N d'ordre : 10/2008-M/IN

République Algérienne Démocratique et Populaire
Ministère de L'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences Et de la Technologie Houari Boumédiène
Faculté D'Electronique et d'Informatique

MEMOIRE

présenté pour l'obtention du diplôme : **MAGISTER**

En : INFORMATIQUE

Spécialité : Systèmes Intelligents et Ingénierie du Logiciel

Par : M^r MITICHE HAKIM

Thème :

Modélisation des Organisations Virtuelles en e-Business. *Application au Grid*

Soutenu publiquement le 01/12/2008, devant le jury composé de :

Mr-H.AZZOUNE	Maitre de Conférence	USTHB	Président
Mme-H.DRIAS	Professeur	USTHB	Directrice de Mémoire
Mr-A.BELKHIR	Maitre de Conférence	USTHB	Examinateur
Mr-Y.AKLOUF	Chargé de Cours	USTHB	Invité

Résumé

Actuellement, l'*Organisation Virtuelle (VO)* émerge comme une nouvelle approche d'interaction dans les systèmes ouverts : e-Business, e-Government et e-Science. En e-Business, en particulier, cette approche révolutionne les modèles classiques, en dotant les partenaires commerciaux d'un moyen *dynamique*, *flexible* et *compétitif* pour établir et maintenir des relations de coopération. Une VO est une alliance temporaire d'actionnaires autonomes, dont le but est de résoudre un problème ou d'exploiter un marché, par échange de services et/ou partage de ressources. Ce concept se trouve à l'intersection des tendances Web actuelles, à savoir, les *services Web*, les *services Grid* et les *systèmes Multi-Agent*. Tandis que pas mal de travaux focalisent sur les mécanismes et outils supportant une VO ; comparativement, peu de travaux s'adressent à la modélisation VO. Or, un modèle est un moyen d'abstraire la complexité des VOs et demeure nécessaire pour faciliter la mise au point et l'adoption des VOs. Partant de ce constat, nous proposons un modèle VO de référence, qui sert de fondement architectural à diverses applications e-Business. Dans ce modèle, nous concevons des rôles usuels dans les VOs, ainsi que les interactions, les structures organisationnelles et le cycle de vie d'une VO. Nous introduisons l'*institution électronique* pour le support et la régulation des VOs. Nous appliquons notre modèle dans *MSVO*, un prototype pour la provision de services multimédias aux appareils mobiles, afin d'illustrer et d'évaluer ce modèle. L'architecture obtenue, *MSVO*, est à base d'agents, générique et réutilisable dans plusieurs scénarios e-Business.

Mots-Clés : Organisation Virtuelle, Service Grid/Web, Institution Électronique, Système Multi-Agent, e-Business.

Remerciements

Tout d'abord, je tiens à remercier mon Dieu tout puissant de m'avoir donné la patience et le courage d'achever ce travail.

J'exprime ma reconnaissance et mes remercîments à ma directrice de mémoire Pr Habiba Drias, de m'avoir fait confiance en me proposant ce sujet. Je la remercie pour ses lectures attentives et pour ses critiques et suggestions qui ont été d'un grand apport pour la finalité de ce travail. Je remercie également Mr.Youcef Aklouf, pour son support. Je remercie Dr Belkhir Adbdelkader, Dr Azzoune Hamid et Dr Aklouf Youcef d'avoir accepté de juger ce travail.

Je voudrais également remercier tout ceux qui mon encouragé : mes parents, mes collègues et tout mes amis.

Un grand Merci à Remache Abderahman et Nadil Malik pour leurs conseils et encouragements.

Je dédie ce travail à tous ceux qui me sont chers...

Mes Parents et mes proches, ma mère, en particulier ;

Mes Frères et Soeurs ;

A tous mes amis.

Hakim, 20 décembre 2008

Table des matières

Table des figures

Liste des tableaux

Introduction Générale

Tout au long des dernières décennies, l'épanouissement de l'informatique a bouleversé notre vie quotidienne dans ses aspects : études, travail et loisirs. Cela n'a été possible que grâce aux innovations qui changent nos façons de vivre et nous offrent des facilités incontournables. L'Internet est une innovation majeure qui permet de relier des humains, à travers le Web, et d'interconnecter des systèmes informatiques. De nos jours, de plus en plus d'applications sont développées pour opérer dans des environnements ouverts. De tels systèmes sont peuplés par des composants autonomes, hétérogènes et développés par des actionnaires indépendants dont les objectifs peuvent diverger. Cependant, dans la plupart des cas, ces entités doivent, inévitablement, coopérer et interagir d'une façon flexible, afin de parvenir à leurs fins. En particulier, une forme intéressante d'une telle interaction, est quand un ensemble d'entités, initialement distinctes, se réunissent pour former une alliance temporaire (ou *Organisation Virtuelle*), afin d'atteindre un certain objectif dont il serait difficile, voire impossible, individuellement.

Organisation Virtuelle

Actuellement, le concept *Organisation Virtuelle (VO)* connaît une émergence remarquable et suscite l'intérêt de diverses communautés de recherche (*ex.*, architectures orientées service, systèmes multi-agent et Grid). Une *Organisation Virtuelle* est une coalition temporaire d'actionnaires autonomes, qui définit un contexte de coopération par échange de services et/ou partage de ressources. Les applications des VOs s'étalent sur des domaines variés, dont le e-Science (*ex.*, partage des ressources de calcul/stockage) et le e-Commerce B2B (*ex.*, services 'à la demande'). Le concept VO prend de l'importance par le fait que beaucoup de systèmes distribués ouverts sont développés et le fait que, dans de tels cas, les VOs fournissent des moyens à des entités inter-connectées de se grouper ensemble pour fournir des services ou résoudre des problèmes, dont aucune ne peut assumer toute seule. A titre d'exemple, considérons la situation où un ensemble de fournisseurs de média (*ex.*, journaux, films et musique), des opérateurs de mobiles, et des constructeurs de téléphones se réunissent pour former une VO, afin de fournir des services multimédias avancés, à des usagers de téléphones portables de *quatrième génération (4G)* [25].

En e-Business, la compétitivité de l'environnement et le changement rapide des besoins, poussent les compagnies à se focaliser sur leur métier mère et sous-traiter le reste. Néanmoins, ces compagnies doivent maintenir un degré de flexibilité et d'adaptabilité élevé, afin d'investir dans des collaborations profitables, lorsque l'opportunité de marché se présente. En effet, cela permet d'augmenter la compétitivité, la profitabilité et maintient la survie de l'organisation. Par ailleurs, la provision de services personnalisés et à valeur ajoutée, à des utilisateurs de plus en plus exigeants, est devenue un facteur décisif aux "avantages compétitifs" d'un fournisseur. D'où cette provision est assez complexe et nécessite la combinaison des connaissances et efforts des organisations concernées.

Les Organisations Virtuelles suscitent une activité académique et industrielle importante. On cite parmi les défis : l'automatisation du cycle de vie et la mise au point de techniques relatives aux VOs (*ex.*, formation de coalitions, partage de ressources et le monitoring des services). Cependant, la diversité des scénarios, la partialité des architectures— par rapport aux contraintes VO considerées— et le manque de normalisation, surgissent le besoin d'un *modèle référence*, à savoir un *framework*. Ces derniers sont primordials pour aboutir à des systèmes effectifs et, par conséquent, à une large adoption. Dans le cadre de notre Magistère, nous abordons la problématique : *modélisation des organisations virtuelles*.

Modélisation des Organisations Virtuelles : Motivation

Le passage de l'approche organisation statique à interactions rigides, vers des organisations virtuelles flexibles à interactions dynamiques, est contraint par deux défis majeurs :

1. *Consensus sur un Modèle VO de Référence :* le concept VO est partagé entre les communautés Grid, Architecture Orientée Service (SOA) et Systèmes Multi-Agent (SMA). Chacune ayant affaire à un aspect particulier. Pendant que les services Grid offrent des techniques de découverte, d'allocation et d'ordonnancement des ressources ; les SOA tentent de standardiser ces techniques pour aboutir à des systèmes interoperables et faiblement couplés. Quant aux SMA, ils ont affaire à l'organisation et l'automatisation du système. Bien que l'objectif poursuivi soit commun, l'alignement de ces approches, par un modèle, reste un challenge décisif à relever.
2. *Mise à disposition d'une Framework de VO :* celle-ci doit faciliter la mise au point et le déploiement des VOs, par leurs actionnaires autonomes. La disponibilité d'une telle framework est décisive à l'adoption et à la banalisation des VOs, comme paradigme d'interaction dans les systèmes informatiques.

Peu de travaux se sont penchés sur ces problématiques. Principalement, nous citons le système CONOISE-G [29], qui propose un modèle d'Organisations Virtuelles basées agent dans un environnement Grid, ainsi qu'un prototype de démonstration. Néanmoins,

le modèle VO de CONOISE-G présente certaines lacunes. Notre travail permet de pallier à ces lacunes, de formaliser le modèle et de l'étendre à un large éventail d'applications.

Nous signifions par *modélisation des organisations virtuelles*, une formalisation et une conceptualisation structuelle et dynamique. Un modèle de VO identifie ses éléments de base, la structure qui regroupe ces élements et leurs schémas d'interaction, à un niveau d'abstraction élevé. Le but est d'offrir un fondement architectural à diverses applications VO et, donc, de faciliter leurs mise au point. En effet, *"L'Organisation Virtuelle fournit une manière d'abstraire la complexité des systèmes ouverts pour les rendre susceptibles à une implémentation"* [29]. Un modèle référence de VO est le premier pas vers l'adoption des VOs comme mode d'interaction inter-entreprises.

Cas d'Étude : Provision des Services Multimédia Mobiles

Afin d'illustrer et d'évaluer le modèle VO proposé, nous considérons un scénario typique au e-Business. Nous développons un système, intitulé *MSVO*, pour assister des usagers lors de séjours (touristiques ou d'affaires). Ce système propose aux clients des services multimédias *personnalisés*, *fiables* et *compétitifs* sur leurs appareils mobiles (*ex.*, Smart Phone, PDA ou Laptop). Par exemple, la "téléphonie", les "News" et d'autres "services du Web". Un tel système contient un ensemble de fournisseurs de *services* et de *ressources*, susceptibles de coopérer dans des coalitions temporaires, selon les opportunités du marché, afin de satisfaire les besoins clients. Dans ce contexte, nous distinguons des services/ressources *primaires*, comme la "bande passante sans fil", et d'autres *composés* ou *à valeur ajoutée*, comme le service "téléphonie mobile" ou le service "News mobile"—résultat de la combinaison du service "News" (contenu) et de la ressource "bande passante sans fil" (média de transport), par des fournisseurs, dans un contexte VO.

Contributions Majeures

Dans ce mémoire, nous proposons un modèle de VO basé rôle, et une architecture VO basée-agent pour un scénario e-Business. Notre contribution est comme suit :

1. *Un Modèle basé-Rôle d'Organisations Virtuelles supportées par une Institution Electronique dans un Environnement Grid :* Nous proposons un ensemble de schémas de rôle, d'interaction et de cycle de vie d'une VO. Ce modèle fournit le fondement architectural pour construire des VOs. Nous introduisons l'Institution Electronique pour supporter (*ex.*, découverte de services) et réglementer (*ex.*, monitoring et imposition de contrats) les interactions dans une VO.
2. *Architecture basée-Agent pour la Provision de Services Multimédias Mobiles par Construction et Exécution des Organisations Virtuelles :* Nous présentons une ar-

chitecture basée sur le modèle VO précédent, pour un scénario donné. Nous introduisons des services d'agents utiles et réutilisables dans divers scénarios e-Business.

Organisation du Document

Le reste du document est organisé comme suit :

Dans le Chapitre 1, nous présentons les Organisations Virtuelles. A travers l'étude de quelques systèmes de la littérature VO, nous identifions les contraintes et défis d'une telle approche. En outre, cette étude nous permettra de cerner les avantages/inconvénients des systèmes existants, afin de les prendre en considération lors de la modélisation VO.

Dans le Chapitre 2, nous faisons une synthèse sur les Architectures Orientées Service, en particulier, les *services Web* et les *services Grid*. Nous argumentons dans ce cas, que ces tendances offrent une infrastructure de base prometteuse aux VOs, qui sera, probablement, à l'avenir une pratique commune.

Dans le Chapitre 3, nous abordons les Institutions Electroniques (EI) où nous mettons en relief les approches d'implémentation existantes. Par ailleurs, nous mettons en évidence l'utilité apporté aux VOs par l'Institution Electronique.

Dans le Chapitre 4, nous proposons un modèle de VO à base de rôle. Nous formalisons une VO, ensuite, nous spécifions une VO en termes des rôles, interactions, structures organisationnelles et cycle de vie VO identifiés. Nous introduisons l'institution électronique comme moyen de support et de régulation des VOs. Ce modèle sert de base architecturale pour établir des VOs diverses en e-Business.

Dans le Chapitre 5, nous présentons un prototype (architecture) de VO, basé sur le modèle précédent, pour un scénario typique au e-Business. A travers ce cas d'étude, nous illustrons et évaluons notre modèle VO.

Finalement, nous concluons notre mémoire par une synthèse sur le travail accompli, une critique et nous donnons les directions de recherches futures.

Chapitre 1

Les Organisations Virtuelles

1.1 Introduction

Dans ce chapitre nous donnons une introduction et un état de l'art sur les Organisations Virtuelles. En particulier, nous focalisons sur les modèles, architectures et mécanismes de VO de type e-Business, pour faire surgir les défis actuels. La Section 1.2 introduit le concept VO et cite ses domaines et avantages. La Section 1.3 s'intéresse au cycle de vie VO et met en évidence des caractéristiques intrinsèques aux VO (*ex.,* le *dynamisme*, l'*ouverture* et la *compétitivité*). Les Sections 1.4, 1.5, et 1.6 étudient trois systèmes VO majeurs. La Section 1.7 termine par une conclusion.

1.2 Organisations Virtuelles

Les *Organisations Virtuelles (VOs)* deviennent un sujet de recherche de plus en plus important dans différents domaines de l'informatique. Un tel besoin se fait vite ressentir quand de plus en plus de systèmes ouverts sont développés : une situation où les VOs offrent un artifice pour assembler des entités inter-opérantes, afin de fournir des services dont aucune entité peut assumer toute seule. Historiquement, le terme VO ou *VE (Virtual Entreprise)* fut utilisé, en premier, en économie pour désigner toute alliance ou coalition d'entreprises. Par la suite, ce terme a été emprunté et divulgué dans beaucoup de disciplines de l'informatique comme le e-Business [35], le e-Government [2] ou le e-Learning [20], et plus récemment, dans le Grid [14].

Définition 1 *"Une VO est une collection d'individus et/ou d'institutions qui mettent en commun des ressources, tout en définissant la politique qui contrôle le partage, de façon à ce que les membres collaborent pour atteindre un objectif commun" [14].*

Définition 2 *Les VOs sont composées d'un ensemble d'individus, de départements ou d'organisations, dont chacun possède un ensemble d'habilités et des ressources à sa dis-*

position. Ces VOs sont formées pour mutualiser des ressources et combiner des services, dans le but d'exploiter un marché en émergence (market niche).

Ces définitions émanent des domaines Grid et e-Business, respectivement. Toutefois, elles partagent des notions comme la coopération, la distribution ou le dynamisme.

En particulier, un environnement e-Business est peuplé par des entités (individus et/ou organisations) autonomes (contrôlent leurs décisions), semi-indépendantes (nécessitent une coopération pour atteindre leurs objectifs) et égocentriques (cherchent à maximiser l'intérêt personnel). Chaque entité possède des habilités pour fournir des services ou des ressources, ainsi ces entités (fournisseurs) sont souvent en compétition dans des places de marché (*marketplaces*). Cependant, dans certains cas il s'avère intéressant pour un fournisseur de s'accorder avec un compétiteur, afin de fournir un service commun (*ie.*, former une *coalition*), ou bien avec un fournisseur dont les services sont complémentaires, afin d'offrir un nouveau type de service (*ie.*, former une *collaboration*)[24]. Ce processus de mise en commun des ressources, une fois constaté nécessaire ou profitable, passe à travers la formation d'une VO dans le but d'exploiter les opportunités qui se présentent. Un environnement e-Business est compétitif et connaît, constamment, des changements rapides, obligeant, ainsi, les organisations à se concentrer sur leurs compétences de base. Néanmoins, ces organisations maintiennent un degré de flexibilité élevé en demeurant ouvertes aux opportunités de coopération, afin d'augmenter leur compétitivité [11]. A titre d'exemple, on cite les VOs suivantes :

- Un ensemble de fournisseurs de média (*ex.*, actualité, films et musique), d'opérateurs de téléphone cellulaire, et des fabricants d'appareils mobiles qui s'assemblent pour former une VO, afin de fournir des services multimédia avancés et personnalisés, destinés à des utilisateurs de téléphonie mobile de quatrième génération (4G) [26].
- Deux compagnies aériennes, relativement petites et dont les routes sont complémentaires, se mettent en coalition et coordonnent leurs services pour offrir de nouveaux vols, entre destinations d'un large éventail, afin de devenir plus compétitives dans le marché [14].

Nous citons, dans ce qui suit, d'autres caractéristiques et avantages de l'approche Organisation Virtuelle par rapport à l'approche Organisation Classique :

- Fournir des services personnalisés aux utilisateurs et des produits à valeur ajoutée.
- Flexibilité structurelle. Permet à une VO de s'adapter à la dynamique de l'environnement, comme la mauvaise performance (ou le départ) d'un partenaire, le changement des besoins clients ou l'émergence d'opportunités d'affaire plus profitables.
- Opérer comme un seul bloque cohésif et avoir une seule identité face à la fourniture d'un service composé, tout en gardant l'identité individuelle des membres pour d'éventuelles transactions hors VO [26].
- Un cycle de vie pour automatiser et dynamiser l'exécution d'une VO.

1.3 Cycle de Vie d'une VO

L'Organisation Virtuelle, par opposé aux coopérations inter-entreprises traditionnelles, est une alliance *temporaire* faite 'à la demande' — *ex.,* lors d'une opportunité de marché, lors du besoin d'un service en dehors des habilités, ou par simple *outsourcing*(sous-traitement) pour une meilleure compétitivité. Cette alliance est maintenue aussi longtemps que ses membres jugent cela profitable (*ie.*, l'initiateur de la VO et ses partenaires). Cependant, dans des scénarios réels, certains partenaires sont sujet d'une mauvaise performance, d'un non respect de contrat ou d'un départ. D'autre part, les utilisateurs peuvent altérer leurs besoins ou résilier leurs contrats. De même, l'apparition de fournisseurs plus compétitifs est envisageable. Tous ces facteurs déterminent la *trajectoire* d'une VO ou ce qu'on dénote par *Cycle de Vie VO* (Figure 1.1).

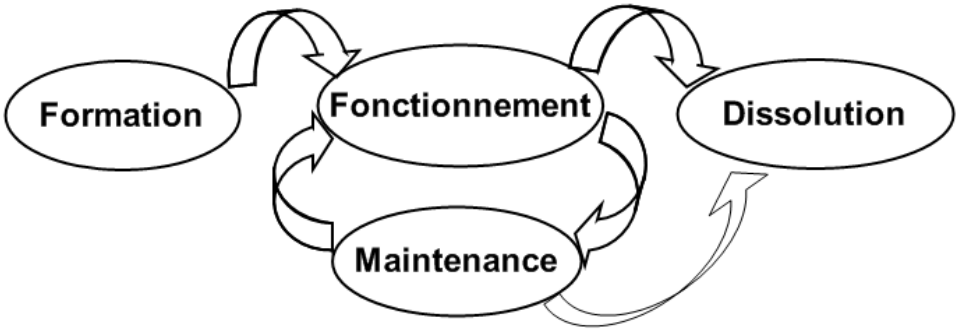

FIG. 1.1 – Cycle de Vie d'une Organisation Virtuelle

Formation : C'est une phase initiale, où les partenaires sont sélectionnés étant donnés les besoins clients. Elle est sanctionnée par des contrats issus d'accords entre des partenaires VO. Entre autres, la recherche des services, la négociation et le choix de la meilleure coalition, sont faits pendant cette étape.

Fonctionnement : phase de déroulement des transactions pour l'échange des services, selon les contrats établis lors de la formation VO.

Maintenance : Consiste à adapter la VO aux changements de l'environnement (comme la demande), soit en redistribuant les tâches ou la charge de travail sur les partenaires (*ie.*, une re-configuration), ou bien en remplaçant certains membres qui ont quitté ou qui n'ont pas respecté leurs engagements (*ie.*, une re-formation).

Dissolution : Consiste à achever une VO par l'arrêt des services et la mise à terme des relations de coopération. Ceci est le cas lorsqu'une VO termine son contrat, ou lorsqu'elle n'est plus viable à cause d'une perturbation (*ex.*, la VO est incapable de remplacer certains partenaires sortants).

Les membres d'une VO sont souvent représentés par des *agents*. Afin d'illustrer un cycle de vie VO, on peut considérer, par exemple, le contexte e-Commerce suivant.

Effectivement, il s'agit d'échanger des services et de partager des ressources dans une *marketplace*. On identifie quatre types d'acteurs : le *requérant de services* (RA), le *fournisseur de services* (SP), le *gérant* (et le représentant) de la VO (VOM) et l'*utilisateur final*(User).

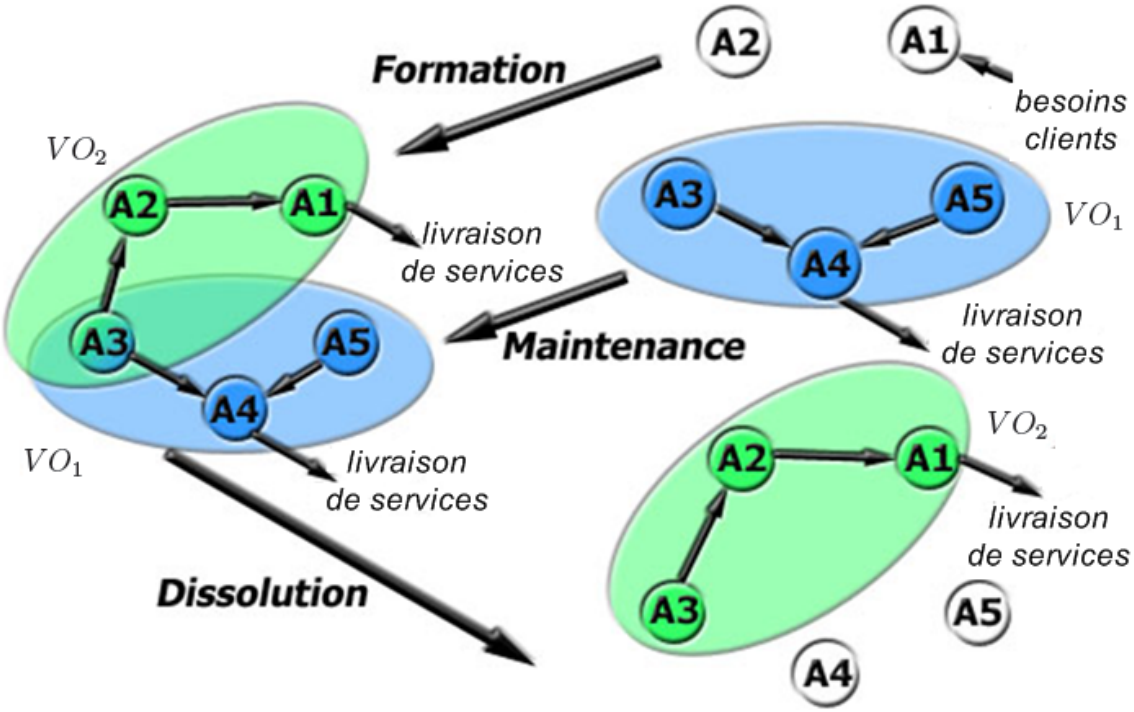

FIG. 1.2 – Dynamisme des Organisations Virtuelles

La Figure 1.2 schématise le cycle de vie VO. Initialement, l'environnement contient A_1, A_2 et la VO : $VO_1 = \{SP(A_3), SP(A_5), VOM(A_4)\}$, dont A_3 et A_5 fournissent les services de base, et A_4 gère et délivre le service final (à des clients ou à d'autres VOs). A l'issue des demandes clients, A_1 forme la VO, $VO_2 = \{VOM(A_1), SP(A_2), SP(A_3)\}$, pour parvenir à ses besoins. VO_2 est indépendante de VO_1, mais partage avec le membre A_3. Par la suite, VO_1 subit une perturbation, par exemple, lors du départ de A_3 —*ex.,* lorsqu'il n'est plus capable de fournir dans les deux VOs (en même temps) et préfère VO_2 (considérée plus profitable). En séquence, VO_1 tente de se re-configurer, de se re-former, ou se dissout, dans le cas extrême (Figure 1.2).

1.3.1 Formation de l'Organisation Virtuelle

Cette étape intiale doit considérer les questions suivantes [26] (Figure 1.3) :

- Un agent sur le point de soumettre une offre, pour joindre une VO, doit déterminer les conditions sous les quelles il est profitable de joindre.
- Un agent doit être capable de reconnaître les circonstances dans les quelles il doit initier la formation d'une VO.

- Un agent qui a initié le processus de formation d'une VO, doit déterminer, étant donnée un nombre d'offres, la meilleure combinaison de partenaires à retenir.
- Quelles sont les mécanismes qui permettent de choisir le meilleur fournisseur/service ? Par exemple, sur la base de la qualité de service ou de la réputation d'un fournisseur (vis-à-vis du respect des contrats).

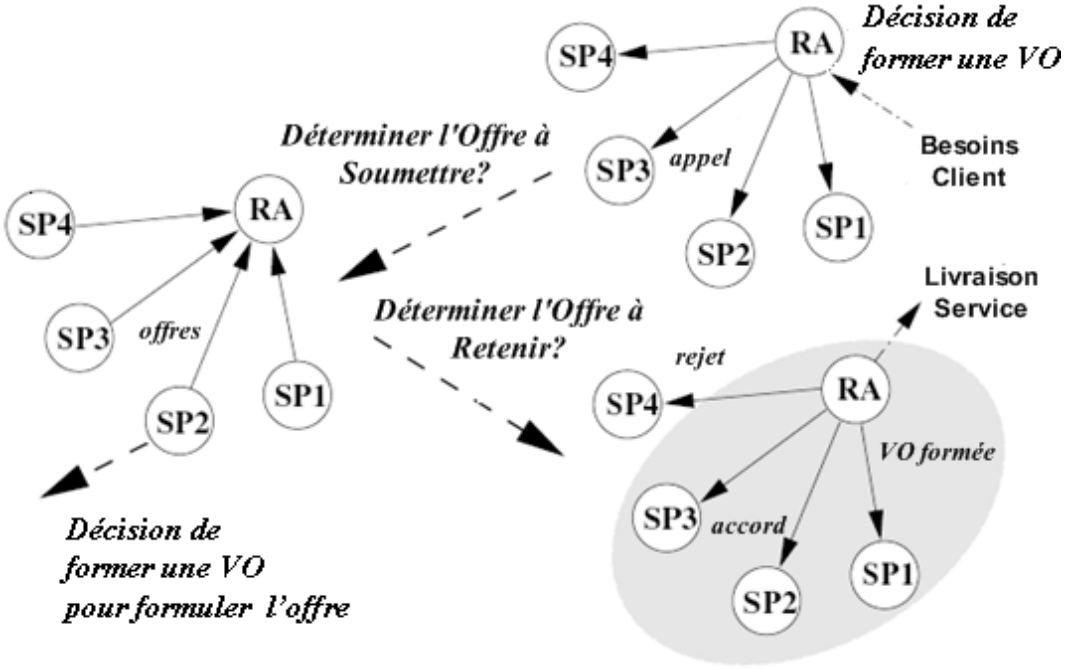

FIG. 1.3 – Formation d'une VO

Déterminer l'Offre à Soumettre ?

Avant qu'un *SP (Service Provider)* formule une offre, il doit être capable non seulement de savoir combien peut-il offrir pour chaque type de ressource demandée, mais aussi combien va-t-il réellement offrir ? De même, il doit être capable, une fois qu'il a identifié un déficit, de jouer le rôle de contractant, et par conséquent, lancer des appels d'offre pour former une VO. Lors d'un appel d'offre, un SP envisage l'une des actions suivantes[39] (Figure 1.4) :

1. Ignorer la requête en décidant de ne pas soumettre une offre.
2. Satisfaire la requête en utilisant ses propres ressources, uniquement.
3. Honorer la requête avec des ressources combinées à partir d'une VO dont il est le représentant (*ie.,* le VOM).
4. Tenter de former une nouvelle VO pour acquérir les ressources manquantes, afin de soumettre.
5. Annuler des engagements existants afin de satisfaire cette requête.

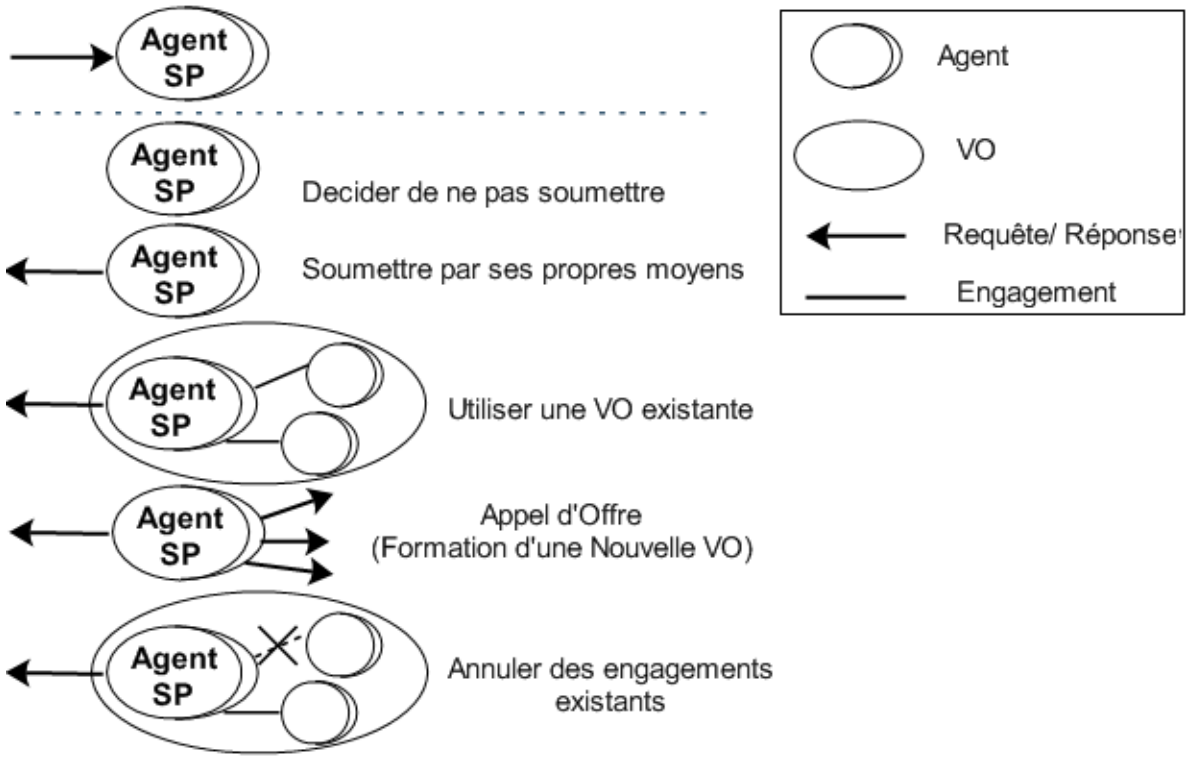

FIG. 1.4 – Prise de Décision lors de la Formation VO

Déterminer l'Offre à Retenir ?

Les VOs se forment dynamiquement et sont susceptibles de changer de structure. Une VO tend à améliorer sa compétitivité en termes de qualité de service et de temps de réponse. D'où, la sélection des partenaires est importante lors d'une formation ou d'une maintenance VO. Ainsi, les facteurs suivants sont à considérer [26] :

- L'ensemble des partenaires le plus **convenable**, celui avec le meilleur *prix* (dans le sens, estimation qui combine la valeur monétaire et d'autres attributs, comme le délai de livraison).
- Un temps **raisonnable** lors de la sélection des partenaires. Ceci qui est déterminant, en particulier, lors de la formation de plusieurs VOs (imbriquées ou parallèles) face à une requête donnée.
- Les partenaires peuvent varier leurs offres selon le degré d'implication dans une VO. Par exemple, un partenaire assez impliqué tend à offrir des services à bon prix (car les coûts intrinsèques sont amortis sur l'ensemble des instances).

1.3.2 Fonctionnement de l'Organisation Virtuelle

Une fois formée, une VO devient le contexte d'échange de services entre partenaires (SPs). Un client émet des requêtes au VOM pour consommer les services achetés. Quant au VOM, il doit coordonner et surveiller les SPs, afin de maintenir la cohérence et la viabilité de sa VO, satisfaisant, ainsi, l'objectif de sa formation.

1.3.3 Maintenance de l'Organisation Virtuelle

C'est une phase entreprise lors d'une perturbation de la VO, suite à des évènements dans l'environnement qui rendent la VO incapable de remplir sa mission. A titre d'exemple, les évènements suivants sont possibles [28] :

- Un SP qui ne remplit plus les obligations stipulées dans son contrat.
- Un problème de communication qui supprime quelques SPs de l'environnement.
- Le VOM décide de réorganiser la VO pour tirer profit des changements de l'environnement (*ex.*, un nouveau SP compétitif fait surface).
- Lors des changements de besoins clients ou la demande d'un nouveau type de service — sachant que la VO courante ne peut s'approprier cette situation.
- La VO a accompli tous ses engagements. Dans ce cas, un nouveau marché doit être décroché, ou bien la VO sera dissoute.

En conséquence, le VOM doit re-former, re-configurer ou achever sa VO. A l'instar de la phase Formation, le VOM cherche des SPs qui peuvent remplacer un service défaillant ou offrir le nouveau service demandé. Par contre, la re-configuration consiste à re-négocier la distribution des tâches ou la charge de travail dans la VO.

1.3.4 Dissolution de l'Organisation Virtuelle

C'est l'étape finale qui consiste à achever la VO, lorsque tous les engagements sont menés à terme, ou bien lors d'une perturbation irréversible. Certaines tâches comme la finalisation des contrats et le payement doivent être effectuées à ce stade.

Ayant présenté et mis en avant le concept VO, nous étudions dans les sections suivantes quelques architectures. L'objectif est de voir les motivations, cerner d'autres contraintes et défis de VO, et discuter les approches adoptées ainsi que les techniques mises au point, pour faire face aux challenges des VOs.

1.4 Système CONOISE-G

Le projet CONOISE-G (Grid-enabled Constraint-Oriented Negociation in an Open Information Services Environment) [1] [49] aspire à construire des VOs *flexibles*, *robustes* et 'à la demande', basées agent et qui opèrent dans un environnement ouvert et compétitif tel que le Grid. Pour cela, CONOISE-G s'adresse aux challenges principaux d'un cycle de vie VO, en fournissant des mécanismes de support, une architecture et un prototype pour un scénario de commerce électronique.

[1] fruit de la collaboration entre les universités Southampton, Aberdeen, Cardiff et British Telecom

1.4.1 Motivations

CONOISE-G part du scénario de motivation suivant [24]. On suppose, Lucy qui visite Londres en 2012 pour les Jeux Olympiques. En utilisant son PDA, elle accède à des services multimédias divers (news, clips des jeux, billets d'évènements, messagerie texte, divertissements) (Table 1.2). Le système CONOISE-G permettra à Lucy, étant donnés plusieurs SPs, de spécifier ses préférences et formera ensuite une VO qui va répondre, au mieux, à sa requête, en trouvant le compromis entre le meilleur prix, qualité de service et fiabilité.

SP	**Divertissement**	**News**	**Texte**	**Clips J.O**	**Billets**
SP1	30	20			5
SP2		10	50		
SP3			100	30	5
SP4	30	10		60	
SP5			50	45	10

TAB. 1.1 – Les Fournisseurs Potentiels de Services dans CONOISE-G

La Table 1.1 illustre cinq fournisseurs de services multimédia pertinents **(SP1, ...,SP5)**. Dans cet exemple, chaque fournisseur est caractérisé par les types et quantités de service qu'il peut fournir à un *moment* donné. Les services peuvent être demandés individuellement ou par paquet, avec la contrainte que les services de **SP2** doivent être pris ensemble. La formule de prix varie en fonction du nombre d'unités achetées, par exemple, **SP1** tarife 20 m-à-j de *News* par jour au prix de 20 £ par mois, et 10 m-à-j par jour au prix de 30 £ par mois. De même, la qualité de service peut varier, par exemple, **SP4** offre les clips des jeux à un taux minimum de 24 frame/secondes — mais réellement ce taux descend plus bas que ce niveau. Finalement, ces fournisseurs ne sont pas toujours crédibles, et ce qu'ils annoncent ne correspond pas toujours à ce que le client va recevoir. Par exemple, **SP5** peut annoncer des billets très demandés qu'il ne possède pas. Si Lucy veut acquérir le paquet de service de la Table 1.2, il est clair qu'il existe divers choix de composition, avec différents prix, qualités et niveaux de confiance.

Service Demandé	**Unités Demandées**
Divertissement	50 minutes par mois
News	10 m-à-j par jour
Messagerie Texte	100 par mois
Clips J.O	60 minutes par jour
Billets	10 alertes par jour

TAB. 1.2 – Exemple d'une requête dans CONOISE-G

1.4.2 Architecture CONOISE-G

L'architecture CONOISE-G (Figure 1.5) est composée d'*agents fournisseurs de services*(SPs) et d'*agents systèmes*(SAs). Les premiers sont invités à fournir les services consommables dans la VO ; tandis que les seconds effectuent des tâches de base lors de la formation et du fonctionnement de la VO, sous forme de *services de l'infrastructure* [28] :

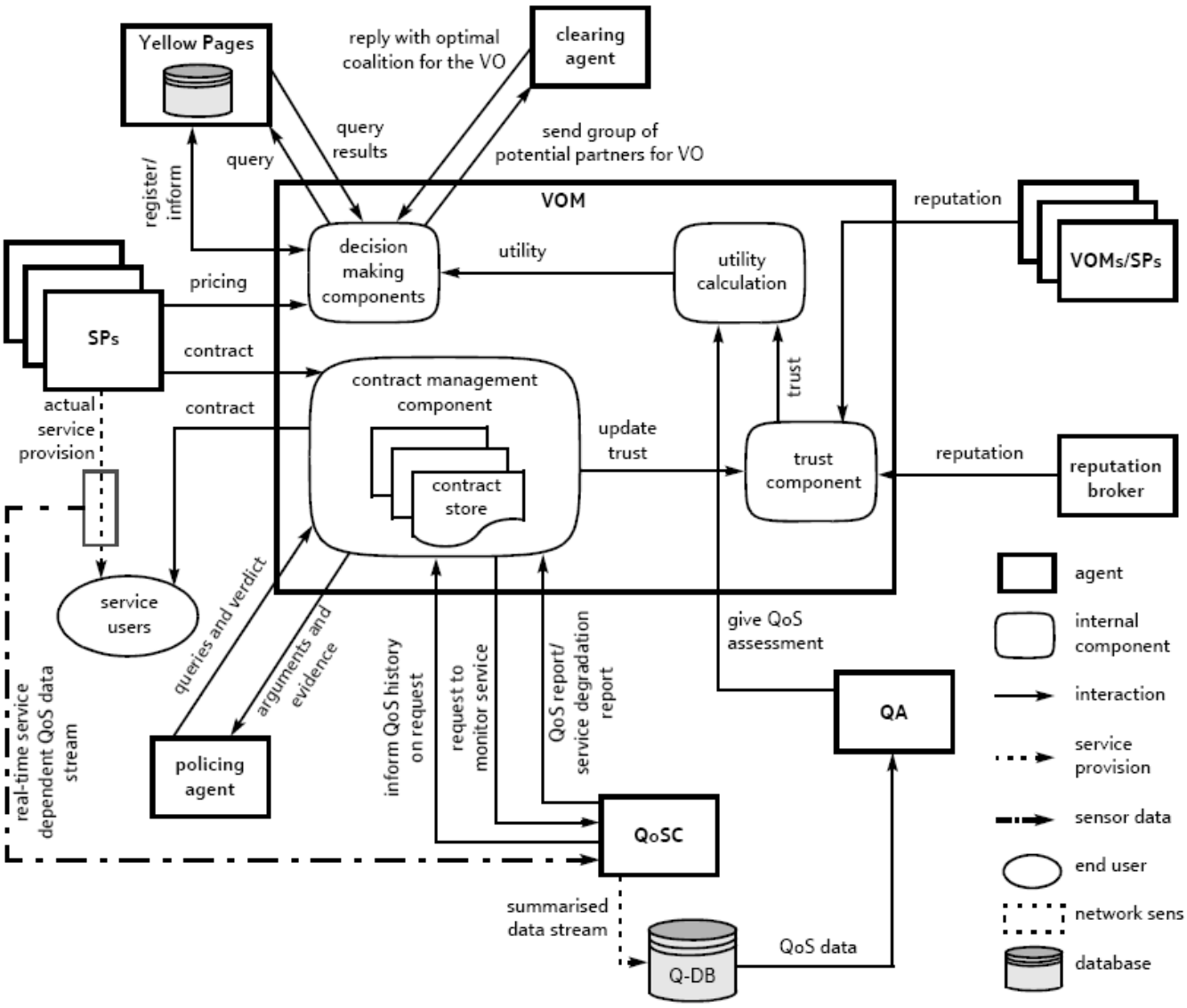

FIG. 1.5 – Architecture CONOISE-G

Yellow Pages (YP) conserve les détails de contrats ainsi que des informations sur les services publiés par des SPs.

Clearing Agent (CA) détermine l'ensemble des offres satisfaisants, *au mieux*, les besoins d'un agent requérant (RA). Pour cela, l'agent CA se base sur la requête agent et sur certains paramètres d'offre (*ex.*, le prix).

Quality Advisor (QA) capable d'estimer la QoS des agents à partir d'un historique.

Qos Consultant (QoSC) abstrait des capteurs dans le système et utilise les *patterns* "publish-subscribe" pour monitorer les services. Un agent intéressé spécifie au QoSC les services, leurs propriétés et les types d'évènements à monitorer et notifier.

Trust System modélise les degrés de confiance entre agents du système. C'est un composant présent au niveau de chaque agent fournisseur (SP).

Reputation Broker (RP) maintient les réputations des agents SPs vis-à-vis la livraison des services. Le RP complète le composant "Trust" d'un agent SP.

Agent Policing (PA) intervient lors d'un conflit entre agents, comme la rupture d'un contrat. Il initie des investigations, rassemble des preuves, et détermine si une violation de contrat a eu lieu. Ainsi, le PA fait des mises à jour du système Trust.

En supposant que les *SPs* ont déjà publié leurs services aux YPs, la formation de la VO est déclenchée par un certain agent *requérant de service* RA (Requester Agent)— agissant au compte d'un SP ou d'un simple client—, qui analyse les besoins, localise les fournisseurs potentiels (SPs) et lance les appels d'offre. La qualité de service (QoS) et la crédibilité (Trust) des offres reçues sont évaluées, respectivement, par l'*Agent Qualité* (QA) et le composant *Trust* de l'agent RA. Le résultat est combiné avec la formule de prix, par l'*Agent Clearing* (CA) afin de déterminer la combinaison des services/fournisseurs qui forme une VO optimale (en termes de prix, qualité et crédibilité). A ce stade la VO est formée, et l'agent RA prend le rôle de VOM, celui responsable d'assurer que chaque membre fournit ses services selon son contrat.

Pendant la phase opérationnelle d'une VO, le VOM peut demander au QoSC de monitorer les services de chaque membre VO. Le QoSC avertit le VOM à chaque dégradation de la QoS (disant le service *News* dans le scénario de la séction précédante) au dessous d'un taux spécifié dans le contrat (dit SLA, *Service Level Agreement*). De même, n'importe quel membre peut invoquer le *Policing Agent* (PA) pour investir les conflits qui surgissent lors de la provision des services. Lors d'une dégradation de QoS ou d'une violation de contrat, ces informations sont injectées au composant Trust, pour pénaliser le fournisseur responsable et mettre à jour sa crédibilité. Dans ce cas, le VOM re-forme la VO : il lance de nouvelles requêtes aux YPs à la recherche d'autres SPs pouvant offrir les services défaillants. Les offres sont évaluées aussitôt reçues, et le CA détermine le meilleur SP pour remplacer le SP sortant. Par la suite, le VOM ordonne au QoSC de stopper le monitoring de l'ancien SP et de le faire pour le nouveau.

1.4.3 Discussion

Le modèle CONOISE-G [29] considère la majorité des contraintes d'un système ouvert, pour permettre des relations de coopération inter-entreprises de type VO. Néanmoins, il présente certains inconvénients, comme la forte-granularité de agent SP— qui doit être

contractant, contracté, fournisseur, consommateur de services et gestionnaire de VO, à la fois. Par ailleurs, il n'est pas commode d'attribuer l'établissement et le monitoring de contrat à l'agent VOM d'un fournisseur impliqué dans un marché (cela doit être fait par une partie tierce). Enfin, au niveau implémentation, ce système est en phase prototype de démonstration et le projet en question s'est plus penché sur des mécanismes, *séparément*, plutôt que sur l'architecture VO qui intègre le tout. Toutefois, CONOISE-G demeure – à notre connaissance – le seul système ayant proposé un nouveau modèle d'interaction B2B de type VO.

1.5 Système KRAFT

Le système KRAFT (Knowledge Reuse And Fusion/Transformation) [32] opte pour une approche orientée agent et basée "médiateur" pour pourvoir une *infrastructure* support aux VOs. Le commerce électronique B2B est l'une des applications ciblées. Les agents *médiateurs* de l'infrastructure KRAFT sont engagés pour échanger des informations riches (dans un format basée contrainte) et fusionner des données, de manière coordonnée avec des sources d'informations (fournisseurs) et des clients, au sein de VOs. Ce système fait appel aux technologies "agents BDI" et "satisfaction de contraintes", utilise un modèle de donnée propre à lui et un langage de communication agent standard. KRAFT a fait l'objet d'une démonstration dans le domaine de la provision de services en Télécommunications.

1.5.1 Motivations

KRAFT fournit une infrastructure générique pour des applications de gestion des connaissances [33]. Parmi ces applications, le commerce B2B pour le support de la formation VO. Dans ce cas, il est prévu que des partenaires échangent des connaissances commerciales sous forme de contraintes. Un des scénarios de motivation – et de démonstration – est la conception des services réseaux en Télécommunication pour la British Telecom (BT).
En effet, il s'agit d'une configuration de réseau du point de vue client, ou le système KRAFT permettra à un "Network Designer" de BT de sélectionner les services et les équipements, éventuellement à partir de plusieurs vendeurs, pour satisfaire les besoins client. KRAFT suppose des partenaires avec des connaissances et modèles de données hétérogènes, et couvre d'autres domaines d'applications comme le planning des voyages.

1.5.2 Architecture KRAFT

L'architecture KRAFT est basée agent et supporte les étapes : identification des besoins et sélection des partenaires lors de la formation d'une VO. On cite ses composants principaux comme suit [33] (Figure 1.6) :

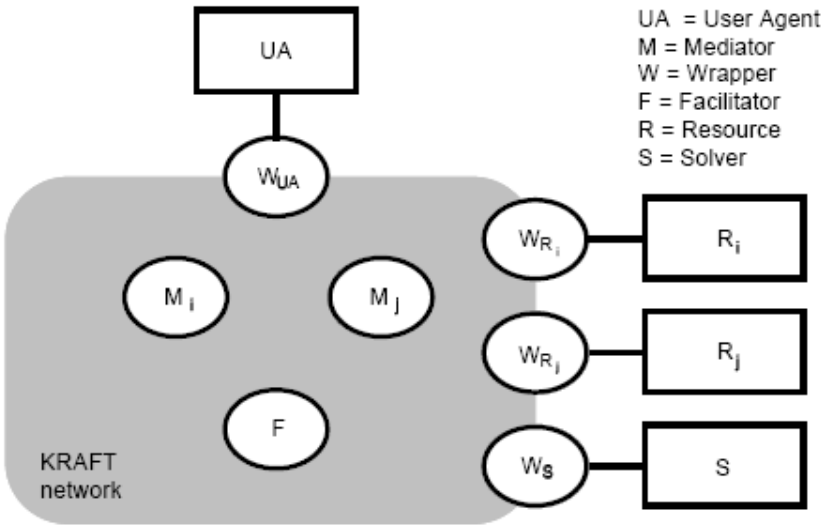

FIG. 1.6 – Architecture KRAFT

- *Facilitator agents* (Facilitateurs) : supportent la description (pages jaunes) et la localisation (*matchmaking*) des sources en ligne.
- *Wrapper agents* (Traducteurs) : jouent le rôle d'intermédiaire entre le système et les sources d'informations (vendeurs et client), afin de transformer les connaissances locales vers et à partir du format d'échange.
- *Mediator agents* (Médiateurs) : supportent le questionnement et la fusion des connaissances à partir des sources — ils fournissent des services à valeur ajoutée.
- *Legacy Solver Engines* : des solutionneurs des problèmes à base de contraintes. Ils sont interfacés par des agents *Wrappers* et exploitables par les agents *Mediators*.

Lorsqu'un agent UA soumet sa requête (Figure 1.6), elle est traduite par l'agent W_{UA} qui va consulter l'agent F, à la recherche d'un médiateur ou d'un traducteur capable d'y répondre. Par la suite, une requête peut être décomposée en sous-requêtes à transmettre à d'autres médiateurs/traducteurs M_i, M_j, W_R. Une fois arrivée à un traducteur W_R, la requête est posée à la source d'information correspondante, R, sous le format et le langage local. Les réponses en retour sont fusionnées par leurs médiateurs respectifs. Ce processus peut nécessiter l'appel d'un solutionneur, S, pour résoudre des contraintes système/utilisateur qui lient les informations récoltées. La solution finale, une fois trouvée, est remise à l'utilisateur sous son format local.

1.5.3 Discussion

Le modèle KRAFT traite un volet important à l'habilitation des VOs, en l'occurrence, la fusion de connaissances et la configuration de système sur la base de contraintes utilisateur, système et vendeur. De telle sort, l'architecture KRAFT permet une personnalisation des services et facilite la coopération entre partenaires. Néanmoins, l'approche proposée reste

partielle, le fait qu'elle a affaire à la sélection des partenaires (*ie.,* phase formation VO), seulement. De même, elle ne considère pas certains critères de VO (*ex.,* la compétitivité).

1.6 Système AGORA

AGORA [31] est une architecture multi-agent pour supporter la formation des *Virtual Entreprises* (VEs). Cette architecture focalise sur la sélection des partenaires en offrant un modèle de VE, un modèle d'agent ainsi qu'un protocole d'interaction agent (AIP). AGORA permet la formation des VEs dans des places de marché.

1.6.1 Motivations

AGORA s'intéresse à la sélection des partenaires lors de la formation d'une VE. Son but est d'assister le processus de décision d'un partenaire commercial et non pas de remplacer un acteur humain. En effet, étant donné que les VEs n'ont pas une structure organisationnelle rigide, et que la VE a besoin d'être formée rapidement pour atteindre son but, la sélection des partenaires devient l'une des plus importantes tâches à automatiser. Un scénario de motivation et d'illustration de AGORA est donné dans [31], où il s'agit de la formation d'une VE pour la conception et la création d'une Intranet.

1.6.2 Modèle AGORA

C'est un modèle qui identifie les entités (et leurs relations) impliquées dans la formation VE. Il identifi un *but* comme étant réalisable par un ensemble d'*activités* regroupées dans des *rôles*. Le choix d'un agent dépend des *compétences* requises par le rôle. Par exemple, lors de la mise au point d'une Intranet, on identifie deux sous-buts : "Conception Intranet" et "Création Intranet". Pour cela, deux rôles sont nécessaires : "Intranet Designer" et "Webpage Developper", dont les compétences nécessaires, pour ce dernier, sont : HTML et Java. En outre, le modèle AGORA définit trois rôles génériques, présents dans chaque scénario VE [30] : (1) Client ; (2) Initiateur VE — celui qui prend l'initiative de former la VO (peut être un client) ; et (3) Partenaires VE — ceux qui forment la VE (l'initiateur peut en être un). Au début un partenaire est dit *intéressé*, ensuite il devient *potentiel* s'il est retenu pour une négociation de contrat. Ensuite, le modèle AGORA définit un *scénario* de formation VE par les étapes [31] :

1. Accord entre le client et l'initiateur VE sur le travail à faire.
2. Identifier les buts et définir les besoins.
3. Définir les activités nécessaires pour atteindre les buts.
4. Identifier les besoins en rôles et compétences pour effectuer ces activités.
5. Trouver et sélectionner des partenaires pour accomplir ces rôles.

1.6.3 Architecture AGORA

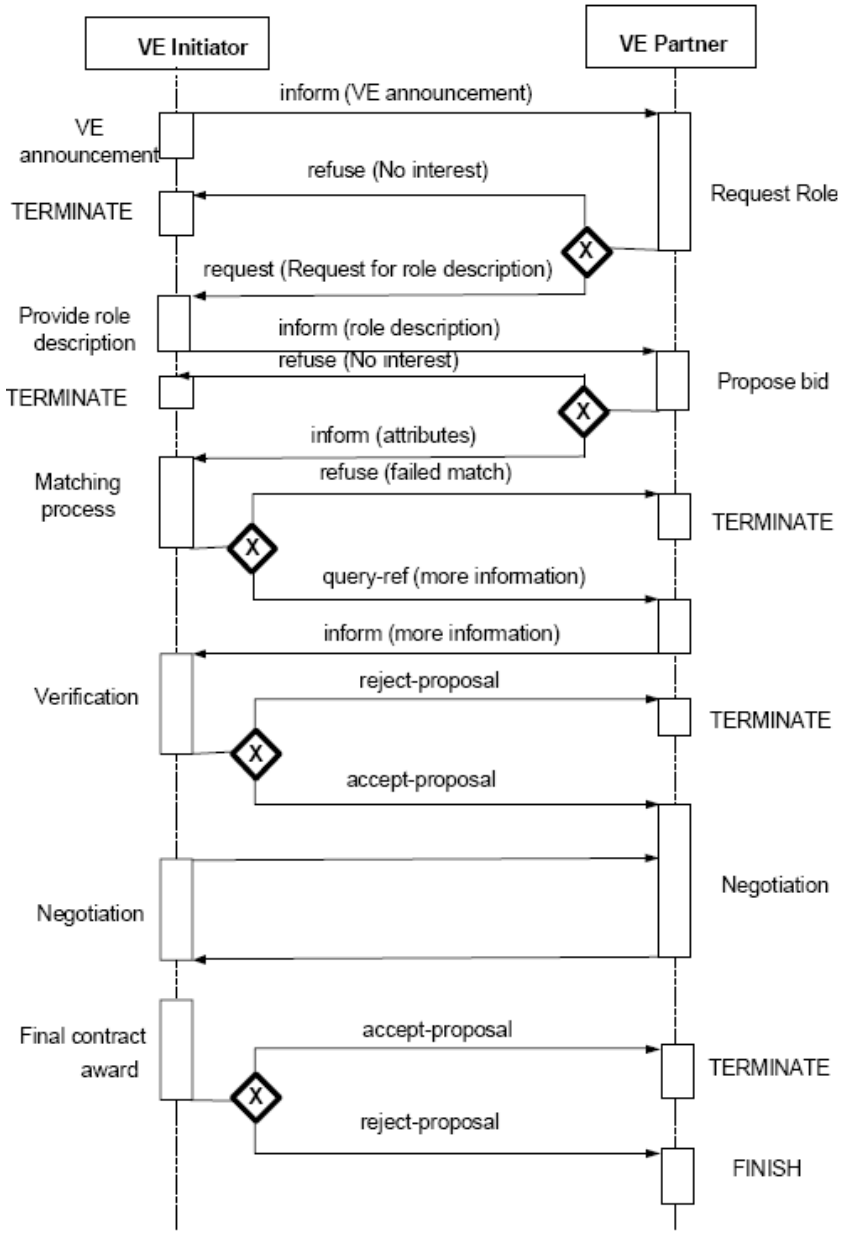

FIG. 1.7 – Protocole d'Interaction Agent pour la formation VE dans AGORA

C'est une infrastructure multi-agent pour supporter l'implémentation des VEs. Elle est composée de nœuds contenant des agents systèmes qui facilitent la communication, la coordination et la négociation entre agents. Par exemple, le *Agora Manager*, effectue une gestion globale et un "matchmaking" ; le *Coordinateur* assure un comportement d'agents cohérant dans un nœud ; et le *Negociator* résout les conflits via négociation. Le protocole d'interaction agent (AIP) pour la formation d'une VE est donné par la Figure 1.7 (exprimé en Agent UML [47]).

Le *VE Manager* commence par annoncer les buts de la VE, et donne la description des rôles aux partenaires. Les partenaires vont faire l'alignement de leurs buts avec ceux de la VE. Ceux intéressés soumettent leurs offres à un processus de sélection qui fait

correspondre les attributs de l'offre à ceux du but recherché—éventuellement, le VE Manager peut demander d'avantage de détails sur le rôle proposé par un partenaire intéressé. Le VE Manager vérifit la validité des informations reçues dans les offres, entreprend des négociations avec les partenaires potentiels, parmi lesquels il contactera les gagnants.

1.6.4 Discussion

AGORA [31] adopte une approche multi-agent basée sur un modèle Entreprise qui identifie et met en relation les entités d'une VE. AGORA définit un Protocole d'Interaction Agent (AIP) pour la formation d'une VE. Cependant, elle se limite à la phase de formation VO, seulement. Par ailleurs, le modèle de VE AGORA est modeste (simple conceputalisation sans schémas formels). De plus, ce modèle et ne spécifit pas comment automatiser la sélection des partenaires.

1.7 Conclusion

L'Organisation Virtuelle (VO) est un paradigme d'interaction prometteur, principalement, en e-Business. Dans ce chapitre, nous avons étudié ce concept et quelques architectures de la littérature VO. A travers cette étude, nous avons mis en relief les problèmes, avantages et lacunes de certains systèmes VO. Notre constat est celui d'architectures *partielles* (par rapport au cycle de vie couvert et aux contraintes VO considérées) et spécifiques à des scénarios. Par conséquant, la nécessité d'un modèle référence, voire une framework VO, s'avère nécesaire afin d'habiliter ce paradigme. L'organisation virtuelle offre un contexte dynamique et flexible, pour l'échange de *service* et pour le partage de *ressource*, entre organisations/individus distribués et autonomes. Bien que l'approche multi-agent pour le support des VOs soit répondue, la tendance actuelle des *services Web/Grid* pour la mise au point des systèmes interoperables, succite beaucoup d'intérêt et mérite d'être le sujet de notre prochain chapitre.

Chapitre 2

Les Architectures Orientées Service

2.1 Introduction

Avec l'avènement Web, les interactions orientées service connaissent un essor fulgurant et deviennent une pratique courante. En e-commerce inter-entreprises, en particulier, les *services Web/Grid* facilitent l'échange de services et le partage de ressources, au sein de systèmes faiblement couplés et inter-opérables. Dans ce chapitre, nous passons en revue l'état de l'art des *Architectures Orientées Service* (SOA), en situant les VOs dans ce cadre. La Section 2.2 introduit les SOA pour aborder les *services Web* dans la Section 2.3. La Section 2.4 introduit le Grid, les VOs selon la perspective Grid, et fait surgir les *services Grid* comme extension des services Web. La Section 2.5 conclut ce chapitre.

2.2 Architectures Orientées Service

On sous-entend par *Service Oriented Architecture* (SOA), un modèle de systèmes informatiques distribués basés sur le concept *service*. La tendance actuelle est celle des *services Web*, bien qu'au début c'était des architectures à composants, comme CORBA (de OMG), D/COM (de Microsoft) ou RMI (de Sun). La particularité des SOA est d'offrir des protocoles et standards qui permettent l'interopérabilité à travers Internet. Singh et Huhns[42] citent deux avantages de cette approche : *(i)* permettre de nouveaux types d'applications e-Business flexibles, dans des systèmes ouverts, qui ne peuvent être possibles autrement ; *(ii)* améliorer la productivité de la programmation et de l'administration des systèmes ouverts. Les SOA permettent de choisir les partenaires commerciaux sur la base des critères de la qualité-de-service (*ex.,* performance, disponibilité et crédibilité), ce qui est utile à la formation des VOs.

2.2.1 Caractéristiques des SOA

On identifi les éléments suivants que nous pouvons analyser à la lumière des VOs [42] :

- ***Faible Couplage*** : *pas de propriétés transactionnelles strictes qui s'appliquent sur les composants. L'accentuation est plutôt sur les relations contractuelles de haut-niveau.*

  C'est une propriété intrinsèque des VOs, car ses entités sont autonomes et ne partagent que les protocoles et connaissances nécessaires à une interaction. Le faible couplage maintient un système fonctionnel lors de l'échec d'un composant.
- ***Neutralité de l'Implémentation*** : *Seulement l'interface importe. On ne peut dépendre sur les détails d'implémentation des composants en interaction.*

  Ce qui est favorable aux VOs qui sont hétérogènes en termes de langages de développement et de plateformes de déploiement.
- ***Configuration Flexible*** : *Le système est configuré en dernier lieu de façon flexible.* Les VOs sont dynamiques (non pré-configurées d'avance) et changent de configuration pour s'adapter à l'environnement et maintenir leur viabilité.
- ***Temps de vie Long*** : *Les composants doivent exister aussi longtemps pour déceler des exceptions et prendre des mesures correctives. Les composants doivent exister aussi longtemps pour être découvertes, et engendrer une confiance sur leurs comportements.*

  Bien qu'une VO soit temporaire, ses entités doivent montrer une stabilité pour être découvrables et avoir une trace de la QoS et de la réputation.
- ***Granularité*** : *Les participants d'une SOA doivent être appréhendés à un niveau de granularité dense. Les interactions et dépendances doivent avoir lieu à un niveau aussi élevé que possible.* C'est le cas des VOs.
- ***Equipes*** : *Il est commode de réaliser les traitements par des parties autonomes. En d'autres termes, au lieu d'un participant qui commande ses partenaires, les traitements dans un système ouvert sont plus une question de partenaires commerciaux agissant comme une équipe.*

  La flexibilité des agents, à la base d'une VO, permet des modes de coordination divers incluant : la négociation (*ex.,* Contract-Net), l'élection d'un *team leader*, et le travail en mode *P2P* (Pair-à-Pair).

2.2.2 Aspects Principaux d'une SOA

Le cycle d'un échange de service est composé des étapes : *information, négociation* et *exécution* (Figure 2.1). La première est la découverte et la sélection de fournisseur/utilisateur à travers des services *registres*. La négociation consiste à établir des accords (contrats) entre fournisseurs/utilisateurs. L'étape exécution réalise les services.

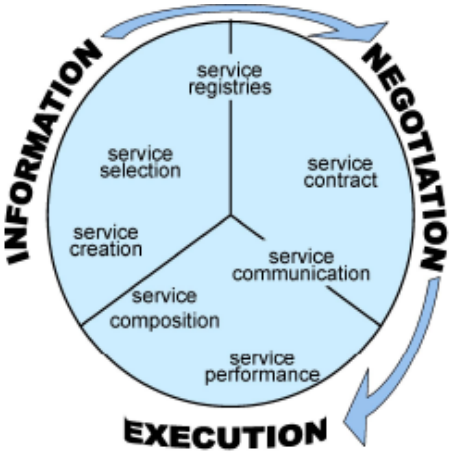

FIG. 2.1 – Cycle de vie d'un Service

2.3 Services Web

Beaucoup de définitions existent. Une consiste à définir un *service Web* comme : *"loosely coupled applications using open, cross-platform standards and which interoperate across organisation"*[44]. Nous pouvons caractériser un service Web par deux aspects importants :

1. *Programmatiquement accessibles* : Les services Web sont conçus pour être invoqués par d'autres services Web ou applications. Ils sont distribués dans le Web et accessibles à travers des protocoles déployés, tels que HTTP et SMTP. Les services Web doivent décrire leurs habilités à d'autres services en termes d'opérations, de messages d'entrés et de sorties, et la façon par laquelle ils peuvent être invoqués.
2. *Faiblement couplés* : la communication entre services Web est *basée-document*. Généralement, les services Web utilisent des documents XML pour échanger des informations les uns avec les autres. C'est cette communication, basée-document, qui permet des relations faiblement couplées entre services Web.

2.3.1 Le Modèle Référence des Services Web

Les interactions entre services Web impliquent trois types de participants : *fournisseurs de service, registre de service, et consommateurs de service* (Figure 2.2).

Les *fournisseurs de service* décrivent et publient leurs services dans des *registres de service*, un répertoire parcourable des descriptions de service. Chaque description contient des détails sur le service, comme les types de données et la localisation réseau. les *utilisateurs de service* utilisent l'opération *find* pour localiser les services d'interêt. Le registre retourne les descriptions des services pertinents. Trois initiatives de standarisation majeurs existent au niveau du W3C [57] : WSDL (*Web Service Description Language*) [58], pour la description des caractéristiques fonctionnelles des services Web ; UDDI (*Universal Description, Discovery and Integration*) [59], qui définit des interfaces pour la publication

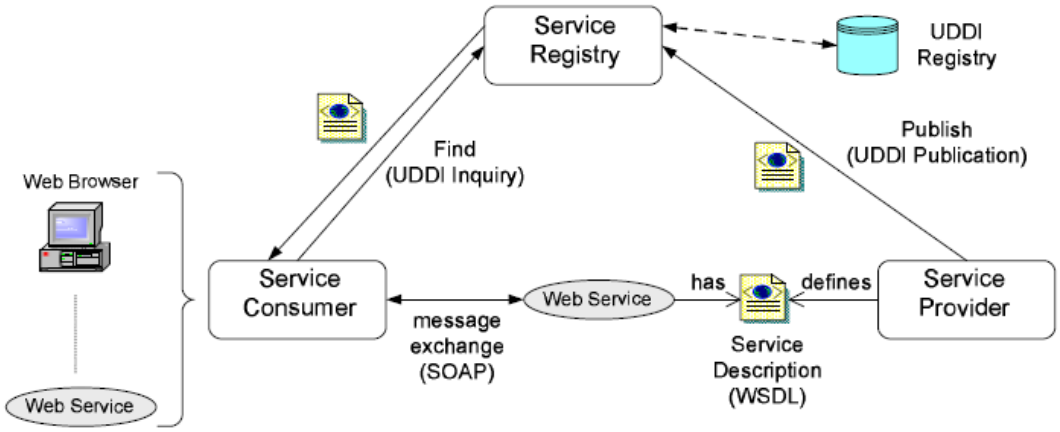

FIG. 2.2 – Le Modèle Référence des services Web

et la découverte des services Web ; et SOAP (*Simple Object Access Protocol*)[60], qui est une framework de messagerie légère pour l'échange de données formatées en XML, entre services Web.

2.3.2 Les Intéractions dans les services Web

Les services Web permettent des interactions au niveau de la *couche communication* à travers le protocole SOAP. L'adoption d'une messagerie basée XML, à travers des protocoles bien établis (*HTTP et FTP*), permet des communications entre systèmes hétérogènes. En effet, la plupart des environnements actuels sont capables de communiquer en HTTP et *parser* (analyser) des documents XML. Cependant, le protocole SOAP demeure toujours en phase de maturité ; les implémentations courantes ne correspondent pas aux contraintes de fiabilité et de charge de travail, exigées dans les entreprises.

Au niveau de la *couche contenu*, les services Web utilisent un langage (WSDL) qui ne supporte pas la description sémantique. Par exemple, il n'y pas de moyen pour décrire les types dans un document (*ex.,* est-ce qu'une opération est une demande de devis ou d'achat). Néanmoins, des efforts récents ont affaire à l'interopérabilité sémantique par le développement d'un langage de marquage de contenu (content markup) comme DAML-S [22], bien que ca reste dans un stage préliminaire.

Les services Web manquent de support aux interactions de la *couche business*. Jusqu'a présent, l'habilitation d'interactions entre services Web a été, largement, un processus *ad hoc* qui implique une programmation répétitive de niveau bas [23]. Les efforts de standardisation, comme BPEL4WS (*Business Process Execution Language For Web Services*) [48], sont en cours pour permettre la définition des *processus métiers* (business process) à travers la composition de services Web.

Les services Web permettent des systèmes distribués faiblements couplés où les composants sont découvrables et invocables, dynamiquement. Les services Web sont accessibles

via des interfaces dont l'invocateur n'est pas concerné et n'a pas à se soucier sur ce qui se passe derrière la scène. Les techniques à la base des services Web sont largement adoptées, et peuvent, par conséquent, habiliter les VOs. Toute fois, les services Web montrent certaines lacunes, comme le manque de sémantique et la centralisation des registres.

2.4 Grid et Services Grid

Le *Grid* (Grille) est *"un partage coordonné de ressources et une résolution distribuée de problèmes dans des organisations virtuelles, dynamiques et multi-institution"* [14].
Ce partage concerne, principalement, des cycles CPU, des ressources réseaux et de stockages, des applications et des équipements (*ex.*, capteurs, instruments de mesure, télescopes, *etc*). Cette innovation est en réponse à une nécessité en ressources informatiques de plus en plus importante. Par ailleurs, le constat de la sous-utilisation des ressources a été une motivation de départ— Il est prouvé que moins de la moitié des ressources informatiques d'une compagnie sont réellement utilisées. Nous citons, à titre d'exemple, un projet Grid célèbre : SETI@HOME (Search for Extra-Terrestrial Intelligence At Home). Il s'agit, en effet, de milliers de personnes qui partagent des cycles-processeur non utilisés de leurs PCs, pour une vaste recherche de signaux d'une vie extra-terrestre dans l'espace. Ce projet de l'université de Berkeley a produit (1999–2004) une capacité de 30 *teraflops* [1], l'équivalent de 10^5 années de calcul PC [18].

Le Grid est considéré comme un "Web Etendu" qui va au dela du partage de l'information, pour permettre à des usagers de partager des ressources informatiques. A l'instar du Web, qui a évolué d'une infrastructure de collaboration scientifique vers un moyen de communication major en e-Business ; il est prévu que le Grid trouve ses principaux arènes dans des applications distribuées commerciales (B2B, e-Commerce, informatique d'entreprise, *etc*) [43].

2.4.1 Les Organisations Virtuelles dans le Grid

L'*Organisation Virtuelle* dans un contexte Grid (dite aussi *Communauté de la Grille*), désigne une collection d'individus et/ou d'institutions qui mettent en commun des ressources tout en définissant la politique qui contrôle le partage, de telle façon à ce que les membres collaborent pour atteindre un objectif commun. Foster [14] cite les exemples : un ASP (*Application Service Provider*), qui fournit des services de type applications ; un SSP (*Storage Service Provider*), qui fournit des services de stockage ; et un fournisseur de cycles CPU. Un type bien connu des *Grid de Données* (où il s'agit de stockage), est celui des systèmes *P2P* (*peet-to-peer*) où il est possible de partager des fichiers audio, video, etc ; dans le Net (*ex.,* les réseaux eDonkey et Gnutella). L'adoption des VOs comme approche

[1] Unité de traitement CPU qui vaut 10^{12} opérations flottantes/seconde

de collaboration dans le Grid est justifiée par une meilleure organisation en communauté d'intérêt et la définition d'une politique de partage et d'un but commun.

2.4.2 Les Services Grid

Les *services Grid* sont une extension des services Web pour permettre l'accès aux ressources. Auparavant, des middlewares, tels que les portails, offraient des services Grid, mais qui été séparés et non inter-opérables. Ainsi, le Grid a pris l'orientation service Web. En effet, les architectures Grid actuelles sont ouvertes et offrent un ensemble de services extensibles à agréger dans des organisations virtuelles. OGSA [55] est la principale architectures spécifiée par la communauté Grid [52, 53].

OGSA définit un *service Grid* comme *une instance de service à état (potentiellement éphémère) qui supporte une invocation fiable et sûre, une gestion de la durée de vie, des notifications, des règles de gestion, une identification et la virtualisation* [15]. Là où les services Web sont sans état et persistants, les instances de service Grid peuvent être soit avec ou sans état et éphémères ou persistants. Les services sans état sont synchrones (*ie.*, les messages ne peuvent être "bufférisés"), point-à-points (*ie.*, utilisables par un seul utilisateur), et interagissent par de simple requête/réponse. Ils retournent simplement une réponse à une invocation précise. Les services à état peuvent être asynchrones, multi-points et interagissent par des conversations.

2.4.3 Architecture Grid Orientée Service (OGSA)

OGSA (*Open Grid Service Architecture*), est une architecture ouverte et orientée service dont le but est de construire des systèmes Grid inter-opérables, en fournissant un ensemble d'interfaces de base. Cette architecture représente les ressources par des services Grid et spécifit des interfaces et leurs comportements standards [15] :

Découverte de service. OGSA ajoute des éléments de données et une opération de recherche, 'FindServiceData', à un service Grid. La découverte des services est faite via un registre.

Création dynamique de service. OGSA permet de créer et gérer dynamiquement des instances de services nouvelles via un service de 'création de service'. Effectivement, OGSA définit une interface standard, dite *Factory*, et une sémantique dont n'importe quel service de 'création de service' doit fournir.

Gestion de durée de vie. Les services OGSA peuvent être créés et détruits dynamiquement. Ils peuvent être détruits soit implicitement, ou bien à cause d'inaccessibilité ou d'une panne système (comme le crash d'un système d'exploitation ou d'un réseau). Des interfaces sont définies pour la gestion de la durée de vie, et en particulier, pour demander les services et états associées à une opération défaillante.

OGSA s'adresse à ce problème par la définition de l'opération 'SetTerminationTime' dans l'interface du service Grid pour une gestion dite *soft-state* des instances de ce service Grid. Les *soft-state protocols* permettent aussi à OGSA de nier les instances de services établis dans une localisation distante, à moins qu'ils soient rafraîchies par un flux de messages *keepalive* (garder en vie) consécutifs.

Notification. Afin de permettre aux services de se notifier sur des changements significatifs de leurs états, de façon asynchrone, OGSA définit des abstractions et des interfaces de service communs, pour l'inscription et la délivrance des notifications. Ceci permet, par exemple, aux services composites d'interagir avec les services simples de façon standard, comme c'est le cas pour la notification d'erreurs.

Maniabilité. Dans une disposition opérationnelle, on peut avoir besoin de piloter et gérer beaucoup d'instances de services Grid. Pour cela, une interface de gestion définit ces opérations essentielles.

2.4.4 Web Service Resource Framework (WSRF)

WSRF (*Web Service Resource Framework*), est un ensemble de spécifications pour faire évoluer OGSA et la rendre compatible avec les services Web. Au départ, OGSA été basée sur OGSI (*Open Grid Service Infrastructure*), une infrastructure Grid qui a vite montré ses limites— comme la non séparation des fonctions qui permet une adoption incrémentale. WSRF, proposée en 2004 [10], voulait exploiter les nouveaux standards des services Web (en particulier, *WS-Addressing* [1]) et elle est concernée par : la création, l'adressage, l'inspection, et la gestion de durée de vie des *ressources à état*. Cette framework introduit le concept *WS-Resource* qui est une ressource à état accéssible via un service Web. Le Globus Toolkit (middleware pour l'implémentation des Grids) [53] à partir de sa quatrième version, incorpore des services implementés selon WSRF. Cette framework présente cinq familles de spécifications techniques séparées :

WS-Resource Lifetime : définit des mécanismes pour gérer le cycle de vie d'une WS-Resource.

WS-Resource Properties : définit le type et les valeurs des composants de l'état WS-Resource, qui peuvent êtres consultés et modifiés par le requérant de service via une interface service Web.

WS-Renewable Reference : définit les mécanismes qui permettent d'obtenir des versions mises à jour de la référence d'un *endpoint*.

WS-Service Group : définit un moyen par lequel les services Web et les WS-Resources peuvent être agrégés ou groupés ensemble selon un domaine spécifique.

WS-Base Faults : définit un type de "XML schema" pour une erreur de base, avec les règles d'usage par un service Web.

Une famille séparée de spécifications, nommée *WS-Notification*, définit une approche service Web standard pour la notification en utilisant les motifs "publish/subscribe". WS-Notification inclut trois spécifications, à savoir, *WS-Base notification*, définit l'interface pour les producteurs et consommateurs de la notification ; *WS-Brokered notification*, définit l'interface service Web pour les *brokers*(intermédiaires) de notification ; et *WS-Topics*, définit un mécanisme pour organiser et catégoriser les sujets d'intérêt pour la souscription à la notification.

2.4.5 Exemple d'un Grid

Le e-Science est le principal domaine d'application des Grids. L'exemple suivant illustre un Grid de Data Mining basée sur OGSA (Figure 2.3). Dans cet exemple, l'utilisateur veut découvrir, acquérir et employer des capacités distantes pour créer une nouvelle base de données (BD), à partir d'une fouille dans plusieurs BDs en ligne :

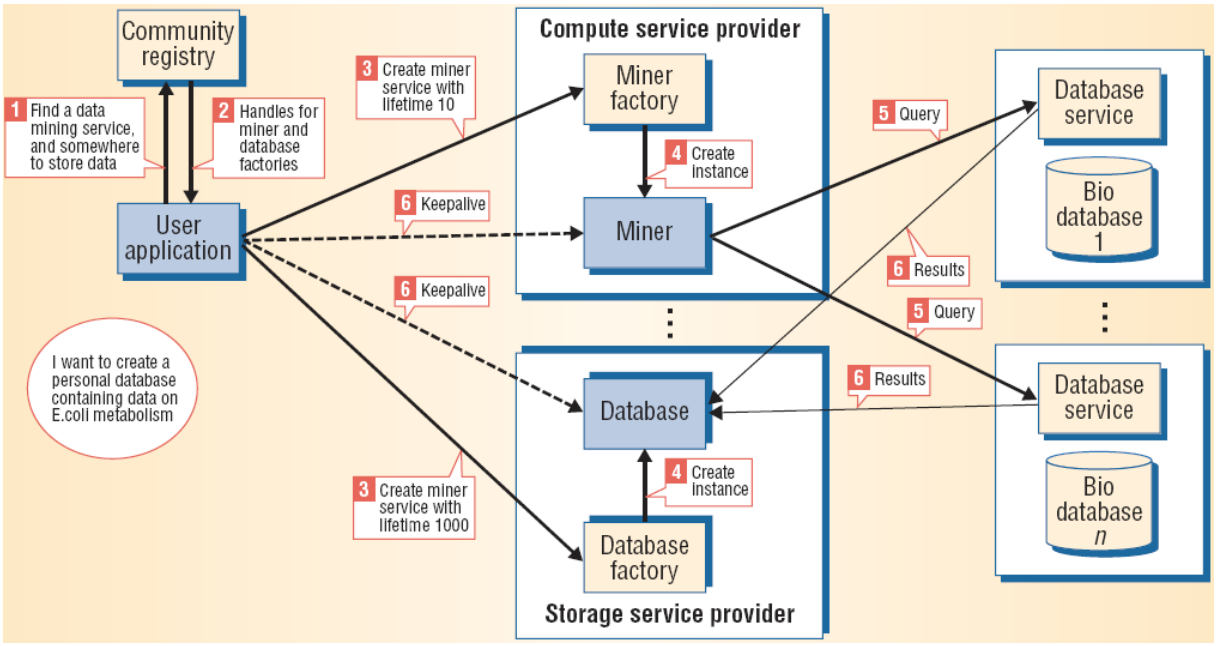

FIG. 2.3 – Exemple d'un Grid OGSA de data mining

1. L'utilisateur — ou un programme à son compte — contacte le registre maintenu par une VO, pour identifier les fournisseurs des services nécessaires (la fouille et le stockage de données). La requête utilisateur peut exiger le coût, la localisation ou la performance.
2. Le registre retourne des *handles* identifiants une *Miner Factory* et une *Database Factory* qui correspondent aux exigences de l'utilisateur.
3. L'utilisateur lance des requêtes aux *Factories Miner* et *Database*, en spécifiant des détails comme l'opération de fouilles de données à faire, la forme de la BD à créer

pour contenir les résultats, et les durées de vie des deux nouvelles instances de service.

4. Supposant le processus de négociation réussi, les deux instances de service sont créées avec les états, les ressources et les durées de vie appropriées.
5. Le service *Miner* lance des requêtes aux BDs appropriées, en agissant au compte de l'utilisateur et s'engage dans d'avantages d'opérations. Les mécanismes de sécurité OGSA s'adressent aux problèmes de délégation qui en découlent.
6. Les résultats sont retournés, soit au *Miner*, soit directement à la base de données nouvellement créée. Entre temps et selon la durée de vie négociée, l'utilisateur maintient l'envoi périodique des messages *keepalive*, pour confirmer un intérêt qui tient toujours.

L'utilisateur est notifié les résultats par le service *Miner*. Après, l'utilisateur peut maintenir les BDs nouvellement créées (toujours par des messages *keepalive*), aussi longtemps qu'il le veut ou qu'il peut se permettre. Lors d'un échec ou fin du processus, les ressources des fournisseurs de service sont libérées.

2.5 Conclusion

Dans ce chapitre, nous avons passé en revue un bref état de l'art sur les services Web/Grid. Ces domaines connaîssent une activité industrielle et de recherche intense. Néanmoins, le Grid en particulier est toujours en phase de maturation. Chacun des services Web et Grid offrent des techniques et standards de base, ayant fait leurs preuves, pour permettre des systèmes intégrables et inter-opérables. Ils ont affaire aux couches communication, contenu et ressource, offrant, ainsi, les moyens de base pour définir les services d'une infrastructure VO. D'un autre coté, il est important de noter que les services Web/Grid et les *Agents* connaissent une convergence vers un objectif commun, qui est la création de VO. On estime que la communauté Grid a, historiquement, focalisé sur le "muscle" : infrastructures, outils, et applications pour un partage de ressources fiable et sécurisé, dans des VOs géographiquement distribuées. Par contre, la communauté Agent a focalisé sur le "cerveau" : des solutionneurs autonomes qui agissent de façon flexible dans des environnements dynamiques [16]. Ces deux communautés encouragent l'intégration des technologies Agent et Grid, et prônent, parmi les directives de recherche, la définition d'une architecture de service intégrée afin de fournir une fondation robuste pour des comportements autonomes [16]. Les VOs évoluent dans des environnements ouverts et dynamiques où les SOA offrent des mécanismes de base, sans avoir affaire aux problèmes de haut niveau tel que l'organisation et le support des interactions. Les *institutions électroniques* sont prévues pour apporter des réponses (à un niveau d'abstraction plus élevé) liées au support et à la régulation des VOs. Nous étudions cela dans le chapitre suivant.

Chapitre 3

Les Institutions Électroniques

3.1 Introduction

Les organisations virtuelles supposent l'existence d'un environnement d'interaction *commun* et *ouvert*, peuplé par des entités autonomes. Dans cet environnement, l'*Institution Electronique* est une framework qui définit et impose des normes de comportement et assiste les interactions entre entités commerciales. Nous étudions dans ce chapitre l'apport des institutions électroniques aux VOs. La Section 3.2 présente les institutions électroniques. La Section 3.3 décrit certains services institutionnels pour le support des VOs. La Section 3.4 décrit l'ingénierie des systèmes multi-agent comme des institutions électroniques. La Section 3.5 est une conclusion.

3.2 Les Institutions Electroniques

Les recherches sur les systèmes multi-agents et les normes ont soulevé le concept *Institution Electronique* (EI ou e-Institution), principalement, pour fournir un environnement normatif. Les systèmes collaboratifs ouverts (*ex.*, les VOs B2B) sont le domaine d'application majeur, où il s'agit d'instaurer l'ordre et permettre l'établissement des relations de coopération. Néanmoins, il n'y a pas de consensus sur la définition d'une EI : plusieurs définitions existent avec des perceptions qui peuvent diverger. *IIIA* donne la définition : *"Electronic Institutions are a way to implement interaction conventions for agents – human or software – who can establish commitments "* [54]. C'est à dire que dans un système ouvert, l'EI est perçue comme un moyen de restreindre les agents – de façon rigide – aux comportements désirés. Par contre, Cardoso et Oliviera perçoivent l'institution électronique autrement : *"an EI is a comprehensive framework that provides a set of institutional services, while assuring norm enforcement through the imposition of sanctions and reputation mechanism"* [6]. D'où, l'EI est supposée offrir un environnement normatif et assister un travail collaboratif moyennant un ensemble de services, tels que : l'enregistrement, le

courtage ou le *mapping* d'ontologies [38]. Adoptant la première perception, précédemment citée, Sierra [41] propose un modèle conceptuel et des outils de développement, pour l'ingénierie des SMA selon une approche EI ; tandis que Cardoso et Oliviera [4, 6], adoptent la deuxième perception et proposent une framework institutionnelle à base de services pour le support des VOs.

3.3 Services Institutionnels pour les Organisations Virtuelles

Cardoso et Oliviera [4] résument les buts principaux d'une EI comme suit :

1. Supporter les interactions d'agents comme une "framework de coordination", rendant l'établissement des accords commerciaux plus efficace ; et
2. Pourvoir un niveau de confiance en offrant un "environnement normatif" imposable (exécutoire).

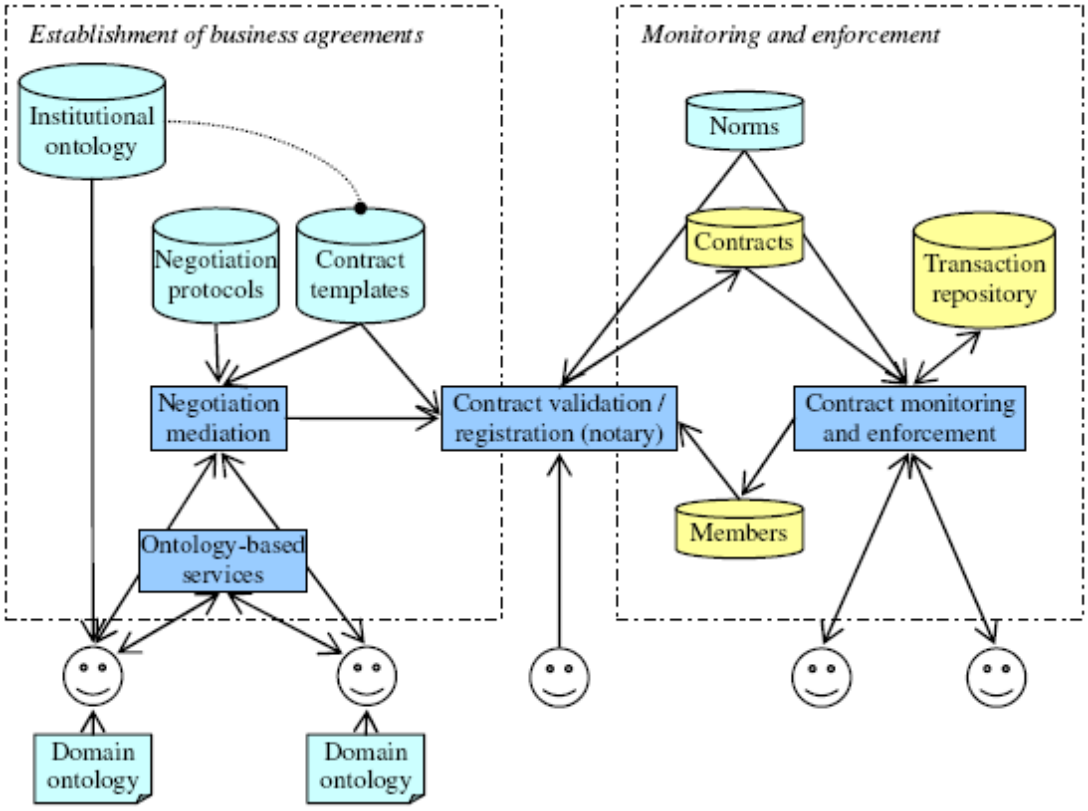

FIG. 3.1 – Services d'une Institution Electronique

Pour parvenir à ces fins, les auteurs identifient les *services institutionnels* comme moyens. Ces services sont basés-agent et sont (Figure 3.1) : la médiation de négociation [27], le mapping d'ontologie [21], le monitoring et exécution de contrat [5]. Ces services sont importants pour les phases création et fonctionnement d'une VO. Par exemple, l'EI assiste

l'établissement des contrats par la médiation de négociation, sur la base de protocoles et templates de contrats appropriés. Ce service est complété par des services basés ontologie. Cela permet d'automatiser le processus tout en gardant l'environnement ouvert, depuis que des vocabulaires de domaines divers, peuvent être utilisés par les différentes entités commerciales. Néanmoins, une ontologie institutionnelle commune est utilisée lorsqu'il s'agit des termes généraux utilisés dans les contrats. Par ailleurs, un "environnement normatif" est instauré à travers des services de *monitoring* et d'*imposition* de contrat. En effet, ces derniers enregistrent des transactions, vérifient l'applicabilité des normes ainsi que leur accomplissement dans les contrats signés (Figure 3.1).

3.3.1 Framework Normative

En considérant la nature continue d'une VO et le fait qu'elle est créée à l'intérieur d'un environnement EI, une *framework normative* structurée est fournie (Figure 3.2). Cette framework considère aussi bien les normes *contractuelles* qu'*institutionnelles* [3].

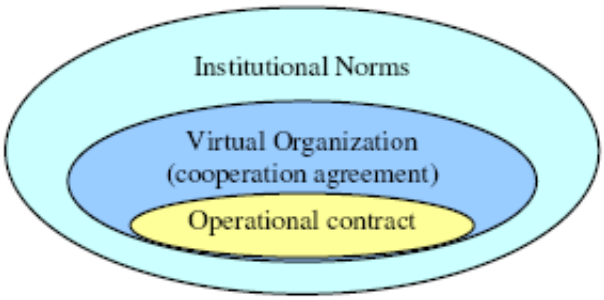

FIG. 3.2 – La Framework Normative

Les *normes institutionnelles* contiennent les clauses de contrat par défaut, permettant, ainsi, la sous-spécification des contrats, et donc, facilitent leur création. Des règlements généraux concernant la nature des consortiums peuvent être aussi définis. Les agents dépendent sur ces règlements pour formaliser les contrats d'une VO. Les règles qui décèlent la violation ou l'accomplissement des conditions, sont aussi définies à ce stade, le fait qu'elles sont indépendantes des contrats. De spécifiques politiques peuvent être définies pour la pénalisation institutionnelle des violations (*ex.,* à travers les mécanismes de réputation).

Les *normes constitutionnelles* de VO décrivent les termes de coopération, auquelles les parties adhèrent. Chaque partenaire cite la charge de travail et les prix de sa contribution, et un processus métier général est schématisé. C'est cet ensemble de normes, auquelles les parties en accord s'engagent, qui réglemente le fonctionnement des VOs, par la suite.

Des contrats spécifiques indiquant les actions à entreprendre forment la troisième couche normative. Les *contrats opérationnels* sont proposés et signés dans le contexte des accords de coopération VO, et leur création et exécution fait sujet de procédure d'imposition et de monitoring.

3.3.2 Monitoring et Imposition de Contrat

En se basant sur la framework normative précédemment décrite, l'infrastructure de monitoring et d'imposition de contrat est illustrée par la Figure 3.3.

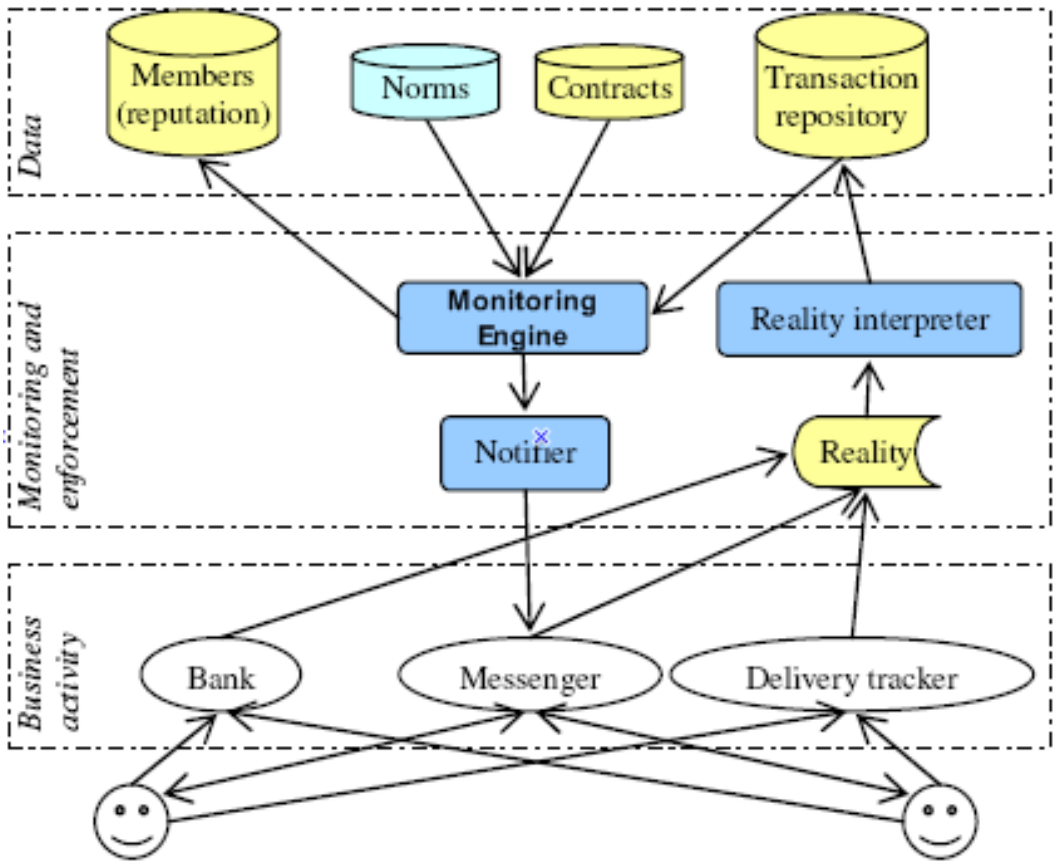

FIG. 3.3 – Monitoring et Imposition de Contrat

Afin de remplir leur promesses contractuelles, les agents ont un ensemble de facilités institutionnelles (des rôles menés par des agents institutionnels certifiés) relatives à différents types d'opérations. Précisément, les actions considérées concernent l'échange d'information, le transfert d'argent, et la livraison de produits. Ces facilités permettent de reconnaître ce qui se passe. Ensemble avec l'interpréteur de réalité, ils établissent des relations autoritaires entre rôles et assertions ; cela compose une réalité qui est interprétée pour vérifier l'accomplissement des transactions contractuelles.

Un moteur basé-règle consulte le répertoire des transactions et applique des normes institutionnelles et contractuelles (Figure 3.3). Certaines de ces normes prescrivent des pénalités en cas de violation de contrat. Ces enfreintes peuvent impliquer des mises à jours de réputation des agents impliqués. Les notifications sont transmises aux parties contractantes concernant l'état de leurs contrats, dans le cas d'un accomplissement, d'une violation ou d'un déclenchement d'obligation.

3.4 Ingénierie des Institutions Electroniques pour les Systèmes Multi-agent

La communauté des systèmes multi-agent est de plus en plus consciente de la nécessité d'un environnement normatif (EI) pour les systèmes ouverts. Toutefois, ces EI nécessitent des modèles conceptuels et outils d'implémentation. Malheureusement, peu de travaux existent, on cite principalement celui de Arcos [17]. Dans [17], les auteurs proposent un modèle conceptuel, un langage de spécification et des outils pour l'implémentation et l'exécution des EIs. L'approche suivie est d'offrir aux agents un environnement d'interaction selon des conventions explicites. De cette façon, on peut prévoir les scénarios possibles et assurer le comportement souhaité envers les agents qui adhèrent à une EI (Figure 3.4).

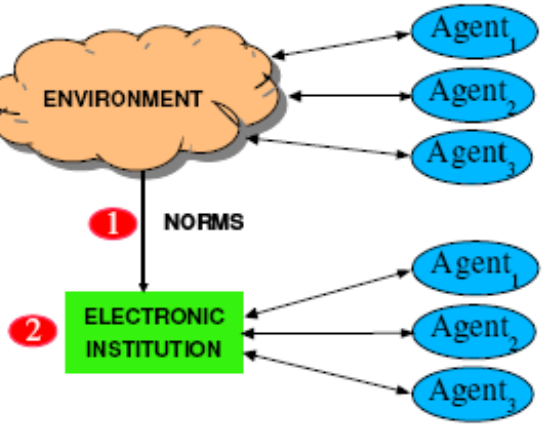

FIG. 3.4 – Ingéniérie des Systèmes Ouverts comme des Institutions Electroniques

3.4.1 Modèle Conceptuel d'une Institution Electronique

Le modèle conceptuel exprime de façon plus précise la notion EI et sert de base à son implémentation. Le modèle de Arcos [17] est basé sur les notions suivantes [41] :

- *Agents et Rôles* : Les agents sont les acteurs dans une institution électronique, interagissent par échange d'actes de langage (ou *illocutions*), tandis que les rôles sont des *patterns* de comportement standards.
- *Framework Dialogique* : définit l'ontologie et le langage de communication mis à la disposition des agents, lors des interactions au sein d'une EI.
- *Scène* : Les interactions entre agents s'articulent sur des *scènes*, grâce à des protocoles bien définis. Le protocole d'une scène est la spécification de tous les dialogues possibles basés sur le rôle au lieu de l'agent.
- *Structure Performative* : c'est un *workflow* composé par la connection des scènes. Sa spécification détermine les transitions entre scènes, selon des pré-conditions dont la satisfaction dépend du rôle de l'agent authorisé et de ses engagements courants.

– *Règles Normatives* : Leur intérêt est d'influencer le comportement des agents en imposant des obligations ou des restrictions.

Le but principal de l'EI est de forcer l'application des normes spécifiées aux agents participants, lors de l'exécution. A cet effet, l'EI fait la *médiation* d'interactions de tous les agents participants, comme c'est indiqué par la Figure 3.5.

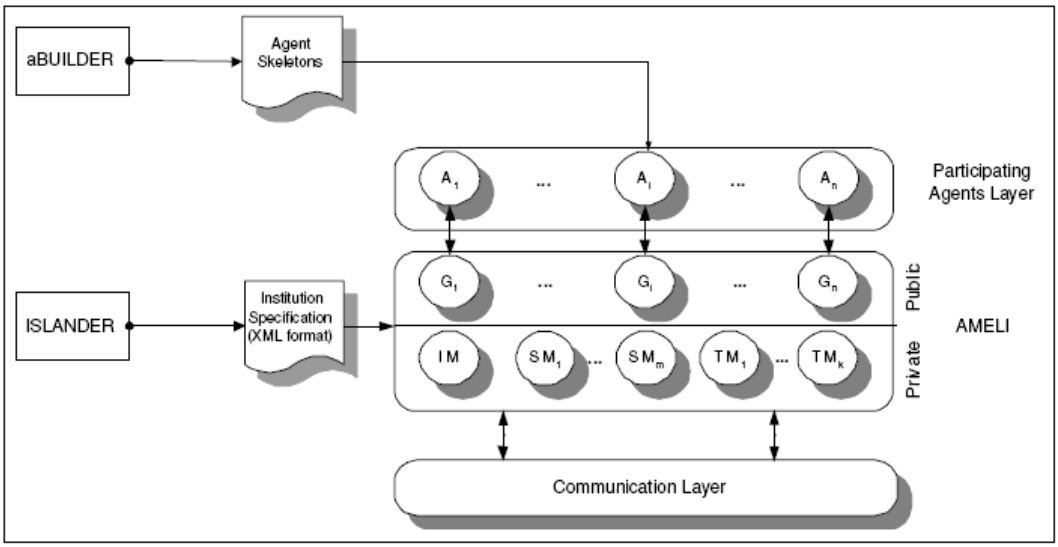

FIG. 3.5 – Médiation-agent via Institutions Electroniques

3.4.2 Environnement de Développement des Insititutions Electronique

IDE-eli (*Integrated Development Environment for Electronic Institutions*) [41], est un outil destiné au support de l'ingénierie des MAS comme institution électronique. Il permet de développer des EIs ainsi que leurs agents participants. IDE-eli adopte une approche organisationnelle "top-down" : l'organisation d'abord, ensuite les individus. Il est composé de (Figure 3.5) :

– **ISLANDER**. Un outil graphique pour la spécification des règles et protocoles.
– **AMELI**. Une plateforme software pour exécuter les institutions eeectroniques specifiées avec ISLANDER.
– **aBUILDER**. Un outil pour le développement des agents.
– **SIMDEI**. Un outil de simulation pour animer et analyser les specifications ISLANDER avant le stade de développement.

IDE-eli supporte tout le cycle de vie du développement d'une institution électronique : conception et vérification, simulation, implémentation, test et déploiement.

3.5 Conclusion

Dans ce chapitre, nous avons fait une étude sur les institutions électroniques (EI). Contrairement aux services Web— qui fournissent des techniques de bas niveau pour l'interopérabilité —, les institutions électroniques offrent des services de haut niveau afin de supporter les VOs et permettre des interactions (entre agents d'un système ouvert) dans des environnements normatifs. Deux approches d'EI existent. L'une est d'imposer des infrastructures contraignantes aux agents et rendre, ainsi, la déviation du comportement souhaité impossible. Cette approche limite sévèrement l'autonomie de l'agent. L'autre possibilité est de réglementer l'environnement, encourager le comportement coopératif à travers des contraintes normatives (récompenses/sanctions), et permettre à l'agent de choisir soit de les respecter ou de les enfreindre. L'étude bibliographique faite sur les Organisations Virtuelles, les services Web/Grid et les Institutions Electroniques nous permet de poser un fondement théorique à notre travail. Nous entamons, dans les deux chapitres suivants, l'élaboration d'un *modèle* suivit par une *architecture* VO pour un cas d'étude.

Chapitre 4

Un Modèle d'Organisation Virtuelle à base de Rôle

4.1 Introduction

La modélisation des Organisations Virtuelles (VOs) est le principal défi à relever en e-Commerce, en particulier, avant de voir de réels systèmes et une large adoption [29]. Dans ce chapitre, nous présentons une formalisation et un modèle VO pour le e-Commerce. Dans la Section 4.2, nous décrivons une VO sur plusieurs niveaux d'abstraction. Dans la Section 4.3, nous traçons la méthodologie adoptée pour concevoir notre modèle VO. Dans la Section 4.4, nous présentons le modèle VO proposé pour le e-Commerce. Dans la Section 4.5, nous concluons ce chapitre.

4.2 L'Organisation Virtuelle en Couches

Nous décrivons tout système de VO sur trois niveaux d'abstraction, du plus abstrait au plus concret, comme c'est indiqué par la Figure 4.1 :

1. **Couche Organisationnelle**. A ce niveau, un système VO est perçu comme l'ensemble des *patterns de coalitions* pouvant avoir lieu. Un pattern de coalition (ou motif) n'est qu'un ensemble de *rôles* en interaction pour accomplir un but précis. Par exemple, le pattern $VO1$ regroupe les rôles A, B et C pour servir le client final : Utilisateur (Figure 4.1). Parmi ces rôle nous distinguons trois types :
   (a) *Rôle Simple* : Peut être accompli par une seule entité (agent) *locale* à une organisation/individu. Il est représenté par un cercle (*ex*, le rôle E).
   (b) *Rôle Virtuel* : Ne peut être accompli que par une coalition d'entités *autonomes*. Il est représenté par un double cercle (*ex*, le rôle C de la coalition $VO2$ nécessite D de $SO1$, il fait appel en plus aux rôles locaux E et F).

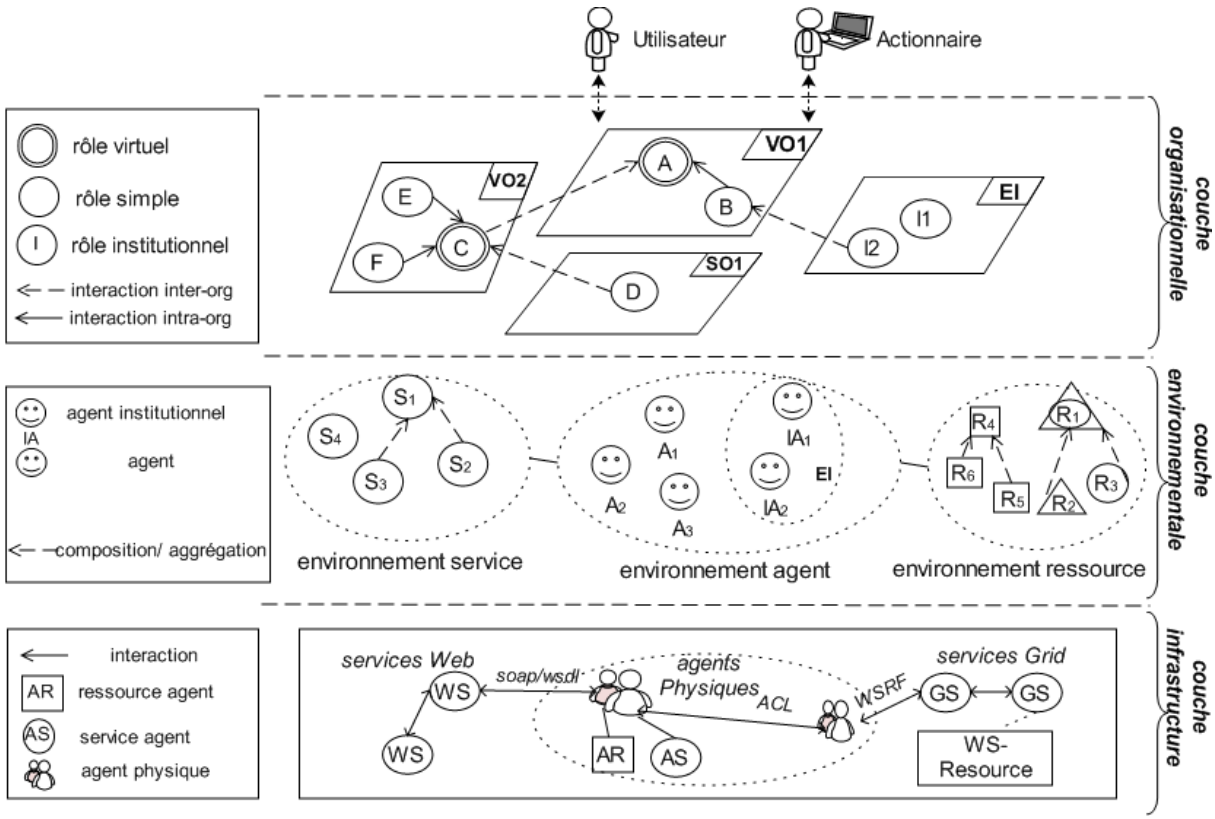

FIG. 4.1 – L'Organisation Virtuelle en Couches

(c) *Rôle Institutionnel* : Appartenant à une *institution électronique* (EI), il supporte ou régularise le comportement d'un système VO (*ex*, le rôle I_1 de EI).

Une *Single Organisation* (SO) est une coalition particulière ayant des rôles simples seulement (*ex., SO_1 ayant le rôle D*). Par conséquent, une SO intervient dans des colations comme fournisseur seulement et sans faire appel à un support externe.

2. **Couche Environnement**. En exécution, une VO est instanciée par l'affectation des rôles aux agents, dynamiquement, dans un environnement ouvert. Nous parlerons alors, selon le type de rôle, de *membres organisationnels* et de *membres institutionnels*. Le rôle de chaque membre consiste à échanger des services et/ou partager des ressources (Figure 4.1). On peut constater, à ce niveau, la virtualisation des membres par une séparation rôle-agent, et la virtualisation des ressources, par une aggrégation en quantité et/ou en type (le principe Grid), ainsi que la virtualisation des services à travers la composition.

   Par exemple (Figure 4.1), le rôle C peut être joué par l'agent A_1 ou A_2 tandis que A_3 peut jouer les rôles E et F, à la fois. La ressource R_1 est l'aggrégation de ressources de types différents (R_3 et R_2), tandis que R_4 est l'aggrégation en quantité de ressources de même type (R_5 et R_6).

3. **Couche Infrastructure**. Au niveau implémentation et déploiement, un système VO est une plateforme d'*Agents Physiques (PA)* (spécifications FIPA [50]), ayant

un langage d'interaction commun et standard (FIPA-ACL [51]). Ces agents (PA) partagent une ontologie commune (*ex.*, e-Commerce) et peuvent offrir leurs propres ressources et services (AR et AS). Par ailleurs, ils peuvent utiliser des ressources Grid (WS-Resource) (via des services Grid (GS) [56]) et appeler des services Web (WS), directement, ou par la médiation d'autres agents (Figure 4.1)

Afin d'éclaircir d'avantage cette formalisation, nous explicitons les aspects structurel et dynamique d'une VO par rapport aux deux premières couches de la Figure 4.1.

4.2.1 Aspect Structurel d'une Organisation Virtuelle

Une VO et une EI définissent conjointement des *structures organnisationnelles*. Ces stuctures sont composées de rôles *organisationels* et *institutionnels* qui interagissent par des protocoles définis, le long d'un cycle de vie VO (Figure 4.2). En exécution, ces rôles sont effectués *dynamiquement* par des agents (*ie.,* les membres VO et EI). Dans une VO, chaque membre VO représente un seul actionnaire ou un ensemble d'actionaires (*ie.,* une autre VO sous-jacente). La *structure organisationnelle* d'une VO définit toutes les possibilités de coopération, de coordination et de régulation entre organisations et entre organisations et institutions.

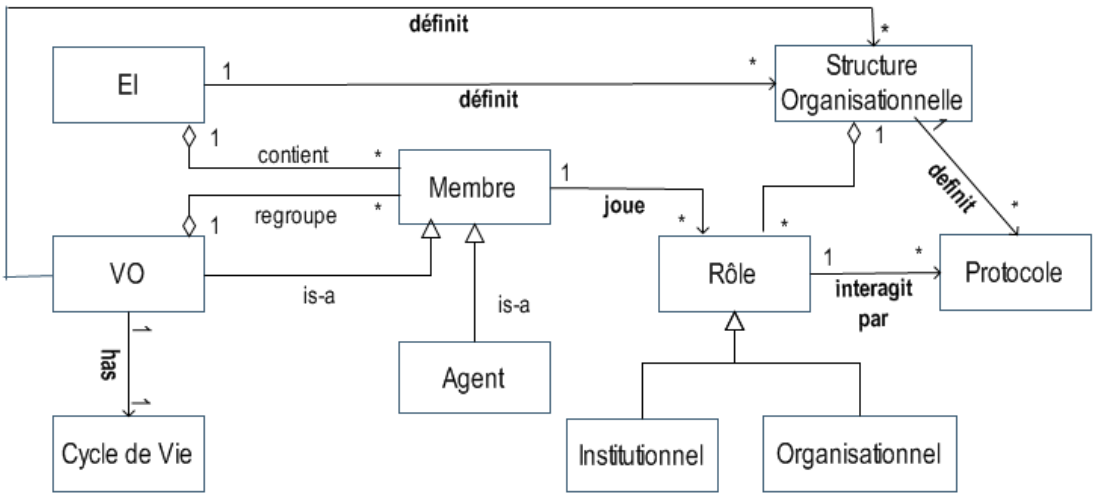

FIG. 4.2 – Schéma Conceptuel Structurel d'une VO

4.2.2 Aspect Dynamique d'une Organisation Virtuelle

La dynamique d'une VO se manifeste par les interactions entre ses membres et ceux de l'EI (Figure 4.3). Un membre organisationnel est un agent autonome, égocentrique et compétitif ; par conséquent, c'est son but (objectif final) qui *contrôle* son comportement. Toutefois, une fois engagé dans une alliance (*ie.*, devenu membre), il prend en charge le

but de son nouveau rôle (un moyen pour atteindre son objectif final). D'où, le rôle d'un agent *influence* son comportement et nous parlons plutôt d'un agent *semi-autonome*. Ce comportement consiste à échanger des services et partager des ressources (qui peuvent être considérées comme des services), selon des engagements établis dans des contrats. Quant aux membres institutionnels, ce sont des agents qui offrent des facilités aux agents organisationnels, d'une part (*ex.*, découverte de services, formation de la coalition optimale, *etc*). D'autre part, ils régularisent le comportement des membres VO, ce qui revient, par exemple, à monitorer la QoS fournis ou à imposer les contrats.

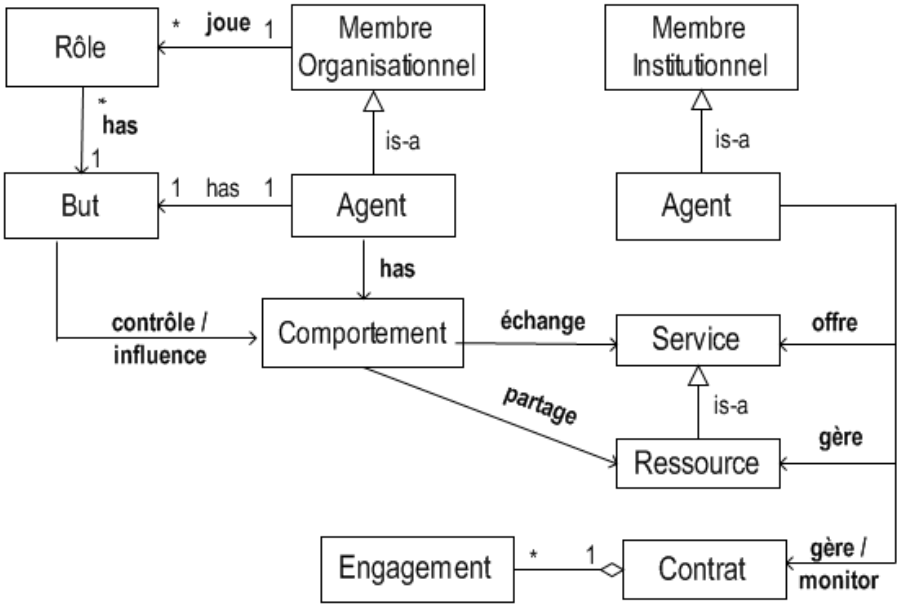

FIG. 4.3 – Schéma Conceptuel Dynamique d'une VO

4.3 Méthodologie de Modélisation

L'approche Multi-agent est largement adoptée pour développer des systèmes complexes tel que les VOs. En effet, l'ingénierie orientée agent fournit des abstractions puissantes et naturelles, comme le *rôle* et la *structure organisationnelle*. Pour concevoir notre modèle VO, nous adoptons une méthodologie basée-agent. Cette méthodologie s'inspire beaucoup des méthodologies Gaia [45, 46] et AGR [13] et comprend quatre étapes (Figure 4.4).

1. **Étape : Subdivision en VOs et EIs**. Étant donnés les besoins en entrée, il s'agit, en premier lieu, de subdiviser le système (l'entité la plus abstraite, qui n'est que l'organisation globale) en sous-systèmes (sous-organisations) faiblement couplés. Chaque organisation est distinguée par le comportement souhaité envers la réalisation d'un objectif partiel. Nous ajoutons à cela l'identification des institutions électroniques dans les quelles ces organisations vont évoluer. Les considérations suivantes peuvent servir de guide :

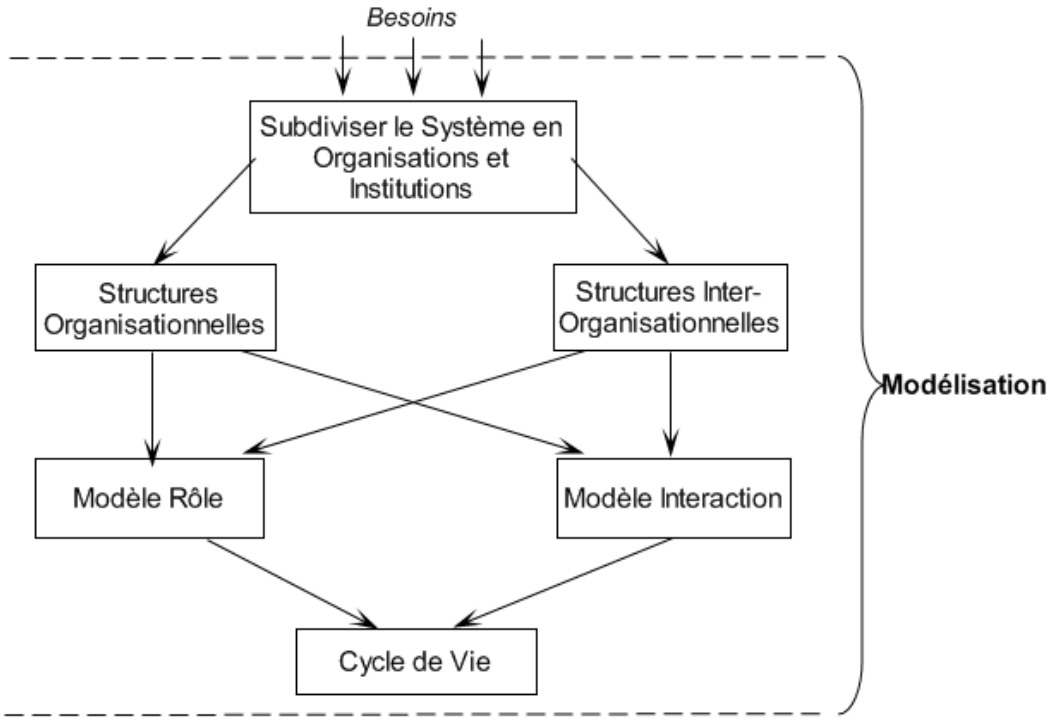

FIG. 4.4 – Etapes de Modélisation d'une VO

(a) L'analyse des besoins a déjà identifié ces organisations et institutions.

(b) Le système désiré imite une structure (organisations et institutions) du monde réel.

(c) Le souci de modularité qui préconise une subdivision devant réduire la complexité globale, par le biais de sous-systèmes facilement maniables.

2. **Étape : Identification des Structures Organisationnelles**. Consiste, à identifier les rôles et protocoles d'interactions *préliminaires* des organisations et institutions, et de les représenter dans deux types de structure organisationnelle :

   (a) *Structure Intra-Organisationnelle :* définit la hiérarchie et dépendance entre rôles, au sein d'une même organisation ou institution.

   (b) *Structure Inter-Organisationnelle :* définit les interdépendances entres organisations, ainsi qu'entre organisations et institutions.

3. **Étape : Élaboration des Modèles de Rôle et d'Interaction**. Revient à formaliser les *rôles* et *protocoles* d'interactions sur lesquels s'articule le système VO, à partir des résultats précédants.

4. **Étape : Élaboration du Cycle de Vie VO**. Consiste à spécifier la dynamique VO en termes des phases de transition, des rôles et des interactions entre rôles lors de chaque phase.

Comme pour chaque méthodologie, ces étapes doivent être répétées plusieurs fois, afin de raffiner et corriger, jusqu'à ce que cela soit jugé suffisant pour passer à la conception.

4.4 Le Modèle d'Organisation Virtuelle Proposé

En appliquant la méthodologie de la Section4.3, nous élaborons un modèle VO en se basant sur les *schémas* de rôles, structures organisationnelles et cycle de vie. Notre perception de l'organisation virtuelle est celle d'une coalition d'agents qui émerge dans un système ouvert, *supporté* et *réglementé* par une infrastructure, dite institution électronique. Suite à l'étude des systèmes VO, nous avons abouti aux rôles communs suivants :

1. *Yellow Page*(YP) : permet la découverte et la notification des services.
2. *Clearing*(CL) : permet de trouver la coalition optimale de services/fournisseurs.
3. *Quality Advisor*(QA) : conseille un client sur la QoS d'un fournisseur.
4. *Reputation Broker*(RB) : évalue la réputation des fournisseurs et consommateurs.
5. *Commitments Manager*(CM) : facilite la gestion des engagements de ressources et permet une allocation efficace.
6. *Contract Manager Monitor*(C2M) : gère, monitor et impose l'exécution des contrats.
7. *Requester*(RQ) : prépare la formation d'une VO qui répond aux besoins clients.
8. *Dealer*(DL) : traite les opportunités d'affaires et les appels d'offres profitables.
9. *Virtual Organisation Manager*(VOM) : gère le cycle de vie d'une VO.
10. *Service Provider*(SP) : fournit des services selon les contrats de VO.
11. *Service Consumer*(SC) : consomme des services selon les contrats de VO.
12. *User*(U) : c'est l'utilisateur final (un cas particulier de *SC*).

Avant de spécifier ces rôles, nous donnons les structures organisationnelles à la base.

4.4.1 Structures Organisationnelles

Dans chaque VO évoluant dans un "environnement institutionnel", trois types d'acteurs sont présents :

- *User* : est l'utilisateur ou bien le consommateur final.
- *Stakeholder* : est un actionnaire capable de fournir et/ou consommer des services.
- *EI-Staff member* : est un des représentants d'une institution électronique.

Les interdépendances entre les rôles de ces actionnaires sont décrites dans deux types de structure : une structure *intra-organisationnelle* et une structure *inter-organisationnelle*.

Structures Intra-Organisationnelles

Il s'agit des relations entre rôles et la hiérarchie dans une même organisation ou institution. Comme c'est indiqué par la Figure 4.5, le rôle *Stakeholder* est abstrait ayant SP et VOM comme sous-rôles (relation *"sub"*). Le rôle VOM contrôle le fonctionnement de son SP

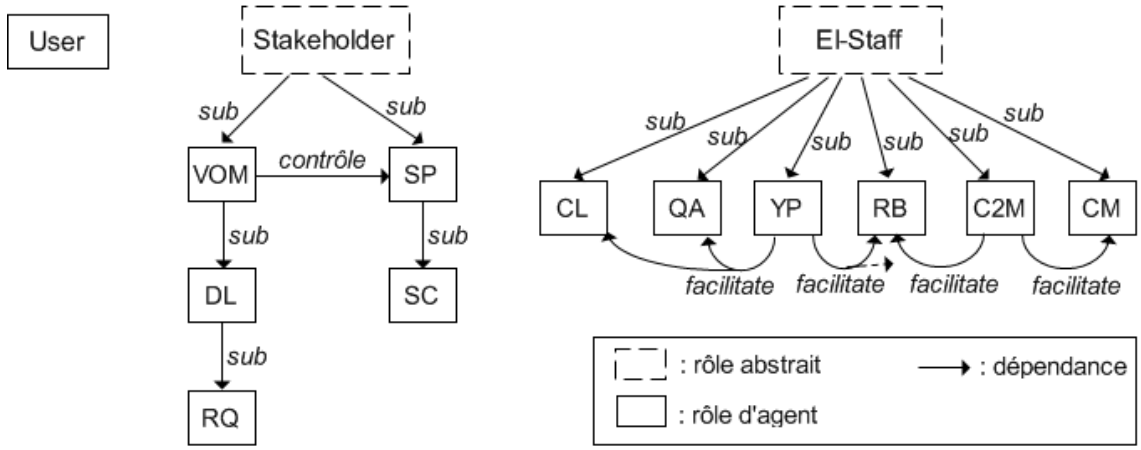

FIG. 4.5 – Structures Intra-Organisationnelles

(relation *"control"*) et son de SC (par transitivité, le fait que SC est un sous-rôle de SP). Le DL est un sous-rôle de VOM, le fait qu'il lui prépare la VO à former, comme le RQ est un sous-rôle de DL vu que ce dernier peut devenir requérant (*ie.,* RQ) pour parvenir aux besoins de son client, dans certains cas. Cette hiérarchie est justifiée par le degrée de dépendance fonctionnelle entre rôles. Cependant, cela n'empêche pas de disperser ces rôles sur plusieurs agents d'une même organisation.

D'autre part, dans une EI, un membre *EI-Staff* peut avoir au moins un des rôles : YP, CL, QA, RB, $C2M$ ou CM. Le rôle YP est un faciltateur (relation *"facilitates"*) pour tout ces rôles, comme le $C2M$ est un facilitateur de RB (lui transmet la réputation des fournisseurs/consommateurs à partir des contrats qu'il monitor). De même, le $C2M$ est un facilitateur de CM, puisqu'il lui communique les engagements actifs qui permettent au CM de trouver une allocation de ressource. Quant au rôle $User$, il opère individuellement.

Structures Inter-Organisationnelles

Elles capturent les dépendances entre organisations/individus et entre organisation et institution, au sein des VOs (Figure 4.6).

Par exemple, le rôle U_1 utilise les services de SP_1 (relation *"uses"*), comme le fait SC_1 (de l'organisation Org_1) avec SP_2 (de l'organisation Org_2) dans le contexte de VO_1. Avant la formation de VO_1, le rôle RQ_1 utilise, entre autres, DL_2 pour parvenir aux besoins de son organisation Org_1. Le VOM_1 utilise les facilités du $C2M_1$ pour établir et surveiller des contrats (relation *"facilitates"*), comme son contrat avec VOM_2. Ce même $C2M_1$ va réglementer le comportement du VOM_1 (relation *"regulates"*), au compte de VOM_2 et de U_1. En outre, les rôles CM et RB facilitent le rôle DL, respectivement, lors de la l'allocation de ressources et le choix de client à bonne réputation. De même, les rôles QA, RB, CL et YP facilite RQ pour le choix du fournisseur au QoS adéquate, ayant une bonne réputation, de trouver une coalition optimale et de découvrir les fournisseurs et services, respectivement.

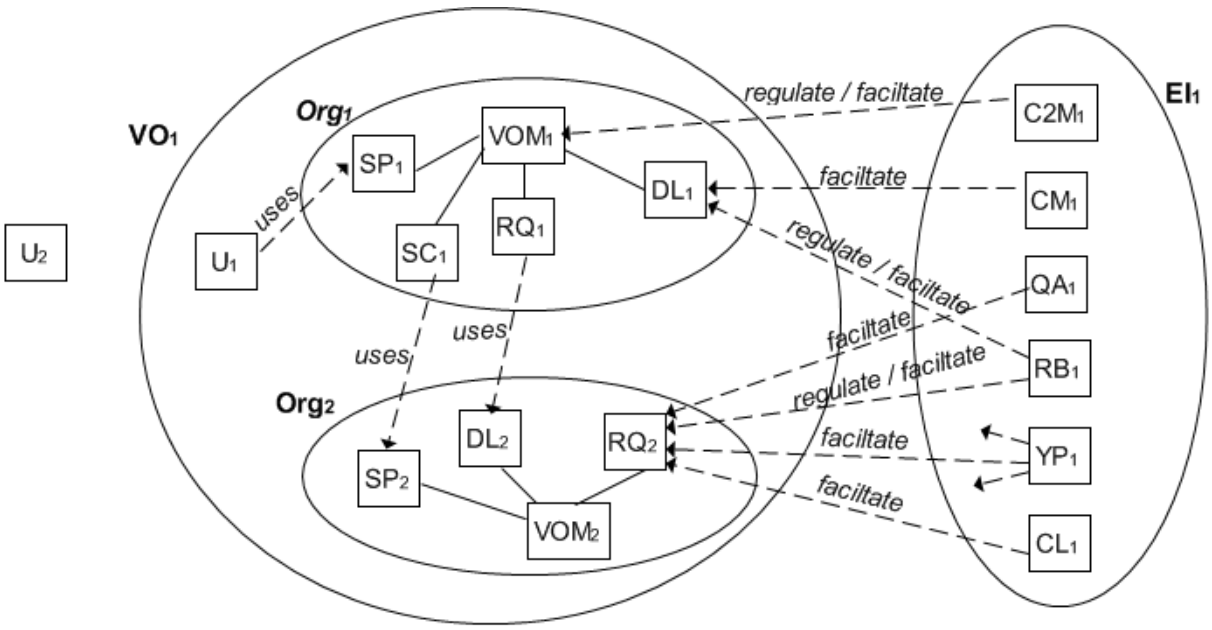

FIG. 4.6 – Structures Intra-Organisationnelles

4.4.2 Schéma de Rôle et d'Interaction

Nous nous somme basé sur la modélisation Gaia[1] [46], pour spécifier les rôles et les interactions du modèle VO proposé. On sous-entend par rôle la position occupée par un agent dans une organisation, décrite par son but, ses permissions, ses activités et protocoles d'interaction, ainsi que par ses responsabilités [46]. Nous ajoutons à cela, le *type* de rôle, ses *super-rôles* et ses *trigger-roles* (rôles déclencheurs) (Table 4.1).

Role Schema :	NOM DE RÔLE
Description :	courte description texctuelle du rôle.
Type :	catégorie de rôle : Institutionnel (EI) ou Organisationnel (VO).
Activities & Protocols :	tâches locales et interactions dans les quelles ce rôle prend part.
Permissions :	"droits" accordés sur des ressources de l'environnement .
Supper-Roles :	tout les rôles dont celui-ci est *subordonné* (liée par la relation "sub").
Trigger-Roles :	tout les rôles qui sollicitent celui-ci.
Responsabilities :	
Liveness	*responsabilités de vivacité*, dictent ce qui doit être fait tant que ce rôle est actif.
Safety	*responsabilités de sûreté*, listent des conditions à maintenir, auquelles nous. ajoutons des règles de gestion d'exceptions

TAB. 4.1 – Le Schéma de Rôle

Les *expressions de vivacité*(liveness) sont des expressions régulières d'*activités* et de *protocoles*, qui permettent de spécifier les responsabilités d'un rôle. Nous utilisons pour cela les opérateurs de la Table 4.2 [45] (en rajoutant l'opérateur $|_\times$).

Nous gardons le schéma de protocole de Gaia [45] pour la spécification des interactions. Les schémas de rôle et de protocole sont illustrés à travers le rôle *YP (Yellow Pages)*, dans la sous-section suivante. Par la suite, en utilisant le schéma de rôle présenté, nous spécifions

[1]Méthodologie basée organisation et rôle pour l'ingénierie des systèmes multi-agent

Opérateur	Interprétation
$x.y$	x suivi par y
$x\|y$	x ou y aura lieu
$x\|_\times y$	x ou y aura lieu, exclusivement
x^*	x aura lieu 0 ou plusieurs fois
x^+	x aura lieu 1 ou plusieurs fois
x^ω	x aura lieu indéfiniment
$[x]$	x est optionnel
$x\|\|y$	x et y sont intercalés (*interleaved*)

TAB. 4.2 – Opérateurs des Expressions de Vivacité

le reste des rôles du modèle VO proposé.

4.4.3 Le Rôle *Yellow Pages*

Role Schema : YELLOW PAGES (YP)
Description : fait le "matchmaking" et le "monitoring" des services fournis. **Type :** Institutionnel.
Protocols & Activities : Register, Recommand, Subscribe, addService, removeService, ...
Permissions : **generates/ reads / modifies** répertoireServices
Super-Roles : EI-Staff member **Triggers-Roles :** N'importe
Responsabilities ***Liveness*** YP = WORK$^\omega$ WORK = FACILITATOR | MONITOR FACILITATOR = (Register ‖ addServices) | (Recommand ‖ searchServices) | (UnRegister ‖ removeServices) MONITOR = Subscribe.(MonitorEvent.Notify)* ***Safety*** • UnSubscribe → StopMonitorNotifyEvent

TAB. 4.3 – Le Schéma de Rôle *Yellow Pages (YP)*

Le Yellow Pages (YP) est un rôle institutionnel. Son expression de vivacité inclut deux sous-rôles parallèles exécutés indéfiniment (Table 4.3). Le premier, FACILITATEUR, fait trois tâches concurrentes : la publication des services, la recommandation d'un fournisseur, et l'annulation d'une publication. Par exemple, la recommandation d'un fournisseur consiste en une *requête*, la *recherche* des services, et une *réponse*. C'est pourquoi le protocole Recommend est intercalé avec l'activité searchServices. A ce niveau, un protocole capture la nature de l'interaction (son but essentiel), sans détailler la séquence et les messages échangés. Par exemple, le protocole Recommand de YP (Figure 4.7) est initiable par n'importe quel rôle (le YP est un participant), il consiste à trouver et transmettre les fournisseurs en sortie, à partir des services recherchés en entrée. Ce rôle notifie les entités inscrites aux événements qu'il monitor, dès occurrence— comme l'arrivée d'un nouveau fournisseur(sous-rôle MONITOR). La seule propriété de surêté est d'arrêter ce monitoring

aussitôt le concerné désinscrit (UnSubscribe).

Par convention dans une expression de vivacité, nous notons un sous-rôle par des MAGISCULES PETITES, les activités en Souligné, et les protocoles en SANS SÉRIF. Quant aux expressions de sûreté, c'est une liste de prédicats et des règles exprimée à partir des activités, protocoles et ressources d'un rôle.

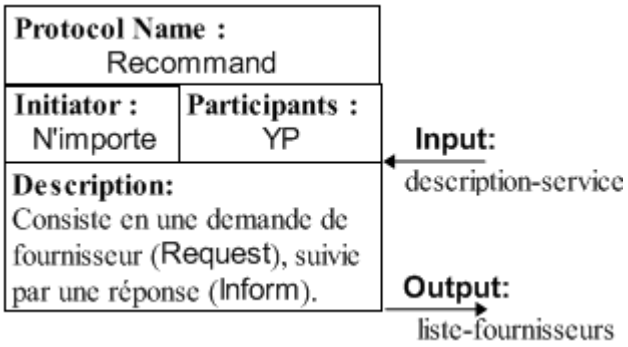

FIG. 4.7 – Exemple d'un Schéma de Protocole pour le rôle *YP*

4.4.4 Le Rôle *Clearing*

Role Schema : CLEARING (CL)
Description : Etant donné les besoins et les offres de services (avec leurs prix et utilités), ce rôle calcule la coalition de fournisseurs optimale.
Type : Institutionnel.
Protocols & Activities :
Register, CoalitionRequest, Clear, CoalitionResponse
Permissions :
reads requête // lire les besoins du requérant
reads offre[i] // lire l'offre de services du fournisseur i. i=1,..,nbr-fournisseurs
reads prix[i][j] // lire le prix du service j de fournisseur i. j=1,..,nbr-services[i]
reads utilité[i][j] // lire l'utilité du service j de fournisseur i. j=1,..,nbr-services[i]
Super-Roles : EI-Staff member
Triggers-Roles : RQ
Responsabilities
Liveness
CL = Register.(CoalitionRequest.Clear.CoalitionResponse)$^{\omega}$
Safety
• (Clear.temps < requête.timeout) ∨ (solution-optimale)

TAB. 4.4 – Le Schéma de Rôle *Clearing (CL)*

Le rôle *Clearing (CL)* filtre des offres afin de constituer une coalition qui satisfait au mieu des besoins donnés. Un actionnaire voulant former une VO, transmet au CL les services désirés, les offres reçues ainsi que leurs prix et utilités (*ie.,* valeurs combinées de la QoS et de la crédibilité). En retour, le CL donne les fournisseurs, services et quantité à retenir. Nous spécifions ce rôle, présent dans CONOISE-G [29], par le schéma de la Table 4.4.

Après son annonce au YP (Register), et à l'issue de la requête d'un RQ (CoalitionRequest), le CL applique des algorithmes de *clearing* (Clear) pour enchères combinatoires [12]. Par exemple, une offre peut contenir les services (s_1, s_2) de quantités (q_1, q_2), dont les formules

de prix sont : $(30 \times q_1 + 3 \times q_2)$ si ($q_1 < 10$ ou $q_2 < 24$), et $(20 \times q_1 + 2 \times q_2)$ si ($q_1 \geq 10$ et $q_2 \geq 24$) (par mois). Le résultat est l'ensemble des fournisseurs retenus avec les prix et quantités des services choisis. Ce résultat est retourné au RQ (CoalitionResponse). Comme propriété de sûreté, ce rôle doit respecter "le délai de réponse" ou bien donner "une solution optimale", tout dépend de l'exigence du RQ. En effet, chacun des algorithmes de clearing présentés dans [12] assure une propriété sans l'autre.

4.4.5 Le Rôle *Quality Advisor*

Role Schema : QUALITY ADVISOR (QA)
Description : évalue la qualité des services des fournisseurs. **Type :** Institutionnel.
Protocols & Activities : QoSAssesmentRequest, Assess, Store, QoSAssessmentResponse
Permissions : **reads/modifies** QoS-DB // consulter l'historique des QoS fournies, modifier son estimation.
Super-Roles : EI-Staff member **Triggers-Roles :** RQ
Responsabilities ***Liveness*** YP = Register.(HANDLEQOSFEEDBACK | QOSADVICE)$^{\omega}$ HANDLEQOSFEEDBACK = UserQoSRate.Store QOSADVICE = QosAssessmentRequest.Asses.QosAssessmentResponse ***Safety*** • true

TAB. 4.5 – Le Schéma de Rôle *Quality Advisor (QA)*

Nous spécifions le rôle *Quality Advisor (QA)* (Conseillé en QoS), présent dans CONOISE-G [29], par le schéma de la Table 4.5. Le QA maintient une base de données des estimations de QoS, selon un modèle orienté utilisateur [40]. En effet, pour chaque attribut q_i d'un service S, on collècte : $\langle QE(S, q_i), QP(S, q_i), QR(S, q_i)\rangle$, où $QE(S, q_i)$ est la qualité *espérée* par l'utilisateur, pour le service S par rapport à l'attribut q_i ; $QP(S, q_i)$ est celle *réellement fournie* (telle enregistrée par les capteurs de l'environnement), et $QR(S, q_i)$ est la QoS *estimée* par l'utilisateur (reçue comme *feedback* après la fourniture de service). La qualité totale d'un service "par rapport à l'attribut" q_i, notée $QR_t(S, q_i)$, est estimée sur la base des $QR(S, q_i)$ des différents utilisateurs. La qualité totale "d'un service", notée $QR_t(S)$, elle est faite sur la base des $QR_t(S, q_i)$.

Le QA procède comme suit (Table 4.5) : il publit son service au YP ; il conseil les utilisateurs (RQ) sur la QoS (QOSADVICE), et traite les estimations de QoS faites par les utilisateurs (HANDLEQOSFEEDBACK). Par exemple, la qualité d'un service "mise à jours de News" par rapport à la fréquence, peut être :
$\langle QE('News', fr) = 0.65, QP('News', fr) = 0.76, QR('News', fr) = 0.85\rangle$.
Cela signifie une espérence de fréquence en dessus de la moyen (0.65), une mise à jour effective plus fréquante (0.76) et, par conséquent, la QoS est considérée – par l'utilisateur – bonne (0.85). Lorsqu'un RQ demande une estimation de QoS (QosAssessmentRequest),

celle-ci est faite (Assess) pour chaque attribut q_i par : $Q(S, q_i) = \sum_{j=1}^{k} \omega_j \times QR_j(S, q_i)$, une somme pendérée des estimations des utilisateurs, si aucune espérance n'est donnée sur q_i, par l'utilisateur requérant. Dans le cas contraire, où $E(S, q_i) = \alpha \in [0, 1]$ est l'espérance sur q_i de S, $Q(S, q_i) = \sum_{j=1}^{m} \omega_j \times QR'_j(S, q_i)$, où $QR'_j(S, q_i)$ est une estimation dont l'espérence $QE'_j(S, q_i)$ est *compatible* avec $E(S, q_i) = \alpha$ (dont les valeurs sont proches). L'estimation totale, $Q(S)$, est retournée par QosAssesmentResponse.

4.4.6 Le Rôle *Reputation Broker*

Role Schema : Reputation Broker (RB)
Description : maintient une base de connaissances sur les réputations des fournisseurs et consommateurs, à partir des expériences passées. **Type :** Institutionnel.
Protocols & Activities : AskOpinion, MakeOpinion, TellOpinion, Register, Subscribe, ...
Permissions : **generates** Rep-DB // créer, lire et modifier une BD sur l'historique des réputations (clients/fournisseurs).
Super-Roles : EI-Staff member **Triggers-Roles :** RQ, DL, YP, C2M, RB
Responsabilities ***Liveness*** RB = Init.(Brokering | HandleEvents)$^{\omega}$ Init = Register.Subscribe$_1$(YP) Brokering = (NewReputationSource.Subscribe$_2$(ReputationSource)) | (NewReputation.Store) HandleEvents = AskOpinion.MakeOpinion.TellOpinion ***Safety*** • source-réputation-authentifiée

Tab. 4.6 – Le Schéma de Rôle *Reputation Broker (YP)*

C'est un rôle institutionnel essentiel dans les systèmes ouverts où les comportements d'agents sont imprévisibles. Le *Reputation Broker (RB)* maintient une base de connaissances sur la réputation des fournisseurs et consommateurs. Initialement, le RB (Table 4.6) publie son service et s'inscrit au YP (Subscribe$_1$(YP)), afin d'être notifié des nouvelles *sources* de réputation (*ex.,* un C2M ou un autre RB). Le RB s'inscrit à une sources (Subscribe$_2$(ReputationSource)), pour se mettre à jour (Store), une fois notifié (NewReputation). Le RB authentifie ces sources (propriété de surêté). Par ailleurs, il donne ses opinions sur les fournisseurs (au RQ) et clients (au DL) (TellOpinion).

4.4.7 Le Rôle *Commitments Manager*

Dans un Grid, un fournisseur dispose de ressources *individuelles* et d'autres *mutualisées* (provenant de VOs où il est membre "consommateur"). Néanmoins, au fur à mesure d'adhérer à des VOs comme "fournisseur", les ressources d'un actionnaire se lient à des engagements. Chaque engagement spécifi le type, la quantité et l'intervalle d'allocation d'une ressource, ainsi que les parties impliquées. Face à une telle situation, nous introduisons le CM (*Gestionnaire d'Engagements*) (Table 4.7) qui *réifit* des engagements pour

Role Schema : COMMITMENTS MANAGER (CM)
Description : supporte l'allocation des ressources par gestion d'engagements. Ce qui revient à trouver des façons d'allocations efficaces (*ie.*, satifaisant un maximum d'engagements de ressource).
Type : Institutionnel.
Protocols & Activities : GetCommitment, ReceiveRequest, AnswerRequest, FormulateCSP, SolveCSP, Register
Permissions : **reads** requête // consulter les besoins en ressources du requérant **reads** engagements // lire les engagements *débiteurs* et *créditeurs* sur les ressources du requérant. **reads** ressources // lire les ressources "individuelles" du requérant (types et quantités).
Super-Roles : EI-Staff member
Triggers-Roles : DL
Responsabilities ***Liveness*** CM = Register.WORK$^{\omega}$ WORK = ReceiveRequest.SOLVERESOURCEALLOCATIONPROBLEM.AnswerRequest SOLVERESOURCEALLOCATIONPROBLEM = GetCommitments.FormulateCSP.SolveCSP ***Safety*** • requérant-authentifié ∧ requérant-autorisé

TAB. 4.7 – Le Schéma de Rôle *Commitments Manager (CM)*

trouver la meilleure allocation de ressources—*i.e,* qui satisfait un maximum d'engagements et qui considère des priorités entre engagements, le cas échéant. Effectivement, cela revient à résoudre un CSP (*Constraint Satisfaction Problem*) [9], qui peut, en plus, mettre en évidence un déficit de ressources et inciter le DL à contracter ailleurs.

A l'issue d'une requête, le CM rassemble tout les engagements *débiteurs* et *créditeurs* du DL requérant, à partir du C2M (GetCommitments). Ensuite, le CM formule, résoud et transmet le CSP d'allocation. Pour une surêté de fonctionnement, ce rôle doit authentifier chaque requérant—pour éviter de donner une solution d'allocation à un concurrent.

4.4.8 Le Rôle *Requester*

Le *Requester (RQ)* est un rôle organisationnel responsable de trouver une coalition de fournisseurs adéquate, lors de la *pre-formation* VO. La nécessité de former une VO est signalée par un DL ou un VOM. C'est alors, que le RQ demande au YP de lui recommander des fournisseurs potentiels. Le RQ fait sa sélection selon le processus : *(i) Appel d'Offre* (CFP), ou *(i) Tableau Noir* [19], tout dépend de la complexité. En effet, ce dernier est une approche *bottom-up* pour trouver la combinaison de fournisseurs qui va résoudre un problème donné. Dans ce cas, le RQ invite à la soumission des solutions (PromptAll), initialise un "tableau noir" (NewBlackBoard), et à plusieurs reprises reçoit et transmet la solution partielle courante (PutPartialSolution et GetPartialSolution). Un participant qui consulte la solution partielle, essay de la compléter en se mettant d'accord avec celui qui la proposée. La solution finale est construite itérativement et ce processus peut être relancé si la solution de départ n'aboutit pas. Une fois obtenue, la solution finale est soumise au processus de décision (MAKEDECISION). A ce stade, le RQ obtient l'évaluation des QoS (à partir d'un QA), des crédibilités (via le "composant Trust" de son agent), éventuellement, les ré-

Role Schema : REQUESTER (RQ)
Description : trouver des partenaires satisfaisant aux mieux les besoins d'une VO en formation.
Type : Organisationnel.
Protocols & Activities :
CFP, RecommandAll, AskQuality, AskReputation, Decide, ...
Permissions :
reads besoins // lire le paquet de services désiré ainsi que les contraintes dessus.
generates structureVO // générer la structure de la VO à former.
modifies structureVO // modifier la structure de la VO à re-former.
Super-Roles : VOM, DL
Triggers-Roles : VOM, DL
Responsabilities
Liveness
RQ = (ReceiveRequest.PREPAREVO.AnswerRequest)$^{\omega}$
PREPAREVO = RecommandAll.(PROCESS1$|_{\times}$PROCESS2)
PROCESS1 = CFP $\parallel$ MAKEDECISION
PROCESS2 = PromptAll.(NewBlackBoard.(GetPartialSolution$|$ PutPartialSolution)*)*.MAKEDECISION
MAKEDECISION = (AskQuality.AssesTrustworthiness.[AskReputation])$^{nbr-fournisseurs}$.ClearOffers.Decide
Safety
• besoins.requérant $\notin$ CFP.participants
• ($\forall s$: besoins.service, $\forall f$: CFP.participant, (Trust(f,s).confidence$<$ *seuil* $-$ *min*) $\vee$ ($\neg\exists$ composant-Trust) $\rightarrow$ AskReputation(f,s)
• PREPAREVO.temps $<$ besoins.timeout

TAB. 4.8 – Le Schéma de Rôle *Requester (RQ)*

putation des fournisseurs (à partir d'un RB). En effet, le RQ sollicite un RB, si son agent ne dispose pas d'un "composant Trust", ou bien s'il n'a pas eu assez d'expériences pour juger la crédibilité d'un fournisseur—*ie.,* son niveau de *confidence* pour un service s de fournisseur f, est inférieur à un certain *seuil-min* (Table 4.8). Par la suite, le RQ demande au CL de trouver la coalition optimale à partir des offres, utilités (valeurs combinées de Qos et crédibilité) et prix de services. Cette coalition est transmise au requérant (DL ou VOM).

Dans un environnement dynamique compétitif, les VOs doivent être rapidement formées, afin d'exploiter un marché. En outre, la "pré-formation" d'une VO peut impliquée celles de plusieurs VOs sous-jacentes. Par conséquent, cette phase (PREPAREVO) est contrainte par le temps spécifié par l'actionnaire à l'origine (propriété de sûreté de la Table 4.8).

4.4.9 Le Rôle *Dealer*

Le rôle DL (*Dealer*) traque les opportunités d'affaire profitables (Table 4.9). Ce rôle organisationnel analyse chaque appel d'offres pour formuler une offre s'il le juge intéressant, et notifit au VOM la nouvelle VO à former. L'analyse d'une offre consiste à évaluer la crédibilité du client (AssessTrustworthiness), éventuellement, à se renseigner sur sa réputation (AskReputation), ensuite à demander au CM les possiblités d'allocation courantes (GetRessourceAllocationsSolutions). Après, lors de la prise de décision (DECIDE), le RQ décide soit de ne pas soumettre (DontBid) ou de soumettre (PrepareBid), et prépare une

Role Schema : DEALER (DL)
Description : traite les opportunités de marché avec des clients/partenaires commerciaux. Pour cela, il répond aux appels d'offres et met en évidence, éventuellement, le besoin de contracter de nouveaux fournisseurs
Type : Organisationnel.
Protocols & Activities :
ReceiveCFP, PreformVO, DontBid, GetResourceAllocationSolutionn, ...
Permissions :
reads requête // lire les besoins et attributs de l'appel d'offres.
reads sollutions-allocation // lire les façons optimales d'allocation face à une demande.
Super-Roles : VOM
Triggers-Roles : VOM, RQ
Responsabilities
Liveness
DL = Register.WORK$^{\omega}$
WORK = ReceiveCFP.PROCESS.AnswerCFP.OfferAcknowledgment.NotifyVOM
PROCESS = ANALYSE.DECIDE
ANALYSE = AssessTrustworthiness.[AskReputation].GetResourceAllocation
DECIDE = DontBid $\|_{\times}$ PREPAREBID
PREPAREBID = BidLonly | BidFromMyVO | BidFromToBreakVO | (PreFormVO.DECIDE) | BidFromNewVO
Safety
• PROCESS.time < requête.timeout
• OfferAcknowledgment.timeout → CancelOffer
• ($\forall c$: client, (Trust(c).confidence< $seuil - min$) $\vee$ ($\neg\exists$ composant-Trust) → AskReputation(c))

TAB. 4.9 – Le Schéma de Rôle *Dealer (DL)*

offre selon l'une ou une combinaison des alternatives suivantes :

- Soumettre à partir de ses propres moyens (BidLonly).
- Soumettre à partir des ressources mutualisées par sa VO (BidFromMyVO).
- Soumettre à partir des ressources d'une VO qu'il va abondonner (où il est membre fournisseur), s'il juge cela profitable (BidFromToBreakVO).
- Soumettre à partir d'une nouvelle VO (BidFromNewVO) dont il envisage la formation (PreFormVO.DECIDE), si son offre sera acceptée (OfferAcknowledgment).

La pré-formation d'une VO (PreFormVO) peut se solder par un échec, ce qui justifit une deuxième délibération (Decide). Le DL a trois responsabilités de sûreté (Table 4.9). Il doit respecter le délai de soumission, s'il veux voir son offre considérée. Il doit annuler son offre après un certain *deadline*, s'il ne reçoit pas de confirmation. Enfin, le DL demande la réputation de son client potentiel, lors de l'absance du "composant Trust" ou lors de manque de confidence.

4.4.10 Le Rôle *Virtual Organisation Manager*

Ce rôle organisationnel est responsable de former et de maintenir la viabilité de sa VO. Cela revient à demander l'établissement des contrats (ContractMembers), suivre leurs déroulement (ContractMonitoringReport$^{(1)}$), et de traquer les partenaires intéressants (Opportunity Emergence$^{(2)}$) et adapter la VO au changement de besoins (Requirements Changes$^{(3)}$). C'est le VOM qui active les rôles DL et RQ, comme il peut initialiser un sub-VOM pour lui déléguer la gestion d'une VO sous-jacente. Dans chaque phase VO, le VOM

Role Schema : VIRTUAL ORGANISATION MANAGER (VOM)

Description : gère le cycle de vie d'une VO dont il doit maintenir la viabilité. Il coordonne ses membres, et surveille l'environnement pour traiter les exceptions et tirer profits des nouvelles opportunités d'affaire.

Type : Organisationnel.

Protocols & Activities :

Activate(Roles), ContractMembers, PreFormVO, Subscribe, Decide, ...

Permissions :

reads/modifies structureVO // pour former ou re-former une VO.

generates/modifies contratsVO // créer et m-à-j les contrats de la VO.

Super-Roles : Stakeholder

Triggers-Roles : DL, YP, subVOM, C2M

Responsabilities

Liveness

VOM = INIT.LIFECYCLE$^{\omega}$

INIT = Activate(DL,RQ) | Activate(subVOM)

LIFECYCLE = HANDLEEVENTS.(FORMVO $|_{\times}$ REFORMVO $|_{\times}$ RECONFIGUREVO $|_{\times}$ DISBANDVO)

HANDLEEVENTS = (ContratMonitoringReport | OpportunityEmergence | RequirementChange).Decide

FORMVO = ContractMembers.Activate(SP,SC).Subscribe$_1$(YP,C2M)

REFORMVO = PreFormVO.[RemoveMembers].FORMVO

RECONFIGUREVO = Negociate.FORMVO

DISBANDVO = RemoveAllMembers

Safety

• HANDLEEVENTS → [StopServiceProvisionConsumption]

TAB. 4.10 – Le Schéma de Rôle *Virtual Organisation Manager (VOM)*

entreprend les tâches suivante :

1. **Formation VO** : le VOM contracte ses membres, active la consommation et/ou la fourniture des services, et s'inscrit au YP et au C2M, pour les notifications "Opportunity Emergence" (*ex*, l'arrivée d'un fournisseur, ou la disponibilité d'un client important) et "ContractMonitoringReport" (*ex*, l'annulation d'un contrat ou la mauvaise performance d'un membre).
2. **Re-Formation VO** : pareil à la formation, sauf que le VOM remplace certains membres d'une VO déjà formée. C'est une maintenance suite aux événements (1),(2) ou (3).
3. **Re-Configuration VO** : Dans cette phase de maintenance, à l'issue de l'événement (1) ou (3), le VOM redistribut les tâches et/ou les charges de travail sans changer les membres VO.
4. **Dissolution VO** : en cas d'échec de maintenance (*ex.,* l'indisponibilité d'un service nécessaire, ou le départ de tous les clients (événement (3)), le VOM dissout sa VO.

4.4.11 Le Rôle *Contract Manager and Monitor*

Nous introduisons le rôle C2M pour gérer, monitorer et imposer des contrats. Appartenant à une institution électronique, ce rôle représente une partie tière et crédible où des contrats peuvent être manipulés. Le C2M offre trois services (Table 4.11) :

1. *Gestion de contrat* : le C2M établit, met à jour et annule les contrats.

Role Schema : CONTRACT MANAGER AND MONITOR (C2M)
Description : gère, surveille et force l'application des contrats des membres VO.
Type : Institutionnel.
Super-Roles : EI-Staff member
Triggers-Roles : CM, VOM
Protocols & Activities :
ContractMembers, ResourceCommitmentsRequest,Monitor, AnswerRequest, ...
Permissions :
generates/reads/modifies contrats // tout les droits sur les contrats.
Responsabilities
Liveness
C2M = Register.WORK$^{\omega}$
Work = MANAGEMONITORCONTRAT | PROVIDERESOURCECOMMITMENTS
MANAGEMONITORCONTRACT = (ContractMember$^{nb-membres}$.(Monitor.Notify)*) | Subscribe.(Monitor.Notify)*
PROVIDERESOURCECOMMITMENTS = ResourceCommitmentsRequest.FindCommitments.AnswerRequest
Safety
• participant-authentifié ∧ participant-autorisé

TAB. 4.11 – Le Schéma de Rôle *Contract Manager & Monitor(C2M)*

2. *Monitoring de contrat :* le C2M surveille et notifie l'occurrence des événements. Par exemple, il notifie au VOM : la disponibilité d'un fournisseur/client, les obligations de sa VO, la mauvaise performance de ses partenaires.
3. *Calcul des engagements :* Le C2M calcul les engagements débiteurs et créditeurs en ressource d'un actionnaire, pour supporter le CM.

4.4.12 Cycle de vie VO

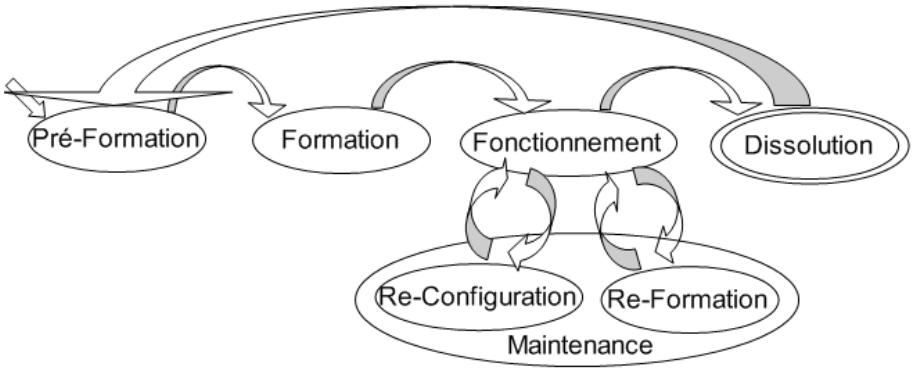

FIG. 4.8 – Cycle de Vie de L'Organisation Virtuelle

Selon notre modèle, le cycle de vie VO est schématisé par la Figure 4.8.

Le cycle de vie VO identifie les rôles institutionnels et organisationnels impliqués dans chaque phase, ainsi que les séquences d'interaction. Le diagramme de collaboration AUML (Agent Unified Modeling Language) [47], en cours de normalisation, est dédié à la spécification de la dynamique des systèmes Multi-agent. Il s'avert intéressant pour spécifier la dynamique (cycle de vie) d'une VO, le fait qu'il décrit les séquences d'interaction en fonc-

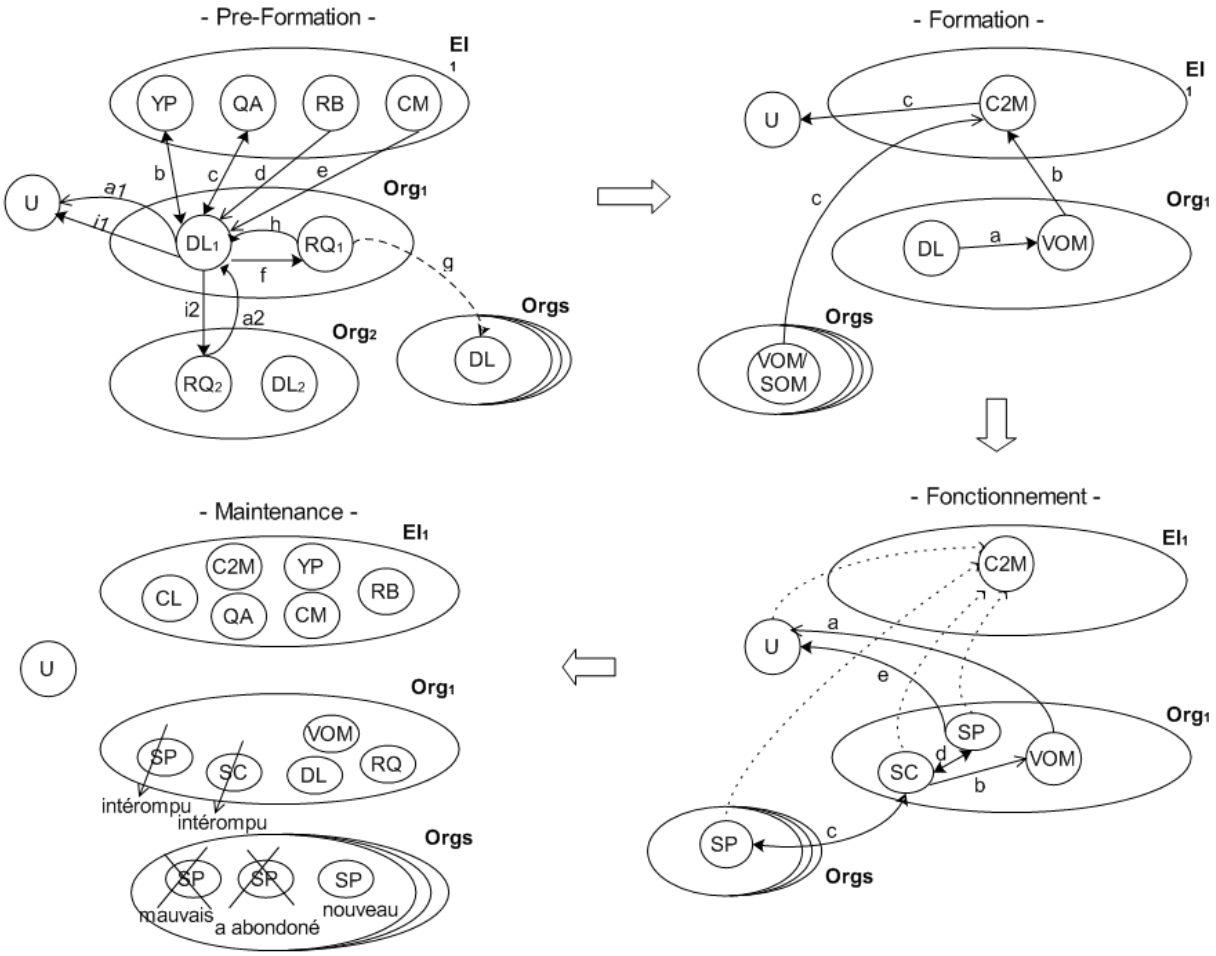

FIG. 4.9 – Les Interactions d'un Cycle de Vie VO

tion du rôle d'agent. Par souci de détail, nous nous restreindrons aux schémas informels de la Figure 4.9, qui résument les interactions dans une VO.

Pre-Formation : Déclanchée par un client (*ex.,* un individu, U, ou une organisation, Org_2 à travers RQ_2), cette phase est menée par le *Dealer* (DL_1) et est assistée par des rôles institutionnels (YP, QA, RB et CM). L'organisation sollicitée (Org_1) peut nécessiter d'autres organisations ($Orgs$). Le Requester (RQ_1) se charge de trouver une coalition en utilisant des facilités de EI_1 (YP, QA et RB). Finalement le DL_1 formule et soumet l'offre au client.

Formation : Une fois notifié par le DL, le VOM demande au $C2M$ d'établir les contrats entre clients et fournisseurs. Chaque VOM veille au bon fonctionnement de sa VO en faisant recours aux facilités de l'EI.

Fonctionnement : Pendant cette phase, les services sont demandés (par U et SC), fournies (par SP) et surveillés (par $C2M$).

Maintenance : implique tout les agents de l'EI et de la VO. Le VOM interrompe la provision et la consommation des services pour remplacer des fournisseurs ayant une

mauvaise performance ou qui ont achevé leur contrat. Le *VOM* peut reformer la VO lors d'opportunité plus profitable (nouveau fournisseur/client compétitif).

4.5 Conclusion

Dans ce Chapitre, nous avons proposé un modèle d'Organisations Virtuelles (VOs) à base de rôles. Nous avons commencé par décrire une VO sur plusieurs niveaux d'abstraction. Nous avons, ensuite, tracé la méthodologie suivie pour modéliser des VOs. Enfin, nous avons présenté notre modèle VO. Ce modèle est basée sur CONOISE-G [29] où nous avons intégré les rôles usuels : *Gestionnaire des Engagements de Ressource* et *Gestionnaire et Monitor de Contrat*. Conceptuellement, nous avons introduit l'institution électronique (EI) pour séparer entre rôles organisationnels et institutionnels, considérons ainsi l'EI un support aux interactions et un environnement normatif pour les VOs. Par ailleurs, nous avons eu affaire à tout le cycle de vie d'une VO. Notre modèle adopte une approche multi-agent basée rôle, pour mettre au point des VOs qui échangent des services et partagent des ressources. Ce modèle réduit la complexité d'une VO en identifiant et en formalisant sa structure et sa dynamique. Il sert comme fondement architectural pour un éventail de scénarios VO en e-Commerce. Cependant, il faut offrir une framework *software*, qui implémente les rôles institutionnels dans des agents et qui intègre les rôles organisationnels dans des agents— ces agent seront raffiné, par la suite, selon le cas en étude. Le Chapitre suivant est consacré à un cas d'étude afin d'illustrer et d'évaluer le modèle VO proposé.

Chapitre 5

Provision des Services Multimédia Mobiles par Construction et Exécution des Organisations Virtuelles

5.1 Introduction

Ayant déjà élaboré un modèle de VO à base de rôle, le but de ce chapitre est de l'instancier par une architecture à base d'agent, pour un scénario de e-Commerce. Mise à part l'illustration et l'évaluation du modèle VO, cette architecture nous permet de mettre en évidence l'apport à CONOISE-G [29]. Ce chapitre est organisé comme suit. Dans la Section 5.2, nous donnons l'énoncé d'un cas d'étude. Dans la Section 5.3, nous présentons *MSVO* : une architecture basée sur notre modèle VO, pour le scénario en étude. Dans les Sections 5.4 et 5.5, nous dévoilons deux nouveaux agents introduits à CONOISE-G [29] et nous montrons leur intégration à MSVO. Respectivement, il s'agit de l'agent *"Gestionnaire et Monitor de Contrat"* et de l'agent *"Gestionnaire d'Engagements de Ressource"*. Finalement, nous concluons ce chapitre par la Section 5.6.

5.2 Cas d'étude : ***Provision de Services Multimédia Mobiles*** (MSVO)

La motivation pour le système en étude, intitulé *MSVO* (*Multimédia Services Virtual Organisations*), est d'assister des usagers en séjours (touristiques ou d'affaires). Ce système propose des services multimédias *personnalisés*, *fiables* et *compétitifs* aux appareils mobiles (*ex.*, Smart Phone, PDA ou Laptop). A la base, il comprend un ensemble de fournisseurs de *services* et de *ressources*, susceptibles de coopérer dans des coalitions temporaires, selon les opportunités du marché, afin de satisfaire les clients finales. Dans ce contexte, nous distinguons des services/ressources *primaires*, comme la "bande passante

Actionnaire \Service	Téléphonie	Web Personnalisé Mobile	Web Personnalisé	Bande Passante Sans Fil
User	C	C	-	-
MMSP	F/C	F/C	C	C
MPO	F	-	-	C
MCWP	-	F	C	C
CWP	-	-	F	-
WNO	-	-	-	F/C

TAB. 5.1 – Fournisseurs, Consommateurs et Services dans *MSVO*

User	Téléphonie	Web Personnalisé Mobile
User1(à t_1)	(120 minutes, 1 mois)	("Mobile Business News", 10 m-à-j/jour, débit-moyen>=16 kbytes/s)
User1(à t_2)	-	("Mobile Business News", 10 m-à-j/jour, débit-moyen>=32 kbytes/s)
User2(à t_3)	15 jours	("Mobile Touristic Support", débit-moyen>=256 kbytes/s, prix-max=40$)

TAB. 5.2 – Exemples de Requêtes Utilisateurs dans *MSVO*

sans fil", et d'autres *composés* ou *à valeur ajoutée*, comme la "téléphonie mobile" ou le "Web personnalisé mobile " (Table 5.1).

Dans la Table 5.1, **F/C** désigne la fourniture/consommation d'un service. ***MMSP*** (*Mobile Multimedia Service Provider)* est le fournisseur des "services multimédias mobiles", ***MPO*** (*Mobile Phone Operator*) est l'opérateur de "téléphonie mobile", ***MCWP*** (*Mobile Customised Web Provider*) est le fournisseur de "Web personnalisé mobile", ***CWP*** (*Customised Web Provider*) est le fournisseur de "Web personnalisé", et enfin ***WNO***(*Wireless Network Operator*) est le fournisseur de "bande passante sans fil". Diverses coopérations existent entre ces fournisseurs. Par exemple, **MMSP** peut fournir le service "Web personnalisé mobile" à un utilisateur **User**, en le consommant à partir de **MCWP** qui, à son tour, le fourni en consommant les services "Web personnalisé" et " bande passante sans fil " à partir de **CWP** et **WNO**, respectivement. Les utilisateurs finales, **Users**, ont affaire uniquement à **MMSP** pour consommer les services de MSVO.

La Table 5.2 donne des exemples de requêtes utilisateurs dans MSVO. Le premier utilisateur est un homme d'affaires qui demande le paquet de services "téléphonie mobile" et "Mobile Business News" (Web personnalisé Mobile) à t_1, ensuite, il modifie ses besoins par la requête de t_2. Le second utilisateur est un touriste qui demande un support Web pour les sites à visiter dans une ville, à travers le service "Mobile Touristic Support".

5.3 Architecture de MSVO

Le système MSVO délivre des services aux utilisateurs finales grâce à la collaboration de plusieurs VOs. De ce fait, ce système comporte des VOs de type B2B entre des partenaires commerciaux (*ex.*, $MMSP$, MPO et WNO), et d'autres de type B2C entre fournisseurs et clients finales (*ex.*, $MMSP$ et $User$). Dans ce cas, nous identifions :

1. Un ensemble d'organisations *simples* : E_{SO}
   $E_{SO} = \{SO_1, SO_2, SO_3\} = \{WNO, MPO, CWP\}$
2. Un ensemble d'organisations *virtuelles* : E_{VO}
   $E_{VO} = \{VO_1, VO_2, VO_3, VO_4\} = \{MCWP, MPO, WNO, MMSP\}$ où :
   (a) VO_1=alliance{CWP,WNO}=$MCWP$
   (b) VO_2=alliance{MPO,WNO}=MPO
   (c) VO_3=coalition{WNO,WNO}=WNO
   (d) VO_4=coopération{MPO,$MCWP$}=$MMSP$

Les fournisseurs CWP, MPO et WNO (Organisations Simples), fournissent des services sans contracter ailleurs. Le WNO est un cas particulier, il peut contracter d'autres WNO, c'est à dire, former une *coalition* (services de même type), VO_3, pour combler un déficit de "bande passante". MPO forme l'*alliance* (services complémentaires), VO_2, avec WNO, pour fournir des appels téléphoniques. VO_1, représentée par $MCWP$, est une *alliance* entre CWP et WNO. Quant à VO_4, représentée par $MMSP$, elle est la principale VO, c'est une *coopération* (services indépendants) entre MPO et $MCWP$, et elle offre les services "téléphonie" et "Web personnalisé mobile" à l'utilisateur $User$. La Figure 5.1 résume les dépendances entre les actionnaires de MSVO. La relation"*uses*" signifie la possibilité d'utiliser des services d'un l'actionnaire, tandis que "*sub*" signifie l'habilité de générer les services d'un actionnaire. Par exemple, MPO disposant de la ressource "Bande Passante sans Fil" peut jouer le rôle WNO à son compte, au compte d'un autre MPO ou d'un $MCWP$.

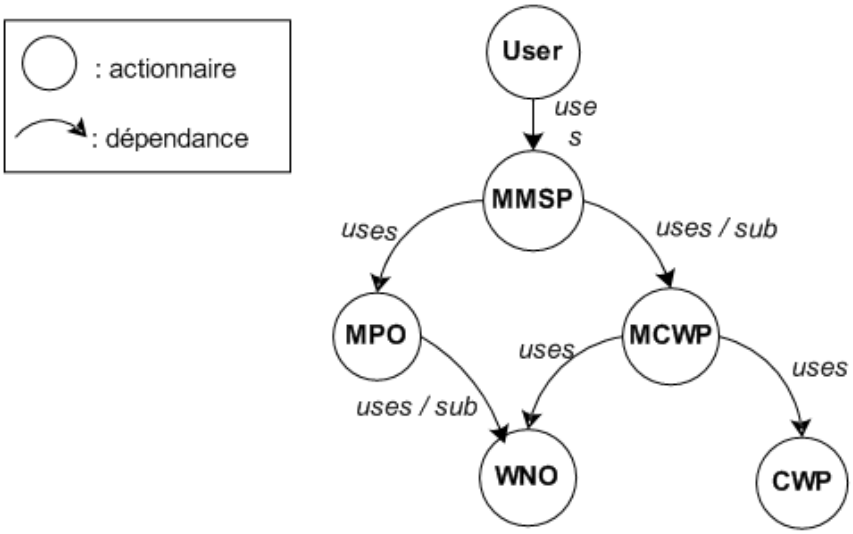

FIG. 5.1 – Graphe de Dépendances entre Actionnaires dans MSVO

Nous identifions une seule institution électronique (EI), qui supporte et réglemente MSVO. L'architecture MSVO est composée d'agents *organisationnels* et d'autres *institutionnels*. Les premiers sont les membres fournissant les services de base des VOs, tandis que les seconds offrent les services de l'EI.

5.3.1 Agents Organisationnels

Chaque actionnaire, susceptible de former et/ou joindre une VO, intervient selon l'un des rôles : *VOM*, *DL, RQ, SP* et *SC*, à la fois. Un agent peut regrouper plusieurs rôles, néanmoins, dans ce qui suit nous considérons l'attribution : "un rôle par agent".

1. *VOM Agent :* gère et maintient le cycle de vie d'une VO.
2. *DL Agent :* traque des opportunités d'affaires.
3. *RQ Agent :* trouve l'ensemble des fournisseurs appropriés face aux besoins de l'organisation en termes de ressources et/ou services.
4. *SP Agent :* fournit des services et partage/offre des ressources dans une VO.
5. *SC Agent :* consomme des services et/ou des ressources dans une VO.

Dans le cas particulier d'une *organisation simple*, l'agent *RQ* n'est pas présent, et l'agent *VOM* est dit *Single Organisation Manager (SOM).*

5.3.2 Agents Institutionnels

L'institution électronique (EI) offrent des services aux membres VO à travers des agents. Ces agents institutionnels sont classés dans deux catégories : *support* ou *régulation.*

1. **Agents de Support :** offrent des facilités aux agents membres de VO.
   (a) *Yellow Pages (YP) Agent :* permet la découverte de services, à travers un *matchmaking* entre fournisseurs et consommateurs.
   (b) *Quality Advisor (QA) Agent :* analyse les QoS fournis afin de conseiller des agents sur les services des fournisseurs.
   (c) *Commitment Manager (CM) Agent :* Gère efficacement les ressources d'une VO. Il donne les combinaisons d'allocation optimales, étant donnés les besoins et les engagements sur les ressources.
   (d) *Clearing (CL) Agent :* trouve la coalition de fournisseurs optimale face à une demande, à partir des offres.
2. **Agents de Régulation :** Ses agents de l'EI font partie d'un environnement normatif, qui permet d'établir des contrats VO et impose des *normes*, pour maintenir un minimum d'ordre.
   (a) *Contract Manager and Monitor (*C2M*) Agent :* offre une framework contractuelle qui supporte des agents pour gérer, surveiller et imposer des contrats.
   (b) *Reputation Broker (RB) Agent :* renseigne les agents sur la réputation des clients et fournisseurs, obtenue à partir de surveillance des contrats passés.

La Figure 5.2 décrit ces agents et leur dynamique dans *MSVO.*

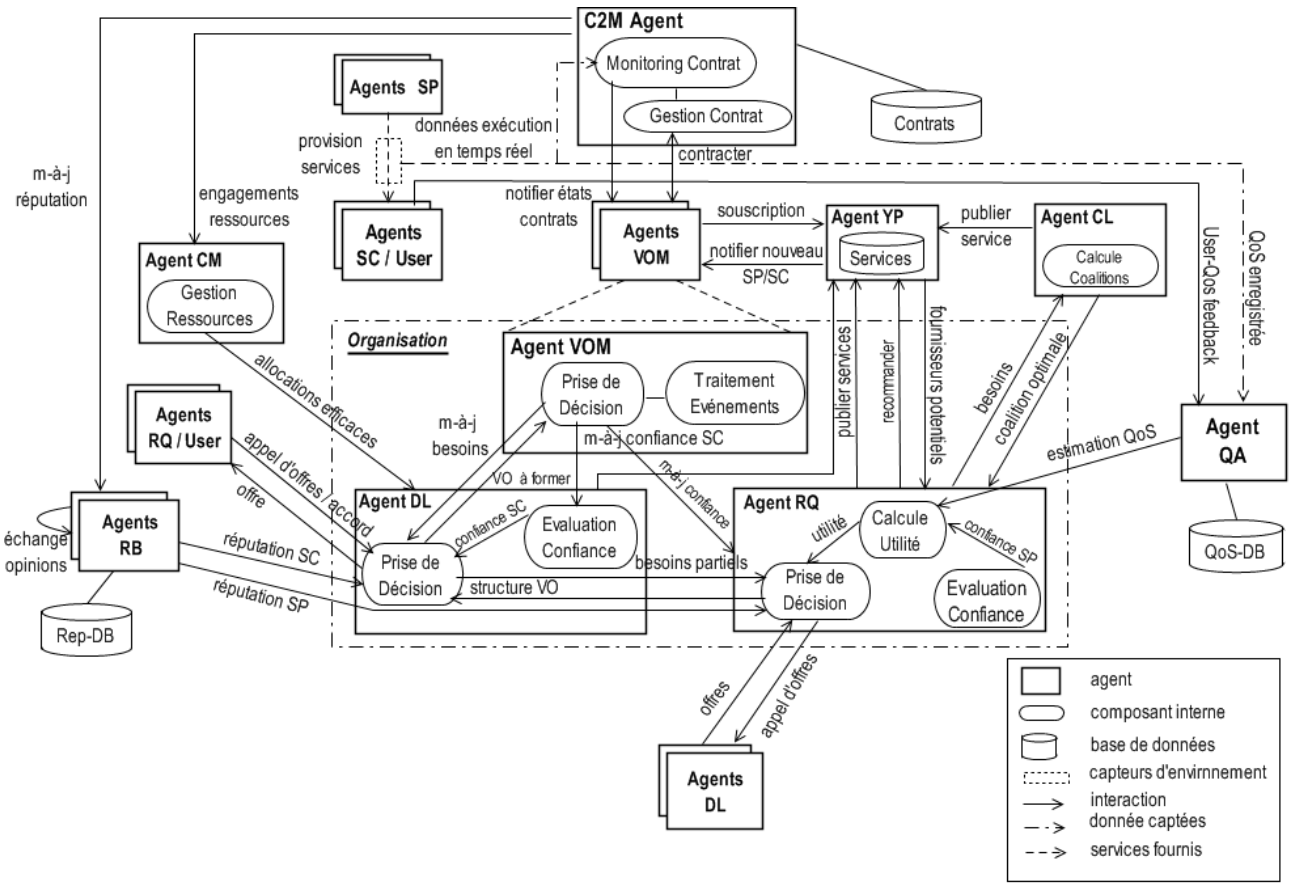

FIG. 5.2 – Architecture MSVO

5.3.3 Dynamique de MSVO

Les agents de *MSVO* s'organisent dans des VOs supportées et réglementées par une EI. Nous donnons un aperçu sur le fonctionnement de MSVO par un scénario (Figure 5.3).

Phase de Formation VO

Dans VO_1 (Figure 5.3), l'agent *DL* (Figure 5.2) de $MMSP_1$ traite les demandes : il filtre les bons clients en termes de crédibilité/réputation (composant "Evaluation Confiance"/ agent *RB*). Le *DL* délègue à l'agent *RQ* la recherche des meilleurs fournisseurs. Le *RQ* lance un appel d'offres aux fournisseurs potentiels (*ex.,* communiqués par un agent *YP*). A titre d'exemple (Table 5.2, Section 5.2), il peut s'agir de trouver les fournisseurs de "téléphonie mobile" et de "Mobile Business News". C'est les agents *DL* (Figure 5.2) de *MPO* et *MCWP* (Figure 5.3) qui répondent aux appels d'offres et optent — ou se voient obligés — de former d'autres VOs, afin de formuler leurs offres. Par la suite, le *RQ* de *MMSP* calcule les utilités des offres reçues en combinant crédibilité et estimation de QoS obtenues de l'agent institutionnel *QA*. Afin de trouver la meilleure coalition de services/fournisseurs, le *RQ* transmet au *CL*, les besoins, les offres, les prix et les utilités. La coalition obtenue est transmise par le *RQ* au *DL* (de *MMSP*), pour formuler et soumissionner l'offre. Une

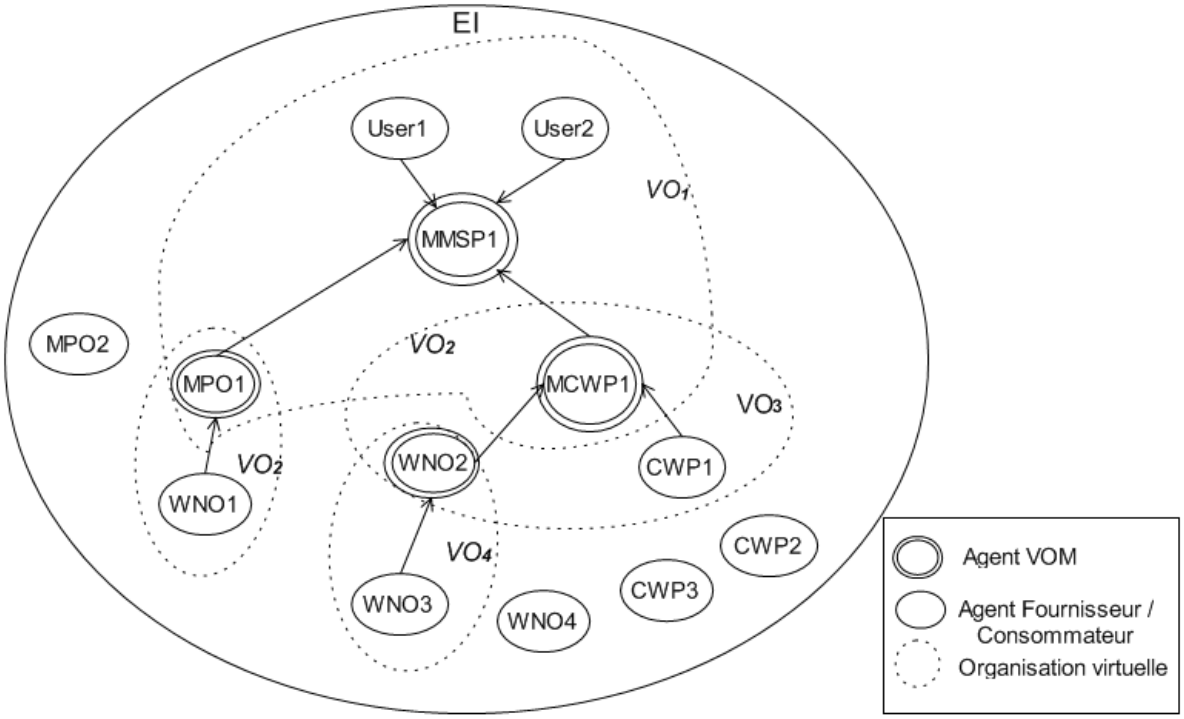

FIG. 5.3 – Un Scénario de *MSVO*

fois que l'utilisateur confirme, le *DL* notifit le *VOM* de $MMSP$ qui forme, effectivement, la VO regroupant le client et les fournisseurs. Cela revient à demander l'établissement des contrats à l'agent institutionnel *C2M* et initialiser les agents organnisationnels *SP* et *SC* de $MMSP$.

Il est possible qu'un service indispensable soit *non disponible* ou *non profitable*. Par exemple : le service "Mobile Business News" (Table 5.1, Section 5.2). En réponse dans MSVO, les agents peuvent incorporer plusieurs rôles pour s'adapter à de telles situations. En revenant à notre scénario, le rôle $MCWP$ est un sous-rôle de $MMSP$ (Figure 5.1 Section 5.3). Ainsi, le *VOM* de $MMSP$ va initier un autre *VOM* (sous gérant de VO) pour former une VO de type $MCWP$, afin de s'acquérir le service "Mobile Business News".

Phase de Fonctionnement VO

Commence dès la mise en vigueur des contrats. Dans cette phase, le *VOM* maintient la viabilité de la VO, tandis que le *C2M* monitor et impose les contrats. Principalement, les contrats contiennent des engagements de type *SLA* (*Service Level Agreement*), comme le respect de 10 m-à-j/jour à un débit moyen de 16 kbytes/s, pour le service "Mobile Business News" par $MCWP_1$ (Table 5.2 Section 5.2). Les engagements des consommateurs consistent, généralement, à respecter les délais de payement. Le *VOM* surveille des évènements comme le changement des besoins client (*ex.*, l'augmentation de la bande passante pour $User1$), ou l'arrivée d'un nouveau client (*ex.,* $User2$ qui demande le nouveau service "Mobile Touristic Support"). Par ailleurs, étant donné la compétitivité du marché et la

pro-activité de l'agent *VOM*, ce dernier traque les fournisseurs compétitifs. En outre, le *VOM* ajuste les propriétés de service des agents *SPs* et *SCs*, selon l'état de contrat.

Phase de Maintenance VO

Dans un environnement VO ouvert, dynamique et compétitif, les exceptions et les erreurs sont une règle plutôt qu'une exception. La perturbation des VOs, par conséquent la maintenance, sont inévitables. Trois mécanismes sont prévus à cet effet dans *MSVO*, en l'occurrence, "Monitoring", "Publish/Notifiy" et "Traitement d'évènements" ; ils sont dotés aux agents *C2M, YP* et *VOM* (Figure 5.2). Nous énumérons les situations suivantes :

1. ***Violation ou rupture d'un contrat :*** l'agent *C2M* scrute, régulièrement, les fournisseurs via les capteurs de l'environnement et son composant "Monitoring Contrat". Il vérifi la conformité des services fournis à ceux promis, et notifie à l'agent *VOM* concerné. Le *C2M* notifie, en plus, la mise à terme d'un contrat. La diffusion des "Mobile Business News" à raison de 8 m-à-j/jour aulieu 10 m-à-j/jour, est un exemple de non respect de contrat (Table 5.2 Section 5.2).
2. ***Changement des besoins :*** L'agent *DL* est fréquemment sollicité par de nouveaux clients et pour des services nouveaux, ou par des clients qui altèrent leurs besoins. Le *DL* notifie le *VOM* pour s'accommoder les besoins nouveaux, en *re-formant* ou en *re-configurant* sa VO. Par exemple (Table 5.2, Figure 5.2 et 5.3), le *DL* de $MMSP_1$ est sollicité par le nouveau client $User_2$ à t_2 (en phase de fonctionnement), qui demande le nouveau service "Mobile Touristic Support". En même temps, l'ancien client, $User_1$, demande une augmentation de la "bande passante sans fil" à 32 kbytes/s pour le service "Mobile Business News".
3. ***Emergence d'opportunités :*** L'agent *VOM* recherche, constamment, des fournisseurs meilleurs. En e-commerce B2B, en particulier, le *VOM* doit, en plus, traquer les consommateurs importants. Pour cela, il est assisté par les agents *YP* et *C2M*. Le *VOM* fait un *subscribe* au *YP* pour certains services. Le *YP*, à travers le "subscribe/publish/notifiy", surveil les services publiés (*publish*) et les notifi (*notify*) aux intéressées (ayant fait un *subscribe* auparavant). De même, le *C2M* notifit au *VOM* les fournisseurs/consommateurs devenus plus *disponibles* — lors de la libération de ressources et/ou accroissement de la capacité de servir, pour un fournisseur ; et lors de la fin de contrat, pour un consommateur—. Le *VOM* peut re-former, éventuellement, dissoudre une VO pour former une autre plus profitable. Par exemple, dans VO_3 (Figure 5.3), lors de l'arrivée de CWP_2 (service "Business News"), le *YP* informe le *VOM* de $MCWP_1$. Quoique déja engagé avec CWP_1, le *VOM* de $MCWP_1$ prospecte CWP_2 : il demande une offre et l'évalut. Un autre cas, par exemple, est lorsque le fournisseur de " bande passante sans fil", WNO_1, devient à 45% de ressources disponibles. C'est alors que le *C2M* notifie au *VOM* de $MMSP_1$. Ce *VOM*

prospecte ce fournisseur, pour un éventuel remplacement de WNO_3, soit à cause d'une mauvaise performance, soit pour combler un déficit en ressources, ou simplement pour le prix compétitif de CWP_2. De même, le *C2M* de VO_4 notifie au $MCWP_1$ la disponibilité de WNO_1, lorsque MPO_1 achève son contrat avec. Ceci permettra à $MCWP_1$ d'anticiper sa demande à WNO_1, et de rompre avec WNO_3.

En réponse à de tels événements, un agent *VOM* opère selon deux alternatives, après délibération, dans le but de maintenir sa VO :

1. ***Re-Configuration :*** consiste à renégocier les contrats des membres actuels, en ajoutant des services et/ou en ajustant les quantités de certaines ressources fournies.
2. ***Re-Formation :*** consiste à ajouter et/ou enlever un membre VO, au moins.

En guise d'illustration, considérant la requête du client $User_2$ à t_2 dans VO_2 (Table 5.2, Figure 5.3). Elle consiste à offrir un support touristique via Web, pendant 15 jours à une qualité de service en dessus de 256 kbytes/s. Sachant que ce service n'est pas disponible, le *VOM* de $MCWP_1$ va recruter CWP_2 (*ie.*, reformer VO_3). De même, il doit re-négocier son contrat avec WNO_2 (service " bande passante sans fil") en cas de déficit, voir de contracter un autre WNO, pour satisfaire la requête de $User_2$ à t_2 dans VO_3.

Phase de Dissolution VO

La perturbation d'une VO peut être irréversible (*ie.,* où maintenance est impossible), ainsi elle sera dissoute par l'agent *VOM*. Cela revient à annuler les contrats des fournisseurs et consommateurs, au niveau de l'agent *C2M*, qui va en plus notifier aux parties concernées les obligations en instance, ainsi que celle qui découlent de la dissolution (*ex.*, des payements). En même temps l'agent *VOM* va interrompre ses agents *SP* et *SC*, et reste en *"stand by"* pour des marchés futurs.

5.4 Support de VO dans MSVO

Le besoin dynamique en ressources est à la base des VOs. Le support de VO est fait par les agents de l'EI, comme le *QA* pour le conseil en QoS ou le *YP* pour la découverte des services. Nous introduisons l'agent institutionnel *CM*(*Commitments Manager*), pour le *support* de la "gestion des engagements de ressource" dans MSVO.

5.4.1 Agent Gestionnaire d'Engagements de Ressource

L'agent *CM* (Figure 5.4) comporte un composant "Formulateur-CSP" pour définir le problème d'allocation de ressource, qui est un *CSP (Constraint Satisfaction Problem)* [8]. Le composant "Solutionneur-CSP" sert à résoudre un CSP. Suite à la demande d'un agent *RQ* client, le *DL* sollicite le *CM* pour trouver une allocation optimale. Il accompagne

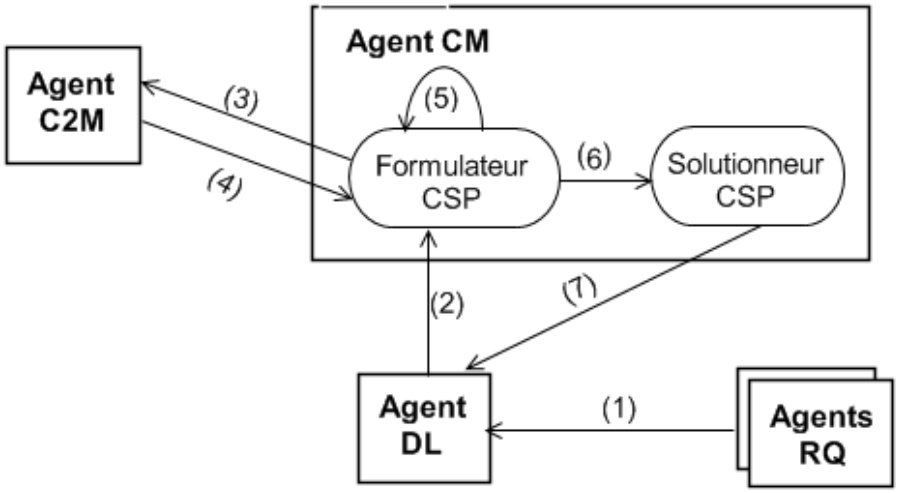

FIG. 5.4 – Agent *"Gestionnaire Engagements" (CM)* de MSVO

sa requête par ses besoins et ressources individuelles. Cependant, mis à part celles individuelles, les ressources à la disposition d'un actionnaire peuvent être *collectives* (*ie.,* provenant de VOs où il est membre consommateur). En outre, ces ressources peuvent être déjà allouées dans des VOs (*ie.,* où le membre est un fournisseur). En conséquence, l'agent *CM* demande au *C2M* les engagements *créditeurs* et *débiteurs* du *DL* requérant, respectivement, pour calculer les ressources collectives et les allocations actives (engagements). Par la suite, le *CM* formule le CSP d'allocation à partir de ces ressources et des besoins. La résolution d'un CSP donnent les meilleures solutions d'allocations – si elles existent–, qui permettent à l'agent *DL*, une fois remises, de soumettre son offre à *RQ*.

5.4.2 Gestion des Engagements et Ressources dans MSVO

Afin d'illustrer la dynamique de l'agent *CM* au sein de l'architecture MSVO, considérant le scénario précédant (Section 5.2). Supposons que l'utilisateur $User_1$ (Figure 5.3) demande une augmentation de bande passante de 16 kbytes/s à 32 kbytes/s (Table 5.2). Cette requête sera routée de $MMSP_1$ (VO_1) à WNO_2 (VO_4), en passant par $MCWP_1$ (VO_3). Supposons aussi qu'à ce moment VO_4 inclut WNO_2 et WNO_3, VO_2 inclut MPO_1 et WNO_1. Les engagement de ressource sont représentés par la Figure 5.5.

La ressource en question est la "Bande Passante sans fil", dénotée par BW, et les engagements dessus sont dénotés par ci. Chaque agent peut fournir une certaine quantité de BW (2048 unités pour WNO_1, 1024 unités pour WNO_2 et 512 unités pour WNO_3). D'après les Figures 5.3 et 5.5, les ressources de WNO_1 sont *individuelles* et sont fournies dans VO_2. Les ressource de WNO_3 sont *individuelles*. Les ressource à la disposition de WNO_2 sont *individuelles* et *collectives* (partagées par WNO_3 dans VO_4) et sont fournies dans VO_3. Les engagements sur ces ressources sont chronologiquement ordonnés comme

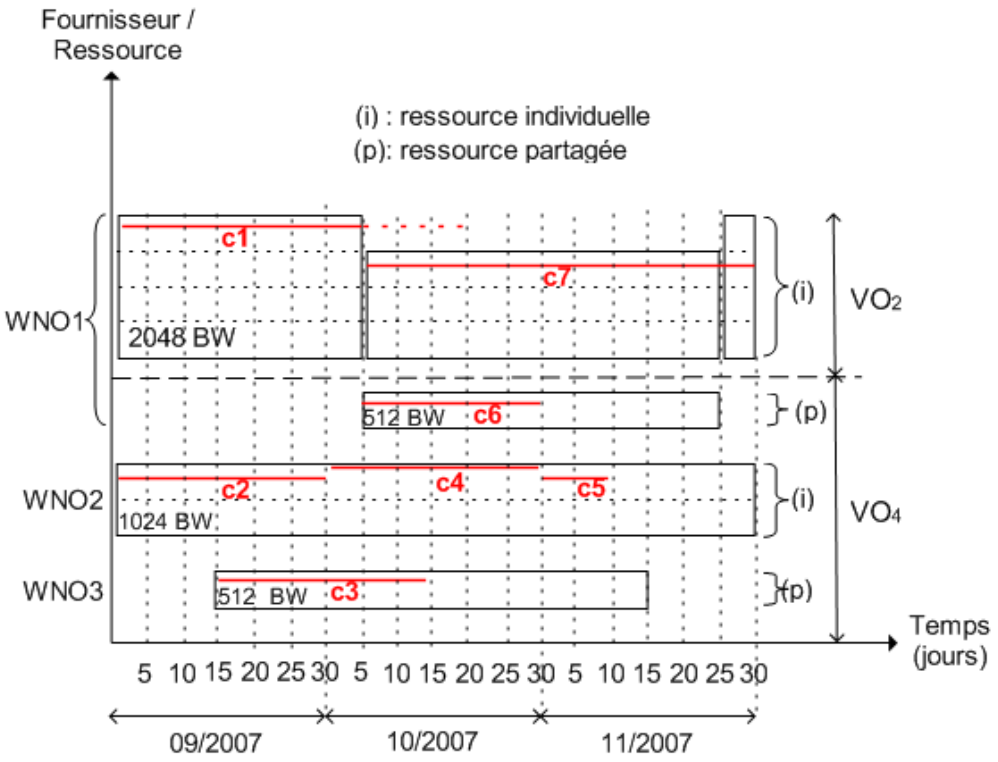

FIG. 5.5 – Etat des Engagements de Ressources des WNO dans MSVO

suit (Figure 5.5) :

- ***c1*** : $2040BW$ pendant la période $01/09/07 \rightarrow 04/10/07$ sur WNO_1
- ***c2*** : $1020BW$ pendant la période $01/09/07 \rightarrow 30/09/07$ sur WNO_2
- ***c3*** : $0504BW$ pendant la période $15/09/07 \rightarrow 15/10/07$ sur WNO_3
- ***c4*** : $1024BW$ pendant la période $01/10/07 \rightarrow 30/10/07$ sur WNO_2
- ***c5*** : $1020BW$ pendant la période $01/11/07 \rightarrow 10/11/07$ sur WNO_2
- ***c6*** : $0016BW$ pendant la période $06/10/07 \rightarrow 30/10/07$ sur WNO_1
- ***c7*** : $1500BW$ pendant la période $05/10/07 \rightarrow 25/11/07$ sur WNO_1

La nouvelle requête de $User_1$, $r1$, au niveau de WNO_2 est représentée comme suit :

- ***r1*** : $16BW$ pendant la période $04/10/07 \rightarrow 30/10/07$ pour $MCWP_1$

Face à une telle requête (datée le 3/05/2007 à 10 : 00), le processus suivant aura lieu :

1. Au niveau des ressources *individuelles* :
   - L'agent *DL* du fournisseur WNO_2 transmet, à l'agent *CM*, sa capacité totale en *BW* (qui est égale à 1024 kbytes/s).
   - L'agent *CM* demande à l'agent *C2M* tout les engagements **débiteurs** en *BW* de l'agent WNO_2. Le résultat d'après la Figure 5.5 est dans ***c4***, qui peut être

représentée comme une *contrainte* selon la syntaxe de [36] :

$$(\forall\ ?t \in Time)\quad ?t \geq 1/10/07\ \wedge\ ?t < 1/11/07\ \implies\ (\exists\ c \in commitment)$$
$$hasService(?c, ?s)\ \wedge\ hasServiceType(?s,'BW')\ \wedge\ hasAmount(?s, 1024)$$

C'est à dire que WNO_2 a au moins un engagement comme fournisseur de service de ressource BW (individuelle), dont la quantité totale allouée est 1024 unités, pendant la période : 1/10/2007 au 30/11/2007. Ces consommateurs peuvent être des $MCWP$, MPO ou d'autres WNO (*ex.*, $MCWP_1$ dans VO_3).

2. Au niveau des ressources *mutualisées* :
   WNO_2 dispose des ressources partagées par WNO_3 dans VO_4. Par conséquent :
   - L'agent *CM* demande tous les engagements des fournisseurs *créditeurs* de WNO_2. D'après la Figure 5.5, il s'agit seulement de la totalité de la ressource BW de WNO_3 (512 kbyte/s) entre le 15/9/07 et le 15/11/07.
   - L'agent *CM* demande au *C2M* tous les engagements, sur ressources collectives, *débiteurs* aux clients de WNO_2. Dans la Figure 5.5, au moment de la requête de $User_1$, il s'agit de ***c3***, qui signifie que WNO_3 a au moins un engagement pour fournir la quantité 504 de sa ressource BW, entre le 15/9/07 et le 15/10/07, au compte de VO_4 (*ie.,* aux clients de WNO_2, comme $MCWP_1$).
3. L'agent *CM* formule un CSP à partir de ***r1, c3*** et ***c4*** dont il transmet la résolution à l'agent *DL* de WNO_2. Ce dernier constate l'impossibilité d'allouer sans violer la contrainte ***c3*** ou ***c4***. Par conséquent, ce *DL* agit comme suit :
   - Il ignore la requête de son client (l'agent *RQ* de $MCWP_1$) ; ou
   - Il formule son offre et enfreint (à priori) ses engagements (***c3*** et/ou ***c4***) ; ou
   - Dans le meilleur cas, Il contracte d'autres fournisseurs, si possible.

   Dans le dernier cas, supposons que le *C2M* notifit WNO_2 de la disponibilté de WNO_1 à t égale au 5/10/07 (Figure 5.5). Après, le *VOM* de WNO_2 établit un contrat de coopération avec WNO_1 pour fournir 512 kbytes/s de BW du 5/10/07 au 25/11/07. Ainsi en résolvant le CSP à nouveau, deux coalitions peuvent satisfaire ***r1*** : $\{WNO_1\}$ ou $\{WNO_1, WNO_3\}$. En optant pour la première option, un engagement débiteur, ***c6***, est mis sur la ressource partagée BW de WNO_1 dans VO_4.

   • ***c6*** : $16BW$ pendant la période 05/10/07 → 30/10/07 sur WNO_1 dans VO_4.

Nous nous somme basé sur le travail de [37] ("*Un Service de Gestion des Engagements Réutilisable basé Web Sémantique*"), pour élaborer l'agent *CM*. Cet agent est introduit comme facilité *institutionnelle* commune aux VOs, pour de meilleurs performances. En réalité, le rôle *CM* est intégré à l'agent *SP* dans CONOISE-G [29], ainsi notre séparation agent-rôle et organisation-institution, simplifie ce système. Précisément, nous avons factorisé le problème de gestion des ressources sur trois rôles (*DL*, *CM* et *C2M*), où nous

avons identifié la séquence d'interactions intra-institution (entre *CM* et *C2M*), intra-organisation (*DL* et *RQ*) et inter institution-organisation (entre *DL* et *CM*). Ainsi, nous permettons une résolution performante et scalable.

5.5 Régulation de VO dans MSVO

5.5.1 Framework Normative

Nous adoptons l'approche qui considère l'EI comme tièr crédible qui définit et force l'application des normes, en admettons leur violation [6]. D'où, l'EI définit des sanctions et monitor l'exécution des contrats. Pour maintenir l'ordre dans un système, l'EI est supportée par une *framework normative* composée d'un *environnement normatif* et d'un *modèle de contrat*. Le premier surveille la provision de services : il capte les événements de l'environnement pour construire une *réalité institutionnelle* qui interprète ces évènements par rapport à leurs contextes (contrats construits à partir du *modèle de contrat*).

Spécification d'une Norme

Une *norme* est une règle qui spécifie le comportement agent souhaité, lorsqu'une certaine situation se produit. Nous utilisons la représentations suivante [6] :

`[Context] Situation` $\longrightarrow$ `Prescription`

Un *contexte* est le cadre d'application d'une norme (*ex.,* l'environnement normatif ou un contrat). La *situation* est une condition qui spécifie le "Quand", tandis que la *prescription* spécifie un "élément de la réalité institutionnelle" (*ex.,* une obligation).

Réalité Institutionnelle

La *Réalité Institutionnelle (RI)* contient des événements enregistrés dans l'environnement (*ex.,* actions d'agents, passage du temps) et des faits générées par des normes. Nous ajoutons aux *éléments de la RI (IRE)* [6], l'événement *iaction* (Table 5.3).

Elément RI	Structure	Signification
Fait Institutionnel	`ifact(<IFact>,<Timestamp>)`	Le fait `<IFact>` a eu lieu au temps `<Timestamp>`.
Obligation	`obligation(<Agent>,<IFact>,<Deadline>)`	L'agent `<Agent>` doit réaliser `<IFact>` avant l'écoulement de `<Deadline>`
Accomplissement	`fulfilled(<Agent>,<IFact>,<Timestamp>)`	<Agent> a accompli <IFact> à <Timestamp>.
Violation	`violated(<Agent>,<IFact>,<Timestamp>)`	<Agent> n'a pas réalisé <IFact> à <Timestamp>.
Action Institutionnelle	`iaction(<IFact>,<Timestamp>)`	L'EI va effectuer l'action qui réalise <IFact> à <Timestamp>.
Temps	`time(<Timestamp>)`	L'horloge système indique <Timestamp>.

TAB. 5.3 – Eléments de la Réalité Institutionnelle (IRE)

Ces éléments (IRE) composent une norme, comme ils peuvent être générés par.

Modèle de Contrat

Les contrats formalisent des engagements de coopération entre agents et spécifient l'ensemble des normes applicables. Une fois intégré à l'environnement normatif d'une EI, un contrat est monitoré et imposé. C'est pourquoi un contrat doit contenir des informations d'*usage* et d'autres de *monitoring*. Nous adaptons le modèle de contrat, *e-contract* [7], comme suit :

- **Section Entête** : contient le minimum d'informations pour rédiger un contrat :
  1. *Type* : classe de contrat pour un certain domaine d'usage.
  2. *ID* : numéro unique de l'instance contrat.
  3. *Super-Context* : le contexte dans lequel ce contrat est créé. Ca peut être un contrat parent, ou bien l'environnement normatif de l'EI, par défaut.
  4. *When* : date d'établissement du contrat.
  5. *Starting-Situation* : condition de mise en vigueur (pas forcement *When*).
  6. *Ending-Situation* : condition d'achèvement du contrat, à priori (*ex.,* une date).
  7. *Contractor* : la partie qui demande l'établissement du contrat.
  8. *Contractee* : la partie contractée.
  9. *Provider* : un fournisseur dans le contrat (Par défaut, le *contractor*).
  10. *Consumer* : un consommateur dans le contrat (Par défaut, le *contractee*).
  11. *Contractual Informations* : informations complémentaires sur les services.
- **Section Règles** : ensemble des règles sur les *faits institutionnels* (*ex.,* la règle d'équivalence pour résoudre une hétérogénéité sémantique).
- **Section Normes** : ensemble des normes spécifiques au contrat. Les *normes institutionnelle* seront héritées et appliquées dans ce contrat, tand qu'elles non pas été surchargées par ses normes spécifiques.

5.5.2 Agent Gestionnaire et Monitor de Contrat

Les tâches et interactions de l'agent *C2M* sont (Figure 5.6) :

- Gestion de contrat : établissement, mise à jour et achèvement.
- Monitoring de l'exécution des contrats : capter des événements de l'environnement comme faits bruts, ensuite les interpréter dans des contextes (contrats), pour mettre à jours la réalité institutionnelle.
- Notifications des états de contrats débiteurs : obligations actives, sanctions et récompenses.
- Notifications des états de contrats créditeurs : comme le non respect d'un SLA.
- Calcul des engagements d'un fournisseur pour supporter l'agent *CM*.

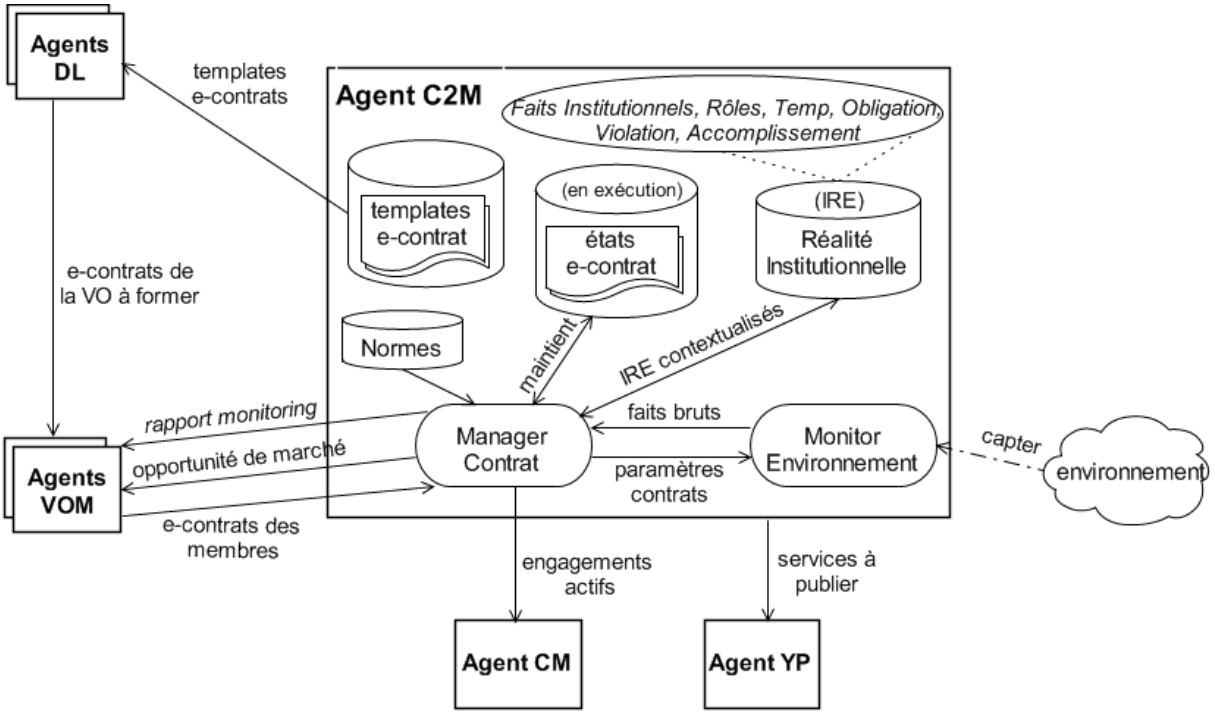

FIG. 5.6 – Agent *"Gestionnaire et Monitor de Contrat"(C2M)* de MSVO

Nous revenons à notre cas d'étude (Section 5.2, Figures 5.2 et 5.3), afin de décrire le fonctionnement de l'agent *"Gestionnaire et Monitor de Contrat"* (C2M) .

5.5.3 Etablissement des Contrats

L'agent *DL* utilise les motifs de contrat (*templates*) mis à disposition par l'agent *C2M*, pour conclure des accords (Figure 5.6). Une fois négocié, l'agent *DL* transmet un contrat à l'agent *VOM*, qui va l'établir officielement au niveau de l'agent *C2M*. Par exemple (Figure 5.3), la requête de $User_1$ à t_1 (Table 5.2) implique les contrats suivants dans VO_1.

[EI_1] ifact(contract("MMSP-User-Contrat", vo_1-cont01, EI_1, 1/10/07, 30/10/07, $MMSP_1$-VOMagent, USERagent01, service-paquet(service("appel-tel",60 mn/jour,3$), service("Mobile-Business-News",10 m-à-j/jour,≥16 kbytes/s,20$)), 27/9/07) .....(1)

[EI_1] ifact(contract("MCWP-MMSP-Contrat", vo_1-cont02, EI_1, 1/10/07, 30/10/07, MMSP1-VOMagent, MCWP1-VOMagent, service("Mobile-Business-News",10 m-à-j/jour,≥ 16 kbytes/s,14$,$MMSP_1$-SCagent), 28/9/07) .....(2)

Dans VO_3, le contrat suivant peut être établit pour satisfaire la requête précédante :

[EI_1] ifact(contract("MCWP-CWP-Contrat", vo_3-cont01, EI_1, "lors-1ère-requête", "1-mois-après", MCWP1-VOMagent, CWP1-SOMagent, service("Business-News",10 m-à-j/jour,5$, MCWP1-SCagent), 30/9/07) ...(3)

Dans VO_4, c'est un accord de coopération qui a été déja établit entre les $WNOs$:

[EI_1] ifact(cooperation-agreement(VO_4-ca001, participants(WNO2-VOMagent, WNO3-VOMagent), Resources(''Bande Passante Sans Fil'') .....(4)

5.5.4 Gestion et Monitoing des Contrats

Le contrat (1) à été établit le 27/09/07 et s'active le 1/10/07. Dans un premier lieu, le C2M surveille la condition d'activation, ensuite, il calcule et transmet les obligations actives. Par exemple, l'agent *C2M* à t égale au 1/10/07 active le contrat, génère et transmet l'obligation de `MMSP-agent1` suivante (contrat (1)) :

[EI_1 :VO_1-cont01] obligation(MMSP1-VOMagent, delivery(MMSP1-SPagent, service(‘‘Mobile-Business-News'',10 m-à-j/jour,≥16 kbytes/s), USERagent01), 23 :59-1/10/2007).....(5)

Vérification de l'Accomplissement des Obligations

Chaque obligation générée est mise en instance d'accomplissement dans la réalité institutionnelle (RI). L'agent *C2M* surveille l'accomplissement moyennant la RI et la norme institutionnelle suivante (Figure 5.6) :

[?Context] ifact(?IFact,?T) ∧ obligation(?agent,?IFact,?deadline) ∧ (?T <?deadline) ⟶ fullfiled(?agent,?IFact,?deadline).....(6)

En outre, l'agent *C2M* utilise des règles normatives pour générer des faits institutionnels :

[?Context] ifact(IFact1,?T) ⟶ ifact(IFact2,?T).....(7)

Par exemple, le fait institutionnel (référencé (8)) de l'obligation (5) peut être déduit à partir du fait institutionnel suivant :

[EI_1 :VO_1-cont01] ifact(delivery(MMSP1-SPagent, service(‘‘Mobile-Business-News'',10 m-à-j/jour,=18 kbytes/s), USERagent01), 20 :00-1/10/2007) .(9)

L'agent *C2M* conclut l'accomplissement de l'obligation (5) en appliquant (6) et (7) :

(9) ∧ ((9) ⟶ (8)) ∧ (20 :00-1/10/07 < 23 :59-1/10/07) ∧
(5) ⟶ fulfilled(MMSP1-VOMagent, delivery(MMSP1-SPagent1, service(‘‘Mobile-Business-News'', 10 m-à-j/jour, ≥16 kbytes/s), USERagent01))
(10)

Détection de Violation des Obligations

Le *C2M* utilise une norme institutionnelle pour détecter la violation d'une obligation :

[?Context] obligation(?agent, ?IFact, ?deadline) ∧ Time(?deadline) ∧ ¬ fullfiled(?agent, ?IFact, -) ⟶ violated(?agent, ?IFact, ?deadline) (11)

Par exemple dans VO_3, le *C2M* commence par la déduction suivante :

[EI_1 :VO_3-cont01] ifact(delivery(CWP1-SPagent, service(''Business-News'', 8 m-à-j/jour), MCWP1-SCagent), 23 :59-2/10/2007) ∧ Time(23 :59 :2-2/10/07) ⟶ ¬ fulfilled(CWP1-SOMagent1, service(''Business News'', 10 m-à-j), MCWP1-SCagent1), -) ... (12)

Ensuite, Le *C2M* applique la règle (11) pour conclure la violation de l'obligation du contrat (3) :

[EI_1 :VO_3-cont01] ¬ fulfilled(CWP1-SOMagent1, service(‘‘Business News'', 10 m-à-j), MCWP1-SCagent1), -) ∧ obligation(CWP1-SOMagent, delivery(CWP1-SPagent, service(''Business News'', 10 m-à-j/jour), MCWP1-SCagent1), 23 :59-2/10/07) ∧ Time(23 :59-2/10/07) ⟶ violated(CWP1-SOMagent1,delivery(CWP1-SPagent, service(‘‘Mobile-Business-News'', 10 m-à-j/jour), MCWP1-SCagent), 00 :00-3/10/07) ..... (13)

Application des Sanctions et Récompenses

EI_1 applique ses normes par défaut lorsqu'une obligation est enfreinte ou respectée. EI_1 applique, en plus, les normes contractuelles des parties en accord. Par exemple, la norme contractuelle appliquée lors de l'accomplissement de l'obligation (5) est :.

[EI_1 :VO_1-cont01] fulfilled(MMSP1-VOMagent1, delivery(MMSP1-SPagent1, service(''Mobile-Business-News'',10 m-à-j,16 kbytes/s), USERagent01), 23 :59-1/10/07) ⟶ obligation(USERagent01, payment(USERagent1,0.75$,MMSP1-VOMagent), 23 :59-2/10/07) (14)

Cette norme génère une recompense de paiement pour l'accomplissement de obligation (5). L'exemple suivant montre la sanction appliquée lors la violation d'obligation déduite par (13). La sanction est de recevoir que 10% du prix de prestation.

```
[EI1 :VO3-cont01]  violated(CWP1-SOMagent1, delivery(CWP1-SPagent,
                   service(''Business News'',10 m-à-j), MCWP1-SCagent1),
                   00 :05-3/10/07) ⟶ obligation(MCWP1-VOMagent1,
                   payment(MCWP1-VOMagent1, 10%×5$,CWP1-SOMagent),
                   00 :05-4/10/07) ∧ iaction(incremented(fault-counter, 1), -)
                   (15)
```

Rapport de Monitoring

L'agent *VOM* surveille la performance de ses fournisseurs en spécifiant, à l'agent *C2M*, les paramètres de service à monitorer. Par exemple, dans (15) un compteur d'erreurs (*fault-counter*) est incrémenté chaque fois qu'un SLA est non respecté. C'est l'action institutionnelle `iaction` qui fait cela. L'agent *VOM* spécifie, en plus, les actions à entreprendre pour certaines valeurs limites de ce compteur, comme c'est illustré par la norme suivante.

```
[EI1 :VO3-cont01]  ifact(equal(fault-counter, 4), ?T) ⟶
                   obligation(CWP1-SOMagent1, delivery(CWP1-SPagent1,
                   service(''Business News'', 7 m-à-j, MCWP1-SCagent),
                   anyday :23 :59) ..........................................(16)
```

L'agent *VOM* de $MCWP1$ spécifit au *C2M* de réduire la fréquence des m-à-j Web à 7 par jour, après quatre non respect du SLA "Business-News". D'autre part, le *VOM* définit une limite pour laquelle il considère un fournisseur comme mauvais. Dans ce cas, le *C2M* avertit le *VOM* par un rapport, comme c'est le cas dans l'exemple suivant :

```
[EI1 :VO3-cont01]  ifact(equal(fault-counter, 10), ?T) ⟶
                   iaction(notified(MCWP1-VOMagent1, bad-provider(CWP1-SOMagent1)),
                   -) ..................................................... (17)
```

De même, l'agent *C2M* notifie l'agent *RB* avec les m-à-j de réputation à la fin des contrats qu'il monitor.

```
[?Context]  ifact(end-contrat(idContract, ?contractor, ?contractee), ?T) ⟶
            iaction(notified(EI1-RBagent, new-reputation( ?contractor, ?contractee)),
            -).....................................................................(18)
```

L'agent *C2M* est basée sur la framework normative présentée dans [6, 5]. Cet agent décharge un actionneur d'établir, de monitorer et d'imposer ses contrats VO. De plus, il support l'agent *RB* pour maintenir la réputation. L'extraction de ce rôle de l'agent *SP* pour l'affecter à l'EI, permet de simplifier MSVO par rapport à *CONOISE-G* [29], d'avoir des performances meilleures et d'assurer l'ordre, dans un système ouvert, par une entité tière et crédible (EI).

5.6 Conclusion

Dans ce chapitre nous avons illustré notre modèle VO par un cas d'étude. Nous avons élaboré une architecture basée-agent d'un système de provision des services multimédias mobiles. Nous avons montré comment des agents opèrent et coopèrent au sein de VOs, pour fournir des services innovants de façon compétitive, fiable et efficace. A travers notre cas d'étude, nous avons mis en relief notre apport au système CONOISE-G [29] (principal système étudié sur lequel notre modèle VO est basé). Principalement, nous avons décrit comment une institution électronique, peut supporter, assister et réglementer une VO, grâce aux agents "gestionnaire des engagements de ressource" et "gestionnaire et monitor de contrat", respectivement.

Conclusions et Perspectives

Dans ce mémoire, nous avons entamé un domaine prometteur portant sur les Organisations Virtuelles (VO). Nous avons commencé par un état de l'art où nous avons mis en exergue les défis majors et posé notre problématique. Nous avons aussi passé en revue l'état de l'art des architectures orientées service (services Web/Grid) et des institutions électroniques, deux domaines connexes aux VOs. Notre travail consistait à modéliser des VOs. Notre contribution à ce thème est un modèle de VO à base de rôle. Ce modèle facilite la construction et l'exécution de VOs fiables, efficaces et compétitives, de façon dynamique, dans des environnements ouverts. Principalement, nous avons introduit l'institution électronique comme moyen de support et de régulation des VOs, comme nous avons introduit et formalisé des rôles et des interactions usuels. Finalement, nous avons étudié l'application du modèle VO proposé, à un scénario e-Commerce, où il s'agit de fournir des services multimédias mobiles. Cela nous a permis d'illustrer et d'évaluer notre modèle VO, sur la base des critères recensés lors de l'étude bibliographique.

Dans le futur, il est probable que les clients obtiennent leurs services à partir des coalitions de fournisseurs (VOs). Ces VOs doivent être fiables, scalables et doivent être, réellement, créées et maintenues dans des environnements ouverts, dynamiques et compétitifs. Effectivement, les VOs sont perçues comme : *"a way of abstracting the complexity of open systems to make them amenable to application development"* [29]. Cependant, les pratiques courantes sont *ad-hoc*—se basent sur diverses techniques : service Web, Grid ou agent, sans se référer à un modèle integré. Dans ce travail nous avons proposé un modèle de VO pour former et maintenir des VOs. Principalement, ce modèle spécifit des rôles organisationnels et institutionnels usuels. Par ailleurs, nous avons identifié et spécifié : les structures organisationnelles, les protocoles d'interactions ainsi que le cycle de vie qui caractérisent une VO. Nous avons appliqué notre modèle dans MSVO, un système de VOs à base d'agent, pour la fourniture des services multimédias mobiles.

Comme perspectives, nous identifions les directions suivantes : mettre au point une framework à base d'agents pour supporter les systèmes VO ; élaborer le modèle VO proposé en introduisant d'autres rôles utiles ; et automatiser le cycle de vie VO pour une meilleure adaptation aux environnements dynamiques et compétitifs.

Bibliographie

[1] Box, D., et al.(2004). *'Web services addressing (WS-Addressing)'*, W3C Member Submission, http ://www.w3.org/Submission/2004/SUBM-ws-addressing-20040810

[2] Janice Bum and Greg Robins, *A Virtual Organisation Model for E-Government.* Australasian Journal of Information Systems. Vol 9, No 2, 2002.

[3] Lopes Cardoso, H., Oliveira, E. *Virtual Enterprise Normative Framework within Electronic Institutions.* In Proceedings of the Fifth International Workshop Engineering Societies in the Agents World (ESAW04), Toulouse, France, 20-22 October 2004.

[4] Lopes Cardoso H, Malucelli A, Rocha AP, Oliveira E. *Institutional services for dynamic virtual organizations.* In : Camarinha-Matos LM, Afsarmanesh H, Ortiz A (eds) *Collaborative networks and their breeding environments* 6^{th} IFIP working conference on virtual enterprises (PRO-VE05). Springer, pp 521-528.

[5] Lopes Cardoso H, Oliveira E. *Assisting and regulating virtual enterprise interoperability through contracts.* In : Berre A, Elms K, Fischer K, Mueller JP (Eds) Proc. Agent-based Technologies and applications for enterprise interOPerability. The Netherlands, Utrecht 2005.

[6] Henrique Lopes Cardoso and Eugenio Oliveira. *Electronic institutions for B2B : dynamic normative environments.* Springer Science+Business Media B.V. 2007.

[7] Henrique Lopes Cardoso, Eugénio Oliveira. *A Contract Model for Electronic Institutions.* COIN Workshop in AAMAS 2007.

[8] S. Chalmers, P.M.D. Gray, and A. Preece. *Supporting Virtual Organisations using BDI Agents and Constraints.* In M. Klusch, S. Proceedings of the 6th International Workshop on Cooperative Information Agents (CIA 2002), n.2446 in Artificial Intelligence, pages 226-240, Madrid, Spain, September 2002. Springer Verlag.

[9] S. Chalmers, A. Preece, T. J. Norman, and P.M.D. Gray. *Commitment management through constraint reification.* in Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multi-Agent Systems, 2004.

[10] Czajkowski, K., Ferguson, D., Foster, I., Frey, J., Graham, S., Maguire, T., Snelling, D. and Tuecke, S.(2004) *From open grid services infrastructure to WS-resource framework : refactoring & evolution.*

[11] Viet Dung Dang. *Coalition Formation and Operation in Virtual Organisations*. PhD thesis, ECS department, Southampton University, UK. December 2004.

[12] Dang, V. D., Jennings, N. R. *Polynomial algorithms for clearing multi-unit single-item and multi-unit combinatorial auctions*. Artificial Intelligence 2004.

[13] Ferber, J., O. Gutknecht and F. Michel, *From agents to organizations : an organizational view of multiagent systems*, in : Procs. of AOSE'03, LNCS 2935 (2003), pp. 214-230.

[14] FOSTER I., KESSELMAN C., TUECKE S., *The Anatomy of the Grid : Enabling Scalable Virtual Organizations* , Supercomputer Applications, vol. 15, no 3, 2001.

[15] Foster,I., Kesselman, C., Jeffrey M.Nick, Steven, T., *The Physiology of the Grid : An Open Grid Serivices Architecture for Distributed Systems Integration*, Open Grid Service Infrastructure WG, Global Grid Forum, The Globus Alliance, June 2002.

[16] Ian Foster, Nicholas R. Jennings, Carl Kesselman. *Brain Meets Brawn : Why Grid and Agents Need Each Other*, Proceeding of the 3rd International Conference on Autonomous Agents and Multi Agent Systems, New York, USA, 2004.

[17] Josep Luis Arcos, Pablo Noriega, Juan A. Rodriguez-Aguilar, and Carles Sierra, *E4MAS Through Electronic Institutions*. in Springer-Verlag Berlin Heidelberg 2007, pp. 184-202.

[18] C. Kesselman. I. Foster.*Globus : A metacomputing infrastructure toolkit*. In International J. Supercomputer Applications, page 115. 1997.

[19] M. Kollingbaum, T. Norman, N. Mehandjiev, K. Brown. *Engineering Organisation-Oriented Software*. In WISER06, May 20, 2006, Shanghai, China.

[20] D.Laforenza, P.Ritrovato, S.Salerno et D.Talia. *Virtual Organisation and Virtual Learning Communities in ELeGI* High Performance Computing lab, ISTI-CNR.

[21] Malucelli A, Palzer D, Oliveira E (2005) *Combining ontologies and agents to help in solving the heterogeneity problem*. In : Proceedings of the international workshop on data engineering issues in e-commerce (DEEC2005). IEEE Computer Society, Tokyo, Japan, pp. 26-35

[22] S. A. McIlraith, T. C. Son, and H. Zeng. *Semantic Web Services*. IEEE Intelligent Systems, 16(2) :46-53, March 2001.

[23] Brahim Medjahed. *Semantic Web Enabled Composition of Web Services*. PhD thesis in Computer Science and Application. Virginia Tech University, USA. January 19^{th}, 2004.

[24] D.Nguyen, S.Thompson, J.Patel, L.W.T.Teacy, N.R.Jennings, M.Luck, V.Dang, S.Chalmers, N.Oren, T.J.Norman, A.Preece, P.M.D.Gray, G.Shercli, P.J.Stockreisser, J.Shao, W.A.Gray, N.J.Fiddian. *Delivering Services by Building and Running Virtual Organisations*. BT Technology Journal. Vol 24 No 1. January 2006.

[25] T. J. Norman, A. Preece, S. Chalmers, N. R. Jennings, M. Luck, V. D. Dang, T. D. Nguyen, V. Deora, J. Shao, W. A. Gray, and N. J. Fiddian. *Conoise : Agent-based formation of virtual organisations.* Research and Development in Intelligent SystemsXX : Proceedings of AI2003, the Twentythird SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, pages 353-366, 2003.

[26] T. J. Norman, A. Preece, S. Chalmers, N. R. Jennings, M. Luck, V. D. Dang, T. D. Nguyen, V. Deora, J. Shao, W. A. Gray, and N. J. Fiddian. *Agent-based formation of virtual organisations.* Knowledge Based Systems, 17 :103-111, 2004.

[27] Oliveira E, Rocha AP (2000) *Agents advanced features for negotiation in electronic commerce and virtual organisations formation process.* In : Dignum F, Sierra C (eds) *Agent mediated electronic commerce : the european agentlink perspective.* Springer, pp 78-97

[28] Oren, N. ; Norman, T. ; Preece, A. ; and Chalmers, S. *Policing virtual organizations.* In Proceedings of the 2004 European Workshop on Multi-Agent Systems (EUMAS 2004).

[29] J. Patel, W.T. Luke Teacy, N.R. Jennings, M. Luck, S. Chalmers, N. Oren, T.J. Norman, A. Preece, P.M.D. Gray : *CONOISE-G : Agent Based Virtual Organisations.* In Proceeding of AAMAS06 (ACM), pages 1460-1460. May 8-12 2006, Hakodate, Hokkaido, Japan.

[30] Petersen, S. A., Gruninger, M., 2000. *An Agent-based Model to Support the Formation of Virtual Enterprises.* In : International ICSC Symposium on Mobile Agents and Multi-agents in Virtual Organisations and E-Commerce.

[31] S.A Petersen, Jinghai Rao, Mihhail Matskin. *AGORA Multi-agent Architecture for Implementing Virtual Enterprises.* In Proceeding of CAiSE 2003 Forum, Velden, Austria, June 2003.

[32] Preece, A. D., Hui, K., Gray, A., Marti, P., Bench-Capon, T., Jones, D., Cui, Z., 2000. *The KRAFT architecture for knowledge fusion and transformation.* Knowledge-Based Systems 13 (2-3), 113-120.

[33] Alun Preece, *A Mediator-Based Infrastrcture for Virtual Organisations.* Project repport, BT and Aberdeen, Cardief and Liverpool Universities. http ://cgi.csd.abdn.ac.uk/%7Eapreece/research/download/agents2001.pdf

[34] S. Chalmers, P.M.D. Gray, and A. Preece. *Supporting Virtual Organisations Using BDI Agents and Constraints.* M. Klusch, S. Ossowski, and O. Shehory (Eds.) : CIA 2002, LNAI 2446, pp. 226-240, 2002. Springer-Verlag Berlin Heidelberg 2002.

[35] A. Preece, S. Chalmers, N. Oren, T.J. Norman, P.M.D. Gray, G. Shercliff, P.J Stockreisser, J. Shao, W.A. Gray, N.J. Fiddian, J. Patel, L. Teacy, N.R. Jennings, M.

Luck, S. Thompson, *Managing Virtual Organisations on the Grid using Agent Technology.* Agent Research Overview, In AgetLink News, Issue 17, April 2005.

[36] Alun Preece, Stuart Chalmers, Craig McKenzie, Jeff Z. Pan and Peter Gray. *Handling Soft Constraints in the Semantic Web Architecture.* In WWW2006, May 22-26, 2006, Edinburgh, UK.

[37] Alun Preece, Stuart Chalmers and Craig McKenzie. *A reusable commitment management service using Semantic Web technology.* In Knowledge-based Systems 20 (2007), 143-151. Elsevier, www.sciencedirect.com, www.elseiver.com/locate/knosys

[38] Rocha, A.P., Oliveira, E. *Electronic Institutions as a framework for Agents Negotiation and mutual Commitment.* In P. Brazdil, A. Jorge (eds.), Progress in Artificial Intelligence : Knowledge Extraction, Multi-agent Systems, Logic Programming, and Constraint Solving, LNAI 2258, Springer, pp. 232-245, 2001.

[39] J. Shao, W. A. Gray, N. J. Fiddian, V. Deora, G. Shercliff, P. J. Stockreisser, T. J. Norman, A. Preece, P. M. D. Gray, S. Chalmers, N. Oren, N. R. Jennings, M. Luck, V. D. Dang, T. D. Nguyen, J. Patel, and W. T. L. Teacy. *Supporting formation and operation of virtual organisations in a grid environment.* In Proc. UK OST e-Science 2nd All Hands Meeting (AHM04), 2004.

[40] G. Shercliff, P. Stockreisser, J. Shao, W. Gray, N. Fiddian. *Supporting qos assessment and monitoring in virtual organisations*, in : In Proceedings IEEE International Conference on Services Computing, 2005, pp. 249-250.

[41] Carles Sierra, Juan A. Rodryguez-Aguilar, Pablo Noriega, Josep Ll, Arcos Marc Esteva. *Engineering Multi-agent Systems as Electronic Institution* Research Repport, Articificial Intelligent Research Institute (IIIA), Barcelona 2004. http ://www2.iiia.csic.es/m̃arc/novatica-english.pdf

[42] Munindar P. Singh and Michael N. Huhns. *Service-Oriented Computing, Semantics, processes, agents.* John Wiley & Sons, 2005.

[43] Domenico Talia. *The Open Grid Service Architecture : Where the Grid Meets the Web.* IEEE INTERNET COMPUTING, http ://computer.org/internet/ NOVEMBER-DECEMBER 2002

[44] S. Tsur, S. Abiteboul, R. Agrawal, U. Dayal, J. Klein, and G. Weikum. *Are Web Services the Next Revolution in e-Commerce ? (Panel).* In Proceedings of the International Conference on Very Large Databases, pages 614-617, Roma, Italy, September 2001.

[45] M. Wooldrige, N. R. Jennings and D. Kinny, *The Gaia Methodology for Agent-Oriented Analysis and Design*, Autonomous Agents and Multi-Agent Systems, volume 3, pp 285-312, Kluwer Academic Publishers, The Netherlands, 2000.

[46] Zambonelli, F., N. Jennings and M.Wooldridge, *Developing multiagent systems : The Gaia methodology*, ACM Transactions of Software Engineering and Methodology Vol.12. No. 3. (July 2003), pp. 317-370.

[47] http ://www.auml.org

[48] BEA, IBM, and Microsoft. Business Process Execution Language for Web Services (BPEL4WS). http ://xml.coverpages.org/bpel4ws.html.

[49] CONOISE and CONOISE-G project Web Site. http ://www.conoise.org

[50] Site Web de Foundation for Intelligent Physical Agent. http ://www.fipa.org

[51] Spécification d'un message ACL (Agent Communication Language). http ://www.fipa.org/repository/aclspecs.html

[52] Site Web de Global Grid Forum (GGF). http ://www.ggf.org/

[53] Site Web de Globus. http ://www.globus.org/

[54] Site Web de Artificial Intelligence Research Institute (IIIA). http ://e-institutions.iiia.csic.es/

[55] Site Web de : The Open Grid Services Architecture (OGSA). http ://www.globus.org/ogsa/.

[56] Web Service Resource Framewrok (WSRF). http ://www.globus.org/wsrf/

[57] World Wide Web Consortium. http ://www.w3c.org

[58] W3C. Web Services Description Language (WSDL). http ://www.w3.org/TR/wsdl.

[59] W3C. Universal Description, Discovery, and Integration (UDDI). http ://www.uddi.org.

[60] W3C. Simple Object Access Protocol (SOAP). http ://www.w3.org/TR/soa