figure 1.3).

Plus précisément, pour contrôler l'exécution, le suivi des Workflows, et permettre l'interopérabilité des Workflows, l'OMG a défini une spécification (Workflow Management Facility) [64] et a proposé une interface standard pour les moteurs Workflows et un protocole de communication CORBA. Elle est basée sur un modèle d'objets de Workflow, qui inclut un ensemble de classes standards pour la manipulation des processus lors de l'exécution. Le standard CORBA établit la façon dont les objets distribués peuvent participer à un Workflow au moyen d'invocations par un ORB (Object Request Broker) [66]. Toute implémentation de cette spécification [66] est guidée par des choix architecturaux ou liés à l'infrastructure, mais aussi, fortement par le formalisme de définition des processus.

5.3 Le BPMI et l'interopérabilité des Workflows

Pour le BPMI, l'interopérabilité des Workflows (appelés processus d'affaires) est assurée par la technologie des Web services qui fournit un moyen efficace pour l'intégration des processus métiers dans un contexte d'échanges B2B (Business to Business). L'intégration se fait au niveau de la couche métier comportant différents langages de composition de Web services. Chaque

Web service représente un service métier qui est un fragment d'un processus d'affaire et l'interopérabilité de processus est favorisée via une interopérabilité des Web services. Parmi ces langages de composition de Web services, le BPMI a adopté le standard BPEL4WS.

5.4 ebXML

ebXML (Electronic Business using eXtensible Markup Language) [98] a été conçu pour servir dans un contexte de marché global où des entreprises de différentes tailles pourraient effectuer des transactions électroniques via l'échange de messages basés sur XML. L'objectif d'ebXML est donc, de fournir un cadre sémantique assurant l'interopérabilité commerciale inter-entreprises en offrant une infrastructure qui assure l'interopérabilité des échanges de données inter-entreprises et de construire des partenariats pour la conduite de projets communs.

L'intégration des processus métiers à l'aide d'ebXML doit se faire en trois phases :

1. Phase d'implantation des spécifications ebXML : le référentiel ebXML contient des spécifications des processus métiers d'entreprises et des scénarii qui sont souvent utilisés dans les transactions commerciales. Ce référentiel contient les profils de collaboration des entreprises qui se sont déjà inscrites comme utilisant les transactions ebXML avec leurs partenaires.

2. Phase de recherche et de négociation des informations ebXML des partenaires : pour qu'une entreprise (A) puisse coopérer avec une entreprise (B), l'entreprise (A) récupère le profil de l'entreprise (B) du référentiel ebXML, puis étudie les capacités de (B) (i.e les processus métiers, les messages à échanger, etc.). Les entreprises négocient les termes de contrats avant de commencer la coopération. Puis, l'entreprise (A) envoie à l'entreprise (B) un contrat de partenariat ebXML appelé CPA (Collaborative Partner Agreement). Finalement, (A) et (B) se mettent d'accord sur un contrat de partenariat commun. Durant cette phase, des réunions sont organisées entre des décideurs des entreprises (A) et (B) afin de valider le contrat industriel de collaboration.

3. Phase d'exécution des transactions ebXML : une fois que le contrat de partenariat a été négocié, les transactions peuvent commencer selon une manière pré-définie où chaque entreprise joue un rôle prédéterminé. Les transactions consistent en l'envoi de messages ebXML suivant le service d'envoi de message ebXML.

Bien qu'ebXML permette d'intégrer des processus métiers dans un contexte B2B (Busines to Business), il présente une insuffisance dans la préservation des systèmes de gestion de Workflows existants des partenaires puisque tous les participants doivent se conformer à une infrastructure commune d'interopérabilité commerciale définie pour adapter et ajuster leurs interfaces. Ce qui réduit la flexibilité.

5.5 CrossWork

Le projet CrossWork [95], [96] a été développé dans le cadre du programme IST (Information Society Technologies) et du 4éme PCRD (Programme Cadre de Recherche et Développement) de la commission européenne. CrossWork se focalise sur le développement de Workflows inter-entreprises distribués dans le cadre de l'industrie automobile afin de faciliter l'échange d'informations entre les différentes entreprises participantes et d'adapter leurs modèles. Il se base sur la technologie des agents afin de déterminer les partenaires et de former

le Workflow coordonnant leurs activités respectives. L'approche utilise la plateforme de modélisation XRL (eXchangeable Routing Language) [97] basée sur les réseaux de pétri. A chaque fois que le système de gestion de Workflow central rencontre une activité, il invoque l'agent du partenaire correspondant pour l'exécuter.

L'approche CrossWork nécessite une génération d'interfaces pour les participants à chaque exécution. Ceci la rend peu flexible et ne préserve pas les systèmes de gestion de Workflow existants. En outre, cette approche se base sur la notion de boîte noire pour présenter les Workflows des entreprises participantes, ce qui n'est pas adéquat parfois dans un contexte de coopération où on a besoin d'un certain degré d'inter-visibilité des Workflows des partenaires.

5.6 WISE

Le projet WISE [102], [103], [24] (Workflow based Internet SErvices) propose une plate-forme pour la décomposition des processus d'entreprises virtuelles à travers des interfaces de processus de plusieurs entreprises. L'architecture de WISE est formée de quatre composants : définition, exécution/contrôle, surveillance/analyse, et coordination/communication de Workflows.

Le composant de définition de Workflows permet de définir les Workflows en utilisant comme bloc de construction les entrées d'un catalogue où les entreprises peuvent publier leurs services. Le composant d'exécution de Workflows permet de compiler la description du Workflow en une représentation appropriée à l'exécution et de contrôler l'exécution du Workflow en invoquant les services correspondants. Le composant de surveillance et d'analyse est un outil permettant de garder la trace de l'évolution de l'exécution du Workflow ainsi que des états de tous les composants actifs dans le système. Enfin, le composant de coordination et de communication permet de véhiculer des informations appropriées entre tous les participants en utilisant les informations produites par les Workflows comme la principale source de routage. La plate-forme de WISE ne préserve pas l'autonomie des partenaires et par suite leurs systèmes de gestion de Workflow existants puisque tous les participants doivent se conformer à des interfaces bien définies. Ce qui limite un peu la flexibilité.

5.7 Point de Synchronisation

Le modèle de synchronisation [104] proposé par les auteurs est un modèle ainsi qu'une architecture pour supporter la collaboration, la coopération et la coordination de plusieurs entreprises. Ce modèle se situe entre les systèmes de gestion de Workflow et les systèmes de groupware afin de profiter des avantages mutuels. Les points de synchronisation permettent de spécifier les relations entre les partenaires pour une activité donnée. Pour préserver l'autonomie des partenaires, le point de synchronisation se base seulement sur les résultats et les événements de notification des participants. La phase de synchronisation permet de réconcilier les différences entre les versions des objets manipulés. Cette phase n'est pas complètement automatique. En effet, le système permet de capturer les différences entre les versions mais la décision finale demeure humaine.

Ce modèle permet ainsi à différents partenaires de travailler ensemble en se basant sur des règles de coopération correspondant à la spécification du contrat les réunissant. Il permet de coordonner la progression du travail et de fournir aux partenaires, les informations nécessaires sur l'évolution du travail. Ce modèle permet d'ajuster la définition du processus de collaboration en mettant à jour le point de synchronisation durant la progression du travail

grâce à l'inclusion des fonctions de gestion de flots de données et de contrôle. Ce qui permet une flexibilité des processus.

L'inclusion des fonctions de gestion de flots de données et de contrôle permettent une définition flexible de procédés puisqu'il est possible d'ajuster la définition du processus de collaboration en mettant à jour le point de synchronisation durant la progression du travail. Grâce à un certain degré d'abstraction, le modèle permet l'intégration des processus d'entreprises existants, et la composition de plusieurs processus (considérés comme des composants. Le modèle de synchronisation est considéré comme portable et supporte l'interopérabilité entre les entreprises coopérantes tout en préservant leurs autonomies puisqu'il n'impose pas la compatibilité des autres modèles avec les lesquels il veut s'intégrer.

5.8 CrossFlow

Le projet CrossFlow [105], [106], [107], [108] a pour objectif de fournir un support à haut niveau pour des Workflows inter-organisationnels dans des entreprises virtuelles formées dynamiquement lorsque certaines activités sont sous-traitées à des fournisseurs externes. Ce support logistique aux entreprises est réalisé par la proposition de services abstraits, servant à trouver des partenaires potentiels et à réaliser une coopération étroite. Les services de coopération sont gérés par "des gestionnaires de contrats" (contract managers).

CrossFlow utilise une approche basée sur des contrats afin de décrire les relations entre les différentes entreprises et permet ainsi la définition et l'exécution des processus inter-entreprises. L'externalisation de service est marquée par trois phases : établissement du contrat, installation de l'infrastructure et exécution du service.

La phase d'établissement de contrat permet de déterminer les services à externaliser au niveau processus. Tout d'abord, les fournisseurs de services publient des modèles contenant les détails sur les services au sein d'un moteur de correspondance de services. Le client (consommateur de service) désirant externaliser un service contacte un moteur de correspondance de service en lui fournissant les modèles de leurs services afin de récupérer la liste des modèles de services des fournisseurs qui correspondent à sa requête. A la réception de cette liste, le client choisit un fournisseur avec lequel il établit un contrat.

La seconde phase consiste à construire une infrastructure pour l'exécution de services en se basant sur les détails extraits du contrat précédemment établi. Cette phase consiste en effet à sélectionner les modules adéquats et de les paramétrer afin d'obtenir le comportement recherché. La dernière phase consiste alors à exécuter le service externalisé. Une fois l'exécution du service terminée, l'infrastructure dynamiquement construite est abandonnée.

L'approche CrossFlow ne préserve pas l'autonomie. En effet, l'exécution du Workflow inter-entreprises est assurée grâce à des passerelles spécialisées configurées à priori et le mécanisme de communication entre ces passerelles est basée sur des interfaces CORBA-IIOP [109] bien définies.

5.9 CoopFlow

CoopFlow [20], [110] est une approche et une plate-forme pour la coopération de Workflows inter-entreprises. Elle s'inspire de l'architecture orientée services qui est basée sur trois opérations : la publication, la recherche et la connexion. Les fournisseurs de services publient

les différents services qu'ils offrent. Les demandeurs de services cherchent les différents services répondant à leurs besoins. Une fois que les services sont trouvés, ces demandeurs s'y connectent afin d'exécuter un ensemble d'opérations offertes par les services. De même que cette approche est composée de trois étapes : (1) publication de parties de Workflows d'entreprises pouvant être exploitées par d'autres entreprises, (2) interconnexion de Workflows et (3) coopération et surveillance de Workflows conformément à un ensemble de politiques de coopérations (contraintes d'interactions).

Cette approche est basée sur le modèle SOA et considérée comme flexible. Elle préserve l'autonomie et le savoir-faire des entreprises, elle n'est pas conceptuelle dans le sens où elle utilise directement la technologie des Web services associée à l'architecture SOA.

5.10 Autres approches

L'interopérabilité des modèles de Workflow a été également adressée par un ensemble de projets dans un contexte d'entreprises virtuelles. Dans ce qui suit, nous présentons brièvement quelques projets :

- Le projet NIIP [111] est développé pour supporter les entreprises virtuelles et offrir les technologies nécessaires à la collaboration d'un ensemble de partenaires hétérogènes. Il est basé sur des technologies ouvertes et standards telles que Internet et CORBA. La collaboration des entreprises consiste à partager la totalité de leurs services et ressources y compris les ressources humaines. NIIP ne préserve pas les systèmes de gestion de Workflow. En effet, l'intégration d'un nouveau système de gestion de Workflow exige la conformité à des interfaces statiques pré-définies à la conception que les partenaires doivent déterminer avant même de coopérer.

- Le projet X-CITTIC [112] s'intéresse aux entreprises virtuelles dans le cadre de l'industrie des semi-conducteurs où la fabrication des pièces est effectuée par un réseau de partenaires distribués à la demande. Ce projet est destiné aux entreprises virtuelles statiques partageant un processus métier où la constitution de ces entreprises, ainsi que leur configuration sont effectuées d'une manière manuelle, centralisée et statique. L'exécution du processus partagé est réalisée grâce à des passerelles spécifiques et elle est basée sur le principe des événements dont la sémantique est fortement couplée au processus lui-même. La conception de l'entreprise virtuelle est centralisée. X-CITTIC ne préserve pas l'autonomie des partenaires. En effet, la gestion du processus partagé est basée sur l'envoi d'événements entre les passerelles des différents partenaires dont la sémantique est déterminée a priori. Ainsi, lorsqu'un partenaire souhaite s'intégrer dans l'entreprise virtuelle, il doit développer une passerelle appropriée permettant d'interpréter les différents événements échangés. De ce fait, les partenaires sont amenés à adapter leurs passerelles en fonction des partenariats.

- D'autres projets cités dans la littérature tels que : ACE-FLOW [113] (Agile Customer-Supplier Chain and Efficient Process Management with Federated Workflow Systems) qui s'intéresse à la gestion des workflows inter-entreprises et son objectif consiste à automatiser les opérations "business-to-business" des entreprises participantes telles que la spécification et l'exécution d'un Workflow global formé des différents Workflows autonomes et distribués des différents partenaires. Le projet MARVELOUS [114] qui est utilisé dans l'industrie maritime, se base sur des

standards ouverts et des technologies orientées objet pour l'exécution de ces processus métiers. Les relations entre les partenaires sont fortement couplées et pré-définies. Par conséquent, MARVELOUS ne préserve pas l'autonomie des partenaires.

5.11 Synthèse

L'analyse de ces des différentes approches d'interopérabilité de Workflows inter-entreprises selon des critères définis tels que l'autonomie des partenaires, le respect de leur savoir-faire et la flexibilité de la coopération, nous a permis de les classer en deux catégories pour relever certains points :

- Les approches standards : Elles reposent sur l'utilisation des interfaces définies telles que celles proposées par les organisations (WfMC, OMG et OASIS[4]) ou sur l'utilisation des langages de composition de Web services tel que BPEL4WS du BPMI ou encore de se conformer à une infrastructure commune d'interopérabilité telle que celle d'ebXML permettant à tous les participants d'effectuer des transactions commerciales. Ces approches basées sur des interfaces standards ne préservent ni les Workflows ni les systèmes de gestion de Workflow existants. Ils demandent des adaptations et des modifications des parties voire même la totalité des Workflows et/ou des systèmes de gestion de Workflow existants des partenaires. De plus, elles obligent les partenaires à se conformer à des spécifications et/ou un ensemble d'interfaces et de fonctionnalités bien définies. Ce qui réduit l'autonomie des partenaires, et par suite la flexibilité de la coopération.

- Les approches spécifiques : Elles sont adaptées à des projets dans un contexte d'entreprises virtuelles où les partenaires doivent se conformer à des interfaces statiques prédéfinies dès la conception de l'entreprise virtuelle et l'intégration d'un nouveau Workflow pour la coopération exige la conformité de son interface vis-à-vis du partenariat. Ce qui ne préserve pas l'autonomie des systèmes de gestion Workflows et par suite la flexibilité de la coopération.

L'autonomie des partenaires, le respect du savoir-faire des partenaires, la préservation de leurs Workflows pré-établis et l'intégration de leurs systèmes de gestion de Workflows représentent des critères fondamentaux à prendre compte dans un contexte des coopération souple et ouvert. De ce fait, la forme d'interopérabilité que nous adoptons ainsi que l'approche que nous proposons doivent en tenir compte : C'est l'objet du chapitre suivant. Mais avant cela, nous allons d'abord, étudier les principales techniques qui sont utilisées pour la mise en oeuvre de l'interopérabilité.

6. Techniques d'interopérabilité utilisées

Plusieurs techniques peuvent être mises en oeuvre pour la réalisation des projets d'intégration ou d'interopérabilité des systèmes d'information [3], [15], [16], [17]. Ces techniques sont regroupées principalement en deux catégories permettant l'interopérabilité syntaxique et l'interopérabilité sémantique.

[4]Organization for the Advancement of Structured Information Standards :http ://www.oasis-open.org/

6.1 Interopérabilité syntaxique

L'interopérabilité syntaxique est basée sur des formats standardisés ou unifiés, sur des intergiciels de communication et de distribution (middlewares), sur des outils EAI (Enterprise Application Integration), sur des moteurs BPM (Business Process Management), sur des architectures orientées services (SOA – Service-Oriented Architecture) et sur des techniques d'ingénierie à base de modèles.

6.1.1 Techniques ad hoc basées sur des conversions

Elles permettent de définir des passerelles entre les modèles. Pour cela, elles se basent fondamentalement sur l'utilisation de techniques de codage pour relier les modèles de processus entre eux. Cette approche est intuitive (figure 2.1) et nécessite N*(N-1) convertisseurs (N étant le nombre modèles des partenaires).

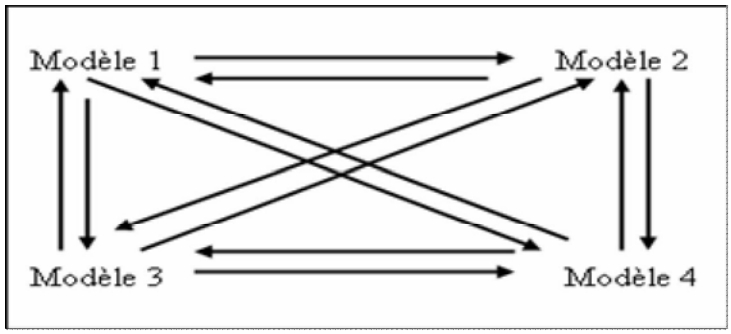

Figure 2.1 : Approche basée conversion.

6.1.2 Techniques basées sur des modèles standards ou formats unifiés

Ces techniques se focalisent sur l'adoption d'un modèle standard ou d'un format unifié pour l'échange de modèles entre différents acteurs coopératifs. Une multitude de langages permettent de représenter les modèles de données, les modèles de processus ou encore les échanges de données ou de processus. En se focalisant sur les échanges de processus, l'analyse de la littérature sur l'interopérabilité des entreprises, issue des travaux du réseau d'excellence européen INTEROP [115] et du projet ATHENA [122], montre qu'il existe deux manières pour échanger des modèles de processus ou coopérer. La première est d'utiliser les standards et la deuxième est d'adopter des formats unifiés.

- **Techniques basées sur des modèles de processus standardisés**

  Ces techniques permettent à des acteurs coopératifs de définir un langage commun pour permettre l'échange des modèles de processus. Elles sont basées sur l'adoption d'un modèle commun qui sera partagé par tous les acteurs de l'interopérabilité. L'approche par modèle commun nécessite seulement 2*N convertisseurs (figure 2.2).

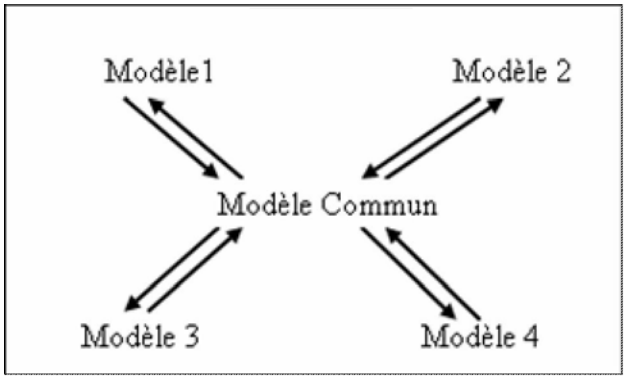

Figure 2.2 : Approche basée modèle commun.

Plusieurs modèles sont utilisés dans l'industrie pour la description et l'échange de processus. Parmi ces modèles, nous pouvons citer principalement : le modèle IDEF3 [116], BPML [100], ebXML [98], RosettaNet [117] et XPDL [4].

- IDEF3 [116] (Integration Definition) est un formalisme défini par la société KBSI en collaboration avec quelques universités, destiné à la capture des séquences d'activités. Il permet de représenter graphiquement les processus sous forme d'un enchaînement d'étapes, appelées unités de comportement. Une unité de comportement est une activité du processus qui est représentée par une boîte rectangulaire. Cette unité est le résultat de la décomposition modulaire hiérarchique et descendante des activités du système étudié. IDEF3 est découpé selon deux perspectives : la perspective processus et la perspective objet.

- BPML (Business Process Modeling Language) [100] est un langage de modélisation des processus métiers dont les premières spécifications sont apparues en 2001 au sein du consortium BPMI. Il permet de définir un modèle abstrait d'interaction entre collaborateurs participant à une activité de l'entreprise, voire entre une organisation et ses partenaires. Les processus métiers sont représentés par un flux de données, un flux d'événements sur lesquels on peut influer en définissant des règles métier, des règles de sécurité, des règles de transactions. Cette norme est à la base de la définition de BPEL4WS qui constitue un standard lié aux Web Services.

- ebXML (e-Business XML) [98] est utilisé dans le cadre des échanges inter-entreprises et particulièrement dans un scénario B2B (Business to Business). ebXML est un standard proposé en 1999 à la fois par l'OASIS[1] et par l'UN/EDIFACT[5] pour les échanges dans le cadre du commerce électronique. ebXML se focalise sur les processus métier et la gestion des transactions. Les processus métiers, au sens ebXML, sont considérés comme l'enchaînement de tâches donnant lieu à l'échange de messages conventionnels (documents commerciaux) entre les partenaires.

[1]Organization for the Advancement of Structured Information Standards :http ://www.oasis-open.org/

[5] United Nations Center for Trande Facilitation : http ://www.unece.org/cefact/

- RosettaNet [117] est un standard similaire à ebXML proposé par le consortium RosettaNet, consortium indépendant d'industriels des secteurs informatique, électronique et semi-conducteurs. Il a pour but de formaliser en XML, le dialogue entre les partenaires d'une transaction commerciale, afin de définir les éléments nécessaires à l'automatisation des Web Services ayant trait à la chaîne logistique.

- PIF (Process Interchange Format) [118] a été développé par un groupe de travail en octobre 1993 comprenant des représentants de plusieurs universités (de Toronto, d'Hawaii, etc.) et de sociétés de développement (KSBI, DEC, etc.) dans le but de partager des modèles de processus. Les spécifications de PIF définissent à la fois un méta-modèle de processus et une syntaxe basée sur KIF (Knowledge Interchange Format). PIF est basé sur un ensemble de concepts permettant d'exprimer des modèles de processus simples. PIF permet à des groupes de travail de définir des extensions locales au modèle à savoir, de nouvelles classes ou de nouveaux attributs.

- XPDL (XML – Process Definition Language) [4], défini par la WfMC (Workflow Management Coalition) en tant que langage standard de modélisation de Workflow, utilise XML comme formalisme d'échange commun de processus entre plusieurs produits Workflows différents. Ainsi, les outils spécifiques de fournisseurs Workflows peuvent transférer leurs modèles via la norme XPDL. Un des éléments importants de XPDL est son extensibilité grâce à l'utilisation de XML, pouvant supporter des attributs spécifiques de fournisseurs Workflows pour l'utilisation dans une définition de processus. Quant à l'utilisation de XPDL par un fournisseur Workflow, elle nécessite des opérations ou des fonctions d'import/export de définition de processus.

- XMI (XML Metadata Interchange) [119] est un format d'échange en XML, conçu pour permettre l'échange de modèles UML. Le standard XMI permet de rendre tout modèle compatible MOF (Meta-Object Facility) [120] de MDA (Model Driven Architecture) en un document XML et d'assurer ainsi, l'échange de modèles qui sont compatibles au MOF.

- **Techniques basées sur des formats unifiés**

  Basées sur ces techniques, les entreprises se mettent d'accord pour unifier la manière de présenter leurs données, processus ou applications, mais aussi leurs outils, méthodes et stratégies pour faciliter l'interopérabilité de leurs systèmes d'information. Dans le domaine de processus, certains projets européens tels que INTEROP [121] et ATHENA [122] ont développé des modèles unifiés pour réaliser l'interopérabilité des modèles d'entreprises. Le modèle UEML [3], [123] *(Unified Enterprise Modeling Language)* est un exemple de ces modèles. Il a été adapté dans le projet INTEROP comme un langage commun de modélisation d'entreprise. C'est un langage unifié de modélisation d'entreprise et un langage pivot pour l'échange de modèles d'entreprises différents. UEML peut être utilisé pour échange d'information entre des outils de modélisation d'entreprise. Similairement, POP* est proposé dans le projet ATHENA comportant un ensemble de concepts de modélisation de base communs pour supporter la modélisation et l'échange de modèles entre des outils différents.

6.1.3 Techniques basées sur l'ingénierie des logiciels

Ces techniques d'interopérabilité sont basées sur des intergiciels de communication et de distribution (middlewares) qui sont des logiciels principalement dédiés à la gestion des communications inter-applications. Ces intergiciels sont des outils qui peuvent être mis en oeuvre en vue de l'intégration syntaxique et sont principalement destinés à prendre en charge l'aspect communicationnel de l'intégration des applications ou des systèmes.

Ils permettent l'interopérabilité des applications distribuées hétérogènes. Dans le milieu industriel, différentes communautés ont développé des middlewares implémentés dans des environnements de développement dont le principe est de masquer toute l'hétérogénéité technique liée aux plates-formes (langages de programmation, systèmes d'exploitation et réseaux, etc.).

Parmi ces environnements, nous citons brièvement :

- *L'environnement CORBA* (Common Object Request Broker Architecture) est une architecture proposée par l'OMG (Object Management Group), qui offre une solution au problème d'intégration des applications distribuées hétérogènes à partir des technologies orientées objet. Ces applications distinctes sont donc, orientées objet, ne sont pas nécessairement implémentées dans les mêmes langages, et peuvent être déployées sur différents systèmes. Comme toutes les autres solutions distribuées, elle exploite les objets distribués et le principe d'appel à des méthodes distantes. La particularité de l'architecture CORBA réside dans l'intégration de services implantés par des fournisseurs de logiciels en exploitant le langage IDL (Interface Definition Language) et le protocole IIOP (Internet Inter-ORB Protocol), pour la communication sur le réseau afin de faciliter l'échange entre applications distantes indépendamment des systèmes informatiques ou langages utilisés.

  Dans cette architecture, les objets dialoguent par l'intermédiaire du bus CORBA qui se charge de masquer les différences entre les langages d'implantation des objets, les systèmes d'exploitation et les architectures matérielles. De plus, ce bus rend totalement transparente la localisation des objets permettant ainsi de simplifier l'utilisation des objets sur le réseau. L'approche que nous avons préconisée dans [130] pour l'interopérabilité des modèles de Workflow est basée sur cet environnement, qui est une approche à base d'événements au dessus de CORBA.

- *L'environnement .NET* est environnement de développement de Microsoft basé sur la technologie COM (Component Object Model)/DCOM (Distributed COM) [125], [126] qui sont des composants qui ne peuvent être utilisées que sous les systèmes d'exploitation Windows afin de faciliter l'interopérabilité entre des applications réparties hétérogènes et/ou distantes. Elle est proche de l'architecture CORBA. DCOM est une extension de COM pour le traitement distribué. Il est basé sur l'environnement DCE (Distributed Computing Environment), dont il exploite en particulier les appels de procédures distantes RPC (Remote Procedure Call), comparables dans leur principe au système RMI. Par conséquent, à chaque fois qu'on envoie un message DCOM à un objet distant, ce message est en réalité transporté via RPC.

Les objets COM et leurs interfaces sont spécifiés en utilisant le langage de description d'interface propriétaire de Microsoft qui s'appelle MIDL (Microsoft Interface Description Language) pour uniformiser la représentation des applications. Le protocole SOAP (Simple Object Access Protocol) est utilisé pour permettre l'invocation de méthodes à distance à travers le réseau Internet et utilise l'interface WSDL (Web Service Description language) écrite à l'aide du langage XML..

.

- *L'environnement Java* qui est proche de celui défini par CORBA. Il se base sur le langage Java pour définir des composants homogènes. La stratégie de Java consiste à tout redéfinir dans son langage de programmation objet. Plus qu'un langage, Java est surtout une plate-forme d'interopérabilité de systèmes informatiques. Cet environnement a été conçu pour apporter des solutions aux mêmes types de problèmes d'interopérabilité que ceux traités par le bus CORBA. Pour fournir un cadre de travail homogène, la communauté Java a basé son environnement sur l'utilisation d'un langage unique de programmation. Donc, toute application développée dans ce langage peut interagir avec l'ensemble des applications Java existantes. De ce fait, la communauté Java s'est focalisée surtout sur l'aspect universel et multi plateformes de ce langage. En effet, Java est un langage objet interprété. Cette plate-forme utilise un compilateur de code lors de l'exécution du programme. Une machine virtuelle, chargée de la compilation, est définie pour chaque système d'exploitation existant.

  Pour faciliter l'interopérabilité entre applications distribuées, la communauté Java a défini une architecture ou un environnement appelé J2EE (Java 2 Platform, Enterprise Edition) [127] constitué de différentes technologies et d'APIs telles que EJB (Enterprise Java Beans), (JDBC – Java DataBase Connectivity), JSP (Java Server Pages), (RMI - Remote Method Invocation), (JMS - Java Message System), RMI-IIOP (Internet Inter-ORB Protocol), JTS (Java Transaction Service).

6.1.4 Techniques basées sur des outils EAI

Les outils EAI (Enterprise Application Integration) constituent un type d'outils récent dans le monde industriel pour interconnecter des applications hétérogènes. Ces outils permettent en effet, de faire communiquer tout type d'application : CRM (Customer Relation Management), ERP (Enterprise Resource Management Planing) ou SCM (Supply Chain Management) (figure 2.3). Plusieurs définitions ont été données au concept d'EAI [135], [124] [134]. Pour [124] l'EAI est défini comme étant « *le processus qui consiste à réunir dans un même contexte les machines physiques, les logiciels et les processus métiers de telle sorte qu'une fois que ces éléments sont combinés, les composants sont parfaitement interfacés, l'information est aisément partagée et les systèmes collaborent en synergie* ».

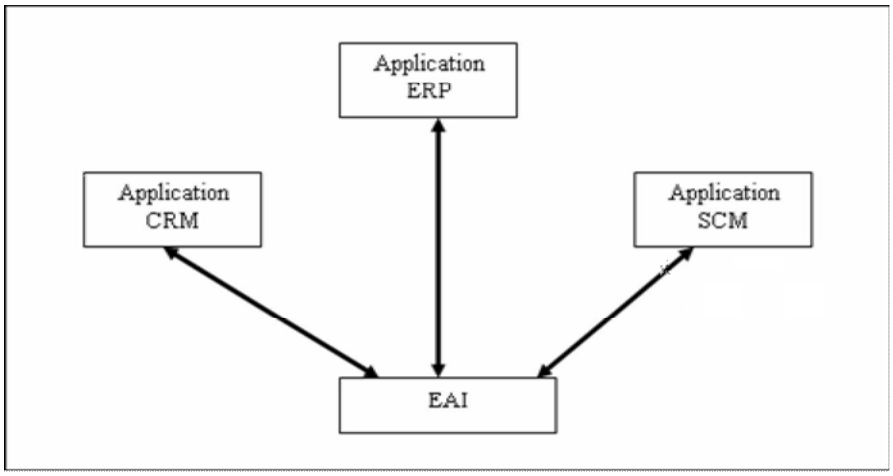

Figure 2.3 : Principe de l'EAI

Parmi les outils EAI les plus populaires du marché, on peut citer principalement [135][136] : BEA Weblogic Integration, Microsoft Biztalk, IBM Websphere Business Integration, SeeBeyond Integration Platform, Tibco Active Enterprise, Vitria BusinessWare, WebMethods Integration Platform.

Selon [134], EAI est défini comme étant "*un outil intégré supportant trois grandes fonctions : transport, connectivité, transformation/adaptation des flux de données, et l'automatisation/orchestration des processus*". Il s'agit en quelque sorte du mariage d'une multitude d'intergiciels (pour le transport et la connectivité), des outils ETL (Extract, Transform, Load) et des moteurs de Workflow. Les outils EAI permettent généralement l'utilisation du format pivot pour assurer la communication entre les applications hétérogènes.

L'architecture générale d'un outil EAI [17] [137] est composée de quatre couches permettant la liaison entre les processus de l'entreprise et ses applications. Ces couches permettent au logiciel d'EAI de récupérer des données d'une application, puis les router vers leurs applications destinatrices après les avoir converties dans le format approprié. Les couches de transformation et routage sont souvent regroupées dans la littérature sous l'expression de "moteur d'intégration".

Le nombre des connexions nécessaires pour intégrer *N* modèles est donc de l'ordre de *2*N*. Parmi les limites majeures de ces outils, on peut citer leur lourdeur, leur caractère propriétaire, ce qui limite considérablement la flexibilité. La logique d'intégration d'un EAI étant propriétaire, les éléments de l'outil (connecteurs, transformateurs de données, orchestrateur des processus) ne sont pas standardisés, ce qui lie l'entreprise à l'éditeur qu'elle aura choisi, avec toutes les conséquences coûteuses engendrées tels que coût des interventions ou maintenance.

Cependant, pour pallier aux limites apportées par les outils EAI, de nouvelles approches orientées services comme SOA et la technologie des Web services ont émergé pour modifier le principe de l'intégration. Ainsi, une nouvelle technologie a été introduite en 2003 par le Gartner Group : il s'agit du concept d'ESB (Enterprise Service Bus) ou Bus de Services d'Entreprise [138] que nous présentons dans la section suivante.

6.1.5 Techniques basées sur le bus ESB

Ces techniques sont basées sur l'utilisation d'un nouveau type de middleware appelé " ESB (Enterprise Service Bus) " [138]. La technologie ESB est centrée sur la notion de bus, qui permet d'assurer une intégration distribuée des différents services métiers (de fournisseurs et de consommateurs) connectés au bus (figure 2.4).

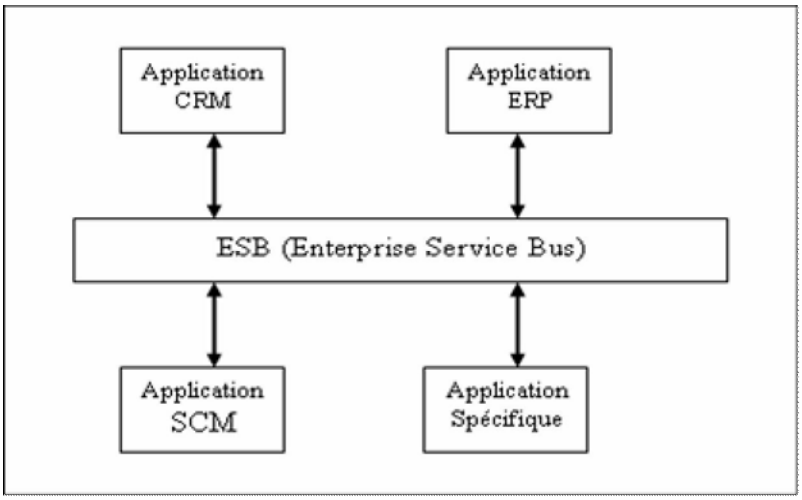

Figure 2.4 : Principe d'intégration basé ESB.

C'est une solution d'intégration distribuée basée sur un middleware orienté message (Message-Oriented Middleware ou MOM) et fournissant des services de transformation des données, de routage, en s'appuyant sur l'utilisation systématique des standards des Web services (SOAP, WSDL, UDDI) [16]. C'est donc, une évolution du concept d'EAI combinant MOM, Web services, transformation et routage.

Le bus permet aux différents services métiers de modèles de processus de communiquer et de coopérer. Il se base sur certains services de base [139] :

- le moteur d'orchestration jouant le rôle de service d'orchestration, permet d'exécuter des processus basés sur BPEL4WS;
- le service de localisation qui permet de trouver de façon transparente les services;
- les services utilitaires qui sont des services techniques et qui sont sollicités par les services métiers, permettent le routage, la transformation et le support des transactions.
- les services d'infrastructure qui permettent de fournir un support système et d'infrastructure aux services métiers tels que les services liés à la sécurité et au monitoring.

[140] présente les fonctionnalités principales d'un ESB (figure 2.5) :

- Transmission de messages : une demande d'un service (message) est transmise au fournisseur de service ;
- Routage : permet d'envoyer la demande de service au fournisseur de service nécessaire, en utilisant des règles prédéfinies ;
- Transformation : permet de transformer le format de message en un autre format ;

- Adaptateur : généralement dans un ESB, les applications utilisent le standard SOAP pour l'échange de messages. Cependant, pour des applications ne supportant pas le protocole SOAP, l'utilisation des adaptateurs est nécessaire pour transformer les messages ;
- Orchestration : un moteur d'orchestration permet de gérer les flux de contrôle d'un service à un autre, constituant les processus métiers.

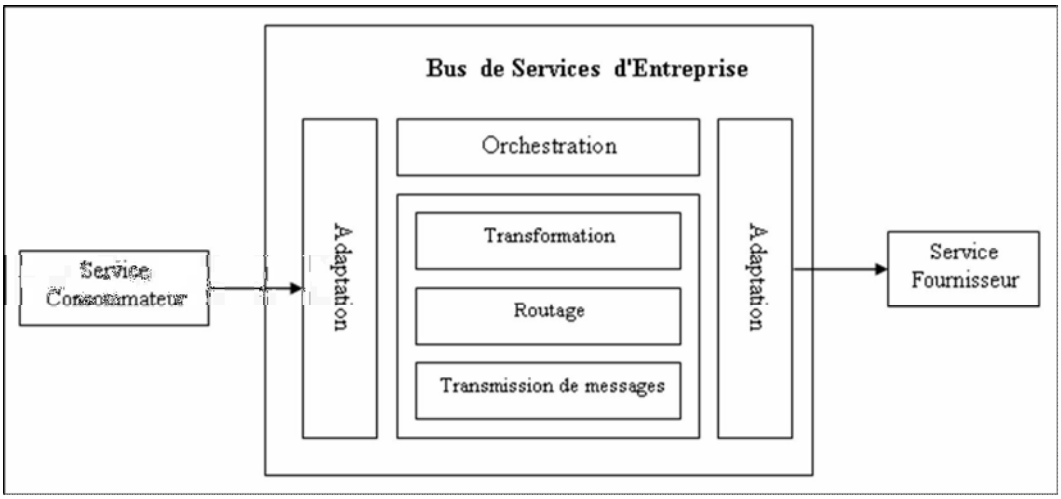

Figure 2.5 : Architecture d'un ESB [140].

ESB est donc, une nouvelle architecture qui exploite les Web services et les normes qui lui sont associées telles que la sécurité, l'orchestration des services, le middleware de la messagerie, le routage et la transformation.

6.1.6 Techniques basées sur des outils de gestion de processus BPMS

Les outils de gestion de processus BPMS (Business Process Management System) [141], [142], [143] sont basés sur la notion de processus métier qui est très souvent confondue avec la notion de Workflow. Ce dernier constitue une automatisation totale ou partielle d'un processus métier.

Le principe d'intégration des processus d'entreprise via les moteurs BPM et/ou systèmes Workflow est d'offrir une vue simplifiée à un utilisateur métier. Le BPMS constitue en effet, un médiateur entre des acteurs humains et des processus (ou applications) à interagir avec les processus (figure 2.6). Il permet d'orchestrer les services d'un processus interne propre à une entreprise et de chorégraphier l'enchaînement des opérations à réaliser dans une collaboration inter-entreprises. Il s'appuie pour cela sur les nouvelles technologies tels que les Web services ainsi que les différentes langages de composition de services tel que BPEL4WS (Business Process Execution Language for Web Services).

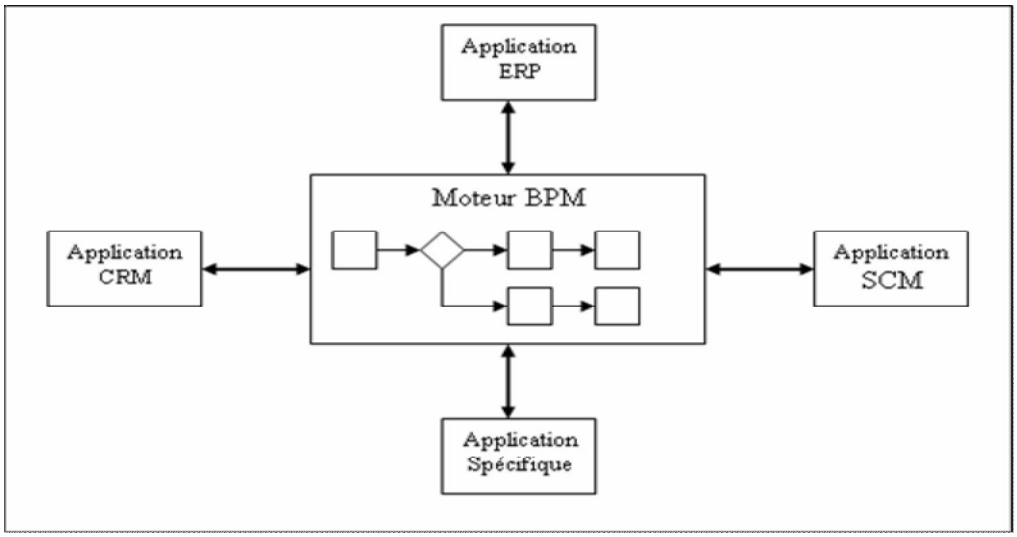

Figure 2.6 : Principe d'intégration par le BPM.

Il est important de noter que le BPM n'impose généralement aucune contrainte sur le type de participant à utiliser dans les scénarios d'intégration. Une des approches les plus pertinentes à l'heure actuelle, pour mettre en oeuvre les techniques basées sur le BPM est de s'inscrire dans le contexte des Web services et des architectures orientées services. A cet effet, de nombreux standards de BPM existent pour orchestrer les Web services tels que BPEL4WS, XLANG, ou WSFL.

6.1.7 Techniques basées sur l'architecture de services SOA

D'autres techniques relatives à l'interopérabilité ou plutôt à l'intégration commencent à investir les milieux industriels : il s'agit principalement de la technique basée sur les architectures orientées services. L'architecture orientée services (SOA - Service-Oriented Architecture) constitue un style architectural de développement et d'intégration dynamique des applications d'entreprise. C'est un style d'architecture fondée sur la description de services et de leurs interactions. La notion d'architecture orientée services n'est pas nouvelle, elle est apparue dès le développement des approches client/serveur.

Elle se base sur la notion de service qui est considéré comme l'élément de base de cette architecture. La notion de service définie dans le modèle SOA, correspond à une fonctionnalité offerte par un composant applicatif et dont l'interface est indépendante de la plate-forme ou de la technologie utilisée pour le développement du composant.

Le principe de cette architecture consiste alors à structurer le système d'information d'entreprise comme un ensemble de services qui exposent leur interface fonctionnelle et qui communiquent par messages. Dans une SOA, l'accent est mis sur le service. Par conséquent, le développement des applications est basé sur un modèle orienté service, ce qui est différent d'un modèle orienté composant. Un modèle orienté service se concentre sur les exigences déterminées au niveau de la stratégie et du processus métier, alors que le modèle orienté composant se concentre sur les composants du logiciel utilisé pour livrer les services.

❖ **Le modèle SOA**

Dans le modèle SOA, chaque service est défini par un fournisseur. Le fournisseur de services publie (publish) la description de son service dans des registres de services spécialement conçus en vue d'être interrogés par des clients. Les clients de services (applications clientes) localisent (find) leurs besoins en termes de services en effectuant des recherches sur le registre de services. Une fois le service localisé, le client récupère sa description du registre et sur la base des informations fournies dans la description du service, le client interagit (bind) alors avec le service en vue de l'exécuter.

Dans ce modèle, nous avons fondamentalement trois principaux rôles: le fournisseur de services (service provider), le client du service (service requestor) et le registre de services (service registry) ainsi que trois types d'opérations basiques (figure 2.7) : publier (publish), rechercher (find) et interagir (bind) [114], [145].

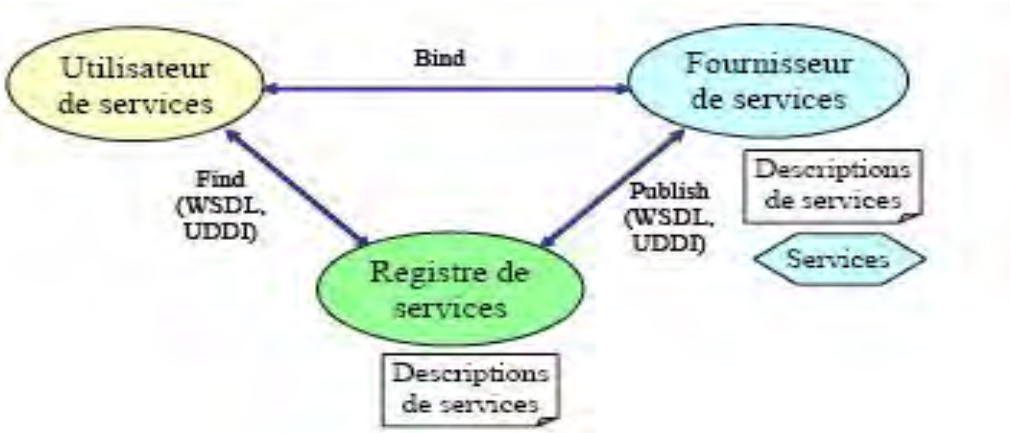

Figure 2.7 : Principe des SOA [145].

❖ **Les Web services**

Plusieurs définitions des Web Services existent dans la littérature [10], [146], [147], [148]. Selon le dictionnaire Webopedia [149], un Web Service est défini comme étant « une manière standardisée d'intégration des applications basées sur le Web en utilisant les standards ouverts XML, SOAP, WSDL, UDDI et les protocoles de transport de l'Internet. XML est utilisé pour représenter les données, SOAP pour transporter les données, WSDL pour décrire les services disponibles, et UDDI pour lister les fournisseurs de services et les services disponibles ».

Les Web services fournissent le support pour la description et l'infrastructure de communication des services, alors que SOA décrit comment un système composé de services peut fonctionner. Les principaux apports de la technologie des Web services sont notamment l'universalité, la standardisation, le couplage faible entre les services, et la virtualisation qui permet de définir une indépendance vis à vis de la localisation et de l'implémentation des services. Ils permettent alors aux applications d'entreprises de dialoguer ensemble à distance via Internet, indépendamment des plates-formes techniques et des langages de programmation.

❖ **Composition des Web services**

La composition des Web services est un processus par lequel un Web service est créé par l'enchaînement ou l'orchestration de deux ou plusieurs Web services. L'objectif de cette

composition est alors de faciliter l'intégration d'applications afin de favoriser les échanges d'informations entre entreprises partenaires dans un contexte B2B (Business to Business).

La composition des Web services a plusieurs similarités avec la technologie de Workflow [150]. Les deux ont pour but de spécifier un processus métier par la composition des entités autonomes, de simple et de forte granularité. Leur différence réside dans la nature de l'entité. Dans le cas d'un Workflow, les entités sont des activités, des tâches, des sous-processus ou des applications conventionnelles invoquées, alors que dans celui des Web services, les entités sont des services. Différents langages de composition de services sont intégrés dans la pile standard des Web services (BPEL4WS, XLANG, WSFL, etc.) et précisément au niveau de la couche métier afin de faciliter la composition de ces services.

Parmi ces langages, BPEL4WS est considéré comme le langage de composition de Web services le plus adapté pour la description des processus métier intra et inter-entreprises. Il présente beaucoup d'avantages par rapport aux autres langages. BPEL4WS contient plusieurs caractéristiques de XLANG et WSFL, ainsi, il est défini comme un langage structuré en blocs et graphes directs. De plus, il est basé sur XML et étend WSDL. BPEL4WS contient des éléments qui rendent la création de processus métiers abstraits et exécutables possibles.

Dans le domaine du Workflow (ou business process), l'idée fondamentale derrière cette composition, est de morceler les applications et les processus d'affaires ou métiers en morceaux réutilisables appelés « Service » de sorte que chacun de ces segments effectue une tâche distincte. Ces services peuvent alors servir à l'intérieur et à l'extérieur de l'entreprise, facilitant ainsi, l'interopérabilité ou plutôt l'intégration entre tous ses services. Par leurs caractéristiques majeures (se basant sur des protocoles industriels standardisés : XML, SOAP, WSDL, UDDI)), les Web services [10] :

- Permettent d'interconnecter différentes entreprises, différents matériels, différentes applications ou différents clients pour échanges de données.
- Font interagir des composants hétérogènes, distants, et indépendants avec un protocole standard (SOAP).
- Permettent d'intégrer des applications ou différents processus métiers d'une ou de plusieurs organisations pour réaliser un objectif organisationnel commun.
- Offrent aux partenaires d'une entreprise (clients, fournisseurs, sous-traitants, etc.) un accès direct à la fonctionnalité dont ils ont besoin.
- Permettent aux entreprises d'intégrer des applications de bout en bout, de manière facile à implanter autant hors des frontières qu'à l'intérieur de celles-ci.

❖ **Apports de SOA pour l'interopérabilité des modèles de Workflow**

Dans une perspective d'intégration ou d'interopérabilité de modèles de processus (ou business processus), les entreprises cherchent en premier lieu, à réduire leurs coûts d'intégration, s'ouvrir à leur environnement, gagner en flexibilité afin de s'adapter facilement face l'évolution du marché et réagir rapidement aux changements pour adapter ses processus métiers internes et externes aux exigences des clients et aux comportements des concurrents.

Dans ce contexte d'intégration ou d'interopérabilité, SOA contribue à l'évolutivité, la flexibilité et la pérennité d'un système d'information et offre beaucoup d'avantages. Nous en citons seulement quelques uns :

- Flexibilité : SOA donne la capacité à un système d'information ou un processus métier de s'intégrer facilement pour pouvoir interagir avec d'autres systèmes ou processus métiers. Elle est liée à la manière d'appeler (ou découvrir) un service indépendamment de sa logique d'implémentation et de le composer dynamiquement avec d'autres services.

- Réutilisation : SOA permet de favoriser la mutualisation et la réutilisation par variantes de services. Il est donc plus facile de réutiliser un service puisqu'il n'est pas directement lié à d'autres services.

- Ouverture : SOA est une architecture ouverte, basée sur un ensemble de standards industriels pour le développement, le déploiement et l'intégration d'applications Internet tels que XML, SOAP, WSDL, UDDI, BPEL4WS.

- Urbanisation : le système d'information de l'entreprise est découpé en services. le niveau de granularité n'est plus l'application mais le service. SOA permet une organisation (ou réorganisation) et une structuration efficaces du système d'information, en considérant les multiples composants d'un système d'information comme des services d'entreprise. Le système d'information de l'entreprise est alors découpé en services. Le niveau de granularité n'est plus l'application mais le service. [16], [151].

- Uniformisation : grâce à cette urbanisation orientée service, SOA permet de représenter de façon uniforme tous les composants d'entreprise concernés par le projet d'intégration ou d'interopérabilité des processus métiers.

- Intégration à faible couplage et à faible coût : SOA permet l'intégration à faible couplage et à faible coût de systèmes d'information hétérogènes. Dans le domaine du Workflow ou dans un contexte B2B (Business to Business), SOA contribue à intégrer des processus inter-entreprises, en permettant à différents logiciels de gestion tels que les ERP (Enterprise Resource Planning), les CRM (Customer Relationship Management), les SCM (Supply Customer Management) et autres de communiquer entre eux. On parle alors de composition, d'orchestration ou de chorégraphie de Web services pour dire combinaison ou interaction de Web services élémentaires afin d'obtenir des services métiers plus élaborés.

  Par contraste, cette intégration à l'aide de technologies classiques nécessite la conception de ponts logiciels ad-hoc coûteux et à fort couplage. L'entretien de telles solutions nécessite une forte cohésion entre les différents acteurs impliqués dans l'intégration.

- Encapsulation de la complexité des applications ou des processus : il s'agit de fragmenter puis de structurer les applications ou les processus en des services plus réduits et potentiellement plus simples à faire évoluer [152]

- Améliorant son agilité : SOA permet de structurer d'une manière dynamique un système d'information. Ce dynamisme est lié à la possibilité de changer facilement une configuration de services : principe de couplage faible (ajouter ou enlever un service, modifier l'ordre de communication de services). De plus, il est possible d'intervenir sur le contenu de service sans toucher à son interface d'accès (changement interne) [17].

6.1.8 Techniques basées sur l'ingénierie des modèles (MDA)

Ces techniques sont utilisées par l'OMG (Object Management Group) [153], [154] pour faire face à l'hétérogénéité des technologies existantes et à venir utilisées dans le cadre du génie logiciel. Elles sont préconisées par MDA (Model-Driven Architecture) en tant que nouvelle orientation ou approche d'intégration et d'interopérabilité d'applications.

L'approche MDA est orientée modèle parce qu'elle fournit les bases pour l'utilisation des modèles pour guider la compréhension, la conception, la construction, le déploiement, la maintenance et la modification des systèmes développés. Les trois principaux objectifs du MDA sont la portabilité, l'interopérabilité et la réutilisabilité à travers la séparation de l'aspect dépendant de la plateforme ou de l'application étudiée et de l'aspect plus abstrait indépendant de l'application. L'objectif principal est tout d'abord, séparer les éléments que l'on peut considérer stables dans la logique métier de l'entreprise, de l'évolution des supports technologiques. En effet, le métier des entreprises est stable et capitalise le savoir-faire de l'entreprise et n'évolue que faiblement par rapport aux technologiques qu'elles utilisent.

❖ **Principe de MDA**

Dans le contexte MDA, tout est considéré comme modèle, aussi bien les spécifications des applications que le code source ou le code binaire. Le principe de l'approche MDA consiste alors en l'élaboration de modèles indépendants des plates-formes (PIMs) qui sont des modèles métiers et la transformation de ceux-ci en modèles dépendants des plates-formes (PSMs) qui sont des modèles techniques spécifiques et cela pour préparer et faciliter la génération de code vers une plate-forme technique choisie. Les techniques employées sont principalement des techniques de méta-modélisation et de transformation de modèles.

L'approche MDA permet donc, de réaliser le même modèle sur plusieurs plates-formes grâce à des projections standardisées. Elle permet aux applications d'interopérer en reliant leurs modèles et supporte l'évolution des plates-formes et des techniques. La mise en oeuvre du MDA est entièrement basée sur les modèles et leurs transformations. La transformation de modèles est le processus qui transforme un modèle en un autre modèle. Plus précisément, un PIM suffisamment détaillé est projeté vers un PSM.

Une transformation de modèles est donc une fonction dont les paramètres d'entrée et de sortie sont des modèles structurés par des méta-modèles. Si cette transformation s'exécute au niveau des modèles, elle se spécifie au niveau des méta-modèles. En effet, une transformation exprime des correspondances structurelles entre les modèles source et cible. Ces correspondances structurelles s'appuient sur les méta-modèles des modèles source et cible.

De nombreux travaux importants dans le domaine MDA existent. Ce sont notamment les travaux portant sur les transformations de modèles. Une fois les règles de transformation définies, il est nécessaire de disposer d'un langage de spécification de ces règles. Quelques

outils MDA existent tels que XSLT (Extensible Stylesheet Language Transformation) [155] ATL (Atlas Transformation Language) [156], ModFact [157], MTRANS [158], etc. Pour la standardisation d'un langage de transformation de modèles, l'OMG a défini MOF 2.0 QVT (Query/-Views/Transformations) (QVT) [142]. Le langage ATL est un langage hybride regroupant à la fois les paradigmes de programmations déclarative et impérative et qui répond à la spécification MOF 2.0 QVT. La figure 2.8 présente le principe d'une transformation

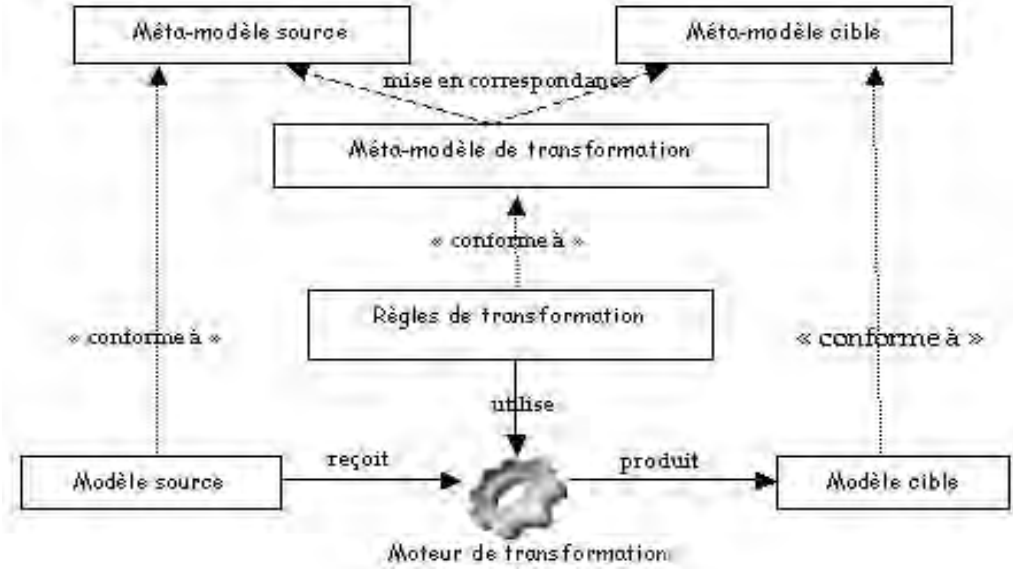

Figure 2.8 : Principe Transformation de modèles en MDA.

❖ **Architecture à quatre niveaux**

Les modèles manipulés par MDA sont de natures diverses : modèles de processus, modèles de service, modèles de plates-formes, etc. Afin d'organiser et de structurer tous ces modèles, l'OMG a défini une architecture à quatre niveaux. Ces quatre niveaux sont :

- Le niveau M0 qui est le niveau des données réelles, est considéré comme le monde réel.
- Le niveau M1 permet de décrire les informations du niveau M0. Typiquement un modèle UML appartient au niveau M1.
- Le niveau M2 appelé méta-modèle, est composé de langages de définition des modèles d'information. Le méta-modèle UML qui est décrit dans le standard UML appartient au niveau M2.
- Le niveau M3 appelé méta-méta-modèle ou le MOF (Meta-Object Facility) [20], est le langage unique de définition des méta-modèles. Ce dernier définit la structure de tous les méta-modèles qui se trouvent au niveau M2. Le MOF est réflexif, ce qui permet de dire que le niveau M3 est le dernier niveau de l'architecture.

L'OMG a déjà défini plusieurs standards pour le MDA dont les plus importants sont le MOF (Meta-Object Facility) [120], UML [Unified Modeling Language), préconisé pour l'élaboration de PIMs et de PSMs et XMI (XML Metadata Interchange) [119] qui permet de rendre tout modèle compatible MOF en un document XML et d'assurer ainsi, l'échange de modèles qui sont compatibles au MOF.

❖ Apports de MDA pour l'interopérabilité

Les apports de MDA pour l'interopérabilité ont été cités par beaucoup d'auteurs [3], [15], [17], [18], [72], [158], [159], [160] que nous résumons par les points suivants :

- *L'augmentation de la portabilité et de l'interopérabilité* des systèmes, des applications à travers les techniques de méta-modélisation et de transformation des modèles.

- *L'augmentation des niveaux d'abstraction* par lesquels les humains peuvent communiquer avec d'autres humains de manière plus productive.

- *La pérennité du savoir-faire*, afin de permettre aux entreprises de capitaliser leur métier sans à se soucier de la technique. En effet, le métier des entreprises n'évolue que faiblement par rapport aux technologies qu'elles utilisent pour construire leurs applications informatiques. Cela permet de rendre les spécifications métier pérennes, indépendamment des spécifications techniques.

- *Les gains de productivité*, afin de permettre aux entreprises de réduire les coûts de mise en œuvre des applications informatiques nécessaires à leur métier. Il est toujours important pour une entreprise d'augmenter sa productivité afin de rester compétitive. Une façon bien connue d'augmenter la productivité est d'automatiser certaines étapes de production. C'est ce que fait MDA en préconisant l'automatisant des transformations de modèles.

- *La prise en compte des plates-formes techniques d'exécution*, afin de permettre aux entreprises de bénéficier des avantages des autres plates-formes sans souffrir d'effets secondaires.

❖ MDA et l'interopérabilité des modèles de Workflow

Plusieurs travaux ont porté sur l'interopérabilité des modèles d'entreprises au niveau de plusieurs projets européens tels que ATHENA [121], INTEROP [122] et de certains projets industriels tels que [15], [72], [158], [160] qui ont montré la maturité de MDA, ainsi que leurs apports dans un contexte réel de coopération inter-entrprises.

Dans le domaine du Workflow, parmi les travaux qui ont particulièrement attiré notre attention, est celui de [72] où l'auteur s'est concentré principalement sur la mise en oeuvre du processus MDA dans un projet réel dans une entreprise industrielle. Ce projet concerne un processus métier de tierce maintenance applicative et consiste à sous-traiter la maintenance d'un parc logiciel à une société tierce, entrant dans le cadre d'un partenariat entre l'entreprise industrielle Sodifrance et l'université de Nantes (France).

Pour réaliser l'interopérabilité des processus Workflows, l'auteur a proposé une chaîne d'ingénierie de modèles de processus, permettant de définir de processus Workflows, de les transformer pour alimenter certains outils du marché (outils de planification, moteurs de Workflows etc.), ou au contraire de récupérer des modèles de Workflow pré-existants. La solution développée pour résoudre le problème de l'interopérabilité s'articule autour d'un méta-modèle commun de processus. Ce dernier est conçu à partir de différents formalismes de représentation des processus et qui représente le bus d'interopérabilité des modèles.

La chaîne d'ingénierie de processus à mettre en œuvre est constituée ainsi, de trois outils basés sur le formalisme MOF de MDA, chacun ayant un rôle spécifique :

- ✓ Un outil de modélisation graphique pour la définition des processus Workflows ;
- ✓ Un outil de génération *"Script-G"* pour la génération des fichiers textuels en format XML des modèles sources ;
- ✓ Un outil de transformation *"Script-T"* pour la transformation des modèles générés afin d'alimenter des moteurs Workflows ou d'autres outils ou progiciels tel que celui de MS-Project utilisé pour la planification de la société Microsoft.

Pour pallier aux limites de MDA en termes de spécification du comportement des processus, l'auteur suggère d'exprimer les règles qui vont spécifier ce comportement. Pour ce faire, il propose d'intégrer dans l'architecture MDA au niveau du MOF, les concepts de base d'AS_UML (Action Semantics for UML) pour exprimer la sémantique des opérations associées aux entités du méta-modèle. La solution préconisée consiste alors à reprendre AS_UML et à le remonter au niveau du MOF, comme il a été déjà fait pour OCL (Object Constraint Language), qui était à l'origine dédié à UML, et qui a été ensuite introduit dans le MOF. De ce fait, l'auteur affirme avoir disposer des mécanismes permettant d'exprimer la sémantique des opérations, qui sont au fait, des actions sur chacune des entités du méta-modèle pour le rendre opérationnel. Donc, élever AS_UML au niveau du MOF et le baptiser "Action Semantics for MOF" permettrait ainsi, de spécifier le comportement des entités dynamiques d'un méta-modèle.

6.1.9 Discussions

L'analyse des différentes techniques d'interopérabilité syntaxique vis-à-vis de notre problématique conceptuelle et des objectifs visés, nous a permis de recenser les points essentiels suivants :

- Les techniques basées sur l'ingénierie des logiciels sont fortement liées aux environnements de développement tels que CORBA, DotNet ou la plate forme Java. Ce qui ne leur permet pas d'être réutilisés facilement ailleurs. Ce qui implique un manque en matière de flexibilité.

- Les techniques basées sur des outils EAI permettent d'intégrer l'ensemble des fonctionnalités de l'entreprise au niveau d'une application monolithique. Leur principale limite est leur lourdeur et leur caractère propriétaire. Leur utilisation est beaucoup plus technique, ce qui limite considérablement la flexibilité.

- Les techniques basées sur le modèle architectural SOA contribuent à l'évolutivité, à la flexibilité et la pérennité d'un système d'information et fournit une réponse conceptuelle adaptée à la problématique de l'interopérabilité. Cependant, sa principale limite est que, c'est un modèle point à point qu'on doit se conformer selon la vision SOA (consommateur et fournisseur de services).

- Les techniques basées sur les outils BPMS s'appuient principalement sur la technologie des Web services et les langages de composition de Web services. Leur utilisation est beaucoup plus technique, Leur principale limite est de s'inscrire dans un contexte des Web services et des architectures orientées services (SOA).

- Les techniques basées sur le bus ESB sont orientées vers une utilisation technique qui exploite aussi, les Web services et les normes qui lui sont associées et impose à l'utilisateur de se connecter au bus pour pouvoir intégrer son processus.

- Les techniques basées MDA nous offrent un niveau d'abstraction élevé en utilisant les techniques de méta-modélisation et de transformation de modèles et contribue à la flexibilité en définissant un modèle indépendant des plateformes qui, après raffinements successifs, se projette dans un modèle spécifique à une plateforme. Cependant, l'application de MDA implique parfois, un processus complexe de transformations dans un contexte standard.

Pour résumer, nous dirons que cette synthèse nous a permis de dégager essentiellement deux techniques, qui nous permettent de tenir compte des trois critères importants que nous avons fixés dans la thèse à savoir, l'aspect conceptuel et les aspects d'ouverture et de flexibilité.

Ces deux techniques sont complémentaires et peuvent être mises en oeuvre dans le cadre de l'interopérabilité des modèles de processus, mais dans un contexte industriel. Il s'agit de l'approche basée sur l'ingénierie des modèles (MDA) et des architectures orientées services (SOA). Ces deux approches prises conjointement peuvent favoriser considérablement l'interopérabilité des modèles de Workflow dans un contexte standard.

Cependant, la combinaison de ces deux techniques ne permet pas de prendre en charge l'aspect sémantique de façon explicite afin d'éviter des ambiguïtés d'interprétation des modèles de processus lors l'échange et de la coopération. Pour favoriser l'interopérabilité sémantique, d'autres techniques ont été introduites dans certains milieux industriels. Il s'agit des techniques d'interopérabilité sémantique : C'est l'objet de la section suivante.

6.2 Interopérabilité sémantique

L'interopérabilité sémantique est basée sur la notion d'ontologie qui est utilisée comme l'un des moyens les plus efficaces pour la représentation formelle de la sémantique. Dans un contexte industriel, plusieurs approches sémantiques peuvent être mises en œuvre. Ces approches sont dites "approches orientées services" et se basent principalement sur le concept de Web services sémantiques [16], [151] qui constitue à notre sens l'un des moyens les plus efficaces pour mettre en œuvre l'interopérabilité sémantique des modèles de processus via le Web. Ce concept a pour objectif de combler certaines limites des Web services traditionnels qui ne permettent pas de prendre en compte l'aspect sémantique.

Dans ce contexte, plusieurs travaux existent qui sont basés sur le concept du Web service sémantique [161], [162], [163], [164] et s'articulent principalement autour du développement des langages de représentation d'ontologies basés Web pour la description sémantique des Web services.

6.2.1 Techniques basées sur des langages de représentation d'ontologies basés Web

Parmi les langages de représentation d'ontologies langages basés Web, nous citons principalement : OWL-S [21], WSMF [165], WSMO [23], METEOR-S [167] et IRS-III [168].

❖ OWL-S

Le langage OWL-S (Ontology Web Language for Services) [21] est une ontologie pour les Web services basée sur OWL (Ontology Web Language) [28] dont la syntaxe est XML. C'est donc une ontologie pour la description sémantique des Web services dont l'objectif est de décrire de façon non ambiguë les Web services de façon à ce qu'un agent logiciel puisse exploiter automatiquement ces informations. Le but est de permettre l'automatisation de la découverte, de l'invocation et de la composition et l'interopérabilité des Web services et le contrôle d'exécution.

Une ontologie OWL-S de service est composée de parties (sous-ontologies) [21] (figure 2.9). Chacune d'elles est décrite par un fichier XML correspondant :

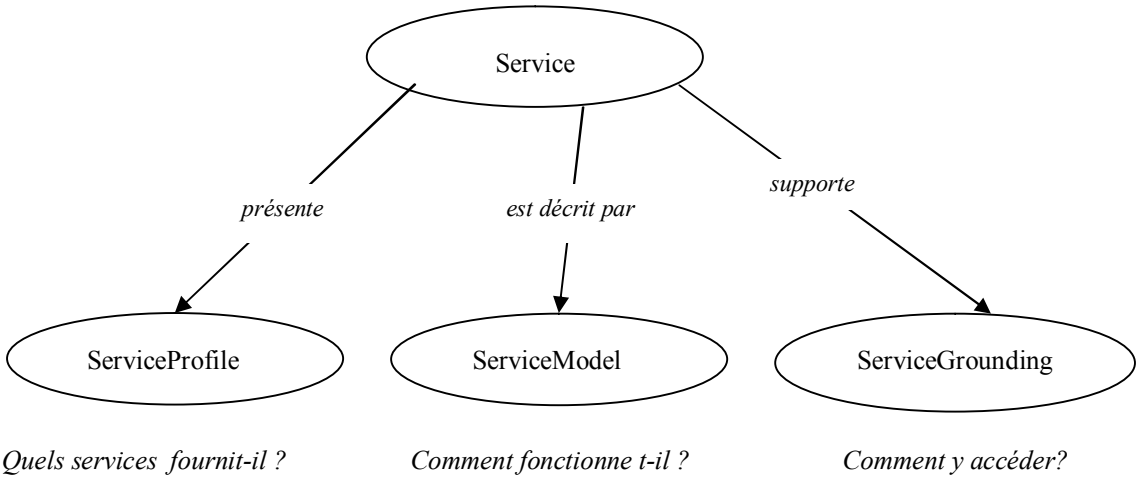

Figure 2.9 : Ontologie OWL-S [21].

- **Profil de service** : il décrit ce que fait le service. il est utilisé pour des objectifs de découverte et de publication des services. Il fournit des informations à l'utilisateur du service tels que le nom du service, la provenance du service, une description textuelle du service. Il spécifie les fonctionnalités offertes par le service qui sont représentées par les Inputs, les Outputs, les Pré-conditions et les Effets (IOPEs). Les Inputs et Outputs représentent les entrées demandées par un service et les sorties produits par ce service. Les pré-conditions et les effets spécifient respectivement les conditions préalables qui doivent être vérifiées avant l'exécution de ce service et les effets produits à la terminaison de ce service. En plus de ces informations fonctionnelles (IOPE), il décrit d'autres informations non fonctionnelles tel que le degré de qualité offert par le service ou la catégorie.

- **Modèle de service** : il décrit le fonctionnement interne du service sous forme de processus, en décrivant l'ensemble des opérations à effectuer. Un modèle de processus peut avoir un ou plusieurs processus simples (simple process), atomiques (atomic process) ou composites (composite process). Les processus atomiques sont directement invocables, ils correspondent à des opérations WSDL. Les processus composites sont décomposables en d'autres processus qui peuvent être à leur tour atomiques ou composites. Leur composition est spécifiée en utilisant les opérateurs de contrôle de séquences, de choix, de boucles, etc. Ces processus ne sont pas directement exécutables, ils permettent de spécifier l'ordre dans lequel les opérations concrètes du service peuvent être exécutées.

- **Grounding du service** : il définit comment accéder au service pour l'utiliser. Il spécifie les détails techniques d'accès au service comme par exemple, le protocole de communication, les formats de messages, la sérialisation, le transport et l'adressage. Il présente donc, la correspondance (mapping) entre la définition abstraite et la définition concrète des éléments nécessaires pour interagir avec le service. Le grounding du service définit dans OWL-S est basé sur WSDL. Un document WSDL est composé de deux parties. La partie abstraite correspond à la description des opérations offertes par le service. La partie concrète, quant à elle, décrit les informations de liaison qui permettent l'accès effectif au service. Le grounding décrit le mapping entre les concepts de OWL-S et ceux de WSDL. Ainsi, OWL-S utilise la partie concrète de WSDL pour les informations de liaison et effectue une correspondance des éléments qu'il définit avec la partie abstraite de WSDL (figure 2.10). Ainsi, chaque opération WSDL correspond à un processus atomique de OWL-S et les messages WSDL correspondent aux éléments Inputs et Outputs de OWL-S.

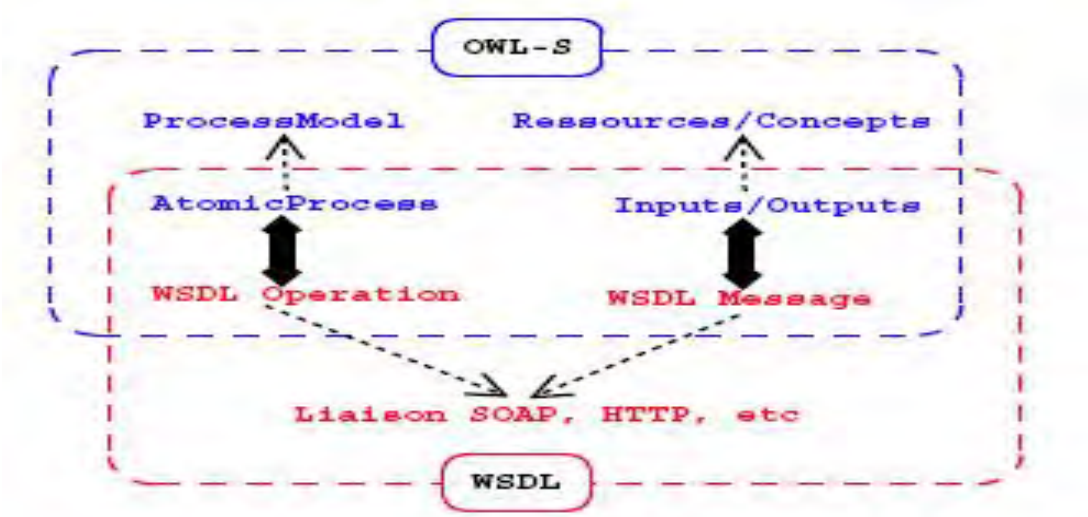

Figure 2.10 : Relation entre OWL-S et WSDL [21].

❖ **WSMO**

WSMO (Web Service Modeling Ontology) [23] est une ontologie pour la description de Web services sémantiques. Cette ontologie est basée sur l'infrastructure WSMF (Web Service Modeling Framework) qui offre un cadre générique pour la réalisation des services sémantiques. Il constitue donc un framework qui permet d'offrir beaucoup plus un cadre méthodologique qu'une infrastructure opérationnelle. L'ontologie WSMO permet la description des Web services en considérant les aspects suivants : les propriétés non-fonctionnelles, les médiateurs utilisés par le service, la fonctionnalité, les interfaces et les groundings des Web services.

L'architecture Web Service Modeling Ontology (WSMO), proposée par le laboratoire DERI, est une architecture conceptuelle ou un méta-modèle, visant à expliciter la sémantique des Web services. Elle est organisée autour de quatre éléments principaux **:**

- *les Web services* qui fournissent une fonctionnalité. Une description est associée à chaque service, dans le but de décrire sa fonctionnalité, son interface et ses détails internes ***;***

- *les Objectifs* qui servent à décrire les souhaits des utilisateurs, en termes de fonctionnalités requises. Un objectif décrit la fonctionnalité, les entrées/sorties, les pré-conditions et les post-conditions d'un Web service ;
- *les Médiateurs* qui sont utilisés pour résoudre de nombreux problèmes d'incompatibilité telles que les incompatibilités de données, de processus et de protocoles lors de l'établissement des communications;
- *les Ontologies* qui fournissent la terminologie de référence aux autres éléments de WSMO afin de spécifier le vocabulaire du domaine de connaissances d'une manière interprétable par les machines.

Contrairement à OWL-S, WSMO inclut les médiateurs comme des composants centraux de son architecture. Les hétérogénéités rencontrées lors de l'utilisation de Web services sont gérées par les types de médiation suivants :

– La médiation de données résout les incompatibilités de représentation des données ;
– La médiation de processus est relative à la logique applicative de la composition ;
– La médiation de protocoles adapte les différents protocoles de communication utilisés.

Ces types de médiation sont pris en charge par quatre familles de médiateurs :

- *Les GG-médiateurs* (GG signifiant « goal-goal ») permettent d'effectuer la médiation entre deux objectifs en se servant des ontologies d'objectifs disponibles dans le cadre de WSMO ;
- *Les WG-médiateurs* (**W**eb service-**G**oal) établissent les correspondances entre les fonctionnalités offertes par les Web services et les requêtes des utilisateurs. Le but est permettre la découverte et la sélection de Web.services ;
- *Les WW-médiateurs* (WW pour « **W**eb service-**W**eb service ») établissent les correspondances entre les Web services. Leur tâche est de résoudre les conflits au niveau des données, du protocole et du processus de composition. Ils sont mis en oeuvre lors de l'orchestration des Web services au sein d'une composition ;
- *Les OO-méediateurs* (OO pour « ontology-ontology ») qui sont destinés à résoudre les conflits entre ontologies. Le travail des OO-médiateurs consiste à établir des correspondances entre les terminologies contenues dans les différentes ontologies pour les intégrer en une représentation homogène des données. Cette représentation homogène permet de résoudre les hétérogénéités sémantiques et de répondre aux requêtes soumises par les composants de l'architecture WSMO.

Dans l'approche WSMO [23], on distingue deux composants principaux qui sont WSML (Web service Modeling Language) et WSMX (Web Service Execution Environment). WSML fournit un langage formel pour la description des éléments définis dans l'architecture WSMO. WSML est basé sur différents langages logiques qui sont la logique de description, la logique de premier ordre et la logique de programmation [16]. WSMX est un environnement d'exécution qui permet d'offrir un certain nombre de modules utilisables en run-time permettant de prendre en charge la découverte, la sélection, la médiation et l'invocation des services sémantiques.

❖ IRS-III

IRS-III (Internet Reasoning Service) [168] est un framework pour les Web services sémantiques. Le but principal d'IRS-III est de supporter la découverte et la récupération de

services de librairies distribuées sur Internet et leur configuration semi-automatique, dans le but de réaliser des tâches spécifiques en fonction des exigences des utilisateurs. IRS-III tente d'apporter l'adaptabilité et une médiation flexible entre problèmes et services [169]. Il permet aux applications de décrire sémantiquement et d'exécuter des Web services. Il est basé sur la structure UPML (Unified Problem Solving Method Development Language) [170] pour l'annotation des services en utilisant diverses ontologies:

- Ontologie du domaine (Domain model) : elle permet de décrire le domaine d'une application (ex : véhicules, maladies) ;
- Ontologie de tâche à résoudre (Task models) : elle fournit une description générique de la tâche à résoudre, spécifie les types d'entrées (input) et sortie (output), le but à atteindre et les pré-conditions à satisfaire ;
- Ontologie des méthodes de résolution d'un problème (Problem Solving Methods (PSMs)): elle sépare la description de ce qu'un service fait des paramètres et des contraintes d'une mise en oeuvre particulière ;
- Liens (bridges) : ils permettent la correspondance entre les différents modèles d'une application.

Les principaux composants de l'architecture IRS-III sont le serveur IRS-III (IRS-III server), l'éditeur de services (IRS-III Publisher) et la partie client (IRS-III Client). Ces trois composants interagissent entre eux via le protocole SOAP.

6.2.2 Techniques basées sur des langages d'annotation sémantique

Ces langages sont proposés pour étendre les normes existantes par des annotations, en utilisant les éléments d'extensibilité prévus à cet effet. Ces annotations enrichissent les langages existants afin de d'écrire la sémantique des Web services et sont attachées à concepts référencés grâce à une URL.unique. Parmi ces langages, nous citons principalement METEOR-S [167], WSDL-S [22] et SA-WSDL [166].

❖ METEOR-S

METEOR-S (METEOR for Semantic Web Services) [167], [171] a pour objectif de fournir des Web services sémantiques dans le cadre du projet METEOR (Managing End-To-End OpeRations). Il a précisément pour but d'intégrer les standards des Web services (BPEL4WS, WSDL et UDDI) avec les technologies du Web sémantique pour l'annotation, la découverte, la composition, la qualité de service et l'exécution de Web services. Le projet METEOR-S s'est développé selon trois phases principales qui ont permis d'introduire trois concepts importants pour :

- La mise en place de l'infrastructure : qui consiste à installer une infrastructure de découverte sémantique définie au dessus d'un registre UDDI et qui est appelée MWSDI (METEOR-S Web Service Discovery Infrastructure) ;
- L'annotation sémantique : qui permet d'enrichir sémantiquement les Web services en utilisant une extension enrichie de WSDL appelée WSDL-S (WSDL Semantics) via l'outil MWSAF (METEOR-S Web Service Annotation Framework). Le rôle de WSDL-S [22] est d'ajouter un niveau sémantique aux Web services à travers l'enrichissement des fichiers WSDL, mais également du registre enrichi UDDI ;

- La composition et l'exécution de services : qui a pour rôle d'assembler des services pour composer des services complexes en se basant sur BPEL4WS. L'outil se chargeant de cette tâche s'appelle MWSCF (METEOR-S Web Service Composition Framework).

❖ **WSDL-S**

WSDL-S [22] est une extension sémantique de WSDL (Web Service Description Language) [172]. Il fournit plusieurs extensions relatives aux opérations et messages d'entrées/sorties des Web services. Ces extensions contiennent des références aux concepts décrits dans les ontologies de description du domaine de connaissances associé au service Web, afin de spécifier la sémantique des messages, mais aussi les pré-conditions et les effets des opérations. Le service est alors annoté avec ses inputs, ses outputs, ses pré-conditions et ses effets. Ces annotations permettent la découverte automatique du service.

WSDL-S vise à supporter l'utilisation des concepts sémantiques analogues à ceux de OWL-S tout en étant agnostique à un langage de représentation sémantique Afin d'annoter les Web services, WSDL-S se sert des éléments d'extensibilité de WSDL. L'attribut d'extension <*wssem:modelReference*> supporte les connections entre un document de WSDL et une ontologie (par exemple, en OWL). Le deuxième attribut <*wssem:schemaMapping*> résoud les différences de schémas XML des attributs par l'application de feuilles de styles XSL (XML Transformation Style). Le troisième élément <*wssem:category*> correspond à la classification du service.

Les élements <*wssem:precondition*> et <*wssem:effect*> sont des sous éléments de <*wsdl:operation*> qui sont utilisés pour la découverte de service. WSDL-S représente donc, une solution pour combiner les documents WSDL avec des ontologies. Il est indépendant de n'importe quel langage particulier de représentation sémantique et offre ainsi, une flexibilité aux développeurs en choisissant leurs langages ontologiques préférés.

❖ **SAWSDL**

SAWSDL (Semantic Annotation for WSDL [166] est une extension de WSDL 2.0 [172] qui permet l'annotation de certains éléments d'une déclaration de service par des concepts sémantiques référencés grâce à une URL unique. Le World Wide Web Consortium a proposé SAWSDL comme un moyen d'annoter les descriptions WSDL 2.0 tout en supportant WSDL 1.1. SAWSDL est un ensemble d'attributs d'extension permettant de d´écrire la sémantique des éléments contenus dans les documents WSDL. L'objectif de SAWSDL est de définir comment une annotation doit être réalisée, tout en laissant le choix du langage utilisé pour la description sémantique. SAWSDL fournit des mécanismes permettant d'attacher des concepts décrits dans des ontologies aux annotations des descriptions WSDL. Cette proposition étend WSDL-S, elle peut être considérée comme une continuité de ce langage. Elle vise à apporter la valeur ajoutée de la sémantique non seulement lors de l'invocation des Web services, mais aussi durant la phase de découverte. Trois attributs d'extensibilité sont définis par défaut : l'attribut «modelReference » permet l'association entre un composant WSDL et un concept d'une ontologie, et les attributs « liftingSchemaMapping » et « lowering-SchemaMapping » sont ajoutés aux définitions de types pour spécifier les types de correspondances entre les éléments du schéma des données et l'information sémantique de l'ontologie.

6.2.3 Discussions

Différents auteurs [173], [174], [175], [176] [16] ont analysé les similarités et les différences entre ces langages de Web services sémantiques (WSSs) dont l'objectif de montrer les avantages et les limites de chacun d'eux. Par exemple, [173] s'est inspiré de leur infrastructure qui peuvent être caractérisées selon trois dimensions orthogonales : activités utilisées, architecture et ontologie de service Ces dimensions concernent les besoins pour les WSSs aux niveaux métiers, physiques et conceptuels. Par contre, [16] s'est focalisée principalement sur des critères spécifiques choisis pour l'analyse des langages sémantiques à différents niveaux : abstraction, ouverture, degré de maturité, capacité de description sémantique et la prise compte des mécanismes de la médiation sémantique.

Cependant, une synthèse de ces comparaisons faite par ces auteurs nous a permis de révéler les points suivants :

- WSMO est une ontologie basée sur des fondements conceptuels issus de WSMF pour la réalisation des WSSs. Il est beaucoup plus orienté utilisateur dans le sens où elle permet d'offrir un langage très proche des utilisateurs. Mais elle demeure toutefois immature du fait qu'il s'agit d'une initiative en cours de développement. De plus, l'une des limites les plus importantes est le fait que cette approche ignore les standards industriels existants, ce qui suppose que des mappings doivent être définis afin de garantir l'intégration avec les standards existants au sein des entreprises.

- IRS-III permet de réaliser les services sémantiques à travers le framework UPML. Ceci constitue à la fois son point fort et son point faible. De part sa formalisation UPML est considérée comme un point fort de l'approche IRS-III. Mais il constitue aussi l'un des griefs les plus cités à son encontre du fait qu'il est fortement lié à ce framework (UPML).

- METEOR-S, WSDL-S, et SASWDL constituent des langages basés sur l'extension de standards industriels. Le principe d'enrichissement sémantique est basé principalement sur l'utilisation des fichiers WSDL-S qui sont le résultat de l'annotation sémantique de fichiers WSDL. Bien que ces langages reposent sur des standards, il n'en demeure pas moins qu'elle manque d'abstraction dans le sens où ils se basent sut le niveau technique plutôt que sur le niveau conceptuel.

- OWL-S constitue une ontologie générique de services permettant principalement de décrire les Web services dans un cadre général. Sa caractéristique principale est qu'il est basé sur OWL, d'une part, et qu'il est compatible avec les standards industriels notamment WSDL et ce grâce au ServiceGrounding qui permet de définir des mappings entre OWL-S et WSDL. De plus, il fournit un mécanisme uniforme pour décrire la sémantique d'un Web service et permettant la découverte, l'invocation, la composition et l'intégration automatique de Web services.

En résumé, nous pouvons dire que OWL-S fournit une ontologie générique de services permettant de décrire les capacités et les propriétés des Web services. OWL-S est générique, basée sur le standards WSDL et OWL. WSMO est une initiative qui raffine WSMF, basée sur un langage propriétaire combinant plusieurs logiques. METEOR-S est une initiative basée sur l'extension de certains standards des Web services tels que WSDL et UDDI. IRS-III est fortement lié au framework UPML qui est basé sur les tâches de résolution de problèmes.

Quant à WSDL-S et à SASWDL, ce sont des langages d'annotation sémantique basés sur WSDL.

En tenant compte de certains critères que nous avons fixés dans notre travail et qui sont liés principalement aux trois aspects (conceptuel, ouverture et flexibilité), il nous paraît que le langage OWL-S, est celui qui présente le plus d'ouverture, de flexibilité et de généricité par rapport aux autres langages. En effet, l'analyse de OWL-S nous a permis de détecter ces trois aspects à différents niveaux :

- *Flexibilité :* OWL-S est flexible car il permet la création de plusieurs groundings ou binding pour un simple Web service sémantique [177].
- *Généricité :* OWL-S offre un framework générique pour décrire un Web service dans un cadre général. Dans ce sens, il fournit un niveau d'abstraction. Ce qui lui procure la caractéristique d'être conceptuel. En effet, il contient un ensemble de primitives de base pour spécifier n'importe quel Web service. Ces primitives peuvent être enrichies avec la connaissance d'un domaine spécifique.
- *Ouverture :* OWL-S est basé sur des standards industriels WSDL et OWL

Outre ces aspects, OWL-S présente les avantages suivants :

- *Support de Raisonnement :* OWL-S est un sous-ensemble de OWL qui fournit des mécanismes de raisonnement basé sur OWL DL (Description Logic), offrant ainsi beaucoup d'avantages (maximum d'expressivité, complétude des calculs, etc.).
- *Modularité :* une autre caractéristique de OWL-S est la décomposition de sa description en plusieurs concepts. Les différents aspects de cette description sont décomposés en trois sous-ontologies: *ServiceProfile*, *ServiceModel* et *ServiceGrounding*. Les avantages de cette modularité sont nombreux [178]. Puisque la description est fractionnée en plusieurs instances, alors il est possible de réutiliser certaines parties.

❖ **Apports de Web services sémantiques dans l'interopérabilité des Workflows**

Le concept des Web services sémantiques (WSSs) apporte une nouvelle manière pour l'interopérabilité des processus métiers pour les entreprises se tournant vers l'Internet en offrant une intégration sémantique, ouverte et flexible. Pour ce faire, les Workflows d'entreprise (ou busines process) peuvent être fragmentés en plusieurs entités Workflows (activités, tâches) réutilisables appelées « services métiers » de sorte que chacun réalise une fonction distincte. Ces services seront d'abord, dotés de descriptions sémantiques (appelés services sémantiques) puis exposés en tant que des Web services sémantiques permettant ainsi, la coopération des entreprises à travers le Web.

Ces Web services sémantiques (WSSs) permettent donc à une entreprise '' d'exporter '' son processus métier sur Internet afin d'être découverts sémantiquement, invoqués puis exécutés par des partenaires collaboratifs. L'interopérabilité des business process est donc favorisée via l'interopérabilité des WSSs. Quant aux bénéfices apportés par les WSSs, ils sont résumés comme suit :

- Automatisation de la découverte sémantique des Web services, c'est-à-dire, localisation automatique des Web services qui fournissent une fonctionnalité particulière et qui répondent aux propriétés demandées par l'utilisateur ;

- Automatisation de l'invocation : cela implique l'automatisation de l'exécution du service Web par le programme d'utilisateur ou par un agent ;
- Composition automatique des Web services ;
- Favorisant l'interopérabilité sémantique des Web services.

Ainsi, l'ajout de cette technique sémantique basée OWL-S aux deux dernières techniques syntaxiques (MDA et SOA), nous permet de mieux appréhender l'aspect sémantique. MDA permet de capturer l'aspect conceptuel et SOA pour les aspects de flexibilité et d'ouverture. Ce qui permet d'enrichir l'architecture SOA est une architecture dite "SOA sémantique ".

Enfin, nous pouvons dire que ces trois approches (MDA, ontologies et SOA) prises conjointement peuvent favoriser considérablement l'interopérabilité des modèles de Workflow, mais dans un contexte orienté industriel.

7. Bilan des différentes techniques d'interopérabilité

En résumé, nous pouvons dire que l'analyse faite sur l'état de l'art concernant les différentes techniques d'interopérabilité, nous a apporté les ingrédients nécessaires pour :

- Capturer l'aspect conceptuel d'une solution d'interopérabilité des modèles de Workflow grâce aux concepts de MDA qui nous offrent un niveau d'abstraction élevé, en utilisant les techniques de méta-modélisation et de transformation de modèles.

- Prendre en charge l'aspect sémantique grâce au concept d'ontologies, en donnant une signification aux différents éléments d'un modèle de processus qui peuvent être des activités, des données ou des fonctions (appelées parfois opérations).

- Répondre aux besoins d'une coopération flexible et ouverte entre des partenaires hétérogènes grâce au concept de SOA, et en particulier au paradigme des Web services qui nous offrent un cadre architectural bien adapté pour le partage des services (métiers).

8. Conclusion

Dans ce chapitre, nous avons étudié les différentes formes, analysé les principales approches qui existent pour d'interopérabilité des Workflows inter-entreprises ainsi que les différents techniques qui peuvent être mises en oeuvre dans le cadre de l'intégration et de l'interopérabilité des systèmes d'information en général, et en particulier dans le cadre des modèles de Workflow.

Cependant, l'étude de ces approches et de ces techniques vis-à-vis de la problématique conceptuelle posée et des objectifs fixés, nous a permis de recenser les points suivants :

- Le premier point concerne la forme d'interopérabilité qui peut être adaptée pour assurer une certaine flexibilité entre les partenaires coopératifs et qui est pour nous, la sous-traitance.

- Le deuxième point porte sur les différentes approches proposées et dont le bilan est que la plupart de ces approches ne préservent ni l'autonomie des Workflows des partenaires, ni leurs systèmes de gestion de Wokflow existants.

- Le troisième point concerne les différentes techniques qui sont utilisées pour la mise en œuvre de l'interopérabilité. Concernant ce dernier point, nous avons, en particulier, retenu trois approches : l'approche basée sur l'ingénierie des modèles (MDA) et celle qui est orientée SOA. Ces deux approches permettent en effet, de prendre en compte l'aspect conceptuel, ainsi que les critères de flexibilité et d'ouverture. Quant à l'approche basée Web services sémantiques qui intègre le concept d'ontologie, permet ainsi, de mieux appréhender l'aspect sémantique des modèles de processus.

De même que l'analyse de la littérature sur l'interopérabilité des entreprises, issue des travaux du réseau d'excellence européen INTEROP [121], du projet ATHENA [122] et de plusieurs travaux [3], [16], [17], montre qu'il existe principalement deux approches pour traiter la problématique de l'interopérabilité :

- Une approche de standardisation : Dans cette approche, les entreprises se mettent d'accord pour standardiser la manière de présenter leurs données, processus et applications, mais aussi leurs outils, méthodes et stratégies dans un contexte standard orienté industriel. Ce qui leur permet de bénéficier des nouvelles technologies de l'information, et en particulier de la technologie Web, ainsi que les standards qui lui associés (langages, plates-formes, protocoles de communication, etc.) pour faciliter la mise en ouvre de l'interopérabilité de leurs systèmes d'information, applications ou modèles de processus. Comme exemple, ils se mettent sur un modèle commun standard pour l'échange de modèles de processus basé selon les concepts du MDA [72], [129], [131].

- Une approche d'unification : Dans cette approche, les entreprises se mettent d'accord pour unifier la manière de présenter leurs données, processus et applications, mais aussi leurs outils, méthodes et stratégies pour faciliter l'interopérabilité de leurs systèmes d'information, applications ou modèles de processus. Comme exemple, ils se mettent d'accord sur un langage commun tel que UEML (Unified Enterprise Modeling Language) [3] [18] qui est un langage unifié de modélisation d'entreprise. Il offre en effet un moyen d'entente aux partenaires, et représente alors un modèle commun pour l'échange des modèles différents d'entreprises.

Partant de ce constat, nous allons proposer dans la deuxième partie du document deux approches qui, à notre sens, permettent de résoudre la problématique conceptuelle de l'interopérabilité des modèles de Workflow (ou modèles de processus). Cette partie comporte deux chapitres. Dans le chapitre III, nous présentons la première approche qui préconise l'utilisation des techniques de méta-modélisation et de transformation de modèles. Quant au chapitre VI, il est consacré à la deuxième approche qui porte sur les techniques d'annotation sémantique de modèles.

PARTIE 2

APPROCHES CONCEPTUELLES D'INTEROPERABILITE DES MODELES DE WORKFLOW

Chapitre III

Utilisation des Techniques de Méta-modélisation et de Transformation de Modèles

Paraphrase d'Herbert A. Simon

« Modéliser, c'est raisonner »,

Notre première conviction dans cette thèse est :

« Pour interopérer, modéliser, puis formaliser pour raisonner»

1. Introduction

L'étude des travaux portant sur l'interopérabilité des systèmes d'information nous a permis de comprendre principalement trois niveaux d'interopérabilité : syntaxique, technique et sémantique. Le dernier niveau constitue un problème crucial pour les entreprises et auquel nous nous intéressons particulièrement dans notre domaine de Workflow. A notre connaissance, ce problème n'est toujours pas correctement traité au niveau conceptuel et notre travail s'inscrit dans ce contexte.

De même que l'analyse de la littérature sur l'interopérabilité des entreprises, issue des travaux de différents travaux d'interopérabilité [3], [14], [15], [16], [17], [18] montre qu'il existe principalement deux approches pour traiter la problématique de l'interopérabilité : *i) une approche de standardisation* orientée dans un contexte industriel afin de bénéficier en particulier, de la technologie Web, ainsi que les standards qui lui sont associés (langages, protocoles de communication, etc.) pour faciliter la mise en ouvre de l'interopérabilité, *ii) Une approche d'unification* orientée dans un cadre unifié et dans un contexte ni standard, ni industriel.

Partant de ce constat, nous allons proposer dans ce chapitre, notre première approche, qui à notre sens, permet de résoudre la problématique conceptuelle de l'interopérabilité des modèles de Workflow (ou modèles de processus). Pour aborder cette approche nous sommes partis d'une motivation inspirée de la conviction suivante :

« Pour interopérer, c'est d'abord modéliser, puis formaliser pour raisonner »

Plusieurs formes d'interopérabilité ont été étudiées dans le chapitre précédent. Parmi ces formes, nous présentons d'abord, une forme qui est flexible, ne nécessitant pas un couplage fort entre les partenaires collaboratifs. Puis, nous introduisons les scénarios d'interopérabilité qui peuvent exister.

2. Forme d'interopérabilité adoptée

L'interopérabilité entre deux entreprises suppose donc, que deux acteurs peuvent échanger leurs modèles (ou fragments de modèles) et coopérer. La figure 3.1 présente un modèle *M1* envoyé par l'acteur *Act1* vers l'acteur coopératif *Act2*.

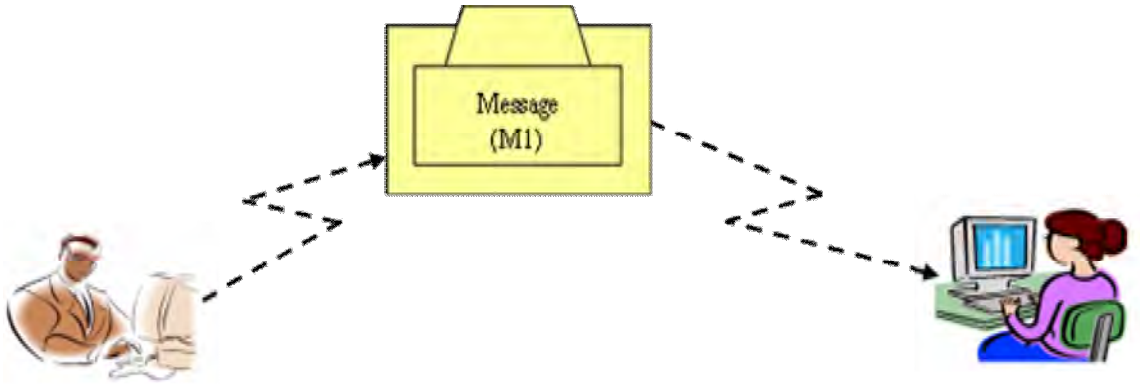

Figure 3.1: Echanges de modèles de processus.

Parmi les formes d'interopérabilité des Workflows d'entreprise que nous avons présentées précédemment, nous nous sommes intéressés particulièrement à une forme particulière de coopération dite « la sous-traitance » qui est considérée comme un mécanisme souple et flexible. Cette flexibilité est généralement liée à la manière d'appeler un service métier (activité ou tâche) qui est publié sur Internet et cela indépendamment de sa logique d'implémentation.

Cette forme d'interopérabilité permet ainsi, à un partenaire principal de déléguer l'exécution d'une partie de son processus métier à d'autres partenaires, par exemple, l'envoi d'un fragment de modèle telle qu'une activité de commande d'achat à des partenaires (clients, fournisseurs, etc.) pour sa sous-traitance. Ces derniers coopèrent en exécutant l'activité via leurs processus métiers de livraison de commande. Dans la suite du document, nous nous servons de cet exemple comme une étude de cas pour illustrer notre approche conceptuelle d'interopérabilité.

3. Scénarios d'interopérabilité

L'interopérabilité des modèles suppose donc, qu'il y a au moins deux acteurs qui peuvent échanger et travailler sur un modèle ou un fragment de modèle. Cependant, pour échanger des modèles de processus et coopérer, deux approches existent. L'approche par conversion dite intuitive (figure 3.2) qui nécessite N*(N-1) convertisseurs (N étant le nombre de modèles à convertir) et l'approche par modèle commun dit "modèle canonique", qui nécessite seulement 2*N convertisseurs (figure 3.3) et qui est par conséquent, la plus avantageuse.

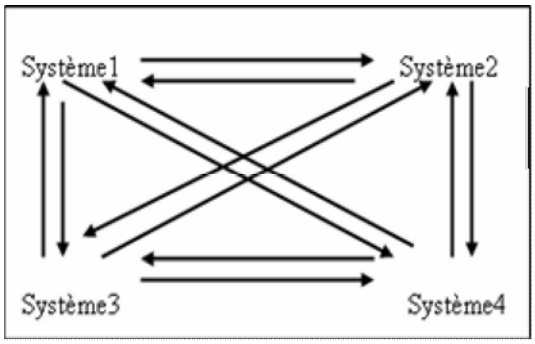

Figure 3.2 : Approche par conversion.

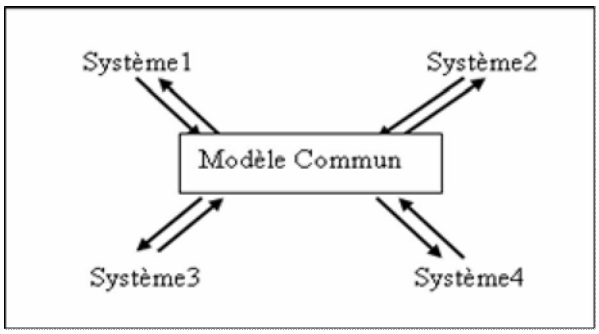

Figure 3.3 : Approche par modèle commun.

4. Approche conceptuelle d'interopérabilité

Avec l'émergence de la technologie des Web services sémantiques, l'interopérabilité sémantique des modèles de Workflow (ou business process) peut être mise en oeuvre via leurs services métiers (ou business services) que nous considérons comme des Web services sémantiques. Cependant, l'utilisation des langages de description sémantique tels que OWL-S [21], WSDL-S [22], SAWSDL [166] nécessitent beaucoup d'efforts d'apprentissage et de compréhension pour les utilisateurs. Par conséquent, pour familiariser l'utilisateur habitué aux outils UML et à utiliser la technologie des Web services sémantiques, nous préconisons une approche basée MDA pour faciliter son adoption. Dans nos travaux précédents [131], [132], nous avons utilisé l'approche MDA pour intégrer le contexte du Web sémantique afin de générer une ontologie Workflow en une description OWL pour favoriser l'interopérabilité des modèles de Workflow. Tandis que notre travail actuel se focalise à créer une passerelle entre MDA et les deux contextes : le Web sémantique et le Web service sémantique.

Notre objectif est donc, de proposer une approche conceptuelle permettant de décrire la sémantique des Web services de manière indépendante de tout langage de description sémantique tels que OWL [28], OWL-S [21] ou WSDL-S [22]. Cette approche est basée sur un processus MDA pour la génération semi-automatique d'une ontologie d'un service métier afin de favoriser l'interopérabilité sémantique des modèles de Workflow.

4.1 Principe de l'approche

Pour adresser le problème de la description sémantique des services métiers, et capturer l'aspect conceptuel de notre solution, nous nous basons sur les différents méta-modèles qui sont basés sur OWL, OWL-S ou WSDL-S pour générer de manière semi-automatique une ontologie de services qui sera décrite dans un langage ontologique correspondant à l'un des méta-modèles cités ci-dessus. Cependant, l'analyse de ces différents méta-modèles faite dans le chapitre précédent nous a incité à se focaliser sur le méta-modèle OWL-S en tant que PSM (Platform Specific Model) cible car il présente beaucoup d'avantages tels que flexibilité, modularité, composition, invocation, découverte dynamique.

Cette approche préconise l'utilisation des techniques de méta-modélisation et de transformation de modèles. Elle propose alors de combiner à la fois l'approche MDA, les ontologies et les Web services. Elle nous permet de modéliser les ontologies pour l'élaboration de différents méta-modèles correspondant aux langages de représentation d'ontologies tels que OWL, OWL-S, WSDL-S, et de construire le méta-modèle de la plate-forme technique choisie par les partenaires coopératifs en tant que PSM cible.

En effet, l'association conjointe de ces trois approches nous permet effectivement *i*) de nous offrir un haut niveau d'abstraction, et cela grâce aux concepts de MDA *ii)* de prendre en compte l'aspect sémantique des différents éléments des processus métiers (activités, tâches, etc.) par l'intégration du concept d'ontologies *iii)* de nous offrir un cadre architectural partag*é* et flexible par l'introduction du modèle SOA.

Pour pouvoir traiter cette problématique conceptuelle, et répondre aux objectifs fixés dans la thèse, cette approche préconiser d'utiliser :

- D'abord, la démarche MDA qui nous fournit un cadre conceptuel général en se basant sur les techniques de méta-modélisation et de transformations de modèles pour en arriver à une solution industrielle implémentée sur une plate-forme technique de notre choix à partir des modèles métiers qui sont indépendants de toute plate-forme et de tout langage ontologique d'implémentation. Le choix de MDA pour une solution conceptuelle est motivé par le fait qu'elle nous permet :

  - La modélisation des ontologies et cela, en définissant les différents méta-modèles de définition d'ontologies correspondant aux langages de représentation d'ontologies basé Web. Le but est donc de décrire la sémantique des Web services de manière indépendante à ces langages.

  - La modélisation de la plate-forme d'interopérabilité qui est pour nous, le modèle SOA, et en particulier la plate-forme des Web services pour permettre à des acteurs coopératifs de communiquer et de coopérer.

Dans notre contexte, nous utilisons le paradigme des Web services sémantiques pour doter les entités Workflows (activités, tâches ou sous-processus) d'une sémantique, et en particulier le méta-modèle OWL-S qui nous est d'un grand apport. En effet, il permet de mieux de décrire les Web services dans un cadre général.

Aussi, nous utilisons le modèle SOA qui se base sur le concept des Web services. Le choix des Web services est principalement motivé par le fait qu'ils constituent une technologie flexible basée sur des standards industriels, tandis que le choix des ontologies peut être justifié par le fait qu'elles constituent la technologie la plus prometteuse pour la prise en compte de l'aspect sémantique des Web services.

4.2 Workflow versus Web services sémantiques

Dans un contexte d'échanges B2B, l'Internet constitue une avancée majeure pour les entreprises coopératives se tournant vers le Web. Dans ce contexte, les processus Workflows, appelés parfois processus d'affaires, peuvent être organisés et structurés de façon à les 'externaliser', c'est-à-dire, les publier en tant que des Web services sémantiques permettant à des partenaires coopératifs de les intégrer éventuellement dans leurs propres progiciels CRM (Customer Relation Mangement ou gestion de relation client). Ces Web services sémantiques (WSSs), seront alors découverts puis invoqués pour exécution. Ce qui permet de favoriser considérablement l'interopérabilité des processus Workflows via les WSSs.

Puisque notre choix s'est porté sur le modèle SOA, et en particulier sur la plate-forme des Web services, alors nous utilisons le paradigme des Web services et nous considérons un processus Workflow comme un ensemble de services Workflow (ou 'service' tout court) dans la mesure ou cette architecture se base sur le concept de service.

Ces processus Workflows seront alors modélisés, implémentés puis publiés en tant que des Web services sémantiques (WSSs) et l'interopérabilité est donc, réalisée via l'interopérabilité de ces WSSs. Ce qui nous incite à procéder un glissement des concepts du Workflow de la WfMC (Workflow Mangement Coalition) [4] vers ceux des WSSs du W3C (World Wide Web Consortium). Puisque le méta-modèle OWL-S a été retenu comme PSM, alors notre processus Workflow sera translaté vers une ontologie OWL-S de services.

Cependant, l'étude détaillée du méta-modèle XPDL [4] et du langage d'ontologie de services OWL-S [21] nous ont permis d'établir les correspondances entre leurs concepts comme le montre le tableau suivant.

Concepts de Workflow (XPDL de la WfMC)	**Concepts des WSSs (OWL-S de W3C)**
WorkflowProcessDefinition	*OWL-S Service* *ServiceModel (ou ServiceProcess)*
WorkflowProcessActivity	▪ *OWL-S AtomicProcess* : si l'activité est automatisée et atomique ; ▪ *OWL-S CompositeProcess,* : si l'activité est automatisée et composite (ou sous-processus) ▪ *Opération WSDL de notification* : si l'activité est manuelle.
WorkflowParticipantSpecification	----------
WorkflowApplicationDeclaration	*OWL-S Resource*
WorkflowRelevantData	*Inputs et Outputs*

Table 3.1 : Correspondances entre les concepts de XPDL et d'OWL-S.

Selon cette table de correspondance, le modèle de Workflow est alors traduit en un WSS qui sera publié sur Internet pour une éventuelle découverte sémantique, puis invocation par les partenaires coopératifs. Dans la littérature, plusieurs algorithmes existent [231], [232], [233], [234], [235] et permettent la découverte dynamique des WSSs. On entend par découverte dynamique la possibilité de localiser automatiquement un Web service qui répond à des besoins particuliers.

Notre approche étant conceptuelle, nous allons dans la section suivante, monter comment utiliser MDA comme support pour *i)* modéliser les langages ontologiques afin de décrire les aspects sémantiques des services Workflows (considéré comme des Web services) *ii)* modéliser la plate-forme des Web services comme intergiciel partagé pour la coopération.

4.3 Modélisation des ontologies

Pour capturer l'aspect sémantique de manière conceptuelle indépendamment des langages de représentation d'ontologies basés Web, et surtout faciliter leur manipulation via leur utilisation par des outils UML, nous avons utilisé les techniques de méta-modélisation pour modéliser ces langages en élaborant leurs méta-modèles. Ces derniers seront alors intégrés dans une architecture MDA basée ontologies pour nous permettre de générer une description sémantique d'un service Workflow dans n'importe quel langage ontologique. C'est l'objet de la section suivante.

4.3.1 Architecture de modélisation des ontologies

L'architecture générale que nous proposons doit donc, intégrer des concepts ontologiques pour pouvoir supporter la modélisation des Web services sémantiques. Sa construction utilise à la fois, l'approche MDA, le Web sémantique et les Web services sémantiques. En fait, l'intégration de l'approche MDA a été utilisée par beaucoup d'auteurs durant cette décennie [195], [196], [197], [198].

Au fait, l'architecture que nous proposons dans cette thèse permet d'intégrer simultanément les deux contextes : le Web sémantique et le Web service sémantique. C'est une architecture MDA basée sur des ontologies pour la modélisation des Web services sémantiques qui comporte (figure 3.4) :

- Un Profil UML pour Web services sémantiques (UPSWS) ;
- Un Profil UML d'ontologie pour Web services sémantiques (OUPSWS) ;
- Un méta-modèle de définition d'ontologie (ODM).
- Différents langages ontologiques pour le Web sémantique (RDFs, OWL, etc.).
- Et différents langages ontologiques pour les services (WSML, WSDL-S, OWL-S, etc.).

Le noyau de cette architecture est par conséquent, le méta-modèle de définition d'ontologie (ODM) qui représente une passerelle entre MDA et les deux espaces technologiques : le Web sémantique et les Web services sémantiques.

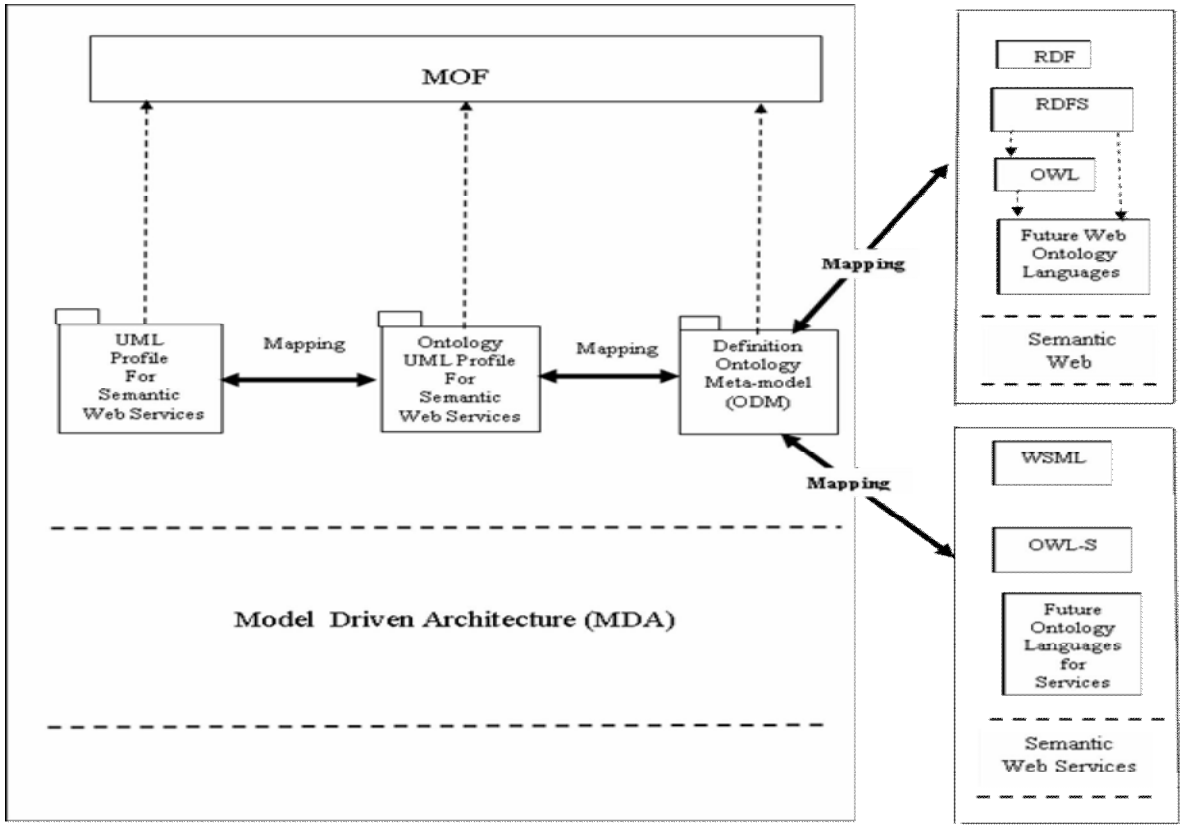

Figure 3.4 : Architecture MDA-Ontologies pour la modélisation d'un Web service sémantique.

Selon cette architecture, deux types de mappings sont décrits pour deux espaces technologiques différents : le Web sémantique et le Web service sémantique. Le premier mapping concerne la traduction du méta-modèle de définition d'ontologie (ODM) vers le méta-modèle OWL (si le méta-modèle OWL est choisi comme plate-forme cible dans le Web sémantique) et le deuxième mapping concerne le méta-modèle OWL-S (si le méta-modèle OWL-S est choisi comme plate-forme cible dans le Web service sémantique).

Notre architecture est donc, une architecture ontologique basée MDA qui inclue un Profil UML pour Web services sémantiques (UPSWS). Pour construire ce Profil, nous choisissons d'abord, le package qui contient l'ensemble de stéréotypes qui nous permet de projeter ce Profil vers le méta-modèle cible (PSM) qui peut être OWL, OWL-S ou WSDL-S. Ensuite, nous exploitons les ontologies pour modéliser les aspects sémantiques des Web services pour définir un Profil UML d'ontologie pour les Web services sémantiques qui sera traduit vers un méta-

modèle de définition d'ontologie (ODM) [181]. Ce méta-modèle regroupe tous les concepts ontologiques. Il est défini selon le MOF (Meta-Object Facility) et sa construction est basée sur le langage OWL [182].

Une fois ODM construit, il est traduit vers le méta-modèle OWL, par exemple pour générer une description OWL ou vers le méta-modèle OWL-S pour la génération d'une description OWL-S. Par conséquent, la passerelle entre MDA, le Web sémantique et le Web sémantique est créée via l'utilisation de ODM (figure 3.4) qui offre plusieurs avantages tels que flexibilité, ouverture [132]. En nous focalisant uniquement sur le méta-modèle OWL-S, nous allons proposer dans la section suivante, une approche basée MDA pour construire une ontologie de services.

4.3.2 Approche MDA pour la construction d'une ontologie de services

Pour faciliter l'adoption des Web services sémantiques dans un contexte MDA, beaucoup d'auteurs [183], [184], [185] [209] ont utilisé l'approche orientée modèle (MDA) pour la génération des descriptions OWL-S. Elenius et autres [186] ont développé des plugs-in pour OWL-S dans Protégé : l'éditeur de OWL-S. Cet éditeur est développé pour l'environnement Protégé qui permet à des utilisateurs de développer des instances de l'ontologie OWL-S de services de manière graphique. Il fournit une interface utilisateur pour créer et modifier une description OWL-S comprenant les trois parties : le profil de service, le modèle de service, et le grounding de service. L'éditeur d'OWL-S supporte la composition par l'utilisation d'un éditeur visuel. Il est centré sur le langage OWL-S et par conséquent, toutes les compositions sont établies grâce à l'utilisation des constructions d'OWL-S.

Dans [187], les auteurs ont créé un outil de mapping pour une réalisation concrète d'un Web service sémantique en évitant ainsi, l'utilisation d'environnements tel que Protégé. Par contre, dans [188], les auteurs ont développé une approche pour produire automatiquement des spécifications d'OWL-S pour un fichier WSDL donné. Leur approche génère des grounding complets, des profils incomplets et des modèles de processus. L'approche est basée sur la traduction de l'opération WSDL vers un processus atomique d'OWL-S et par la traduction des types XSD vers des concepts d'OWL-S. Leur approche originale a été développée pour le format de DAML-S (un précurseur d'OWL-S). Jaeger et d'autres auteurs [194] ont aussi développé un processus comportant trois étapes pour générer des spécifications d'OWL-S.

Notre approche s'inspire de leurs approches, mais elle est générale dans le sens où elle permet de prendre en compte tous les langages de représentation des ontologies tels que RDFs [16], OWL [28] ou les langages d'ontologies de services tels que WSDL-S [22], WSML [23] ou OWL-S [21], dans le sens où elle utilise un ensemble de stéréotypes spécifiques à chaque langage. Cet ensemble est regroupé sous forme de packages que l'utilisateur peut utiliser pour modéliser son service Workflow (ou métier) dans n'importe quel méta-modèle.

Aussi, elle diffère dans le processus de génération. D'abord, nous élaborons trois modèles séparés correspondant respectivement aux concepts structurels, dynamiques et techniques (grounding) extraits à partir d'une spécification OWL-S. Pour la génération semi-automatique, nous utilisons les mappings définis dans la section 4.3.4. Quant à la validation de l'ontologie OWL-S de service, plusieurs outils de validation existent dans la littérature [189], [190] [191], mais dans notre travail, nous utilisons simplement l'outil ontologique Protégé [191].

La figure 3.5 montre les différentes étapes de construction d'une ontologie de services à partir d'un service Workflow considéré comme un Web service.

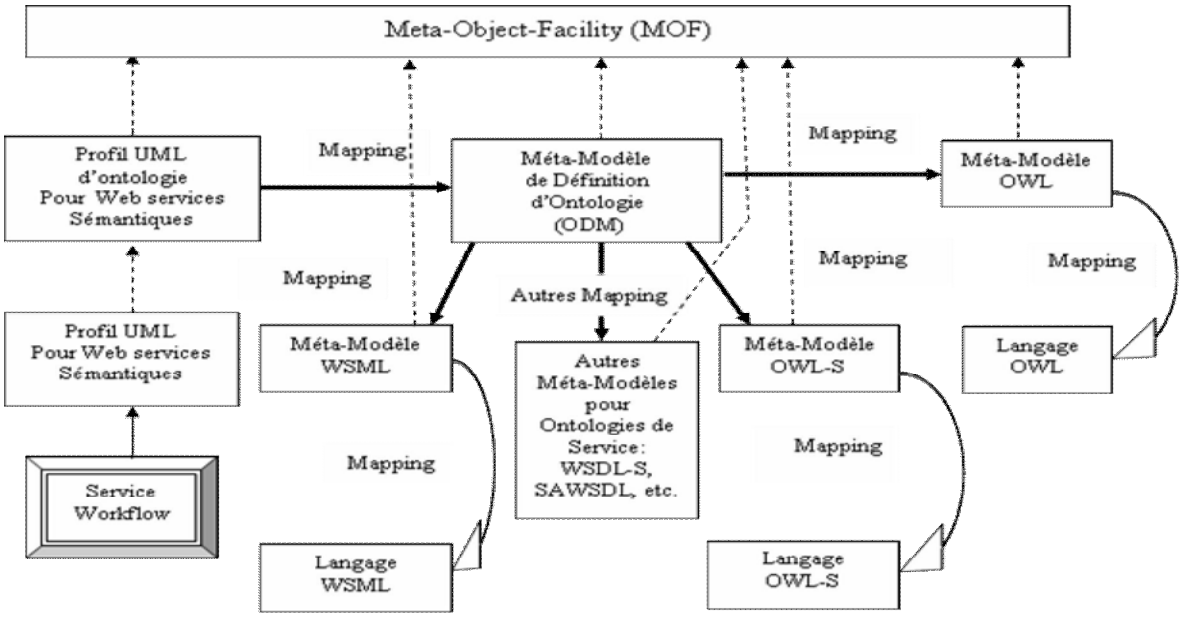

Figure 3.5 : Etapes de construction d'une ontologie de services.

Cette architecture repose principalement sur la construction préalable de trois Profils UML pour Web Services Sémantiques (UPSWS), c'est-à-dire, pour OWL-S concernant la partie "profil de service" (*ServiceProfil*), la partie "processus de service" (*ServiceModel ou ServiceProcess)* et la partie "Grounding du service" *(ServiceGrounding*). Ces trois Profils UML sont présentés comme suit :

- **Profil UML pour Web Services Sémantiques (UPSWS).**

Ce Profil UML est utilisé pour modéliser les aspects sémantiques d'un service Workflow (ou 'service' tout court). Un groupe d'extensions d'UML constitue un Profil UML. UML offre trois mécanismes d'extension pour étendre les éléments du méta-modèle de base d'UML :

- les stéréotypes qui introduisent de nouveaux éléments en étendant l'élément de base,
- les valeurs étiquetées (ou tagged values) qui introduisent de nouvelles propriétés,
- les contraintes qui sont des restrictions sur les nouveaux éléments en respectant les éléments de base.

Nous utilisons un Profil UML pour modéliser les différents constructeurs OWL-S en conjonction avec le diagramme de classe UML. En terme de MDA, les stéréotypes, les valeurs étiquetées (ou tagged values) et les types de données servent à marquer le modèle indépendant de la plate-forme, appelé PIM (Platform Independent Model) pour faciliter sa transformation vers le méta-modèle cible PSM (Platform Specific Model), qui dans notre cas le méta-modèle OWL-S.

Ce PSM nous permet par la suite de générer une ontologie de services dans le langage ontologique OWL-S. La figure suivante montre un exemple de Profil UML pour OWL-S (UPSWS) concernant la partie "Profil de service" *(ServiceProfil*).

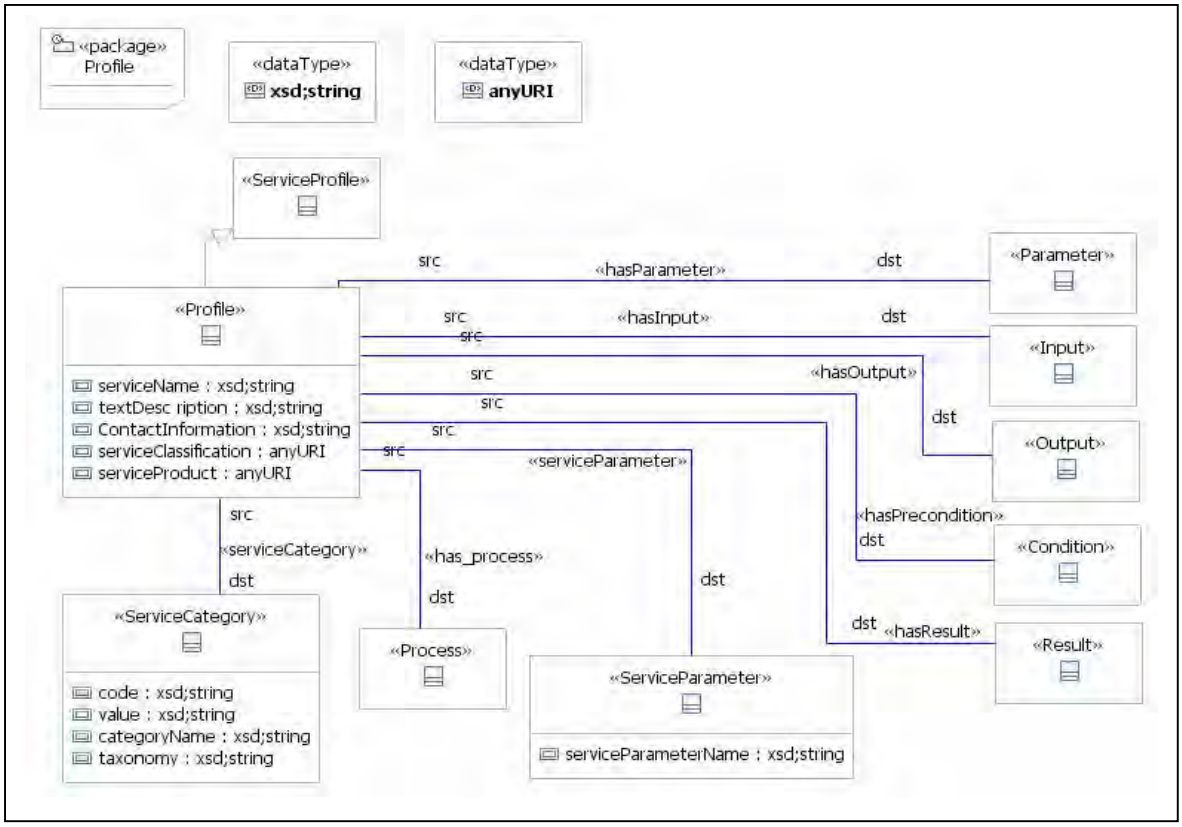

Figure 3.6 : UPSWS (*Partie "ServiceProfil"*).

La figure 3.7 suivante montre un exemple de Profil UML pour OWL-S (UPSWS) concernant le modèle de service *(ServiceModel* ou *ServiceProcess*).

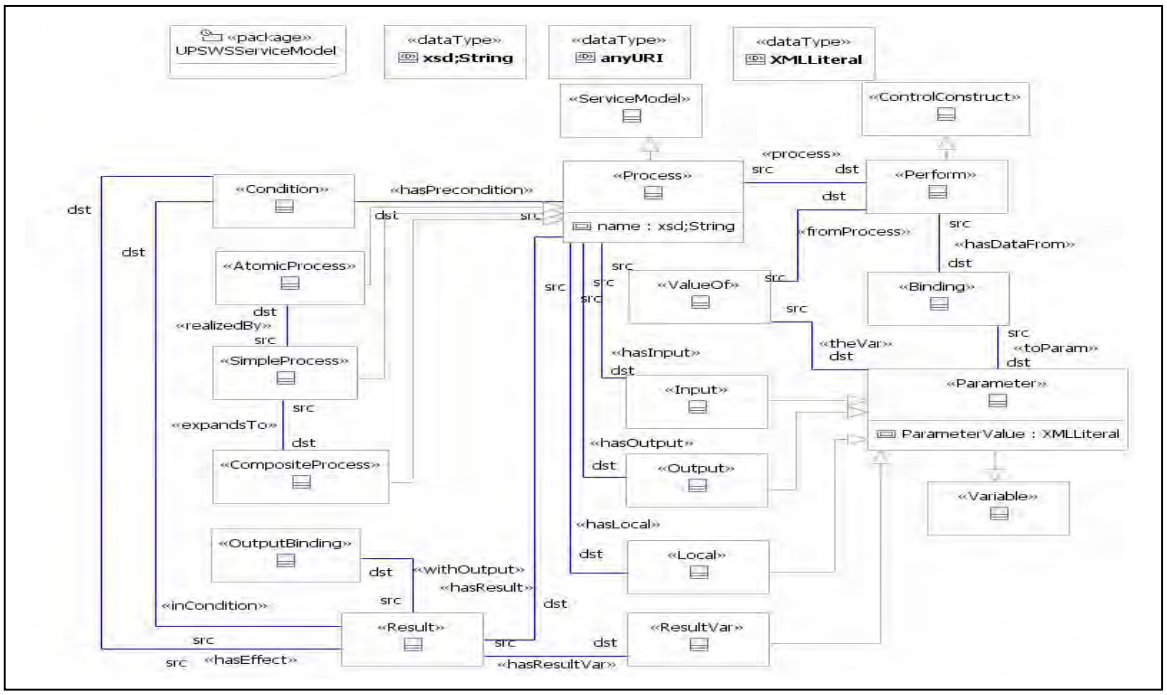

Figure 3.7: UPSWS (*Partie "ServiceProcess"*).

La figure suivante montre un exemple de Profil UML pour OWL-S (UPSWS) concernant le service Grounding *(ServiceGrounding)*.

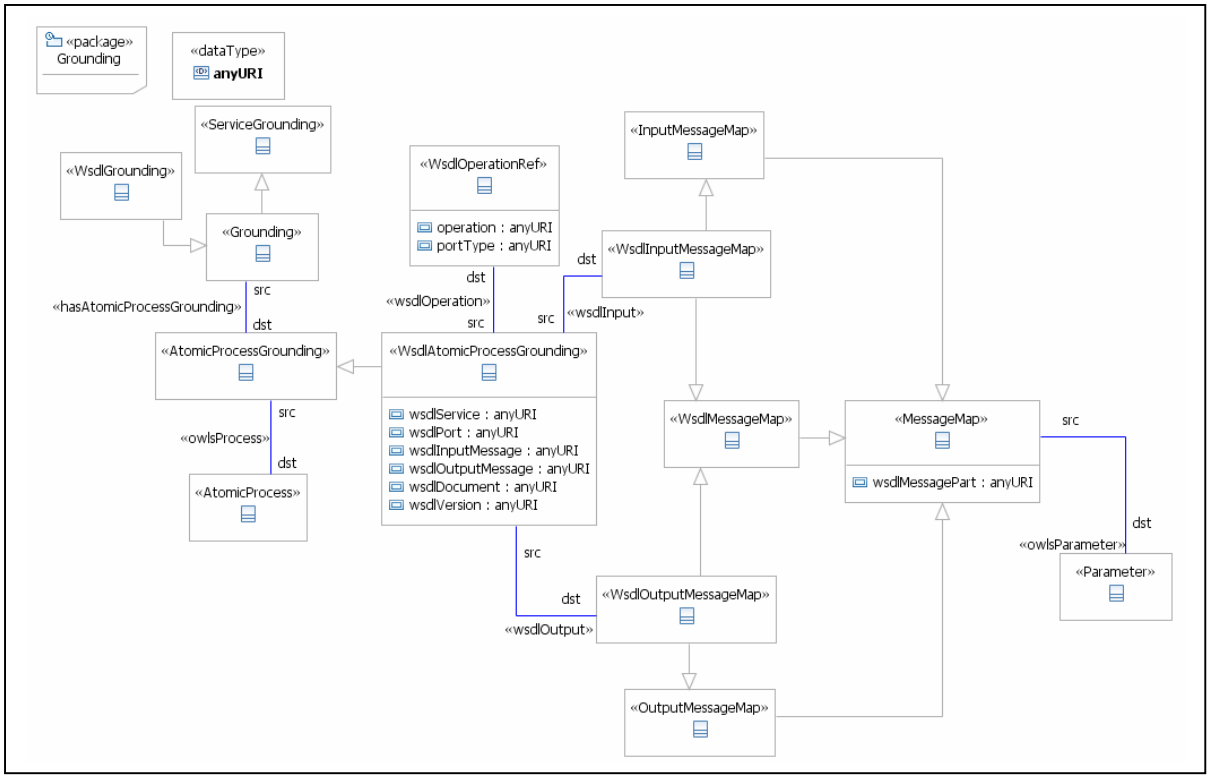

Figure 3.8 : UPSWS (*Partie "ServiceGrounding"*).

Cependant, pour générer une ontologie OWL-S de service, nous avons adopté une méthodologie basée MDA comportant plusieurs étapes que nous décrivons dans la section suivante.

4.3.3 Méthodologie de génération d'une ontologie de services

Les ontologies de services sont créées pour réaliser des Web services sémantiques. Aussi, il est souhaitable de les créer de manière automatique. Bien que certains outils comme Protégé [191] fournissent des environnements d'édition d'ontologies, mais l'utilisation technique des langages d'ontologies de services tels que OWL-S [21], WSDL-S [22], SAWSDL [166] ou WSMO [23], nécessitent beaucoup d'efforts d'apprentissage pour les utilisateurs. Par conséquent, pour faciliter leur utilisation, nous proposons une méthodologie basée MDA pour familiariser l'utilisateur habitué aux outils UML, à adopter la technologie des Web services sémantiques. Le principe est donc, de pouvoir générer une ontologie de services de manière automatique et efficace.

Comme le montre la figure 2.9, l'ontologie OWL-S fournit un mécanisme uniforme pour décrire la sémantique d'un Web service. Elle est composée de trois parties principales :

- Le Profil de service décrit les fonctionnalités du service. Dans ce sens, il contient des concepts structurels.
- Le Modèle de service (ou Modèle de Processus) décrit comment le service fonctionne. Il spécifie le comportement du service et permet d'exprimer les structures

de contrôle de flux telles que les boucles, les séquences, les conditions. Dans ce sens, il comporte des concepts dynamiques (ou comportementaux).

- Le Grounding du service décrit les modalités d'invocation du service. Il contient les caractéristiques techniques pour accéder au service au moyen des messages. Dans ce sens, il contient les concepts techniques.

Notre méthodologie de génération d'une ontologie OWL-S est similaire au processus de la rétro-ingénierie dans le sens il permet de générer une ontologie de services à partir de la spécification OWL-S ou de son schéma XML. Pour une génération automatique de l'ontologie OWL-S, nous extrayons d'abord les informations relatives aux concepts structurels, dynamiques et techniques à partir d'une description d'OWL-S. Les concepts structurels sont liés au profil de service d'OWL-S qui contient les propriétés telles que les IOPEs (entrées, sorties, pré-conditions et effets). Les concepts dynamiques sont liés à l'aspect comportemental du service. Les concepts techniques sont liés aux aspects techniques qui incluent des concepts de OWL-S (binding et grounding).

Dans notre extraction, nous utilisons les outils UML (par exemple UML 2.1 Plug-in sous la plate-forme Eclipse [179] pour créer *i)* un méta-modèle correspondant au profile de service d'un service Workflow, *ii)* un méta-modèle correspondant au processus de service *iii)* un méta-modèle correspondant au grounding du service. Ces trois méta-modèles créés sont ainsi stéréotypés avec des concepts OWL-S que nous appelons UPSWS (UML Profil pour des Web services sémantiques) qui sont au fait, des UML Profil pour OWL-S.

Pour générer une ontologie de services décrite en OWL-S, nous utilisons les mappings entre les concepts d'UPSWS, d'OUPSWS, d'ODM et d'OWL-S. Quant à la validation de cette ontologie de services générée, nous utilisons parmi les outils de validation cités précédemment, l'outil Protégé [191]. La figure 3.9 montre le processus de génération d'une ontologie OWL-S.

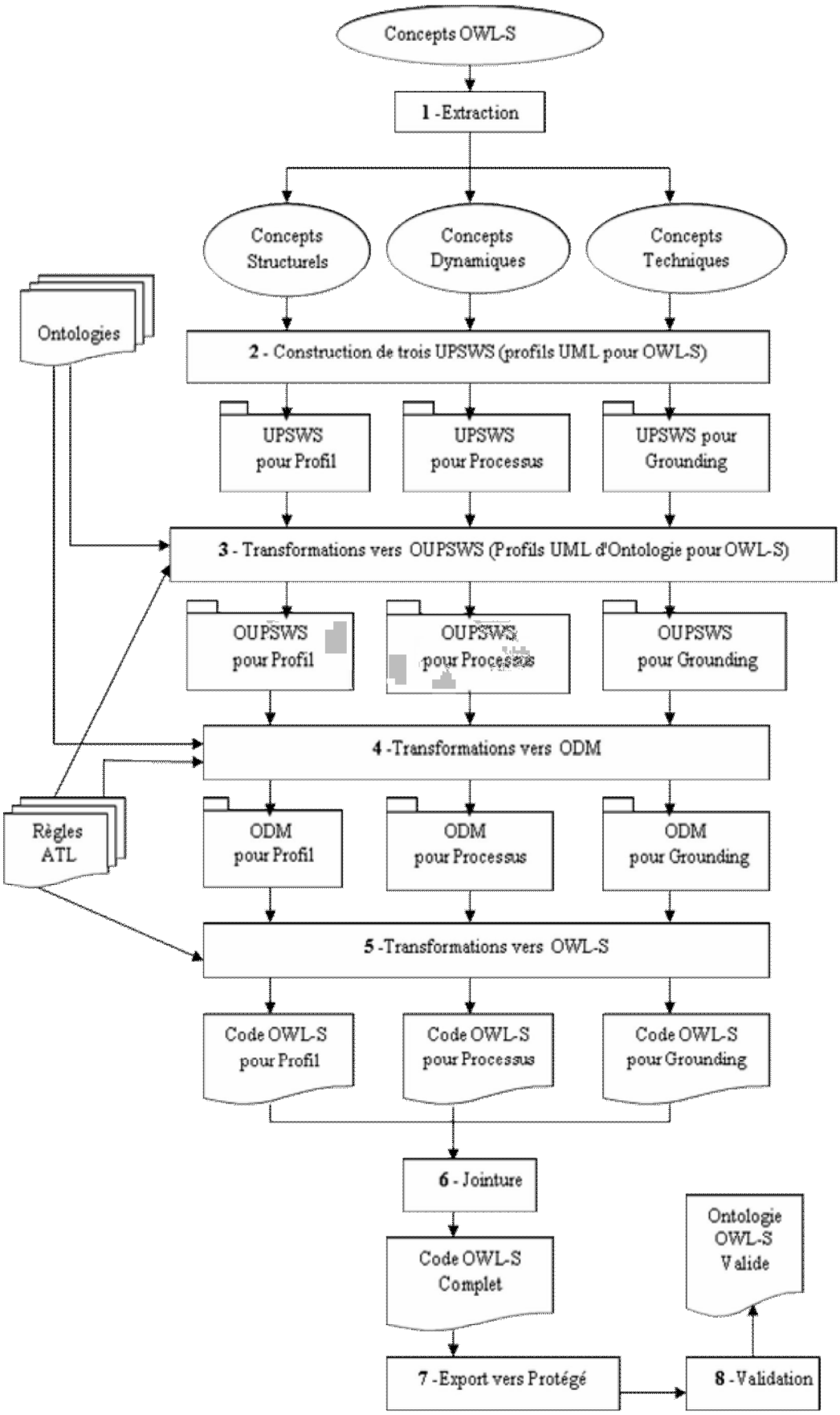

Figure 3.9 : Processus de génération d'une ontologie OWL-S**.**

Ce processus comporte les étapes suivantes :

- ***Etape 1** : Extraction de l'information relative aux concepts structurels, dynamiques et techniques de la description d'OWL-S.*

  Dans cette étape, d'abord, nous recherchons les informations relatives aux concepts structurels. Ces concepts correspondent à la description du profil de service qui contient une description textuelle pour les utilisateurs humains et une description fonctionnelle du service qui est référencée par les IOPEs (entrées, sorties, pré-conditions et effets). Les concepts dynamiques sont liés à l'aspect comportemental du service. Ils permetent de spécifier le flux de contrôle et de décrire la composition du service. Les concepts techniques sont liés aux aspects qui incluent les concepts WSDL de liaison (binding) et ceux liés au grounding.

- ***Etape 2 :** Construction de trois UPSWS (c.-à-d. trois Profils UML pour OWL-S).*

  Après extraction, nous utilisons l'outil UML (UML 2.1 de la plateforme Eclipse [179]) pour créer trois Profils UML avec des stéréotypes OWL-S qui incluent les constructeurs d'OWL-S (figure 3.11). Dans cette étape, nous obtenons trois méta-modèles : un méta-modèle pour le Profil de service *(ServiceProfile*), un méta-modèle pour le modèle de Processus (*ServiceProcess)* et enfin un méta-modèle pour le Grounding du Service (*Service*Grounding).

- ***Etape 3 et étape 4 :** Transformations d'UPSWS vers OUPSWS et d'OUPSWS vers ODM.*

  Dans ces étapes, nous transformons ces trois UPSWS vers leurs OUPSWS correspondants : un Profil UML d'ontologie pour le Profil de service (ServiceProfile), un Profil UML d'ontologie pour le modèle de Processus (ServiceProcess) et un Profil UML d'ontologie pour le Service Grounding (ServiceGrounding), puis de ces trois Profils OUPSWS vers ODMs correspondants (les méta-modèles de définition d'ontologies).

  Puisque tous les méta-modèles sont conformes au MOF (Meta-Object Facility) de l'architecture de MDA, nous générons d'abord des documents XMI (XML Metadata Interchange) pour ces méta-modèles (des formats XMI pour UML, OUPSWS et pour ODM). Ensuite, nous employons le langage ATL (ATLAS Transformation Language) [180] pour transformer les représentations XMI sources vers leurs représentations cibles correspondantes. Les règles de transformations utilisées pour les étapes 3 et 4 sont définies respectivement dans la table 3.2 et la table 3.3. Une fois, ces règles sont exécutées, .nous aurons enfin, trois représentations de XML pour les méta-modèles ODMs concernant le profil de service, le processus de service et le grounding du service.

- ***Etape 5 :** Mapping ODM vers OWL-S.*

  Dans cette étape, si nous désirons générer une description OWL, la translation d'ODM vers OWL est facile puisque les concepts d'OWL sont similaires à ceux d'ODM et les règles de transformations utilisées sont définies la table 3.3. Puisque nous sommes focalisés vers OWL-S, alors nous traduisons les concepts d'ODM vers ceux d'OWL-S en utilisant les règles de transformations définies dans la table 3.5, la table 3.6, la table 3.7 et la table 3.8.

- ***Etape 6 :*** *Jointure des trois documents OWL-S*

  La sixième étape est de joindre ces trois documents XML correspondant respectivement au profil de service, processus de Service, et au grounding du Service pour créer le code OWL-S complet correspondant à une ontologie de services.

- ***Etape 7*** *: Export du code OWL-S pour validation.*

Afin d'identifier toutes les erreurs syntaxiques dans la spécification générée OWL-S, les documents XMI sont exportés vers l'outil Protégé pour leur validation.

- ***Etape 8 :*** *Validation de l'ontologie OWL-S.*

  La dernière étape concerne la validation de l'ontologie afin de détecter éventuellement des erreurs syntaxiques ou des incohérences possibles dans la description OWL-S finale de l'ontologie. Ce qui nous incite à la consulter pour la corriger d'avantage.

4.3.4 Transformations et validation

Pour rester toujours dans le contexte de standardisation défini par MDA, nous avons utilisé le langage ATL (Atlas Transformation Language) [180] pour la réalisation de transformations de modèles, qui répond à la spécification MOF 2.0. QVT (Query/Views/Transformations) [142] définie par l'OMG (Object Management Group) en tant que méta-modèle de transformation de modèles.

Les méta-modèles UPSWS, OUPSWS, ODM et OWL-S sont d'abord, traduits vers des représentations XMI, puis nous réalisons les transformations via ATL. Ces transformations concernent les mappings de UPSWS vers OUPSWS, puis d'OUPSWS vers ODM et enfin d'ODM vers OWL ou OWL-S. L'outil ontologique Protégé permet l'édition de l'ontologie OWL-S et autres opérations semblables de manipulation, mais pour notre objectif, il sert uniquement comme un outil de validation pour des descriptions OWL-S générées.

4.3.4.1 Transformation UPSWS vers OUPSWS

Pour traduire UPSWS vers OUPSWS, nous utilisons les règles de transformation définies par la table 3.2. Les éléments de UPSWS dénotent des concepts ontologiques dans OUPSWS. Par conséquent, OUPSWS étend UML avec de nouveaux constructeurs pour supporter nos concepts ontologiques spécifiques dans le domaine des Web services sémantiques (WSSs) tels que les classes stéréotypes : << *Ontology* >> et << *OntClass* >>.

Les classes stéréotypées sont converties vers des classes ontologiques avec le stéréotype <<*OntClass*>>. Les valeurs étiquetées (ou tagged values) qui correspondent aux propriétés (ou attributs) sont converties vers des données ontologiques avec le stéréotype <<*DatatypeProperty*>>. Le stéréotype <<*Package*>> est transformé en <<*Ontology* >>. Quant aux associations stéréotypées, elles sont translatées vers le concept <<*ObjectProperty*>>.

Concepts UPSWS (Stéréotypes ou tags)	**Concepts OUPSWS (Concepts Ontologiques)**
« Package »	« Ontology »
« Class »	« OntClass »
«Association »	« ObjectProperty »
« Attribute »	« DatatypeProperty »

Table 3.2 : Mappings entre les concepts UPSWS et OUPSWS.

Par exemple, le stéréotype « ServiceModel » dans UPSWS deviendra un concept ontologique dans OUPSWS avec le stéréotype *« OntClass »* identifié par un identificateur de la classe.

4.3.4.2 Transformation OUPSWS vers ODM et OWL

Avant de faire le mapping, nous rappelons que le méta-modèle de définition d'ontologie (ODM) est conçu avec des concepts ontologiques et sa construction est basée sur OWL [28]. La figure 3.10 présente un extrait du méta-modèle ODM.

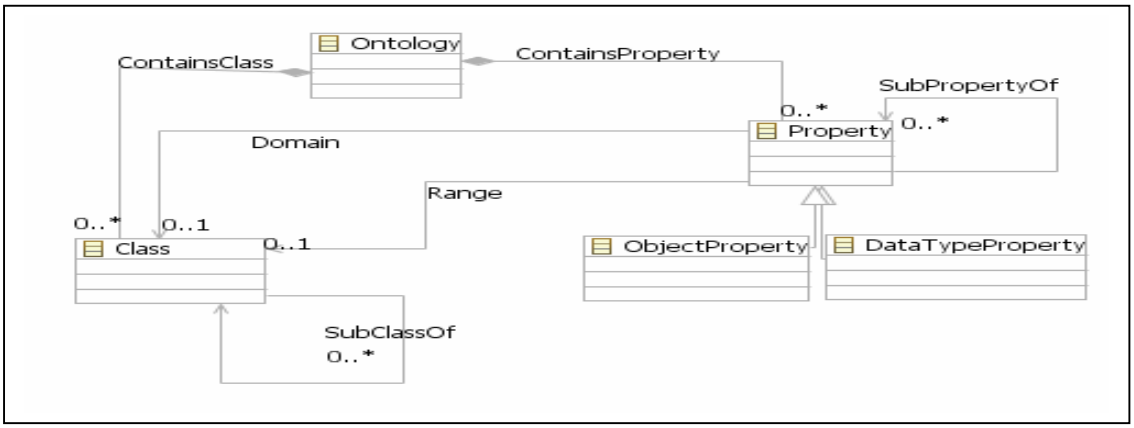

Figure 3.10 : Un extrait du méta-modèle ODM.

Les règles de transformations du OUPSWS au méta-modèle ODM et du ODM au méta-modèle OWL sont simples et directes du moment que les concepts d'ODM sont utilisés en tant que des stéréotypes dans OUPSWS, d'une part, et ODM est conçu selon les concepts du méta-modèle OWL, d'autre part. (ODM et OWL ont des concepts similaires).

Dans OUPSWS, les classes stéréotypées <<*Ontology*>> et <<*OntClass*>> sont utilisées pour identifier respectivement un domaine et un concept. Ils seront alors représentés comme des classes dans ODM. Le stéréotype <<*ObjectProperty*>> qui peut avoir seulement des individus dans sa portée et dans son domaine, est représenté comme des classes 'ObjectProperty' dans ODM. Quant au stéréotype <<*DatatypeProperty*>>, il est considéré comme une classe 'DatatypeProperty' dans ODM.

La table 3.2 montre les mappings entre les concepts OUPSWS, ODM et OWL.

Concepts OUPSWS (Concepts ontologiques comme stéréotypes ou tags values)	**Concepts ODM (Concepts ontologiques)**	**Concepts OWL**
«Ontology »	Class Ontology	OWL : Ontology
«OntClass»	Class Class	OWL : Class
«ObjectProperty»	Class ObjectProperty	OWL : ObjectProperty
«DatatypeProperty»	Class DatatypeProperty	OWL : DatatypeProperty
«Wsdl » «port » «portType » «service » «operation »	Class Grounding Wsdl Class Grounding portType Class Grounding port Class Grounding service Class Grounding operation	OWL : Grounding Wsdl OWL : Grounding portType OWL : Grounding port OWL : Grounding service OWL : Grounding operation

Table 3.3 : Mappings entre les concepts OUPSWS, ODM et OWL.

4.3.4.3 Transformation ODM vers les méta-modèles d'ontologie de services

Pour rester dans un cadre général de génération d'une ontologie de services, nous avons essayé d'établir la correspondance entre les concepts de trois méta-modèles d'ontologies de services différents tels que WSML, WSDL-S et OWL-S.

- **Mapping entre les concepts ODM, WSML et WSDL-S**

Les parties des ontologies dans WSML correspondent aux ontologies OWL dans le document OWL-S ou à la partie d'OUPSWS. Il n'y a aucun support pour les buts (Goals) dans OWL-S ou dans notre Profil UML d'ontologie spécifique. Cependant, on pourrait introduire un nouveau stéréotype appelé 'Goal' qui partage toutes les propriétés du concept de Web service sémantique excepté les quatre valeurs étiquetées de WSDL (tagged values de WSDL).

Les médiateurs traitant les transformations et les relations peuvent être classés en quatre catégories : OO-, WW-, GG- et WG-médiateurs. Le médiateur OO est la traduction d'une ontologie à une autre. Le WW-médiateur manipule l'interopérabilité entre les Web services. Le GG-médiateur exprime des relations entre différents buts (goals). Le WG-médiateur fait la correspondance entre un but avec un Web service. Il n'y a aucun équivalent des médiateurs dans OWL-S. Les parties d'un Web service de WSML correspondent à notre Profil de Web service sémantique dans OUPWSS et dans OWL-S.

En revanche, WSDL-S fournit cinq éléments et attributs qui, une fois utilisés dans les parties appropriées d'un document de WSDL, fournissent des moyens pour attacher la sémantique aux composants de WSDL. Ces attributs sont : < *wssem:modelReference* >, < *precondition* >, <*wssem:schemaMapping* > , < *effect* > et < *category* >.

Par exemple, l'attribut d'extension <*wssem:modelReference*> supporte les connections entre un document WSDL et une ontologie (par exemple, modélisée en OWL). La table 3.4 énumère un ensemble de directives élémentaires que nous avons identifiées pour traduire les concepts d'ODM dans les concepts de WSML et de WSDL-S.

Concepts ODM	Concepts WSML	Concepts WSDL-S
Class Class Semantic Workflow service	Web service	Web service Semantics : Wssemantics referenced by URI
Class Ontology	ontology	Imported ontology (or external semantic model), referenced by URI
Class Class	concept	concept referenced from a semantic model (ontology)
Class Class Input	imported Concepts avec mode hasvalue wsml # in	Semantic Annotations of Inputs by referring to a part of a semantic model (or ontology)
Class Class Output	imported Concepts avec mode hasvalue wsml # out	Semantic Annotations of Outputs by referring to a part of a semantic model (or ontology)
Class Class PreCondition	precondition	wssem : precondition wssem : modelReference or wssem:schemaMapping
Class Class PostCondition	postcondition	wssem : postcondition wssem : modelRreference or wssem:schemaMapping
Class Class Effect	effect	wssem : effect wssem : modelReference or wssem:schemaMapping

Table 3.4 : Mappings entre les concepts d'ODM, WSML et WSDL-S.

4.3.4.4 Transformation ODM vers OWL-S

Dans les langages ontologiques, les attributs ou les associations s'appellent des propriétés. Une propriété est une relation binaire entre une ressource 'sujet' et une ressource 'objet'. Dans ODM, nous distinguons deux types de propriétés : *Class ObjectProperty*, dont la portée peut être seulement un individu, et Class *DatatypeProperty* dont la portée peut être une seule valeur de données [182]. Ainsi, si nous souhaitons mapper l'attribut à *OwlProperty*, nous devons connaître la portée de la propriété.

Chaque classe d'ODM identifiée par un identificateur de classe correspond à une classe OWL-S (*OWL: Class*) dans l'ontologie OWL-S de service. Chaque attribut d'ODM est donc traduit en *Owl:DataProperty* dans OWL-S si le type d'attribut peut avoir des valeurs d'un certain type de données (par exemple : chaînes de caractères), sinon en *Owl: ObjectProperty*. Chaque relation établie entre les classes d'ODM est traduite vers *OWL : ObjectProperty* dans OWL-S. Quant à la partie 'Grounding' du modèle de service de OWL-S, elle contient des pointeurs vers le fichier WSDL qui définit les détails techniques pour accéder au Web service.

De plus, OWL-S utilise *WsdlGrounding* pour se référer à des constructions de WSDL. Chaque instance de *WsdlGrounding* contient à son tour, une liste d'instances de *WsdlAtomicProcessGrounding*. Une instance de *WsdlAtomicProcessGrounding* se réfère aux éléments spécifiques de la spécification WSDL, utilisant les propriétés telles que wsdlService, wsdlPort, wsdlInputMessage. Par exemple, le wsdlService, wsdlPort présentent l'URI d'un service WSDL (ou port) qui offre l'opération donnée. Pour plus de détails sur OWL-S Grounding, nous nous référons le lecteur à [21]. La table suivante montre les mappings entre entre les concepts d'ODM et d'OWL-S.

Source : Concepts ODM	Cible : Concepts OWL -S
Class Ontology	owl: Ontology rdf : about=" "
Class Class	Owl : Class rdf : ID = " "
Class DatatypeProperty, si le type de propriété est lié aux valeurs de données.	OWL: DatatypeProperty, si le type de propriété est lié aux valeurs de données.
Class ObjectProperty, si le type de propriété est lié aux classes.	OWL: ObjectProperty, si le type de propriété est lié aux classes.
Class Grounding Wsdl Class Grounding Port Class Grounding PortType Class Grounding Service Class Grounding Operation	URI : wsdlVersion, wsdlDocument URI: wsdlPort URI: wsdlPortType URI: wsdlService URI: wsdlOperation En fait : OWL-S Grounding Ontology et les correspondances avec le Document WSDL: constructeurs d'Owl-S liés à ceux de WSDL. Les Propriétés et les Classes définies dans Grounding.owl : Constructeurs d'OWL-S sont :: wsdlReference, wsdlDocument, wsdlOperation, wsdlOperationrRef, wsdlInputMessage, wsdlInputMessageMap, wsdlMessagePart, owlsParameter, wsdlOutputMessage, wsdlOutputputMessageMap, WSDL Constructs are : message, portType, service, port

Table 3.5 : Mappings entre les concepts d'ODM et OWL-S.

4.3.4.5 Validation de l'ontologie de services

Plusieurs outils de validation existent. Néanmoins la plupart d'entre eux ne fonctionnement pas ou ne sont pas matures. Par exemple, le site [192] disponible du W3C pour le test au niveau RDF, des validateurs de OWL sont disponibles, l'outil [193] pour vérifier l'ontologie, OWL-S API [189] qui est une API Java pour créer, valider, et exécuter des spécifications OWL-S, OWL-S Editor [190]. Dans notre travail, nous utilisons l'outil Protégé [191] pour valider l'ontologie de services générée.

4.3.5 Génération de l'ontologie OWL-S de service

Trois sous-ontologies sont générées. La première correspond au profil de service, la deuxième au processus de service et la troisième au service Grounding.

4.3.5.1 Génération du Profil de service

La table suivante montre les mapings entre quelques concepts ODM du profil de servicevers ceux d'OWL-S.

Source : Concepts ODM du Profil de service	Cible : Concepts OWL-S du Profil de service
Class Ontology	owl: Ontology rdf : about=" "
Class Class Service	Owl : Class rdf : ID = " Service "
Class Class Profile	owl:Class rdf :I D ="Profile"
Class Class ServiceProfile	Owl : Class rdf : ID = " ServiceProfile "
Class Class Category	Owl : Class rdf : ID = " Category "
Class DataTypeProperty CategoryName	Owl : DataTypeProperty rdf : ID =" CategoryName "
Class DataTypeProperty Taxonomy	Owl : DataTypeProperty rdf : ID =" Taxonomy "
Class DataTypeProperty Value	Owl : DataTypeProperty rdf : ID =" Value "
Class DataTypeProperty Code	Owl : DataTypeProperty rdf : ID =" Code "
Class Class Input	Owl : Class rdf : ID = " Input "
Class ObjectProperty supports	Owl : ObjectProperty rdf : ID =" supports "
Class ObjectProperty presents	Owl : ObjectProperty rdf : ID =" presents "
Class ObjectProperty hasCategory	Owl : ObjectProperty rdf : ID = " hasCategory "
Class ObjectProperty describedBy	Owl : ObjectProperty rdf : ID = " describedBy "
Class ObjectProperty hasInput	Owl : ObjectProperty rdf : ID = " hasIntput "
Class ObjectProperty hasOutput	Owl : ObjectProperty rdf : ID = " hasOutput "
Class ObjectProperty hasPreCondition	Owl : ObjectProperty rdf : ID = " hasPreCondition "
Class ObjectProperty hasEffect	Owl : ObjectProperty rdf : ID = " hasEffect"
Class DataTypeProperty serviceName	Owl : DataTypeProperty rdf : ID =" serviceName"
Class DataTypeProperty textDescription	Owl : DataTypeProperty rdf : ID =" textDescription"

Table 3.6 : Mappings des concepts ODM du profil de servicevers ceux d'OWL-S.

4.3.5.2 Génération du processus de service

Dans OWL-S, nous distinguons des processus atomiques et des processus composés. Les processus atomiques [21] sont directement invocables et s'exécutent en une seule étape.

Un processus composé comporte un certain nombre d'étapes qui peuvent représenter n'importe quel type de processus. Pour modéliser les flux de contrôle, des structures de contrôle (Sequence, Split, Split-Join, Any-Order, If-Then-Else, etc.) sont disponibles pour composer des sous-processus. Ainsi, pour générer un processus OWL-S de service qui décrit l'aspect dynamique du service, nous devons s'appuyer principalement sur le diagramme d'activité du Profil UML, ensuite, nous le translatons vers le Profil UML d'ontologie et puis vers ODM qui contient toutes les structures de contrôle définies dans le processus OWL-S de service.

Par exemple, le stéréotype «If-Then-Else» qui est défini dans le Profil UML pour OWL-S, sera défini par « OntClass » identifié par un identificateur de classe dans notre Profil UML d'ontologie (OUPSWS) et deviendra une classe ODM. Par conséquent, le mapping entre les concepts ODM et ceux d'OWL-S est simple. La table suivante montre le mapping entre quelques concepts ODM de processus de service vers ceux définis dans OWL-S.

Source : Concepts ODM du processus de service	Cible : Concepts OWL-S du processus de service
Class Class CompositeProcess	owl:Class rdf : about=" CompositeProcess"
Class Class SimpleProcess	owl:Class rdf : about=" SimpleProcess"
Class Class AtomicProcess	owl:Class rdf : about=" AtomicProcess"
Class ObjectProperty hasInput	owl: wl:ObjectProperty rdf:ID=" hasInput"
Class ObjectProperty hasOutput	owl:ObjectProperty rdf:ID=" hasOutput"
Class ObjectProperty hasPreCondition	Owl : ObjectProperty rdf : ID = " hasPreCondition "
Class ObjectProperty hasEffect	Owl : ObjectProperty rdf : ID = " hasEffect"
Class ObjectProperty hasResult	owl:ObjectProperty rdf:ID=" hasResult"
Class Class Input	Owl : Class rdf : ID = " Input "
Class Class Output	Owl : Class rdf : ID = " Output "
Class ObjectProperty ControlConstruct	owl:Class rdf:ID=" ControlConstruct"
Class ObjectProperty Sequence	owl:Class rdf:about=" Sequence"
Class ObjectProperty Any-Order	owl:Class rdf:about=" Any-Order"
Class ObjectProperty Choice	owl:Class rdf:about=" Choice"
Class ObjectProperty Split-Join	owl:Class rdf:ID=" Split-Join"
Class ObjectProperty If-Then-Else	<<owl:Class rdf:ID=" If-Then-Else"
Class ObjectProperty Repeat-While	owl:Class rdf:ID=" Repeat-While"
Class ObjectProperty Repeat-Until	owl:Class rdf:ID=" Repeat-Until"

Table 3.7 : Mapping entre les concepts ODM du processus de service vers ceux d'OWL-S.

4.3.5.3 Génération du grounding de service

Pour obtenir le grounding du service, nous devons traduire les classes grounding d'ODM qui sont des concepts ontologiques dans OWL-S (partie Grounding) pour générer une instance du grounding du service (ServiceGrounding). Ce dernier contient tous les éléments spécifiques de la spécification WSDL, et utilise les concepts du grounding d'OWL-S suivants (Classes, DataTypes ou ObjectPropeties) et les propriétés à partir de l'instance WsdlAtomicProcessGrounding qui fait référence aux éléments spécifiques de la spécification WSDL tels que wsdlVersion, wsdlOperation, wsdlService, wsdlPort, wsdlInputMessage, wsdlInput, wsdlOutput, WsdlInputMessageMap, wsdlOutputMessage. La table suivante montre le mapping entre quelques concepts ODM et ceux d'OWL-S Grounding.

Source : Concepts ODM Grounding	Cible : Concepts OWL-S Grounding
Class Class Grounding	owl:Class rdf:ID="Grounding"
Class Class AtomicProcessGrounding	owl:Class rdf:ID="AtomicProcessGrounding"
Class ObjectProperty hasAtomicProcessGrounding	owl:ObjectProperty rdf:ID="hasAtomicProcessGrounding"
Class ObjectProperty owlsProcess	owl: ObjectProperty rdf:ID="owlsProcess"
Class Class WsdlGrounding	owl:Class rdf:ID="WsdlGrounding"
Class Class WsdlAtomicProcessGrounding	owl:Class rdf:ID="WsdlAtomicProcessGrounding"
Class ObjectProperty wsdlOperation	<owl:ObjectProperty rdf:ID="wsdlOperation"
Class DatatypeProperty wsdlService	owl:DatatypeProperty rdf:ID="wsdlService"
Class DatatypeProperty wsdlPort	owl:DatatypeProperty rdf:ID="wsdlPort"
Class DatatypeProperty wsdlInputMessage	owl:DatatypeProperty rdf:ID="wsdlInputMessage"
Class ObjectProperty wsdlInput	owl:ObjectProperty rdf:ID="wsdlInput"
Class DatatypeProperty wsdlOutputMessage	owl:DatatypeProperty rdf:ID="wsdlOutputMessage"
Class ObjectProperty wsdlOutput	owl:ObjectProperty rdf:ID="wsdlOutput"
Class DatatypeProperty wsdlVersion	owl:DatatypeProperty rdf:ID="wsdlVersion"
Class Class WsdlOperationRef	owl:Class rdf:ID="WsdlOperationRef"
Class DatatypeProperty portType	<owl:DatatypeProperty rdf:ID="portType"
Class DatatypeProperty operation	owl:DatatypeProperty rdf:ID="operation"
Class Class WsdlMessageMap	owl:Class rdf:ID="WsdlMessageMap"
Class DatatypeProperty wsdlMessagePart	owl:DatatypeProperty rdf:ID="wsdlMessagePart"

Table 3.8 : Mappings entre les concepts ODM Grounding vers ceux d'OWL-S.

Toutefois, certains aspects de cette génération peuvent être critiqués. Ce qui nous incite à la valider via un outil ontologique tel que Protégé, et la corriger éventuellement. Certainement, notre méthode de génération d'ontologie de services n'est pas complètement automatisée, mais elle semble efficace parce qu'elle s'exécute dans un environnement familier aux développeurs tel que UML 2.1 (Unified Modeling Language) d'éclipse). Ce qui nous permet de fournir des informations nécessaires qui sont demandées lors du développement ou de la génération de l'ontologie. Cet environnement familier signifie que nous utilisons seulement des Profils UML pour modéliser notre service Workflow, et que nous n'avons pas besoin de comprendre les concepts du langage OWL-S pour décrire la sémantique de notre service.

4.4. Etude de cas : génération d'une ontologie OWL-S d'un service Workflow de livraison d'une commande d'achat

Dans cette section, nous présentons un exemple simple d'un service Workflow que nous que nous modélisons comme un Profil UML pour Web services sémantiques, c'est-à-dire, un Profil UML pour OWL-S. Ce processus ne comporte qu'une seule activité qui est "livrer commande d'achat " (PurchaseOrderShipping), et qui est équivalente à un processus atomique dans OWL-S. Une fois, ce Profil UML pour OWL-S est établi, nous suivons le processus de génération de l'ontologie de services selon les étapes que nous avons définies précédemment.

Ce processus Workflow concerne une livraison d'une commande d'achat qui peut être employé dans le cadre d'une sous-traitance (coopération entre des processus Workflows). C'est un processus atomique, équivalent à un service Workflow (ou 'service' tout court) qui livre des produits (par exemple, des micro-ordinateurs) à un client d'une compagnie A, qui sous-traite chez une compagnie B. Les entrées du service sont les propriétés du produit commandé (code, description, quantité et date), ainsi que les informations de la carte de crédit du client (carte de crédit, n° carte crédit, date expiration de la carte). Les sorties du service (conditionnelles et exclusives) sont au nombre de trois. La première sortie indique que le produit est acheté et livré à l'adresse du client, alors que la deuxième sortie fournit un message informant que le produit n'existe pas en stock. Quant à la troisième sortie, un reçu est délivré par le service. Les pré-conditions pour livrer le produit doivent vérifier qu'on doit avoir la carte de crédit valide avec sa date d'échéance valide. Elles sont attachées aux paramètres d'entrée. Les post-conditions qui sont liées aux paramètres de sortie doivent être vérifiées dès que le service termine son exécution. Enfin, l'élément d'effet indique le résultat de la réussite de l'exécution du service de livraison. Il indique que la carte de crédit est débitée. Ce service Workflow appelé " Purchase Order Shipping" est modélisé en tant que Web service sémantique, en utilisant la notation UML stéréotypée pour OWL-S (figure 3.11).

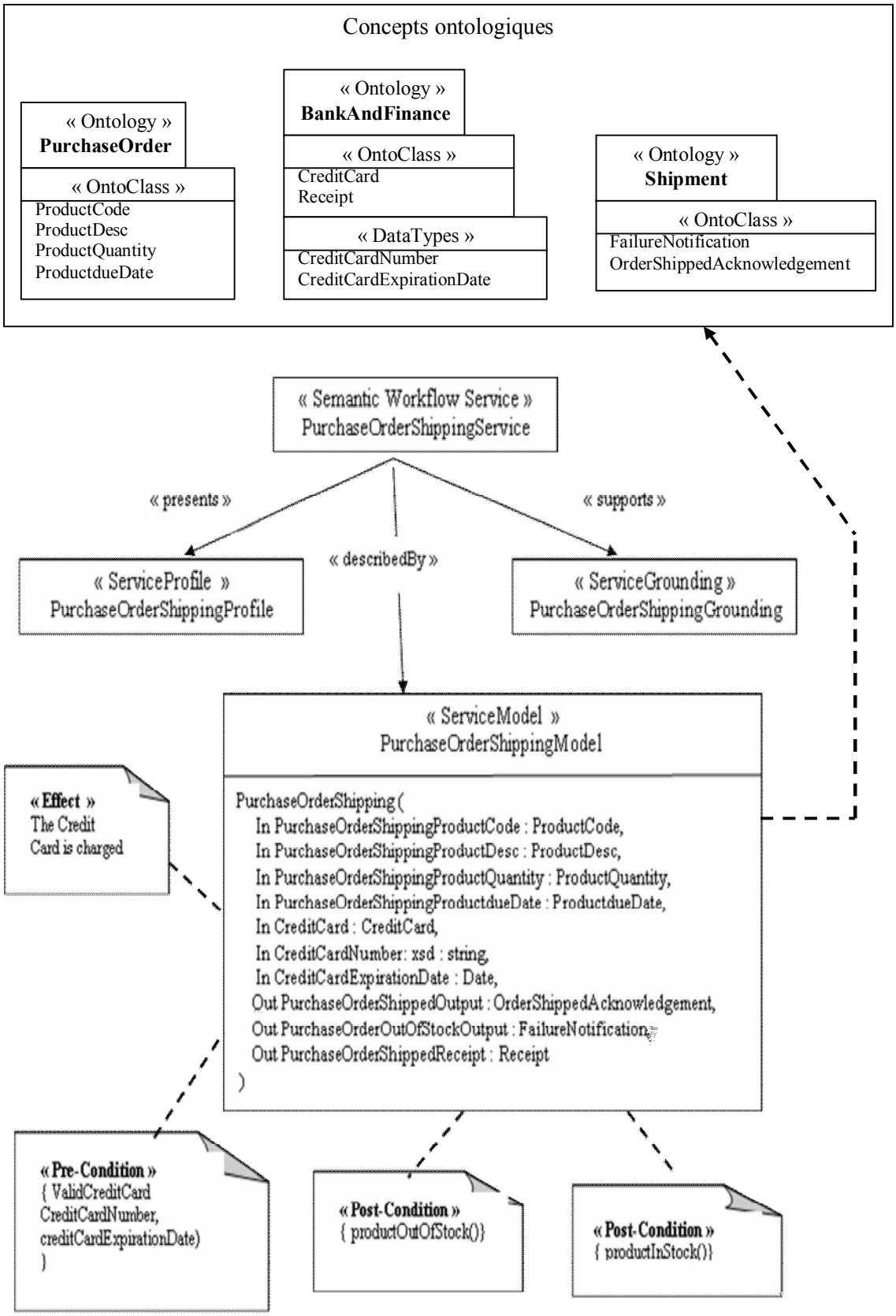

Figure 3.11 : Modélisation d'un service de livraison d'une commande d'achat.

Pour générer notre ontologie OWL-S de service Workflow, nous avons utilisé l'environnement Eclipse avec UML 2.1 (Unified Modeling Language) et ATL (ATLAS Transformation Language) [180] comme des plug-ins. UML 2.1 est utilisé pour construire des Profils UML et ATL est choisi comme un langage qui est conforme à la spécification MOF 2.0 /QVT de MDA [142] pour les transformations. Pour valider notre ontologie, nous avons utilisé l'outil Protégé [191].

La figure suivante présente une capture d'écran de notre service sous Eclipse.

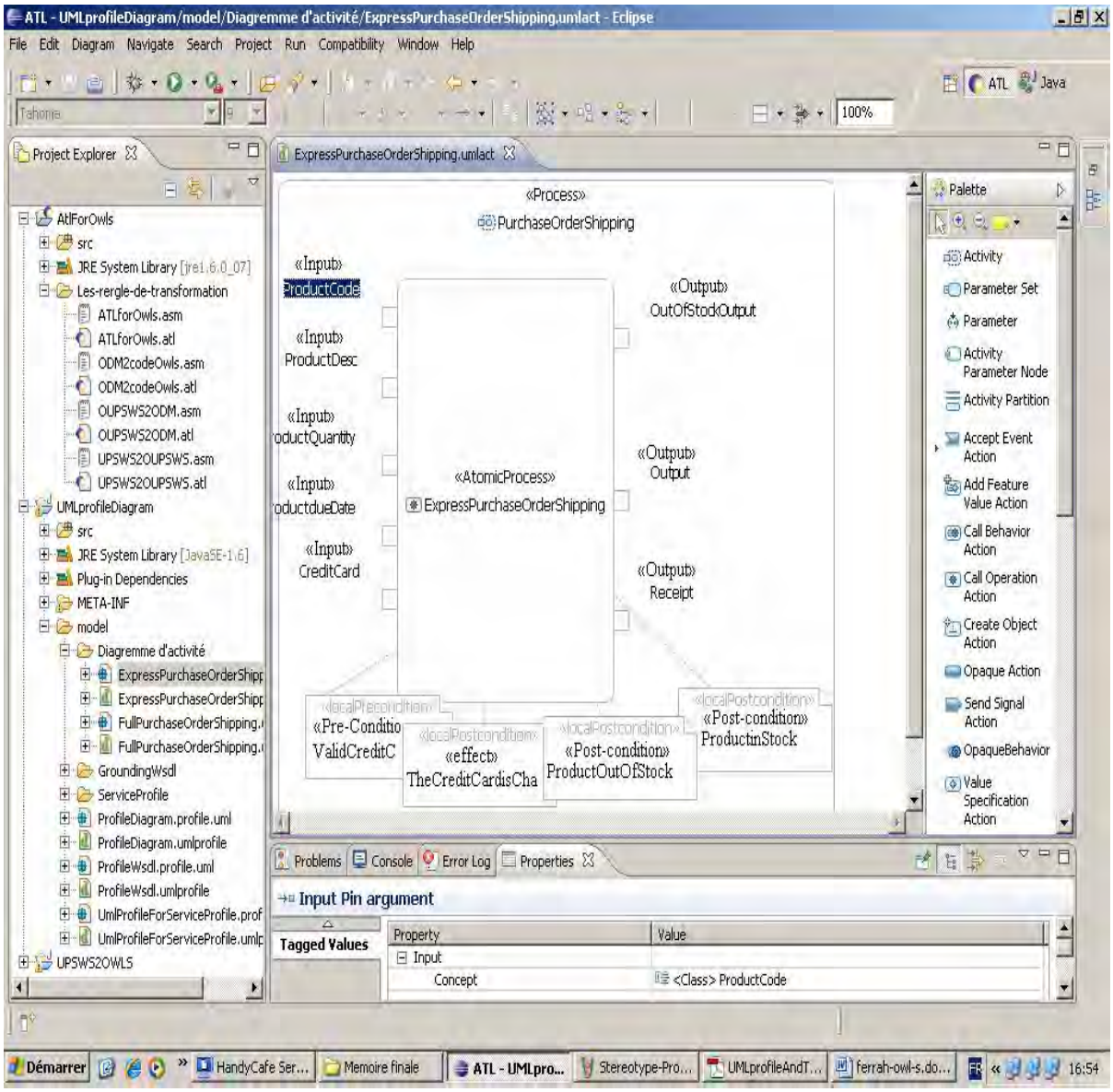

Figure 3.12 Service Workflow : 'PurchaseOrderShipping'

Pour illustrer notre processus MDA de génération décrit précédemment, nous allons présenter uniquement quelques captures d'écran liées à l'élaboration des Profils UML pour OWL-S, à leurs transformations, puis à la validation de l'ontologie de services.

4.4.1 Construction des Profils UML pour OWL-S (UPSWS)

L'outil de modélisation est UML 2.1, alors pour construire les trois Profils UML pour OWL-S, nous devons d'abord, construire le package qui contient l'ensemble des stéréotypes OWL-S. Ces stéréotypes correspondent au fait à l'ensemble des concepts de la spécification modulaire OWL-S. La figure suivante montre une capture d'écran de ce package.

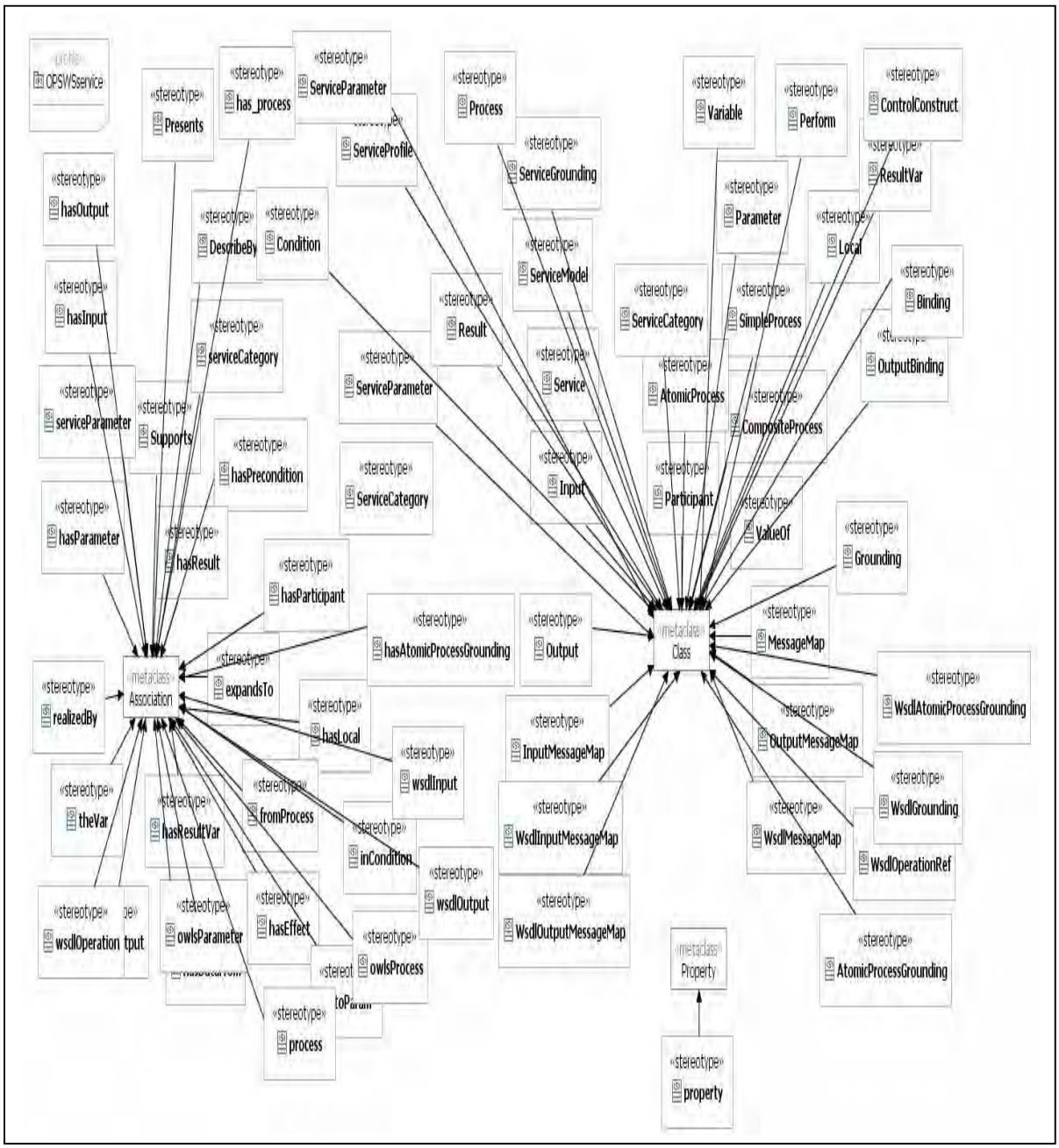

Figure 3.13 : Capture d'écran du Package de stéréotypes OWL-S.

Une fois, le package construit, l'utilisateur construit quatre Profils UML (UPSWS) correpondant à la spécification modulaire d'OWL-S et concernent quatre parties : *1)* la partie "service" qui est racine de la spécification dont on a besoin par la suite pour la jointure lors de la génération et de la validation *2*) la partie "profil de service (ServiceProfile)" *3)* la partie "processus de service (ServiceProcess)" *4)* la partie "service grounding (serviceGrounding)".

Pour construire ces quatre profils UML pour OWL-S (nommés "UPSWS"), l'utilisateur doit d'abord, sélectionner les stéréotypes OWL-S correspondant à chaque partie ("service", "profil de service", "processus de service" ou "service grounding") à partir du package des stéréotypes OWL-S disponible en haut à gauche dans le menu de la fenêtre d'écran. Une fois, élaboré, chaque Profil UML est alors sauvegardé dans son package correspondant. Puisque les transformations sont nombreuses, nous nous contentons uniquement de présenter quelques étapes de construction et de transformation de ces Profils UML.

❖ Construction du Profil UML (UPSWS) pour la partie "service"

C'est le premier Profil UML à construire et concerne la partie "service". Pour le construire l'utilisateur doit sélectionner, les stéréotypes OWL-S qui correspond à cette partie à partir du package des stéréotypes OWL-S disponible dans le menu de la fenêtre d'écran. La figure 3.14 montre une capture d'écran de cette partie.

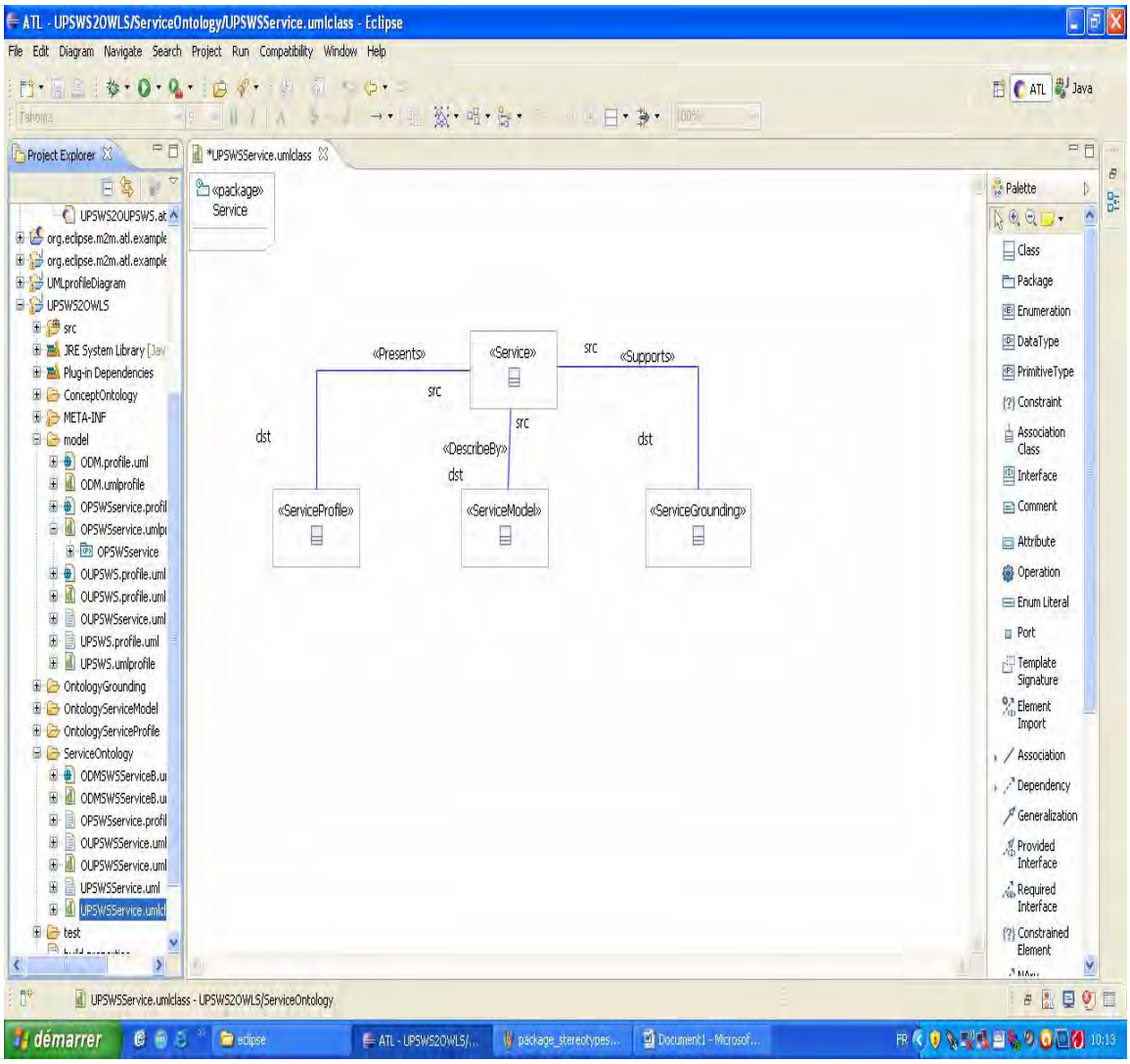

Figure 3.14 : Capture d'écran d'UPSWS (partie "Service").

Une fois, la partie "service" construite, l'utilisateur commence à construire les profils UML concernant les autres parties (profil de service, procesus de service et service grounding) en choisissant les stéréotypes OWL-S correspondant à chaque partie à partir du package OWL-S.

❖ Construction des trois Profils UML (UPSWS)

Afin de réduire le nombre de captures d'écran à présenter, nous nous contentons de présenter uniquement un Profil UML pour OWL-S (partie "Profil de service").

▪ Profil UML (UPSWS) pour la partie "profil de service"

La figure 3.15 montre une capture d'écran du Profil UML de la partie "profil de service".

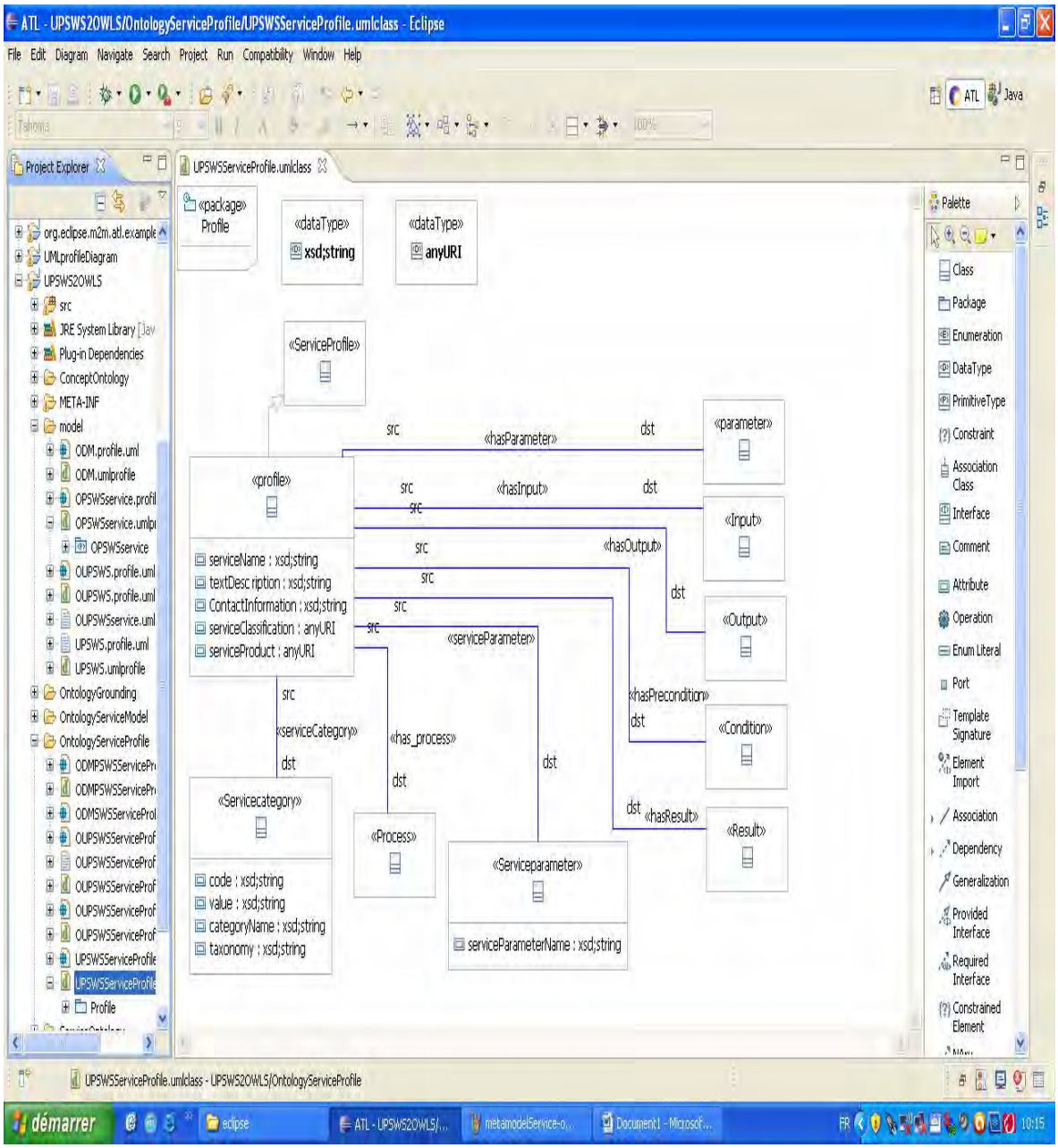

Figure 3.15 : Capture d'écran d'UPSWS (partie "profil de service").

Une fois les trois Profils UML sont construits (profil de service, processus de service et service grounding), l'utilisateur effectue les transformations que nous avons définies dans les tables de correspondance entre les concepts de UPSWS, OUPSWS, ODM et OWL-S.

4.4.2 Transformations UPSWS vers OUPSWS

Ces transformations concernent les mappings de concepts d'UPSWS vers ceux d'OUPSWS. OUPSWS est un Profil UML d'ontologie pour OWL-S et ses concepts sont ontologiques. Avant d'effectuer ces tranformations, nous devons d'abord construire un package contenant un ensemble de stéréotypes pouvant supporter des concepts ontologiques tels que "Ontology", "Ontoclass", "ObjectProperty", "DatatypeProperty". Les stéréotypes que nous créons devraient par la suite supporter le méta-modèle de définition d'ontologie (ODM) dans le but de faciliter le mapping entre OUPSWS et ODM. Une fois, crées, l'utilisateur peut effectuer ses transformations. La figure suivante une capture d'écran du package des stéréotypes OUPSWS.

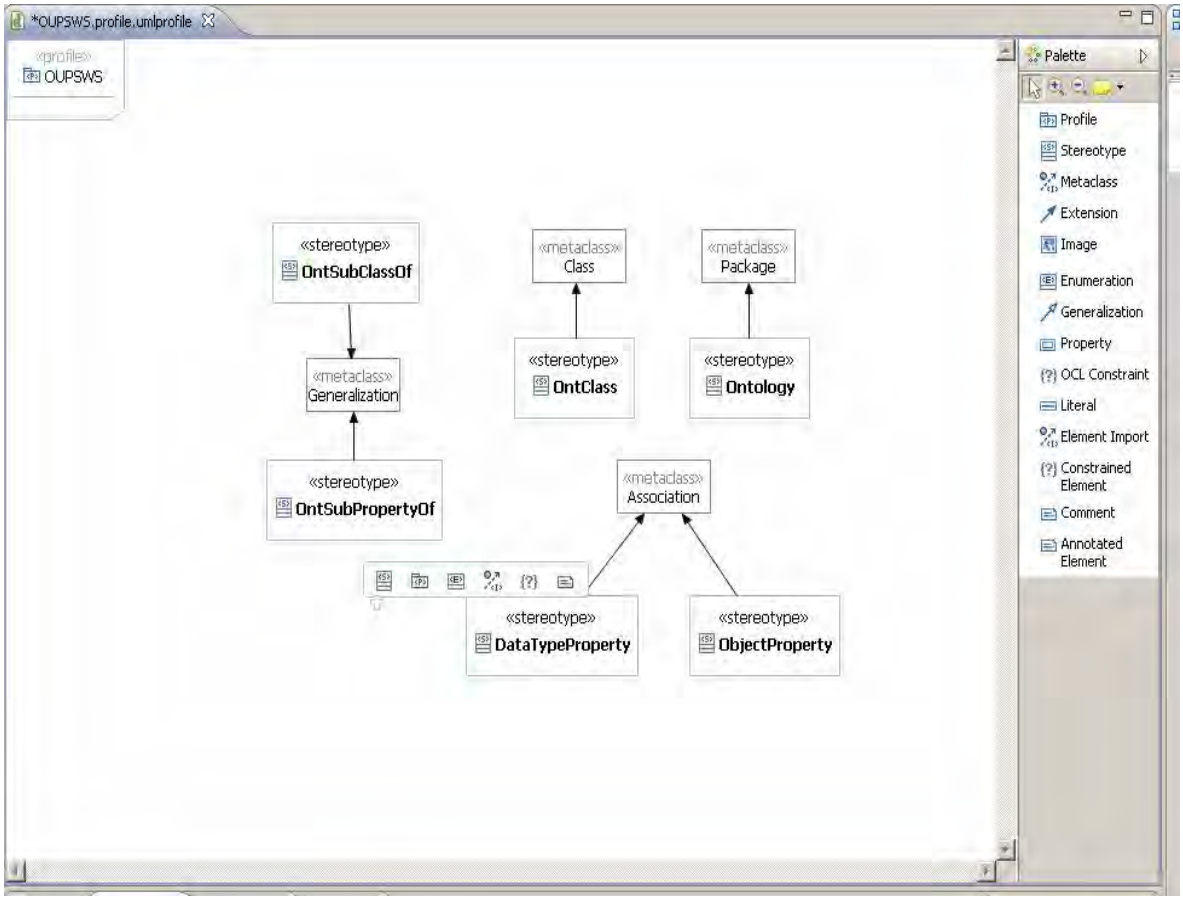

Figure 3.16 : Capture d'écran du Package des stéréotypes OUPSWS.

Une fois, les stéréotypes OUPSWS créés, nous traduisons les quatre Profils UML (UPSWS), construits précédemment, vers leurs correspondants OUPSWS. Nous rappelons que ces quatre Profils UML concernent les parties : "service", "profil de service", "processus de service" et " service grounding Pour ce faire, l'utilisateur exécute les règles de transformations définies la table 3.2. La figure suivante montre une capture d'écran des règles ATL d'exécution d'UPSWS vers OUPSWS.

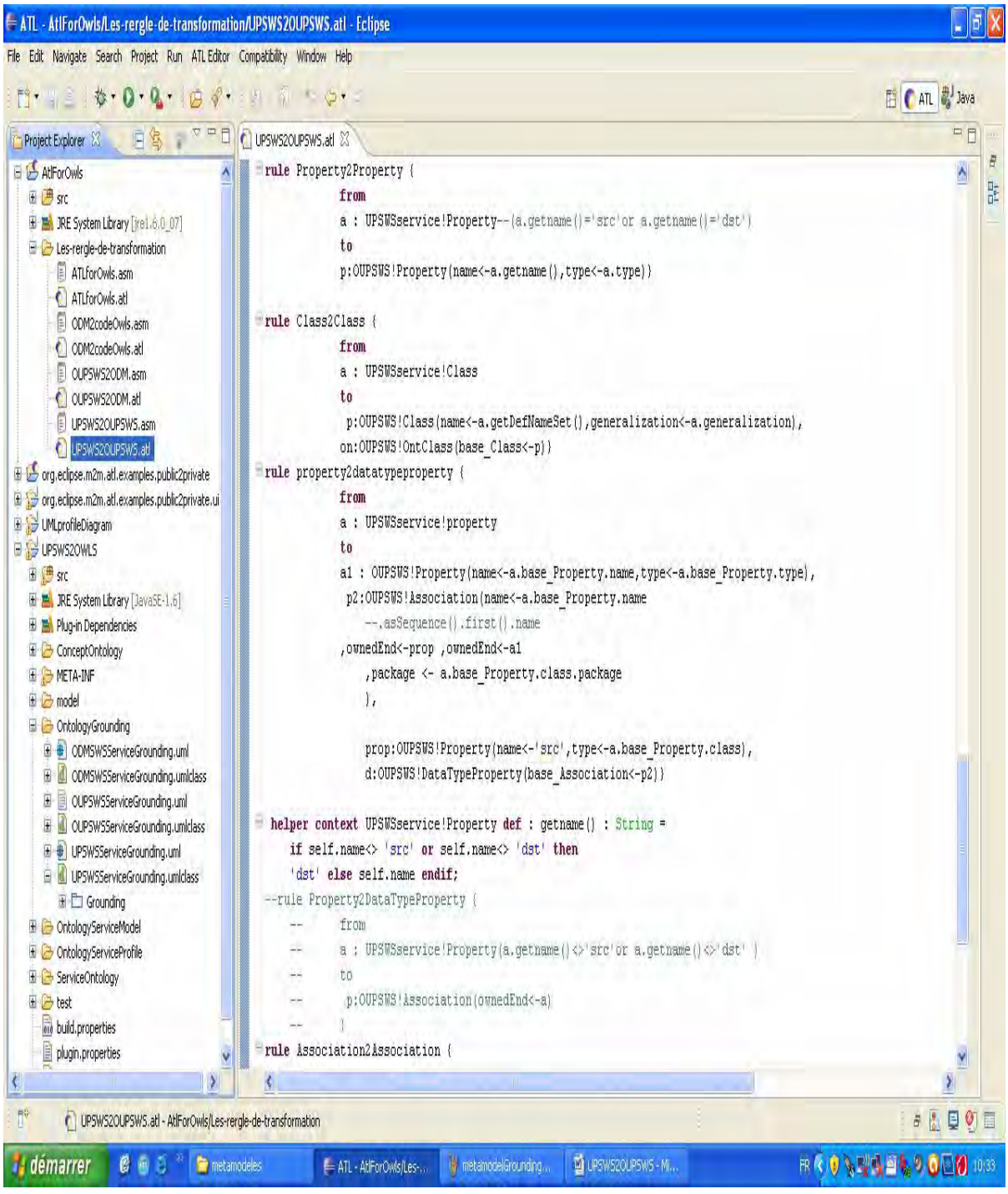

Figure 3.17 : Fenêtre d'écriture des règles ATL d'UPSWS vers OUPSWS

Ces règles génèrent quatre profils UML d'ontologies (OUPSWS) correspondant à ces quatre parties. En sélectionnant la partie concernant "le profil de service", les règles ATL de UPSWS vers OUPSWS s'exécutent comme le montre la figure suivante.

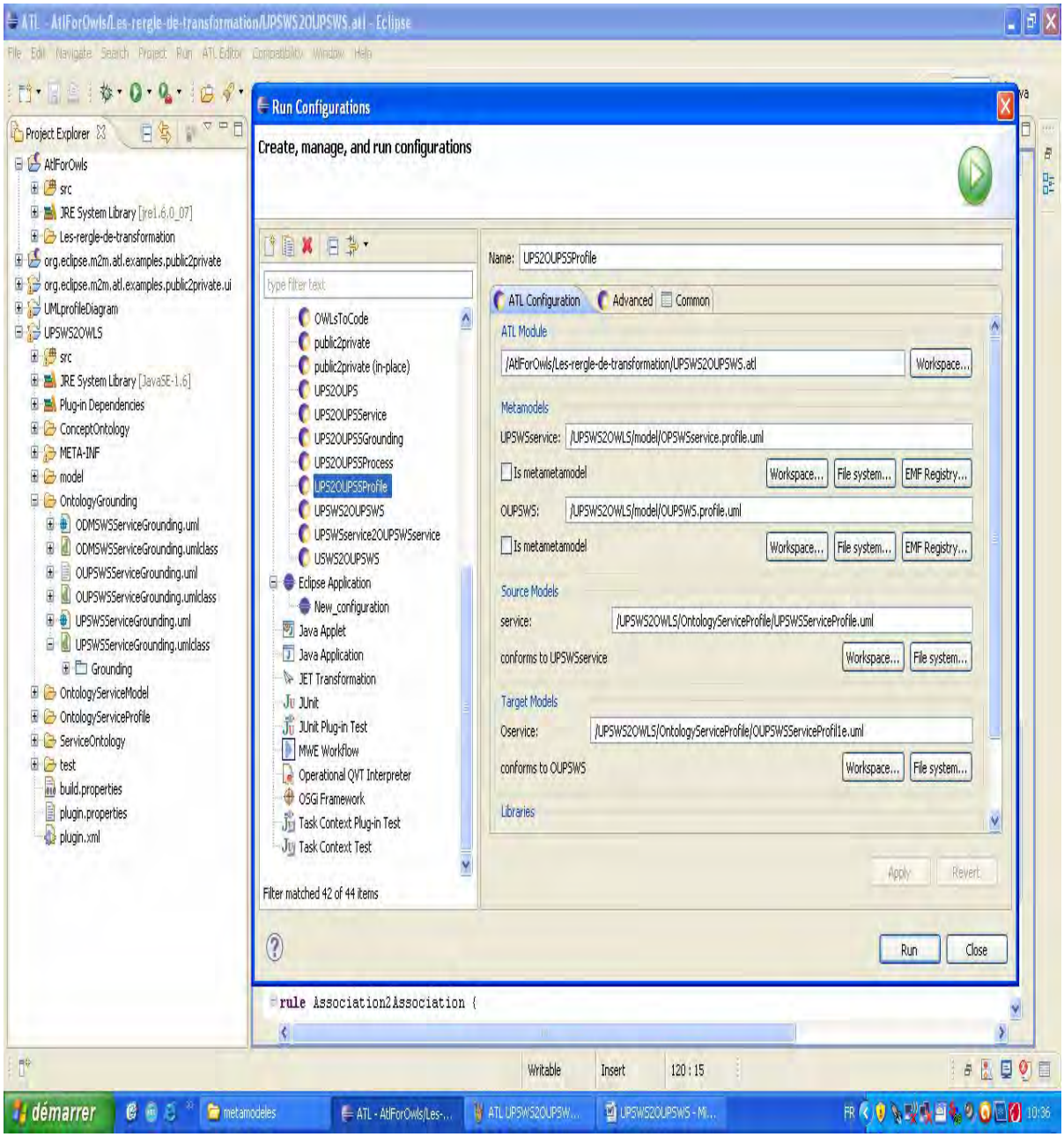

Figure 3.18 : Règles d'exécution ATL de UPSWS vers OUPSWS (partie "profil de service").

L'exécution complète de ces règles nous génère quatre parties OUPSWS et nous présentons seulement la partie qui concerne le profil de service (figure 3.19).

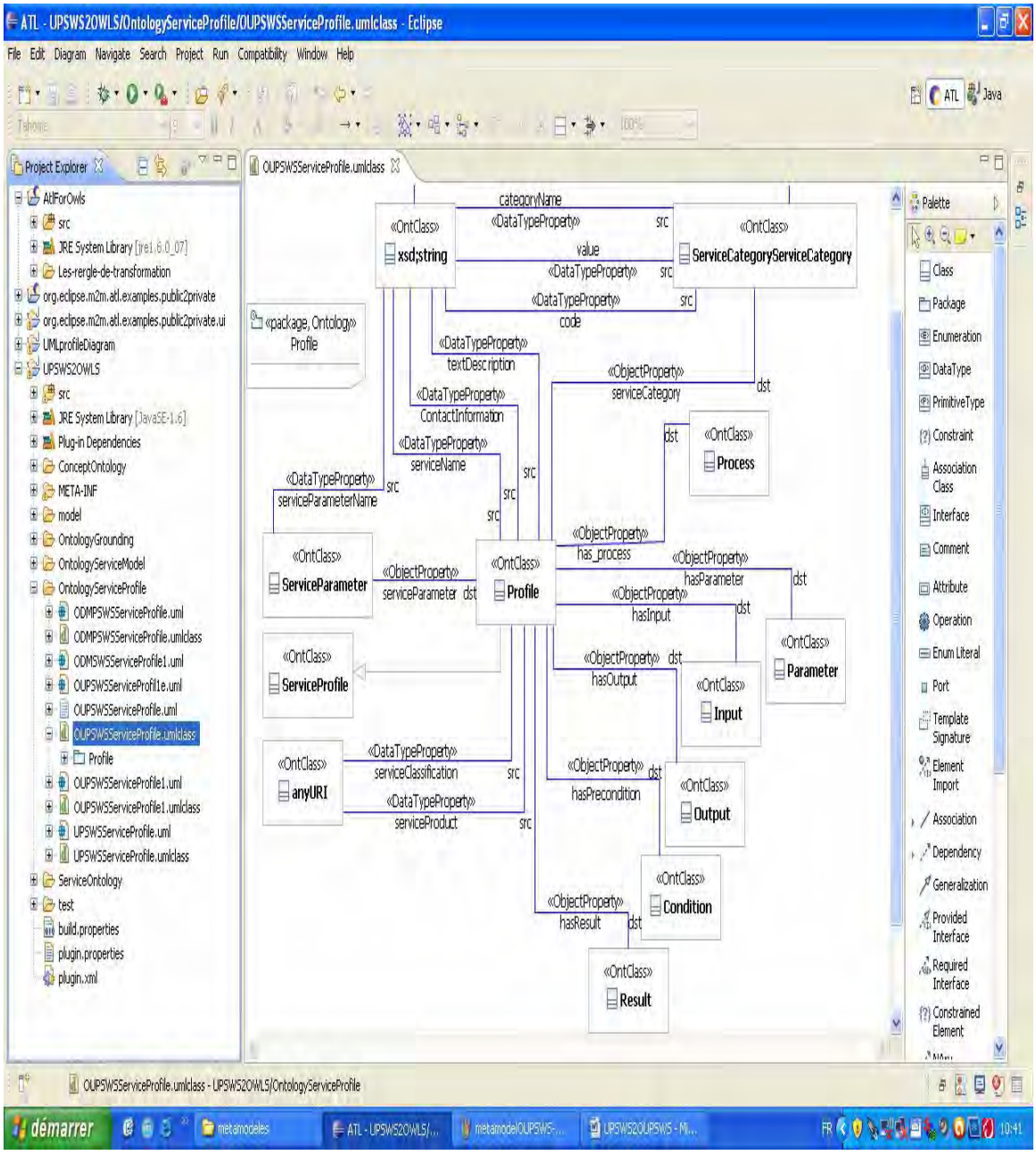

Figure 3.19 : Capture d'écran d'OUPSWS générée (partie "Profil de service").

4.4.3 Transformations OUPSWS vers ODM

Les règles transformations des Profils UML d'ontologies (OUPSWS) vers ODM sont simples du moment que les concepts ODM sont utilisés en tant que des stéréotypes dans OUPSWS. Ici, nous effectuons les transformations définies dans la table 3.3. De même, nous traduisons quatre OUPSWS concernant la partie "service", la partie "profil de service", la partie "processus de service" et la partie "service grounding". La figure suivante montre une capture d'écran des règles ATL d'exécution d'OUPSWS vers ODM.

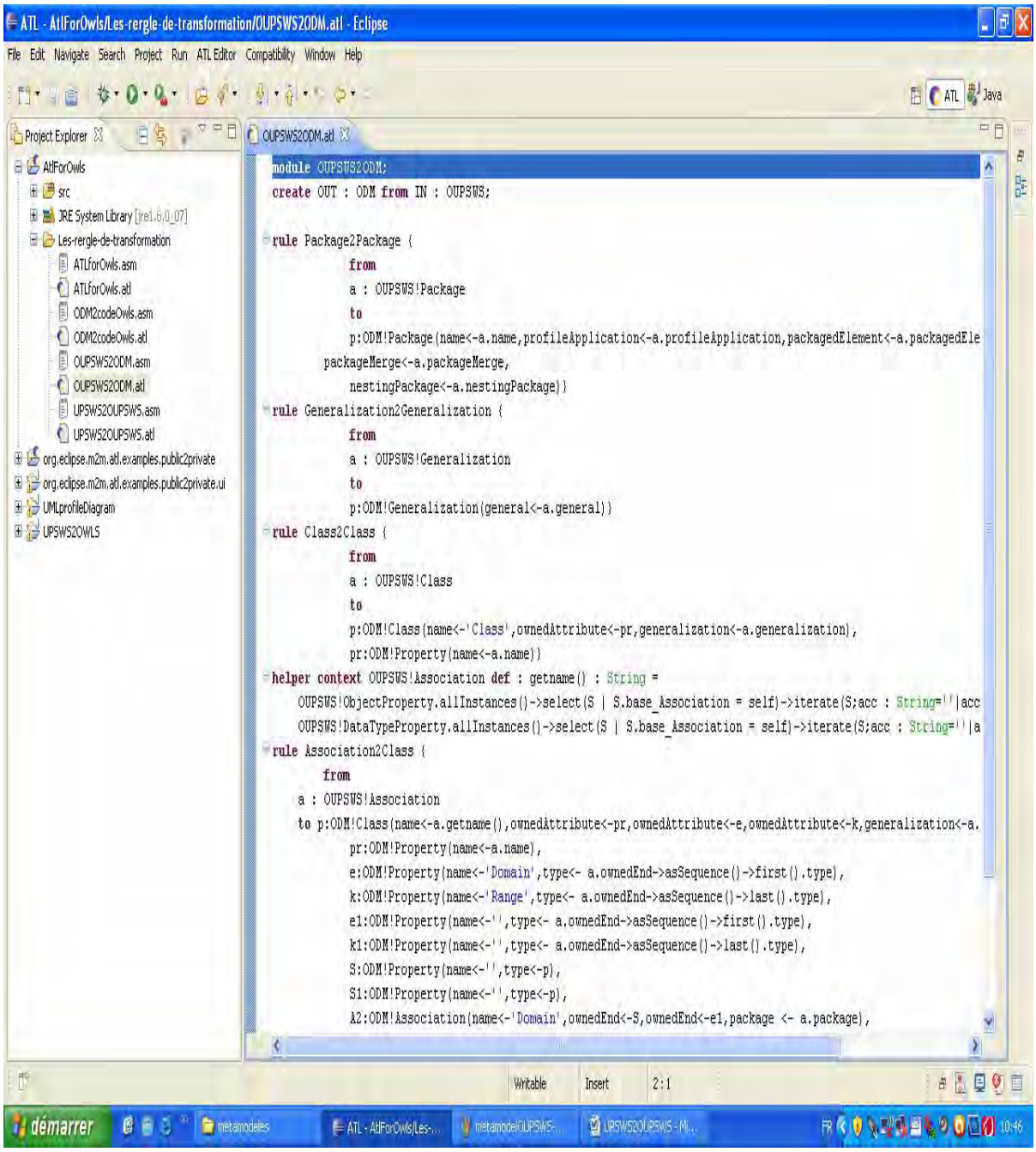

Figure 3.20 : Fenêtre d'exécution des règles ATL de OUPSWS vers ODM.

En sélectionnant par exemple, la partie OUPSWS concernant le processus de service, nous obtenons après exécution des règles correspondantes à cette partie, la partie ODM générée au processus de service (figure 3.21).

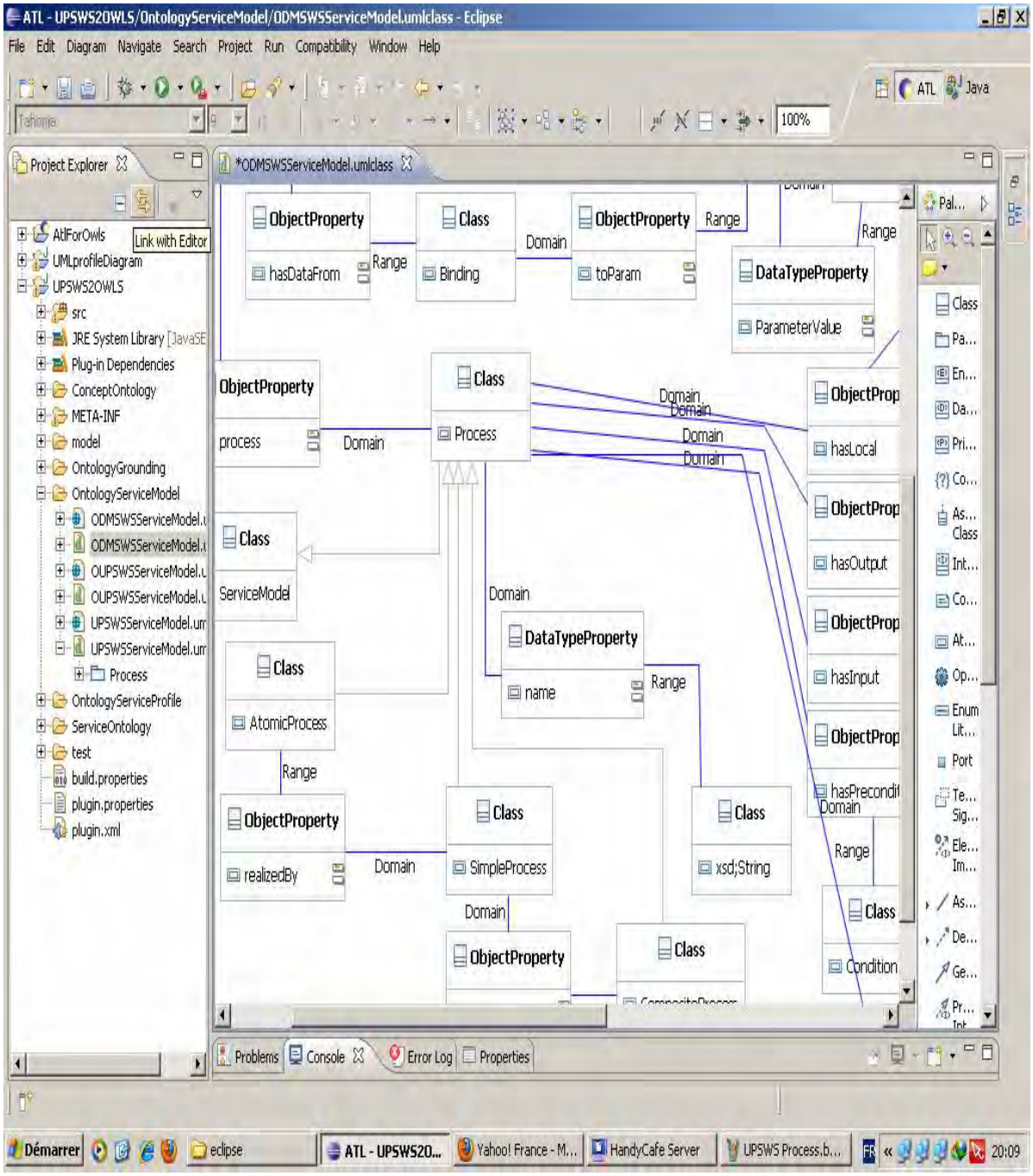

Figure 3.21 : Capture d'écran d'ODM générée (partie Processus de service).

4.4.4 Transformations ODM vers OWL-S

Une fois, les quatre parties ODM sont générées, nous exécutons les règles de transformations correspondante à chaque partie pour générer quatre parties d'OWL-S correspondant au code OWL-S: pour les parties : "service", "profil de service", "processus de service" et "service grounding". Ces règles de transformations sont déjà définies dans la section 4.3.4. La figure suivante montre une capture d'écran des règles ATL d'exécution vers le code OWL-S.

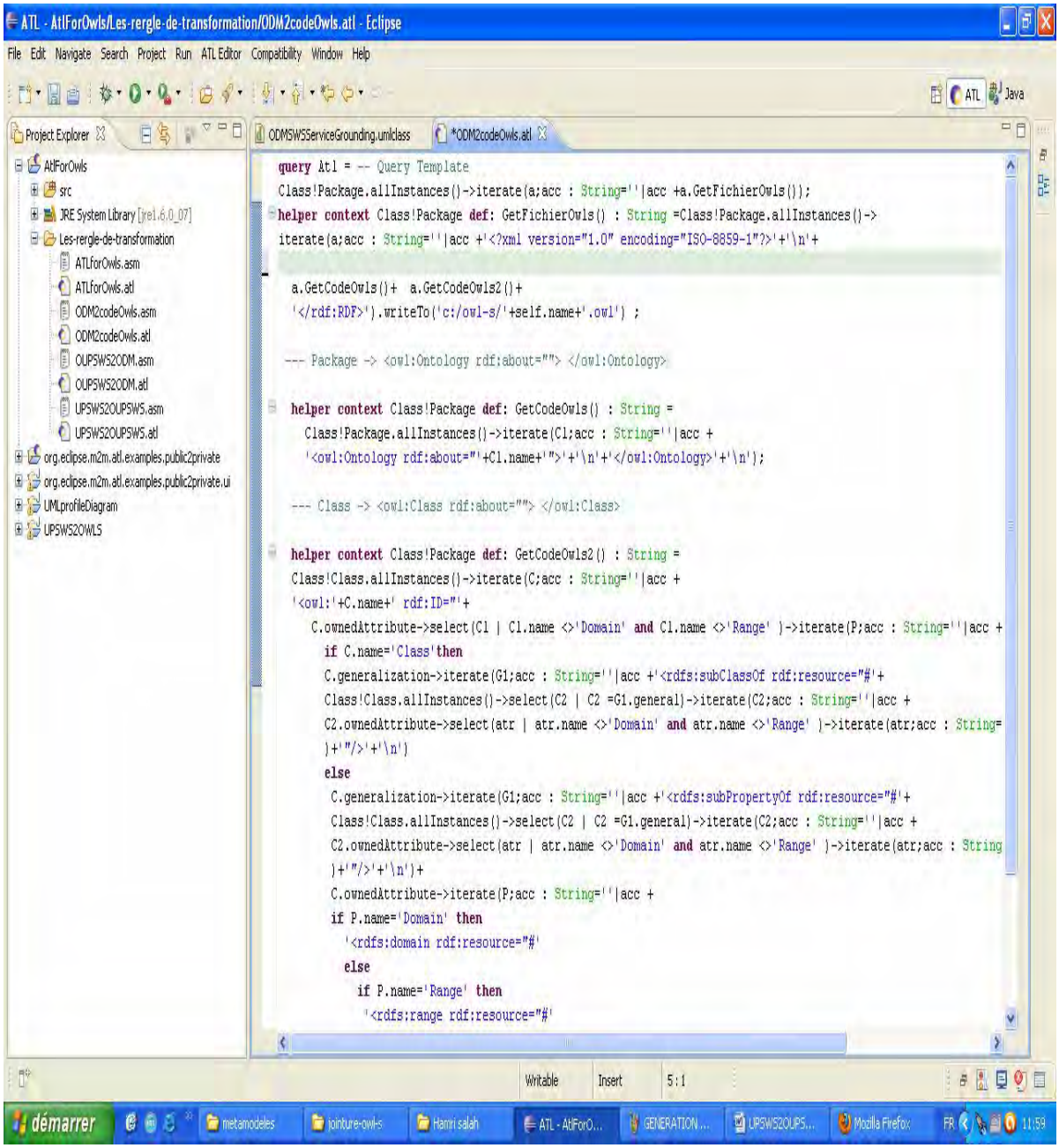

Figure 3.22 : Règles ATL de transformation d'ODM vers OWL-S

Pour exécuter ces règles de transformation, nous sélectionnons dans la partie gauche de l'écran le fichier ATL correspondant à la partie que nous désirons générer. Une fois, les règles ATL sont exécutées pour chaque partie d'ODM, nous obtenons quatre descriptions OWL-S distinctes que nous présentons dans les figures suivantes.

❖ Génération de la partie "service"

En sélectionnant la partie "service", le code OWL-S associé à cette partie est affiché en format XML comme le montre la figure suivante.

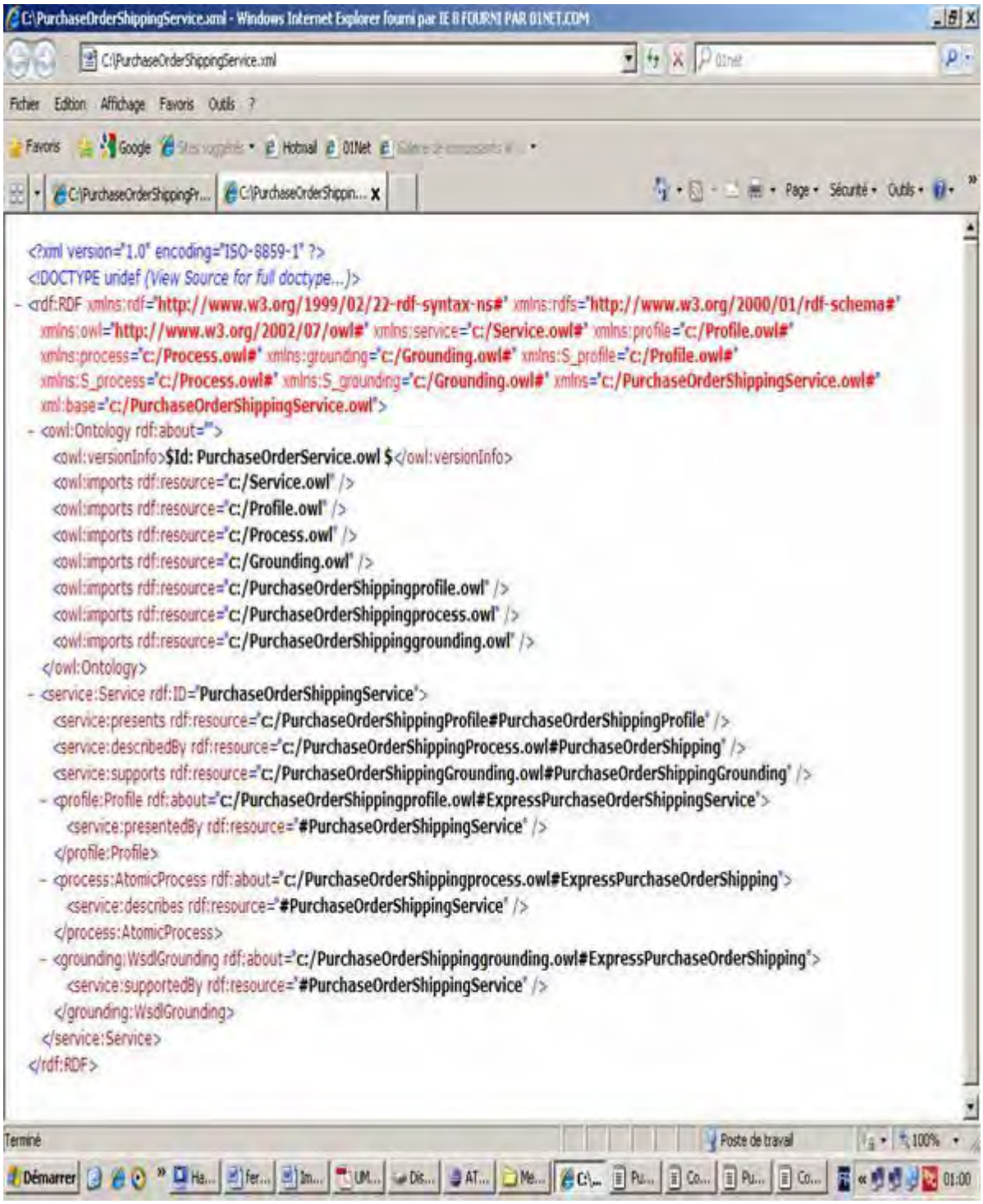

Figure 3.23 **:** Code OWL-S - PurchaseOrderShippingService.owl.

❖ **Génération de la partie "profil de service"**

En sélectionnant les règles ATL pour la partie "profil de service", le code OWL-S associé à cette partie est affiché en format XML comme le montre la figure suivante.

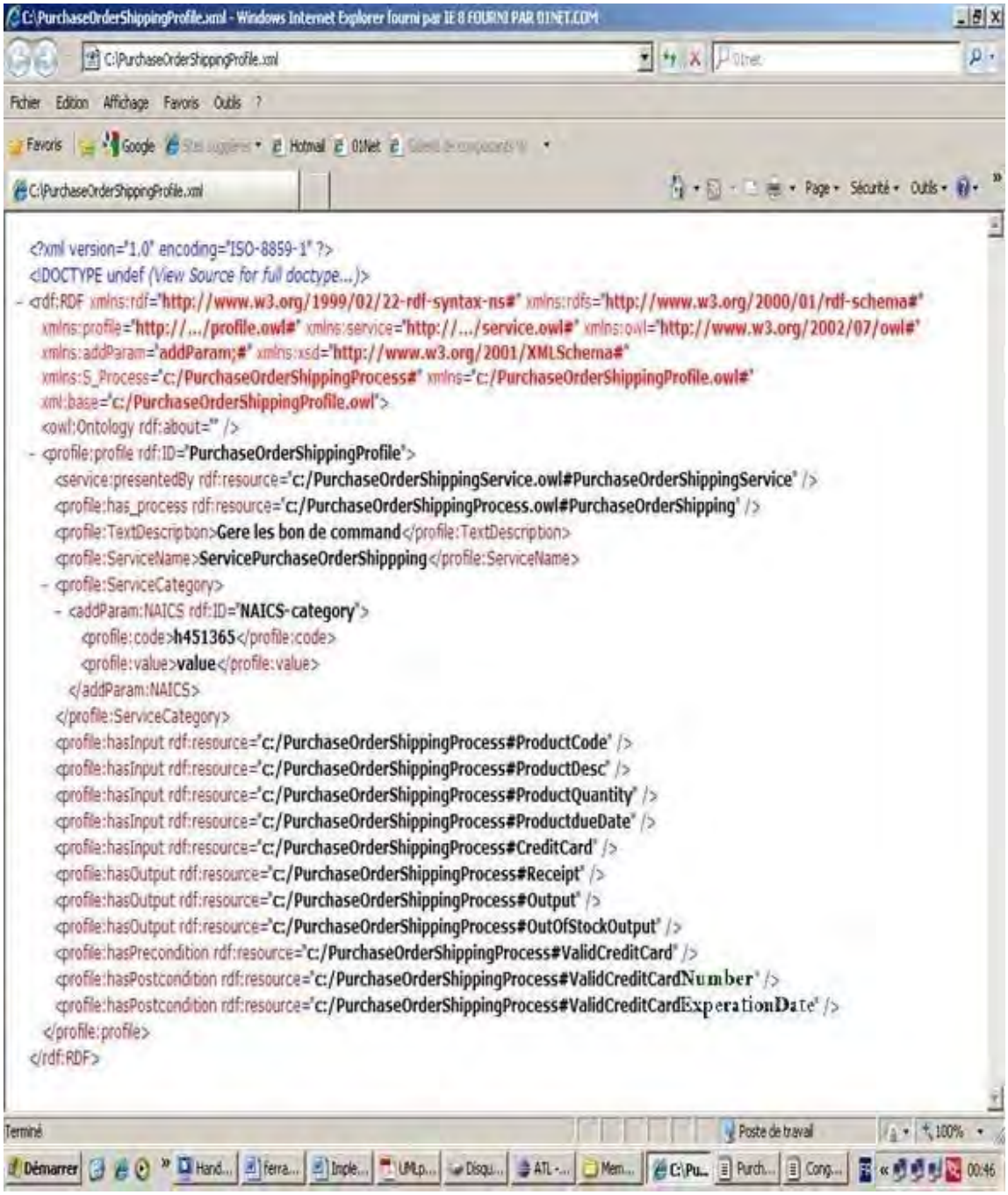

Figure 3.24: Code OWL-S - PurchaseOrderShippingProfil.owl.

❖ **Génération du processus de service**

De même que l'application des règles ATL à la partie "processus de service", nous permet de générer le code OWL-S associé à cette partie comme le montre la figure suivante.

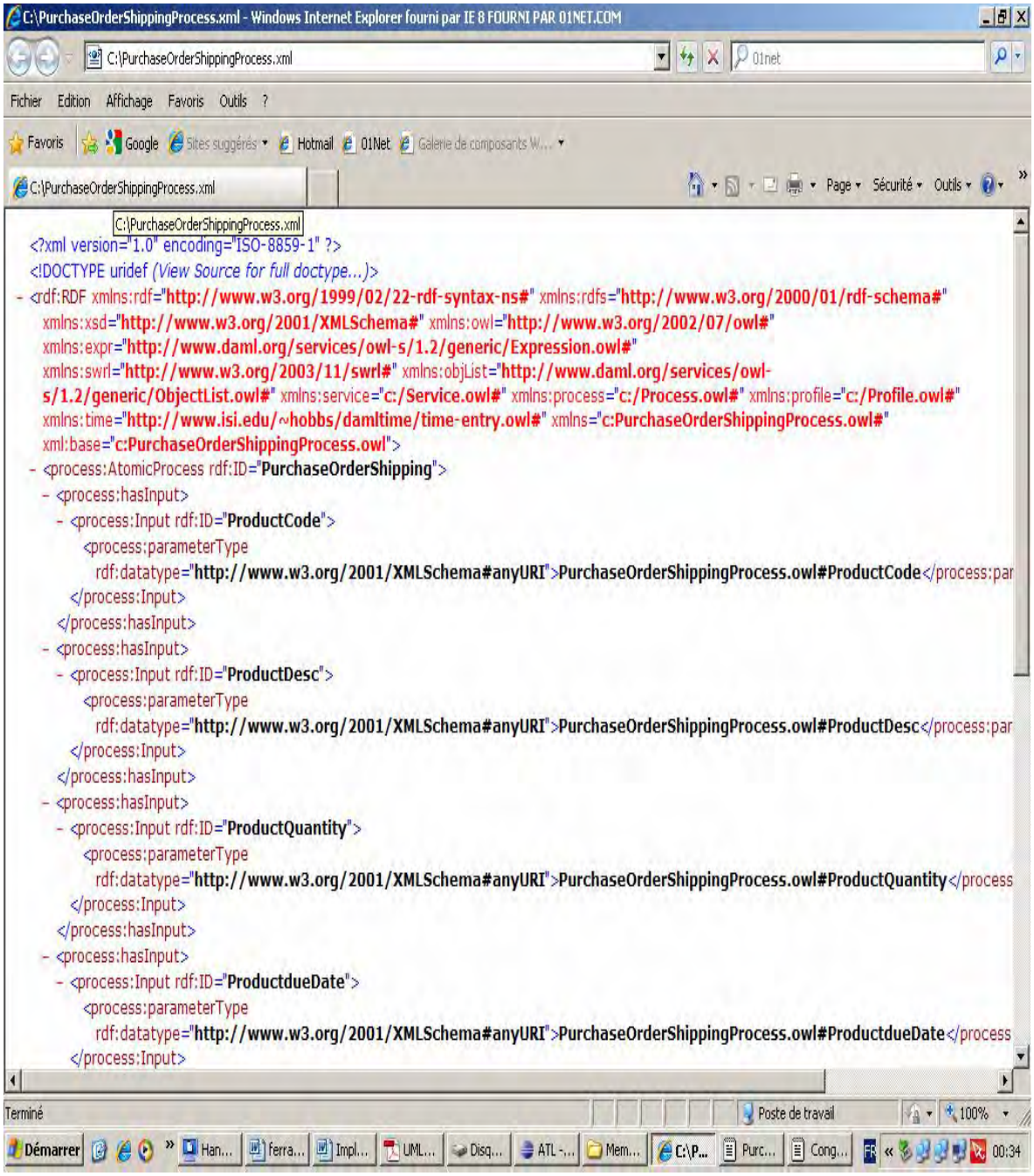

Figure 3.25 : Code OWL-S - PurchaseOrderShippingProcess.owl

❖ Génération du grounding de service

Enfin, la sélection des règles ATL pour la partie " service grounding", nous permet de générer le code OWL-S à cette partie comme le montre la figure suivante.

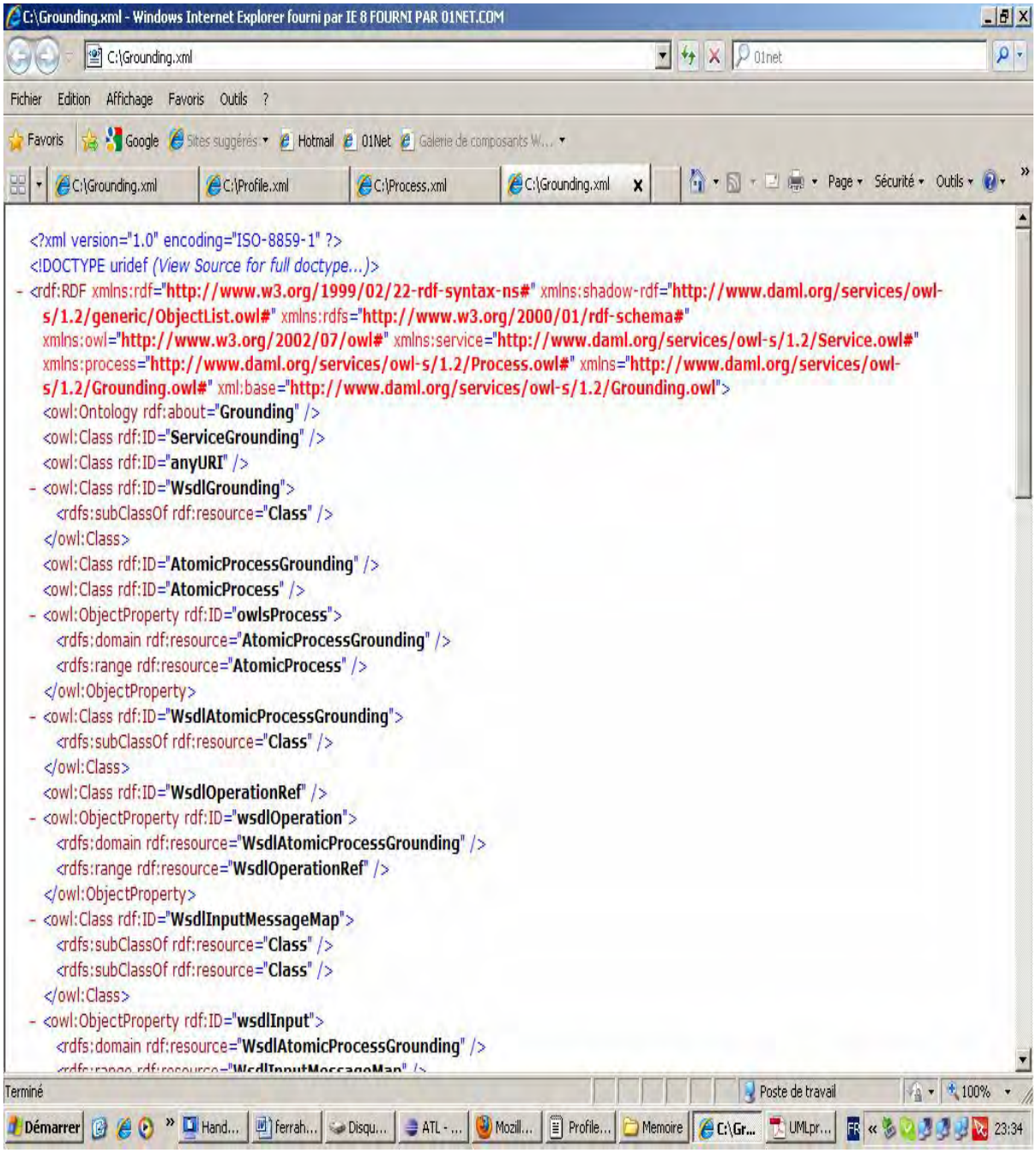

Figure 3.26: Code OWL-S - PurchaseOrderShippingGrounding.owl.

4.4.5 Jointure des trois fichiers OWL-S générés

Dans cette étape on fusionne les quatre fichiers XML générés manuellement, dans un seul fichier et nous l'exportons vers l'outil Protégé pour le valider.

4.4.6 Validation du code de l'ontologie OWL-S

Une fois exporté, nous ouvrons d'abord, le fichier OWL-S avec l'outil Protégé (figure 3.27).

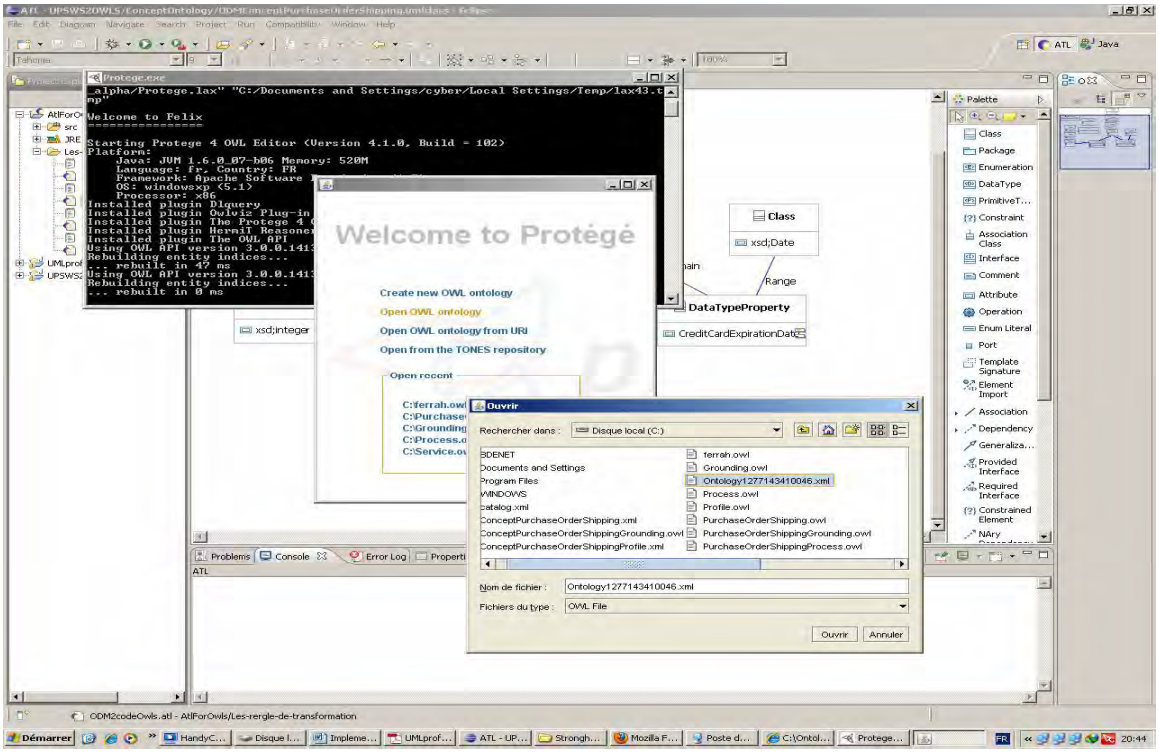

Figure 3.27 : Ouverture de code OWL-S généré avec l'outil Protégé.

Puis, nous explorons le code pour le valider. La validation consiste à vérifier s'il y a des erreurs syntaxiques au niveau de la structure du code comme le montre la figure 3.28.

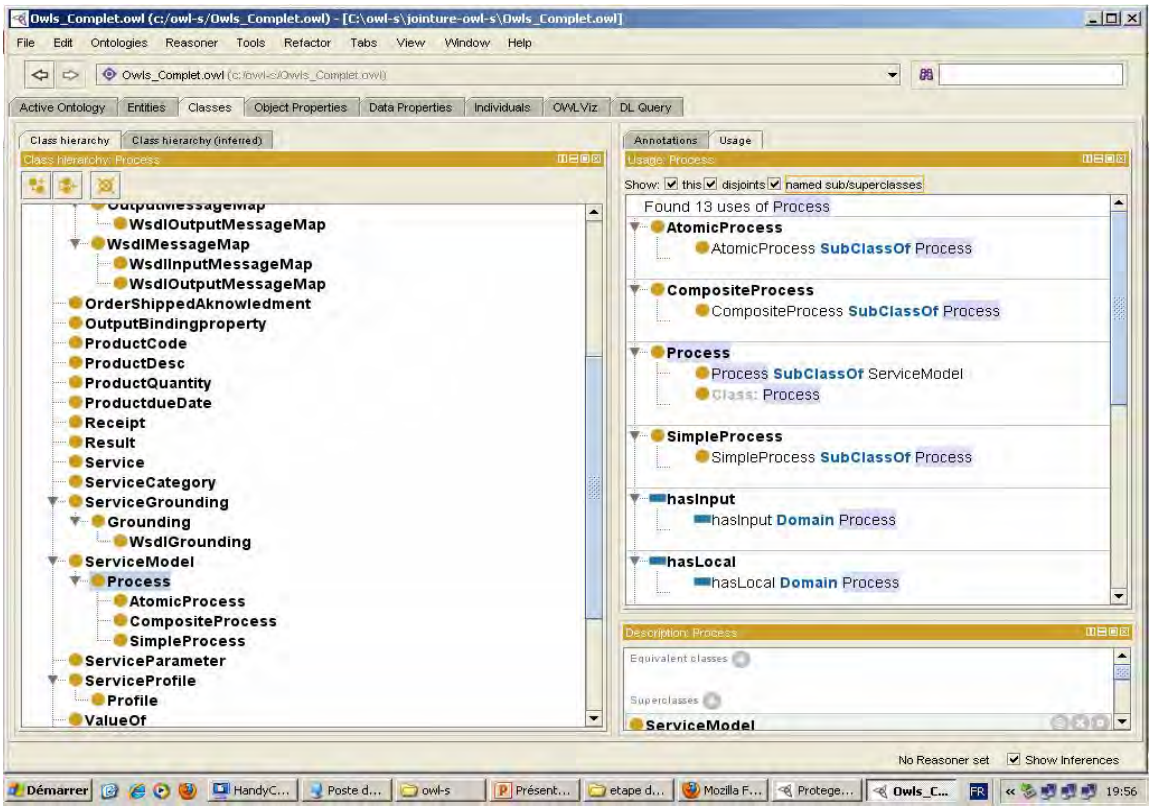

Figure 3.28: Code OWL-S valide dans Protégé.

Ensuite, nous consultons l'ontologie. La figure suivante montre une capture d'écran d'une hiérarchie de classes de l'ontologie OWL-S générée du service Workflow ' Purchase Order Shipping' sous l'éditeur Protégé.

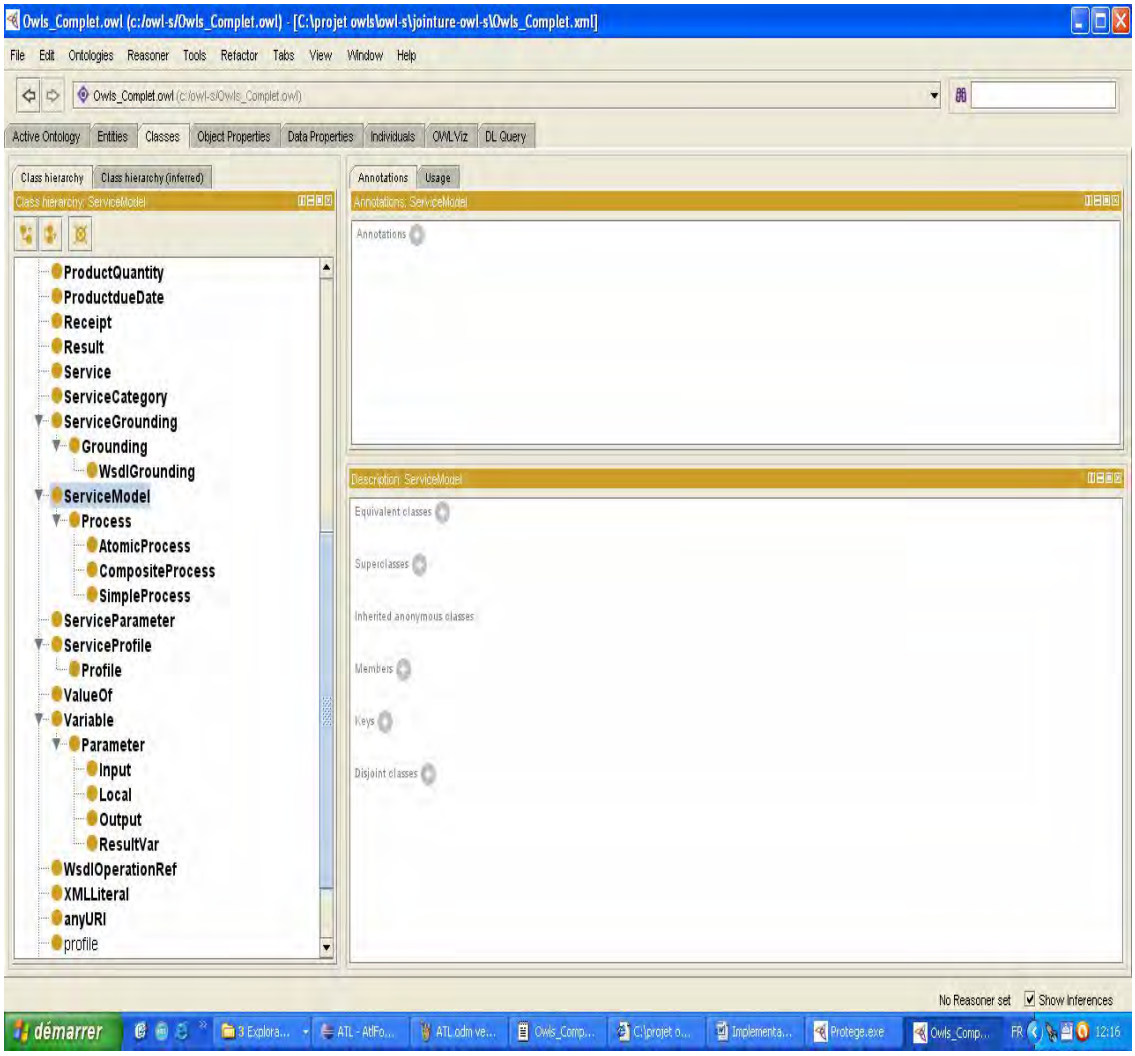

Figure 3.29: Capture d'écran d'une hiérarchie des classes de l'ontologie OWL-S générée.

4.5 Modélisation de la plate forme d'interopérabilité

Pour faire coopérer des acteurs, il faut bâtir une architecture d'interopérabilité permettant aux partenaires d'échanger des modèles de processus. Plusieurs plates-formes existent pour supporter le développement d'applications orientées Internet : J2EE, dotNET et CORBA. Parmi ces plates-formes, nous nous sommes orientés vers celles qui répondent aux critères que nous sommes fixés pour l'interopérabilité des Workflows telles que flexibilité et ouverture. En particulier, nous avons retenu le modèle SOA (Service Oriented Architecture), et plus précisément la plate-forme des Web services qui est devenue l'intergiciel préféré par rapport à ces prédécesseurs et le plus adapté pour l'intégration de systèmes d'information sur Internet.

Beaucoup de travaux ont porté sur l'application de l'approche MDA à la plate-forme des Web services [199], [160] [200]. Parmi ces travaux, certains ont déjà démontré la viabilité de

l'application de MDA aux Web services [201], [202] et considèrent que ces derniers sont une plate-forme très importante aujourd'hui et que c'est la plate-forme cible qui paraît être la plus étudiée et la plus convoitée pour construire des nouvelles applications Internet et pour permettre aux applications du passé (legacy applications) d'être adaptées à l'Internet. [203], [204], [205].

Pour notre cas, nous nous inspirons de ces travaux pour appliquer MDA pour la modélisation de la plateforme des Web services et qui représente pour nous, le PSM (Platform-Specific Model). Quant à notre PIM (Platform-Independent Model), il est représenté par un Profil UML pour Web services (UPSWS).

Dans [200], les auteurs ont étudié la création d'un méta-modèle qui soit le plus précis possible pour les Web services, à partir de schémas ou d'autres spécifications existantes. Ils ont ainsi proposé un méta-modèle pour WSDL [206] à partir de son schéma. La figure suivante montre un extrait de ce méta-modèle.

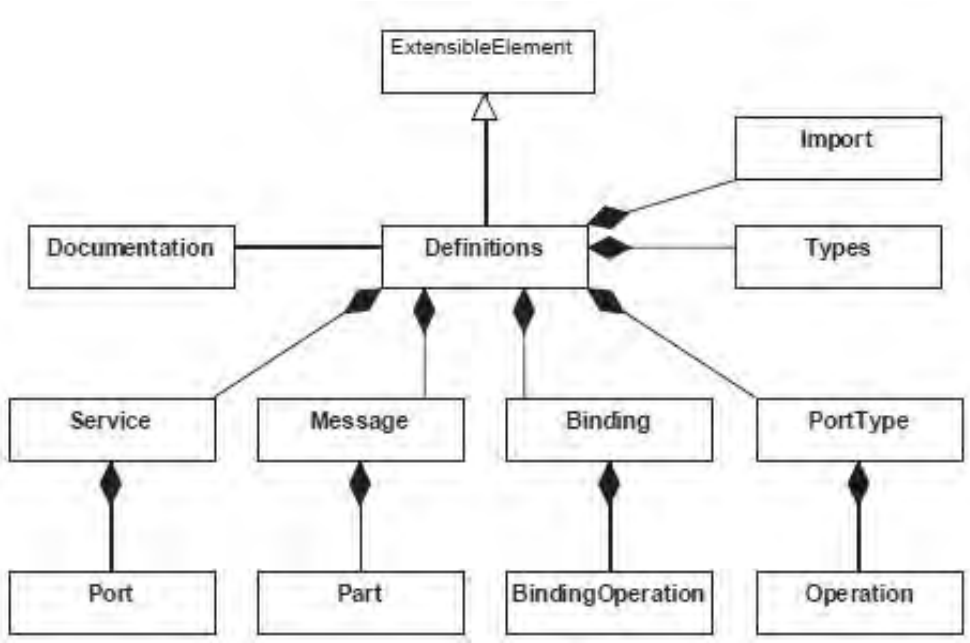

Figure 3.30: Un extrait du méta-modèle de WSDL.

Par contre, dans [202], les auteurs ont présenté un fragment de correspondance entre un modèle UML et WSDL. Une *classe* stéréotypée "BusinessService" en UML correspond au *PortType* en WSDL. Une *méthode* en UML correspond à une *Opération* en WSDL. Un *paramètre d'entrée ou de sortie* en UML correspond à un *Message d'entrée ou de sortie* en WSDL (figure 3.31**).**

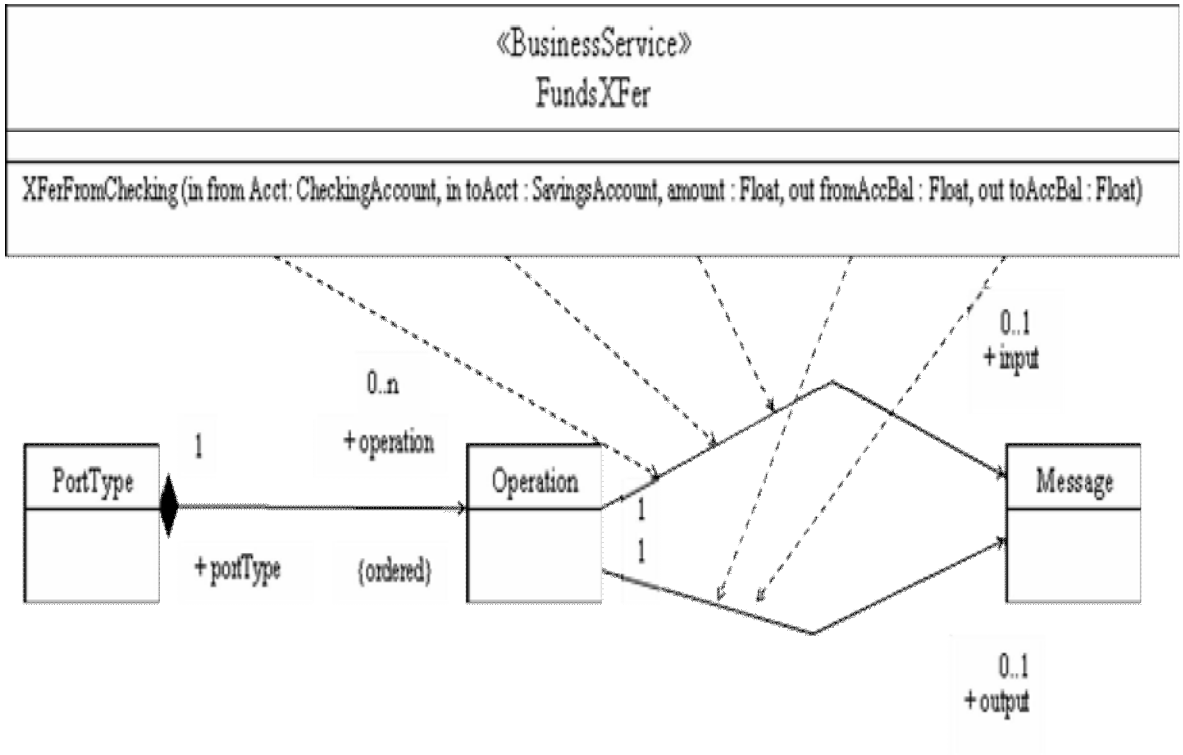

Figure 3.31 : Correspondance entre un modèle métier en UML et WSDL [202].

R. Gronmo et al. [207] ont proposé une méthodologie pour développer des Web services et ont établi une correspondance entre un modèle UML et WSDL. La figure 3.32 présente un modèle UML et son équivalent en WSDL. Selon cette figure, une *Classe* en UML est l'équivalent de Types en WSDL. Une *Interface* en UML est l'équivalent de *PortType* et *Binding* en WSDL. Une *méthode* en UML est l'équivalent d'*Operation* en WSDL. Une *classe* stéréotypée par BusinessServices est l'équivalent de *Service* en WSDL. Ensuite, R. Gronmo et al. ont utilisé UMT (UML Transformation Tool) pour créer les règles de transformation de UML en WSDL [208].

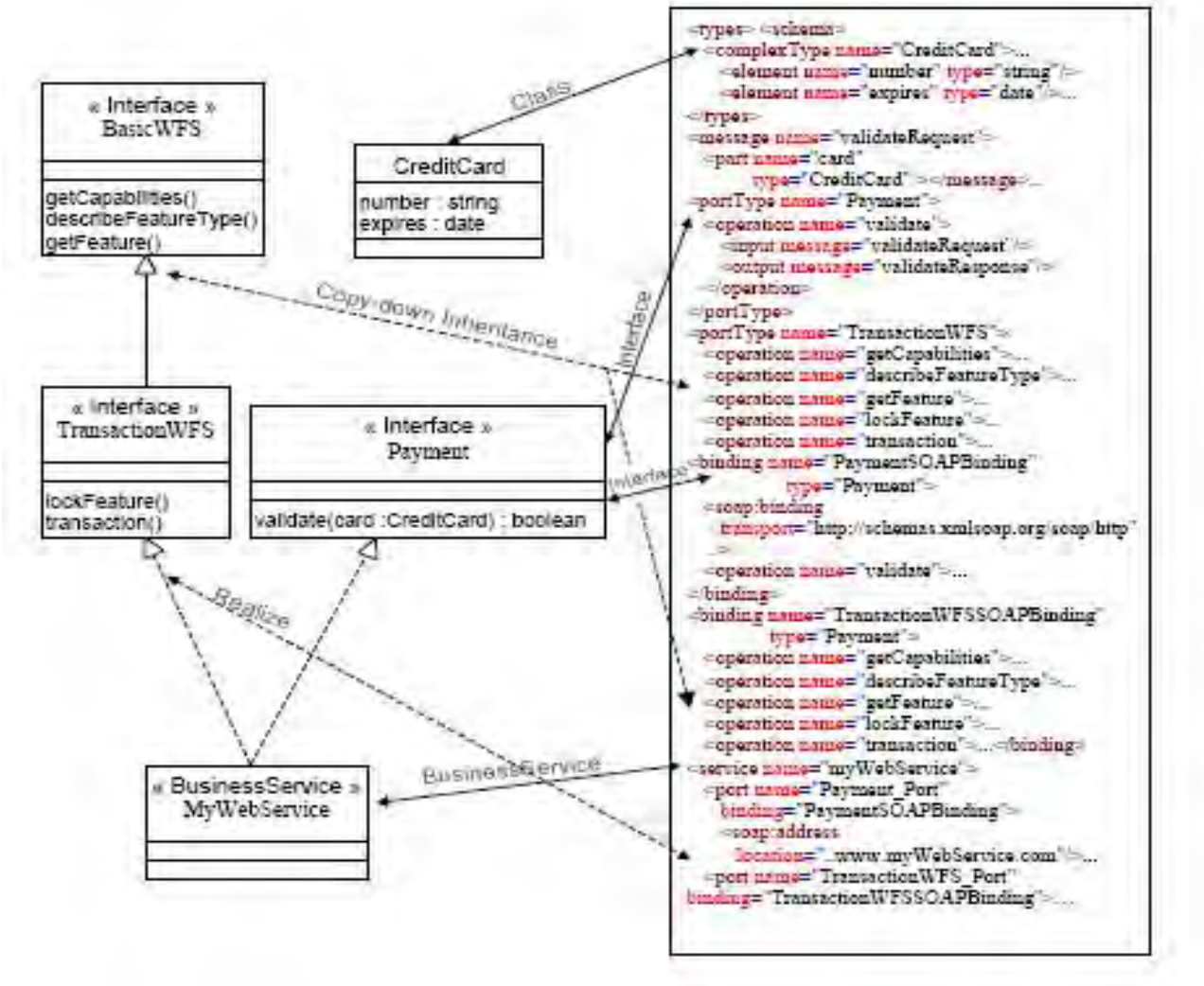

Figure 3.32 : Conversion entre un modèle UML et un document WSDL [208].

❖ **Transformation UPSWS vers WSDL**

Pour notre travail, nous devons convertir un Profil UML pour Web services sémantiques (UPWSS) défini précédemment (considéré comme un PIM) vers le méta-modèle WSDL (considéré comme un PSM). Une *classe* stéréotypée *« Ontology »* en Profil UML (considérée comme un package dans le standard UML) correspond à la *'Definition'* en WSDL. Une *classe* stéréotypée *« Ontoclass »* en Profil UML (considérée comme une classe dans le standard UML) correspond aux *'Types'* en WSDL. Une *classe* stéréotypée *«Semantic Workflow service»* en Profil UML correspond aux *'service'* en WSDL. Une *Methode* (qui est purchaseOrderShipping (ou livrer commande d'achat)) définie dans le Profil UML correspond en WSDL à *"Operation*, *PortType* et *Binding*" Les valeurs étiquetées (tagged values) pour les paramètres d'entrées/sorties (« datatypes») correspondent aux messages d'entrées et sorties en WSDL (ou parties des messages). Quant aux stéréotypes « Effect », « Pre-Condition » et « Post-Condition », ils n'ont aucune équivalence dans le méta-modèle WSDL. De la même manière que les précédentes transformations, nous utilisons le langage de transformation ATL pour réaliser la transformation.

5. Synthèse

Avec l'émergence du paradigme des Web services sémantiques (WSSs), l'interopérabilité sémantique des processus Workflows (ou processus d'affaires) peut être réalisée, et cela en considérant ces processus comme des services Workflows, que nous modélisons d'abord, comme des Web services, puis nous les dotons d'une sémantique en utilisant le concept d'ontologie qui est considéré comme un des moyens le plus favorable pour décrire la sémantique de ces services.

Par conséquent, pour aborder le problème conceptuel de l'interopérabilité des processus Workflows, nous avons proposé une méthodologie basée sur l'intégration de trois concepts : MDA, Web sémantique et les Web services sémantiques. Cette intégration s'articule principalement autour de la construction d'une architecture MDA basée sur des ontologies pour la construction de différents méta-modèles à savoir :

- Un Profil UML pour Web services sémantiques (UPSWS) qui est considéré en tant que modèle indépendant de la plate-forme (PIM de niveau 0 ou PIM0) ;

- Un Profil UML d'ontologie pour Web services sémantiques (OUPSWS) qui est considéré comme un modèle indépendant de la plate-forme (PIM de niveau 1 ou PIM1) ***;***

- Un méta-modèle de définition d'ontologie (ODM) qui est considéré comme un modèle indépendant de la plate-forme (PIM de niveau 2 ou PIM2) ;

- Un méta-modèle OWL et un méta-modèle OWL-S, qui sont considérés comme des modèles spécifiques à la plate-forme (PSMs), vont nous permettre de nous projeter vers le contexte du Web sémantique ou du Web service sémantique pour générer respectivement des descriptions OWL ou OWL-S.

Cette architecture nous permet ainsi, de générer une ontologie de services dans n'importe quel langage de représentation d'ontologies basé Web tels que OWL ou OWL-S. Parmi les avantages que présente cette méthodologie basée MDA, nous pouvons citer principalement les points suivants :

- Elle permet aux développeurs d'utiliser simplement un outil UML standard pour modéliser la sémantique et de générer des descriptions OWL-S. Par conséquent, les barrières pour l'adoption de la technologie des Web services sémantiques peuvent être surmontées grâce aux concepts de MDA.

- L'architecture ainsi bâtie, nous fournit une passerelle entre le génie logiciel et l'ingénierie ontologique par l'utilisation du méta-modèle de définition d'ontologie ODM pour décrire la sémantique de manière conceptuelle, indépendamment de tout langage ontologique.

- L'approche proposée est flexible et ouverte. En effet, elle permet de nous projeter vers le Web sémantique ou la technologie des Web services sémantiques et de supporter n'importe quel langage ontologique de service (par exemple, WSML ou WSDL-S) ou autre, en utilisant simplement le principe d'ODM.

- De plus, ODM sert comme point d'intégration permettant de réduire considérablement le nombre de mappings. En effet, en cas d'utilisation d'un méta-modèle passerelle, c'est-à-dire, ODM, il nous faut 2*N mappings pour mapper N méta-modèles d'ontologies de services. Par contre, en cas de non existence de ce méta-modèle, il faut définir N*(N-1) transformations.

- Finalement, notre méthodologie est conforme au principe de MDA basé sur des transformations entre les PIMs et les PSMs pour générer du code. En conclusion, nous pensons que les obstacles pour adopter les techniques basées sur l'utilisation des Web services sémantiques, peuvent être surmontés afin d'accroître les compétences des développeurs et de faciliter leur adoption par les utilisateurs. La passerelle devrait être ainsi, créée en utilisant les outils MDA pour les utilisateurs déjà familiers à ces outils. Cependant, nous pensons que l'approche conceptuelle que nous avons proposée pour générer pour la description sémantique des Web services s'applique à n'importe quel méta-modèle cible d'ontologies de services tels que WSDL-S ou WSML.

De même, pour aborder le problème conceptuel lié à la plate-forme technique, nous avons choisi la plate-forme qui présente beaucoup d'avantages en termes de flexibilité et d'ouverture, et qui est pour nous, la plate-forme des Web services.

6. Conclusion

Pour résoudre la problématique conceptuelle de l'interopérabilité des modèles de Workflow, nous avons proposé une première approche basée les techniques de méta-modélisation et de transformation de modèles qui nous procurent un haut niveau d'abstraction pour aborder :

- La modélisation des ontologies afin de capturer l'aspect sémantique des entités des modèles de processus indépendamment des langages de représentation d'ontologies basés Web.

- La modélisation de la plate-forme technique qui sera utilisée pour la coopération entre des acteurs coopératifs.

Aussi, nous avons tenu compte des objectifs fixés dans la thèse pour la génération d'une ontologie de services. En effet, le choix du méta-modèle OWL-S permet d'apporter plus de flexibilité par rapport aux autres méta-modèles de représentation d'ontologies, tandis que le choix de la plate-forme des Web services est principalement motivé par le fait qu'ils constituent une plate-forme flexible basée sur des standards industriels.

Enfin, nous pouvons dire que l'utilisation conjointe de l'approche MDA, SOA et ontologies dans notre approche conceptuelle peut favoriser considérablement la flexibilité, tout en apportant une solution au problème conceptuel de l'interopérabilité des modèles de Workflow dans les milieux industriels.

Chapitre IV

Utilisation des Techniques d'Annotation Sémantique

« Modéliser, c'est raisonner »,

Paraphrase d'Herbert A. Simon

Notre deuxième conviction dans cette thèse est :

« Pour interopérer, modéliser, annoter, puis formaliser pour raisonner»

1. Introduction

Pour résoudre la problématique conceptuelle de l'interopérabilité des modèles de Workflow (ou modèles de processus), nous proposons cette fois-ci, une autre approche basée sur les techniques d'annotation sémantique. Elle préconise alors d'explorer le concept d'annotation sémantique, déjà utilisé pour les documents, les articles, les pages Web, etc., pour l'étendre vers les modèles de processus.

Plusieurs types d'annotations existent : annotation textuelle, annotations de lien et annotation sémantique. Ces annotations peuvent être attachées aux modèles selon les utilisations possibles ou les besoins de l'interopérabilité. Nous distinguons les annotations textuelles qui sont destinées à des humains et les annotations sémantiques pour les machines. Le principe d'annotation sémantique des modèles consiste alors à rajouter aux entités d'un modèle des informations (annotations), puis définir des liens avec des ontologies.

Une annotation sémantique est donc une information supplémentaire dans un document qui définit la sémantique d'une partie de ce document. Dans notre contexte, les annotations sémantiques sont des éléments d'informations complémentaires ajoutés dans des modèles de processus, que nous considérons comme des artefacts (ou documents). Elles définissent la sémantique en se référant à un modèle sémantique qui décrit les concepts et les termes utilisés dans un domaine particulier de connaissances (domaine d'intérêt) que nous appelons simplement "ontologie du domaine de référence".

Ces annotations sont produites par le fournisseur d'annotations afin qu'elles soient compréhensibles par le consommateur d'annotations (humains ou machines). Dans notre contexte d'échange et de coopération, l'usage des annotations sémantiques nous est d'un grand apport pour la compréhension commune des modèles interopérables. Aussi, le lien entre ces annotations et les ontologies permet d'améliorer la lisibilité et l'interprétation de ces modèles et éviter ainsi, les ambiguïtés de compréhension par des agents coopératifs (humains ou machines).

Cependant, pour échanger des modèles de processus ou coopérer, deux alternatives se présentent pour décrire un processus d'annotation. Ce dernier dépend de l'environnement dans lequel les partenaires coopèrent : homogène ou hétérogène. Dans la section 2, nous présentons l'interopérabilité dans un contexte homogène, tandis que la section 3, elle est consacrée au contexte hétérogène.

2. Interopérabilité dans un environnement homogène

L'interopérabilité sémantique dans un contexte homogène, ne constitue vraiment pas un problème crucial pour les entreprises coopératives. Néanmoins, pour éviter des ambiguïtés et de mauvaises interprétations et avoir une bonne compréhension des modèles lors de la coopération, il est préférable de les annoter.

Dans cet environnement, les partenaires (ou acteurs) coopératifs se connaissent, partagent le même domaine de connaissances (même domaine de référence, par exemple, le secteur automobile). Donc, ils doivent s'entendre sur la description sémantique des annotations, ainsi que sur les différents types d'annotations qui peuvent être ainsi élaborées en commun. De ce fait, l'ontologie du domaine de référence est connue. Ainsi, deux acteurs échangent leurs

modèles *M1* et *M2,* en utilisant les mêmes notations pour leurs modèles et se réfèrent à un même méta-modèle *MM1* ou *MM2* (*MM1* identique à *MM2*) et à une même ontologie de processus *O1* ou *O2* (*O1* identique à *O2*). Dans ce contexte homogène, l'annotation lexicale décrite par un acteur *Act1* d'un modèle *M1* suffit à l'acteur *Act2* d'interpréter le modèle reçu.

2.1 Principe de l'approche

Pour illustrer notre approche basée sur les techniques d'annotation, nous nous servons dans la suite du document, d'un exemple simple d'un modèle de processus (ou fragment de modèle) telle qu'une activité de commande d'achat, qu'il s'agit d'annoter.

La figure 4.1 présente un exemple d'annotation de type structurel d'un fragment de modèle (activité d'une commande d'achat) qui peut être échangé par des acteurs dans un contexte homogène, en utilisant les mêmes ontologies (de processus, banque et finances, puis livraison). Quant à l'ontologie du domaine, elle est connue par les acteurs de l'interopérabilité. L'annotation utilisée est de type structurel correspondant à la notation UML et fait référence aux concepts définis dans ces ontologies.

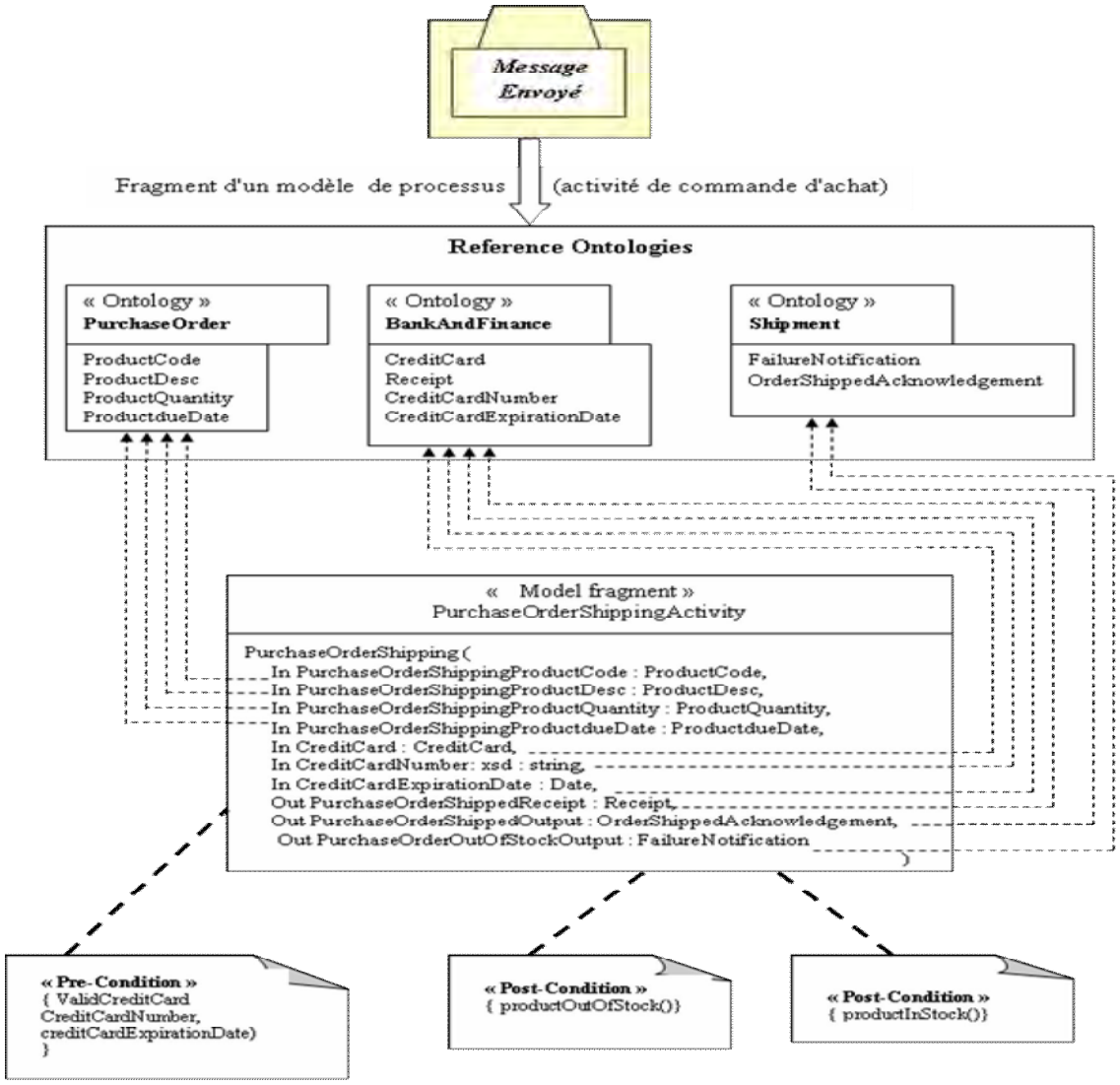

Figure 4.1 : Annotation sémantique structurelle d'un fragment de modèle (activité).

Plusieurs types d'annotations ont été citées dans la littérature [210] [211], [212] [213] pour annoter les processus: de décoration, de liaison, d'identification d'instance, de profil, de but, etc. Dans notre travail, nous employons des annotations :

- Structurelles : une annotation est dite 'structurelle', si l'information de l'annotation est liée à la structure du modèle ou à une partie du modèle. Ces annotations sont dites de collaboration. Dans un contexte homogène, la notation UML (Unified Modeling Language) suffit pour la compréhension des modèles lors de l'échange entre deux acteurs coopératifs. Elles sont destinées à annoter un diagramme de classe UML représentant le modèle de processus. Ces annotations correspondent donc, à des notes (commentaires) explicatives attachées à un élément d'un modèle qui peut être un attribut (ou propriété), une méthode (appelée parfois, fonction ou opération). Elles nous fournissent l'information supplémentaire sur les éléments. Elles sont exprimées en langage textuel et sont liées aux éléments annotés du modèle par l'ancre.

  Dans notre travail, ces annotations structurelles sont attachées aux différents concepts qui sont utilisés par les langages de modélisation de processus tels que XPDL, BPML ou ebXML, et se réfèrent à des ontologies identiques ou à des ontologies qui s'avèrent ne sont totalement identiques. Elles sont liées aux modèles de processus et aux différents aspects de base de type fonctionnel, organisationnel, informationnel comportemental et de ressources.

- Terminologiques/lexicales : Elles référent à une liste de termes qui sont utilisés dans notre domaine Workflow, et qui sont reliés entre eux par des relations synonymiques, hiérarchiques et associatives. Dans ce contexte homogène, les annotations lexicales que nous employons se réfèrent à un glossaire tel que celui de la WfMC (Workflow Management Coalition) [36].

Cependant, pour résoudre l'hétérogénéité sémantique de ces concepts, nous proposons une structure d'annotation sémantique commune. Celle-ci est basée sur l'élaboration d'un méta-modèle commun d'annotation sémantique des modèles de processus (MCASMP) que nous décrivons dans la section suivante.

2.2 Le méta-modèle commun d'annotation sémantique des modèles de processus

Dans un contexte homogène, les ontologies de processus des différents acteurs d'interopérabilité sont identiques. Alors pour annoter leurs méta-modèles de processus *MM1* et *MM2,* les acteurs *Act1* et *Act2* utilisent les concepts d'une même ontologie de processus qui est *O1* ou *O2* (*O1* identique à *O2*) comme des méta-données pour annoter la sémantique des concepts de leurs méta-modèles de processus.

La figure 4.2 montre un scénario d'annotation sémantique des modèles de processus où l'annotateur annote les méta-modèles *MM1* et *MM2* des acteurs *Act1* et *Act2* en utilisant les concepts d'une même ontologie de processus qui est *O1* ou *O2*. Dans ce cas, pour élaborer le méta-modèle commun d'annotation sémantique des modèles de processus (MCASMP), nous utilisons alors les concepts de l'ontologie *O1* ou de l'ontologie *O2* (*O1* identique à *O2*) comme le montre la figure 4.2.

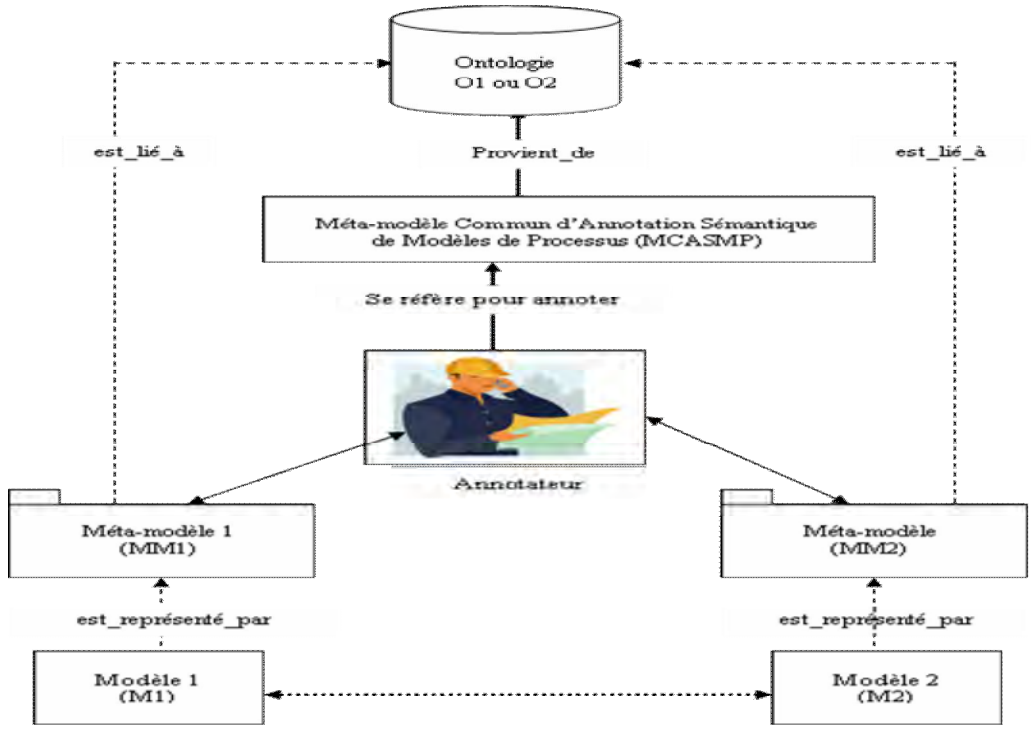

Figure 4.2 : Scénario d'annotation sémantique des modèles de processus.

Le méta-modèle commun d'annotation sémantique des modèles de processus (MCASMP) est donc, défini par un ensemble de concepts (activités, rôles, acteurs, etc.) provenant de l'ontologie *O1* ou O2. La formalisation du MCASMP est la suivante :

MCASMP = (AV, AR, AC, AF, AE, AS, AT, RS)

où : AV est un ensemble d'activités composant le processus et AVi est une activité;
AR est un ensemble de rôles qui interagissent dans le processus et ARi est un rôle ;
AC est un ensemble d'acteurs qui participent dans le processus et ACi est un acteur ;
AF est un ensemble d'artefacts qui sont utilisés dans le processus et AFi est un artefact ;
AE est un ensemble de paramètres d'entrées (inputs) et AEi représente les inputs de l'activité AVi
AS est un ensemble de paramètres de sorties (outputs) et ASi représente les outputs de l'activité AVi ;
AT définit les conditions de transition pour l'exécution du processus et ATi dénote les conditions de transition de l'activité AVi ;
RS est un ensemble de ressources qui sont invoquées pendant l'exécution du processus et RSi dénote les ressources utilisées par l'activité AVi ;

- Une activité est une étape d'un processus (manuelle ou automatisée) qui peut être atomique (ou composite). Elle est réalisée par un rôle et exécutée via les conditions de transition. Une activité quelconque (AVi) est alors décrite comme suit :

AVi = (id, name, has_Role, has_Actor, has_Artifact, has_Input, has_Output, has_Transition, has_Resource).

Chaque élément dans le MCASMP a un identificateur (*id)* et un nom (*name)* pour identifier de manière unique l'élément. Les relations (*has_Role, has_Actor, has_Artifact, has_Input,*

has_Output, has_Transition, has_Resource, sont des relations sémantiques qui sont définies dans l'ontologie O1 ou O2 entre le concept d'activité 'Activity' et les autres concepts tels que "Resource', 'Artifact'. Ces concepts sont dénotés par des URI (Uniform Resource Identifier) dans le MCASMP.

- Un rôle est un concept organisationnel qui interagit avec l'activité (unité organisationnelle, équipe, manager, superviseur, etc.). Un rôle est alors décrit comme suit :

  ARi = (id, name, performs).

- Un acteur est une personne, un agent, un utilisateur qui participe à l'exécution de l'activité ou du processus. Nous pouvons le décrire comme suit :

  ACi = (id, name, plays).

- Un artefact est un objet (document, feuille de style, formulaire électronique, etc.) qui est manipulé par une activité ou un processus. Il est décrit comme suit :

  AFi = (id, name, related-activity).

- Les entrées (inputs) et les sorties (outputs) sont définies comme des paramètres d'entrées et de sorties d'une activité comportant des types de données. Elles sont souvent reliées à des artefacts manipulés par les activités. Nous pouvons les décrire comme suit :

  AEi = (id, name, data-type, related-artifact).
  ASi = (id, name, data-type, related-artifact).

- Contrairement à une entité organisationnelle, une ressource est une entité physique possédant une matérialisation au sein d'une organisation qui est consommable et pouvant être demandée par une activité pour son exécution. Nous pouvons la décrire comme suit :

  RSi = (id, name, related-activity).

- Une condition de transition est représentée par des expressions pour contraindre les inputs et les outputs qui sont définis comme des paramètres de l'activité (pré-conditions et post-conditions). Les conditions de transition permettent de spécifier le contrôle de flux qui s'exprime à travers les connecteurs AND, OR, XOR qui sont des opérateurs logiques de base utilisés par la plupart des processus Workflows [215], [213]. Ce contrôle de flux est défini globalement sous forme de séquences, de mise en parallèle et de branchement. Nous pouvons annoter la transition comme suit :

  ATi = (id, name, related_activity).

2.3 Annotation sémantique des méta-modèles de processus

Dans un contexte homogène, pour interopérer deux solutions peuvent se présenter pour annoter sémantiquement les méta-modèles de processus :

1. Les acteurs coopératifs Act1 et Act2 d'une alliance ne se méfient pas de leurs partenaires et partagent une même ontologie de processus O1 ou O2 (O1 identique à O2). Dans ce cas, l'annotation sémantique des méta-modèles de processus consiste alors à utiliser les concepts de l'ontologie O1 ou O2 comme des méta-données pour annoter la sémantique des concepts des différents méta-modèles de processus. Dans ce cas, les concepts du MCASMP proviennent uniquement de l'ontologie O1 ou O2.

2. Cependant, parfois même dans un contexte homogène, on se confronte à des partenaires d'une alliance qui partagent de manière implicite une ontologie du domaine de référence (ou secteur d'activité), mais qui se méfient de leurs partenaires et veillent à ne pas dévoiler leurs ontologies de processus par mesure de protection de leur expérience et technique de travail. Dans ce cas, leurs ontologies de processus s'avèrent ne sont pas sémantiquement équivalentes et par conséquent, l'élaboration du méta-modèle MCASMP nécessite la connaissance de l'ensemble des concepts qui sont décrits dans les différentes ontologies afin d'annoter les concepts d'un certain méta-modèle de processus par les concepts des autres méta-modèles, qui lui sont sémantiquement équivalents, et cela en se référant à un alignement des concepts défini par la table de correspondance des concepts des différents méta-modèles de processus (table 4.1).

Bien que la première solution soit la plus avantageuse puisqu'elle fait référence à un ensemble minimum de concepts provenant d'une seule ontologie, néanmoins elle nous paraît à notre sens qu'elle n'est pas pratique dans un contexte réel de coopération et d'un marché très concurrentiel, où les entreprises préfèrent ne pas divulguer leur savoir-faire. Aussi, nous pensons que l'adoption d'une ontologie unique nécessite des consensus qui sont difficiles à atteindre. Par conséquent, la deuxième solution nous semble la plus adaptée.

Le principe consiste alors à annoter, par exemple, un concept *C1* de l'ontologie *O1,* défini par l'acteur *Act1* par un autre concept *C2* de l'ontologie *O2,* défini par l'acteur *Act2*. Cette étape est faite par les experts métiers qui connaissent mieux les différents langages de modélisation de processus définis dans le domaine du Workflow (ou business process) afin de les aligner sémantiquement et les comparer selon leurs objectifs définis par leurs concepteurs. La table 4.1 montre l'alignement des concepts de quelques méta-modèles de processus Workflow tels que XPDL, ebXML et BPML.

.

Concepts du Workflow	**Concepts du ebXML**	**Concepts du XPDL**	**Concepts du BPML**
Processus	MultipartyCollaboration BinaryCollaboration	Workflow Process	Process Abstract
Activité	BusinessActivity CollaborationActivity BusinessTransactionActivity	Workflow Activity	Activité et ses spécialisations
Ressource	Business Transaction	Workflow Application	Participant
Transition	Transition Join Fork	Transition Information	All Sequence Foreach
Objet	BusinessDocument	Relevant Data	Message
Rôle	BusinessPartnerRole	Workflow Participant	Participant
Acteur	AutorizedRole	Workflow Participant	Participant

Table 4.1 : Alignement des concepts des différents méta-modèles de processus.

Cet alignement nous permet de décrire les correspondances entre les concepts qui sont sémantiquement équivalents. La procédure d'annotation consiste alors à élaborer des règles de mapping entre les concepts de ces méta-modèles de processus. Ces règles concernent la correspondance des concepts et peuvent être de plusieurs types (1-1, 1-n, n-1, etc.). Dans notre cas, et pour plus de simplicité, nous nous limitons uniquement à la correspondance des concepts atomiques de type un à un (ou 1-1) et la correspondance entre un concept atomique et un concept énuméré de type un à plusieurs (ou 1-n).

Par exemple, le mapping un à un est utilisé pour la correspondance entre le concept 'Workflow Activity' de XPDL et de celui d''Activity' de WSFL qui est un concept atomique. Quant au mapping un à plusieurs, il est alors utilisé pour la correspondance entre le concept 'Workflow Activity' de XPDL et le concept énuméré d'ebXML à savoir : 'BusinessActivity', 'CollaborationActivity' ou 'BusinessTransactionActivity'.

Par exemple, dans un environnement coopératif entre trois acteurs utilisant trois modèles différents XPDL, WSFL et ebXML dont les ontologies correspondantes s'avèrent ne sont pas totalement identiques, l'annotateur annote le concept d'activité '*Workflow Activity*' du méta-modèle XPDL par le concept d'activité du méta-modèle WSFL *'Activity'*, et par ceux d'ebXML ('*BusinessActivity', 'CollaborationActivity', 'BusinessTransactionActivity*', et cela en indiquant seulement l'équivalence sémantique entre ces concepts (voir table 4.1).

D'un point de vue technique, ces annotations sont codées en XML, et les règles de correspondances sont d'abord, traduites dans une représentation XMLSchéma, puis sont envoyées par l'acteur *Act1* dans un fichier XML ou RDFs, permettant ainsi, à l'acteur *Act2* de mieux interpréter les concepts définis par l'acteur *Act1*.

2.4 Annotation sémantique des modèles de processus

Dans un contexte homogène, l'ontologie du domaine est connue et partagée de manière implicite par les acteurs *Act1* et *Act2*. Dans le cas où les ontologies O1 et O2 sont identiques, pour annoter les modèles de processus *M1* et *M2*, nous utilisons uniquement des annotations lexicales pour expliciter les termes utilisés par les acteurs coopératifs et qui présentent des ambiguïtés. Ce type d'annotation fait référence à une base de données lexicale telle que WordNet [214] et à un glossaire qui contient tous les termes utilisés dans le domaine du Workflow tel que celui de la WfMC [36].

Dans le cas où les ontologies O1 et O2 ne sont pas vraiment identiques, nous annotons les modèles (ou fragments de modèles) en indiquant la correspondance sémantique entre les concepts de ces modèles (ou fragments de modèles) par l'utilisation du symbole d'équivalence '≡'.

La figure 4.3 suivante montre un exemple d'annotation sémantique d'un fragment de modèle, par exemple, une activité d'achat 'Purchase' qui est décrite dans l'ontologie *O1* (correspondant *à* XPDL). L'annotateur l'annote par l'activité 'Buying' décrite dans l'ontologie *O2* (correspondant *à* WSFL), en indiquant la correspondance sémantique entre l'activité 'Purchase'(définie par le concept 'WorkflowActivity' de l'ontologie *O1)* et l'activité 'Buying' (définie par le concept 'Activity' de l'ontologie *O2*).

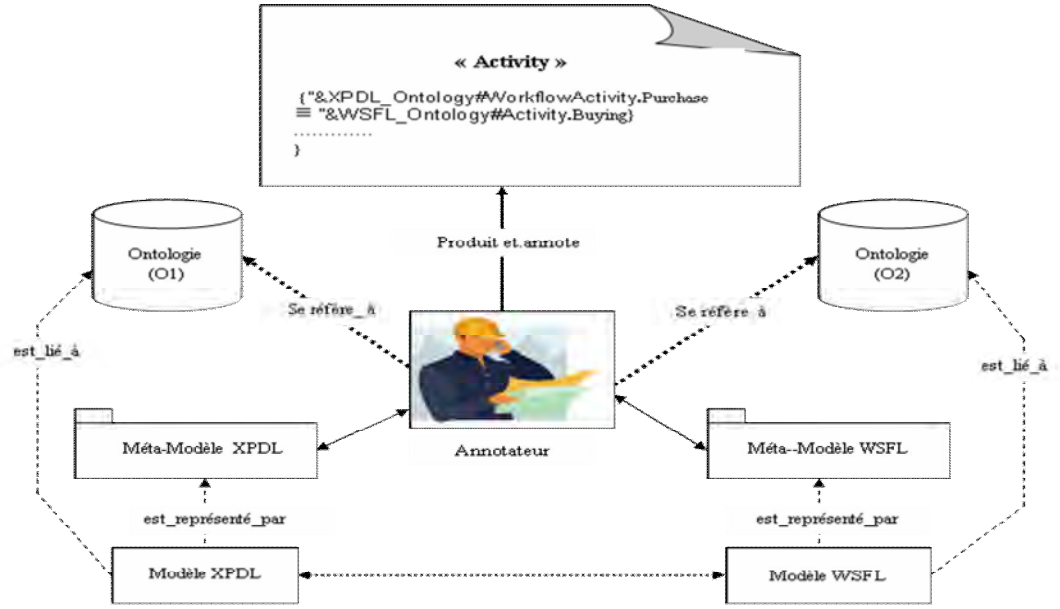

Figure 4.3 : Annotation sémantique d'une activité de commande.

D'un point de vue technique, ces annotations sont codées en XML, en utilisant un namespace 'semAnno' (figure 4.4) et les règles de correspondances sont d'abord, traduites dans une représentation XMLSchéma .La figure 4.6 présente un exemple d'une codification en XML pour annoter les équivalences sémantiques entre trois instances de concepts de XPDL, de WSFL et d'ebXML et qui sont respectivement '*Purchase*', '*Buying*', et *Purchase*'. Cette annotation fait référence respectivement aux trois concepts : '*WorkflowActivity*', '*Activity*' et *(BusinessActivity, CollaborationActivity, BusinessTransactionActivity)* de XPDL, de WSFL et du ebXML

```
<semAnno:WorkflowActivity rdf:ID="Purchase"
<semAnno:refers_to
        rdf:resource="&XPDL_Ontology#WorkflowActivity"/>
            </semAnno:has_equivalent>
                <semAnno: AtomicConcept
                    rdf:resource="&WSFL_Ontology#Activity"/>
                        <semAnno: Activity rdf:ID="Buying">
                 </semAnno:AtomicConcept>
                <semAnno:EnumeratedConcept
                    <semAnno:has>
                            <semAnno: AtomicConcept
                        rdf:resource="&ebXML_Ontology#BusinessActivity"/>
                            <semAnno: BusinessActivity rdf:ID="Purchase">
                        </semAnno:AtomicConcept>
                            <semAnno: AtomicConcept
                        rdf:resource="&ebXML_Ontology#CollaborationActivity"/>
                            <semAnno: CollaborationActivity rdf:ID="Purchase">
                    </semAnno:AtomicConcept>
                            <semAnno: AtomicConcept
                        rdf:resource="&ebXML_Ontology#BusinessTransactionActivity"/>
                            <semAnno: BusinessTransactionActivity rdf:ID="Purchase">
                    </semAnno:AtomicConcept>
                </semAnno:has>
            </semAnno:EnumeratedConcept>
            </semAnno:has_equivalent>
  </semAnno: refers_to>
</semAnno:WorkflowActivity>
```

Figure 4.4 : Codification de l'annotation sémantique d'une activité de commande.

Pour une perspective d'interopérabilité orientée buts ou profils et pour des objectifs de découverte sémantique et de réutilisation des modèles dans d'autres projets spécifiques, nous proposons des annotations structurelles en faisant référence aux ontologies des buts et des profils.

2.5 Annotation sémantique des profils des modèles de processus

Il s'agit d'annoter ce que font les modèles de processus afin de les découvrir sémantiquement par leurs propriétés fonctionnelles (inputs outputs, etc.) et par leurs propriétés non fonctionnelles (domaine d'application, catégorie, etc.).

Pour annoter sémantiquement les profils des modèles de processus, nous utilisons une seule ontologie de profil, dans le cas où les ontologies des partenaires sont identiques. Dans le cas contraire, nous annotons le profil d'un modèle d'un partenaire par les concepts décrits dans l'ontologie du profil d'un autre partenaire en se référant à un alignement des concepts des profils.

2.6 Annotation sémantique des buts des modèles de processus

L'annotation des buts des modèles de processus est basée sur une ontologie de buts afin de spécifier les capacités et les objectifs que doivent atteindre les processus afin de les découvrir sémantiquement par leurs objectifs métiers. Cette ontologie correspond donc, à un ensemble de concepts et de relations sémantiques entre ces concepts que nous évoquons par la suite, dans la section 3.6. Pour annoter sémantiquement les buts des modèles de processus, nous utilisons une seule ontologie de buts, dans le cas où les ontologies des partenaires sont identiques. Dans le cas contraire, nous annotons les buts d'un modèle d'un partenaire par les concepts décrits dans l'ontologie des buts d'un autre partenaire en se référant à un alignement des concepts des buts.

Cependant, il ne faut pas oublier que par nature, différents aspects sont intrinsèquement liés à tout modèle de Workflow, et qui sont très importants tels que aspect informationnel, fonctionnel, comportemental. En effet, ils facilitent l'interprétation des modèles et aident les acteurs d'interopérabilité à mieux les comprendre lors de l'échange et de la coopération. Donc, afin d'éviter toute ambiguïté et avoir une bonne compréhension de ces modèles, nous proposons d'annoter ces aspects de base.

2.7 Annotation sémantique des aspects de base de modèles de processus

En se référant à l'alignement des concepts des différents méta-modèles de processus (table 4.1), nous proposons un type d'annotation sémantique de type structurel en utilisant la notation UML, qui utilise des notes (ou commentaires) pour annoter tout élément d'une classe UML. Quant aux annotations sémantiques employées, elles sont donc de type organisationnel, informationnel, fonctionnel ou comportemental, et elles font référence à des ontologies.

Dans ce contexte homogène, nous proposons deux solutions :

1- Les ontologies correspondant aux différents aspects de base des modèles de processus sont identiques, Dans ce cas, l'annotation de chaque type d'aspect se réfère à son ontologie unique correspondante qui peut être de type informationnel, organisationnel ou autre.

2- Les ontologies correspondant aux différents aspects de base des modèles de processus s'avèrent ne sont pas vraiment identiques. Dans ce cas, l'annotation sémantique d'un concept d'une ontologie d'un partenaire (par exemple, de type informationnel) se fait en faisant référence à un concept d'une ontologie de même de type d'un autre partenaire (c'est-à-dire informationnelle).

De même, nous pensons que dans un cas réel de coopération et surtout dans un marché caractérisé par la concurrence, les partenaires d'une alliance se méfient toujours de leurs partenaires dans le même secteur afin de protéger leurs techniques de travail. Aussi, l'utilisation des ontologiques identiques pour les différents aspects des modèles de processus, nécessite des engagements des partenaires qui sont difficiles à respecter. Par conséquent, la deuxième solution nous semble la plus adaptée.

2.7.1 Annotation sémantique de type organisationnel

Ce type d'annotation permet de faire la correspondance entre les concepts organisationnels qui sont définis dans les différentes ontologies organisationnelles. Par exemple, pour deux ontologies organisationnelles correspondantes aux méta-modèles de processus XPDL et ebXML, l'annotation d'un concept organisationnel tel qu'un rôle, consiste à indiquer l'équivalence sémantique entre les concepts "*WorkflowParticipant*" et *"BusinessPartnerRole"*, qui sont définis respectivement par l'acteur *Act1* dans l'ontologie organisationnelle XPDL et par l'acteur *Act2* dans l'ontologie organisationnelle ebXML.

2.7.2 Annotation sémantique de type informationnel

Ce type d'annotation permet de faire la correspondance sémantique entre les paramètres d'entrées (inputs) et de sorties (outputs) d'une activité, qui sont définis dans leurs ontologies informationnelles respectives. Par exemple, pour deux ontologies informationnelles correspondantes aux méta-modèles de processus XPDL et ebXML, l'annotation d'un concept informationnel tel qu'un paramètre d'entrée, consiste à indiquer l'équivalence sémantique entre les concepts ("*ProductCode*" et "*ProductQuantity*") de l'ontologie informationnelle XPDL et les concepts ("*Code"* et *"Quantity"*) de l'ontologie informationnelle ebXML.

2.7.3 Annotation sémantique de type fonctionnel

Ce type d'annotation concerne les opérations (appelées aussi fonctions) qui permettent de décrire l'aspect fonctionnel d'un modèle de processus. Ces mêmes opérations nous permettent aussi, de spécifier l'aspect dynamique du processus que nous représentons par un diagramme d'activité UML ou par un réseau de pétri.

Bien que ce contexte soit homogène, deux acteurs d'interopérabilité *Act1* et *Act2* peuvent utiliser deux notations différentes. La première correspond au diagramme d'activité UML qui est utilisé par l'acteur *Act1*, tandis que la deuxième correspond à l'utilisation d'un réseau de pétri qui estutilisé pour l'acteur *Act2.*

La figure suivante montre un diagramme d'activité UML correspondant à une activité d'une commande d'achat comportant différentes opérations que doit exécuter l'activité pour sa réalisation à savoir : '*CreateOrder', 'CheckOrder', 'CancelOrder', 'NotifyCustomer', 'FillOrder', 'ShipOrder', 'CloseOrder'*.

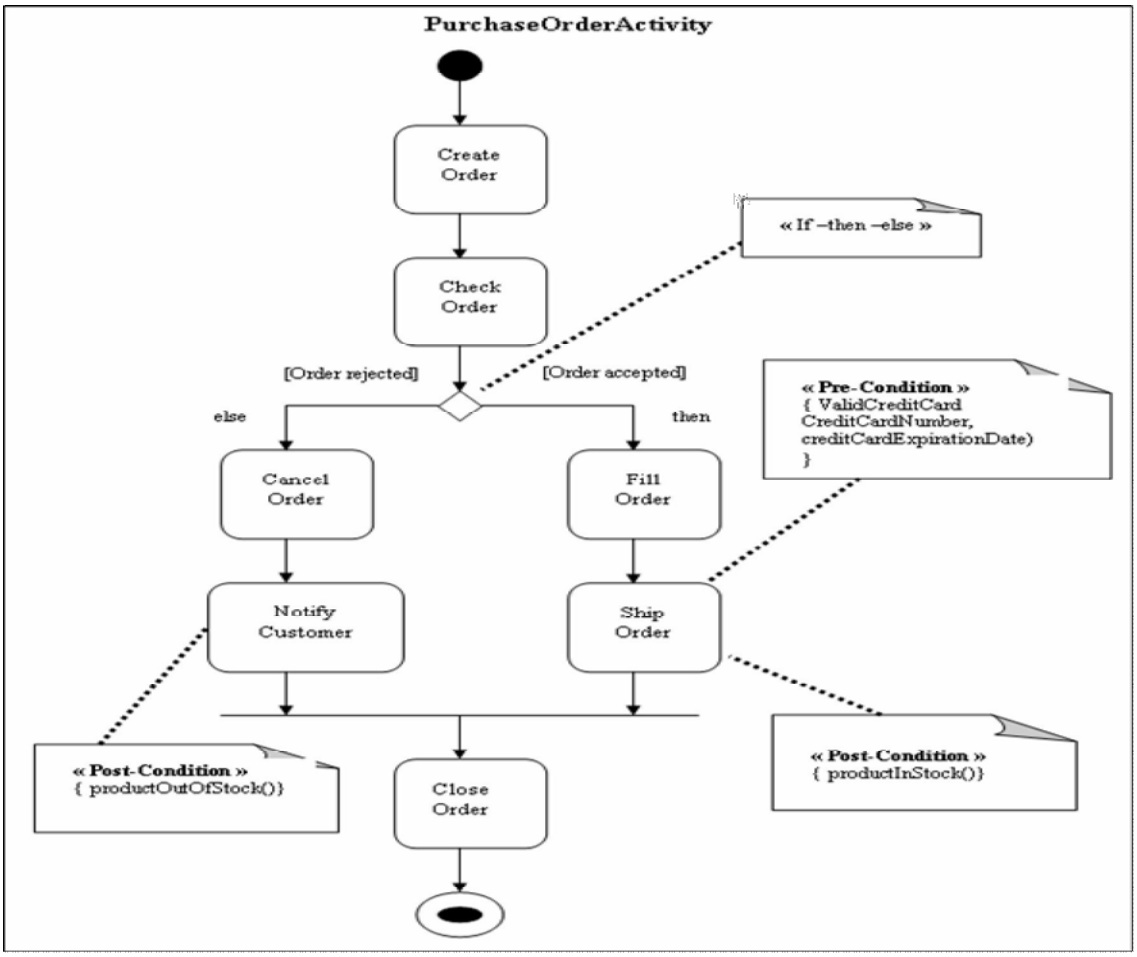

Figure 4.5 : Modélisation d'une d'activité d'une commande d'achat à l'aide d'un diagramme d'activité UML.

La représentation graphique d'une activité d'une commande d'achat peut être aussi, présentée par un réseau de pétri (figure 4.6).

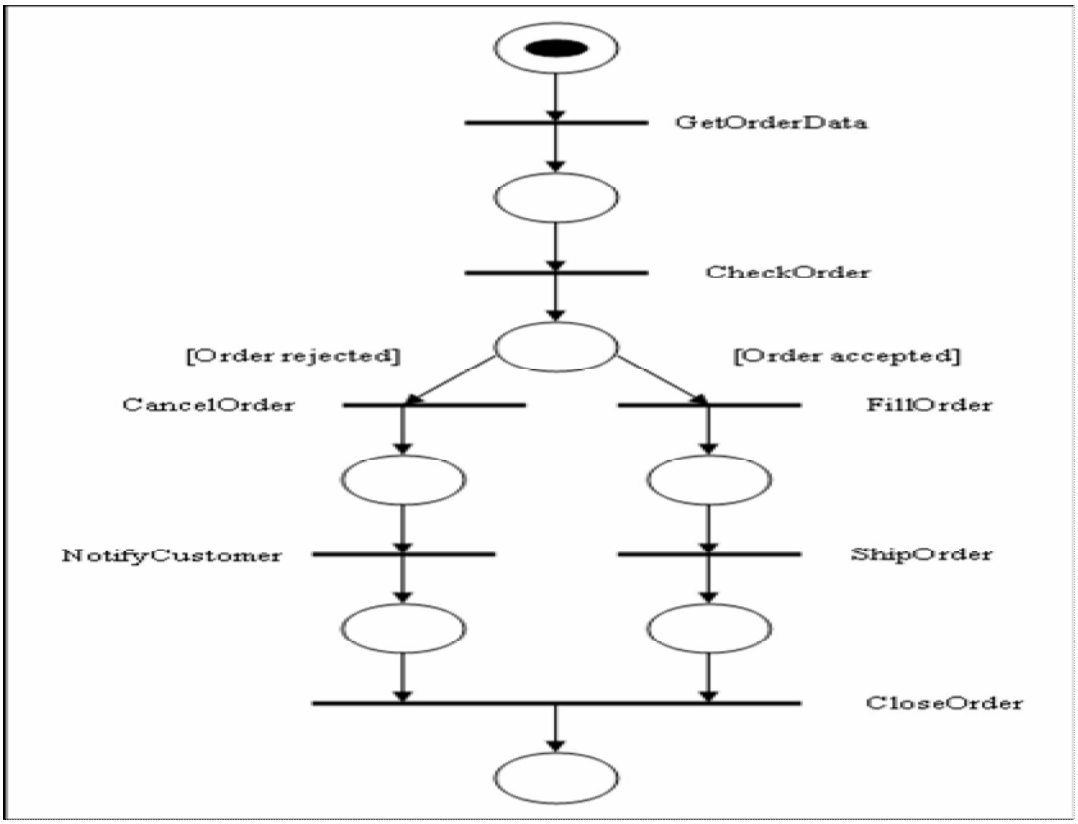

Figure 4.6 : Modélisation d'une d'activité d'une commande d'achat à l'aide de réseau de pétri.

Ces deux types de notation (diagramme d'activité UML et réseau de pétri) nous permettent d'établir les correspondances suivantes (table 4.2) :

Concepts du diagramme d'activité UML	**Concepts du réseau de pétri**
Opération avec conditions d'activation vérifiées	Arc et Transition mise à feu
Opération avec conditions d'activation non vérifiées	Arc et Transition non mise à feu
Etat d'une activité	Place

Table 4.2: Correspondances entre les concepts de diagramme d'activité UML et réseau de pétri.

L'annotation fonctionnelle consiste à décrire la correspondance entre les concepts fonctionnels correspondant entre les concepts utilisés dans le diagramme d'activité UML (opérations) et les concepts du réseau de pétri (transitions). Par exemple, l'opération "*CreateOrder*" définie le digramme d'activité correspond à la transition "*GetOrderData*" qui est définie dans le réseau de pétri.

2.7.4 Annotation sémantique de type comportemental

Pour annoter le comportement d'un modèle de processus, l'annotateur se réfère aux concepts utilisés par le diagramme d'activité UML et le réseau de pétri pour faire la correspondance entre les opérations UML et les transitions. Pour spécifier des contraintes éventuelles sur les opérations UML, nous proposons de les exprimer par le langage OCL (Object Constraint Language) qui est intégré à UML. Il est simple, possédant une grammaire élémentaire et bien adapté aux diagrammes de classes UML.

Donc, l'utilisation conjointe des trois types d'annotation (diagramme d'activité, réseau de pétri et diagramme d'états/transitions) permet aux acteurs de l'interopérabilité de mieux interpréter l'aspect dynamique des modèles de processus.

La figure 4.7 montre un diagramme d'états/transitions d'un fragment d'un modèle de processus tel que l'objet "commande" qui montre les différents états passés depuis sa création jusqu'à sa terminaison.

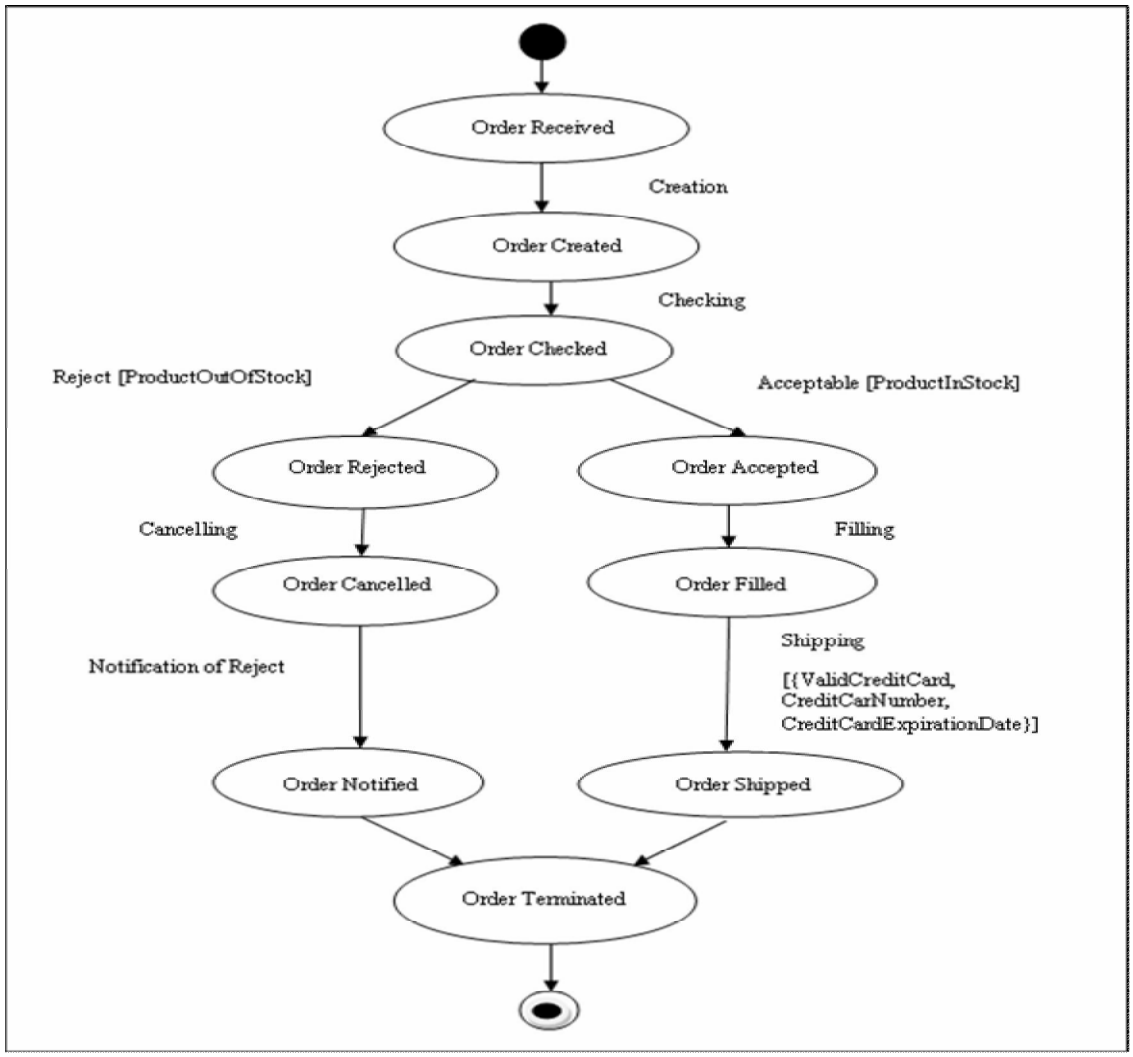

Figure 4.7 : Diagramme d'états/transitions d'une commande d'achat.

La figure 4.8 montre un exemple d'annotation comportementale de l'opération "*CreateOrder*" de l'activité commande d'achat dont l'événement associé est "*Creation*" et l'état correspondant à cette opération est "*Order Created*". Pour annoter, nous nous référons à l'aspect dynamique du modèle défini par le diagramme d'activité UML, le diagramme d'états/transitions et le réseau de pétri. L'annotation consiste alors à décrire les correspondances structurelles entre le comportement spécifié par l'opération "*Create Order*" (définie par l'acteur *Act1* dans la figure 4.5) et le comportement spécifié par la transition "*GetOrderData*" (définie par l'acteur *Act2* dans la figure 4.6*)*, en se référant à la table de correspondance entre les concepts du diagramme

d'activité UML et le réseau de pétri (table 4.2) et aux contraintes OCL qui sont définies éventuellement sur les opérations UML.

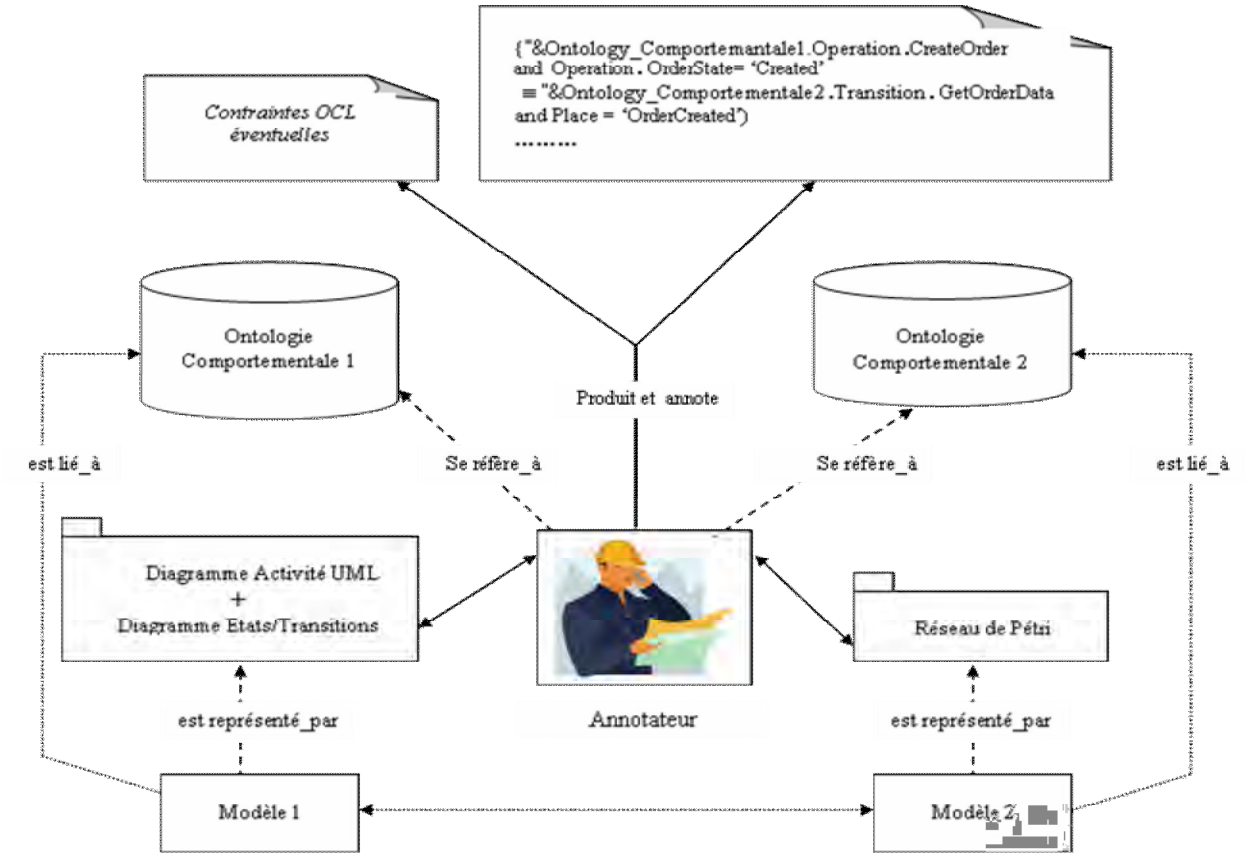

Figure 4.8 : Annotation comportementale d'une activité de commande.

D'un point de vue technique, ces annotations comportementales sont traduites par l'acteur *Act1* en une représentation Schéma XML et envoyées vers l'acteur *Act2* pour interprétation (figure 4.9).

```
<semAnno : Concept rdf : resource= " uri:// Behaviour_Ontology1#Operation"/>
        <operation_element id = "Op1" >
            <name > CreatedOrder> </name>
            <has_event> Creation> </has_event >
            <has_state> OrderCreated> </has_state >
        </operation_element>
    <semAnno:refers_to
            rdf:resource= " uri:// Behaviour_Ontology2#Transition"/>
                < transition_element id ="T1" >
                    <name > GetOrderData> </ name>
                    <has_place> OrderCreated > </has_place >
                </transition_element >
     <semAnno:refers_to>
</semAnno : Concept>
.....
```

Figure 4.9 : Codification en XML d'une annotation comportementale.

Aussi, il ne faut pas oublier que pour annoter le comportement d'un modèle de processus, il faut d'abord, recenser tous les différents états/transitions passés par les différentes activités du processus, ensuite les annoter individuellement.

2.7.5 Annotation sémantique des ressources

Ce type d'annotation concerne les ressources matérielles (tableur, éditeur), logicielles (programmes, applications externes), humaines (personnes compétentes bien déterminées) ou temporelles (temps prévu et réalisé par l'activité). Puisque le contexte est homogène, l'annotation consiste à alors décrire la correspondance entre des les différents concepts des ressources qui sont utilisées par les différents acteurs en faisant référence à une même ontologie de ressources.

Cependant, dans le cas où les ontologies des ressources des partenaires s'avèrent ne sont pas identiques, l'annotation d'un concept de ressources se fait en faisant référence à un autre concept défini par un autre partenaire dans son ontologie de ressources. Par exemple, pour deux ontologies des ressources correspondantes aux méta-modèles de processus XPDL et ebXML, l'annotation d'un concept de ressource tel que "*WorkflowApplication*" qui est défini dans l'ontologie XPDL se fait en faisant référence au concept de ressource *"Participant*" défini dans l'ontologie ebXML .

2.8 Synthèse

L'interopérabilité dans un contexte homogène ne constitue pas un problème crucial en soi car la sémantique est implicite et elle existe dans la mentalité des acteurs coopératifs. En effet, le domaine de connaissances est connu par tous les partenaires qui partagent un même référentiel commun et des ontologies identiques de type processus, informationnelle, organisationnelle, fonctionnelle, comportemental et de ressources.

Dans cet environnement homogène, l'utilisation de ces ontologies identiques suffit pour des acteurs d'interpréter les modèles de processus lors de l'échange et de la coopération. Cependant, dans un contexte réel de coopération très compétitif, bien qu'on soit dans un contexte homogène et dans un même domaine, les partenaires d'une alliance veillent à protéger leur savoir-faire et leur expérience de travail des autres partenaires. Dans ce contexte, nous avons proposé deux techniques d'annotation sémantique : *i)* basée sur des ontologies uniques *ii)* basées sur les ontologies des autres partenaires.

Aussi, nous avons proposé deux types d'annotations :

- Annotations lexicales : elles permettent de mieux expliciter les termes utilisés par les acteurs coopératifs en faisant référence à une base de données lexicale telle que WordNet [214]**,** ou plutôt à un glossaire comme celui de la WfMC (Workflow Mangement Coalition) [36] qui contient tout le vocabulaire de base utilisé dans le domaine du Workflow.

- Annotations structurelles : elles permettent d'expliciter le sens des éléments (attribut, opération, etc.) de tout modèle de processus représenté par un diagramme de classe UML. Le but est de permettre de comprendre la signification de ces éléments lors de l'échange et de la coopération. Ces annotations sont alors attachées aux concepts de ces modèles de processus et font référence aux différentes ontologies qui sont totalement identiques on non.

Pour mieux illustrer notre propos, nous donnons un exemple d'annotation structurelle d'un fragment de modèle de processus correspondant à une activité de commande d'achat qu'il s'agit

d'annoter (figure 4.10). Nous supposons que les ontologies des partenaires ne sont pas vraiment identiques. le schéma d'annotation est alors élaboré par un annotateur afin qu'il puisse être interprété de manière commune par trois acteurs distincts *Act1, Act2* et *Act3* désirant échanger des modèles de processus : XPDL, ebXML et BPML.

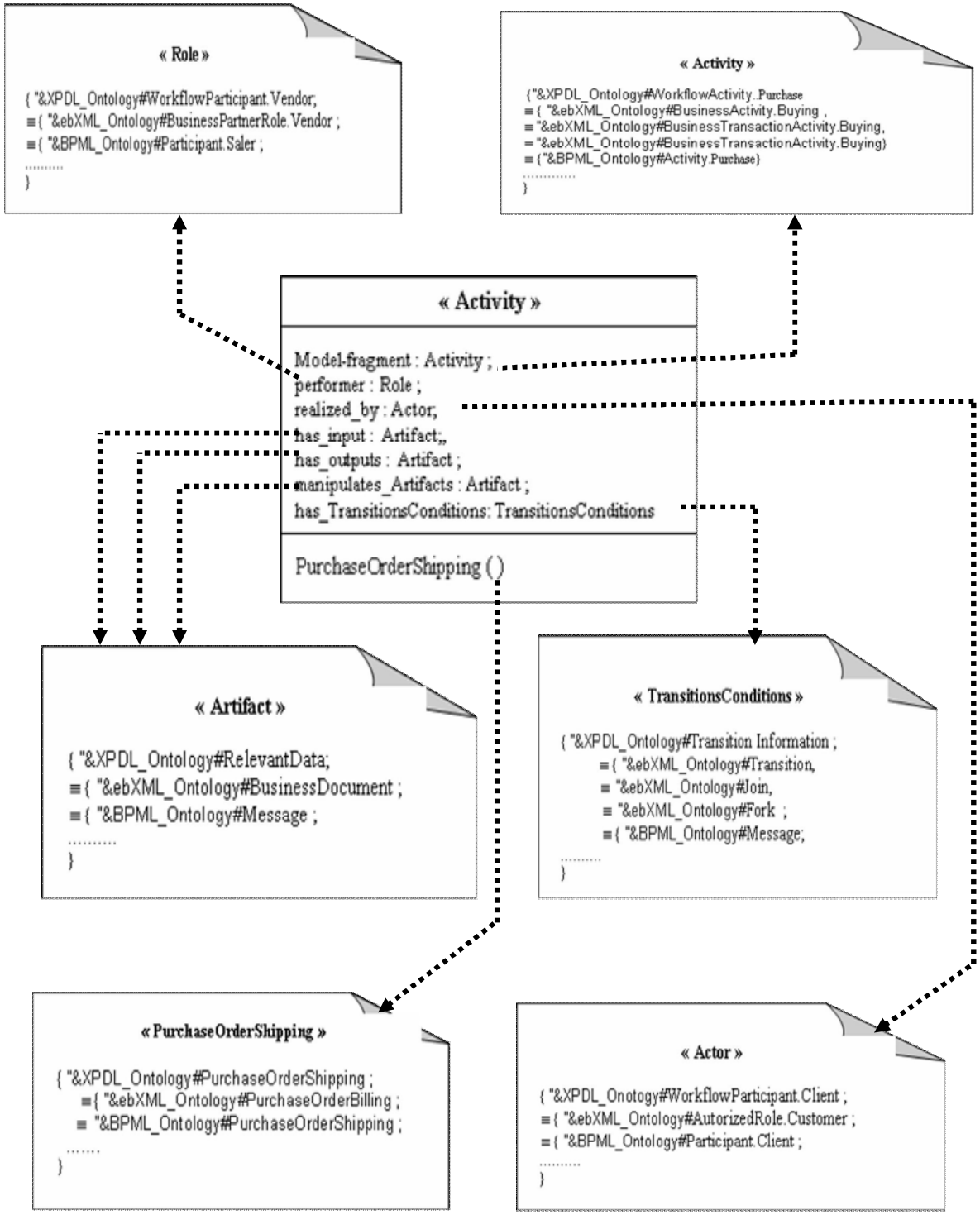

Figure 4.10 : Schéma commun d'annotation structurelle d'une activité de commande.

Contrairement au contexte précédent, l'interopérabilité dans un environnemment hétérogéne réprésente un défi majeur pour des acteurs coopératifs où la notion de sémantique formelle doit être présente afin d'éliminer les différences d'interprétation et de mauvaise compréhension des modèles lors de l'échange et de la coopération. C'est ce que nous allons aborder dans la section suivante.

3. Interopérabilité dans un environnement hétérogène

Dans un contexte hétérogène, deux acteurs *Act1* et *Act2* peuvent utiliser leurs propres notations pour élaborer leurs modèles (*M1* et *M2*), et par conséquent leurs propres méta-modèles (*MM1* et *MM2*), leurs propres ontologies (*O1* et *O2*), et éventuellement leurs propres schémas d'annotation (*A1* et *A2*).

Cependant pour coopérer, ces deux acteurs doivent évidemment procéder à des mappings et à des transformations à chaque niveau hétérogène, c'est-à-dire, entre leurs modèles, leurs méta-modèles, leurs schémas d'annotation et enfin entre leurs ontologies (figure 4.11).

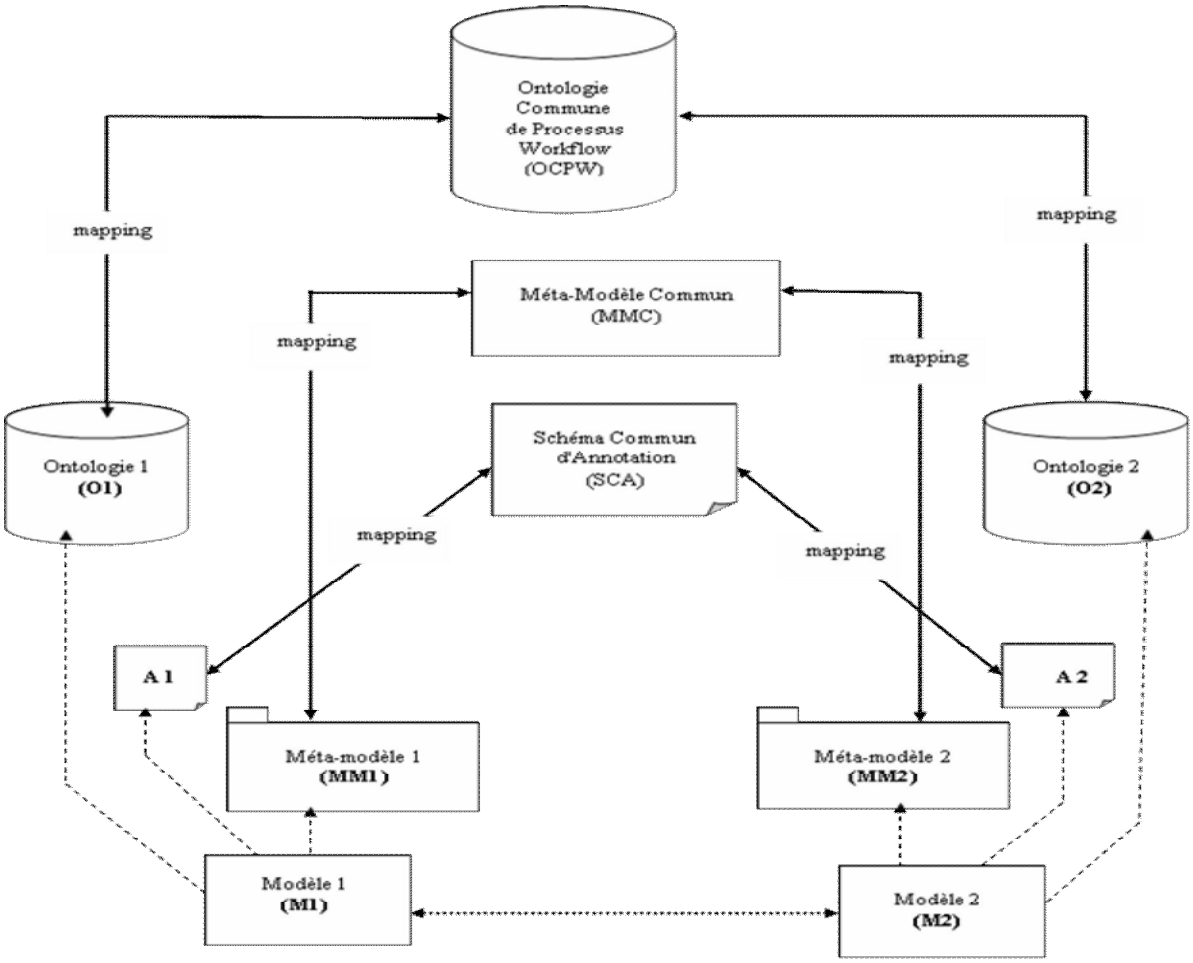

Figure 4.11 : Différents mappings dans un contexte hétérogène d'interopérabilité.

Dans ce contexte hétérogène, certains auteurs [210] ont défini un scénario d'annotation d'un modèle où le fournisseur d'annotations utilise à la fois, les services de gestion d'ontologies (mapping d'ontologies, de méta-modèles, etc.) et les services de gestion d'annotations (stockage et réutilisations d'annotations, contrôle de consistance des annotations, etc.) pour la

modélisation des différents aspects tels que fonctionnel informationnel qui sont liés aux modèles de processus d'entreprise. Ces annotations sont attachées aux modèles (ou fragments de modèles) et font référence à des ontologies (figure 4.12).

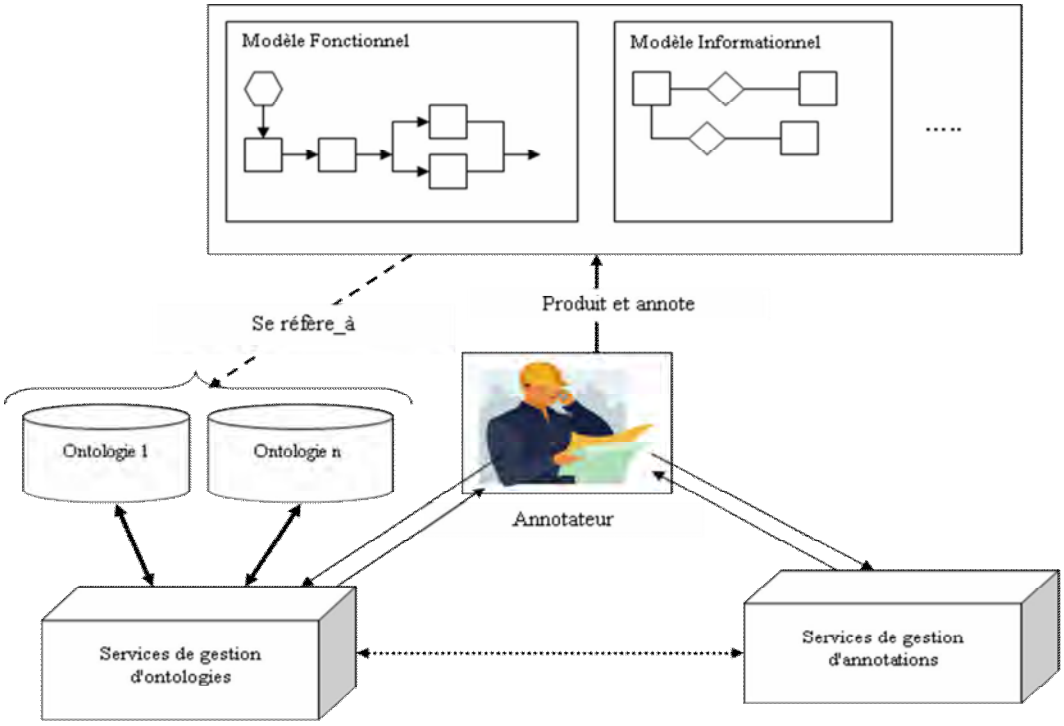

Figure 4.12 : Scénario d'un modèle d'annotation [210].

L'objectif de ces annotations sémantiques est donc, de donner une signification commune à la diversité d'annotations qui peut être attachée à un modèle ou à un fragment de modèle (par exemple, une activité), c'est-à-dire, un accord commun sur la sémantique donnée aux annotations elles-mêmes doit être établi au préalable, ainsi que sur la forme sous laquelle ces annotations ont été fournies selon un modèle ou un schéma d'annotation qui peut comporter plusieurs types d'annotations. Dans ce contexte hétérogène, plusieurs travaux se sont intéressés aux techniques d'annotation sémantique des modèles de processus [216], [217], [211] [212].

3.1 Principe de l'approche

L'approche que nous préconisons permet de résoudre l'hétérogénéité sémantique des modèles de processus rencontrée à différents niveaux :

- **Au niveau des méta-modèles** : le but est de traiter l'hétérogénéité des méta-modèles de processus (au niveau des concepts). L'annotation sémantique des méta-modèles de processus consiste alors à utiliser les concepts de l'ontologie commune de processus Workflow (OCPW) comme des méta-données pour annoter la sémantique des concepts des différents méta-modèles de processus. OCPW est construite à partir d'un méta-modèle commun de processus Workflow (MMCPW) élaboré et issu de l'alignement des concepts des différents méta-modèles de processus tels que XPDL, BPML, ebXML.

- **Au niveau des modèles** : le but est de traiter l'hétérogénéité des modèles de processus (au niveau des contenus). L'annotation sémantique des modèles de processus consiste alors à utiliser plusieurs ontologies communes : l'ontologie du domaine de référence par exemple, SCOR (Supply-Chain Operation Reference-model) [226], thésaurus, l'ontologie des buts (SCOR aussi), et l'ontologie des profils. Thésaurus est utilisé comme une forme d'ontologie pour expliciter les termes qui présentent des ambiguïtés d'interprétation. L'ontologie des buts sert à annoter les buts ou les objectifs que devraient atteindre les processus. Son rôle est d'enrichir la sémantique des objectifs visés par les processus et permettre la découverte sémantique des modèles de processus (qui sont basés sur des objectifs métiers). L'ontologie des profils sert à annoter les profils des modèles exprimés en termes de fonctionnalités des processus. Son rôle est d'enrichir la sémantique des profils publiés par les acteurs coopératifs afin de découvrir sémantiquement les modèles de processus (qui sont basés sur ces profils)

- **Au niveau des aspects de base des modèles** *:* le but est de traiter l'hétérogénéité des aspects des modèles qui sont par nature liés à tout modèle de processus. L'annotation sémantique de ces aspects consiste à utiliser les ontologies communes de type informationnel, comportemental, fonctionnel, organisationnel et de gestion des ressources.

La technique d'annotation sémantique des modèles de processus que nous proposons est basée alors sur des ontologies afin d'enrichir et de réconcilier la sémantique de ces modèles à différents niveaux d'hétérogénéité Elle s'inspire principalement des approches de [212], [213], mais elle est générale dans le sens où elle permet aussi, de prendre en compte l'hétérogénéité sémantique rencontrée au niveaux des différents aspects de base qui sont intrinsèquement liés à tout modèle de processus.

Cependant, pour réduire le nombre de mappings entre les différentes annotations de modèles, de méta-modèles, de schémas d'annotation, ainsi que sur la variété des concepts utilisés par les acteurs coopératifs, l'approche que nous proposons est basée sur un schéma commun d'annotation sémantique. Elle s'appuie donc, sur l'utilisation d'une structure d'annotation sémantique commune pour décrire la sémantique des modèles. Cette description sémantique fait référence à un ensemble d'ontologies communes telles que ontologie de processus Workflow, ontologie du domaine de référence, sans oublier d'autres types d'ontologies qui sont liés aux aspects des modèles. L'avantage qu'elle procure, est qu'elle permet une bonne interprétation commune des modèles annotés. La technique d'annotation sémantique utilisée est une technique incrémentale Elle constitue d'abord, à élaborer un méta-modèle commun d'annotation sémantique (MCASMP) permettant d'annoter tous les modèles de processus qui sont destinés à être interopérables, ensuite de l'étendre et l'enrichir au fur et à mesure pour la prise en compte des différents aspects.

Pour mieux illustrer notre approche, nous présentons le processus d'annotation sémantique des modèles de processus qui comporte plusieurs étapes (figure 4.13) :

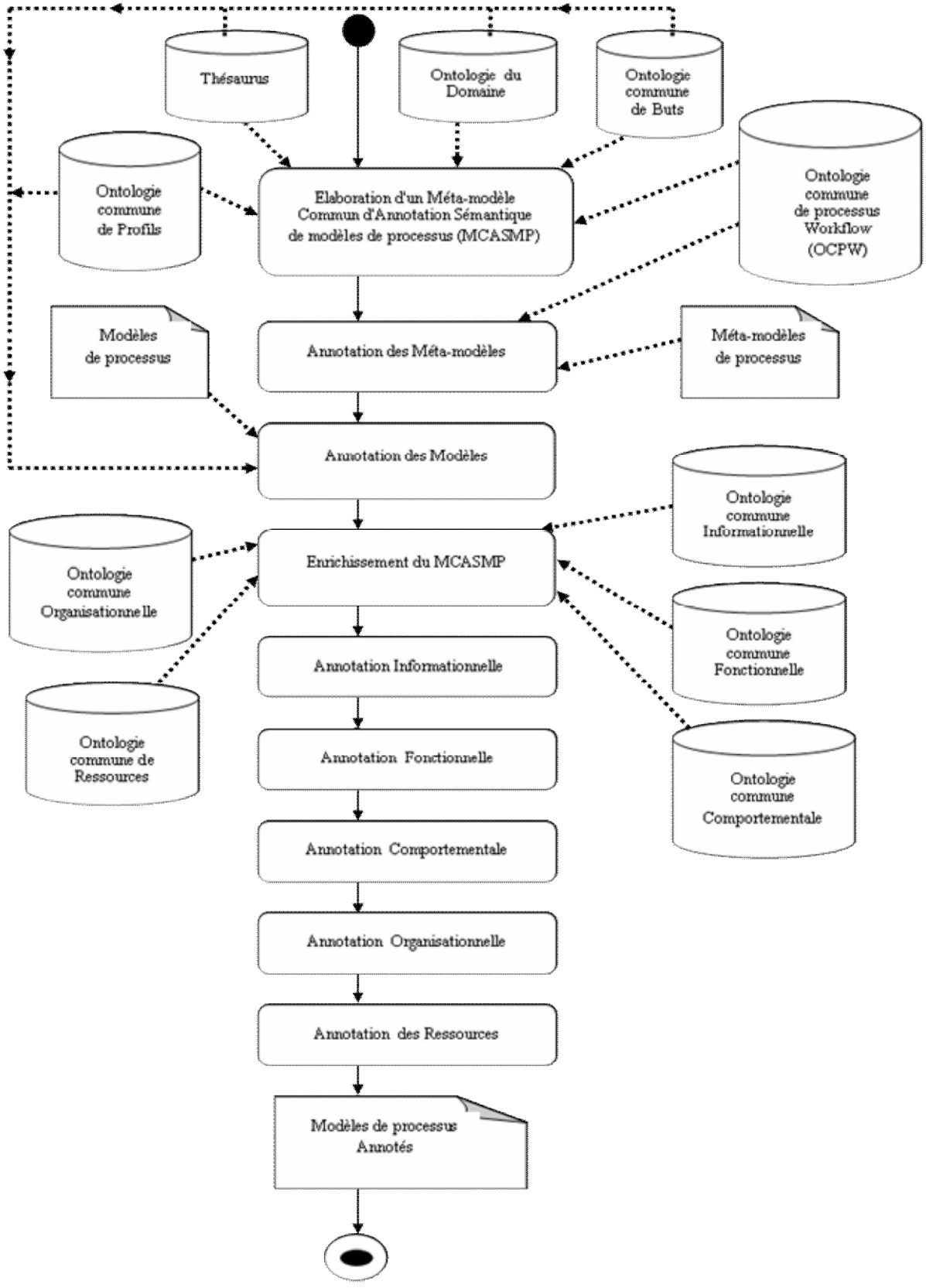

Figure 4.13 : Processus d'annotation sémantique des modèles de processus.

Ce processus est basé principalement sur quatre étapes que nous décrivons brièvement comme suit :

1. **Elaboration du méta-modèle commun d'annotation sémantique de processus :** Elle consiste à élaborer un méta-modèle commun d'annotation sémantique de processus (MCASMP) dont les concepts proviennent de l'ontologie commune de processus Workflow (OCPW), de l'ontologie du domaine de référence, de l'ontologie des buts et enfin de l'ontologie des profils. L'ontologie des buts et l'ontologie des profils sont associées au processus d'annotation sémantique dans un contexte d'interopérabilité orienté buts ou profils, et cela pour des objectifs de découverte sémantique des modèles et de leur réutilisation. Ce méta-modèle sera enrichi et étendu par d'autres concepts liés aux différentes ontologies communes de type informationnel, organisationnel, etc.

2. **Annotation sémantique des méta-modèles de processus : E**lle consiste à annoter les concepts des différents méta-modèles de processus (XPDL, ebXML, etc.) par les concepts de l'ontologie commune de processus (OCPW). Dans cette étape, nous utilisons l'ontologie commune OCPW comme un médiateur pour annoter sémantiquement les concepts des différents méta-modèles de processus.

3. **Annotation sémantique des modèles de processus :** Une fois, les concepts des méta-modèles de processus sont annotés par ceux de l'ontologie OCPW, nous procédons à l'annotation des contenus des modèles en se référant aux différentes ontologies communes qui sont partagées telles que l'ontologie du domaine de référence, l'ontologie des buts et l'ontologie des profils, sans oublier les différentes ontologies qui sont rattachées aux aspects de base des modèles et qui sont de type informationnel, etc. Aussi, l'ajout d'un thésaurus comme une forme d'ontologie, nous permet de mieux expliciter les termes utilisés par les modèles et éviter des ambiguïtés d'interprétation.

4. **Annotation sémantique des différents aspects des modèles de processus :** Cette étape consiste à annoter les différents aspects de base qui sont intrinsèquement liés aux modèles de processus et qui ont certainement un impact pour une bonne compréhension des modèles.

3.2 Le méta-modèle commun d'annotation sémantique des modèles de processus

Le méta-modèle commun d'annotation sémantique de modèles (MCASMP) contient les concepts de l'ontologie commune de processus Workflow (OCPW) et les concepts des autres ontologies telle que l'ontologie du domaine et l'ontologie des buts qui sont choisies par les experts métiers pour annoter leurs modèles de processus.

Les concepts du MCASMP dérivent de l'ontologie commune OCPW. Pour élaborer le MCASMP, nous devons d'abord, construire cette dernière. Comme la notion de méta-modèle est fortement liée à la notion d'ontologie [228], alors nous procédons d'abord, à l'élaboration d'un méta-modèle commun de processus Workflow (MMCPW) qui devrait contenir l'ensemble des concepts communs qui sont partagés par les acteurs coopératifs. Ce méta-modèle commun MMCPW sera alors considéré comme une ontologie commune de processus Workflow (OCPW), et sera implémenté en utilisant l'outil ontologique Protégé [191].

3.3 Annotation sémantique des méta-modèles de processus

Dans la pratique, l'annotation des méta-modèles de processus est effectuée par les experts métiers du domaine. Cependant, pour annoter la sémantique des concepts de ces méta-modèles

par des concepts ontologiques, nous employons les concepts de OCPW comme des méta-données. Par conséquent, nous considérons OCPW en tant que médiateur sémantique pour l'annotation des concepts des différents méta-modèles de processus.

Les concepts de l'ontologie commune OCPW sont donc mappés vers ceux des différents méta-modèles de processus. Par exemple, les concepts des méta-modèles XPDL et ebXML qui sont respectivement '*WorkflowActivity'* et '*BusinessActivity*' sont annotés tous les deux par le même concept *'Wf-Activity'* de l'ontologie OCPW, se trouvent ainsi, remplacés par le même concept de l'ontologie commune OCPW qui est *'Wf-Activity'* .

Quant à la procédure d'annotation, elle permet d'établir des règles de mappings entre les concepts des différents méta-modèles de processus et ceux de l'ontologie commune OCPW. Ainsi, tous les concepts des méta-modèles de processus se trouvent remplacés par ceux de l'ontologie OCPW et par conséquent, les modèles qui sont représentés par ces méta-modèles se trouvent représentés par des concepts ontologiques de OCPW.

- **Le méta-modèle commun de processus Workflow**

Pour définir l'ensemble des concepts du méta-modèle commun de processus Workflow (MMCPW), nous nous sommes basés dans notre travail précédent [131] sur l'étude des différents formalismes de modélisation utilisés dans le domaine du Workflow (ou business process) tels que XPDL [4], XLANG [5], WSFL [6], BPML [10], ebXML [11]. Nous avons comparé leurs concepts spécifiques en les alignant selon leurs objectifs définis par leurs concepteurs. Ensuite, nous avons extrait un noyau de concepts de base qui est commun à l'ensemble des modèles de processus utilisés dans le domaine du Workflow ou celui de business process.

Les concepts que nous avons extraits sont génériques tels que processus, activité, ressource. La table 4.3 montre les correspondances entre les concepts de Workflow et ceux de quelques méta-modèles de processus Workflow tels que: XPDL, ebXML et BPML.

Concepts du Workflow	**Concepts du méta-modèle commun (MMCPW)**	**Concepts du ebXML**	**Concepts du XPDL**	**Concepts du BPML**
Processus	Wf-Process	MultipartyCollaboration BinaryCollaboration	Workflow Process	Process Abstract
Activité	Wf-Activity Wf-Step Wf-Task	BusinessActivity CollaborationActivity BusinessTransactionActivity	Workflow Activity	Activité et ses spécialisations
Ressource	Wf-Resource	Business Transaction	Workflow Application	Participant
Transition	Wf-Transition	Transition Join Fork	Transition Information	All Sequence Foreach
Objet	Wf-Artifact Wf-Data	BusinessDocument	Relevant Data	Message
Rôle	Wf-Role	BusinessPartnerRole	Workflow Participant	Participant
Acteur	Wf-Actor	AutorizedRole	Workflow Participant	Participant

Table 4.3 : Correspondances entre les concepts des méta-modèles de processus Workflow.

L'ensemble de ces concepts communs constitue le méta-modèle commun de processus Workflow (MMCPW) (figure 4.14) qui est unifié, générique et accepté par les acteurs de l'interopérabilité.

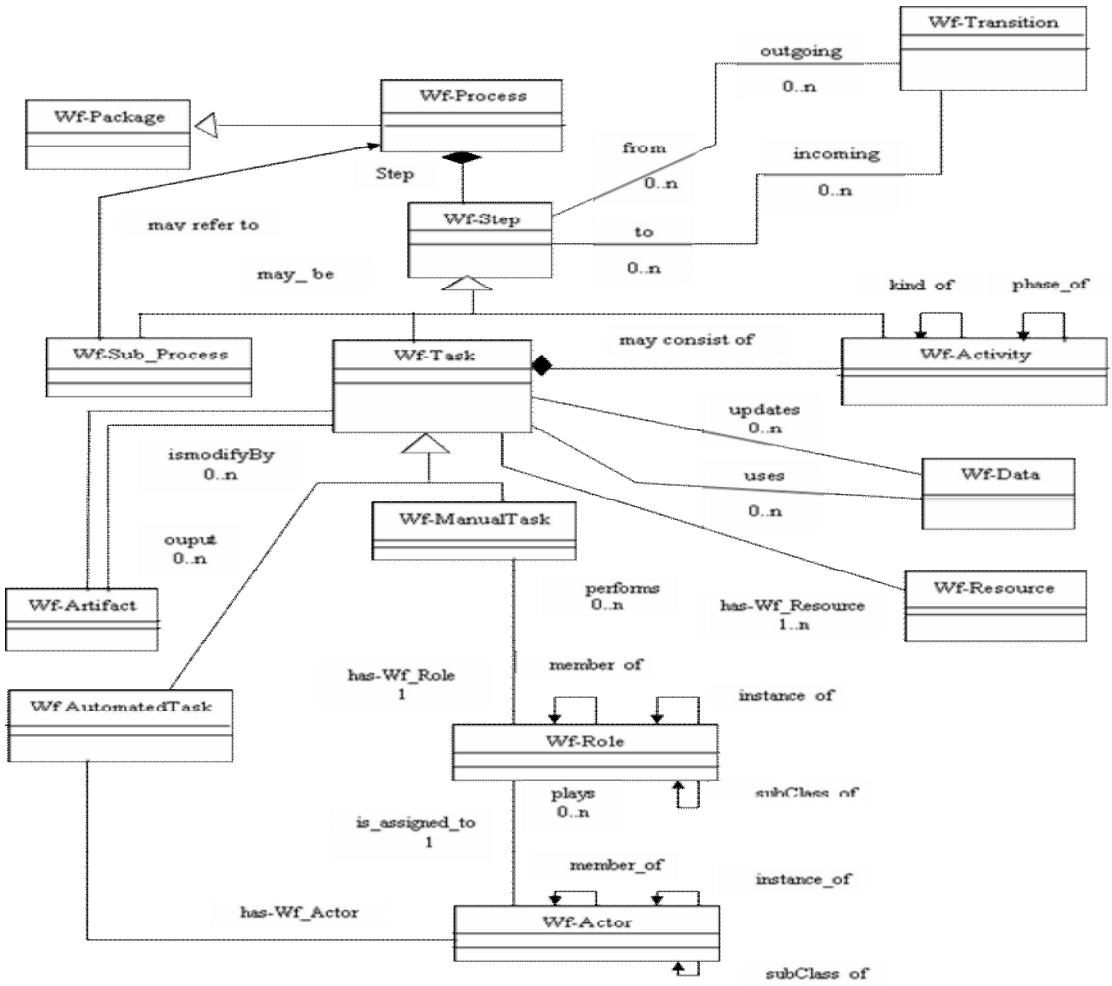

Figure 4.14 : Le méta-modèle commun de processus Workflow.

- **L'ontologie commune de processus Workflow**

Comme il a été évoqué plus haut dans la section 3.2 que la notion de méta-modèle est fortement liée à la notion d'ontologie et de ce fait, nous considérons notre méta-modèle commun (MMCPW) comme une ontologie commune de processus Workflow (OCPW) que nous implémentons comme une ontologie de Workflow décrite en OWL [131], [132], en utilisant l'outil ontologique Protégé [191].

Puisque nous utilisons OCPW comme un moyen pour annoter la sémantique des concepts des différents méta-modèles de processus, alors nous devons décrire les correspondances des concepts de ces méta-modèles vers ceux de l'ontologie commune OCPW.

- **Correspondance des concepts de méta-modèles de processus vers ceux d'OCPW**

Pour faire correspondre les concepts des différents méta-modèles de processus vers ceux de l'ontologie commune OCPW, nous nous référons au tableau de correspondance (table 4.3) qui

contient la plupart des concepts des méta-modèles de processus tels que ebXML, XPDL, BPML.

Les règles de correspondance ou de mapping concernent :

- Le mapping un à un : un concept OCPW (par exemple, OCPW : Wf-Transition) qui est un concept atomique, réfère à un concept atomique (par exemple, XPDL : TransitionInformation) ;
-
- Le mapping un à plusieurs : un concept OCPW (par exemple, OCPW : Wf-Transition) qui est un concept atomique, peut référer à un concept énuméré contenant plusieurs concepts (par exemple, ebXML : Transition, ebXML :Join, ou ebXML :Fork)

Cependant ces règles de mapping peuvent être complexes telles que : la correspondance entre un concept atomique de l'ontologie OCPW et un concept composite d'un certain méta-modèle de processus (ou entre un concept composite de OCPW et un autre concept composite d'un certain méta-modèle. Ces règles de mappings seront codifiées en XML Schéma.

La figure suivante montre un exemple d'une codification en XML pour annoter les équivalences sémantiques entre le concept d'activité de OCPW, de XPDL, et d'ebXML et qui sont respectivement 'Wf-Activity, '*WorkflowActivity*' et *('BusinessActivity', 'CollaborationActivity', 'BusinessTransactionActivity')* d'ebXML.

```
<semAnno : Concept rdf : resource= "uri://OCPW_Ontology#Wf-Activity"/>
< same_as
    <semAnno: AtomicConcept
          rdf:resource=" uri://XPDL_Ontology#WorkflowActivity"/>
    </semAnno:AtomicConcept>
    <semAnno:EnumeratedConcept
          <semAnno:has>
               <semAnno: AtomicConcept
                   rdf:resource="uri://ebXML_Ontology#BusinessActivity"/>
               </semAnno:AtomicConcept>
               <semAnno: AtomicConcept
                    rdf:resource  ="uri://ebXML_Ontology#CollaborationActivity"/>
               </semAnno:AtomicConcept>
              <semAnno: AtomicConcept
                    rdf:resource  ="uri://ebXML_Ontology#BusinessTransactionActivity"/>
             </semAnno:AtomicConcept>
          </semAnno:has>
    </semAnno:EnumeratedConcept>
.....
</same_as>
 </semAnno : Concept>
```

Figure 4.15 : Annotation sémantique des concepts d'activité de XPDL, d'ebXML par OCPW.

- **Formalisation du MCASMP**

En utilisant les concepts ontologiques fournies par OCPW et par l'ontologie du domaine, nous pouvons définir le MCASMP comme étant un ensemble d'éléments (activités, rôles, acteurs, etc.) que nous pouvons formaliser de la manière suivante :

MCASMP = (AV, AR, AC, AF, AE, AS, AT, RS, DO)

où : AV est un ensemble d'activités composant le processus où AVi est une activité;
AR est un ensemble de rôles qui interagissent dans le processus où ARi est un rôle ;
AC est un ensemble d'acteurs qui participent dans le processus où ACi est un acteur ;
AF est un ensemble d'artefacts qui sont utilisés dans le processus où AFi est un artefact;
AE est un ensemble de paramètres d'entrées (inputs) et AEi représente les inputs de l'activité AVi ;
AS est un ensemble de paramètres de sorties (outputs) et ASi représente les outputs de l'activité AVi ;
AT est un ensemble de conditions de transitions pour l'exécution du processus où ATi dénote les conditions de transitions de l'activité AVi ;
RS est un ensemble de ressources qui sont invoquées pendant l'exécution du processus où RSi dénote les ressources utilisées par l'activité AVi ;
DO est un sous-ensemble des concepts de l'ontologie du domaine (Ontology Domain).

Dans l'ontologie commune OCPW, une activité est un fragment de modèle qui est modélisée comme une étape d'un modèle de processus, pouvant être un sous-processus ou une activité atomique (tâche).

- Une activité quelconque (AVi) d'un modèle de processus (MP_l) est décrite comme suit :

AVi = (id, name, same_as, equivalent_name, has_Wf-Role, has_Wf-Actor, has_Wf-Artifact, has_Wf-Input, has_Wf-Output, has_Wf-Transition, has_Wf-Resource, kind_of, phase_of).

Chaque élément dans le méta-modèle MCASMP a un identificateur (*id)* et un nom (*name)* pour identifier de manière unique l'élément. '*Same-as'* est utilisé pour annoter l'élément 'Wf-Activity' avec son concept équivalent défini au niveau de l'ontologie du domaine (niveau concept) et *'equivalent_name*' fournit l'équivalent (ou synonyme) du nom défini dans thésaurus (niveau terminologie).

Les relations (*has_Wf-Role, has_Wf-Actor, has_Wf-Artifact, has_Wf-Input, has-Wf-Output, has_Wf-Transition, has_Wf-Resource)* sont utilisées pour annoter les relations sémantiques entre l'élément annoté (Wf-Activity) avec les autres concepts qui lui sont liés. Ces concepts sont définis dans l'ontologie commune OCPW et sont dénotés par des URI (Uniform Resource Identifier) dans le méta-modèle commun d'annotation sémantique de processus (MCASMP).

Quant aux relations sémantiques ('*kind_of', 'phase_of'*), elles nous permettent d'annoter les activités avec l'ontologie du domaine, c'est-à-dire, en les reliant aux concepts définis dans l'ontologie du domaine de référence. Toutefois, nous pouvons utiliser d'autres relations sémantiques telles que *'is_as', 'step_of', 'subClass_of', 'instance_of'*. correspondant à des relations d'agrégation, de généralisation ou de classification pour mieux enrichir l'annotation sémantique des relations entre l'activité et l'ontologie de domaine.

Une fois les concepts du MCASMP sont identifiés, il ne reste plus qu'à les formaliser. La description et la formalisation de ces concepts sont déjà faites dans un contexte homogène (section 2.2). Néanmoins, il faut ajouter dans la formalisation de ces concepts les éléments suivants :

- Trois relations : *'member_of', 'subClass_of' et 'instance_of'* qui sont utilisées pour annoter les relations sémantiques entre les concepts de rôle, d'acteur et ceux définis dans l'ontologie du domaine.

- Deux relations : *'kind_of', et 'part_of'* qui sont utilisées pour annoter les relations sémantiques entre le concept d'artefact et celui défini dans l'ontologie du domaine.

- les éléments : '*same_as' et 'equivalent_name*' qui nous permettent d'annoter respectivement l'activité en faisant référence à l'ontologie du domaine (niveau concept) et à un thésaurus (niveau terminologique) pour mieux comprendre sa signification.

3.4 Annotation sémantique des modèles de processus

Les concepts des méta-modèles de processus étant annotés par ceux de l'ontologie OCPW. Dans ce cas, ils sont remplacés par les concepts de cette ontologie, et par conséquent, leurs modèles respectifs sont alors représentés par cette ontologie. Cette dernière contient les concepts tels que Wf-Activity, Wf-Rôle, Wf-Transition. qui sont considérés comme des concepts abstraits et des mots-clés pour les modèles de processus. Par conséquent, pour annoter ces modèles, il faut les lier à l'ontologie du domaine qui est partagée par tous les acteurs coopératifs afin de résoudre l'hétérogénéité sémantique des contenus des modèles de processus. De même, pour éviter des ambiguïtés d'interprétation de ces modèles, nous proposons de les annoter en faisant référence à un thésaurus comme une forme d'ontologie afin de mieux expliciter les termes en nous fournissant leurs équivalents (ou leurs synonymes).

Par exemple, si nous disposons de deux modèles de processus concernant tous les deux le domaine de réservation de voyages. Dans le premier modèle *M1*, un concept appelé *'client'* qui est annoté par le concept de rôle *'Wf-Rôle'* défini par l'acteur *Act1*, alors que le concept nommé *'customer'* est aussi annoté par le même concept *'Wf-Rôle'* défini par l'acteur *Act2* dans un autre modèle *M2*. En mappant ces deux concepts vers un seul concept *'Wf-Rôle'* dans l'ontologie commune de processus Workflow (OCPW), nous saurons que ces deux concepts sont reliés. Néanmoins, nous avons besoin en plus, d'annoter les contenus de ces modèles par un même concept de l'ontologie du domaine référence (supposé 'client', par exemple) afin que les deux acteurs *Act1* et *Act2* puissent mieux interpréter et de la même manière ces deux modèles. Ainsi, l'annotation des modèles nous facilite la découverte sémantique et la navigation des fragments de modèles, qui sont des parties de ces modèles de processus.

L'annotation des modèles est donc effectuée avec l'aide du méta-modèle MCASMP en se référant à une ontologie du domaine et au thésaurus qui est un type d'ontologie pour faire référence aux termes synonymes utilisés dans le domaine Workflow. L'ontologie du domaine est choisie par les experts métiers pour annoter leurs modèles de processus. Dans notre travail, nous utilisons l'ontologie de tâches SCOR (Supply-Chain Operation Reference-model) [226] qui contient des concepts d'activités et de tâches pour notre domaine d'application de processus Workflow. Cette annotation consiste à décrire les correspondances des concepts des modèles de processus vers ceux de l'ontologie du domaine SCOR.

❖ **Correspondance des concepts des modèles vers ceux de l'ontologie du domaine**

Plusieurs stratégies de correspondance existent entre les concepts des modèles de processus et ceux de l'ontologie du domaine SCOR. Elles peuvent être :

- Par simple référence : les concepts des modèles de processus se réfèrent directement à des concepts de l'ontologie du domaine, en utilisant, la relation sémantique '*refers_to'*. Evidemment, cela suppose que presque tous les concepts des modèles de processus ont approximativement des concepts équivalents dans l'ontologie du domaine.

- Par des relations sémantiques raffinées : les concepts des modèles de processus sont généralement très variés et sont définis initialement pour différents objectifs. Cependant, il est difficile de trouver exactement des concepts définis dans l'ontologie du domaine pour ces modèles. Toutefois, nous pouvons créer ou raffiner certaines relations sémantiques entre les concepts des modèles et ceux de l'ontologie du domaine. Parmi ces relations sémantiques, nous pouvons représenter des relations de type : '*kind_of', 'step_of', 'phase_of', subCass_of'* afin de lier le concept d'activité *'Wf-Activity'* avec celui de l'ontologie du domaine. Quant aux relations sémantiques de type : '*member_of', 'is_as', 'instance_of',* elles sont définies pour lier le concept de Rôle 'Wf-Role' avec celui de l'ontologie du domaine.

La stratégie de correspondance par simple référence est simple à utiliser et facile à appliquer par rapport à la deuxième stratégie qui est complexe.

La figure 4.16 montre un schéma commun d'annotation sémantique de deux modèles de processus. Chaque modèle est représenté par son méta-modèle correspondant et lié à son ontologie.

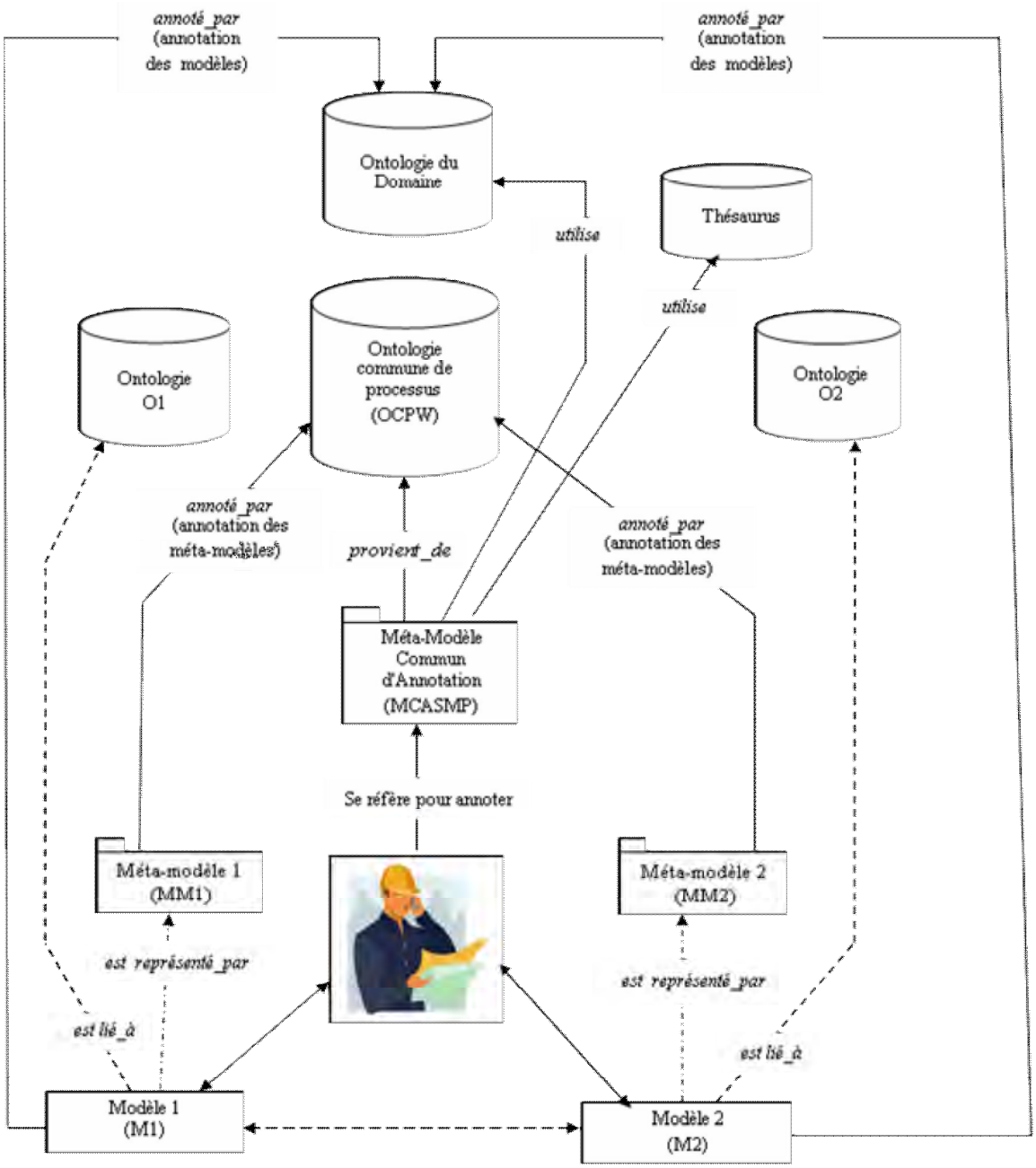

Figure 4.16 : Schéma commun d'annotation sémantique des modèles de processus.

Cependant, dans une perspective de coopération orientée buts et pour des objectifs de découverte sémantique et de réutilisation des modèles dans d'autres projets spécifiques, nous raffinons ces annotations sémantiques par d'autres types d'annotations tels que :

- Annotations sémantiques des profils des modèles de processus : il s'agit d'annoter ce que font les modèles de processus afin de les découvrir sémantiquement par leurs propriétés fonctionnelles (inputs outputs, etc.) et par leurs propriétés non fonctionnelles (domaine d'application, catégorie, etc.).

- Annotations sémantiques des buts des modèles de processus : il s'agit d'annoter les objectifs ou les buts que doivent atteindre les processus afin de les découvrir sémantiquement par leurs objectifs métiers.

3.5 Annotation sémantique des profils des modèles de processus

Le profil d'un modèle de processus décrit ce que fait le modèle. L'annotation du profil d'un modèle de processus consiste alors à annoter sémantiquement les caractéristiques fonctionnelles offertes par le modèle qui sont représentées par les entrées (inputs), les sorties (outputs), les pré-conditions, et les post-conditions d'un modèle de processus. D'autres caractéristiques non fonctionnelles sont décrites dans un modèle tels que le nom du modèle, une description textuelle du modèle, la catégorie. Ces annotations sont parfois nécessaires pour des objectifs de découverte contextuelle d'un modèle de processus.

La figure suivante montre le méta-modèle d'un profil de modèle de processus contenant uniquement les propriétés fonctionnelles.

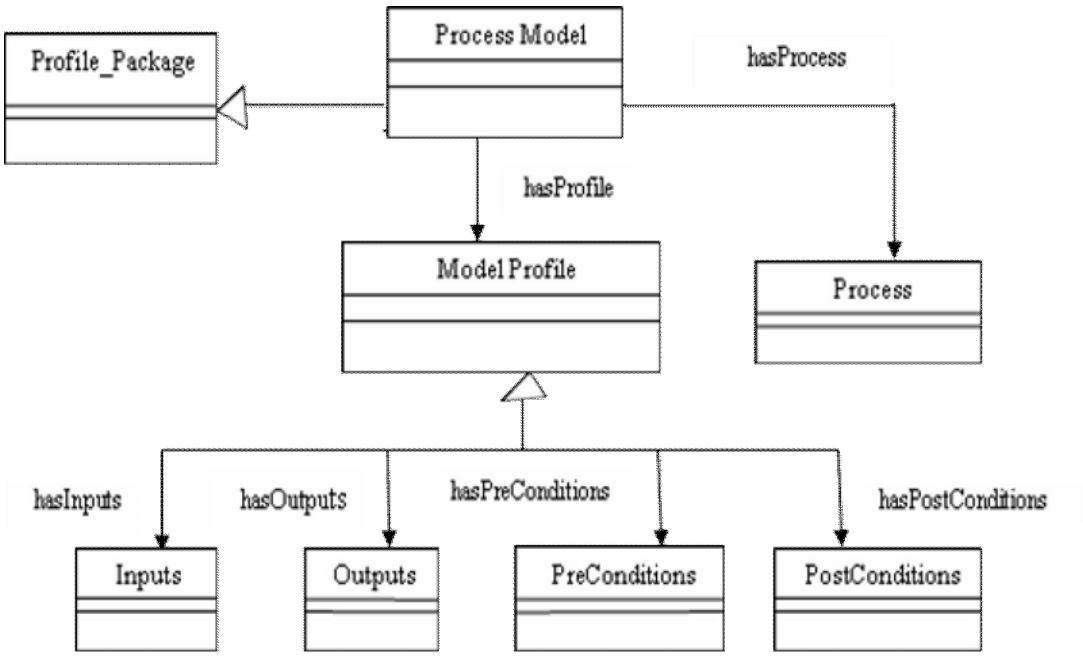

Figure 4.17 : Le méta-modèle d'un profil de modèle de processus.

Ce méta-modèle est alors considéré comme une ontologie et sera implémenté en utilisant l'outil Protégé [191]. Puis, il sera sauvegardé afin qu'il puisse être invoqué pour être réutilisé par des partenaires coopératifs. Le principe d'annotation d'un profil de modèle de processus est illustré par le schéma suivant (figure 4.18).

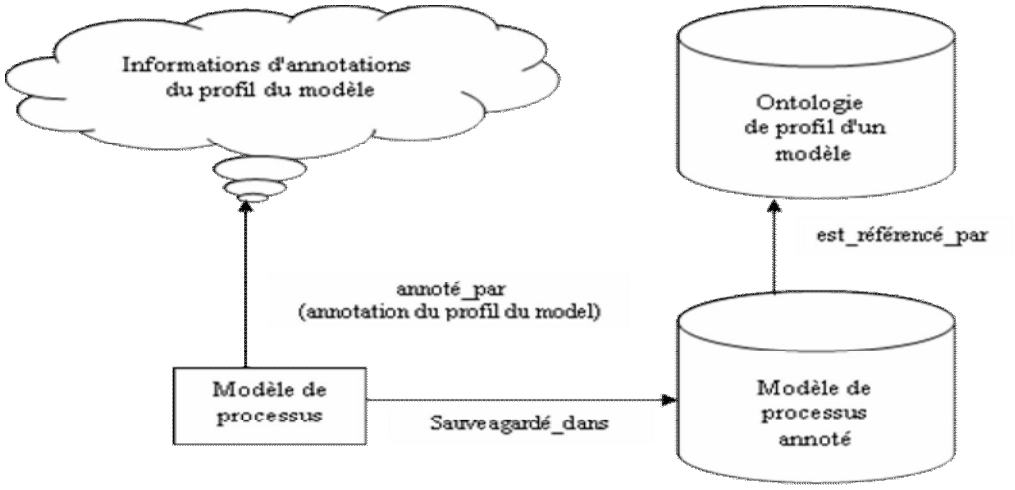

Figure 4.18 : Principe d'annotation d'un profil de modèle de processus.

Les modèles de processus sont annotés sémantiquement en se réfèrent à leurs ontologies de profils via la relation sémantique *'has_Wf-Profile'*. Ainsi, notre méta-modèle d'annotation sémantique des modèles de processus (MCASMP) est étendu pour prendre en compte ce type d'annotation. Il sera alors enrichi par l'ajout d'un sous-ensemble des concepts de l'ontologie des profils (Profile Ontology**) (PO)**, et sera formalisé de la manière suivante :

MCASMP = (AV, AR, AC, AF, AE, AS, AT, RS, DO, PO).

Et une activité quelconque (AVi) d'un modèle de processus (MP_I) est décrite comme suit :

AVi = (id, name, same_as, equivalent_name, has_Wf-Role, has_Wf-Actor, has_Wf-Artifact, has_Wf-Input, has_Wf-Output, has_Wf-Transition, has_Wf-Resource, kind_of, phase_of, has_Wf-Profile).

La figure 4.19 montre un scénario d'annotation sémantique des profils des modèles de processus.

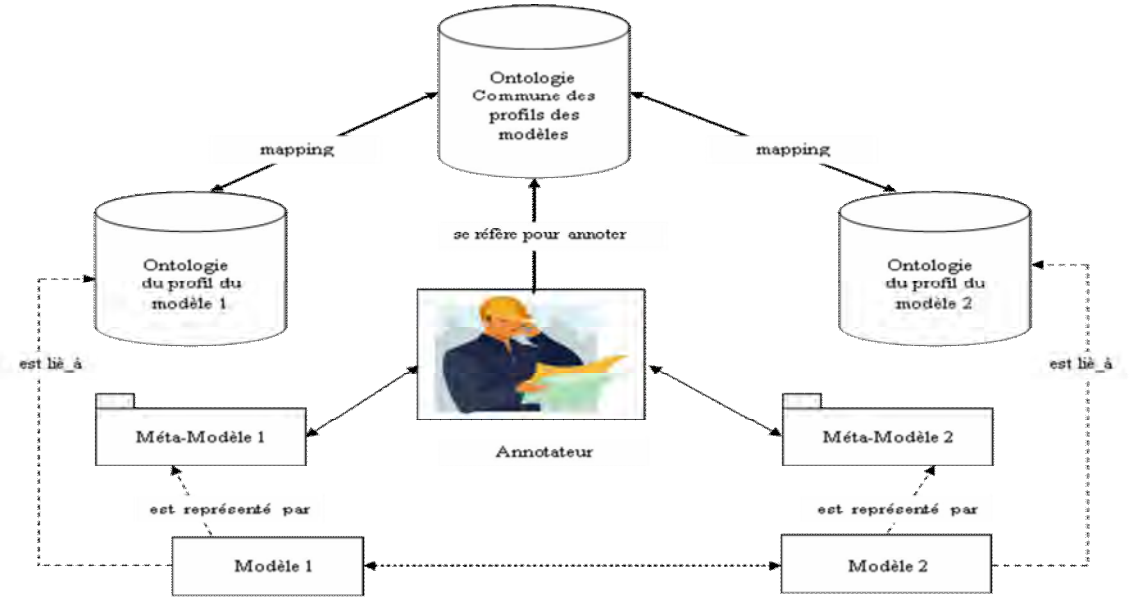

Figure 4.19 : Scénario d'annotation sémantique des profils des modèles de processus.

3.6 Annotation sémantique des buts des modèles de processus

Les systèmes de gestion de Workflow destinés à automatiser des processus sont conçus généralement pour atteindre des objectifs spécifiques qui sont associés aux processus métiers. La notion d'*objectif* (ou *de but*) appelé '*Goal*' en anglais, est fondamentale pour la modélisation de ces processus. Bien qu'un processus soit orienté vers la réalisation de certains objectifs ou buts, la relation entre les buts et les procesus n'est pas explicitement représentée dans beaucoup de processus.

Certes, l'analyse des buts est initialement séparée de la modélisation des processus, mais il existe une relation entre les buts et les activités. Comme décrit dans [229], les buts (états désirés du monde) et les activités (actions effectuées pour réaliser un état particulier) sont clairement définis. Cependant, l'analyse en parallèle des activités et des buts nous permet de décomposer un but en sous-buts afin qu'il soit réalisé et sa décomposition est faite en des activités atomiques pour être effectuées.

Certains langages de modélisation de processus et outils supportent la modélisation des buts comme une partie de la modélisation des processus, tel que EEML (Extended Enterprise Modeling Language) [218] qui est implémenté dans l'outil Metis [219]. Ainsi, un ensemble de buts peut être modélisé localement pour être lié à des modèles de processus.

L'annotation des buts des modèles de processus ou des fragments de modèles (activités) nous permet d'aligner la représentation sémantique des buts pour qu'ils soient compréhensibles et interprétables par la machine. Elle est basée sur une ontologie de buts afin de spécifier les capacités et les objectifs des processus. Cette ontologie correspond donc, à un ensemble de concepts et de relations sémantiques entre ces concepts.

La création d'une ontologie globale des buts nous permet de définir la sémantique associée aux buts ou aux objectifs assignés par les processus métiers d'entreprise. Elle nous permet de construire des relations entre les buts locaux et les buts globaux qui sont définis dans l'ontologie globale. Ainsi, deux types de relation sont créées pour dénoter que l'activité peut réaliser des buts: une relation de type *'achieves_local_but'* entre l'activité et le but local, et une autre relation de type *'achieves_global_but'* entre l'activité et le but global. Ce qui signifie évidemment que le méta-modèle du processus doit au préalable intégrer le concept de but local pour nous faciliter ainsi, l'annotation.

Plusieurs approches de modélisation orientées buts [220], [221], [222], [223] ont été proposées dans la littérature pour définir le concept d'ontologie de buts. KAOS (Knowledge Acquisition in autOmated Specification) [224] et i*/GRL (Goal-oriented Requirement Language) sont considérées dans la communauté de l'intelligence artificielle, et en particulier dans l'ingénierie des besoins, comme les approches orientées buts les plus significatives permettant de modéliser les objectifs. Ces deux approches s'appliquent dans les systèmes d'information associés à des environnements organisés tels que les processus d'entreprise. Les concepts définis dans KAOS [224] sont : objet, action, agent, but, contrainte. Par contre, ceux de i*/GRL [225], sont les suivants : acteur, rôle, position, but.

Pour exprimer des buts, [220] définit des verbes mots-clés comme ceux utilisés par KAOS tels que : achieve, avoid, maintain. En fait, l'utilisation de ces verbes débouche sur une classification des buts : buts hard et buts soft [221], [230]. Les buts hard concernent des buts fonctionnels qui sont supportés par les processus. Un besoin fonctionnel définit un besoin

potentiel que le système doit satisfaire. Les buts soft sont liés à des besoins non fonctionnels concernant les qualités supplémentaires attendues telles que performance, qualité de service, coût et délai.

Cependant, les ontologies des buts appartiennent à la classe des ontologies des tâches qui décrivent des tâches ou des activités visant des objectifs à atteindre. Dans ce contexte, les ontologies des tâches sont considérées comme des ontologies des buts. Par exemple, l'ontologie des tâches SCOR contient des concepts de buts [226], [227] qui sont formalisés par des buts hard et des buts soft. La figure suivante montre un scénario d'annotation sémantique des buts des modèles de processus.

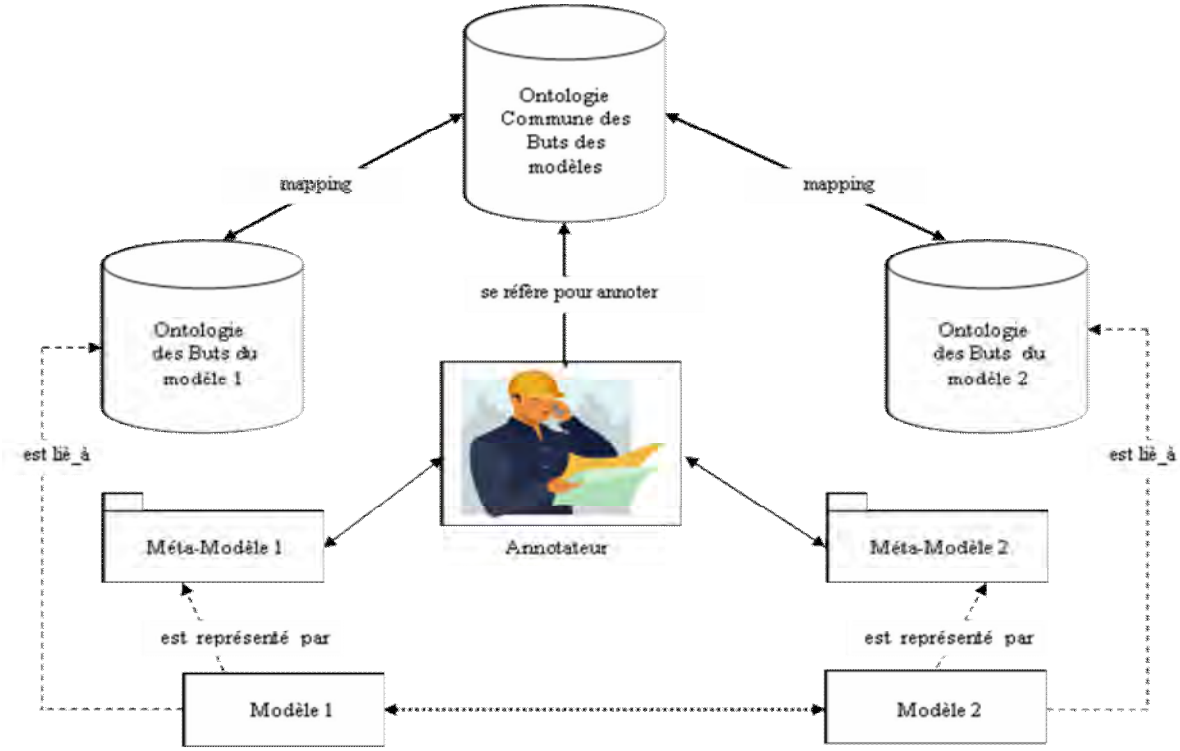

Figure 4.20 : Scénario d'annotation sémantique des buts des modèles de processus.

❖ **SCOR comme ontologie du domaine de référence**

SCOR (Supply Chain Operations Reference model) ou plutôt le modèle SCOR [226] a été développé par le Supply Chain Council (SCC) qui est une organisation sans but lucratif regroupant des entreprises les plus performantes. Les membres de cette organisation ont mis en exergue qu'il n'existe pas de différence entre une entreprise industrielle et une entreprise délivrant des services : le point commun à tout modèle économique est le client. En effet, il n'existe pas de gestion de chaîne logistique ou d'approvisionnements sans client (ou SCM pour Supply Chain Management).

Le but de cette organisation représentée sur 4 continents est de structurer un référentiel de processus logistiques types et de mettre en évidence par la même occasion les critères de performance, les indicateurs et les meilleures pratiques. Le conseil de la chaîne d'approvisionnements (SCC) est structuré en groupes de travail, chaque groupe ayant la responsabilité d'un domaine fonctionnel (achat, fabrication, livraison, retours, mesure).

Basé sur ce postulat, le modèle SCOR sert, à ce jour, de référence à de multiples secteurs industriels et de services dans le monde (chimie, agroalimentaire, électronique, grande distribution, prestations logistiques). En outre, de par sa structure complète, ce modèle est devenu un standard de fait sur le marché. Le modèle SCOR présume que toute chaîne logistique peut être subdivisée en 5 types de processus : planification (Plan), approvisionnement (Source), fabrication (Make), livraison (Deliver) et gestion des retours (Return).

Le modèle SCOR s'organise autour des besoins du client (commandes, réclamations, demandes d'informations), et décrit les activités métiers avec toutes les phases d'une requête de commande d'achat. Il recouvre les processus impliqués dans :

- ✓ les interactions avec le client depuis la réception de la commande jusqu'au paiement de la facture,
- ✓ les échanges depuis le client jusqu'au fournisseur,
- ✓ les interactions liées à la demande depuis son analyse jusqu'à l'exécution de chaque commande.

Pour annoter sémantiquement nos modèles de processus, nous nous appuyons dans notre travail sur des référentiels métiers existants, notamment le modèle formalisé SCOR comme une ontologie de référence pour notre domaine d'application Workflow..

Il y a trois niveaux de détails des processus dans le modèle de référence. Le niveau supérieur définit la portée et le contenu de SCOR. Le deuxième niveau est le niveau "catégories de processus. Le niveau 3 du référentiel est le niveau "processus". Ce niveau décompose les catégories de processus en des éléments de processus, en fournissant assez de détails de référence pour des activités du domaine. Dans notre travail, nous modélisons les concepts d'ontologie du domaine en se basant sur les processus de SCOR de niveau 3 pour des objectifs d'annotation des modèles.

L'ontologie SCOR est formalisée en OWL. Pour organiser ces concepts, nous classons les concepts du domaine en quatre catégories (Wf-Activity, Wf-Artifact, Wf-Actor, Wf-Role) définis dans notre ontologie commune de processus Workflow (OCPW). Le but de cette classification est d'établir la relation de mapping entre le méta-modèle MCASMP et l'ontologie de référence SCOR au moment de l'annotation. Les concepts sont modélisés comme des classes OWL et sont organisés en une hiérarchie de subsumption.

La figure 4.21 présente un exemple d'un processus de niveau 3 de SCOR concernant un produit stocké par approvisionnement ('S1 Source Stocked Product'). Pour notre cas, nous utilisons le modèle SCOR 'S1 Source Stocked Product' comme une ontologie de tâches pour notre processus de commande d'achat. Chaque élément d'un modèle de processus est un concept de l'ontologie de tâches qui est référencé par le concept d'activité 'Wf-Activity' du méta-modèle MCASMP.

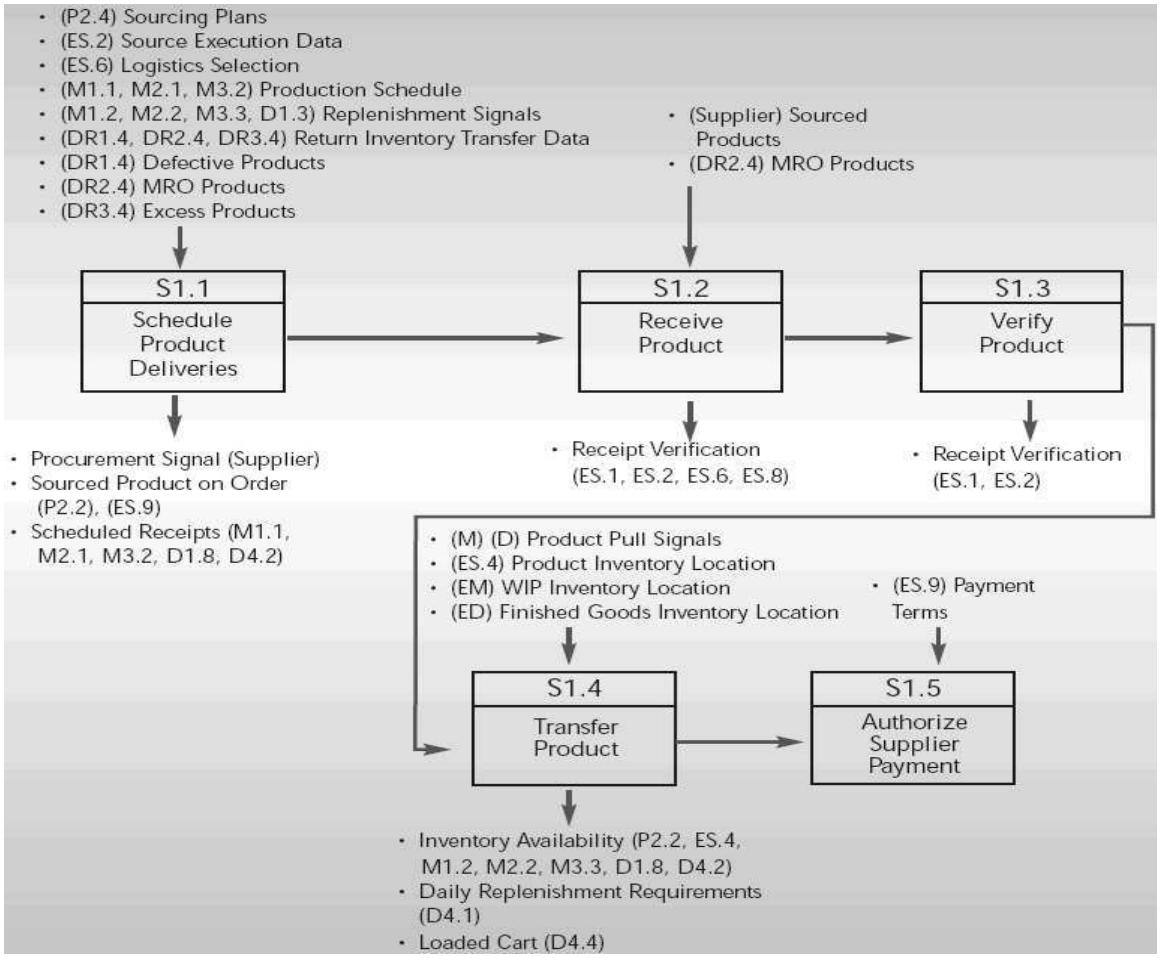

Figure 4.21 : Processus de SCOR S1 d'un produit stocké par approvisionnements [226].

Par exemple, les activités *'Check Order'* et *'Verify Order'* qui sont définies dans deux modèles de processus différents, sont toutes les deux annotées par une même activité qui est *'kind_of Wf-Activity : Verify.Product'* définie dans SCOR qui est pour nous l'ontologie du domaine de référence spécifique à notre domaine d'application (processus de commande).

De même, les inputs et les outputs sont des concepts ontologiques objets du domaine utilisés pour annoter les artefacts dans le MCASMP. Par exemple, l'artefact 'Purchase Order' ou simplement 'Order' est annoté avec l'objet ontologique du domaine qui est *'Source Product On Order'*.

Pour bien illustrer notre technique d'annotation basée sur l'ontologie du domaine de référence SCOR, nous donnons un exemple simple d'annotation (figure 4.22) de deux activités élémentaires de création d'une commande d'achat :*'CreateOrder'* et *'GetOrderData'* qui sont définies respectivement dans les modèles de processus XPDL et ebXML. Ces deux activités sont toutes les deux annotées par le même concept 'Wf-Activity' de l'ontologie commune OCPW, d'une part, et par l'activité *'kind_of Wf-Activity : Shedule.Product Deliveries'* qui est définie dans SCOR, d'autre part. Pour plus de détails sur SCOR, le lecteur est invité à se référer à [226].

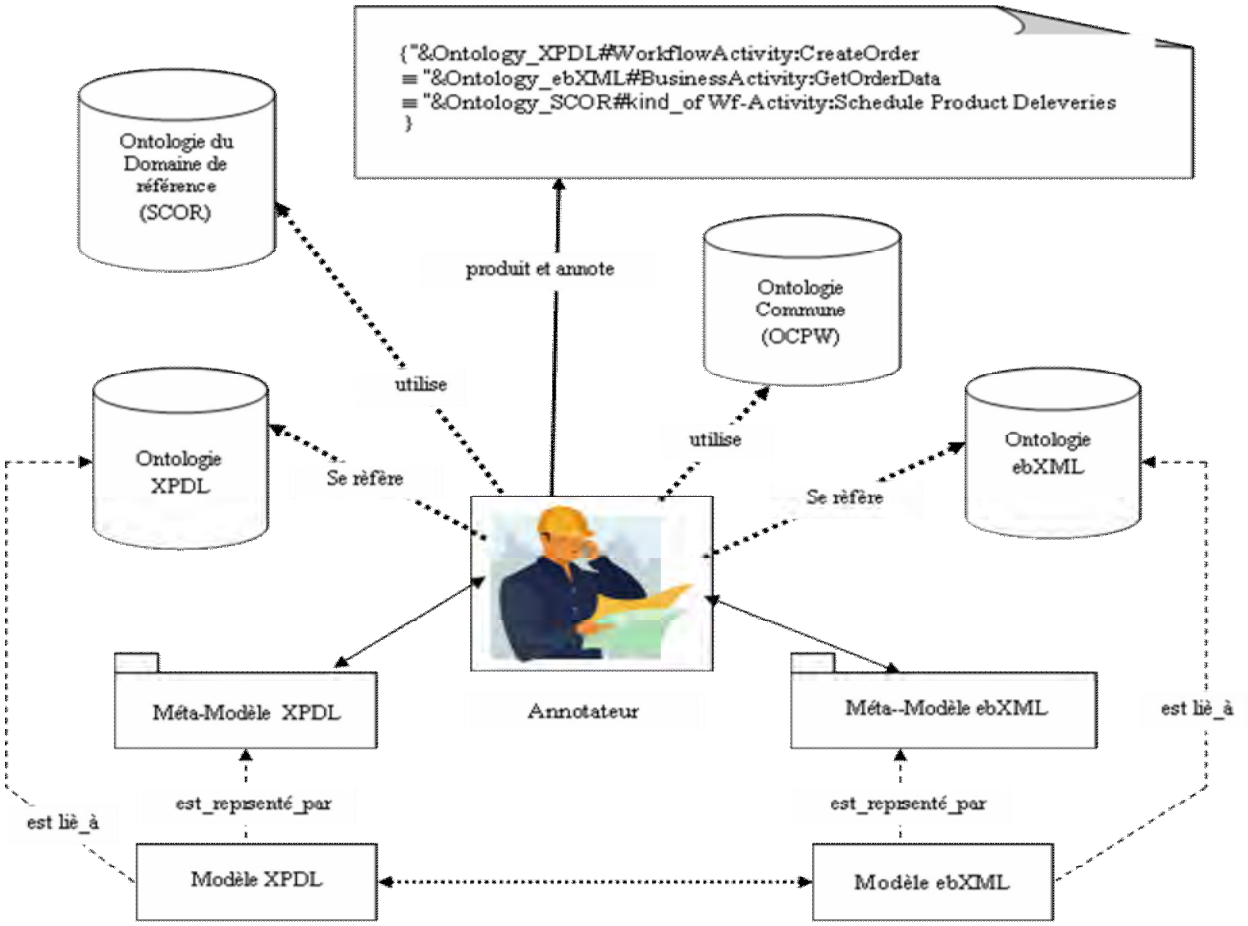

Figure 4.22 : Annotation sémantique d'une activité de "création d'une commande" d'achat.

❖ **SCOR comme ontologie des buts**

Puisque les ontologies des buts appartiennent à la classe des ontologies des tâches qui décrivent des activités ou des tâches visant des objectifs à atteindre. Dans notre contexte d'interopérabilité, nous considérons les ontologies des tâches comme des ontologies des buts. Ainsi, l'ontologie de buts pour notre domaine du Workflow est également SCOR qui contient des concepts de buts qui sont formalisés par des buts hard et des buts soft.

Souvent, les buts hard sont dérivés des processus et également de leurs inputs et outputs. Les attributs de performances définis dans SCOR peuvent être modélisés comme un ensemble de catégories de buts soft généraux tels que : la fiabilité, le temps de réponse, la flexibilité, le coût, et la performance. Des buts soft du domaine spécifique sont dérivés des attributs de mesures de performances d'exécution [226].

En analysant le modèle SCOR, plusieurs types de buts peuvent être déduits et extraits du processus de niveau 3. Quelques exemples de buts orientés activités (appelés buts hard) peuvent être cités tels que *'Sourced Product are On Order'; 'Order is Placed', 'Order is Validate', 'Order is Consolidated'; 'Order is Processed', 'Available Inventory', 'sourced products are verified','End Items are Delivered'.* De même, *qu'il* existe des buts orientés rôles tels que *"Procurement Notification to Supplier','Payment is Authorized to Supplier'.*

La table suivante montre un exemple d'annotation sémantique des activités d'un processus de commande d'achat avec les buts hard associés, qui sont basés sur l'ontologie SCOR

Activités du modèle de processus de commande d'achat	**Annotation sémantique des activités des modèles avec l'ontologie du domaine SCOR**	**Annotation sémantique des buts des activités des modèles avec l'ontologie des buts SCOR**
Create Order (ou GetOrderData)	phase_of Wf-Activity : Schedule Product Deliveries	Sourced Product are On Order; Order is Placed
Check Order	kind_of Wf-Activity : Verify Product	Sourced Product are Verified
Cancel Order	kind_of Wf-Activity : Verify Product	Reduce Order Processing Time; Reduce Order Processing Costs; Reduce Order Receipt Time; Reduce Verification Costs
Notify Customer	phase_of Wf-Activity : Manage Supplier Agreements	Procurement Notification to Supplier
Fill Order	kind_of Wf-Activity : Transfer Product	Order is Placed; Order is Processed; Order is Validated; Available Inventory
Ship Order	kind_of Wf-Activity : Manage Deliver Information	Delivery is Scheduled; Delivery Terms are Generated
Close Order	same_as Wf-Activity : Consolidate Orders	Order is Processed; Order is Consolidated

Table 4.4 : Annotation sémantique des activités et des buts basés sur l'ontologie SCOR.

Souvent dans un contexte d'interopérabilité orienté buts et d'intégration externe de processus métiers, les buts assignés aux modèles de processus sont liés à la qualité de service fournie au client, au délai de livraison et à la performance d'exécution du processus. Ces buts soft ont un impact direct sur l'exécution du processus métier qu'on peut énumérer par quelques verbes mots clés liés à la fiabilité, performance, délai, coût et qui sont par exemple : *"improve supplier delivery to date performance* (performance*)"; "invoices processed without error* (fiabilité*)", "Reduce Order Processing Time* (temps*)"; "Reduce Order Processing Costs* (coût)" *, "Reduce verification costs* (coût)"*; "Improve Supplier On Time* Delivery Performance" (délai)".

Par exemple, si deux buts locaux: *"Ensure_Supply_Delivery_to_Date_Performance*" et *'Improve_Deliver_to_Customer_On time_Delivery_Perfor*mance' sont définis respectivement dans les modèles XPDL et ebXML, alors nous pouvons les annoter tous les deux par le but global *"Improve_delivery_performance*" qui est défini dans l'ontologie commune des buts qui est SCOR.

L'ontologie SCOR étant choisie comme une ontologie globale de buts (Goal Ontology) et notre méta-modèle d'annotation sémantique de processus (MCASMP) sera alors étendu pour annoter les modèles (ou fragments de modèles) avec cette ontologie via la relation sémantique *'achieves_global_but'* . Il sera donc, enrichi par l'ajout d'un sous-ensemble des concepts de cette ontologie (GO), et sera formalisé de la manière suivante :

MCASMP = (AV, AR, AC, AF, AE, AS, AT, RS, DO, PO, GO)

Et une activité quelconque (AVi) d'un modèle de processus (MPi) sera décrite comme suit :

AVi = (id, , name, same_as, equivalent_name, has_Wf-Role, has_ Wf-Actor, has_Wf-Artifact, has_Wf-Input, has_Wf-Output, has_Wf-Transition, has_Wf-Resource, kind_of, phase_of, has_Profile, achieves_global_but).

Aussi, pour avoir une bonne compréhension des modèles de processus lors de l'échange et de la coopération, il est préférable d'annoter certains aspects qui sont intrinsèquement liés à ces modèles. Pour ce faire, nous devons d'abord, élaborer les méta-modèles communs correspondant à ces aspects, puis les regroupons sous forme de packages. Ensuite, nous les considérons comme des ontologies communes pour les implémenter en utilisant l'outil ontologique Protégé [191]. L'annotation sémantique de ces modèles du point de vue informationnel, fonctionnel ou organisationnel, est basée ainsi sur l'ontologie correspondant à chaque aspect de base. De ce fait, notre méta-modèle d'annotation sémantique de processus (MCASMP) sera encore étendu et enrichi pour la prise en compte de ces aspects. Les annotations sémantiques de ces aspects se réfèrent à un ensemble commun d'ontologies via la relation sémantique *'se réfère_à'*, comme le montre la figure suivante.

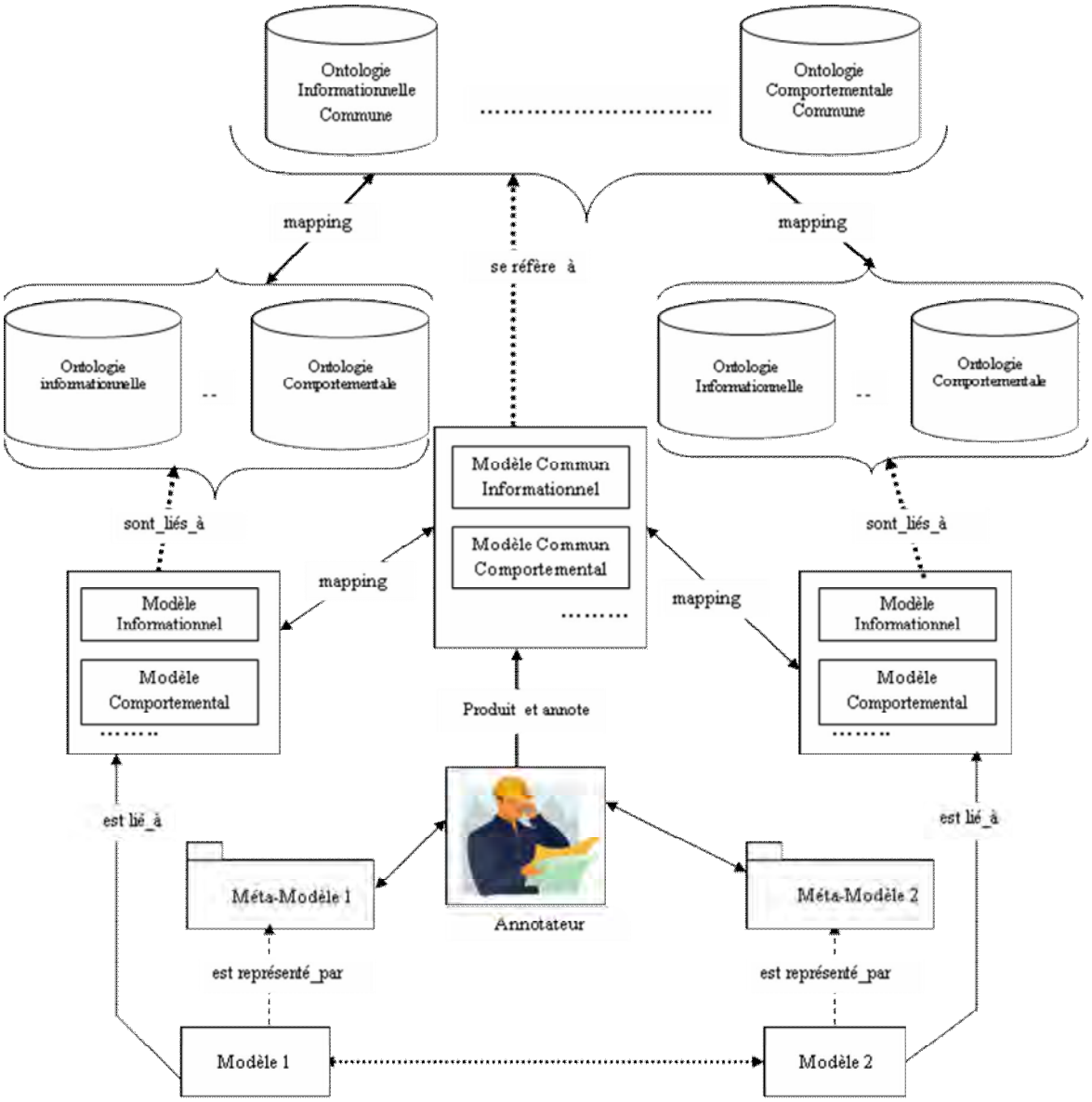

Figure 4.23 : Scénario d'annotation sémantique des aspects des modèles de processus.

Cet ensemble commun d'ontologies provient de plusieurs méta-modèles communs que nous devons construire. C'est l'objet du paragraphe suivant.

❖ Les méta-modèles communs de base

L'élaboration des méta-modèles communs est faite en collaboration par des experts métiers du domaine et par les acteurs coopératifs qui connaissent mieux le domaine du Workflow (ou busines process) afin de les aligner sémantiquement selon leurs concepts définis..

➢ Le méta-modèle commun informationnel

Ce méta-modèle décrit les artefacts (documents, feuilles de style, formulaires électroniques, etc.) qui sont utilisés par les activités, ainsi que les données manipulées pour l'exécution du processus (figure 4.24).

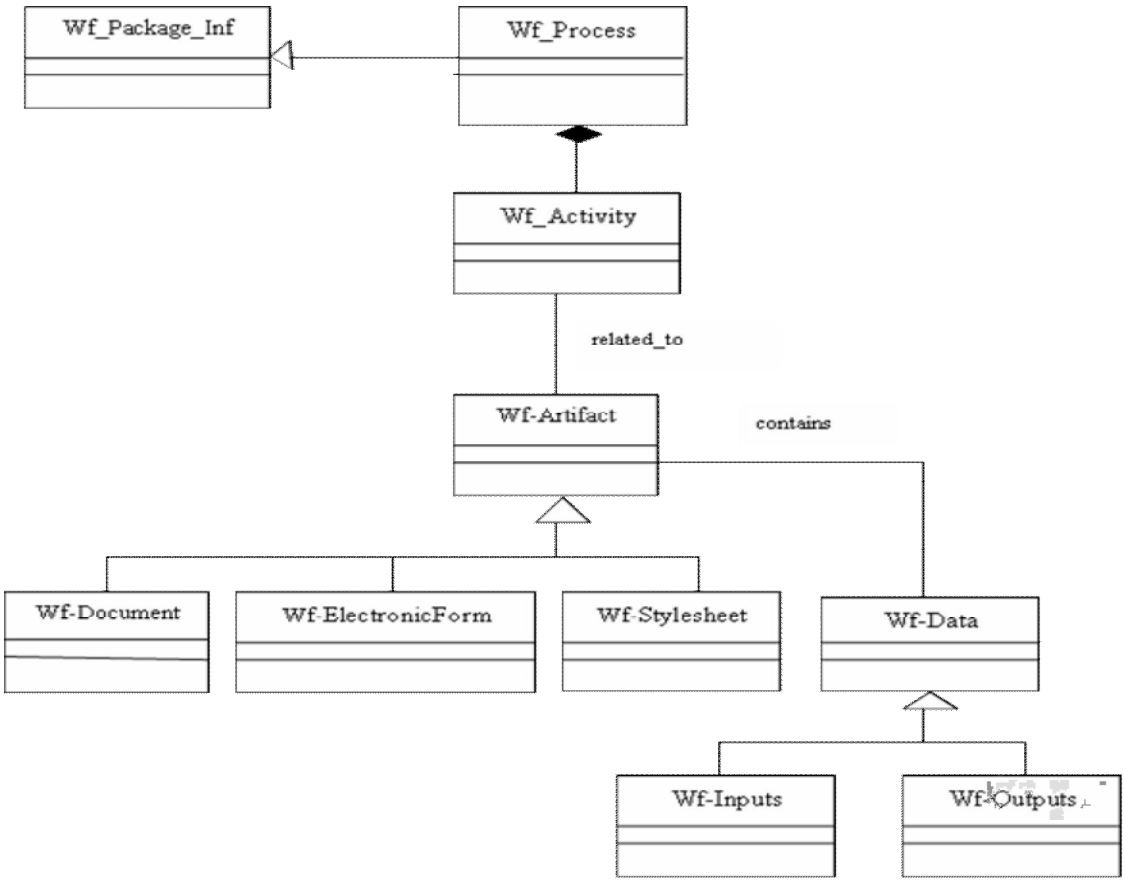

Figure 4.24.: Le méta-modèle commun informationnel.

❖ Le méta-modèle commun organisationnel

Ce méta-modèle décrit la structure organisationnelle de l'entreprise dont l'élément de base est l'unité organisationnelle qui peut correspondre à n'importe quelle entité structurelle d'entreprise (département, service, etc.) et qui regroupe tous les concepts organisationnels définis dans une structure d'entreprise tels que rôle, acteur, équipe (figure 4.25).

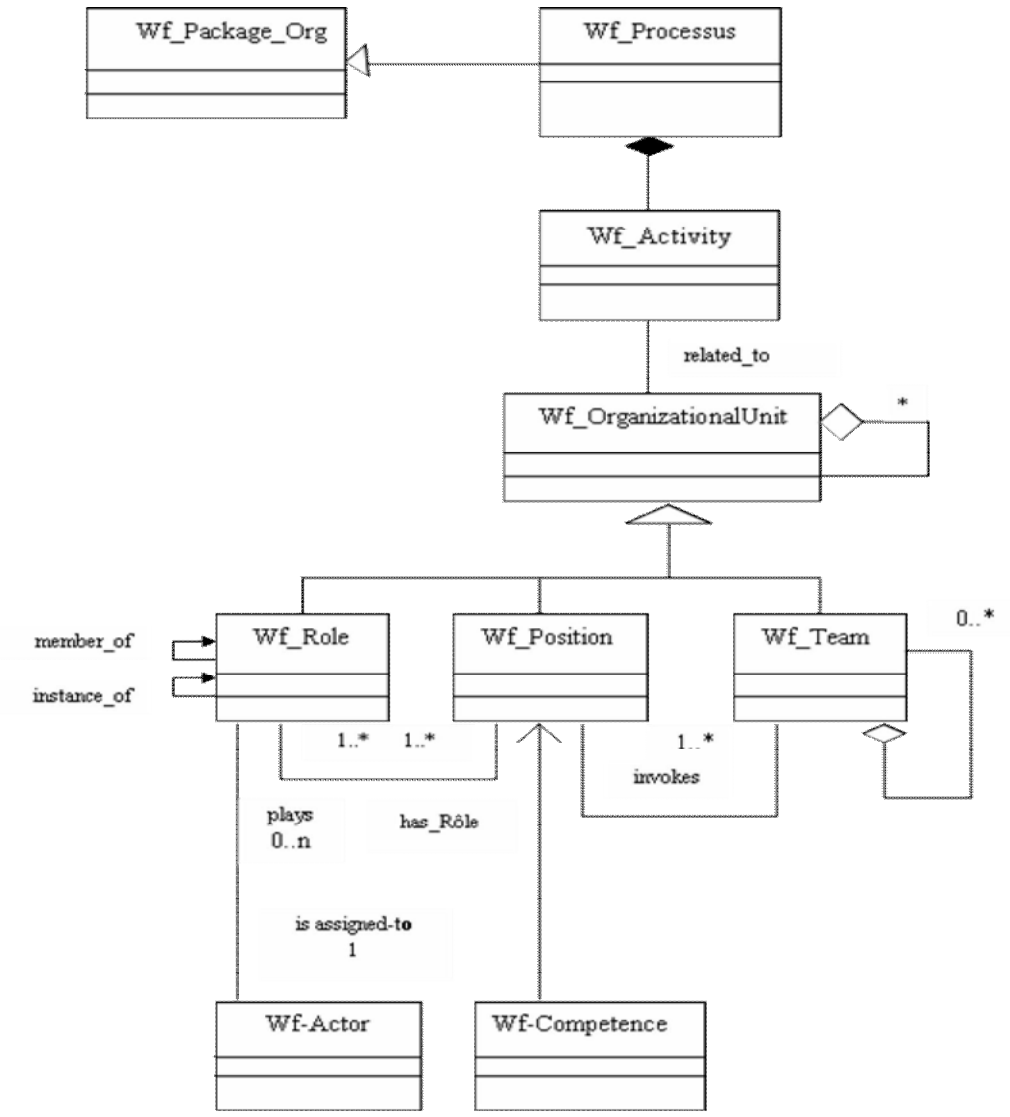

Figure 4.25 : Le méta-modèle commun organisationnel.

Pour exprimer le comportemental commun des modèles de processus, deux méta-modèles dynamiques sont définis. Le premier sert à spécifier le contrôle de flux et le deuxième s'intéresse à représenter les différents états/transitions passés par un processus ou une activité.

- **Le méta-modèle commun dynamique**

Ce méta-modèle s'intéresse en particulier à définir le contrôle du flux du processus s'exprimant globalement sous forme de séquences, de mise en parallèle et de branchement (figure 4.26).

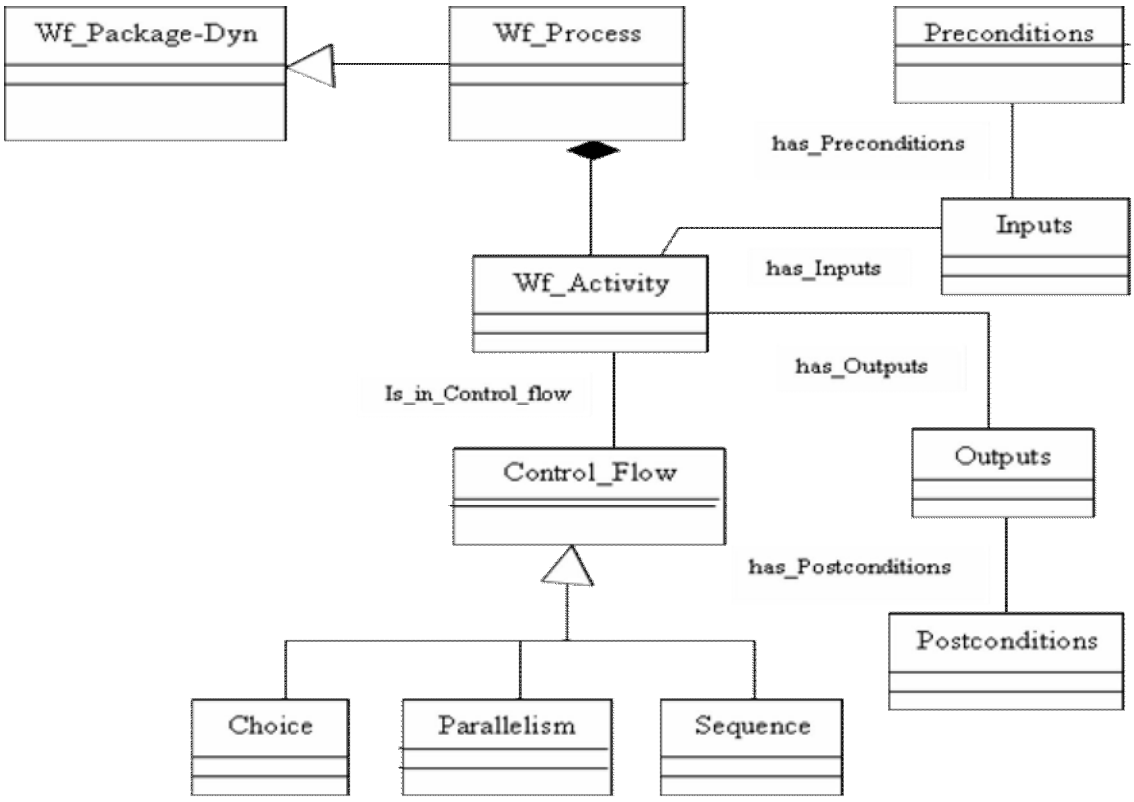

Figure 4.26 : Le méta-modèle commun dynamique (contrôle du flux).

Aussi, le comportement dynamique d'un processus ou d'une activité peut-être mieux exprimé à l'aide d'un diagramme d'état/transitions (figure 4.27) pour représenter les différents états d'un processus ou d'une activité. Dans notre approche, le méta-modèle commun du diagramme d'état/transitions est inspiré du diagramme d'état/transitions du méta-modèle UML dont les concepts de base sont : StateMachine, State, Transition, Event, Guard et dont les contraintes sont exprimées et formalisées par le langage OCL (Object Language Constraint) associé à UML.

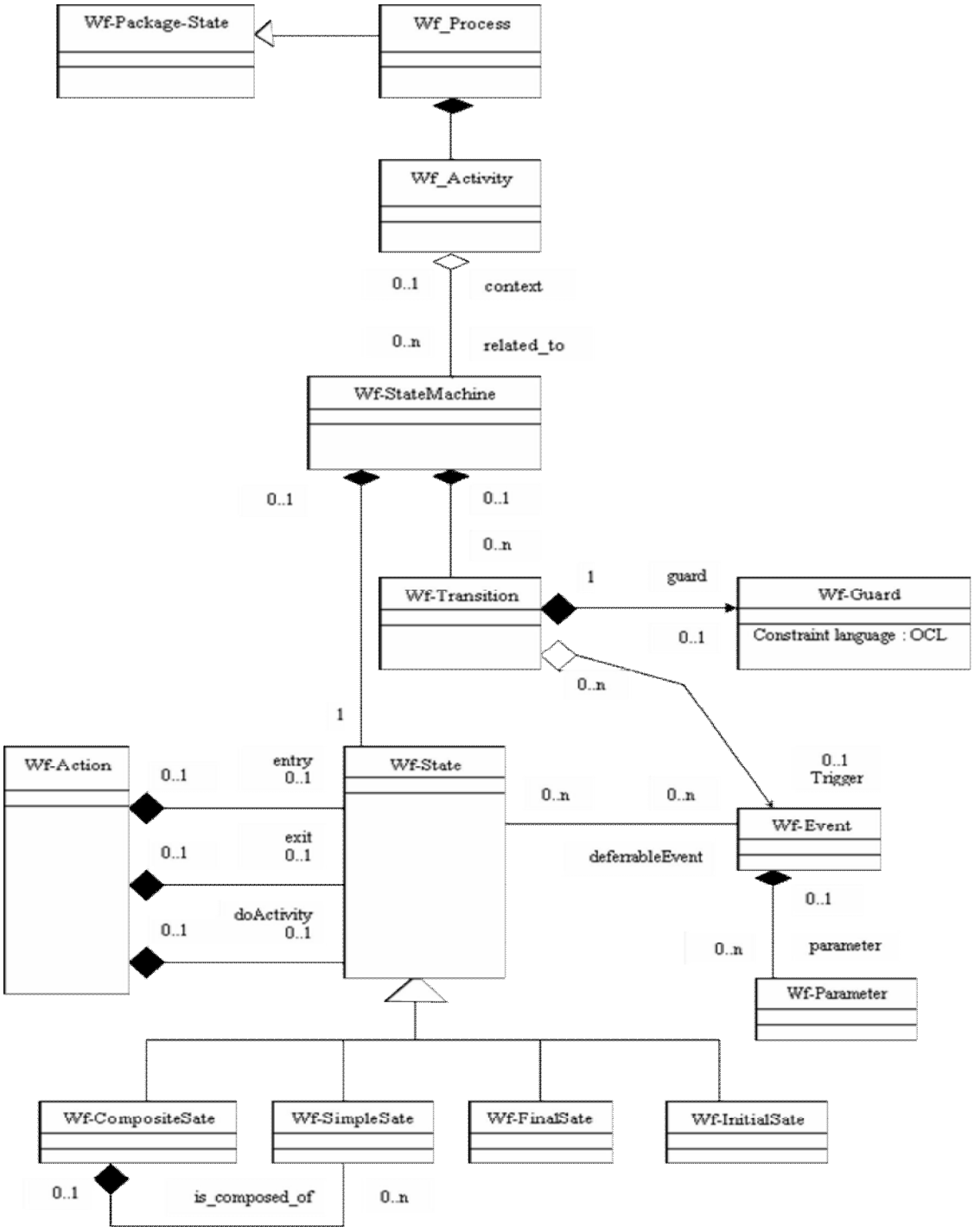

Figure 4.27 : Le méta-modèle commun dynamique (états/transitions).

- **Le méta-modèle commun fonctionnel**

Ce méta-modèle permet de définir les différentes fonctions associées aux activités d'un modèle de processus. Ces fonctions correspondent généralement à des opérations exécutées par un processus. Dans notre approche, une fonction est décrite par une action (ou verbe représentant l'action) et un objet contribuant à la réalisation de l'action associée à la fonction. Le méta-modèle fonctionnel peut être représenté de la manière suivante (figure 4.28).

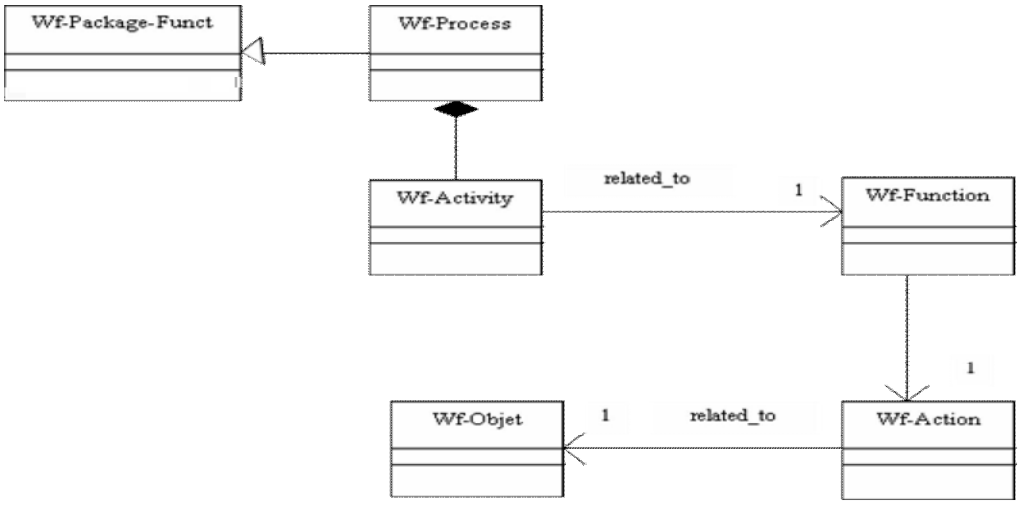

Figure 4.28 : Le méta-modèle commun fonctionnel d'un modèle de processus.

- **Le méta-modèle commun des ressources**

Ce méta-modèle décrit cinq catégories de ressources : les ressources humaines, logicielles, matérielles, temporelles et des ressources externes (applications externes ou courrier électronique) que peut demander un processus ou une activité pour son exécution (figure 4.29)

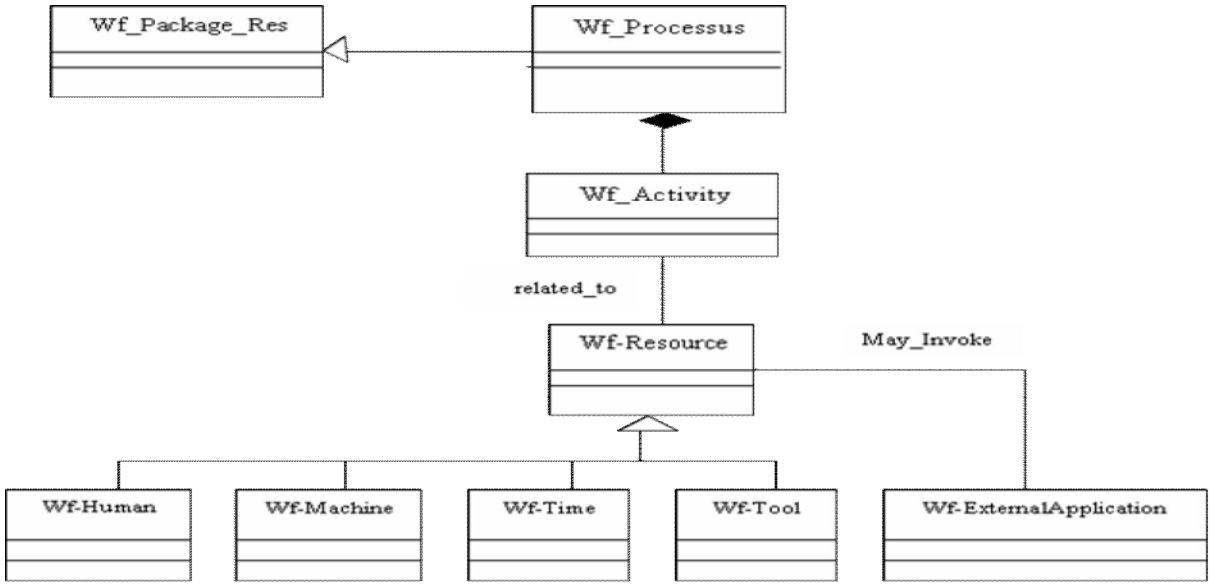

Figure 4.29 : Le méta-modèle commun des ressources.

Comme il a été évoqué plus haut, ces méta-modèles communs sont considérés, puis implémentés comme des ontologies communes et la technique d'annotation sémantique proposée pour chaque aspect des modèles est basée ainsi sur son ontologie correspondante, qui fait l'objet de la section suivante.

3.7 Annotation sémantique des aspects des modèles de processus

Les annotations sémantiques proposées sont les suivantes :

3.7.1 Annotation sémantique de type informationnel

Les données sont souvent reliées à des artefacts manipulés par les activités. L'annotation sémantique de type informationnel est décrite par le concept "Wf-Artifact" et les concepts "'Inputs " et "Outputs" de l'ontologie informationnelle. Ils sont décrits comme suit :

Wf-Artifact = (id, name, equivalent_name, same_as, kind_of, part_of, related_Wf-Input, related_Wf-Output).

Wf-Inputs *= (id, name, equivalent_name, data-type, related-artifact).*
Wf-Outputs *= (id, name, equivalent_name, data-type, related-artifact).*

L'annotation sémantique informationnelle d'une activité quelconque (AVi) se fait en faisant référence à l'ontologie informationnelle via la relation ***"related_Wf-Artifact***" et l'activité sera décrite comme suit :

AVi = (id, name, same_as, equivalent_name, kind_of, phase_of, related_Wf-Artifact).

3.7.2 Annotation sémantique de type comportemental

L'annotation sémantique de type comportemental des modèles de processus (ou fragments de modèles) est basée sur le contrôle de flux (figure 4.26) et sur les différents états/transitions (figure 4.27). L'annotation sémantique fait donc référence à deux ontologies dites "dynamiques" implémentées à partir des méta-modèles dynamiques de contrôle de flux et d'états/transitions.

Le contrôle de flux (CFi) défini dans un modèle de processus représente le type d'ordre d'exécution des activités qui est représenté en général par six types d'opérateurs : la jonction en entrée d'activités (AND Join), la jonction en sortie des activités (AND Split), la disjonction exclusive en entrée des activités (XOR Join), la disjonction exclusive en sortie des activités (XOR Split), la disjonction non exclusive en entrée des activités (OR Join) et la disjonction non exclusive en sortie des activités (OR Split). Nous pouvons décrire cet aspect de contrôle de flux comme suit :

CFi = (id, name, equivalent_name, related_Activity).

Nous distinguons généralement trois types de contrôle de flux exprimés sous forme de séquences, de mises en parallèle, et de branchement. Pour une activité quelconque (AVi), nous pouvons décrire ces trois types de contrôle de flux comme suit :

$Sequence_i$ = (id, name, equivalent_name, has_inActivity, has_outActivity).

$Choice_i$ = (id, name, equivalent_name, has_inActivity, has_outActivity, has_logicConnecteur).

Choice$_i$ correspond à la jonction ou disjonction en entrée des activités (*Join$_i$*) et l'élément '*has_logicConnecteur'* de *Choice$_i$* possède la valeur 'XOR' pour *XOR Join$_i$* , la valeur 'OR' pour *OR Join$_i$* et la valeur 'AND' pour *AND Join$_i$* .

parallel$_i$ = (id, name, equivalent_name, has_inActivity, has_outActivity, has_logicConnector).

parallel$_i$ correspond à la jonction ou la disjonction en sortie des activités (*Split$_i$)* et l'élément '*has_logicConnecteur'* de *parallel$_i$* possède la valeur 'XOR' pour *XOR Split$_i$* , la valeur 'OR' pour *OR Split$_i$* et la valeur 'AND' pour *AND Split$_i$*.

Pour une activité quelconque (AVi), les inputs (***Inp$_i$***) et les outputs (***Oup$_i$***) sont définis comme des paramètres d'activité comportant des types de données. Ils sont souvent associés à des artifacts manipulés par l'activité. Nous pouvons les décrire comme suit :

Inp$_i$ = (id, name, equivalent_name, data-type, related_artifact).

Oup$_i$ = (id, name, equivalent_name, data-type, related_artifact).

De même, les pré-conditions (***Pre$_i$***) et les post-conditions (***Post$_t$***) du méta-modèle commun dynamique décrivant le contrôle de flux peuvent être décrits comme suit :

Pre$_i$ = (id, name equivalent_name, related_input).

Post$_t$ = (id, name, equivalent_name, related_output).

Aussi, l'aspect dynamique des modèles de processus est exprimé à travers les différents états/transitions que nous avons présentés par le méta-modèle d'états/transitions, et qu'il convient de l'annoter. L'annotation comportementale d'une activité quelconque (AVi) est décrite en faisant référence à son état de transition via la relation " *related_Wf-StateMachine*".

AVi = (id, name, equivalent_name, same_as, kind_of, phase_of, related_Wf-StateMachine)

3.7.3 Annotation sémantique de type fonctionnel

Elle consiste à annoter le concept de fonction (ou d'opération) d'une activité, en faisant référence à l'ontologie fonctionnelle via la relation ***"related_Wf-function"***. Pour une activité quelconque (AVi), l'annotation de type fonctionnel est décrite comme suit :

AVi = (id, name, equivalent_name, same_as, kind_of, phase_of, related_Wf-function).

Wf-function= (id, name, equivalent_name, same_as).

3.7.4 Annotation sémantique de type organisationnel

Elle est décrite pour annoter les concepts organisationnels (unité organisationnelle, rôle, acteur, etc.). Pour une activité quelconque (AVi), l'annotation de type organisationnel est décrite via la relation *"related_Wf-OrganizationalUnit"*.

AVi = (id, name equivalent_name, same_as, kind-of, phase_of, related_Wf-OrganizationaUnit).

Wf-OrganizationalUnit= (id, name, equivalent_name, same_as, is_composed_of).

La relation sémantique ' *is_composed_of* ' fait référence aux concepts organisationnels définis dans l'ontologie organisationnelle tels que rôle, acteur, équipe. L'annotation d'un concept organisationnel tel que rôle, par exemple, peut être décrite de la manière suivante :

Wf_Role = (id, name equivalent_name, same_as, member_of, instance_of).

3.7.5 Annotation sémantique des ressources

Elle consiste à annoter les concepts de ressources (humaines, logicielles, matérielles, temporelles) qui sont invoquées pour l'exécution de l'activité. Pour une activité quelconque (AVi), ce type d'annotation est décrit via la relation *" related_Wf-Resource"*.

AVi = (id, name, equivalent_name, same_as, kind_of, phase_of, related_Wf-Resource).

Wf-Resource = (id, name, equivalent_name, same_as).

Ainsi, notre méta-modèle commun d'annotation sémantique de processus (MCASMP) est étendu et enrichi par l'ajout d'un ensemble des concepts liés à plusieurs ontologies communes de type informationnel, comportemental, organisationnel, fonctionnel et de ressources.

Finalement, notre MCASMP est formalisé comme suit :

MCASMP = (AV, AR, AC, AF, AE, AS, AT, RS, DO, PO, GO, IO, FO, CO, OO, RO).

Où :

DO dénote un sous-ensemble des concepts communs de l'ontologie du domaine,
PO dénote un sous-ensemble des concepts communs de l'ontologie des profils,
GO dénote un sous-ensemble des concepts communs de l'ontologie des buts,
IO dénote un sous-ensemble des concepts communs de l'ontologie informationnelle,
FO dénote un sous-ensemble des concepts communs de l'ontologie fonctionnelle,
CO dénote un sous-ensemble des concepts communs de l'ontologie comportementale,
OO dénote un sous-ensemble des concepts communs de l'ontologie organisationnelle,
RO dénote un sous-ensemble des concepts communs de l'ontologie de ressources.

Pour résumer, nous dirons que pour annoter une activité quelconque (AVi) d'un modèle de processus (MPi), nous utilisons des relations sémantiques pour lier l'activité aux concepts qui sont définis dans toutes les ontologies communes de référence et qui sont partagées par tous les acteurs de l'interopérabilité telles que ontologie de domaine, thésaurus (comme une forme d'ontologie), ontologie des profils, ontologie des buts, ontologie informationnelle, organisationnelle, comportementale, fonctionnelle et de ressources:

Et une activité quelconque AVi d'un modèle de processus MPi sera donc, décrite de la manière suivante :

AVi = (id, name, same_as, equivalent_name, related_Wf-function, related_Wf-OrganizationaUnit, related_Wf-Artifact, is_in_Control_Flow, related_Wf-StateMachine, related_Wf-Resource, kind_of, phase_of, has_Profile, achieves_global_but).

Une activité quelconque AVi est considérée comme un élément dans MCASMP. Elle a un identificateur (*id)* et un nom (*name)* pour identifier de manière unique l'élément. '*Same-as'* est utilisé pour annoter l'élément 'Wf-Activity' avec son concept défini au niveau de l'ontologie du domaine (niveau concept) et *'equivalent_name*' fournit l'équivalent (ou le synonyme) du nom défini dans thésaurus (niveau terminologie).

Quant aux relations (*'related_Wf-function'*, *'related_Wf-Artifact, 'related_Wf-StateMachine', 'related_Wf-Resource*', *'related_Wf-OrganizationaUnit', 'is_in_Control_Flow'*), elles sont utilisées pour annoter les relations sémantiques entre l'élément annoté *'Wf-Activity'* avec les concepts qui lui sont liés et définis respectivement dans les ontologies suivantes : ontologie fonctionnelle, ontologie informationnelle, ontologie comportementale (états/transitions), ontologie des ressources, ontologie organisationnelle, et enfin ontologie comportementale (contrôle de flux). Quant aux relations sémantiques ('*kind_of'* et *phase_of'*), elles nous permettent d'annoter l'activité en utilisant l'ontologie du domaine de référence qui est dans notre travail l'ontologie SCOR.

Enfin, les relations sémantiques (*'has_Profile' et 'achieves_global_but'*), nous permettent d'annoter l'activité en utilisant respectivement l'ontologie des profils et des buts des modèles, qui est pour nous l'ontologie SCOR. Tous les concepts ontologiques sont dénotés par des URI (Uniform Resource Identifier) dans le méta-modèle d'annotation sémantique de processus (MCASMP).

Pour mieux illustrer notre approche, nous revenons à notre exemple d'une activité de commande d'achat (figure 4.1) qui est représentée par les inputs, les outputs, les pré-conditions et les post-conditions. Nous supposons que cette activité est définie dans deux modèles de processus différents (par exemple, XPDL et ebXML) et manipulée par deux acteurs d'interopérabilité *Act1* et *Act2*. Pour l'annoter sémantiquement, l'annotateur se réfère aux différentes ontologies communes via des relations sémantiques correspondantes à chaque ontologie.

Cette activité (AV) est décrite de la manière suivante :

AV = (id, name, same_as, equivalent_name, has_Wf-Role, has_Wf_Actor, has_Wf-Artifact, has_Wf_Inputs, has_Wf_Outputs, has_Wf_Preconditions, has_Wf_Postconditions, is_in_Control_Flow, has_Wf-State, has_Wf-Resource, kind_of, has_Profile, achieves_global_but).

Les relations sémantiques (*'has_Wf-Role' et 'has_Wf_Actor'*) font référence à l'ontologie organisationnelle. Les relations sémantiques *('has_Wf_Inputs', 'has_Wf_Outputs', 'has_Wf-Artifact')* réfèrent à l'ontologie informationnelle, Les relations sémantiques *('has_Wf_Preconditiuons', 'has_Wf_Postconditions', 'is_in_Control_Flow', 'has_Wf-State*') réfèrent aux ontologies dynamiques (contrôle de flux et états/transitions). La relation sémantique *'has_Wf-Resource'* fait référence à l'ontologie des ressources.

D'un point de vue technique, ces annotations sont codifiées dans un langage XML, en utilisant un namespace 'OCPW' (abréviation d'Ontologie Commune de Processus Workflow), et sont envoyées par l'acteur *Act1* à l'acteur *Act2* pour interprétation de l'activité (figure 4.30).

```
<OCPW: Wf-Activity rdf:ID="AV1">
<OCPW: model_fragment rdf: resource="&ebxml_purchase#id2"/>
<OCPW: model_fragment rdf: resource="&xpdl_purchase#id3"/>
<OCPW: name>Purchase</OCPW: name>
<OCPW: equivalent_name>Buying</OCPW: equivalent_name>
<OCPW: kind_of rdf: resource="&SCOR_DomainOntology#buy">
<OCPW: achieves_global_goal = "&SCOR_GoalOntology#product on order">
<OCPW: has_profile ="&Profile_Ontology#purchaseordershipping">
</OCPW: Wf-Activity>
<OCPW: has_Wf-Role>
      <OCPW: Wf-Role rdf:resource="&Organizational_Ontology#R1"/>
 </OCPW: has_Wf-Role>
<OCPW: has_Wf-Actor>
       <OCPW: Wf-Actor rdf: resource="&Organizational_Ontology #C1"/>
</OCPW: has_Wf-Actor>
<OCPW: has_Wf-Artifact>
     <OCPW: Wf-Artifact rdf: resource="&Informational_Ontology#Ar1"/>
</OCPW: has_Wf-Artifact>
<OCPW: has_Wf-Inputs>
       <OCPW: Inputs rdf: resource="&Informationnal_Ontology#Inp1"/>
</OCPW: has_Wf-Inputs>
<OCPW: has_Wf-Outputs>
       <OCPW: Outputs rdf: resource="& Informational _Ontology#Oup1"/>
</OCPW:has_Wf-Outputs>
<OCPW: has_Wf-Preconditions>
        <OCPW:  Preconditions rdf: resource="&BehavioralFlow_Ontology#Pre1"/>
</OCPW: has_Wf-Preconditions>
<OCPW: has_Wf-Postconditions>
        <OCPW:  Postconditions rdf: resource="&BehavioralFlow_Ontology#Post1"/>
</OCPW: has_Wf-Postconditions>
<OCPW: is_in_Control_Flow>
        <OCPW: Control_Flow rdf: resource="&BehavioralFlow_Ontology#Cf1"/>
</OCPW: is_in_Control_Flow>
<OCPW: has_Wf-Resource>
        <OCPW: Wf-Resource rdf: resource="&Resources_Ontology#Rs1"/>
</OCPW: has_Wf-Resource>
<OCPW: has_Wf-State>
        <OCPW: Wf-State rdf: resource="&BehavioralStatMachine_Ontology#S1"/>
</OCPW: has_Wf-State>
<OCPW: Wf-Role rdf:ID="R1">
<OCPW: model_fragment rdf: resource="&ebxml_purchase#id4"/>
<OCPW: model_fragment rdf: resource="&xpdl_purchase#id8"/>
<OCPW: name>client</OCPW: name>
<OCPW: equivalent_name >customer></OCPW: equivalent_name>
</OCPW: Wf-Role>
<OCPW: Wf-Artifact>
<OCPW: model_fragment rdf: resource="&ebxml_purchase#id10"/>
<OCPW: model_fragment rdf: resource="&xpdl_purchase#id25"/>
<OCPW: name>Order form</OCPW: name>
<OCPW: equivalent_name>Order sheet</ equivalent_name >
</OCPW: Wf-Artifact>
.....
```

Figure 4.30 : Codification d'une annotation sémantique d'une activité de commande d'achat basée sur différentes ontologies.

3.8 Synthèse

La problématique de l'interopérabilité des modèles de processus dans un environnement coopératif hétérogène est très complexe. Pour gérer l'hétérogénéité sémantique des modèles de processus et la traiter au niveau conceptuel, nous avons proposé une approche basée sur des annotations sémantiques en faisant référence à un ensemble d'ontologies qui sont communes et partagées par les acteurs de l'interopérabilité.

Notre approche s'article principalement autour de l'élaboration d'un méta-modèle commun d'annotation sémantique de modèles (MCASMP). Ce méta-modèle est étendu et enrichi pour permettre d'annoter les modèles de processus à différents niveaux :

- ***Au niveau des méta-modèles* :** pour résoudre l'hétérogénéité sémantique des méta-modèles au niveau des concepts;
- ***Au niveau des modèles*** *:* pour résoudre l'hétérogénéité sémantique des modèles de processus au niveau des contenus ;
- ***Au niveau des buts et des profils des modèles*** *:* pour résoudre l'hétérogénéité sémantique des objectifs visés par les processus et des profils sont publiés par les acteurs coopératifs pour permettre ainsi, leur découverte sémantique qui peut être basée sur des buts ou des profils.
- ***Au niveau des aspects de base des modèles*** *:* pour résoudre l'hétérogénéité sémantique des modèles de processus au niveau de leurs aspects de base qui leur sont liés et qui sont de type informationnel, comportemental, fonctionnel, organisationnel et de ressources.

4. Conclusion

L'analyse de l'état de l'art sur les différentes techniques utilisées pour résoudre la problématique de l'interopérabilité, nous a révélé que ces dernières s'appuient principalement sur le coté technique, en utilisant des outils tels que ceux de l'E.A.I (Entreprise Integration Application), de la technologie ESB (Enterprise Service Bus), de la technologie des Web services ou directement en se basant des langages de représentation d'ontologies ou d'annotation sémantique basés Web pour la description sémantique des Web services.

En effet, les solutions apportées permettent effectivement de résoudre l'interopérabilité, mais de manière concrète et dans un contexte industriel en utilisant directement la technologie Web avec les standards qui lui sont associés. Bien que ces solutions semblent efficaces pour les industriels, néanmoins, elle engendre une autre problématique pour les chercheurs : il s'agit de la problématique conceptuelle de l'interoprabilité, indépendemment de tout contexte industruel.

De même que l'analyse des approches existantes nous a montré que les besoins exprimés par un partenariat désirant être souple dans un environnement compétitif tels que le respect du savoir-faire,la flexibilité, l'ouverture et l'autonomie, ne sont pas totalement satisfaits.

Par du constant des limites de ces techniques industrielles et de ces approches qui ne sont pas ou sont peu flexibles, nous avons proposé dans ce chapitre deux approches conceptuelles pour résoudre la problématique de l'interopérabilité des modèles de Workflow :

- La première approche aborde l'aspect conceptuel dans un cadre général placé à un niveau d'abstraction élevé, et cela en utilisant les techniques de méta-modélisation et de transformation de modèles pour permettre de *i)* modéliser les ontologies afin de mieux appréhender l'aspect sémantique de manière conceptuelle *ii)* modéliser la plate-forme technique cible qui sera utilisée pour l'interopérabilité. L'approche préconisée combine l'approche MDA, les ontologies et les Web services qui, prises conjointement permettent de répondre aux besoins d'une coopération flexible et ouverte. L'avantage d'une telle approche par rapport aux approches citées dans l'état de l'art, est qu'elle traite d'abord, le problème de l'interopérabilité au niveau conceptuel grâce aux concepts de MDA, puis de se concentrer par la suite, sur les méta-modèles de la plate-forme technique et du langage cible qui présentent plus d'avantages en termes de flexibilité et d'ouverture pour la mise en oeuvre de l'interopérabilité.

- La deuxième approche aborde, par contre, l'aspect conceptuel dans un cadre plus général que la précédente en se basant sur un haut niveau d'abstraction, et cela en utilisant les techniques d'annotation sémantique qui nous permettent d'annoter sémantiquement les modèles de processus à différents niveaux : *1)* au niveau des méta-modèles de processus *2)* au niveau des modèles *3)* au niveau des buts et des profils des modèles *4)* au niveau des aspects de base des modèles, et cela en faisant référence à plusieurs ontologies communes. L'avantage majeur de la deuxième approche par rapport aux approches existantes dans l'état de l'art, est qu'elle est purement conceptuelle puisqu'elle elle ignore complètement le contexte industriel, ainsi que l'existence de toute technologie Web avec tous les standards qui lui sont associés. Elle est dite "agnostique " dans le sens où le mécanisme d'annotation sémantique fait abstraction de tout langage de représentation d'ontologies et ignore tout contexte industriel.

  De plus, elle permet de répondre aux besoins exprimés par le partenariat en termes de flexibilité, d'ouverture, d'autonomie et de savoir-faire. La flexibilité est dûe au mécanisme d'annotation qui est totalement séparé de la description sémantique des modèles. Ce qui procure une certaine souplesse pour les annotateurs d'annoter leurs modèles indépendamment de tout langage de représentation d'ontologies tels que OWL, OWL-S ou WSDL-S. Quant à l'ouverture, cette approche peut s'inscrire dans le cadre d'utilisation des standards industriels tels que WSDL et OWL.

  Pour les besoins d'autonomie et de savoir-faire, cette approche les satisfait totalement dans la mesure où les partenaires coopératifs préservent leurs Workflows et leurs systèmes de gestion de Workflow pré-établis, ainsi que leur savoir faire dans la mesure où ils ne doivent se conformer à aucune interface standard pour adapter ou ajuster leurs modèles de Workflow (ou leurs systèmes de gestion de Workflow).

Conclusion

Conclusion

Le spectre de l'interopérabilité est très large et la problématique de l'interopérabilité est très complexe. Elle est encore d'autant plus complexe dans le domaine du Workflow (ou business process) qui chevauche plusieurs champs d'application tels que le génie logiciel, la modélisation d'entreprise, les systèmes Workflows et la composition des Web services.

Cette complexité est dûe à plusieurs facteurs. Parmi ces facteurs, nous citons principalement le degré d'ouverture des processus métiers qui sont gérés par les partenaires coopératifs selon le contrat de collaboration. Ce degré peut être de type public, privé ou semi privé et sur lequel se basent les partenaires pour établir leurs politiques de coopération.

Un autre type de complexité concerne l'intégration de plusieurs aspects qui sont par nature liés à tout processus Workflow. Parmi ces aspects, nous citons ceux qui sont considérés comme des aspects de base qui sont de type informationnel, fonctionnel, comportemental, organisationnel et enfin de ressources.

Dans une perspective de coopération et d'échanges de modèles de processus dans un environnement hétérogène, le problème de l'interopérabilité sémantique constitue un problème crucial pour les entreprises dont les modèles sont par nature autonomes, distribués et hétérogènes.

1. Rappel de la problématique et des objectifs

Notre travail de thèse s'inscrit dans cette problématique et a pour objectif principal de résoudre les problèmes d'hétérogénéité sémantique des modèles de Workflow (ou modèles de processus). Il se concentre principalement sur la recherche d'une solution conceptuelle permettant de résoudre cette problématique indépendamment de la technologie du Web, des standards industriels et de toute plate-forme technique.

Outre cet aspect conceptuel de la problématique posée dans cette thèse, notre travail doit tenir compte des objectifs fixés qui sont exprimés en termes de besoins d'une coopération permettant de former des partenariats flexibles entre des participants hétérogènes. Ces objectifs doivent *1)* permettre aux partenaires de coopérer de manière *fle*xible *2)* de préserver *leur savoir-faire 3) leur autonomie* (autonomie de leurs Workflows ou de leurs systèmes de gestion de Workflow) *4)* permettre de *s'ouvrir* d'avantage et élargir la coopération à grande échelle afin de bénéficier des standards industriels.

2. Principales contributions

Notre méthodologie de travail est guidée principalement par deux préoccupations majeures à savoir *i)* recherche d'une solution conceptuelle pour résoudre l'interopérabilité sémantique des modèles de Workflow *ii)* répondre aux objectifs fixés par le partenariat (flexibilité, respect du savoir-faire, autonomie et ouverture). Pour ce faire, nous avons exploré deux espaces

d'investigation liés au domaine de l'interopérabilité et à celui du Workflow (ou business process) afin d'étudier et d'analyser leurs états de l'art pour apporter les ingrédients nécessaires permettant de ***i)*** *prendre en charge l'aspect sémantique de manière conceptuelle pour résoudre l'interopérabilité* ***ii)*** *répondre aux besoins d'une coopération flexible et ouverte entre des partenaires hétérogènes.*

Cependant, l'analyse des différentes approches proposées dans l'état de l'art pour résoudre cette interopérabilité, nous a permis de recenser principalement deux catégories d'approches :

- Les approches standards : Elles sont basées sur des interfaces standards et demandent des adaptations et des modifications des parties voire même la totalité des Workflows et/ou des systèmes de gestion de Workflow existants des partenaires. En effet, elles obligent ces derniers à se conformer à des spécifications et/ou un ensemble d'interfaces et de fonctionnalités bien définies. Par conséquent, elles ne préservent ni les Workflows, ni les systèmes de gestion de Workflow pré-établis. Ce qui réduit l'autonomie des partenaires, et par suite la flexibilité de la coopération.

- Les approches spécifiques : Elles sont adaptées à des projets dans un contexte d'entreprises virtuelles où les partenaires doivent se conformer à des interfaces statiques pré-définies dès la conception de l'entreprise virtuelle et l'intégration d'un nouveau Workflow pour la coopération exige la conformité de son interface vis-à-vis du partenariat. Ce qui ne préserve pas l'autonomie des systèmes de gestion de Workflow, et par suite la flexibilité de la coopération.

De même que l'étude des différentes techniques utilisées pour la mise en œuvre de l'interopérabilité, nous a permis de retenir particulièrement deux techniques permettant de résoudre concrètement l'interopérabilité : *1)* les techniques syntaxiques qui négligent l'aspect sémantique *2)* les techniques sémantiques qui se basent sur le côté technique pour décrire la sémantique des modèles en s'appuyant sur des langages de représentation d'ontologies basés Web tels que OWL, OWL-S ou d'annotation sémantique tels que WSDL-S ou SAWSDL.

En résumé, nous pouvons dire que bien que ces approches et techniques aient été proposées permettent effectivement de résoudre la problématique de l'interopérabilité des modèles Workflows; il en demeure toujours que l'aspect sémantique n'est pas encore traité de manière conceptuelle et que certaines approches ne sont pas flexibles et par contre, d'autres sont peu flexibles. De même que la plupart d'entre-elles ne permettent pas de répondre à l'ensemble des besoins de coopération exprimés par un partenariat flexible.

Partant de ce constat, nous nous sommes partis à la recherche d'une méthodologie et d'une solution flexible permettant de pallier aux limites de ces approches afin de prendre en charge l'aspect sémantique de manière conceptuelle et répondre aux objectifs fixés.

Pour capturer l'aspect conceptuel de la solution proposée, la démarche adoptée pour notre travail est guidée principalement par deux courants de recherche différents qui nous ont orientés vers deux pistes distinctes :

- La première piste que nous avons explorée, s'inscrit dans un cadre conceptuel et utilise les techniques de méta-modélisation et de transformation de modèles. Elle nous offre un niveau d'abstraction élevé pour *i)* modéliser les ontologies afin de décrire l'aspect

sémantique des modèles de processus *ii)* modéliser la plate-forme technique cible qui sera utilisée pour la coopération entre des acteurs collaboratifs. L'approche proposée est basée sur MDA, sur le concept d'ontologies et sur le paradigme des Web services. C'est une approche dite "approche de standardisation" car elle se base sur un méta-modèle commun standard conçu selon les concepts de MDA. Elle est orientée industrielle car elle permet d'utiliser la technologie Web et tous les standards qui lui sont associés.

- La deuxième piste s'inscrit aussi, dans un cadre conceptuel et utilise cette fois ci, les techniques d'annotation sémantique pour annoter sémantiquement les méta-modèles de processus, leurs modèles correspondants et enfin les différents aspects de base qui sont intrinsèquement liés à ces modèles et qui sont de type informationnel, comportemental, fonctionnel, etc. Ces annotations sémantiques font référence à un ensemble commun d'ontologies. Cette approche est dite "approche d'unification " car elle se base sur un méta-modèle commun unifié et accepté par le partenariat. Elle est dite aussi "agnostique" car elle ignore tout contexte industriel, ainsi que toute la technologie du Web.

Partant de ces deux pistes de recherche, nous avons proposé deux approches conceptuelles différentes pouvant être appliquées dans le domaine de l'interopérabilité des modèles de processus.

Notre première contribution porte sur la proposition d'une approche conceptuelle pour l'interopérabilité des modèles de Workflow en utilisant les techniques de méta-modélisation et de transformation de modèles.

Cette approche constitue une concrétisation de l'approche MDA dans un contexte industriel. Elle préconise de combiner à la fois l'approche MDA, les ontologies et les Web services et permet :

- De construire une architecture ontologique basée MDA pour la prise en compte de l'aspect sémantique au niveau conceptuel. Cette construction est basée sur l'intégration de trois espaces technologiques : MDA, le Web sémantique et les Web services sémantiques, qui conjointement semblent être une voie prometteuse pour favoriser l'interopérabilité sémantique.

- De modéliser les langages de représentation d'ontologies basés Web tels que OWL, OWL-S, WSDL-S en élaborant leurs méta-modèles correspondants.

- De modéliser la plate-forme technique cible qui sera utilisée pour l'interopérabilité des modèles de Workflow.

- De faciliter surtout l'adoption des Web services sémantiques dans le contexte MDA pour la génération des descriptions OWL-S. Cette facilité permet aux utilisateurs qui sont déjà, familiers avec les outils MDA (ou les outils UML) d'intégrer facilement leurs outils pour décrire la sémantique ou d'annoter leurs modèles de processus, et cela en déployant moins d'efforts d'apprentissage.

Cependant, pour faciliter d'adoption de ces langages de représentation d'ontologies basés Web, nous avons proposé un processus basé MDA pour la génération semi-automatique d'une

ontologie de services. Ensuite, nous l'avons validé par une implémentation en choisissant OWL-S comme langage cible, vu les avantages qu'il procure pour l'interopérabilité telles que découverte dynamique, composition.

L'avantage majeur d'une telle approche par rapport aux approches existantes, est qu'elle permet de résoudre la problématique conceptuelle de l'interopérabilité des modèles de Workflow et de répondre aux objectifs du partenariat. Les trois approches proposées (MDA, ontologies et Web services) prises conjointement permettent d'adresser effectivement le problème sémantique de manière conceptuelle et de prendre en compte les critères de flexibilité et d'ouverture qui sont liés principalement au principe de MDA, du modèle SOA et du méta-modèle OWL-S.

En effet, MDA permet de capturer l'aspect sémantique de manière conceptuelle en se basant sur un haut niveau d'abstraction, en utilisant le méta-méta-modèle MOF (Meta-Object Facility) de l'architecture MDA. Elle est considérée comme flexible et ouverte puisqu'elle est basée sur des standards industriels. Le choix des Web services est principalement motivé par le fait qu'ils constituent une technologie flexible basée sur des standards industriels, tandis que le choix des ontologies est justifié par le fait qu'elles représentent le moyen efficace pour la prise en compte de l'aspect sémantique des Web services. Quant au choix de OWL-S comme méta-modèle cible, il est justifié par le fait qu'il présente beaucoup d'avantages telles que modularité, flexibilité, maximum d'expressivité et généricité.

Parmi ces limites majeures que nous avons relevées, est qu'elle repose sur une vision orientée services et il est à ce titre indispensable que les partenaires de la collaboration soient en mesure de présenter leurs modèles de Workflow (ou business process) selon cette même vision vis-à-vis de l'extérieur. Cette exigence préalable d'une mise en conformité avec les principes SOA, constitue évidemment une limite tangible à notre première approche (et ce malgré les avantages incontestables de ce type d'approche à l'heure actuelle).

Notre deuxième contribution porte la proposition d'une autre approche conceptuelle pour l'interopérabilité des modèles de Workflow (ou modèles de processus), en utilisant les techniques d'annotation sémantique.

Elle préconise donc, l'utilisation des annotations sémantiques en se référant à un ensemble commun d'ontologies afin de résoudre l'hétérogénéité sémantique des modèles de processus, rencontrée à différents niveaux.

La deuxième approche que nous avons proposée, permet de résoudre l'interopérabilité dans un contexte homogène ou hétérogène. C'est une approche incrémentale, basée sur l'élaboration d'un méta-modèle commun d'annotation sémantique de modèles (MCASMP) qui représente un schéma commun d'annotation établi en collaboration par les acteurs coopératifs. Ce méta-modèle est étendu, puis enrichi afin de résoudre l'hétérogénéité sémantique des modèles de processus à différents niveaux : *1) au niveau des méta-modèles 2) Au niveau des modèles* 3) *Au niveau des buts et des profils des modèles 4) Au niveau des aspects de base des modèles.*

Cette contribution porte sur la proposition de deux approches conceptuelles d'interopérabilité pouvant être utilisées dans deux contextes différents : homogène ou hétérogène.

Dans un contexte homogène, l'ontologie du domaine de référence est implicite et connue par les acteurs coopératifs, et par conséquent, deux types d'annotations ont été proposés :

- **Annotations structurelles** : elles sont décrites en utilisant uniquement la notation UML, en faisant référence à des concepts décrits *i)* dans des ontologies identiques dans le cas il y a vraiment un consensus entre tous les partenaires pour la construction de ces ontologies *ii)* dans les ontologies des partenaires dans le cas où elles s'avèrent ne sont pas totalement identiques. Ces annotations réfèrent à un ensemble commun d'ontologies qui sont liées aux buts, aux profils et aux aspects de base qui sont rattachés aux modèles et qui sont de type fonctionnel, informationnel, comportemental, organisationnel et de ressources.

- **Annotations lexicales** : elles sont utilisées pour mieux expliciter les termes utilisés par les acteurs coopératifs en faisant référence à une base de données lexicale telle que WordNet**,** ou plutôt à un glossaire comme celui de la WfMC (Workflow Mangement Coalition) qui contient tout le vocabulaire de base utilisé dans le domaine du Workflow.

Par contre, dans un contexte hétérogène, nous nous référons à plusieurs ontologies qui sont communes et partagées par tous les acteurs coopératifs. Les annotations proposées concernent

- **L'annotation sémantique des méta-modèles de processus** : Le but de ces annotations est de résoudre l'hétérogénéité sémantique des méta-modèles au niveau des concepts. Ce type d'annotation consiste alors à utiliser les concepts de l'ontologie commune de processus Workflow (OCPW) comme des méta-données pour annoter la sémantique des concepts des différents méta-modèles de processus. OCPW est construite à partir d'un méta-modèle commun de processus élaboré et issu de l'alignement des concepts des différents méta-modèles de processus tels que XPDL, BPML

- **L'annotation sémantique des modèles de processus :** Le but de ces annotations est de résoudre l'hétérogénéité sémantique des modèles de processus au niveau des contenus. Dans notre travail, nous utilisons l'ontologie SCOR (Supply Chain Operations Reference model) comme une ontologie du domaine qui est considérée un modèle de référence dans la gestion des approvisinnements. Elle nous sert à annoter les contenus des modèles.

- **L'annotation sémantique des buts des modèles :** Son rôle est d'enrichir la sémantique des objectifs visés par les processus pour la découverte sémantique de ces modèles (qui sont basés sur des objectifs métiers). Dans notre travail, nous utilisons aussi l'ontologie SCOR, considérée comme une ontologie des buts pour annoter les buts ou les objectifs que devraient atteindre les processus.

- **L'annotation sémantique des profils des modèles** : Son rôle est d'enrichir la sémantique des profils publiés par les acteurs coopératifs pour une découverte sémantique (qui sont basés sur des profils). L'ontologie des profils nous sert à annoter les profils des modèles exprimés en termes de fonctionnalités des processus.

- **L'annotation des termes qui présentent des ambiguités d'interprétation :** le but est de mieux expliciter les termes qui sont employés par les acteurs coopératifs pour avoir une bonne compréhension de ces modèles. Dans notre travail, nous utilisons thésaurus

comme une forme d'ontologie pour annoter les termes ambigus par leurs synonymes ou par leurs équivalents.

- **L'annotation sémantique des aspects de base des modèles de processu**s : le but de ces annotations est de résoudre l'hétérogénéité sémantique des modèles au niveau des aspects de base des modèles et qui sont de type informationnel, comportemental, fonctionnel, organisationnel et de gestion des ressources. Ces aspects s'avèrent parfois, nécessaires pour avoir une bonne compréhension de ces modèles lors de leur exécution.

L'avantage de cette approche par rapport aux approches citées dans l'état de l'art, est qu'elle permet aussi de résoudre la problématique conceptuelle de l'interopérabilité des modèles de Workflow et de répondre aux objectifs du partenariat. En effet, elle permet de mieux appréhender l'aspect sémantique de manière conceptuelle en se basant sur le concept d'annotation sémantique. Elle est considérée comme flexible dans la mesure où le mécanisme d'annotation sémantique est totalement indépendant de tout langage de description ou d'annotation sémantique permettant à la communauté des développeurs de choisir leur langage ontologique souhaité ou de permettre à des acteurs coopératifs d'annoter leurs modèles (ou fragments de modèles) selon le langage de représentation d'ontologies préféré.

Cette flexibilité est dûe au fait que les ontologies sont externalisées, ce qui procure une certaine souplesse pour les annotateurs d'annoter leurs modèles indépendamment de toute technologie existante ou future. Elle est ouverte dans le sens où elle permet l'utilisation des standards industriels pour décrire les annotations sémantiques dans n'importe quel langage d'annotation sémantique basé Web tel que WSDL-S ou SAWSDL.

L'avantage majeur de cette approche par rapport aux approches existantes, est qu'elle est agnostique dans le sens où elle ignore complètement le contexte industriel, ainsi que l'existence de toute technologie Web avec tous les standards qui lui sont associés. Outre cet avantage, elle permet la traçabilité des modèles et le suivi de ces annotations sémantiques qui sont des caractéristiques importantes. En effet, ces caractéristiques sont intrinsèquement liées à la mise en oeuvre de l'interopérabilité et qui sont préservées via les différentes annotations et les transformations exprimées sur les modèles.

3. Bilan des approches proposées

Dans un esprit de critique, nous avons comparé ces deux approches conceptuelles en se basant sur certains critères que nous avons définis (table 5.1).

Critères d'analyse	Approches conceptuelles d'interopérabilité des modèles de processus	
	1ère Approche basée sur les techniques de méta-modélisation (MDA, Ontologies et Web services)	**2ème Approche basée sur les techniques d'annotation sémantique**
Abstraction (Niveau conceptuel)	Conceptuel basé sur les concepts standards de MDA (niveau méta-méta).	Conceptuel basé sur un cadre unifié, accepté par le partenariat.
Ouverture (Standardisation)	Orientée standard (OWL, OWL-S, WSDL-S, etc.) via leurs méta-modèles.	Non orientée standard.
Agnosticisme (Ignorance de tout contexte Web)	Non agnostique au contexte Web (fait référence à la technologie Web).	Agnostique au contexte Web (ignorance de la technologie Web).
Description sémantique des modèles	Basée sur les méta-modèles d'ontologies tels que OWL, OWL-S, WSML, WSDL-S.	Basée sur des annotations sémantiques.
Niveaux d'hétérogénéité sémantique traités	Niveau : - Méta-modèles (concepts) (Interopérabilité dite "par méta-modèles")	Niveaux : - Méta-modèles (concepts), - Modèles (instances) - Aspects (informationnel, etc.). (Interopérabilité dite " par annotation sémantique")
Flexibilité	Flexible : liée au principe de MDA, à la plate-forme cible (SOA), et au méta-modèle OWL-S.	Grande flexibilité: liée au mécanisme d'annotation.
Traçabilité	Possibilité de conserver les traces des transformations réalisées sur les modèles. . .	Possibilité de conserver les traces des annotations sémantiques.
Mode de coopération	Conforme au modèle SOA.	Non conforme au modèle SOA.
Découverte sémantique	Basée sur des algorithmes de découverte dynamique.	Basée sur des annotations sémantiques.
Outils de mise en œuvre	Plusieurs outils (de modélisation, de transformation, etc.).	Un seul outil d'annotation sémantique de modèles.
Plate-forme de coopération	Liée à l'architecture orientée services (SOA).	Indépendante de toute architecture.

Table 5.1 : Tableau comparatif des approches conceptuelles d'interopérabilité proposées.

4. Perspectives

Conscients des limites que présente ce travail et soucieux de sa continuité pour la mise en œuvre de l'interopérabilité des modèles de Workflow, nous envisageons quelques perspectives principales que nous avons résumées par les points suivants :

- La première de ces perspectives concerne la première approche qui débouche vers la découverte dynamique des modèles ou fragments de modèles tels que des services métiers et où il s'agit d'enrichir le processus de découverte et cela par l'introduction de la notion de buts dans les algorithmes de découverte sémantique déjà existants. De même qu'il s'agit d'envisager de prendre en compte d'autres aspects (aspect sécurité, aspect transaction, aspect qualité de service, coût, performance, etc.) qui sont rattachés aux modèles selon le contexte et les objectifs de la coopération. En effet, les acteurs coopératifs désirent parfois, avoir une bonne compréhension d'un ou de plusieurs aspects de ces modèles, par exemple, délai, coût ou performance des processus, qui sont considérés comme des critères importants pour un partenariat pour pouvoir interopérer

dans un contexte très concurrentiel. Ce qui nécessite de les annoter sémantiquement afin d'éviter des ambiguïtés d'interprétation.

- La deuxième perspective concerne le contexte d'interopérabilité des modèles de Workflow (ou business process) où le niveau métier est parfois négligé et qu'il est important de le considérer afin de s'assurer que l'information métier doit être échangée d'une manière compréhensible (niveau sémantique), accessible (niveau technique) et surtout organisée (niveau métier).

- La troisième perspective concerne la deuxième approche où il s'agit de mettre en place un système de gestion sémantique des modèles de Workflow (ou modèles de processus) qui devrait comporter des outils ontologiques permettant à un acteur (ou un utilisateur) d'interagir avec le système pour *1)* annoter sémantiquement ses modèles de processus, *2)* naviguer pour explorer leurs contenus *3)* soumettre des requêtes *4)* découvrir sémantiquement des modèles de processus.

Pour terminer, nous dirons que notre travail ne prétend pas présenter une solution parfaite à une problématique complexe qui est celle de l'interopérabilité des modèles de Workflow, qui reste cependant toujours ouverte. Aussi, nous pensons que la proposition d'une définition d'une ontologie unique ou d'ontologies communes dans un contexte homogène ou hétérogène, nécessite des consensus difficiles à atteindre, surtout dans un contexte de mondialisation actuel de l'économie caractérisé par une concurrence plus intensive et où chaque partenaire cherche à protéger ses secrets.

De même que le problème de la sémantique persiste toujours. En effet, nous pensons que pour le résoudre, il faut définir une couche sémantique à chaque élément d'information (atomique ou composite) lors de l'échange et de la coopération et introduire le concept "d'entreprise sémantique" dans l'univers du Web sémantique. Aussi, un autre problème lié à la sémantique tient du fait qu'elle peut différer d'un système à un autre, en fonction du contexte d'utilisation, qui est actuellement un axe de recherche actif pour la communauté travaillant sur les ontologies contextuelles.

Enfin pour conclure, nous dirons que compte tenu de l'intérêt porté par ce travail, nous demeurons optimistes et convaincus qu'il doit être soutenu et poursuivi.

Bibliographie

[1] M. Yogesth. " Business Process Redesign : An Overview, "IEEE Engineering Management Review, vol.26, no.3, 1998.

[2] T.H Davenport, " Process Innovation". Havard Business School Press, Boston, MA.1993.

[3] H.Panetto, Meta-Modèles et Modèles pour l'Intégration et l'Interopérabilité des Applications d'Entreprises de Production, HDR, Université Nancy 1, 2006

[4] Workflow Management Coalition, Workflow Process Definition Interface, XML Process Definition Language, Document Number TC-1025, May 22, 2001.

[5] S.Thatte, « XLANG, Web Services for business process design », Mai 2001

[6] F.Leyman, "Web Services Flow Language Web Services Flow Language ", IBM Software Group specification, Mai 2001.

http://www-306.ibm.com/software/solutions/webservices/pdf/WSFL.pdf.

[7] "Web Services Choreography Interface1.0", BEA systems, Sun Microsystems, June 2002. http://www.w3.org/TR/wsci/

[8] " Web Services Conversation Language 1.0", 2002, http://www.w3.org/TR/wscl10/

[9] BPEL4WS – Business Process Execution Language for Web Services version 1.1".

http://www-128.ibm.com/developerworks/library/specification /ws-bpel/, 2005.

[10] Jean-Marie Chauvet, : « Web Services avec SOAP,WSDL, UDDI, ebXML…», Editions Eyrolles 2002.

[11] "ebXML, Business Process Specification Schema Version 1.01", ebXML Business Process Project Team, mai 2001, http://www.ebXML.org/specs/ebBPSS.pdf.

[12] D. Vanderhaeghen, S. Zang, A. Hofer, O. Adam, "XML-based Transformation of Business Process Models – Enabler for Collaborative Business Process Management", Proc. of the 2nd GI Workshop XML4BPM , XML Interchange Formats for Business Process Management, 2004.

[13] N.Boudjlida," Intégration et Interopérabilité dans les Environnements logiciels " , Loria, Université Henri Poincaré Nancy I, Projet Architecture.

http://www.aee.inria.fr/cederom/bilan/docbilan/interop.pdf.

[14] A. Mahfoud," Vers une fédération de composants interopérables pour les environnements centrés procédé logiciels ", thèse de Doctorat de l'Université Joseph Fourier de Grenoble I (France), 17 juin 1999.

[15] X. Blanc : "Echanges de Spécifications Hétérogènes et Réparties ", Thèse de Doctorat de l'Université de Paris VI, France, 15 Novembre 2001.

[16] S. Izza, "Intégration des systèmes d'information industriels, une approche flexible basée sur les services sémantiques", thèse de doctorat, Ecole des Mines de Saint-Étienne, 2006.

[17] J. Touzi, "Aide à la conception de Système d'Information Collaboratif support de l'interopérabilité des entreprises", Thèse de Doctorat de l'Institut National Polytechnique de Toulouse (France), 09 Novembre 2007.

[18] S. Baina, "Interopérabilité dirigée par les modèles : approche Orientée Produit pour l'interopérabilité des systèmes d'entreprise", Thèse de l'Université Henri Poincaré Nancy I, (France), 7 Décembre 2006.

[19] Vlad Ingar, Wietrzyk, Vijay khandelwal (University of Western Sidney, Australia), And Makoto Takizawa (University of Japan): “An architecture for Workflows Interoperability Supporting Electronic Commerce ”, 2001.

[20] I. Chebbi, S. Dustdar, and S. Tata.," The View-based Approach to dynamic Inter-organizational Workflow Cooperation." Data and Knowledge Engineering Journal, 56(2) :139–173, 2006.

[21] OWL-S: OWL Services Coalition, OWL-S: Semantic Markup for Web Services. (2004), Available, http://www.daml.org/services/owl-s/1.1

[22] WSDL-S, http://lsdis.cs.uga.edu/projects/WSDL-S/wsdl-s.pdf

[23] WSMO project, http://www.wsmo.org/TR/d16/d16.1/v0.21/20051005/
http://www.wsmo.org/wsml/wsml-syntax#

[24] A.Lazcano, H. Schuldt, G. Alonso, and H.-J. Schek. WISE, "Process based e-commerce". IEEE Data Engineering Bulletin, 24(1) :46–51, 2001.

[25] N. Mehandjiev, I. Stalker, K. Fessl, and G.Weichhart, "Interoperability contributions of crosswork", In invited short paper to Proceedings of INTEROP-ESA’05 Conference, Geneva, February 2005. Springer-Verlag.

[26] Y. Hoffner, H. Ludwig, C. Gülcü, and P. W. P. J. Grefen,"An architecture for crossorganizational business processes", In Second International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems, Milpitas, 2000.

[27] D. Grigori.," Eléments de flexibilité des systèmes de workflow pour la définition et l’exécution de procédés coopératifs", PhD thesis, Thèse en informatique, Université Henri Poincaré-Nancy 1, LORIA, Novembre 2001

[28] OWL, “OWL Web Ontology Language Reference, W3C Recommendation, (2004) http://www.w3.org/TR/2004/REC-owl-ref20040210/,

[29] MDA, http www. omg.org/mda.

[30] WfMC, http://www.wfmc.org.

[31] S.Bandinelli, et al, "Modeling and Improving an Industrial Software Process, IEEE Transactions on Software Engineering, Vol. 21, No. 5, 1995.

[32] J. C Derniame, "Software Process: Principles, Methodology and Technology", LNCS 1500, Springer, Berlin, Germany, 1998.

[33] T. A Brathaug,., and T. A. Evjen," Enterprise Modeling" , STF 38 A96302 ,1996, Trondheim, Norway.

[34] M. S. Fox, and M. Gruninger," Enterprise Modeling", AI Magazine, 2000.

[35] V. Ambriola, et al, "Assessing Process-Centered Software Engineering Environments", ACM Transactions on Software Engineering and Metho, 6, 3, 1997.

[36] The Workflow Mangement Coalition, Workflow Management Coalition Terminology and Glossary, technical report WfMC-TC-1011, February 1999

[37] Gregory Alan Bolcer et Gail Kaiser : " SWAP : Leveraging the Web to manage Workflow", IEEE Internet Computing, Janvier, Février 1999.

[38] D.Georgakopoulos, M. Hornick, A Sheth, "An Overview of Workflow Management : From Process Modeling to Workflow Automation infrastructure", Distributed and Parallel DataBases, An International Journal, 3,1995.

[39] Thomas Schaël, "Théorie et Pratique du Workflow", Springer-Verlag,1997

[40] T.Courtois, "Logiciels & Systèmes N°11, P.46, Septembre-Octobre 1996.

[41] S.Joosten, S.Brinkemper, "fundamental Concepts forWorkflow Automation in Practice", ICIS'95 conference, AMSTERDAM, Décembre 1995.

[42] S.Nurcan, "Analyse des et Conception des Systèmes d'Information Coopératives", Techniques et Sciences Informatiques, Vol.15, n°9, pp1287-1315,1996.

[43] http://fr.wikipedia.org

[44] D. Florent, "Application de Workflow Mise en place dans une Entreprise Industrielle", Master Spécialisé Management des Systèmes d'Information et des Technologies, école des mines de Paris, Novembre 2004.

[45] S.K Levan," Le projet Workflow, Concepts et Outils au Service des Organisations ", Concepts et Outils au service des Organisations, Edition Eyrolles 2000.

[46] K.Saïkali : « Flexibilté des Workflows par l'Approche Objet : 2Flow, un Framework pour Worklows Flexibles », Thèse de Doctorat Ecole Centrale de Lyon, France, Septembre 2000.

[47] "ISO, Life Cycle Management — System Life Cycle Processes ", Document ISO/IEC 15288, janvier 2000, http://www.incose.org/stc/news_ISO15288cd2.htm

[48] R. Medina-Mora, et al, "The Action Workflow approach to workflow management technology", ACM, Proceedings of the Conference On Computer-Supported Coooperative Work, Toronto, November, 1992

[49] F. Vernadat, " techniques de modélisation en entreprise : Applications aux processus Opérationnels ", économica 1999.

[50] R.E.Shelton, "Business Objects -Workflow ". Data Management Review, Mars 1996.

[51] T.H Davenport, J.Short, " The new Industrial Engineering : Information Technology and Business Process redesign ".Sloan Management Review, 1990, pp11-27

[52] C. Morley, "La modélisation des processus : typologie et proposition utilisant UML. Processus et Systèmes d'information", Journées ADELI, France, 2002.

[53] C. Morley, J. Hugues, B. Leblanc, O. Hugues, Processus métiers et S.I : évaluation, modélisation, mise en oeuvre, éditions DUNOD, mars 2005.

[53] F. Théroude, " Formalisme et système pour la représentation et la mise en œuvre des processus de pilotage des relations entre donneurs d'ordre et fournisseurs", thèse de Doctorat de l'Institut National Polytechnique de Grenoble, 2002.

[54] I. Chebbi and S. Tata. " CoopFlow : a framework for inter-organizational workflow cooperation ", In Proceedings of International Conference on Cooperative Information Systems, pages 112–129, Agia Napa, Cyprus, 31 Oct-4 Nov 2005

[55] B.Axenath, et al, "The Aspects of Business Processes: An Open and Formalism Independent Ontology", Technical report, Department of the Unversity of Paderborm, Germany, April 2005. http://www.upb.de/cs/kindler/publications/copies/AKR05.pdf

[56] Workflow Management Coalition: The reference Model, WfMC Document Number TC-1003, issue 2.0, Jan 1995.

[57] Workflow Management Coalition, Interoperability Abstract Specification, WfMC Document Number TC-1012, new version, June 2000.

[58] WfMC. Workflow standard-interoperability, wf-xml binding. Wfmc-tc-1023, v. 1.0, May 2000

[59] WfMC. Workflow Management Application Programming Interface (interface 2 and 3) specification. Wfmc-tc-1009, v. 2.0,, July 1998.

[60] Akazi Technologies, "Le Business Process Management et l'Administration Electronique", Akazi Technologies, Conférence Technoforum Novembre 2002, http://www.akazi.com/

[61] G. Bretin, P.Dahne, " ISNET40 : la vague EAI , la gestion des processus, Décembre 2003. version1.1.'

[62] C.Tanguy, "'Business Process Management, De la modélisation à l'exécution, positionnement par rapport aux architectures orientées services, INTALLIO", 2003.

[63] OMG, Workflow Process Definition. OMG Document BOM/99-10-03, November 1999.

[64] OMG, " Workflow Management Facility Specification" Version 1.2. OMG, April 2000

[65] OMG, " Corba facilities architecture specification",www.omg.org/corba/cf2.html. Technical report, http://www.omg/org, 1998

[66] OMG, "The Common Object Request Broker - Architecture and Specifications", Revision 2.6.Omg document formal/o1-12-01, http://www.omg/org, December 2001

[67] T.Dufresne, J.Martin," Process Modeling for E-Business ", INFS 770- Methodes for Information Knowledge Management and E-Business, University Mason , 2003.

[68] B Curtis, et al, "Process Modeling", Communications of the ACM 35, 1992.

[69] S. Nurcan, Business process modeling and flexibility, Enterprise interoperability: New challenges and approaches II, Springer 2007.

[70] C. Morley, et al, " Processus métiers et S.I : évaluation, modélisation, mise en oeuvre, éditions DUNOD, mars 2005.

[71] Y. Mougin, La cartographie des processus, économica, p.1-8, 2002.

[72] E.Breton,"Contribution à la représentation de processus par des techniques de méta-modélisation", thèse de Doctorat, université de Nantes (France), Juin 2002.

[73] M. Bernauer, et al, "Specification of interorganizational workflows - a comparison of approaches", In Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics, USA, July 2003.

[74] H. Weigand, A. H. H. Ngu,": Flexible specification of interoperable transactions. Data & Knowledge Engineering", Vol. 25, 1998.

[75] F. B Vernadat., "Enterprise modelling and integration: Principles and applications". Chapman & Hall, London, 1996.

[76] Weston R. H., "Steps towards enterprise wide integration: a definition of needs and first generation open solutions". International Journal of Production Research, 31(9), 2235-2254, 1993.

[77] Williams H., Li F, " Interfirm collaboration through interfirm network ", Information Systems Journal, n° 9 :103-115, 1999.

[78] European Interoperability Framework for pan-European e-Government Services, Interoperable Delivery of European e-Government Services to public Administrations, Businesses and Citizens (IDABC), November, Luxembourg, 2004.

[79] IEEE (Institute of Electrical and Electronics Engineers). Standard Computer Dictionary-A Compilation of IEEE Standard Computer Glossaries.. ISBN: 1559370793, 1990.

[80] D. Carney et al, "Topics in Interoperability : System-of-Systems Evolution, Integration of Software-Intensive Systems Initiative, Technical Note, CMU/SEI-2005-002, March 2005.

[81] IDEAS, A gap Analysis –Required activities in Research, Technology and standardisation to close the RTS Gap- Roadmaps and Recommendations on RTS activites, IDEAS, Deliverables, 2003.

[82] R.Klischewski,"Information Integration or Process Integration: How to Achieve Interoperability in Administration", EGOV04 at DEXA, Zaragoza, Spain,September, 2004.

[83] W. M. P. van der Aalst. ""Process-Oriented Architectures for Electronic Commerce and Interorganizational Workflow". Information Systems, 24(9):639–671, 1999.

[84] W. M. P. van der Aalst. "Loosely Coupled Interorganizational Workflows: Modeling and Analyzing Workflows Crossing Organizational Boundaries". Information and Management, 2000.

[85] W. M. P. van der Aalst. "Interorganizational Workflows : an Approach Based on Message Sequence Charts and Petri nets. Systems Analysis - Modelling - Simulation," 1999

[86] R. Tagg." Workflow in Different Styles of Virtual Enterprise." In ITVE '01 : Proceedings of the workshop on Information technology for virtual enterprises, pages 21–28, Washington, DC, USA, 2001. IEEE Computer Society.

[87] J. Grabowski, et al, "Towards a Petri Net based semantics definition for Message Sequence Charts. In Proceedings of the sixth SDL (Specification and Description Language), pages 179–190, North-Holland, 1993.

[88] S. Mauw and M. Reniers," An algebraic semantics of basic message sequence charts", In The Computer Journal, 37(4), pages 269–277, 1994.

[89] ITU-TS. ITU-TS Recommendations Z.120 : Message Sequence Chart, Geneva, 1996

[90] W. M. P. van der Aalst and M. Weske. "The P2P Aapproach to Interorganizational Workflows". In Proceedings of the 13th International Conference on Advanced Information Systems Engineering, pages 140–156. Springer-Verlag, 2001.

[91] W. M. P. van der Aalst. Inheritance of interorganizational workflows to enable business-to business E-commerce. Electronic Commerce Research, 2002.

[92] W. Schulze, et al "Services of Workflow Objects and Workflow meta-objects in omg-compliant Environments. In Proceedings of the 1996 Object-Oriented Programming, Systems, Languages & Applications (OOPSLA) Workshop on Business Object Design and Implementation, San Jose (CA), 1996

[93] W. Schulze, "Fitting the Workflow Management Facility into the Object Management Architecture". 3rd Workshop on Business Object Design and Implementation (OOPSLA'97), pages 109–117, USA, 1997.

[94] R. Snodgrass, " The Interface Description Language: Definition and Use. Computer Science Press, USA, 1989.

[95] CrossWork. project homepage : http ://www.crosswork.info/.

[96] N. Mehandjiev, et al ," Interoperability Contributions of CrossWork", In invited short paper to Proceedings of INTEROP-ESA'05 Conference, Geneva, February 2005.

[97] H. M. W. Verbeek, et al, " XRL/Woflan : Verfication and extensibility of an XML/Petri-net-based Language for Inter-organizational Workflows", Inf. Tech & Manag, 2004

[98] ebXML. http ://ebxml.org., 2002..

[99] OMG, http://www.omg.org.

[100] BPMI, Business Process Management Initiative, BPMI.org, 2003, http://www.bpmi.org

[101] D.Lévy, "Coordination des Web services : langages de description et plates-formes d'exécution ", XRCE Grenoble, Mémoire septembre 2002.

[102] G. Alonso, et al, "WISE: Business to Business e-commerce. In International Workshop on Research Issues on Data Engineering : Information Technology for virtual Enterprises, Australia, March 1999.

[103] A. Lazcano, et al, "The WISE Approach to Electronic Commerce", In International Journal of Computer Systems Science & Engineering, special issue on Flexible Workflow Technology Driving the Networked Economy, vol. 15(No. 5), 2000.

[104] O. Perrin and C. Godart, "A Model to Support Collaborative Work in Virtual Enterprises", Data Knowledge Engineering, 2004.

[105] P. Grefen, et al, " Cross-Organizational Workflow Management for Service Outsourcing in Dynamic Virtual Enterprises ", IEEE Data Engineering Bulletin, 2001.

[106] Y. Hoffner, et al, "Contract-Driven Creation and Operation of Virtual Enterprises. Computer Networks, 2001.

[107] Y. Hoffner, et al, "An architecture for Cross-Organizational Business Processes", In Second International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems, Milpitas, 2000.

[108] M. Koetsier, et al, "Contracts for Cross-Organizational Workflow Management", In Proceedings of the First International Conference on Electronic Commerce and Web Technologies (EC-WEB'00), London, UK, 2000.

[109] OMG. CORBA/IIOP 2.3 Specification, www.omg.org/docs/ptc/98-12-04.pdf. Technical report, http://www.omg/org, 1998.

[110] I. Chebbi and S. Tata,""CoopFlow : a Framework for Inter-Organizational Workflow Cooperation", In Proceedings of International Conference on Cooperative Information Systems, pages 112–129, Agia Napa, Cyprus, 31 Oct-4 Nov 2005.

[111] NIIP, The NIIP reference architecture, http ://www.niip.org, 1996.

[112] H. Richards, et al, "Flow of Orders Through a Virtual Enterprise", Computing & Control Engineering Journal, pages 173–179, August 1997.

[113] ACE-FLOW, project homepage,

http ://www.ifi.unizh.ch/dbtg/projects/aceflow/index.html, 1999.

[114] MARVELOUS. Web page :

http ://www.lar.ee.upatras.gr/icims/related_projects/marvelous.htm, 1997.

[115] D. Konstantas, et al, "Interorperability of Enterprise Software and Applications INTEROP-ESA'05)," 1st International Conference, Geneva, Springer-Verlag, p. v-vi, ISBN-10 1-84628-151-2, February 2005.

[116] R. J. Mayer, "Information Integration for Concurent Engineering (IICE), DEF3 Process Description Capture Method Report". AL-TR-81-4023, Air Force Material Laboratory, Wright Patterson AFB, Ohio 45433, 1995.

[117] RosettaNet, "Rosetta Implementation Framework (RNIF): Core specification". Technical report, RosettaNet, 2002.

[118] Lee J., el al, "The PIF Process Interchange Format and Framework,Version 1.2 ", The Knowledge Review Engineering, Vol.13, No.1 pp. 91-120, Cambridge University, mars 1998, http://spot.colorado.edu/~jintae/pif/pif12.rtf.

[119] OMG. XML Metadata Interchange (XMI) Specification, version 2.1, formal/03-05-02, formal/05-09-01.

[120] OMG. Meta Object Facility (MOF) specification - version 2.0, formal/2006-01-01: January 2006, http://www.omg.org/spec/MOF/2.0/pdf.

[121] INTEROP, (Interoperability research for Networked Enterprises Applications and Software "Deliverables of INTEROP Project", http://www.interop-ne.org, 2005.

et http://interop-vlab.eu/.

[122] Athena, "Athena Project deliverables",.http://www.athena-ip org, 2005

[123] A. Opdahl, G. Berio, "A Roadmap for UEML, Enterprise interoperability:New challenges and approaches," Springer edition. ISBN: 978-1-84628-713-8, p.189-198, 2006.

[124] L., Brown," What is Enterprise Application Integration ", Septembre 2001.

http://www.systeminnovations.net/definingEAI.htm

[125] Microsoft Corporation and Digital Equipment Corporation, "The Component Object Model Specification, Version 0.9 ". 1995, http://groups.csail.mit.edu/medg/ftp/emjordan/COM/

[126] Microsoft Corporation, "Distributed Component Object Model Protocol DCOM 1.0", Network working group, 1996.

[127] Cloux P. Y., Doussot D., and Géron A., "Technologies et Architectures Internet: Corba, COM, XML, J2EE, .NET, Web Services". Dunod, 2003.

[128] S.Hamri, " A Semantic Integration of Workflows Models ", in Proceedings of CISC'04, The International Conference on Complex Systems, September 6-8, 2004, University of Jijel, Algeria., pp 29-36.

[129] S.Hamri, " Une approche Orientée MDA pour l'Interopérabilité des Workflows d'Entreprise ", In Proceedings of SNIB'08, 6ème Séminaire National en Informatique de Biskra, 6-8 mai 2008, Université de Biskra, Algérie, pp 99-106.

[130] S.Hamri, M.Boufaida, N.Boudjlida. " An Architecture for the Interoperability of Workflow Models ", In Proceedings of the first international workshop on Interoperability of Heterogeneous Information Systems (IHIS'05), ACM Workshop, 04 Nov 2005; Bremen, Germany (Allemagne), pp 31-37, ISBN : 1-59593-184-5.

[131] S.Hamri, N. Boudjlida and M. Boufaida. " An Approach for Building an OWL Ontology for Workflow Interoperability", In Proceedings of the 3rd International Conference on Interoperability of Enterprise Software and Applications, Madera, Portugal, 26-30 March 2007, pp.357—364. Springer-Verlag, ISBN: 978-1-84628-857-9

[132] S.Hamri, "Bridging MDA and OWL for Workflow Interoperability", in International Review on Computers and Software (IRECOS), Praise Worthy Prize, ISSN 1828-6003. Vol.4, N. 3 May 2009, pages 374 – 381.

[133] J.Fenner, "Enterprise Application Integration Techniques", UCL Computer Science, 2002, http://www.cs.ucl.ac.uk/staff/W.Emmerich/lectures/3C05-02-03/aswe21-essay.pdf

[134] M.Chevassus, "Enterprise Application Integration : EAI", Techniques de l'ingénieur, H2915, pp.1-20, 2005

[135] 01Net, "Intégration, Nouvelle Génération". 01Net, 2006.

[136] D. Thévenon, et al, " Réussir votre Intégration: XML, EAI, WS". Revue Programmez! Hors Série, Go-o2 SARL, 2003.

[137] L. Blanc," Modélisation des Processus Métier dans une Approche EAI ", thèse de doctorat, Université de Savoie, 2004

[138] D. Chappell., "Enterprise Service Bus", media Inc., 2004.

[139] B. Lublinsky, and D. Tyomkin, "Dissecting Service Oriented Architetcures". Bunsiness Integration Journal, 2003.

[140] K.Kanetkar, " A Roadmap to Building an ESB, Integration for Enterprise", Boston, May 2006, http://www.saterisystems.com/Docs/whitepapers/Roadmap/ESB.pdf.

[141] U. Dayal, et al, "Business Process Coordination: State of the Art, Trends, and Open Issues". In Proceedings of the 27th International Conference on Very Large DataBases, 2001.

[142] P. Johannesson and E. Perjons, "Design Principles for Process Modelling in Enterprise Application Integration", Information System Vol. 26, pp. 165-184, 2001.

[143] D. S Linthicum, "Next Generation Application Integration", Addison-Wesley, 2004.

[144] K.Kontogiannis, et al, "On the Role of Services in Enterprise Application Integration", In the Proceedings of the 10th International Workshop on Software Technology and Engineering, 2002.

[145] H.Kadima and V.Monfort, "Les Web Services". Dunod, 2003.

[146] F. Curbera, et al, " Web Services : Why and how? " Conference on Object-Oriented Programming, Systems, Languages and Application (OOPSLA), Workshop on Object-Oriented Web Services, 2001

[147] W3C. Web Services Architecture Requirements. W3C,
http://www.w3c.org/2002/- ws/arch/2/wd-wsawg-reqs-04232002.html.

[148] J. L Zhao, H. K. Cheng, "Web Services and Process Management : a Union of Convenience or a New Area of Research" Decision Support Systems, Volume 40, Issue 1, July 2005.

[149] http://www.webopedia.com

[150] M.P. Wil et al, "Web Service Composition Languages: Old Wine in New Bottles, 29th EuroMicro Conference, September 2003.

[151] S.Izza, et al, "Ontology Urbanization for Semantic Integration: Dealing with Semantics within Large Enterprises of Heterogeneous and Autonomous Application Systems". The 9th IEEE International EDOC Conference, Enschede, The Netherlands, pp. 83-94, 2005.

[152] G. Raymond, "SOA : Architecture Logique, Principes, Structures et Bonnes Pratiques, www.softeam.fr, 2007.

[153] Object Management Group, "OMG Model Driven Architecture", http://www.omg.org/mda, 2006.

[154] J.Bézivin, et X. Blanc "MDA: Vers un important changement de paradigme en génie logiciel". Développeur Référence, http://www.devreference.net/, 2002.

[155] XSL Transformations (XSLT) Version 2.0, W3C Recommendation 23 January 2007.

http://www.w3.org/TR/2007/REC-xslt20-20070123/

[156] J. Bézivin, el al, "First Experiments with the ATL Model Transformation Language: Transforming XSLT into Xquery, 2nd OOPSLA Workhop on Generative Techniques in the context of Model Driven Architecture, October 2003.

[157] http://modfact.lip6.fr/ModFactWeb/index.jsp

[158] M.Peltier, Techniques de transformation de modèles basées sur la méta- modélisation, thèse de Doctorat, Université de Nantes, 2003.

[159] S. Baina, el al, "Apport de l'approche MDA pour une interopérabilité sémantique", revue Ingénierie des Systèmes d'Information (ISI), Numéro spécial Ingénierie des processus d'entreprise et des SI, Vol. 11, N° 3, pp 11-29, Hermès, Paris, 2006.

[160] D.C.P Lopes, "Etude et applications de l'approche MDA pour des plates-formes de Services Web", thèse de Doctorat, Université de Nantes, juillet 2005.

[161] S.McIlraith et al, "Semantic Web Services", IEEE Intelligent Systems Vol. 16, No. 2, pp. 46-53, 2001.

[162] C.Bussler, "Semantic Web Services: Reflections on Web Service Mediation and Composition". Proceedings of the Fourth InternationalConference on Web Information Systems Engineering (WISE'03), 2003.

[163] C. Bussler., "Semantic Web Services". ICWE'04, Munich, 2004.

[164] P.Kellert, and F.Toumani, "Les Web services Sémantiques", Research Report LIMOS/RR 03-1511, 2003.

[165] D.Fensel, and C.Bussler, "The Web Service Modeling Framework WSMF". In Electronic Commerce Research and Applications, 1(2), Elsevier, Scinces B. V., 2002.

[166] W3C : « Semantic Annotations for WSDL and XML Schema »,
http://www. w3.org/TR/2007/REC-sawsdl-20070828, août 2007.

[167] "METEOR-S project deliverables", http://lsdis.cs.uga.edu/Projects/METEOR-S/, 2005.

[168] C. Liliana, et al, " IRS-III: A Broker for Semantic Web Services Based Applications ", Knowledge Media Institute, The Open University, UK, LNCS 4273, pp 201-214, Berlin / Heidelberg, ISSN 0302-9743, Volume 4273, 2006.

[169] M.Crubézy, and M.Musen, "Ontologies in Support of Problem Solving". In Staab S. and Studer R., editors, Handbook on ontologies, pp. 321-342 Springer, 2003.

[170] M.Crubezy et al, " The Language and Tool Support for Makiing the Semantic Web Alive. Technical report, MIT, 2003.92

[171] H.B. et al, "Comparative Study of Semantic web Services ",In Journal of Information & Communication Technology, Vol. 1, No. 1, (Spring 2007) 01-10

[172] Web Services Description Language (WSDL) 2.0, W3C Recommendation 26 June 2007, http://www.w3.org/TR/2007/REC-wsdl20-20070626

[173] C., Liliana, et al, "Approaches to Semantic Web Services: An Overview and Comparisons", http://eprints.aktors.org/326/01/cabralESWS04.pdf.

[174] R.Lara et al, "A Conceptual Comparison of WSMO and OWL-S", pp. 254-269, Proceedings of the 2004 European Conference on Web Services, Liang-Jie Zhang, Mario Jeckle (Ed.), LNCS, Germany, Vol. 3250, pp. 254-269, September 2004.

[175] M. Paolucci, M. Wagner "Grounding OWL-S in WSDL-S ", Proceedings of the IEEE International Conference on Web Services, Pages: 913 - 914, 2006

[176] M. Paolucci, et al, " Grounding OWL-S in SAWSDL" LNCS; Vol. 4749, Proceedings of the 5th international conference on Service-Oriented Computing, Vienna, Austria, Pages: 416 – 421, 2007.

[177] G.C. R.J.Brodie and J.T E. Timm, ''Foundations for specifying OWL-S groundings'', In International Journal Business Process Integration and Management, Vol. 2, No. 1, 2007.

[178] M. Sabou , "Building Web Service Ontologies", thèse 2006
http://people.kmi.open.ac.uk/marta/papers/thesis.pdf.

[179] http://www.eclipse.org/

[180] http://www.eclipse.org/m2m/atl/doc/

[181] Ontology Definition Metamodel , Sixth Revised Submission to OMG/ RFP ad/2003-03-40, http://www.omg.org/cgi-bin/ doc?ad/06-05-01.pdf.

[182] D.Djuric, D.Gasevic and V.Devedzic, " Ontology Modeling and MDA", Journal on Object Technology, Vol 4, N 1, January-February 2005, Online at http://www.jot.fm.

[183] J. A..César, M.Esperanza "Modeling Semantic Web Services: A Case Study ", ICWE'06, ACM , Palo Alto, California, USA, 2006

[184] G.C. Gannod et al, ''Facilitating the Specification of Semantic Web Services Using Model-Driven Development '' in International Journal of Web Services, July-September 2006.

[185] G.C. Gannod and J.Timm., "An MDA-based Approach for Facilitating Adoption of Semantic Web Service",. In Proceedings of the IEEE EDOC, Workshop on MDSW, September 2004.

[186] D. Elenius et al , "The owl-s Editor - a development tool for semantic web services ". In Proceedings of the Second European Semantic Web Conference, May 2005.

[187] J.T.E Timm and G. C.Gannod, 'A model-driven Approach for Specifying Semantic Web Services', Proceedings of the 3rd International Conference on Web Services, 2005, IEEE.

[188] M. Paolucci el al. "Toward a Semantic Choreography of Web Services: From WSDL to DAML-S" In Proceedings of the First International Conference on Web Services (ICWS'03).

[189] The MINDSWAP Group. Owl-s API. http://www.mindswap.org/2004/owls/validator

[190] http://projects.semwebcentral.org/projects/owlseditor/

[191] Protégé, http://protégé.stanford.edu/plugin.

[192] RDF validator, http://www.w3.org/ RDF/ Validator/

[193] ConsVISor, http://www.vistology.com/consvisor/

[194] M.C. Jaeger el al, "A methodology for developing owl-s descriptions. In First International Conference on Interoperability of Enterprise Software and Applications Workshop on Web Services and Interoperability, February 2005.

[195] D. Djurić et al, " MDA-based Ontology Infrastructure," International Journal on Computer Science and Information Systems , Vol. 1, No. 1, 2004.

[196] Gašević, D et al, "Bridging MDA and Ontologies " , Journal of Web Engineering, Vol, 4, No.2 (2005) 118-143.

[197] C.Gerald et al ‘‘An interactive approach for specifying owl-s groundings ’’, In Proceedings of the 9th IEEE Enterprise Computing Conference (EDOC’05), September 2005.

[198] J.T E. Timm, G.C Gannod ‘‘Specifying Semantic Web Service Compositions using UML and OCL ’’, in International Conference on Web Services (ICWS 2007), 9-13 July 2007

[199] D.Frankel and J.Parodi, “Using Model-Driven Architecture TM to DevelopWeb Services’’. Rapport technique, IONA Technologies PLC, April 2002

[200] B.Bordbar and A.Staikopoulos, “Automated Generation of Metamodels for Web Service Languages”. Second European Workshop on Model Driven Architecture (MDA) with an emphasis on Methodologies and Transformations (EWMDA), September 2004.

[201] J Bezivin, et al, "Applying MDA Approach for Web Service Platform", 8th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2004), pages 58–70, September 2004.

[202] D.Frankel et J.Parodi, "Using Model-Driven Architecture TM to DevelopWeb Services". Rapport technique, IONA Technologies PLC, April 2002.

[203] J.Bézivin, et al, "Applying MDA Approach to B2B Applications: A Road Map", Workshop on Model Driven Development (WMDD2004) – at ECOOP 2004, Oslo, Norway, June 15, 2004.

[204] J.Bézivin, et al, "An Experiment in Mapping Web Services to Implementation Platforms", Research Report, RR-IRIN2004-01, France, March 2004.

[205] D. Lopes and S.Hammoudi, Web Services in the Context of MDA, The 2003 International Conference on Web Services (ICWS2003), Las Vegas, USA, June 23-26, 2003.

[206] W3C. Web Services Description Language (WSDL) 2.0 Part1: Core Language, W3C Working Draft, November 2003.

[207] R.Gronmo, et al, ‘’Model-Driven Web Services Development’, 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE’04), pp 42–45, March 2004.

[208] SINTEF. UML Model Transformation Tool (UMT), december 2004. Disponible sur http://umtqvt.sourceforge.net,

[209] G.C. Gannod, et al, " Foundations for specifying OWL-S groundings’’, in International Journal Business Process Integration and Management, Vol. 2, No. 1, 2007.

[210] N. Boudjlida and H. Panetto,".Enterprise Semantic Modelling for Interoperability". In Proceedings of the 12th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2007, September 25-28 2007, Patras, Greece.

[211] Y.Lin, D. Strasunskas, "Ontology-based Semantic Annotation of Process Templates for Reuse". In Castro, J. and Teniente, E. (Eds.): Proc. of the CAiSE’05 Workshops Vol.1 (EMMSAD Workshop). Porto, Portugal (2005) 593-604.

[212] Y.Lin and H. Ding," Ontology-based Semantic Web Annotation for Semantic Interoperability of Process Models". In Proc. of the Int’l Conf. on Computational Intelligence for Modeling, Control and Automation, Vienna, Austria (2005)

[213] Y. Lin et al, "Semantic Annotation Framework to Manage Semantic Heterogeneity of Process Models". In: Dubois, E., Pohl, K.(eds.) CAiSE 2006. LNCS, vol. 4001, pp. 433–446.

[214] G.Miller et al, "Introduction to WordNet: An on-line lexical database", http://wordnetweb.princeton.edu/perl/webwn.

[215] Van der Aalst,et al, "Advanced Workflow Patterns. In Proc. of the 7th Int'l. Conf. on Cooperative Information Systems, LNCS 1901, Springer-Verlag (2000) 18-29.

[216] C. Diamantini and N. Boudjlida, "About Semantic Enrichment of Strategic Models as Part of Enterprise Models", In Proceedings of the 2nd Interntional Workshop on Enterprise and Networked Enterprises Interoperability, in conjunction with the 4th International Conference on Business Process Management. Pages 348--359.

|217] N. Boudjlida. "Semantic Annotations for Interoperability". In International Research School: "Ontologies: a Smart Way Toward Interoperability". CNRS GDR I3 and GDR MACS, IAE, Paris, April 13, 2006.

[218] J.Krogstie,, D.H Jørgensen," Interactive Models for Supporting Networked Organisations, In Proc. 16th Intl. Conf. on Advanced Information Systems Engineering, LNCS 3084, Springer-Verlag (2004) 550-562

[219] http://www.troux.com.

[220] A.V. Lamsweerde,"Goal-oriented requirements engineering: a guided tour". In 5th International Conference on Requirements Engineering, pages 249–262, 2001.

[221] N.Mellal, "Réalisation de l'interopérabilité sémantique des systèmes, basée sur les ontologies et les flux d'information", Thèse de Doctorat, Université de Savoie (France), 2007.

[222] Yu, E. J. Mylopoulos," Why goal-oriented requirements engineering. In: Proc. of the 4th of International Workshop on Requirements Egnineering: Foundations of Software Quality (1998) http://www.cs.toronto.edu/pub/eric/REFSQ98.html.

[223] M.Lin, , H.Guo, J.Yin," Goal description language for semantic web service automatic composition", In: Proc. of IEEE/IPSJ International Symposium on Applications and the Internet (SAINT 2005), 31 January - 4 February 2005, Trento, Italy. pp. 190–196 (2005)

[224] R.Darimont et al, "Grail/kaos: An environment for goal-driven requirements analysis, integration and layout. 1997.

[225] i*: An agent-oriented modelling framework. http://www.cs.toronto.edu/km/istar/

[226] SCOR: SCOR model.

http://www.supply-chain.org/page.ww?section=SCOR+Model&name= SCOR+Model.

[227] Y. Lin and A.Sølvberg," Goal annotation of process models for semantic enrichment of process knowledge". In Proc. of the 19th International Conference on Advanced Information Systems Engineering (CAiSE 2007) pages 355–369, Norway. LNCS 4495, 2007.

[228] J. Bézivin, "Who's Afraid of Ontologies", OOPSLA'98 Workshop : Model Engineering, Methods and tools Integration with CDIF, Vancouver, October 1998, http://www.metamodel.com/oopsla_cdif-workshop/bezivin1/.

[229] P.Soffer and Y.Wand, "On the notion of soft-goals in business process modeling," Business Process Management Journal 11, 663–679 (2005).

[230] J. Mylopoulos, L. Chung, E.Yu, "From object-oriented to goal-oriented requirements analysis. Commun",. ACM 42, 31–37 (1999).

[231] B. Benatallah, M-S. Hacid, C. Rey & F. Toumani, "Semantic Reasoning for Web Services Discovery", WWW Workshop on E-Services and the Semantic Web, Budapest, Hungary. (2003).

[232] A. Bernstein & M. Klein, "Discovering Services: Towards High Precision Service Retrieval". In CaiSE workshop on Web Services, e-Business, and the Semantic Web: Foundations, Models, Architecture, Engineering and Applications. Toronto, Canada. (2002).

[233] D. Chakraborty, F. Perich, S. Avancha, & A. Joshi D.Reggie, "Semantic Service Discovery for M-Commerce Applications". In Workshop on Reliable and Secure Applications in Mobile Environment, 20th Symposium on Reliable Distributed Systems, (2001), pages 28–31.

[234] J. Gonzàlez-Castillo, D. Trastour, & C. Bartolini, "Description Logics for Matchmaking of Services". In KI-2001 Workshop on Applications of Description Logics Austria, Sept. (2001),. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-44

[235] M. Paolucci, T. Kawamura, T.R. Payne, & K.P. Sycara, "Semantic Matching of Web Services Capabilities", In Int. Semantic Web Conference, Sardinia, Italy, (2002). pages 333–347.

Annexe

Le processus MDA basé sur les transformations est long. Néanmoins nous présentons dans cette annexe, un extrait du code ATL qui nous a permis de générer une ontologie OWL-S d'un service de livraison d'une commande d'achat (Purchase Order Shipping) que nous avons présenté dans l'étude de cas.

```
-- \\\\\\\\\\ Transformations UPSWS VERS OUPSWS \\\\\\\\\\\\

module UPSWS2OUPSWS;
create Oservice : OUPSWS from service : UPSWSservice;
rule Package2Package {
                    from
                    a : UPSWSservice!Package
                    to
                    p:OUPSWS!Package(name<-a.name,profileApplication<-
a.profileApplication,packagedElement<-a.packagedElement,
                    packageMerge<-a.packageMerge,
                    nestingPackage<-a.nestingPackage),
                    O:OUPSWS!Ontology(base_Package<-p)
      }
 helper context UPSWSservice!Class def : getDefNameSet() : String =
      UPSWSservice!Service.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc +'Service')+
      UPSWSservice!ServiceProfile.allInstances()->select(S | S.base_Class
= self)->iterate(S;acc : String=''|acc +'ServiceProfile')+
      UPSWSservice!ServiceModel.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc +'ServiceModel')+
      UPSWSservice!ServiceGrounding.allInstances()->select(S |
S.base_Class = self)->iterate(S;acc : String=''|acc +'ServiceGrounding')+
      UPSWSservice!profile.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc +'Profile')+
      UPSWSservice!Serviceparameter.allInstances()->select(S |
S.base_Class = self)->iterate(S;acc : String=''|acc+'ServiceParameter')+
      UPSWSservice!Servicecategory.allInstances()->select(S |
S.base_Class = self)->iterate(S;acc : String=''|acc +'ServiceCategory')+
      UPSWSservice!Input.allInstances()->select(S | S.base_Class = self)-
>iterate(S;acc : String=''|acc+'Input')+
      UPSWSservice!Output.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc+'Output')+
      UPSWSservice!parameter.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc+'Parameter')+
      UPSWSservice!Process.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc+'Process')+
      UPSWSservice!Condition.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc+'Condition')+
      UPSWSservice!Result.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc+'Result')+
      UPSWSservice!variable.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc+'Variable')+
      UPSWSservice!SimpleProcess.allInstances()->select(S | S.base_Class
= self)->iterate(S;acc : String=''|acc+'SimpleProcess')+
      UPSWSservice!CompositeProcess.allInstances()->select(S |
S.base_Class = self)->iterate(S;acc : String=''|acc+'CompositeProcess')+
      UPSWSservice!AtomicProcess.allInstances()->select(S | S.base_Class
= self)->iterate(S;acc : String=''|acc+'AtomicProcess')+
      UPSWSservice!Participant.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc+'Participant')+
      UPSWSservice!Perform.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc+'Perform')+
      UPSWSservice!Local.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String=''|acc+'Local')+
```

```
UPSWSservice!ResultVar.allInstances()->select(S | S.base_Class = self)-
>iterate(S;acc : String='' |acc+'ResultVar')+
      UPSWSservice!OutputBinding.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String='' |acc+'OutputBinding')+
      UPSWSservice!ControlConstruct.allInstances()->select(S | S.base_Class
= self)->iterate(S;acc : String='' |acc+'ControlConstruct')+
      UPSWSservice!Binding.allInstances()->select(S | S.base_Class = self)-
>iterate(S;acc : String='' |acc+'Binding')+
      UPSWSservice!MessageMap.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String='' |acc+'MessageMap')+
      UPSWSservice!OutputMessageMap.allInstances()->select(S | S.base_Class
= self)->iterate(S;acc : String='' |acc+'OutputMessageMap')+
      UPSWSservice!InputMessageMap.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String='' |acc +'InputMessageMap')+
      UPSWSservice!WsdlOutputMessageMap.allInstances()->select(S |
S.base_Class = self)->iterate(S;acc : String='' |acc+'WsdlOutputMessageMap')+
      UPSWSservice!WsdlMessageMap.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String='' |acc+'WsdlMessageMap')+
      UPSWSservice!WsdlInputMessageMap.allInstances()->select(S |
S.base_Class = self)->iterate(S;acc : String='' |acc+'WsdlInputMessageMap')+
      UPSWSservice!WsdlOperationRef.allInstances()->select(S | S.base_Class
= self)->iterate(S;acc : String='' |acc+'WsdlOperationRef')+
      UPSWSservice!WsdlAtomicProcessGrounding.allInstances()->select(S |
S.base_Class = self)->iterate(S;acc :
String='' |acc+'WsdlAtomicProcessGrounding')+
      UPSWSservice!AtomicProcessGrounding.allInstances()->select(S |
S.base_Class = self)->iterate(S;acc :
String='' |acc+'AtomicProcessGrounding')+
      UPSWSservice!Grounding.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String='' |acc+'Grounding')+
      UPSWSservice!WsdlGrounding.allInstances()->select(S | S.base_Class =
self)->iterate(S;acc : String='' |acc+'WsdlGrounding')+
      UPSWSservice!ValueOf.allInstances()->select(S | S.base_Class = self)-
>iterate(S;acc : String='' |acc+'ValueOf');

 helper context UPSWSservice!Association def : getDefNameSet() : String =
      UPSWSservice!Presents.allInstances()->select(S | S.base_Association =
self)->iterate(S;acc : String='' |acc +'Presents')+
      UPSWSservice!DescribeBy.allInstances()->select(S | S.base_Association
= self)->iterate(S;acc : String='' |acc +'DescribedBy')+
      UPSWSservice!Supports.allInstances()->select(S | S.base_Association =
self)->iterate(S;acc : String='' |acc +'Supports')+
      UPSWSservice!hasInput.allInstances()->select(S | S.base_Association =
self)->iterate(S;acc : String='' |acc
      +'hasInput')+
      UPSWSservice!hasOutput.allInstances()->select(S | S.base_Association =
self)->iterate(S;acc : String='' |acc
      +'hasOutput')+
      UPSWSservice!serviceCategory.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String='' |acc
      +'serviceCategory')+
      UPSWSservice!serviceParameter.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String='' |acc
      +'serviceParameter')+
      UPSWSservice!hasParameter.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String='' |acc
      +'hasParameter')+
      UPSWSservice!has_process.allInstances()->select(S | S.base_Association
= self)->iterate(S;acc : String='' |acc
```

```
UPSWSservice!hasPrecondition.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'hasPrecondition')+
       UPSWSservice!hasResult.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'hasResult')+
       UPSWSservice!expandsTo.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'expandsTo')+
       UPSWSservice!realizedBy.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'realizedBy')+
       UPSWSservice!hasParticipant.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'hasParticipant')+
       UPSWSservice!hasLocal.allInstances()->select(S | S.base_Association
= self)->iterate(S;acc : String=''|acc
       +'hasLocal')+
       UPSWSservice!theVar.allInstances()->select(S | S.base_Association =
self)->iterate(S;acc : String=''|acc
       +'theVar')+
       UPSWSservice!fromProcess.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'fromProcess')+
       UPSWSservice!hasResultVar.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'hasResultVar')+
       UPSWSservice!inCondition.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'inCondition')+
       UPSWSservice!hasEffect.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'hasEffect')+
       UPSWSservice!withOutput.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'withOutput')+
       UPSWSservice!hasDataFrom.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'hasDataFrom')+
       UPSWSservice!toParam.allInstances()->select(S | S.base_Association
= self)->iterate(S;acc : String=''|acc
       +'toParam')+
       UPSWSservice!hasAtomicProcessGrounding.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'hasAtomicProcessGrounding')+
       UPSWSservice!owlsProcess.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'owlsProcess')+
       UPSWSservice!wsdlInput.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'wsdlInput')+
       UPSWSservice!wsdlOutput.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'wsdlOutput')+
       UPSWSservice!wsdlOperation.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'wsdlOperation')+
       UPSWSservice!owlsParameter.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
       +'owlsParameter')+
```

```
UPSWSservice!process.allInstances()->select(S | S.base_Association =
self)->iterate(S;acc : String=''|acc
      +'process');
rule Property2Property {
                    from
                    a : UPSWSservice!Property--(a.getname()='src'or
a.getname()='dst')
                    to
                    p:OUPSWS!Property(name<-a.getname(),type<-a.type)}
rule Class2Class {
                    from
                    a : UPSWSservice!Class
                    to
                     p:OUPSWS!Class(name<-
a.getDefNameSet(),generalization<-a.generalization),
                    on:OUPSWS!OntClass(base_Class<-p)}
rule property2datatypeproperty {
                    from
                    a : UPSWSservice!property
                    to
                    a1 : OUPSWS!Property(name<-
a.base_Property.name,type<-a.base_Property.type),
                     p2:OUPSWS!Association(name<-a.base_Property.name
                           --.asSequence().first().name
                    ,ownedEnd<-prop     ,ownedEnd<-a1
                           ,package <- a.base_Property.class.package
                           ),
                    prop:OUPSWS!Property(name<-'src',type<-
a.base_Property.class),
                           d:OUPSWS!DataTypeProperty(base_Association<-
p2)}

 helper context UPSWSservice!Property def : getname() : String =
      if self.name<> 'src' or self.name<> 'dst' then
      'dst' else self.name endif;
--rule Property2DataTypeProperty {
      --            from
      --            a : UPSWSservice!Property(a.getname()<>'src'or
a.getname()<>'dst' )
      --            to
      --             p:OUPSWS!Association(ownedEnd<-a)
      --            }
rule Association2Association {
                    from
                    a : UPSWSservice!Association
                    to
                     p:OUPSWS!Association(name<-
a.getDefNameSet(),ownedEnd<-a.ownedEnd,generalization<-
a.generalization),
                    on:OUPSWS!ObjectProperty(base_Association<-p)}

rule Generalization2Generalization {
                    from
                    a : UPSWSservice!Generalization
                    to
                    p:OUPSWS!Generalization(general<-a.general)}
rule DataType2Class {
                    from
                    a : UPSWSservice!DataType
                    to
                    p:OUPSWS!Class(name<-a.name),
                     on:OUPSWS!OntClass(base_Class<-p)}
```

```
-- \\\\\\\\\\\\\ Transformations OUPSWS VERS ODM \\\\\\\\\

module OUPSWS2ODM;
create OUT : ODM from IN : OUPSWS;

rule Package2Package {
                    from
                    a : OUPSWS!Package
                    to
                    p:ODM!Package(name<-a.name,profileApplication<-
a.profileApplication,packagedElement<-a.packagedElement,
              packageMerge<-a.packageMerge,
                    nestingPackage<-a.nestingPackage)}
rule Generalization2Generalization {
                    from
                    a : OUPSWS!Generalization
                    to
                    p:ODM!Generalization(general<-a.general)}
rule Class2Class {
                    from
                    a : OUPSWS!Class
                    to
                    p:ODM!Class(name<-'Class',ownedAttribute<-
pr,generalization<-a.generalization),
                    pr:ODM!Property(name<-a.name)}
helper context OUPSWS!Association def : getname() : String =
       OUPSWS!ObjectProperty.allInstances()->select(S | S.base_Association
= self)->iterate(S;acc : String=''|acc +'ObjectProperty')+
       OUPSWS!DataTypeProperty.allInstances()->select(S |
S.base_Association = self)->iterate(S;acc : String=''|acc
+'DataTypdeProperty');
rule Association2Class {
                  from
                   a : OUPSWS!Association
                  to p:ODM!Class(name<-a.getname(),ownedAttribute<-
pr,ownedAttribute<-e,ownedAttribute<-k,generalization<-a.generalization),
                    pr:ODM!Property(name<-a.name),
               e:ODM!Property(name<-'Domain',type<- a.ownedEnd-
>asSequence()->first().type),
                    k:ODM!Property(name<-'Range',type<- a.ownedEnd-
>asSequence()->last().type),
                    e1:ODM!Property(name<-'',type<- a.ownedEnd-
>asSequence()->first().type),
                    k1:ODM!Property(name<-'',type<- a.ownedEnd-
>asSequence()->last().type),
                    S:ODM!Property(name<-'',type<-p),
                    S1:ODM!Property(name<-'',type<-p),
                    A2:ODM!Association(name<-'Domain',ownedEnd<-
S,ownedEnd<-e1,package <- a.package),
                    A3:ODM!Association(name<-'Range',ownedEnd<-
S1,ownedEnd<-k1,package <- a.package)}
rule Property2Property {
                  from
                  a : OUPSWS!Property
                  to
                  p:ODM!Property(name<-a.name,type<-a.type)}
```

-- ///////// Transformations vers le code OWL-S ////////////////

```
query Atl = -- Query Template
-------------------Fonction principale-------------------
Process!Process.allInstances()->iterate(a;acc : String=''|acc
+a.GetFichierProcess())+
Grounding!Package.allInstances()->iterate(a;acc : String=''|acc
+a.GetFichierGrounding())+
Process!Activity.allInstances()->iterate(a;acc : String=''|acc
+a.GetFichierService())+
Profile!Package.allInstances()->iterate(a;acc : String=''|acc
+a.GetFichierProfile());

--  ////////// Partie qui génère le code de Service ///////////

helper context Process!Activity def: GetFichierService() : String =
Process!Activity.allInstances()->iterate(Prof;acc : String=''|acc +
'<service:Service rdf:ID="'+self.name+'Service">'+'\n'+
'<service:presents rdf:resource="'+Profile!Package.allInstances()-
>iterate(a;acc : String=''|acc
+a.name+'Profile#'+a.name+'Profile"/>'+'\n')+
'<service:describedBy rdf:resource="'+Process!Activity.allInstances()-
>iterate(ac;acc : String=''|acc
+ac.name+'Process.owl#'+ac.name+'"/>'+'\n')+
'<service:supports rdf:resource="'+Process!Activity.allInstances()-
>iterate(ac;acc : String=''|acc
+ac.name+'Grounding.owl#'+ac.name+'Grounding"/>'+'\n')+
'</service:Service>'+'\n').writeTo('c:/'+self.name+'Service.owl');

--  ////////// Partie qui génère le code de serviceProfile//////////

helper context Profile!Package def: GetFichierProfile() : String =
Profile!profile.allInstances()->iterate(Prof;acc : String=''|acc +'<?xml
version="1.0" encoding="ISO-8859-1" ?>'+'\n'+
                              '<rdf:
             ' <owl:Ontology rdf:about="">'+'\n'+
             ' </owl:Ontology>'+'\n'+
             '<profile:profile rdf:ID="'+self.name+'Profile">' +'\n'+
             '<service:presentedBy
rdf:resource="'+'c:/'+self.name+'Service.owl#'+self.name+'Service" />
'+'\n'+
             '<profile:has_process rdf:resource="'+'c:/'+
             Process!Activity.allInstances()->iterate(ac;acc :
String=''|acc +ac.name+'Process.owl#'+ac.name+'"/>'+'\n')+
              Profile!textDescription.allInstances()->iterate(a;acc :
String=''|acc +
              Profile!Property.allInstances()->select(Prop | Prop =
a.base_Property)->iterate(Prop;acc : String=''|acc +

'<profile:TextDescription>'+Prop.name+'</profile:TextDescription>' +'\n'))+
              Profile!serviceName.allInstances()->iterate(a;acc :
String=''|acc +
              Profile!Property.allInstances()->select(Prop | Prop =
a.base_Property)->iterate(Prop;acc : String=''|acc +
           '<profile:ServiceName>'+Prop.name+'</profile:ServiceName>'
+'\n'))+
              Profile!serviceProduct.allInstances()->iterate(a;acc :
```

```
'<profile:ServiceProduct>'+Prop.name+'</profile:ServiceProduct>' +'\n'))+
                Profile!ContactInformation.allInstances()->iterate(a;acc :
String=''|acc +
                Profile!Property.allInstances()->select(Prop | Prop =
a.base_Property)->iterate(Prop;acc : String=''|acc +

'<profile:ContactInformation>'+Prop.name+'</profile:ContactInformation>'
+'\n'))+
                Profile!serviceClassification.allInstances()->iterate(a;acc :
String=''|acc +
                Profile!Property.allInstances()->select(Prop | Prop =
a.base_Property)->iterate(Prop;acc : String=''|acc +

'<profile:ServiceClassification>'+Prop.name+'</profile:ServiceClassificatio
n>' +'\n'))+
                Profile!ServiceCategory.allInstances()->iterate(a;acc :
String=''|acc +
                Profile!Class.allInstances()->select(catclas | catclas =
a.base_Class)->iterate(catclas;acc : String=''|acc +
            '<profile:ServiceCategory>'+'\n'+
                '<addParam:NAICS rdf:ID="NAICS-category">'+'\n'+
                catclas.ownedAttribute->iterate(attCat;acc : String=''|acc +
                Profile!code.allInstances()->select(Propcode |
Propcode.base_Property = attCat)->
                iterate(Prop;acc : String=''|acc
+'<profile:code>'+attCat.name+'</profile:code>'+'\n')
                +
                Profile!value .allInstances()->select(Propcode |
Propcode.base_Property = attCat)->
                iterate(Propcode;acc : String=''|acc
+'<profile:value>'+attCat.name+'</profile:value>'+'\n')
                )+
                '</addParam:NAICS>'+'\n'+
                '</profile:ServiceCategory>' +'\n'))+
                Process!AtomicProcess.allInstances()->
iterate(ap;acc : String=''|acc
+ap.getParameterProfile()+'\n'+ap.getOutputProfile()+ap.getPreconditionProf
ile()+ap.getPostConditionProfile())

+'</profile:profile>'+'\n'+'</rdf:RDF>').writeTo('c:/'+self.name+'Profile.o
wl');

--  /////// Partie qui génère le code De serviceGrounding ////////

helper context Grounding!Package def: GetFichierGrounding() : String =
Grounding!I_nterface.allInstances()->iterate(Inerf;acc : String=''|acc +
   Grounding!Class.allInstances()->select(ClInerf | ClInerf =
Inerf.base_Class)->iterate(ClInerf;acc : String=''|acc +
     ClInerf.ownedAttribute->iterate(AttrInter;acc : String=''|acc +
        Grounding!O_peration.allInstances()->select(Op | Op.base_Property =
AttrInter)->iterate(Op;acc : String=''|acc +
          '<grounding:WsdlGrounding
rdf:ID="'+Op.AtomicProcess.name+'ServiceGrounding">'+'\n'+
          '<rdfs:comment>'+'\n'+
          '</rdfs:comment>'+'\n'+
          '<grounding:hasAtomicProcessGrounding
rdf:resource="#'+Op.AtomicProcess.name+'Grounding"/>'+'\n'+
          '</grounding:WsdlGrounding>'+'\n'+
```

```
'<grounding:WsdlAtomicProcessGrounding
rdf:ID="'+Op.AtomicProcess.name+'Grounding">'+'\n'+
                '<grounding:owlsProcess rdf:resource="'+
                Class!Activity.allInstances()->iterate(ac;acc :
String=''|acc +ac.name+'Process.owl#'+Op.AtomicProcess.name+'"/>'+'\n')+
              '<grounding:wsdlOperation>'+'\n'+
              '<grounding:WsdlOperationRef>'+'\n'+
              '<grounding:portType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">'+
                self.name+'Grounding.Wsdl#'+ ClInerf.name+'\n'+
                '</grounding:portType>'+'\n'+
                '<grounding:operation
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">'+
                self.name+'Grounding.Wsdl#'+ Op.base_Property.name+'\n'+
                '</grounding:operation>'+'\n'+
                '</grounding:WsdlOperationRef>'+'\n'+
                '</grounding:wsdlOperation>'+'\n'
              )+
              Grounding!Message_Input.allInstances()->select(In |
In.base_Property = AttrInter)->iterate(In;acc : String=''|acc +
              '<grounding:wsdlInputMessage
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">'+'\n'+
              self.name+'Grounding.Wsdl#'+ In.base_Property.type.name+'\n'+
              '</grounding:wsdlInputMessage>'+'\n'+
              In.base_Property.type.ownedAttribute->iterate(attrIn;acc :
String=''|acc +
              Class!Activity.allInstances()->iterate(ac;acc : String=''|acc
+
              '<grounding:wsdlInput>'+'\n'+
              '<grounding:WsdlInputMessageMap>'+'\n'+
              '<grounding:owlsParameter rdf:resource="'+
              Grounding!Input.allInstances()->select(input |
input.base_Property = attrIn)->iterate(input;acc : String=''|acc +
              ac.name+'Process.owl#'+input.Part_Input.name+'"/>'+'\n'+
              '<grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">'+
              self.name+'Grounding.wsdl#'+attrIn.name))+'\n'+
              '</grounding:wsdlMessagePart>'+'\n'+
              '</grounding:WsdlInputMessageMap>'+'\n'+
              '</grounding:wsdlInput>'+'\n'
              ))+
              Grounding!Message_Output.allInstances()->select(Out |
Out.base_Property = AttrInter)->iterate(Out;acc : String=''|acc +
              '<grounding:wsdlOutputMessage
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">'+'\n'+
              self.name+'Grounding.Wsdl#'+ Out.base_Property.type.name+'\n'+
              '</grounding:wsdlOutputMessage>'+'\n'+
              Out.base_Property.type.ownedAttribute->iterate(attrOut;acc :
String=''|acc +
              Class!Activity.allInstances()->iterate(ac;acc : String=''|acc
+
              '<grounding:wsdlOutput>'+'\n'+
              '<grounding:WsdlOutputMessageMap>'+'\n'+
              '<grounding:owlsParameter rdf:resource="'+
              Grounding!Output.allInstances()->select(output |
output.base_Property = attrOut)->iterate(output;acc : String=''|acc +
              ac.name+'Process.owl#'+attrOut.name+'"/>'+'\n'+
              '<grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">'+
              self.name+'Grounding.wsdl#'+attrOut.name))+'\n'+
```

```
'</grounding:wsdlOutput>'+'\n'
            )
            ----------------------------------------
            )
     )
   )+'<grounding:wsdlVersion
rdf:datatype="&xsd;#anyURI">http://www.w3.org/TR/2001/NOTE-wsdl-
20010315</grounding:wsdlVersion>'+'\n'+
  '<grounding:wsdlDocument
rdf:datatype="&xsd;#anyURI">&S_wsdl_grounding;</grounding:wsdlDocument>'+'\n'
   +'</grounding:WsdlAtomicProcessGrounding>'+'\n'+
   '</rdf:RDF>'
)).writeTo('c:/'+self.name+'Grounding.owl') ;

--  /////////// Partie qui génère le code de serviceModel ////////////

helper context Process!Process def: GetFichierProcess() : String =
Process!Process.allInstances()->iterate(a;acc : String=''|acc +

Process!Activity.allInstances()->select(ac | ac = a.base_Activity)->
iterate(ac;acc : String=''|acc +
if ac.node.size() > 1 then
   a.transformCompositeProcess()  else
      a.transformAtomicProcess()

endif+'</rdf:RDF>')).writeTo('c:/'+self.base_Activity.name+'Process.owl');

----------------------Générer AtomicProcess -----------------------------

helper context Process!Process def: transformAtomicProcess() : String =


Process!Activity.allInstances()->select(ac | ac = self.base_Activity)->
iterate(ac;acc : String=''|acc +'<!DOCTYPE uridef[]>'+'\n'+
            '<rdf:RDF>'+'\n'+   '<process:AtomicProcess
rdf:ID="'+ac.name+'">'+'\n'+
Process!AtomicProcess.allInstances()->
iterate(ap;acc : String=''|acc +
ap.getParameter()+ap.getOutput()+ap.getPrecondition()+ap.getPostCondition()
)
+      '</process:AtomicProcess>'+'\n'  );
helper context Process!Process def: transformCompositeProcess() : String =
Process!Activity.allInstances()->select(ac | ac = self.base_Activity)->
iterate(ac;acc : String=''|acc + '<process:CompositeProcess
rdf:ID="'+ac.name+'">'+'\n'+
'</process:CompositeProcess>'+'\n'

------------Générer les Inputs de la partie "Process" --------------

            );
helper context Process!AtomicProcess def: getParameter() : String =
Process!CallOperationAction.allInstances()->select(CalOp | CalOp =
self.base_CallOperationAction)->
iterate(CalOp;acc : String=''|acc +CalOp.argument->
iterate(INp;acc : String=''|acc +
Process!Input.allInstances()->select(inp | inp.base_InputPin = INp)->
iterate(inp;acc : String=''|acc +
'<process:hasInput>'+'\n'+
```

```
'<process:Input rdf:ID="'+INp.name+'">'+'\n'+'<process:parameterType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">'+'\n'+
Process!Activity.allInstances()->iterate(ac;acc : String=''|acc
+ac.name+'Process.owl#')+
inp.Concept.name+'</process:parameterType>'+'\n'+'</process:Input>'+'\n'+
'</process:hasInput>'+'\n')+

Process!Local.allInstances()->select(inp | inp.base_InputPin = INp)->
iterate(inp;acc : String=''|acc +
'<process:hasLocal>'+'\n'+
'<process:Local rdf:ID="'+INp.name+'">'+'\n'+'<process:parameterType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">'+'\n'+
Process!Activity.allInstances()->iterate(ac;acc : String=''|acc
+ac.name+'Process.owl#')+inp.Concept.name+
'</process:parameterType>'+'\n'+'</process:Local>'+'\n'+'</process:hasLoc
al>'+'\n'))

            );
------------Générer les Inputs de la partie "Profil" --------------

helper context Process!AtomicProcess def: getParameterProfile() : String
=
Process!CallOperationAction.allInstances()->select(CalOp | CalOp =
self.base_CallOperationAction)->
iterate(CalOp;acc : String=''|acc +CalOp.argument->
iterate(INp;acc : String=''|acc +
Process!Input.allInstances()->select(inp | inp.base_InputPin = INp)->
iterate(inp;acc : String=''|acc +
'<profile:hasInput rdf:resource="'+
Process!Activity.allInstances()->iterate(ac;acc : String=''|acc
+ac.name)+'#'+INp.name+'"/>'+'\n')
));
------------Générer les Outputs de la partie "Profil" --------------

helper context Process!AtomicProcess def: getOutputProfile() : String =
Process!CallOperationAction.allInstances()->select(CalOp | CalOp =
self.base_CallOperationAction)->
iterate(CalOp;acc : String=''|acc +CalOp.result->
iterate(INp;acc : String=''|acc +'<profile:hasOutput rdf:resource="'+
Process!Activity.allInstances()->iterate(ac;acc : String=''|acc
+ac.name)+'#'+
INp.name+'"/>'+'\n')

            );
------------Génerer les Outputs de la partie "Process" --------------

helper context Process!AtomicProcess def: getOutput() : String =
Process!CallOperationAction.allInstances()->select(CalOp | CalOp =
self.base_CallOperationAction)->
iterate(CalOp;acc : String=''|acc +CalOp.result->
iterate(INp;acc : String=''|acc +
Process!Output.allInstances()->select(inp | inp.base_OutputPin = INp)->
iterate(inp;acc : String=''|acc +'<process:hasOutput>'+'\n'+
'<process:Output rdf:ID="'+INp.name+'">'+'\n'+'<process:parameterType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">'+'\n'+
Process!Activity.allInstances()->iterate(ac;acc : String=''|acc
+ac.name+'Process.owl#')+
inp.Concept.name)+
'</process:parameterType>'+'\n'+'</process:Output>'+'\n'+'</process:hasOu
tput>'+'\n')
```

```
      );
---- Générer les Preconditions de la partie "Profil" ----

helper context Process!AtomicProcess def: getPreconditionProfile() :
String =
Process!CallOperationAction.allInstances()->select(CalOp | CalOp =
self.base_CallOperationAction)->
iterate(CalOp;acc : String=''|acc +CalOp.localPrecondition->
iterate(INp;acc : String=''|acc +'<profile:hasPrecondition>'+
Process!Activity.allInstances()->iterate(ac;acc : String=''|acc
+ac.name)+'#'+
INp.name+'</profile:hasPrecondition>'+'\n'));

-------- Générer les Preconditions de la partie "Process" ----------

helper context Process!AtomicProcess def: getPrecondition() : String =
Process!CallOperationAction.allInstances()->select(CalOp | CalOp =
self.base_CallOperationAction)->
iterate(CalOp;acc : String=''|acc +CalOp.localPrecondition->
iterate(INp;acc : String=''|acc +'<process:hasPrecondition>'+'\n'+
INp.name+'</process:hasPrecondition>'+'\n'));

----------Génerer les Postconditions de la partie "Profil" ----------

helper context Process!AtomicProcess def: getPostConditionProfile() :
String =
Process!CallOperationAction.allInstances()->select(CalOp | CalOp =
self.base_CallOperationAction)->
iterate(CalOp;acc : String=''|acc +CalOp.localPostcondition->
iterate(INp;acc : String=''|acc +
Process!Postcondition.allInstances()->select(pos | pos.base_Constraint =
INp)->
iterate(inp;acc : String=''|acc +
'<profile:hasResult>'+ Process!Activity.allInstances()->iterate(ac;acc :
String=''|acc +ac.name)
+'#'+INp.name+'</profile:hasResult>'+'\n'))   );

------- Génerer les Postconditions de la partie "Process" ----------

helper context Process!AtomicProcess def: getPostCondition() : String =
Process!CallOperationAction.allInstances()->select(CalOp | CalOp =
self.base_CallOperationAction)->
iterate(CalOp;acc : String=''|acc +CalOp.localPostcondition->
iterate(INp;acc : String=''|acc +
Process!Postcondition.allInstances()->select(pos | pos.base_Constraint =
INp)->
iterate(inp;acc : String=''|acc +
'<process:hasPostcondition>'+'\n'+
INp.name+'\n'+'</process:hasPostcondition>'+'\n')+
Process!Effect.allInstances()->select(pos | pos.base_Constraint = INp)->
iterate(inp;acc : String=''|acc +
'<process:hasEffect>'+'\n'+
INp.name+'\n'+'</process:hasEffect>'+'\n'))         );
```