Coordination de négociations pour la sous-traitance dans le cadre d'alliance inter-organisationnelles

Mihnea Bratu

Coordination de négociations pour la sous-traitance dans le cadre d'alliance inter-organisationnelles

THÈSE

présentée en vue de l'obtention du titre de

Docteur de l'Ecole Nationale Supérieure des Mines

de Saint-Étienne

(Spécialité informatique)

par

Mihnea-Ben BRATU

le 23/01/2007

Composition du jury :

Président :

Jean-Marc ANDREOLI Directeur de recherche – Xerox Research Centre Europe

Rapporteurs :

René MANDIAU Professeur – Université de Valenciennes

Michel OCCELLO Professeur – Université Pierre Mendès France

Examinateur :

François JACQUENET Professeur – Université Jean Monnet – Saint Etienne

Directeur :

Olivier BOISSIER Professeur – ENSM – Saint Etienne

A ma mère

Remerciements

Je commencerai par remercier tout particulièrement Olivier BOISSIER, mon directeur de thèse, pour avoir accepté d'encadrer et de soutenir ce travail de recherche. Je le remercie également pour m'avoir accueilli dans l'équipe SMA à l'Ecole des Mines de Saint-Etienne, pour les nombreux conseils très utiles et finalement pour sa patience quand le final de ma thèse était très loin.

Ma reconnaissance va aussi à l'ensemble des membres de l'équipe SMA pour les échanges enrichissants que nous avons eus : Philipe Beaune, Laurent Vercouter, Maxime Morge, Cosmin Carabelea, Oana Bucur et Guillaume Muller.

Je remercie vivement Jean-Marc Andreoli et Stefania Castellani pour leurs conseils éclairés et leurs encouragements.

Je souhaite remercier René Mandiau et Michel Occello pour avoir accepté d'être mes rapporteurs. Je remercie Francois Jacquenet et encore Jean-Marc Andreoli pour avoir accepté d'être membres du jury.

Je remercie également tous les membres du projet E-Alliance financé par la Région Rhône-Alpes, ainsi que la Région Rhône-Alpes pour la bourse de thèse accordée qui m'a permis de finaliser mes recherches.

Mes derniers remerciements iront à ma famille et principalement à mon épouse sans laquelle je n'aurais pas pu faire tout ça.

Résumé

Les travaux de recherche présentés dans cette thèse visent à proposer un modèle de coordination de négociations déroulées dans le cadre d'une alliance d'entreprises.
Notre recherche se place ainsi dans les domaines des alliances virtuelles, de la négociation automatique et de la coordination. Dans ce cadre large, nous présentons et défendons la thèse selon laquelle un processus de négociation peut être décomposé en trois processus distincts : décision, coordination et communication.
Une autre dimension de la contribution scientifique de cette thèse est de poser les fondements d'un modèle de coordination générique, décentralisé et flexible. Ce modèle sépare le processus de coordination des deux autres en faisant que le processus de coordination est indépendant de la stratégie de négociation ou du protocole de négociation, s'adaptant donc à différents types de négociations.

Abstract

The research work presented in this thesis aim at proposing a model of coordination of negotiations within the framework of an alliance of companies.
Our research is placed thus in the fields of virtual alliances, of automatic negotiation and of coordination. Within these broad frameworks, we present and defend the thesis according to which the process of negotiation can be broken up into three distinct processes: *decision*, *coordination* and *communication*.
Another dimension of the scientific contribution of this thesis is to establish the bases of a generic, decentralized and flexible coordination model. This model separates the process of coordination from the others two, thus making the process of coordination independent of the negotiation strategy or of the negotiation protocol and adaptable to various types of negotiations.

I. Introduction

Contexte et Motivations

Ces dernières années le développement d'Internet a fortement modifié nos habitudes de vie, nos habitudes de travail. Le terme « *Internet* » est un terme générique qui englobe un grand nombre de technologies allant de l'infrastructure de communication jusqu'à ce nouveau médium d'interaction que constitue le World Wide Web.

Plusieurs travaux menés au cours des années 90 ont montré que l'Internet réduit les contraintes géographiques mais aussi temporelles dans l'organisation du travail, dans les organisations industrielles, etc. Les travaux sur les collecticiels (Computer Supported Cooperative Work), par exemple, y ont contribué via des outils tels que messagerie électronique, réunion électronique, outils de planification, management de projets, conférences audio et vidéo. D'autres travaux [Malone'97] ont montré que l'Internet peut non seulement réduire les coûts de communication mais aussi ceux liés aux transactions. Les organisations industrielles, administrations, … ont ainsi été naturellement conduites à vouloir utiliser ce médium pour coordonner leurs productions. Aujourd'hui, plusieurs types de transactions sont réalisées sous forme électronique via l'Internet : achats ou ventes de produits, sites web d'entreprises ou d'administrations dédiés à ce type de transactions donnant lieu à de nouvelles relations entre ces entités (business to business, consumer to business, etc.).

Une autre dimension de l'interaction via l'Internet se manifeste au sein des organisations industrielles en introduisant une décentralisation croissante. C'est ce que souligne Malone dans [Malone et al.99] : « *Information technology makes distance less important in determining where decisions should be made by bringing information to decision makers wherever they are* ». Considérant le lien entre nouvelles technologies de la communication et les relations entre organisations industrielles, nous distinguons principalement deux axes. Le premier axe considère l'Internet et l'informatique principalement comme des technologies supports à la communication entre les humains. Le deuxième axe, plus visionnaire, est concentré non seulement sur la communication mais aussi sur la manière dont les technologies

informatiques peuvent *coordonner* de manière efficace les activités d'êtres humains avec un minimum d'intervention de leur part.

C'est au sein de ce dernier axe que la notion « d'entreprise virtuelle » ou « entreprise en réseau » est apparue pour désigner les situations où plusieurs entreprises décident de se regrouper et de former une organisation virtuelle. Leur objectif est ainsi d'accroître leur profit. La production industrielle se retrouve ainsi distribuée et sous-traitée à diverses unités de production choisies en fonction de leurs capacités de production ou de leurs spécialisations. Dans ce contexte, l'automatisation de la communication doit donc dépasser le cadre strict de l'entreprise classique. Elle doit conduire l'entreprise à réinventer son organisation et son fonctionnement en vue d'optimiser l'exploitation de son capital informationnel. En d'autres termes, elle doit restructurer l'ensemble de son système de pilotage autour des flux d'information. Cette mutation organisationnelle concerne toute sa chaîne de production. Elle repose surtout sur un ensemble d'opérations dont la mise en œuvre est prise en charge par des logiciels.

Le travail mené dans le cadre de cette thèse s'inscrit dans le courant que nous venons de décrire, via notamment sa contribution au projet E-Alliance. Ce projet avait comme objectif de créer un environnement et une infrastructure logicielle pour l'assistance à des activités de sous-traitance entre des organisations autonomes regroupées dans une alliance virtuelle.

Le cas d'application considéré est celui d'alliances d'ateliers d'impression. En dehors des aspects liés à la valorisation du projet dans le domaine de l'impression, des caractéristiques générales présentes dans la spécification de nombreuses entreprises virtuelles ont également été prises en compte pour développer cette infrastructure. De manière naturelle, ces ateliers sont en compétition. Cependant, afin de satisfaire des demandes qui dépassent le périmètre d'un seul atelier et mieux répondre aux exigences du marché, ils doivent s'allier et coopérer pour la réalisation d'une même tâche (Fig. 1). Le terme d'alliance que nous utilisons pour qualifier le type de regroupement met l'accent sur le fait qu'au sein de cette alliance, chaque participant est totalement *autonome,* i.e. responsable de sa propre charge de travail, de la gestion de ses ressources. Il veut conserver ses propres outils pour gérer ses contrats, ses ressources, son plan de charge, ses clients, … tout en cherchant à coopérer avec les autres, via un échange minimal d'informations pour réaliser des tâches globales. Cet échange est *minimal* au sens où l'atelier contrôle et sélectionne les informations échangées. Etant données les contraintes imposées par l'autonomie des participants et aussi du fait que les participants sont a priori en compétition,

nous considérons que le seul moyen acceptable de partager des informations ou des ressources est à travers un *processus de négociation*.

Découlant, de manière inhérente de cette autonomie et des contraintes d'exécution au sein d'un *environnement distribué*, le type d'application auquel nous nous intéressons est *décentralisé*. De plus, il est *ouvert* au sens où chaque participant peut entrer et sortir de l'alliance à tout moment mais aussi des négociations en cours. Par ailleurs, l'environnement dans lequel les alliances d'entreprises envisagées opèrent, est fortement *dynamique* du fait de la fluctuation des marchés, de la fluctuation des capacités de production au sein de chacun des participants suite à des pannes, aux aléas de production, etc. Cette dynamique influence directement les ressources des ateliers et donc indirectement les négociations dans lesquelles ils sont impliqués.

Figure 1 - Alliances d'ateliers d'impression

Dans ce contexte général, la problématique abordée dans notre thèse va s'intéresser au processus de négociation qui peut prendre place dans ce cadre de sous-traitance au sein d'alliance d'entreprises. Comme nous l'avons vu, découlant des caractéristiques de ce contexte général, le processus de négociation devra être un processus ouvert et flexible.

Le support à un tel processus de négociation ne peut pas se limiter à un simple réseau de communication entre les partenaires d'une alliance. Il est nécessaire d'offrir des mécanismes élaborés afin d'aider les ateliers à négocier dans un environnement dynamique, ouvert et où à tout moment les ateliers peuvent démarrer des négociations ou être invités dans des négociations déjà commencées. Il est également nécessaire de structurer les informations sur

les partenaires possibles, sur les travaux à sous-traiter, sur les propositions reçues ou envoyées, etc.

Poursuivant l'analyse des besoins attachés à ce contexte, il apparaît important et nécessaire que ce processus de négociation puisse être automatisé. Chaque atelier pourrait ainsi déléguer cette tâche de négociation avec tous les problèmes d'une transaction commerciale (trouver des acheteurs/vendeurs, évaluer et comparer des propositions, créer des propositions) à l'infrastructure afin d'être déchargé d'une tâche qui est souvent répétitive, difficile à gérer et très coûteuse en temps. Une telle automatisation devrait permettre d'augmenter l'efficacité du processus de négociation en prenant en compte plusieurs options de négociation et trouver l'accord en moins de temps.

Objectif

Etant donné ce contexte, notre objectif dans le cadre de ce travail est de modéliser le processus de négociation prenant place au sein d'une alliance ouverte et flexible. Nous proposons de décomposer ce processus de négociation selon trois processus distincts : *communication*, *coordination* et *décision,* dédiés respectivement à :

i) la gestion à l'identification et à l'acheminement des messages impliqués dans la négociation (processus de communication) ;

ii) à la gestion des modalités d'interaction permises dans une négociation (processus de coordination*)*. Brièvement, c'est à ce niveau que les mécanismes proposés doivent définir et gérer des contraintes afin d'identifier les messages qui peuvent être envoyés ou reçus ;

iii) la gestion du contenu de la communication : évaluation des contenus des messages reçus, création des contenus des messages à envoyer (processus de décision).

Dans ce contexte, cette thèse vise à proposer des mécanismes de base pour le processus de coordination de négociations venant s'interfacer entre le processus de communication que nous considérons comme générique et le processus de décision, qui lui est dédié et stratégique.

Afin de définir ce processus de négociation et répondre à l'exigence de délégation ou d'automatisation dans un environnement *ouvert*, *décentralisé* et *flexible*, nous nous tournons vers les technologies multi-agents. Nous modélisons la négociation comme le mode d'interaction principal entre des agents autonomes.

Les mécanismes de coordination devront ainsi maintenir une cohérence sur les décisions de l'agent en termes des actions qui peuvent être exécutées (initialisation ou finalisation d'une négociation) et des propositions qui peuvent être envoyées. Afin de gérer l'évolution concurrente de plusieurs négociations, cette cohérence doit être garantie tant au niveau local (pour une seule négociation) qu'au niveau global (pour la totalité des négociations dans lesquelles un agent est impliqué). Enfin, les mécanismes de coordination ne doivent limiter en aucune manière l'autonomie et le pouvoir des entités impliquées dans la négociation, donc le processus de coordination doit aussi être distribué au sein de chacun des agents.

Organisation du document

Ce document est structuré en quatre parties. Nous traitons tout d'abord un état de l'art sur la négociation et sur la coordination de négociations (partie II). Nous présentons ensuite notre modèle pour la coordination de négociations (partie III), suivi de sa mise en œuvre (partie IV). Nous terminons, enfin, par l'exposé des conclusions et des perspectives de nos recherches (partie V).

Dans la partie II, état de l'art, nous commencerons avec une analyse du processus de négociation tel qu'il peut être abordé dans différentes disciplines en dehors de l'informatique (chapitre 1). Cette analyse nous permet de mettre en avant un ensemble de caractéristiques que nous confrontons et complétons à la manière dont le processus de négociation est abordé dans le domaine des systèmes multi-agents (chapitre 2). Le chapitre 3 s'intéresse à l'analyse du processus de coordination que nous identifions au sein de la négociation. Nous présentons brièvement les caractéristiques attachées à ce processus que nous illustrons et détaillons par l'étude de différentes propositions de systèmes de négociation mettant en exergue la coordination de négociations. Cette partie se termine par une synthèse mettant en avant la problématique abordée vis-à-vis de l'état de l'art et positionnant nos objectifs par rapport à cet état de l'art (chapitre 4).

La partie III présente notre approche pour la coordination de négociations en commençant par la mise en place des éléments de base du modèle de négociation au sein d'alliances ouvertes et flexibles (chapitre 5). Le chapitre 6 poursuit la définition du modèle de coordination de négociation en proposant les éléments constituant le processus de négociation.

Enfin, les chapitres 7 et 8 proposent respectivement notre modèle de coordination générique et le modèle de coordination stratégique.

La mise en œuvre du modèle au sein de la partie IV, commence avec la présentation de notre proposition de l'infrastructure de négociation (chapitre 9). Dans le chapitre 10, nous détaillons les expérimentations faites sur le modèle de coordination proposé dans cette thèse. Nous terminons enfin cette partie par la présentation de l'implémentation logicielle (chapitre 11) du modèle proposé.

Nous terminons avec les conclusions sur nos travaux de recherche et les perspectives envisagées.

II. Etat de l'art

Bien que la négociation soit vieille comme le monde, le début de la négociation moderne est daté de 1968 au travers du livre « The art of negotiation » de Gerard I. Nierenberg. Le Wall Street Journal et Forbes Magazine le reconnaissent comme « The father of negotiation art and of negotiation training ». Depuis lors, de nouveaux journaux (*Negotiation Journal* [NR]*, International Journal of Conflict Management, International Negotiation* [IN]) publient exclusivement sur les recherches effectuées sur cette problématique. La négociation a été considérée au début, comme étant un processus d'interaction uniquement entre êtres humains. C'est pourquoi, elle a été essentiellement étudiée sur des aspects sociologiques, culturels ou psychologiques. Suite au développement des technologies de l'information et de la communication, la négociation a cessé d'être l'apanage des sciences humaines. D'autres sciences comme l'informatique ont commencé à l'étudier sur d'autres aspects comme l'efficacité, la flexibilité ou le passage à l'échelle.

Même si dans notre travail, nous l'envisageons médiatisée par une infrastructure logicielle, le participant humain est considéré aussi comme partie intégrante du système. C'est pourquoi, nous sommes amenés à prendre en compte aussi les caractéristiques des négociations entre êtres humains. Partant de ces caractéristiques nous nous sommes intéressés à identifier les dimensions impliquées dans une négociation afin de les modéliser et de coordonner leur évolution par des mécanismes informatiques. La suite de cette partie est ainsi structurée de la manière suivante :

- Nous démarrons l'état de l'art avec une présentation générale des différentes dimensions de la négociation, rencontrées dans plusieurs disciplines utilisant et étudiant la négociation, telles que le management, le droit, les sciences économiques, la linguistique, la psychologie et la psychanalyse ;
- Nous reprenons ensuite ces dimensions pour l'étude de la négociation en informatique et plus précisément dans les systèmes multi-agent (SMA). Cette étude permet de caractériser le processus de négociation comme une structuration de trois processus : communication, coordination et décision.
- Finalement, dans le dernier chapitre nous nous focalisons sur le processus de coordination au sein de négociations. Plus particulièrement, dans le cadre d'un SMA, nous

détaillerons les aspects liés au processus de coordination qui gèrent les interactions menées au cours des négociations entre agents.

1. Analyse pluridisciplinaire de la Négociation

Dans les termes les plus simples, « la négociation est une discussion entre deux participants ou plus qui essaient d'établir une solution à leur problème » [NIOTP]. Ce processus interpersonnel ou intergroupe peut se produire tant au niveau d'une seule personne que d'une organisation industrielle ou même au niveau international (diplomatique).

Par la suite, nous identifions certaines caractéristiques du processus de négociation qui sont communes à la majorité des domaines de recherche. Afin de faire une séparation entre les mécanismes qui gèrent la négociation et les aspects qui ne sont pas liés exclusivement à la négociation mais qui influencent le processus de négociation, nous présentons cette vision synthétique des différents travaux sur la négociation selon une grille structurée sur trois catégories de dimensions : *le participant* impliqué dans la négociation, *le contexte* dans lequel la négociation se déroule et *le mécanisme* qui gère la négociation.

1.1. Participant d'une négociation

Dans la plupart des disciplines (hors informatique) qui étudient les mécanismes de négociation, le participant est humain[1]. Dans cette section nous allons détailler deux aspects du participant dans une négociation : 1) *une dimension individuelle* qui décrit les aspects humains du participant et 2) *une dimension sociale* caractérisant les rôles possibles dans une négociation. Nous avons proposé cette division afin de faire plus facilement la distinction entre les caractéristiques du comportement du participant qui peuvent influencer le processus de négociation et inversement quelles sont à ce niveau les contraintes possibles imposées par le processus de négociation.

[1] Le participant comme une entité informatique sera détaillé dans la section 2.1.

1.1.1. Dimension individuelle

Dans les négociations de tous les jours le succès de n'importe quelle transaction dépend étonnamment beaucoup du facteur humain des joueurs qui négocient ces transactions. Les participants sont des individus uniques avec des personnalités et des motivations distinctes, qui ont un comportement bien délimité selon leurs émotions et leurs psychologies. Le terme de "négociation" est même utilisé dans la psychanalyse contemporaine [Wu et Laws'03] pour définir la rencontre clinique, dans un effort de décrire les schémas interactifs du traitement psychanalytique sur le comportement d'un patient.

En dépit de la prédominance de l'émotion dans la négociation, dans les domaines économiques, une grande partie des études suggèrent que les émotions n'aient aucune présence à la table de négociation. Fisher et Ury [Fisher et Ury] dans leur livre « Getter to Yes !», précisent le fait que « la participation émotive d'un côté fait qu'il est difficile à réaliser le détachement nécessaire afin de penser des manières sages de prendre en compte les intérêts des deux côtés. »

Une perspective différente peut être trouvée dans les livres faisant une distinction précise entre *l'esprit du participant* responsable du développement des stratégies et des mécanismes de raisonnement pour la négociation, et *le cœur du participant* caractérisant les implications émotionnelles, psychologiques et les aspects intuitifs de la négociation [Thompson'99].

Une autre approche souligne même le fait que les négociateurs habiles puissent apprendre à manipuler leurs propres émotions et celles exprimées par les autres participants plus efficacement en commençant à identifier deux négociations distinctes, une qu'ils doivent avoir avec leurs émotions et une autre avec la personne à travers la table [Gray'03a]. Alors ils sont libres à répondre de manière constructive à leurs adversaires.

Ceci signifie que les bons négociateurs peuvent, et probablement, cherchent et trouvent des régularités dans les comportements émotionnels des autres participants. En trouvant ces régularités dans les valeurs attachées à la dimension individuelle des différents participants, les négociateurs peuvent mieux ajuster leurs comportements en fonction de ceux des autres participants.

1.1.2. Dimension sociale

Au cours d'une négociation, certaines régularités dans les actions des participants sont liées aux valeurs attachées à la dimension sociale qui identifie les rôles que les participants jouent dans le déroulement du processus.

Dans les études sur la négociation plusieurs rôles ont été proposés. En revanche, les caractéristiques de ces rôles ne suivent pas toujours les mêmes dimensions ayant jusqu'à combiner deux ou plusieurs rôles afin de définir clairement le rôle d'un participant. Nous avons essayé de structurer les rôles proposés selon trois aspects : i) les buts que les participants cherchent à satisfaire et dans ce cas les rôles sont *vendeur* et *acheteur* ; ii) l'implication du participant dans le processus de négociation et dans ce cas les rôles sont *initiateur* et *invité*; et iii) les pouvoirs des participants sur l'exécution du processus de négociation et dans ce cas les rôles sont *médiateur* et *participant actif*.

- *Vendeur/Acheteur :* Le *vendeur* est celui qui se trouve dans la possession d'un objet ou d'un service et qui cherche de le vendre. L'*acheteur* a un but complémentaire, il cherche justement un objet ou un service spécifique afin de l'acheter [IN]. Nous pouvons observer que ces deux rôles sont liés non seulement à la notion du but mais aussi à un scénario typique de négociation dans lequel une partie demande quelque chose et une autre est prête à l'offrir.

- *Initiateur/Invité :* Un autre scénario possible peut supposer que la négociation est démarrée parce que les parties souhaitent créer quelque chose de nouveau que ni l'une ni l'autre ne pourraient faire toutes seules. Dans ce deuxième scénario, la différence entre leurs buts n'est pas assez visible pour faire clairement la séparation entre celui qui vend ou celui qui achète. Dans ce cas une distinction plus précise entre les participants peut être faite au niveau des actions entreprises pour démarrer le processus de négociation.

  L'*initiateur* est celui qui commence la négociation et l'*invité* est le participant à qu'il est proposé d'entrer dans ce processus.

- *Médiateur/Participant actif :* Une caractéristique commune aux participants ayant les rôles présentés au-dessus est le fait qu'ils aient la capacité et le pouvoir de décision sur le fond de la négociation.

Les participants actifs sont donc ceux qui décident le résultat final de la négociation et dans le même temps ils sont les bénéficiaires directs de ce résultat.

Un rôle bien distinct de ceux présentés auparavant est le rôle de *médiateur. Les médiateurs* ou les facilitateurs sont des personnes neutres qui aident un groupe à négocier ensemble plus efficacement. Le médiateur peut être interne ou externe (c'est-à-dire, venant d'une organisation qui n'a pas d'intérêt dans la négociation courante). Dans les deux cas le médiateur doit être accepté par tous les membres du groupe impliqués dans la négociation. Les médiateurs sont vus seulement comme « les chefs du processus de négociation » - ils n'ont aucune autorité dans la prise de décision et ils ne contribuent pas à la substance de la discussion [J.C. Melamed].

Si *le participant actif* cherche pendant le processus de négociation de répondre aux questions : « Qu'est qu'il y a à dire ? » et « Comment le dire ? » [E. Wertheim], le *médiateur* lui répond aux questions : « Quand le dire ? » et « A qui le dire ?».

Bien sûre que prenant en compte des scénarios plus précis et plus détaillés, les rôles peuvent se compléter : un vendeur peut entrer initiateur ou invité dans une négociation, de même que l'acheteur peut aussi commencer une négociation ou être invité.

Le comportement d'un participant dans une négociation est encadré dans certaines limites selon les valeurs attachées aux deux dimensions – individuelle et sociale –. Donc, ces deux dimensions délimitent un espace dans lequel le participant peut négocier à sa manière.

Le déroulement d'une négociation ne dépend pas seulement des participants impliqués mais aussi du contexte dans lequel la négociation est engagée et qui apporte des nouvelles dimensions.

1.2. Contexte d'une négociation

La notion de contexte, ses structurations et ses caractéristiques font aussi l'objet de beaucoup de travaux. Par la suite, nous n'essayons pas de proposer une nouvelle façon de structurer le contexte, mais d'énumérer certaines dimensions de l'environnement et des contraintes qui peuvent influencer le comportement d'un négociateur ou le déroulement du processus de négociation. Sans prétendre être exhaustif et selon notre étude, les dimensions du contexte qui ont une forte influence sur le processus de négociation sont *la culture*, *le pouvoir* et *le temps*.

1.2.1. Culture

La culture est toujours une dimension importante dans une négociation. Un cas plus précis c'est la négociation diplomatique d'un conflit où pour sa résolution la culture joue un rôle central en influençant les processus de négociation.

La notion de culture est souvent associée aux notions de cultures nationales rapportées dans les médias internationaux. Cependant, la culture est beaucoup plus large et entoure les croyances, les attitudes et les comportements des groups divers. La culture différencie également les personnes en fonction de leurs croyances religieuses ou idéologiques, des professions et des milieux éducatifs ou par âge ou gendre (masculin/féminin).

Selon Bass [Bass'03] la culture n'est pas une notion singulière mais elle est le résultat cumulatif de « *l'expérience, les valeurs, la religion, la croyance, les attitudes, les significations, la connaissance, les organismes sociaux, les procédures, la synchronisation, les rôles, les relations spatiales, les concepts des objets d'univers et de matériel et les possessions acquises ou créées par des groupes de personnes, au cours des générations, par l'effort individuel ou par les interactions de groupe* ».

Avec toutes ces variables culturelles ayant des variations significatives dans les différentes cultures, les théories sur la négociation et la résolution des conflits essaient de développer des hypothèses générales sur la façon dont une personne ou un groupe particulier de n'importe quelle culture pourrait se comporter dans les négociations.

Ainsi, les attitudes et les comportements culturels peuvent influencer le déroulement d'une négociation selon plusieurs dimensions (adapté à partir de [Moor et Woodrow'98]) :

- manières d'exécuter les étapes d'une négociation ;
- stratégies et tactiques de négociation ;
- le rôle des rapports et de la confiance entre les participants d'une négociation ;
- orientation vers le gain ou le succès ;
- normes sanctionnées ou acceptables au sujet des résultats - préférences pour les composants substantifs, procéduraux ou psychologiques du résultat d'une négociation.

Tandis qu'il est difficile de caractériser n'importe quelle approche nationale ou culturelle à la négociation, des généralisations sont fréquemment dessinées. Ces généralisations sont présentées par les différentes études sur la négociation comme une sorte de guide pour les participants d'une négociation afin de bien intégrer le processus de la négociation dans un contexte plus large. Donc, la dimension culturelle se manifeste dans les modèles de la langue, du comportement et des activités et fournit des normes pour les interactions acceptables et les modèles de communication utilisés dans un processus de négociation.

1.2.2. Pouvoir

Généralement, le pouvoir peut comporter tout ce qu'établit et maintient la relation d'autorité entre deux ou plusieurs personnes. Ainsi, le pouvoir couvre tous les rapports sociaux, des influences les plus subtiles par lesquelles un homme peut avoir de l'autorité sur les autres pouvant s'exprimer à l'extrême par la violence physique.

Kenneth Boulding [Boulding'89] propose trois types de pouvoir: *le pouvoir intégratif*, *le pouvoir coercitif* et *le pouvoir d'échange*.

- *Le pouvoir intégratif* est défini comme la capacité d'obtenir des autres ce que nous avons besoin et voulons de concret.
- Dans le spectre opposé *le pouvoir coercitif* est défini comme la capacité d'obtenir quelque chose sans tenir compte des autres.
- Différemment de ces deux cas où le pouvoir est associé à une seule entité, *le pouvoir d'échange* est vu comme une relation équitable entre plusieurs parties. De manière immédiate, le pouvoir d'échange est un concept tout à fait similaire à la négociation : « je veux que vous fassiez quelque chose et pour vous convaincre de la faire, je vous offre quelque chose, que vous désirez ».

Donc, avec l'ajout de cet aspect plus économique sur le pouvoir -en dehors de l'autorité- le pouvoir peut être vu comme une mesure de la relation entre les parties impliquées dans une négociation.

Ainsi, dans une négociation le pouvoir devient un *pouvoir relatif* utilisé par les négociateurs pour établir l'ampleur de la dépendance de leurs relations qui existaient avant la négociation, la relation qui se dévoile pendant les négociations et la relation désirée à la fin de la négociation. Les chercheurs [Fisher'91] ont constaté que dans une négociation les individus évaluent leur pouvoir dans une relation et choisissent un certain modèle de comportement afin de concurrencer, s'adapter, collaborer ou de se retirer.

Les parties peuvent évaluer leur puissance par rapport à l'autre partie en comparant leurs capacités en termes des ressources humaines et matérielles disponibles, des qualifications et des connaissances nécessaires, des sanctions et des représailles possibles et aussi celles des autorités organisationnelles. Ces différents aspects aident un participant d'une négociation à déterminer si son rapport avec l'autre partie est basé sur l'indépendance, la dépendance ou l'interdépendance et souvent déterminent si l'un ou l'autre négociateur est motivé pour partager le gain, pour le saisir, ou pour l'offrir.

1.2.3. Temps

Selon les cultures à travers le monde deux orientations différentes du temps existent: *mono chronique* et *poly-chronique* [Kersten et al.'02]. Les approches mono chroniques du temps sont linéaires, séquentielles et supposent la concentration sur une seule chose à la fois[2]. Les orientations poly-chroniques du temps comportent des occurrences simultanées de beaucoup de choses et la participation de beaucoup de gens[3].

Du point de vue de la négociation, les différences entre les deux approches culturelles du temps sont les suivantes : 1) pour les cultures poly-chroniques le temps nécessaire pour accomplir une négociation est élastique et plus important que tout programme, également la communication suppose un flux élevé d'informations et qui ne sont pas toutes pertinentes ; et 2)

[2] Ces approches sont les plus communes dans les cultures Européennes-influencées des Etats-Unis, de l'Allemagne, de la Suisse, et de la Scandinavie. [Bass'03]

[3] Cette orientation est la plus commune dans les cultures méditerranéennes et latines comprenant la France, l'Italie, la Grèce, et le Mexique, aussi bien que quelques cultures orientales et africaines.

pour les cultures mono chroniques le programme de la négociation est fixe, également la communication est fondée sur des informations pertinentes et communiquées séquentiellement.

Une autre approche qui concerne le temps comme une variable du processus de négociation est divisée aussi en deux dimensions selon l'étape de la négociation dans laquelle le temps est pris en compte : le début de la négociation ou la finalisation de la négociation.

Au début de la négociation le temps est une variable de synchronisation. Les théoriciens de la négociation [Zartman'00] emploient souvent le concept de « maturité » (« ripeness »), en comparant la négociation à un fruit. Si un fruit est ramassé trop tôt, il ne sera pas prêt pour être mangé, cependant, s'il est ramassé trop tard, il sera aussi non comestible. La négociation fonctionne de la même manière, si la synchronisation est mal choisie, les négociations n'aboutiront pas. Cela vaut particulièrement pour la négociation par la médiation où le médiateur doit négocier avec les parties quand elles sont toutes prêtes à participer afin de faciliter aux parties d'arriver à un consensus.

Pour la fin de la négociation, le temps est une variable affectant le déroulement de la négociation. Le temps peut faire aussi objet de la négociation - les parties pouvant négocier le moment où la négociation s'arrête – ou c'est seulement une variable qui caractérise pour chaque partie l'urgence d'arriver à un résultat.

Donc le temps peut influencer non seulement le résultat d'une négociation mais aussi la manière dont ce résultat est obtenu.

1.3. Mécanisme de négociation

Toutes les dimensions détaillées jusqu'à maintenant fixent des contraintes sur la dynamique du processus sans présenter les moyens employés pour caractériser et gérer le déroulement du processus de négociation. Par la suite, nous présentons les étapes d'une négociation et quelles sont les dimensions principales qui caractérisent les mécanismes utilisés au cours du processus de négociation.

Un processus de négociation est habituellement composé de deux grandes étapes :

i) *pré négociation* (ou planification de la négociation) qui se rapporte aux discussions qui précèdent les négociations formelles et qui inclut souvent des questions de procédure : qui sera impliqué, où et quand les négociations auront lieu, comment elles seront structurées, quel sera l'objet de la négociation ? Les réponses aux certaines de ces questions seront des valeurs pour les dimensions définies auparavant, comme le participant ou le temps. Afin de répondre aux autres questions des nouvelles dimensions et les possibles valeurs doivent être définies.

ii) *négociation* qui se rapporte aux interactions avec l'échange des propositions et des contre-propositions formulées à partir des stratégies de négociation.

D'un point de vue de l'instrumentation du processus de négociation selon ces deux étapes, trois dimensions principales doivent être prises en compte : i) *les informations manipulées*, dimension relative aux données définissant le cadre et le contenu de la négociation, (ii) *les protocoles de négociation*, dimension relative aux messages (langage, contenu, séquences, nombre de participants) envoyés entre les participants (iii) *le raisonnement ou les stratégies de négociation*, dimension relative à la modélisation du raisonnement des participants impliqués dans la négociation qu'ils utilisent pour satisfaire leurs objectifs.

1.3.1. Informations manipulées

Dans les premières étapes de la planification de la négociation, les négociateurs doivent déterminer leurs buts, prévoir ce qu'ils veulent réaliser et se préparer au processus de négociation. En fonction de leurs buts, les parties vont assembler dans des listes complètes

toutes les informations et les données qui aideront à définir sur quoi se négocient et dans quelle manière se négocient.

Les négociateurs souvent échangent et/ou négocient à l'avance la liste d'issues à discuter. La consultation entre les négociateurs avant la négociation réelle leur permet de convenir des listes des informations qui définissent *l'objet de négociation* par des attributs à discuter, aussi bien que d'autres caractéristiques de la négociation comme l'endroit de la négociation, l'heure et la durée des sessions, les parties impliquées dans la négociation et les techniques à poursuivre si la négociation échoue. Comme nous avons souligné auparavant, le but de la phase de planification de négociation n'est pas d'essayer de résoudre le problème, mais d'obtenir l'information qui permettra de construire une image plus claire des vraies issues à négocier et aussi de la façon dont l'autre partie voit la réalité afin de mieux choisir les stratégies à utiliser par la suite dans la négociation.

Tout cet échange préliminaire est identifié dans les approches économiques de la négociation comme une *conversation par structures* (en anglais: frame[4] [Lewicki et al.'99]).

Pendant l'évolution d'une planification de négociation, les structures agissent en tant que récipients au niveau desquels l'information est recueillie et analysée, les positions sont déterminées (y compris les priorités, les moyens, et les solutions) et des plans d'action sont développés. Selon le contexte, les structures peuvent être employées pour conceptualiser et interpréter, ou pour manœuvrer et convaincre [Gray'03b].

Une de ces structures, appelée *objet de négociation*, contient donc les attributs qu'un des participants envisage à négocier [Putman et Holmer'92]. Dans certains cas il s'agit de négocier uniquement un seul attribut (par exemple - le prix), mais dans d'autres cas il faut aussi négocier plusieurs attributs comme le temps nécessaire pour satisfaire une commande, la qualité des produits, etc. Avant de commencer la négociation un participant fixe non seulement les attributs de l'objet mais aussi il essaie d'être le plus précis que possible en rendant ces objectifs mesurables et opportuns [Lewicki et al.'00]. En fixant aussi des valeurs possibles pour les attributs à négocier, un participant identifie un ensemble des objets de négociation, plutôt qu'un seul objet. En fonction des valeurs fixées pour les attributs, l'ensemble inclut : i) *un objet maximum* – le meilleur résultat possible, ii) *un objet minimum* – le résultat le plus bas acceptable, et iii) *un objet cible* – un résultat fixé avec réalisme.

[4] Une structure ou frame est "collection of perceptions and thoughts that people use to define a situation, organize information, and determine what is important and what is not."

Après la prise de décision concernant l'objet de négociation, les négociateurs doivent donner la priorité à leurs buts et évaluer les possibles différences parmi eux. Les négociateurs doivent se rendre compte de leurs buts et positions et ils doivent identifier les soucis, les désirs, et les craintes qui sont à la base de leurs buts. Ils doivent déterminer quelles sont les questions les plus importantes, aussi bien que si les divers attributs sur lesquels ils négocient sont liés ou indépendants.

Puisque les objets de négociation impliquent typiquement plus d'un attribut, il est utile que les négociateurs prévoient différentes manières d'empaqueter des attributs. Ils peuvent identifier les attributs qu'ils considèrent comme les plus importants afin d'être plus flexibles au sujet des attributs qu'ils considèrent moins importants.

Ainsi, l'analyse des valeurs pour cette dimension fournit aux participants une meilleure compréhension de la dynamique et du développement de la négociation et leur permet d'envisager quelles sont les meilleures stratégies de négociation qu'ils peuvent employer par la suite.

1.3.2. Protocole de négociation

Pendant la phase de négociation et aussi pendant la phase de pré négociation une communication doit être conduite entre les participants. Dans ce cas afin que les participants puissent communiquer et se comprendre certaines règles générales sur le processus de négociation doivent être fixées.

Le protocole de négociation est la dimension du processus de négociation qui établit l'ensemble de ces règles: *i)* les participants possibles dans la négociation, *ii)* les propositions légales que les participants peuvent faire, *iii)* les états de la négociation (par exemple l'état initial où commence la négociation, l'état où on accepte des soumissions ou la fin de la négociation), *iv)* des règles pour déterminer le moment où on est arrive à un accord, et aussi *v)* la durée de la négociation fixant quand il faut s'arrêter parce qu'aucun accord n'a pas pu être trouvé.

Avec la majorité des protocoles, une négociation se présente comme une succession de tours de parole soumise à certains principes de cohérence. La négociation est un processus qui obéit à des règles d'enchaînement syntaxique, sémantique et pragmatique. C'est donc le

protocole qui fixe les règles qui régissent l'interaction et la communication entre les participants pendant une négociation. On en distingue trois catégories [Moore'96] :

- règles qui permettent la gestion de l'alternance des tours de parole,
- règles qui interviennent au niveau de la relation interpersonnelle,
- règles qui régissent l'organisation structurale de la communication.

Ces différentes règles fixent un espace de négociation (« rules of the game » – [Rosenschein et Zlotkin'94]) dans lequel les participants peuvent négocier selon leurs propres stratégies de négociation.

1.3.3. Stratégie de négociation

En se basant sur les informations issues de la phase de pré négociation ou sur les historiques des autres négociations, les parties impliquées dans une négociation décident leurs stratégies à suivre dans la négociation considérée. Une *stratégie de négociation* [Walton et Krabbe'95] peut être définie comme la dimension du processus de négociation qui fournit une réponse à la question :

« Qu'est-ce qu'un participant devrait dire et quand, dans une négociation particulière? ».

Afin d'identifier une stratégie de négociation, différentes approches des plus générales aux plus spécifiques sont utilisées : *i)* approche conflictuelle ou stratégie coopérative/compétitive; *ii)* approche décisionnelle ou stratégie intégrative/distributive et *iii)* approche économique ou stratégie gagnant-gagnant/ gagnant-perdant/perdant-perdant.

Ces différentes approches utilisées pour définir et caractériser des stratégies ne sont pas bien délimitées et souvent plusieurs termes sont utilisés afin de bien identifier les caractéristiques d'une stratégie de négociation. Dans les négociations avec des stratégies intégratives on trouve aussi les termes de négociation coopérative ou de négociation gagnant – gagnant ou de positive-somme et dans les négociations avec des stratégies distributives on trouve les termes de négociation compétitive, gagnant – perdant ou zéro-somme.

Par la suite, nous essayons de définir plus clairement quelles sont les principales caractéristiques de ces trois approches.

Approche conflictuelle

Quand les participants entament un processus de négociation pour résoudre un conflit, ils apporteront à la table une certaine intention vis à vis de l'issue du conflit. Les deux orientations fondamentales découlant de ces intentions sont les négociations coopératives ou compétitives.

Les facteurs les plus importants qui déterminent si un individu approchera une négociation de manière coopérative ou compétitive sont la nature du conflit et les buts recherchés de chaque côté. Souvent les buts des deux côtés sont liés ou interdépendants. Selon [Deutsch'00], le type de stratégie envisagé par les parties d'une négociation sera choisi en fonction de cette interdépendance qui peut être positive ou négative :

- D'une part, les buts avec l'interdépendance positive sont attachés ensemble de telle manière que la chance pour une partie d'atteindre son but est augmentée si l'autre partie atteint son but [Moor et Woodrow'98]. Les buts avec une interdépendance positive ont normalement comme conséquence des approches coopératives à la négociation, parce que n'importe quel participant "atteindra son but si, et seulement si, les autres avec qu'il négocie peuvent atteindre aussi leurs buts." [Boulding'89]
- D'autre part, l'interdépendance négative signifie que la chance pour une partie d'atteindre son but est diminuée si les autres participants dans la négociation atteignent leurs buts avec succès [Fisher'91]. Les buts avec une interdépendance négative forcent des situations compétitives, parce que la seule manière pour qu'un côté réalise ses buts et de gagner et que l'autre côté perd.

Approche décisionnelle

Bien qu'il n'existe pas une différence claire entre les stratégies intégratives et distributives, cette division des stratégies de négociation est la plus utilisée.

Brièvement la différence entre les deux est la suivante [Walton et Krabbe'95] :

- la négociation intégrative suppose une stratégie de négociation dans laquelle les parties collaborent pour trouver la solution à leur conflit. Cette stratégie se concentre afin de développer des accords mutuellement gagnants basés sur les intérêts et les préférences de chacun des participants.
- la négociation distributive suppose une stratégie de négociation dans laquelle le modèle de décision est construit pour un seul participant afin que les intérêts et les préférences de cette personne soient les plus importants.

Cette différenciation peut être acceptée si la stratégie intégrative cherche un gain global maximal en détriment du gain local et la stratégie distributive cherche un gain local maximal en détriment du gain global, mais en théorie les deux stratégies cherchent premièrement un gain local maximal.

Une autre différenciation plus détaillée entre les deux, proposée par Lax et Sebenius [Lax et Sebenius'86] et continuée après par Kersten [Kersten'01], est basée sur la notion de « la tarte » (en anglais « the pie ») qui corresponde à l'ensemble des possibles alternatives dans une négociation.

- Dans la stratégie distributive la tarte est fixe, donc les alternatives possibles dans la négociation sont connues et non modifiables. Ainsi, le plus qu'une des parties obtient à l'issue de la négociation, a comme conséquence la diminution du gain pour l'autre partie.
- Dans la négociation intégrative la tarte est modifiable. Ce concept suppose que les parties n'ont pas un set fixe d'alternatives de négociation. La création de nouvelles alternatives est faite dans le sens d'augmenter l'ensemble d'offres et de les négocier si possible. Cette approche est possible quand "les parties souvent ne savent pas où s'arrêter avec l'agrandissement de la tarte" [Lax et Sebenius'86]. C'est la situation dans laquelle les parties commencent leur négociation avec seulement la connaissance d'un sous-ensemble des alternatives et l'ensemble total d'offres faisables ne peut jamais être bien défini. La négociation intégrative est une bonne manière de rendre la tarte aussi grande qu'elle probablement peut être, mais finalement les parties doivent distribuer la valeur qui a été créée par la négociation. Ils doivent convenir sur qui obtient quoi. L'idée derrière la négociation intégrative est que cette dernière étape ne sera pas difficile une fois que les parties atteignent cette étape.

Lax et Sebenius étaient parmi les premiers à argumenter le fait que réellement toutes les négociations étaient des combinaisons des deux stratégies. Premièrement, les négociateurs essaient de "créer la valeur" en agrandissant la tarte autant qu'ils peuvent. Cependant inévitablement, la tarte devra alors être divisée et ça réclame de la négociation distributive.

Approche économique

Quand les parties négocient, elles s'attendent habituellement de recevoir ou d'offrir quelque chose. Pendant le déroulement d'une négociation les participants évaluent s'ils ont satisfait leurs buts et de quelle manière. Cette évaluation peut être *gagnant - gagnant, gagnant - perdant* ou *perdant - perdant* et le type de stratégie de négociation qui est associé change en conséquence.

Gagnant-gagnant, gagnant-perdant, et perdant-perdant sont des termes de la théorie des jeux qui se rapportent aux résultats possibles d'un jeu ou d'une négociation impliquant deux côtés, et d'une manière primordiale la manière dont chaque côté perçoit leurs résultats relativement à leur position avant le jeu :

- *gagnant – gagnant,* les résultats se produisent quand chaque côté d'une négociation juge qu'ils ont gagné ;
- *gagnant – perdant*, les résultats se produisent quand une seul des deux parties juge qu'elle a gagné ;
- *perdant – perdant*, signifie que toutes les parties finissent la négociation avec un moindre gain que prévu ou c'est une mesure identifiant les plus mauvais gains dans un jeu ou la stratégie gagnant-gagnant est aussi possible.

L'exemple classique dans la théorie du jeu illustrant les trois situations est celui de la situation appelée *le dilemme du prisonnier* dans lequel deux prisonniers doivent décider d'admettre qu'ils ont fait un crime. Ni l'un ni l'autre prisonnier ne savent pas ce que l'autre fera. Les meilleurs résultats pour le prisonnier A se produisent s'il avoue, alors que le prisonnier B ne dit rien. Dans ce cas-là, le prisonnier qui admet le crime est récompensé en étant libéré, et l'autre (qui est ne dit rien) reçoit la peine maximale, car il n'a pas coopéré avec la police, (c'est le résultat de type gagnant - perdant). C'est le même cas pour le prisonnier B. Mais si les deux prisonniers admettent, ils recevrons chacun la peine maximale (c'est le résultats de type perdant - perdant). Si ni l'un ni l'autre n'admettent pas (essayant de tirer profit de leur association), tous les deux recevrons une peine réduite (c'est le résultat de type gagnant - gagnant, bien que la victoire ne soit pas aussi grande que celle qu'ils auraient reçu dans le scénario de gagnant - perdant).

Cette situation se produit assez souvent, car les résultats gagnant - gagnant peuvent seulement être identifiés par la négociation coopérative (ou intégrative), et ils ne sont pas susceptibles d'être possibles avec une position distributive ou compétitive.

Une autre approche issue de la théorie du jeu propose les stratégies de type *zéro-somme, positive-somme, et négative-somme* qui se rapportent aussi aux résultats d'un conflit ou d'une négociation. Elles se réfèrent à la quantité réelle du gain comme des récompenses mesurables (ex. : argent, ressources, temps) que chaque partie reçoit. Bien que semblables et souvent employées pour les mêmes choses, ces stratégies diffèrent des celles " gagnant-gagnant, gagnant-perdant, et perdant-perdant " qui se rapportent aux victoires ou aux pertes par rapport aux expectations.

1.4. Synthèse

Avec cette étude pluridisciplinaire selon les trois axes considérés (participant de la négociation, contexte de la négociation et mécanisme de négociation) nous pouvons dégager plusieurs caractéristiques et dimensions communes de la négociation. Avant de les résumer, arrêtons nous un moment sur deux définitions de la négociation qui mettent en avant deux processus distincts :

- *« La négociation est le processus par lequel plusieurs individus prennent une décision commune. Les participants expriment d'abord des demandes contradictoires, puis ils essaient de trouver un accord par concession ou par la recherche de nouvelles alternatives. »* [Wertheim]
- *« Par négociation, on entend une discussion dans laquelle des individus intéressés échangent des informations et arrivent à un accord en commun. »* [Hall'93]

Ces deux définitions identifient clairement deux processus différents composant le processus de négociation (Fig. 2) : ***le processus de décision*** et ***le processus de communication***.

- le *processus de décision* est le processus par lequel un participant raisonne pour déterminer quelles seront ses propositions futures. Ce processus est principalement influencé par les dimensions *stratégie* et *objet de négociation* que nous venons de voir. Mais dans le même temps, le raisonnement d'un participant à une négociation est influencé : par la dimension *individuelle* ou *sociale* qui caractérisent son comportement dans une négociation, par la dimension *culturelle* qui délimite les valeurs, les croyances et les attitudes de bases du participant, par le *pouvoir* que le participant détient dans la négociation, et finalement par la dimension *temporelle* qui détermine si le participant est pressé ou non par le temps. Ainsi, pour prendre une décision adéquate, un participant doit être capable de faire un raisonnement tenant compte de toutes ces dimensions ;
- négocier c'est échanger. Ainsi l'exercice de la négociation implique l'existence d'un *processus de communication* tout au long du déroulement d'une négociation. Ce processus est principalement modélisé par la dimension *protocole de négociation*. Le choix du protocole de négociation ou les définitions des différentes règles à satisfaire pendant la négociation sont aussi influencées par : la dimension *sociale* du participant qui selon le rôle attribué détermine certaines limites dans la communication, par la dimension *culturelle* qui identifie les normes sur les modèles de communication adoptés par un participant, par la

dimension *temporelle* qui fait que la communication se déroule selon un schéma linéaire ou arborescent (i.e. mono-chronique et poly-chronique), et finalement, par le *pouvoir* qui peut réduire une négociation dans une simple échange d'ordre.

Ces deux processus ne sont pas indépendants mais solidaires. Tout au long d'une négociation les participants doivent pouvoir construire des propositions nouvelles en résonant sur les valeurs des dimensions de la négociation considérée et de communiquer ces propositions en respectant les différentes règles de communication.

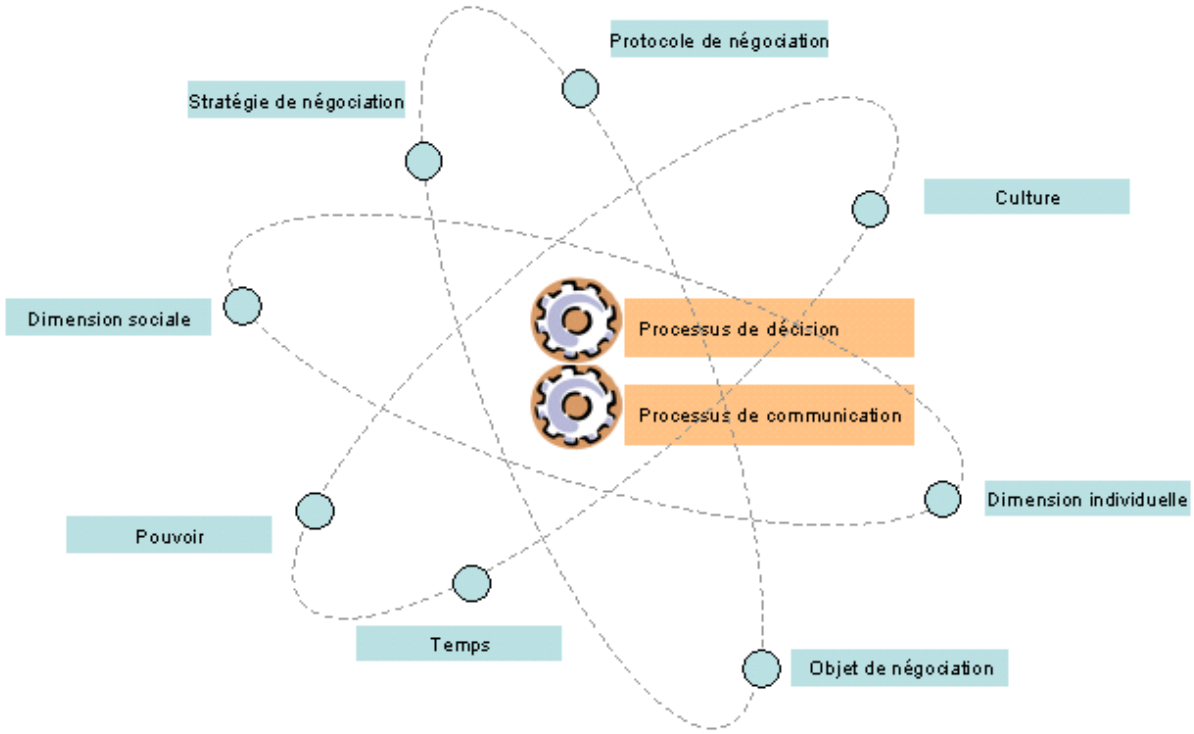

Figure 2. Structuration et dimensions d'un processus de négociation

Finalement, dans cet espace multidimensionnel nous avons identifié un ensemble des caractéristiques de la négociation pouvant être associées à des valeurs fixes afin de caractériser plus précisément le type de négociation envisagée. Par la suite, nous allons reprendre ces différentes dimensions afin d'analyser la mise en œuvre du processus de négociation par des systèmes informatiques. Nous chercherons également à compléter cet espace avec des nouvelles dimensions spécifiques à l'informatique.

2. Négociation dans les Systèmes Multi-Agents

Avec le progrès des technologies de l'information et de la communication, la plupart des formes de négociation décrites dans le chapitre précédent peuvent être maintenant implémentées et gérées de manière automatique par des systèmes informatiques. Parmi ces technologies, les systèmes multi-agents (SMA) ont fortement été sollicités pour les mettre en œuvre. En effet, la négociation est un des comportements les plus utilisés à partir du moment où plusieurs agents autonomes interagissent. Difficile et complexe, de nombreux modèles[5] et cycles de vie sont proposés [Robinson et Volkov'98] [Raiffa'82].

Il apparaît que l'ensemble des négociations mises en œuvre en informatique peuvent s'exprimer selon deux formes de négociation – intégrative et distributive – [Kersten et al.'99] :

– *La négociation distributive* ou compétitive (gagnant-perdant) envisage la négociation entre deux participants comme une situation dans laquelle seulement un des deux peut gagner au dépend de l'autre. Ce type de négociation est basé sur une forte utilisation de propositions/contre-propositions pour satisfaire le but initial. Les différentes méthodes d'enchères [Sandholm'99], par exemple, qui sont fortement utilisées actuellement dans le domaine du commerce électronique, se rattachent à cette catégorie.

– En revanche, *la négociation intégrative* ou coopérative (gagnant-gagnant) envisage l'objet de la négociation comme faisant partie intégrante du processus de négociation : l'objet est construit par les parties impliquées et il évolue au cours de la négociation. Cette construction dynamique s'appuie sur des échanges de propositions fondées sur le fait que les participants les construisent sur les propositions antérieures. Cette approche vise principalement à améliorer les propositions et à augmenter la probabilité d'atteindre un consensus. Ces négociations pourront faire appel, par exemple, à la négociation par argumentation [Parsons'98] ou à l'échange de retours qualitatifs positifs ou négatifs sur les propositions réalisées [Cordoso et Oliveira'00].

[5] Actuellement une communauté centrée sur la négociation se constitue : workshop régulier "E-negotiation" chaque année au sein de la conférence DEXA, liste de diffusion, réflexion et constitution d'un schéma de classification des procédures de négociation.

Tous ces différents types de négociations, qu'ils soient dans une catégorie ou l'autre, ont des dimensions et des caractéristiques communes. Dans ce chapitre nous structurons ces caractéristiques selon les trois axes introduits dans l'analyse pluridisciplinaire de la négociation : *participant*, *contexte* et *mécanisme*. La différence à avoir en tête est que maintenant, le participant n'est plus un être humain mais un agent informatique.

2.1. Participant d'une négociation

Dans les autres domaines, mais aussi dans l'informatique, le terme "agent autonome" est souvent mal utilisé et mal compris. Nous pouvons trouver facilement beaucoup de logiciels décrits comme des agents, bien qu'ils ne soient pas nécessairement corrélés avec la définition employée dans les SMA.

Après l'analyse des plusieurs propositions, Franklin et Grasser [Franklin et Grasser'96] donnent une définition dans laquelle ils expliquent l'essence d'un agent autonome comme:

"*An autonomous agent is a system situated within and a part of an environment that sense that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it sense in the future*."

Dans la théorie du système multi-agent, l'agent autonome est également une entité cognitive. Cela signifie que l'agent n'est pas seulement dans une interaction continue avec l'environnement dans lequel il est situé, mais aussi qu'il vit dans un monde multi-agent et qu'il est aussi en interaction avec les autres agents. A partir de cette idée, Wooldridge et Jennings [Wooldridge et Jennings'95] définissent un agent non seulement par rapport à l'interaction avec l'environnement et à la notion que l'agent peut avoir un comportement autonome dirigé par un but, mais également par le fait que l'agent a des capacités sociales. Les agents se rendent compte de l'existence d'autres agents et ils agissent un sur l'autre par l'intermédiaire d'un certain type de langage de communication et d'un protocole d'interaction entre agents.

En partant de cette définition générale de l'agent, par la suite nous présenterons : *i.)*dans *la dimension individuelle* les différents types d'agents et quelles sont les caractéristiques qui font qu'un agent peut être considéré comme participant dans une négociation ; et *ii.)*dans *la*

dimension sociale seront identifiés les aspects pris en compte par les systèmes multi-agent implémentant un processus de négociation.

2.1.1. Dimension individuelle

La définition et les caractéristiques présentées auparavant constituent une base commune pour tous les agents autonomes, mais en fonction de la façon dont les agents se comportent, ils peuvent être groupés dans trois types d'agents : *délibératifs, réactifs* et *hybrides*.

Les *agents délibératifs* ou « rational agents » peuvent être perçus en tant qu'ayant un comportement le plus semblable aux humains. Ces agents disposent d'un modèle de monde symbolique construit par eux même et dans lequel ils vivent. Ils disposent également de quelques mécanismes mentaux qui leur permettent de faire des inférences sur ce monde afin de décider leurs actions [Wooldridge et Jennings'95].

Les *agents réactifs* disposent d'un certain comportement de base qui leurs permet de réagir aux changements spécifiques de l'environnement. Manipulant des rationnements vraisemblablement simples, ce type d'agent est le plus capable d'avoir un comportement logique et tractable qui dépend seulement de leurs perceptions sur l'environnement. La grande différence entre les agents délibératifs et ceux réactifs est que les agents délibératifs construisent et maintiennent un modèle interne du monde, et les agents réactifs agissent selon leurs perceptions sur un monde externe, parce que "le monde est le seul meilleur modèle" [Brooks'86].

Les *agents hybrides* essayent d'unifier les deux types d'agents. Brièvement l'idée fondamentale est de combiner les inférences rapides et simples avec des buts à long terme et un certain control délibératif [Muller'96].

Du point de vue du processus de la négociation, les parties impliquées ont la capacité de réagir aux changements mais également de prévoir les changements. Ceci implique que les agents doivent avoir la capacité de raisonner sur les différentes solutions qu'ils ont pour continuer la négociation. Les agents doivent être capables aussi de communiquer leurs préférences afin de pouvoir évaluer et faire un choix parmi les différentes alternatives qui se présentent. Ainsi, les plus appropriés à ce type de processus sont *les agents délibératifs*.

La modélisation du processus de négociation avec des agents délibératifs est très similaire ou plutôt essaie d'être similaire avec la négociation humaine. Afin de pouvoir négocier, les agents possèdent des connaissances sur les produits et les attributs qui les intéressent et éventuellement sur les valeurs accordées par les autres agents. Ces informations sont utilisées à chaque instant par l'agent pour évaluer des anciennes propositions ou élaborer de nouvelles propositions. Ces mécanismes de raisonnement sont modélisés et paramétrés par des stratégies de négociation que l'agent va suivre tout au long du processus de négociation [Lomuscio et al.'01].

2.1.2. Dimension sociale

De la même manière vue dans l'approche pluridisciplinaire sur la négociation, dans l'approche SMA *la dimension sociale* fixe des rôles particuliers pour les participants à une négociation. Les rôles des participants dépendent du modèle de négociation choisi et fixent les contraintes sur la conduite des participants pendant la négociation afin de respecter le modèle envisagé. Il y a une forte similitude avec les rôles présents dans les négociations entre humains et ceux dans les négociations modélisées par des agents. Dans les SMA les rôles dans une négociation sont les mêmes que ceux dans les négociations réelles et ils induisent les mêmes contraintes: *vendeur/acheteur*, *initiateur/invité* et *médiateur/participant active*. Comme la négociation modélisée dans les SMA est faite souvent dans le cadre du développement des infrastructures pour l'aide du commerce électronique, les rôles les plus employés sont *d'acheteur* et *vendeur*.

Si les types des rôles possibles dans une négociation sont similaires, par contre au niveau de deux approches le nombre de participants impliqués au même moment dans une même négociation est très distinct. En effet, si dans les négociations de types face-à-face le nombre des participants est limité à un nombre assez réduit, par contre par le biais de l'infrastructure informatique et de l'Internet le nombre de participants possibles dans une même négociation augmente considérablement. De cette manière, le nombre de participants n'a pas une limite fixe et il dépend seulement des capacités computationelles et du temps de réponse du système gérant le processus de négociation.

Selon le nombre des participants impliqués dans une négociation nous pouvons ainsi distinguer différents types de négociation :

- *Négociation bilatérale :* Les négociations bilatérales désignent des négociations entre deux individus qui sont prêts à échanger des propositions sur un objet auquel ils n'attribuent pas la même valeur (par exemple ils ont des prix différents pour l'objet à négocier).
- *Négociation un à plusieurs :* La négociation « un à plusieurs » consiste en général dans une suite d'interactions entre un vendeur (acheteur) et un ou plusieurs acheteurs (vendeurs). En plus des caractéristiques imposées par le rôle, l'agent qui négocie avec plusieurs autres agents a sur les derniers un pouvoir et une influence déterminants. Ceux ci sont déterminés par le fait que cet agent est le seul qui a une connaissance globale sur la négociation et corrobore dans le même temps le fait que les autres agents ne collaborent pas entre eux, étant souvent en compétition.
- *Négociation plusieurs à plusieurs :* La négociation « plusieurs à plusieurs » consiste en général dans une suite d'interactions de type diffusion (broadcast). Souvent dans ce type de négociation les agents ont le même rôle et la négociation est finalisée si tous sont d'accord avec le résultat.
- *Négociation multi-bilatérale :* La négociation « multi-bilatérale » peut être considérée comme une négociation un à plusieurs mais dans ce cas les interactions entre les agents sont faites deux à deux créant différentes négociations bilatérales. Cela fait que les agents ont des rôles bien définis et qui peuvent être très différents un par rapport aux autres. Egalement, entre les différentes négociations bilatérales il y a plusieurs relations de dépendance, ce qui fait que le résultat de la négociation est déterminé par les résultats des certaines négociations bilatérales qui la composent. La négociation « multi-bilatérale » est souvent utilisée dans le commerce électronique de type business to business où les propositions sont faites en fonction du profil du partenaire.

Ainsi, à la différence de la négociation humaine, dans l'approche multi-agent la rationalité de l'agent et la cardinalité des agents impliqués sont très importants, ces dimensions font augmenter le nombre des computations et des interactions et donc influencent l'efficacité du système.

2.2. Contexte de Négociation

Dans le premier chapitre sur l'approche pluridisciplinaire de la négociation, nous avons présenté le contexte en fonction de trois dimensions : la culture, le pouvoir et le temps. Sur ces dimensions et la manière dont elles influencent le processus de négociation, très peu de travaux ont été menés dans l'informatique ou dans la communauté SMA.

2.2.1. Culture

Les travaux de Kersten [Kersten et Noronha'97] [Kersten et Noronha'98] sont les seuls qui étudient l'impact de la culture sur le processus de négociation et les caractéristiques de la négociation inter-culturelle. Plusieurs aspects de la culture sont proposés, mais trois d'entre eux influencent principalement la modélisation du processus de négociation dans un système informatique:

- *individualisme/collectivisme* : dans les cultures collectivistes les buts d'un individu sont englobés dans les buts du groupe ; par contre dans les cultures individualistes il y a une forte distinction entre les buts personnels et ceux du groupe.
- *le statut* : des cultures où la hiérarchie sociale impose des restrictions fortes sur la communication entre des statuts différents et d'autres cultures où ces restrictions sont plus faibles.
- *le temps* : qui, reprenant l'idée présentée auparavant, divise la culture dans une culture *mono-chronique* et une culture *poly-chronique (*voir section 1.2.3*)*.

Dans [Kersten et al.'97] le système *Inspire* est utilisé pour gérer et modéliser le processus de négociation afin d'étudier des négociations dans lesquelles les appartiennent à des pays et à des cultures différentes. Les résultats de l'expérience confirment non seulement que « la culture influence la négociation par ses effets sur la communication » [E. Wertheim], mais suggèrent également une plus large portée de ces influences. Donc, les négociations électroniques ne sont pas en dehors des influences culturelles. La culture affecte premièrement les espérances et les comportements des participants humains, qui à leur tour affectent par leurs contraintes les outils informatiques et de communication utilisés.

D'autres constatations sont assez surprenantes. Dans les négociations tête-à-tête les sujets peuvent modifier leurs comportements et leurs attitudes selon leurs perceptions de la

culture des autres participants. Cependant, dans les négociations électroniques anonymes, les participants ne peuvent pas compter sur ces indices et ainsi leurs comportements dans une négociation ne peuvent pas être influençables par les différentes cultures des autres participants.

Les influences se portent non seulement au niveau des attentes des participants impliqués mais aussi sur la manière de gérer le processus de négociation et les stratégies de négociation. L'influence de la culture sur le processus de négociation fait que certains systèmes de négociation commencent la négociation sur les attributs les plus importants pour se poursuivre sur les moins importants.. Dans d'autres systèmes, la négociation peut débuter avec un accord décrit en termes généraux et les attributs importants sont abordés par la suite. De la même manière, l'influence de la culture sur les stratégies de négociation fait que certaines stratégies de négociation préconisent de commencer avec des propositions très peu acceptables par la contre-partie, alors que d'autres stratégies préconisent de commencer avec des propositions facilement acceptables dès le début. Les stratégies pour continuer la négociation peuvent être aussi cataloguées en *malléables* ou *fermées* en fonction des changements qui sont faits dans des propositions successives [Calantone et al.'98]. Donc la culture influence non seulement les négociations tête-à-tête entre les hommes mais aussi les systèmes informatiques utilisés pour la négociation électronique.

2.2.2. Pouvoir

La notion de *pouvoir* est identifiée souvent dans les systèmes multi-agent à la notion d'autorité d'un agent et elle est relative à un autre agent ou un group d'agents [Santos et al.'97] [Santos et Carmo'96] [Governatori et al.'02]. Dans les travaux sur l'organisation dans un système multi-agent l'autorité est une notion liée au rôle de l'agent. Dans ce cas, le rôle n'est pas relatif seulement au processus d'interaction en fixant de contraintes sur la communication, mais il fixe également une relation d'autorité entre deux ou plusieurs agents. L'autorité permet à un agent d'en commander un autre.

Dans le système multi-agent implémentant le processus de négociation, la notion de pouvoir peut être aussi identifiée aux informations qu'un agent a sur les intentions et les préférences des autres parties [Jones'03]. Ces informations peuvent être utilisées pour le choix à chaque instant de la stratégie de négociation à suivre [Lomuscio et al.'01]. Donc, de cette manière la notion de pouvoir souffre un changement, en passant d'une dimension un peu

abstraite dans l'approche pluridisciplinaire à une dimension plus mesurable dans l'approche SMA permettant de caractériser concrètement les processus de négociation.

2.2.3. Temps

En dehors des caractéristiques culturelles présentées auparavant, le temps est souvent utilisé dans les systèmes multi-agent modélisant la négociation afin de fixer des contraintes temporelles au niveau des mécanismes de communication et des mécanismes de décision utilisés dans le processus de négociation.

Le passage du temps peut être perçu de deux manières différentes en fonction du scénario de la négociation [Kraus'01]. Premièrement, dans un scénario symétrique, le passage du temps est perçu par tous les participants comme une contrainte sur la finalisation du processus de négociation où ils cherchent tous de trouver un accord le plus vite possible. Deuxièmement, dans un scénario asymétrique, le passage du temps est bénéfique pour seulement une des parties impliquées dans la négociation. En fonction de ces deux aperçus, des différents mécanismes de communication et de décision peuvent être choisis.

Une approche est de paramétrer directement le protocole de négociation en fonction des contraintes temporelles. Ces contraintes peuvent être fixées sur les types des messages qu'un agent peut ou ne peut pas envoyer à un certain moment du temps.

Le temps peut aussi modéliser le raisonnement employé dans une négociation. Ainsi, la stratégie utilisée pour négocier peut être en fonction du temps [Faratin'00] [Faratin et al.'98]. En utilisant ces stratégies, les agents génèrent leurs propositions et contre propositions en se basant sur la durée de l'intervalle de temps qui reste pour la négociation (avant d'atteindre la date limite fixée à l'avance).

Finalement, le temps peut être ajouté aux notions d'engagement et de pre-engagement trouvées à la fin d'une négociation [Fornara et Colombetti'02]. Les contrats résultants après une négociation peuvent aussi avoir un état temporel qui contraint leur temps de vie.

Donc, le temps est utilisé afin de paramétrer *l'objet de négociation, le protocole de négociation* et *la stratégie de négociation.* Ces trois dimensions du processus de négociation seront détaillées par la suite.

2.3. Mécanisme de négociation

Si les dimensions caractérisant les mécanismes de négociation utilisés dans l'informatique sont les mêmes que dans l'approche pluridisciplinaire, en revanche, leurs aspects sont assez spécifiques étant dépendants de la mise en œuvre du cycle de vie du processus de négociation par les SMA. Par rapport aux deux étapes de la négociation pluridisciplinaire (i.e. pré-négociation et négociation – voir section 1.3.) le cycle de vie mis en place par des SMA est composé également d'une première étape qui fixe le cadre de négociation. En revanche, le système informatique ne s'arrête pas avec la proposition d'une solution pour la négociation mais il essaie de la définir sous forme d'un contrat et même de surveiller l'exécution du contrat. Ainsi le cycle de vie d'une négociation peut être structuré très simplement en trois étapes [Mullen et Wellman'98]: 1.) rencontre des participants potentiels, 2.) négociation des termes de l'échange et 3.) exécution de la transaction.

La première étape est influencée seulement par le nombre des participants actifs dans une négociation. Cette première étape est souvent mise en œuvre par des systèmes multi-agent modélisant des places de marché avec un système de type brokering [Strobel et Stolze'01] où des agents peuvent se rencontrer afin de participer dans une négociation *un à plusieurs* ou *plusieurs à plusieurs* (voire section 2.1.2.). Une approche particulière est faite dans une *négociation bilatérale* (un à un) ou *multi-bilatérale* où une identification plus précise des partenaires choisis est nécessaire afin d'identifier les meilleurs partenaires. Ce choix est fait par les agents impliqués dans la négociation et participant à la décision finale. Les agents gèrent et raisonnent sur une représentation interne des partenaires possibles [Faratin'00].

La deuxième étape est plus complexe, étant fortement influencée par le nombre des participants impliqués dans une même négociation et par les informations manipulées, notamment par la structure de l'objet de négociation. En reprenant notre structuration des négociations en fonction du nombre des participants (voir section 2.1.2.), cette deuxième étape peut être modélisée de différentes façons.

Ainsi, dans une négociation « bilatérale » la complexité du processus est donc influencée principalement par la structure de l'objet de négociation. Pour trouver une seule solution acceptable pour les deux parties, plusieurs systèmes supposent l'existence d'un arbitre qui décide quelle est la meilleure solution [Strobel'00]. Une autre approche est de modéliser la négociation comme un jeu séquentiel avec des propositions et des contre-propositions de la part des deux parties [Pruitt'75] [Jennings'98].

Avec une négociation « un à plusieurs », étant souvent des enchères [ebay], la modélisation est influencée par le nombre des participants et pas par la complexité de l'objet à négocier. Ces négociations de type enchère imposent l'existence d'une institution centralisée chargée de coordonner l'ensemble des échanges avec des règles très strictes pour garantir l'impartialité dans l'interaction [Sycara'91]. Cependant, le comportement de cette instance peut être simulé par l'agent initiateur, induisant une influence et un pouvoir déterminant pour les autres agents qui participent à la négociation.

Dans une négociation « multi-bilatérale », par rapport à la négociation « un à plusieurs », la complexité n'est pas augmentée par le nombre des participants mais par le nombre des interactions entre les participants et les dépendances existantes entre elles. Les négociations « multi-bilatérale » sont construites pour modéliser une progression en parallèle de plusieurs négociations de type un à un.

Finalement, dans une négociation « plusieurs à plusieurs » la complexité du processus est maximale et afin de garantir une efficacité acceptable du processus de négociation les protocoles de communication choisis sont généralement rigides et lourds. Ils doivent en effet garantir que tous participent à la négociation et que tous sont d'accord avec la proposition finale. Ce type de négociation est adopté par des systèmes qui mettent en place des enchères doubles (un agent peut être acheteur ou vendeur en même temps [Wurman et al.'98]) ou des coalitions avec une négociation par vote [Tsvetovat et al.'00].

La dernière étape du processus de négociation est plus liée à la théorie des contrats qui traite essentiellement des relations bilatérales dans des situations où les parties en présence doivent accomplir certaines tâches en tenant compte de l'existence d'événements aléatoires indépendants de leur volonté.

Ainsi, le processus de négociation est complexe et peut avoir multiples formes d'implémentation en utilisant les SMA. Cette complexité du processus de négociation implémenté par des SMA est aussi visible au niveau des mécanismes utilisés pour le mettre en place : *l'objet de négociation*, *le protocole de négociation* et *la stratégie de négociation*.

2.3.1. Objet de négociation

En reprenant les caractéristiques de l'approche pluridisciplinaire sur la négociation, dans l'informatique un très gros travail est fait sur la structuration des informations manipulées au cours d'une négociation et surtout sur la description de l'objet de négociation.

Dans l'approche informatique sur la négociation, l'objet de négociation décrit les différents attributs autour desquels les accords doivent être cherchés.

En fonction du nombre des attributs qui le compose, l'objet de négociation peut être :

- single attribut : l'objet de négociation contient seulement un attribut, le plus souvent cet attribut est le prix [Maes et Chaves'96]. L'objet de négociation peut être aussi générique, il peut englober plusieurs attributs mais qui ont déjà des valeurs fixes et qui ne sont pas négociables. C'est le cas des négociations sur des ressources disponibles ou pour l'allocation d'une tâche prédéfinie [Mailler et al.'01].
- multiple attributs indépendants: l'objet de négociation est composé de plusieurs attributs, tel que : le prix, le temps, la qualité, les délais, les pénalités, etc. Dans ce cas la négociation peut concerner à un moment de temps la totalité des attributs ou seulement certains d'entre eux [Jennings'93]. Les négociations sur seulement une partie des attributs n'influencent pas les négociations futures sur les autres attributs restantes.
- multiple attributs inter-dépendants: l'objet de négociation est aussi composé des plusieurs attributs et ces attributs ont des dépendances entre eux. Dans ce cas le processus de négociation doit gérer aussi les dépendances existantes entre les attributs [Fatima et al.'04]. Ce type de négociation est souvent modélisé comme une satisfaction de contraintes avec une « connaissance commune » sur les dépendances [Oliveira'01] ou avec la possibilité de mettre en place des dépendances particulières et privées pour chaque partenaire.

L'objet de négociation peut être considéré comme une structure de données qui définit pour un agent ou un group d'agents le but de la négociation. En utilisant l'approche du Jennings avec l'objet décrit par plusieurs attributs, ceux-ci identifient justement un espace de négociation, mais dans cet espace une infinité de solutions est possible. Ainsi, le processus de négociation doit réduire cette espace à un nombre finit des solutions et, si c'est possible, à une seule solution acceptable par les participants de la négociation.

De cette manière, au début de la négociation l'objet de négociation est décrit par les attributs composants et qui sont parfois associés avec certaines valeurs. Au cours de la négociation les propositions et les contre-propositions sont des instanciations de l'objet de négociation, par le fait que les attributs sont toujours associés avec des valeurs. L'échange des

propositions et des contre-propositions peut être réitéré plusieurs fois afin d'arriver à la fin de la négociation, à un accord qui est un objet de négociation complètement instancié.

Ainsi, des opérations différentes peuvent être définies en plus de la structure d'un objet de négociation. Dans le cas le plus simple, un agent fait une proposition dans laquelle les attributs négociés et les valeurs associées sont fixes. Les autres agents ont seulement la possibilité d'accepter ou de rejeter la proposition. A un niveau plus flexible, les agents ont la possibilité de répondre avec des nouvelles propositions qui contiennent les mêmes attributs de négociations mais avec des valeurs différentes.

Finalement, au cours du processus de négociation, les agents peuvent changer dynamiquement la structure de l'objet de négociation en ajoutant ou en supprimant un ou plusieurs attributs. Ces types des contraintes ou des permissions sont souvent indépendantes de l'objet de négociation et elles sont fixées par le protocole de négociation utilisé.

2.3.2. Protocole de négociation

Un protocole de négociation est un modèle formel souvent défini comme un ensemble de règles qui contrôlent le processus de négociation. Comme dans l'approche pluridisciplinaire (voir section 1.3.2.), les règles imposées par un protocole couvrent la plus part des caractéristiques d'une négociation, comme: i) les rôles des agents participants (si l'agent et participant ou médiateur, s'il est vendeur ou acheteur, s'il est initiateur ou invité) ; ii) les différents états de la négociation (tels que l'acceptation des propositions ou la fin de la négociation) ; iii) les évènements qui peuvent être à l'origine du passage de la négociation d'un état à l'autre (tel que l'envoie d'une nouvelle proposition ou l'acceptation d'une proposition) ; et iv) les différentes actions qu'un participant peut effectuer à chaque instant (qui peut envoyer un message, à qui et à quel moment).

Afin de répondre aux attentes des utilisateurs, un protocole de négociation doit répondre à quelques critères. Ainsi, un protocole de négociation doit être : distribué, instantané, efficace, simple, symétrique et stable [Krause'95] :

- distribué : le protocole de négociation doit être distribué. Il ne doit pas y avoir une entité centrale ou un agent qui contrôle le protocole de négociation.
- instantanéité : les conflits doivent être résolus dans un temps fini et raisonnable.

- efficacité : le résultat de la négociation (l'accord) doit être réalisé par une sélection mutuelle et il doit être efficace, c'est-à-dire un résultat de type Pareto-Optimal[6] ou au mois très proche.
- simplicité : le protocole de négociation doit être simple et efficace. Il doit être court et il doit utiliser un nombre raisonnable de communications et de ressources (i.e., les mécanismes de négociation ne doivent pas être coûteuses côté ressources : processeur et réseaux).
- symétrie : les mécanismes de négociation doivent être équivalents/impartiaux pour tous les agents, c'est-à-dire qu'ils ne doivent pas privilégier tel ou tel agent pour n'importe quelle raison qu'elle soit. La symétrie implique que dans n'importe quelle situation, le remplacement d'un agent par un autre agent qui lui est identique ne doit pas changer le résultat de la négociation.
- stabilité : il doit y avoir des points d'équilibre distinguables pour le protocole de négociation. Le protocole de négociation doit permettre aux agents d'atteindre ces points au cours du processus de négociation et ceci en définissant des stratégies simples. Une stratégie simple implique qu'elle est implantable sur une machine (peut être utilisée par un agent) et que les agents sont capables de l'utiliser en temps raisonnable.

Certaines de ces caractéristiques peuvent être considérées comme normales et elles sont l'héritage des caractéristiques d'une négociation (*distribuée, instantanée, efficace, stable*). Le processus de négociation est par définition un processus distribué, dans lequel chaque partie de la négociation évalue les informations de sa propre perspective afin de trouver, dans un temps fini, un accord qui est acceptable par tous.

Par contre, la notion de *simplicité* d'une négociation et par la suite d'un protocole de négociation peut être facilement combattue. Les négociations primaires comme les négociations de type *take-it-or-leave-it*, les systèmes de vote ou les enchères simples [ebay] sont implémentées avec des protocoles assez simples. Dans ces cas les protocoles simples sont traditionnellement représentés en tant que machines d'état fini dans lesquelles les arcs de transition indiquent les actes communicatifs à employer.

[6] Un accord est dit Pareto-Optimal s'il n'y a pas un autre accord qui le domine ; c'est-à-dire qu'il n'existe pas un autre accord qui améliore l'utilité de certains agents sans autant influencer négativement sur l'utilité de l'un des autres agents.

En revanche, les négociations peuvent être très compliquées, comme les enchères doubles, les enchères combinées, les négociations multi-niveaux, les négociations multi-attributs ou les négociations par argumentation. Dans ces cas les protocoles simples ne sont pas très efficaces.

Partant de cette observation beaucoup de travaux ont essayé de combler les lacunes des protocoles simples. Trois directions sont les plus connues.

Une des directions utilise les protocoles simples en essayant de combiner ces protocoles. L'idée fondamentale de cette approche est de représenter les protocoles comme des ensembles des états et les transitions entre eux sont modélisées comme des actions. A partir de cette représentation d'état des protocoles, la composition des protocoles est faite en fonction des actions employées par les protocoles et qui sont plus génériques que les différents actes de langage proposés dans les protocoles traditionnels [Vitteau et Huget'03].

La deuxième direction explore l'idée que les états d'un protocole de négociation sont plus importants que les actes communicatifs [Chang et Woo'94] [Fornara et Colombetti'03]. Ce type d'approche propose une manière d'exécuter des protocoles en les compilant dans des "machines d'engagement" obtenues en paramétrant d'une certaine manière les machines d'état fini des protocoles. Cela fait que le passage d'un protocole à un autre est fait en termes des engagements qui sont par la suite maintenus ou transformés par les différentes actions que les protocoles proposent.

Tandis que ces deux approches essaient de modifier les protocoles, une autre approche utilise les protocoles dans leur forme commune et en fonction des types de problèmes à résoudre des règles d'interaction sont ajoutées au dessus de ces protocoles (voir section 3.2). Cette approche sera plus détaillée dans le chapitre suivant de l'état de l'art.

Finalement, la *symétrie* d'un protocole de négociation peut être vue plus qu'une caractéristique induite par la négociation électronique. Dans une négociation les protocoles n'ont aucune notion sur les caractéristiques d'implémentation d'un agent, par contre un rôle est attaché à chacun des agents entrant dans la négociation. Par la suite, ce sont les caractéristiques du rôle de l'agent et pas ses caractéristiques internes qui seront utilisées par le protocole de négociation afin de gérer la communication entre agents et de limiter les actions possibles pour chacun d'entre eux.

En plus de ces caractéristiques, dans la négociation électronique, les protocoles d'interaction utilisés sont aussi choisis en fonction des autres dimensions du processus de négociation comme le nombre des participants impliqués et le nombre des attributs caractérisant l'objet de négociation. Le choix du protocole a une forte influence non seulement

sur le processus de communication entre les participants mais aussi sur les stratégies de négociation que chaque participant utilise. Comme nous les avons présentés dans la description pluridisciplinaire de la négociation des nombreux types de stratégies de négociation existent et la plus part d'entre eux peuvent être aussi implémentés par des SMA.

2.3.3. Stratégie de négociation

Pendant le déroulement de la négociation, l'agent est amené à prendre plusieurs décisions : de la façon de construire des nouvelles propositions ou d'évaluer les propositions reçues ou du meilleur moment d'accepter une proposition et d'arrêter la négociation. Donc, l'agent a besoin d'avoir un module de raisonnement pour lui permettre d'atteindre le ou les objectifs recherchés.

Le module de raisonnement emploie des stratégies de négociation afin de caractériser de manière structurée et mathématique, le comportement général de l'agent au cours du processus de négociation. Le degré de complexité de ce modèle de prise des décisions est lié au protocole de négociation utilisé, à la nature de l'objet de la négociation et le choix des opérations qui peuvent avoir lieu durant le processus de négociation [Jennings'91].

Auparavant dans l'approche pluridisciplinaire sur la négociation (voire section 1.3.3.), nous avons détaillé que la théorie moderne de la négociation est articulée sur le fait qu'une négociation constitue un jeu à somme nulle ou non-nulle. L'art de la négociation consiste donc à faire céder les interlocuteurs sur la ligne principale d'opposition (un prix, par exemple) ou à trouver des arrangements extérieurs à cette ligne qui apporteront beaucoup à l'un sans coûter trop cher à l'autre. Comme ces types de problèmes ont été beaucoup étudiés dans la théorie du jeu avec un appareil mathématique très élaboré, les stratégies de négociation utilisées dans les systèmes multi-agent sont en majorité issues de cette approche.

Dans la théorie du jeu, le mécanisme de raisonnement utilisé dans une négociation consiste dans un choix de stratégie que chacun des participants fait en fonction de ses prédictions sur les autres participants (jeux coopératifs vs jeux non-coopératifs). Au lieu de décider les mouvements justes avant de les réaliser, un participant peut décider, avant le début du jeu, quel mouvement il peut faire dans chacune des situations possibles qui peuvent exister. Cette approche est nommée stratégie. Dans ce cas, la stratégie est un ensemble d'instructions pour jouer le jeu du premier mouvement jusqu'au dernier. En énumérant les différentes possibilités pour un participant de jouer le jeu, noua pouvons obtenir toutes les stratégies pour

ce participant. Pour chacune de ces stratégies le participant a un gain. Dans la théorie du jeu, l'objectif est de trouver la meilleure stratégie pour un certain participant. La valeur du gain pour chaque participant dépend donc des stratégies choisies par tous les participants.

La théorie du jeu cherche à trouver une stratégie optimale dans une situation particulière. Lors de l'analyse du jeu, la question centrale est de savoir s'il existe une telle stratégie. Dans la plupart des jeux, il est possible de trouver la meilleure stratégie. Néanmoins certaines limitations existent : la valeur de la fonction du gain ne peut pas être précisément présentée ainsi que l'existence de plusieurs points de Pareto optimales, l'existence de critères différents pour les participants donc le jeu n'a pas des informations parfaites en avance. De plus, le coût du calcul augment de façon combinatoire en fonction du nombre de participants et avec un espace de négociation multicritères [Sandholm'99]. Cette approche n'est donc pas la meilleure pour les systèmes où chaque participant manipule de manière confidentielle certaines informations (ex. les critères d'évaluation, la fonction du gain), et où le temps de réaction est très limité et donc le système n'a pas la possibilité de calculer et de prévoir tous les mouvements futurs.

Gardant également la notion du jeu mais à un autre niveau d'abstraction, d'autres approches pour la modélisation du processus de négociation sont apparues. Donc, la négociation entre agents peut être vue comme un jeu qui prend en compte les règles spécifiées par les protocoles de négociation. Cette approche est très utilisée dans les places de marché électroniques qui utilisent avec prédilection les protocoles de type enchères pour gérer le processus de négociation. Les places de marché et les enchères sont des mécanismes de négociation qui garantissent un résultat optimal même avec des informations incomplètes sur les fonctions d'utilités et les préférences des agents impliqués dans le processus.

Dans les enchères classiques un vendeur essaye de vendre un objet à un ensemble de participants et il cherche à obtenir le prix plus élevé. En revanche, les autres participants ont des intérêts orthogonaux, dans la mesure où ils essayent d'acheter l'objet pour le plus bas prix. Ce type de négociation est très utilisée pour la gestion des relations commerciales de type B2C (business to consumer). Les formes les plus connues sont les catalogues électroniques, la vente en ligne [ebay] et les places de marché [Maes et Chaves'96].

Dans ce type de modèle d'interaction, le processus de négociation est modélisé comme une séquence d'offres et contre-offres. Le modèle fait l'hypothèse que les fonctions d'utilité de chaque participant sont fixées et connues, qu'une zone de compromis existe et elle est stable dans le temps. Mais ces types des suppositions ne sont pas très raisonnables dans le commerce

électronique de type B2B (business to business) où les entités sont autonomes et elles ne veulent par faire partager certaines de leurs informations ou préférences.

Dans la mesure où dans ce type de commerce les participants sont des entreprises autonomes et l'objet à négocier est souvent une tâche à sous-traiter [Oliveira'01], le processus de négociation devient plus complexe: l'objet de négociation est caractérisé par plusieurs attributs; les partenaires peuvent changer en cours de route, le protocole devient aussi plus complexe [Kumar et al.'02] et l'environnement est dynamique et ouvert. Donc, la stratégie de négociation, elle aussi devient plus complexe afin de s'adapter de manière dynamique à la complexité de la négociation et à un environnement fluctuant. La stratégie ne cherche pas de maximiser une seule fonction d'utilité, mais de gérer le choix parmi plusieurs fonctions d'utilité calculées des manières différentes. La stratégie de négociation est une composition des plusieurs *tactiques de négociation*, chaque tactique peut être en fonction des paramètres différents, comme les trois tactiques proposées par Faratin [Faratin'98]:

- tactique basée sur les ressources – en utilisant cette tactique, les agents génèrent les offres en se basant sur la quantité des ressources disponibles.
- tactique basée sur le temps – en utilisant cette tactique, la construction des différentes offres est faite en fonction d'un moment de temps et de la durée limitée d'une négociation.
- tactique basée sur le comportement – avec cette tactique, chaque agent essaie d'imiter le comportement de l'agent qui est en face de lui en se basant sur les offres précédentes de l'agent à imiter.

Le modèle proposé par Faratin suppose seulement deux participants. Les scénarios avec plus de deux participants sont caractérisés par la possibilité de réaliser des coalitions [Newmann et al.'44] [Bergstresser et Yu'77]. La mise en place des coalitions réduit le nombre de participants. En appliquant le processus de manière itérative ce nombre peut être réduit à deux. Toutefois, les problèmes apparaissent au moment de décider comment les coalitions sont mises en place et ensuite comment le gain d'une coalition est distribué entre ses participants.

Finalement, nous pouvons constater que les stratégies de négociation dépendent fortement des autres dimensions du processus de négociation, comme le nombre des participants, les contextes de négociation (temps, cultures) ou les objets et les protocoles de négociation. Donc, la conception d'un bon système de négociation nécessite la prise en compte de toutes les dimensions présentées et également des dépendances existantes entre elles.

2.4. Synthèse

Cette partie de l'état de l'art sur le processus de négociation modélisé par des systèmes multi-agent nous a permis de présenter quels sont les aspects spécifiques aux SMA des dimensions comme *l'objet de négociation*, *le protocole de négociation* et *les stratégies de négociation* et aussi de rajouter de nouveaux paramètres au processus de négociation, comme *le nombre de participants*.

Dans le tableau suivant (voir Tableau 1.) nous avons mis en exergue au sein des différents travaux le nombre des participants impliqués dans une négociation (un à un, plusieurs pour un, plusieurs vers plusieurs) et le nombre d'attributs de l'objet de négociation (un seul ou plusieurs attributs qui sont soit indépendants soit dépendants).

Que négocier ? / **Qui négocie ?**	***Un seul attribut***	***Multiple attributs indépendants***	***Multiples attributs dépendants***
Négociation bilatérale	Attribut = prix	Attribut = plusieurs critères d'achat	Attribut = plusieurs attributs pour décrire un objet
Négociation un à plusieurs	Attribut = prix	Enchère avec un ensemble fixe des attributs	Résolution de la négociation par médiateur
Négociation multi-bilatérale	Attribut = prix Attribut = allocation d'une ressource	Négociation avec un ensemble de partenaires par un médiateur ou par l'agent lui-même	Négociation menée avec des protocoles gérant plusieurs itérations
Négociation plusieurs à plusieurs	Attribut = prix Enchère double – un agent peut être acheteur ou vendeur	Des enchères sur plusieurs objets (combinatorial auctions)	Système de coalition avec une négociation circulaire ou par vote

Tableau 1 : La complexité du processus de négociation en fonction du nombre des participants et de la structure de l'objet de négociation

Nous pouvons ainsi distinguer sur les lignes du tableau (voir Tableau 1) :

– les négociations bilatérales se déroulant sur des objets de négociation qui peuvent être composés d'un seul attribut ou de plusieurs attributs interdépendants. Les caractéristiques de ce type de négociation sont liées au mécanisme de choix du partenaire. Souvent, est impliqué un troisième participant ayant des fonctionnalités de match-making pour trouver le meilleur partenaire.

– les négociations impliquant un agent contre plusieurs. La plupart de ces négociations utilisent un protocole de type enchère. Elles peuvent également se dérouler sur des objets de négociation caractérisés seulement par le prix ou sur des objets caractérisés par plusieurs attributs interdépendants. Le système le plus connu utilisant ce type de mécanisme est eBay [ebay].

– les négociations impliquant plusieurs négociations de type un à un. Ce type de négociation est souvent utilisé dans les interactions business-to-business où sont impliqués de nombreux partenaires. La confidentialité des interactions est souvent nécessaire dans ce contexte. Elle est garantie par le mécanisme de négociation un à un où les propositions ne sont connues que par les participants impliqués directement.

– les négociations de type plusieurs à plusieurs sont aussi utilisées dans les interactions de type business-to-business avec l'implication de multiples partenaires. En revanche, tous les participants sont impliqués dans une seule et même négociation. Ce type de négociation est fortement coûteuse en temps du fait de la diffusion des offres et du fait des algorithmes assurant le consensus de la prise de décision.

D'un point de vue informatique, la complexité du processus de négociation augmente avec le nombre de participants impliqués mais aussi avec la complexité de l'objet de négociation en terme du nombre et de la nature des attributs. Comme nous venons de le voir, cette complexité est accrue par de nouvelles dimensions :

- *nombre de cycles* au sein des différents protocoles des négociations utilisés pour gérer la négociation.
- *nombre de stratégies* et *d'objectifs* à satisfaire : chacun des participants peut décider de manière autonome quelle est sa stratégie et quel est son but pour la négociation dans laquelle il est impliqué.

Ces dimensions sont fortement dépendantes l'une de l'autre. La négociation automatique comme son analogue humain, est donc un processus très complexe avec des représentations très différentes.

3. Coordination dans les systèmes de négociation

Dans la communauté multi-agent, la négociation elle-même est souvent caractérisée comme un mécanisme de coordination permettant d'orienter les actions des agents tout en leur laissant la liberté de choix. Dans les deux premiers chapitres de l'état de l'art nous avons montré les caractéristiques pluridimensionnelles du processus de négociation. Intuitivement, ces différentes caractéristiques influencent les interactions qui sont menées dans une négociation. De manière abstraite, la négociation est une structuration d'interactions qui nécessite elle-même un modèle de coordination afin d'augmenter le résultat de la négociation ainsi que les performances du système de négociation.

La coordination n'est nullement limitée au domaine de l'informatique. Malone et Crowston [Malone et Crowston'94] lui attachent un domaine de recherche émergeant avec un intérêt multidisciplinaire et jouant un rôle clé dans divers domaines comme l'économie, la recherche opérationnelle, les sciences des organisations ou la biologie. Plusieurs définitions de la coordination ont été proposées, mais probablement la plus générale reste celle que Malone et Crowston ont proposée : « *La coordination est la gestion des dépendances entre des activités.* ».

Ainsi, de manière générale, la théorie de la coordination essaie d'identifier quels types d'activités peuvent être coordonnées et quelles sont les dépendances génériques qui peuvent exister entre elles. C'est sur cette base théorique que s'appuie notre étude. Cette base est fortement pluridisciplinaire et donc par la suite nous l'adaptons aux domaines des systèmes informatiques de négociation.

Partageant la réflexion « *Research on coordination possibly represents the most un-coordinated activity in the history of computer science* » [Omicini et al.'03], dans ce chapitre nous commençons par détailler de manière générale la gestion des interactions au sein d'un SMA. Nous fixons ensuite quelles sont les activités à coordonner et quelles sont les caractéristiques des modèles de coordination qui réalisent cette gestion. Nous terminerons ce chapitre en présentant plusieurs systèmes qui proposent des modèles de coordination de négociations.

3.1. Interactions dans les Systèmes Multi-Agents

Nous allons nous intéresser ici à une revue rapide de la problématique de la gestion des interactions dans un système multi-agent. Dans un SMA et aussi dans tous les systèmes informatiques l'interaction est importante dans la mesure où c'est à travers elle qu'on peut négocier, entrer en compétition, fournir un service ou combiner des services existants. De manière générale le concept d'*interaction* pourrait bien s'exprimer en fonction des concepts *d'action*, *d'agent* et *d'environnement* [Chaib-draa et al.'92]. Cette approche est soutenue par l'idée que dans une interaction il y a un agent qui réalise l'action et l'environnement qui subit les effets de l'action ou que parfois c'est l'environnement qui agit et l'agent qui subit les conséquences.

Mais, l'interaction peut aussi avoir lieu entre un agent et d'autres agents peuplant son environnement. Ainsi le concept d'interaction peut être défini en termes *d'actions* et *d'agents* et dans ce cas un aspect très important de l'interaction est la communication entre agents. De cette manière l'interaction est une dimension fondamentale dans un SMA puisqu'elle permet d'installer une coordination entre les agents, en terme d'actions de délégation et d'adoption ; d'une part - flux de contrôle entre les agents- et échanges d'informations pour aider à la coopération -flux d'observations entre les agents- d'autre part.

Afin que des agents autonomes puissent collaborer et coopérer entre eux, les modèles d'interaction développés en SMA visent à mettre en place une forme évoluée de communication capable de mettre en oeuvre des formes explicites de délégations et adoptions, capables d'échanger des informations (connaissances, croyances, buts, plans, etc.). Cet objectif se décline principalement selon deux thématiques qui sont en forte relation :

– "*ACL*" (Langage de Communication Agent) qui s'intéresse principalement aux problèmes de syntaxe, de sémantique des langages de communication dans un contexte de temps et de ressources de calcul limités.

– "*Conversation*" qui s'intéresse à la pragmatique de la communication, à la structuration des échanges et à la coordination de ce processus de communication via par exemple des protocoles d'interaction ou des Conversation policies.

Une conversation entre plusieurs agents peut être caractérisée par :

– le but pour lequel elle est menée (pour résoudre un conflit, pour établir un contrat, pour délibérer sur l'exécution d'une certaine action ou seulement pour argumenter [Greaves et al.'00]). Un but de conversation défini et géré par un agent, peut être local ou

partagé par plusieurs agents, secret ou non. Un but de conversation peut également être attaché au système (cf. place de marché). Dans ce cas, il est connu de tous les agents du système.

– par le nombre de participants et leurs identités (voire les dimensions des négociations). Ces informations peuvent être fixées à l'initialisation de la conversation mais elles peuvent également être fixées et déterminées en cours de conversation. L'identité devient particulièrement importante s'il existe des interactions répétées entre les mêmes participants dans le cas des systèmes fermés ou avec un nombre relativement petit de participants.

– par l'ensemble des règles qui la gouvernent (dans ce cas une conversation devient une instanciation d'un protocole d'interaction qui fixe des contraintes sur la synchronisation et la concurrence des messages ainsi que sur leur contenu [Zhang et Lesser'02]).

Ces différents paramètres peuvent, lors d'interactions répétées, évoluer et être appris (apprentissage de stratégie, apprentissage de préférences du partenaire).

La conversation est donc un mécanisme qui facilite l'interaction entre agents et aussi la compréhension des informations communiquées. En prenant le cas des grands systèmes d'agents [Sandholm et Lesser'95] [Sandholm et Lesser'97] [Jennings et al.'96], tous les agents impliqués exécutent certains calculs localement mais souvent, les agents ne peuvent pas entièrement ou exactement finaliser leur traitement local sans communiquer avec d'autres agents. Dans ces systèmes, produire une solution à un problème n'est souvent pas un simple assemblage des solutions locales des agents. Plutôt, ce processus peut être un processus multi-étapes dans lequel les agents dialoguent en communiquant des informations aux multiples niveaux d'abstraction. Chaque étape du dialogue peut exiger de l'agent de raisonner au sujet de la nouvelle information qu'il doit envoyer par rapport à sa connaissance existante. Dans ce cas, une bonne manière de regarder la décision au sujet de comment mieux communiquer peut être encadrée en tant qu'un problème complexe d'optimisation sur la façon dont les actions de communication contribuent à la manière dont le système ou les agents atteignent leurs objectifs. La solution efficace de ce problème est présentée d'une manière quantitative par les avantages et les coûts d'un processus de coordination des conversations [Moehlman et al.'92] [Gasser'92].

Le besoin de comportement satisfaisant pendant l'exécution en temps réel du système, combiné avec la nature complexe de l'interaction entre les agents du système justifie, dans certaines situations, la recherche des mécanismes efficaces de coordination.

3.2. Coordination d'interactions

Dans cette section, nous introduisons la notion de coordination en délimitant les types de coordination selon les mécanismes utilisés, ainsi que les propriétés des ces modèles de coordination des interactions implémentés dans des différents SMA.

3.2.1. Coordination générique, coordination stratégique

Dans l'informatique la recherche sur la coordination se fait dans plusieurs communautés comme celles sur la programmation distribuée et concurrente, l'intelligence artificielle distribuée ou le génie logiciel. Cela fait que les approches sur la coordination sont multiples et très différentes. Les caractéristiques des approches peuvent être groupées en fonctions de trois questions [Coo'94] : qu'est-ce qui est coordonné ?, quel est le médium de coordination ? quels sont les protocoles ou les règles utilisées pour la coordination ?

En répondant à la première question du point de vue du développeur en informatique, nous pouvons délimiter deux types de coordination très différents : le premier, une approche ***générique*** où les composants du processus de coordination sont les entités coordonnées, le deuxième, une approche ***stratégique*** où les composants du processus de coordination sont les entités qui coordonnent. Ces deux approches sont parfaitement illustrées par les deux définitions suivantes sur la coordination :

«*Coordination is the process of building programs by gluing together active pieces*». [Carrieri et Gelernter'92].

« *Coordination, the process by which an agent reasons about its local actions and the (anticipated) actions of others to try and ensure the community acts in a coherent manner, is perhaps the key problem of the discipline of Distributed Artificial Intelligence* ». [Jennings'93]

Coordination générique : Avec la première définition, en soulignant que la coordination est une façon de programmer en spécifiant seulement les interactions entre des programmes existants, on montre que les modèles de coordination appartiennent à une approche intergicielle ou langage de coordination. Des exemples typiques pour cette approche sont les intergiciels, comme Corba [Vinoski'97], qui visent précisément le développement

d'applications et la réutilisation du code dans des environnements répartis; ou les langages de coordination et de communication de bas niveau comme Linda [Gelernter'85] [Carrieri et Gelernter'89], JavaSpaces. Dans ce type de coordination les valeurs des données sont transparentes et la communication est anonyme, de plus, les entités coordonnables ne sont pas précisément spécifiées.

Coordination stratégique : La deuxième définition, propre à la communauté de l'intelligence artificielle distribuée, s'intéresse aux moyens d'effectuer la coordination en spécifiant la manière dont les entités se comportent et interagissent. Dans cette direction, plusieurs approches peuvent être considérées prometteuses. Les approches sur la coordination dans les Systèmes Multi-Agent sont classées dans deux catégories : une coordination objective et une coordination subjective [Schumacher'01] :

- La coordination objective maîtrise le système multi-agent sans agir directement sur le comportement des agents composants, mais plutôt en définissant comment les interactions entre les agents doivent entre modélisées. Principalement, la mise en œuvre est faite dans deux modes d'interaction: (i)interaction indirecte par l'environnement commun partagé par les agents, (ii)interaction directe par l'envoi de messages entre les agents.
- En revanche, au niveau de la coordination subjective l'accent est mis sur les moyens de modéliser les comportements des agents. Ces moyens sont classés par Ossowski [Ossowski'99] en trois familles : (i) *planification multi-agent* – les agents se coordonnent en suivant un plan pour arriver à un certain but, (ii)*négociation* - les agents construisent le plan et ce plan peut faire aussi l'objet d'une re-négociation [Nwana et al.'97] (iii)*organisation*- dans une organisation le rôle d'un agent détermine ses compétences et ses responsabilités, et l'agent agit en concordance avec elles.

En partant de ces deux grandes directions (i.e. *générique* et *stratégique*) et afin de présenter par la suite plusieurs travaux proposant des modèles pour la coordination des négociations, nous commençons par expliciter deux principales caractéristiques des modèles de coordination des interactions dans les SMA : *coordination locale/globale* et *coordination centralisée/distribuée*.

3.2.2. Coordination locale vs. Coordination globale

Dans les sections précédentes nous avons présenté qu'à un moment donné, au sein d'un agent, coexistent plusieurs conversations. L'agent doit donc coordonner l'évolution de ces conversations ainsi que les relations qui existent entre elles. Plusieurs modélisations de cette coordination existent soit en mettant en place une gestion purement locale de la coordination des interactions entre les agents (*modèles locaux* de coordination d'interaction), soit par la mise en place de *modèles globaux* tels que des règles générales pour toutes les conversations (conversation policies) [Bradshaw et al.'97], des protocoles de conversations ou de plans de conversations [Barbuceanu et Fox'98].

Coordination locale : Quelques travaux initiaux dans le domaine ont proposé des modèles de coordination d'interaction locale aux agents pour la gestion des conversations. Ils ont tenté d'équiper chacun des agents du système des mécanismes pour anticiper les séquences de messages pouvant satisfaire un objectif donné. Ces approches ont donné lieu à des propositions de mécanismes de planification de dialogue [Cohen et Perrault'79] ou [Bretier'95] à partir de performatifs de base.

Cependant, étant donnée la combinatoire des enchaînements possibles, l'ambiguïté de la sémantique du vocabulaire utilisé (notamment des performatifs), de telles approches se sont avérées difficiles à mettre en place. Ainsi, comme c'est le cas pour les ACL, la thématique des conversations a majoritairement conduit à l'élaboration de modèles globaux pour la spécification de conversations.

Coordination globale : Un modèle global d'interaction est un modèle d'interaction partagé entre les différents agents du SMA. Les modèles globaux existants se distinguent-en : Protocoles d'Interaction et en Politiques de Conversation (Conversation Policies).

Un protocole d'interaction désigne un *schéma d'interaction* entre agents, représenté explicitement et imposé aux agents pour contrôler et structurer leurs interactions. Les schémas d'interaction disponibles dans la littérature concernent des conversations d'une ampleur plus ou moins grande [Smith et Davis'81], [Sian'91], [Berthet et al.'92], [Barbuceanu et Fox'98], dédiés à un domaine d'application [Demazeau et al.'94] ou à des situations de dialogue [Chang et Woo'93], [Reed'98]. En particulier, en [Chang et Woo'93] les auteurs se sont basés sur le travail décrit dans [Ballmer et al.'81] et ils ont pris en considération toutes les situations de

conflit qui peuvent prendre place dans un système où le contrôle est distribué et où aucune réponse globale ne peut exister.

Dans son effort de standardisation, la FIPA a proposé un ensemble de protocoles d'interaction [FIPA'00] qui, bien qu'étant loin de répondre aux besoins particuliers et à la diversité des différentes applications, fournit un "référentiel" de dialogues communs entre les différentes applications multi-agent. La majorité de recherches, jusqu'à récemment, s'est attachée au développement des formalismes, pouvant répondre aux différentes propriétés que l'on cherchait à vérifier et valider [Koning'01]. Ceux-ci se fondent sur les réseaux de transition (automates, réseaux de pétri, sur des langages descriptifs tels que AUML (Agent Unified Modelling Language) [Bauer'00], etc.), sur des formalismes à base de logique, etc. Une des préoccupations actuelles est de définir des formalismes permettant de définir et composer les protocoles entre eux [Huget'01].

Il manque cependant encore la définition d'une sémantique précise et explicite pour ces schémas d'interaction. Il manque surtout de permettre aux agents d'interagir avec plus de flexibilité en réagissant au contexte de conversation qui se co-construit tout au long des échanges.

Récemment, le terme de Conversation Policy a été introduit pour chercher à répondre aux griefs faits aux protocoles d'interaction. Les Conversation Policies désignent des procédures de décision, avec une sémantique propre, qui permet aux agents de sélectionner et de produire des messages en prenant en compte : leurs intentions, le contexte défini par les précédents messages, le contexte défini par les interactions avec les autres agents [Greaves'00].

Les approches s'appuyant sur des modèles globaux sont multiples. On y trouve en effet des approches purement globales où chaque agent interprète et exécute un schéma global d'interaction. On y trouve également des approches combinant la prise en compte de schémas globaux tout en permettant à l'agent de développer une stratégie propre d'interaction. C'est ce qu'on retrouve dans les travaux de [Faratin'98] par exemple, où chaque agent tout en utilisant des protocoles globaux de négociation communs aux autres agents, peut développer localement des stratégies propres de négociation. Celles-ci lui permettent de générer de nouvelles propositions tenant compte de ses préférences, des propositions antérieures et de différentes tactiques débouchant sur des négociations cherchant à limiter le nombre de messages échangés, le temps de négociation, etc.

Alors que ces deux démarches sont mises en oeuvre au sein de l'agent, d'autres approches utilisent des modèles globaux qui sont mis en oeuvre dans un service générique

extérieur aux agents. C'est notamment le cas des travaux de [Andreoli'96c], [Rousseau'95], [Singh'98].

3.2.2. Coordination centralisée vs. Coordination distribuée

Une autre caractéristique du modèle de coordination peut être identifiée en répondant à la question « qui fait la coordination ? ». Dans un SMA la coordination entre les agents est faite soit par une entité centrale et dans ce cas nous avons une coordination *centralisée*, soit par les agents eux-mêmes et dans ce cas nous parlerons de coordination *distribuée*.

Coordination centralisée : Les approches centralisées sur la coordination proposent une entité centrale qui a le pouvoir sur les interactions dans le système. Cette approche vise principalement à coordonner les interactions, mais aussi à faciliter et à optimiser ces interactions. En fonction de l'entité chargée de la coordination deux directions peuvent être soulignées.

Dans la première, l'entité peut être un agent central qui selon ces fonctions est appelé un broker [Bayardo'97], un mediator [Tung et Raymund'05], un facilitator [Nwana'99], un middle agent [Decker et al.'97] ou match maker [Foner'96]. Ayant souvent les mêmes attributions, la coordination centralisée peut être aussi faite par une institution chargée de médiatiser les interactions entre plusieurs agents. Cette approche donne une garantie sur l'impartialité et l'équidistances vis à vis des autres agents [Sycara'91].

Cette manière centralisée de coordonner les interactions entre plusieurs agents peut être acceptable dans certains cas, mais à partir du moment où chacun des agents veut gérer par lui-même ses interactions, la coordination faite par une tierce partie n'est plus possible et le processus de coordination doit devenir distribuée.

Coordination distribuée : C'est dans le commerce électronique de type B2B (business to business) avec la mise en place d'un système de coalition [Tsvetovat et al.'00] ou d'alliances entre entités autonomes (PRODNET-II [prodnet]), où un mécanisme de coordination distribué des interactions est le plus nécessaire.

Certaines approches décrivent la coordination distribuée comme une exécution dispersée de plusieurs actions comme étapes dans le contexte d'une division du travail. Dans ce cas la coordination est souvent soutenue par l'organisation du système qui identifie les

différents rôles pour les agents et leurs rapports potentiels avec des actions ou des procédures visant d'arriver à un but selon un plan global [Ephrati'94]. Des autres approches décrivent la coordination distribuée en liaison avec la notion d'autonomie où l'agent a le pouvoir d'effectuer des interactions indépendamment du reste du système [Castefranchi'95] et de créer et de contrôler les exécutions des ses propres plans [Brussel et al.'93].

Récemment, d'autres travaux attachent à la coordination distribuée la notion de conscience (« awareness ») de l'agent. La conscience de l'interaction peut se traduire par le fait que l'agent a les connaissances sur la nature du plan effectué et de l'espace dans lequel il agit et en fonction de ça il décide d'interagir et de se coordonner avec les autres [Koch'98] [Omicini et al.'04].

Cette première partie sur état de l'art de la coordination des interactions nous permet ainsi de fixer deux points de vue différents sur la coordination - *générique/stratégique*. De manière orthogonale à ces deux points de vue, nous avons mis en évidence plusieurs caractéristiques déterminantes pour la coordination des négociations - *local/global* et *centralisé/distribué* -. Dans la section suivante, nous allons présenter différents systèmes proposant des modèles ou des mécanismes pour la coordination de négociations en commençant par les approches proposant une coordination stratégique, suivie par les approches proposant une coordination générique.

3.3. Coordination de négociations

Dans cette section nous présentons les principaux travaux qui mettent en place des systèmes de négociation où chaque participant peut être impliqué dans plusieurs conversations à la fois. Les différents systèmes présentés par la suite identifient et résolvent les problèmes du processus de coordination de ces différentes conversations.

Dans les descriptions des systèmes suivants, nous avons essayé d'identifier certains aspects de la coordination : *i)* comment est implémenté le médium de coordination (*stratégique/générique*); *ii)* comment est définit l'objet à coordonner (*locale/globale*); *iii)* comment est géré le processus de coordination (*centralisée/distribuée*) ; et *iv)* quelles sont les caractéristiques des négociations sur lesquelles sont définies les dépendances gérées par le processus de coordination.

La structuration des travaux a été faite en deux parties. Nous commençons la présentation avec les systèmes qui modélisent la coordination de manière stratégique afin de continuer avec les systèmes modélisant la coordination de manière générique. Egalement, dans les deux parties nous avons structuré les systèmes en fonction de la complexité des caractéristiques des négociations modélisées. Ainsi, en utilisant l'analyse de la complexité du processus de négociation en fonction du nombre des participants et de la structure de l'objet de négociation (voir section 2.4.) la présentation est faite en partant des systèmes qui coordonnent des négociations de type *un à un* faites sur un même objet vers des systèmes qui coordonnent des négociations multi-participant (i.e. *un à plusieurs*, *multi-bilatéral* ou *plusieurs à plusieurs*) faites sur différents objets de négociation multi-attribut.

3.3.1. Coordinations stratégiques de négociations

Une caractéristique commune pour les quatre systèmes présentés par la suite est le fait qu'ils proposent des SMA où la coordination entre les actions des agents est le résultat d'un processus de décision réalisé par les agents eux-mêmes. Selon la complexité du processus de négociation et des dépendances à gérer, les agents décident de leurs actions en prenant en compte certaines dimensions du processus de négociation, comme les aspects liés à : 1) la complexité de l'objet de négociation et du protocole de négociation, 2) les identités des autres participants, 3) l'historique des conversations avec eux, 4) les stratégies de négociation employées.

3.3.1.1. ADEPT

Dans le projet ADEPT (Advanced Decision Environment for Process Tasks) [Jennings et al.'90] [Alty et al.'94] [Jennings et al.'96], N. R. Jennings, J. A. Alty, T. J. Norman, P. Faratin et P.O'Brien[7] ont proposé une infrastructure multi-agent pour la mise en place et l'exécution distribuée d'un processus industriel composé par plusieurs services. Cette technologie a été déployée dans deux vrais applications : une avec British Telecom et une avec ICI Engineering.

Ce projet s'adresse principalement aux grandes entreprises qui sont divisées en plusieurs organisations autonomes, chacune gérant ses ressources et ses informations et aussi ses services spécifiques qu'elles exécutent dans le cadre du processus industriel global (voir Fig. 4).

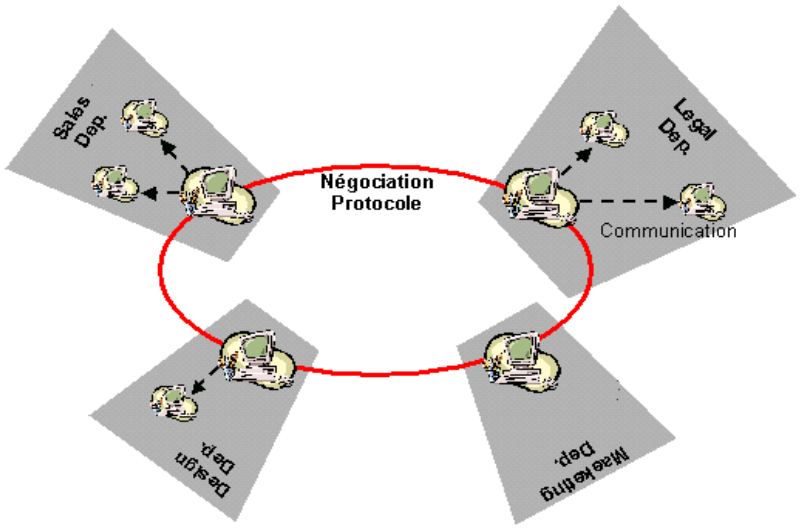

Figure 3. ADEPT : réseau d'organisations autonomes

Etant données ces caractéristiques, l'approche propose de modéliser le processus industriel de manière distribuée sur plusieurs agents autonomes gérant des parties distinctes du processus et qui négocient entre eux au moment où leurs services ont des interdépendances.

---

[7] Department of Electronic Engineering, Queen Mary and Westfield College, University of London, London E1 4NS, U.K.

Dans ce contexte, une organisation peut être regardée comme un agent représentant une entité capable d'exécuter un ou plusieurs *services*.

Un *service* est vu comme une unité d'un processus plus complexe. Un service peut être défini récursivement en termes des sous-services, jusqu'au service atomique le plus simple qui peut être géré par le système ADEPT, appelé *tâche*. Ces unités atomiques peuvent être composées pour former des sous-services et par la suite des *services complexe*s qui peuvent faire l'objet d'un échange entre deux agents. La composition des sous-services suppose l'existence des contraintes d'exécution (*les tâches* peuvent être exécutées en parallèle, doivent être exécutés en parallèle ou doivent être exécutés séquentiellement) et d'une entité de contrôle qui supervise ces contraintes. Dans ces cas, un service complexe est une composition des plusieurs *tâches* ou d'autre sous-services. Cette définition récursive d'un service soutient l'approche qu'un processus industriel complexe peut être vu comme un service exécuté de manière distribuée.

Les services sont associés à un ou plusieurs agents, qui gèrent la mise en place et l'exécution du service. Mais pour chaque service un seul agent est vu comme responsable, même si d'autres agents gèrent les sous-services qui les composent. Les ressources d'un agent peuvent être des services atomiques (*tâches*) ou d'autres agents. Le dernier cas permet la mise en place d'un système hiérarchique d'agents dans lequel les agents de plus haut niveau réalisent leur fonctionnalité en se basant sur les agents plus bas. Les agents plus bas ont la même structure que ceux des niveaux élevés et peuvent donc, avoir aussi des sous-agents comme ressources dans leur agence.

Donc, un agent peut être engagé dans deux types de conversations. La première est une négociation externe avec des agents du même niveau hiérarchique sur des services qu'il veut utiliser et dans ce cas il a le rôle *d'acheteur* ou sur des services qui lui sont demandés et dans ce cas il a le rôle de *vendeur*. La deuxième est une communication interne avec les agents qui sont au-dessus dans la hiérarchie organisationnelle et qui est une simple requête d'exécution.

Le problème complexe de la coordination est au niveau d'un agent responsable d'un service qui peut être impliqué dans les deux types de conversation. Celui-ci peut être engagé dans une négociation avec un autre agent du même niveau sur la demande d'un de ses services et dans le même temps il doit négocier avec ses sous-agents pour voir s'il est capable d'exécuter le service au moment de la demande. Dans ce cas, deux sous-problèmes existent : *i)* comment choisir entre plusieurs sous-agents qui peuvent exécuter le même service et *ii)* comment coordonner la négociation externe avec les négociations internes.

- Le premier problème est résolu au niveau décisionnel avec une stratégie qui indique à l'agent responsable avec quel sous-agent il doit négocier en premier. Afin de négocier dans l'environnement ADEPT, les agents disposent des modèles de raisonnement complexes. Le modèle de raisonnement dispose des représentations détaillées des services que l'agent peut proposer ainsi que des descriptions des autres services proposés par les agents présents dans système.
- Le deuxième problème est résolu aussi au niveau décisionnel. C'est le modèle de raisonnement qui définit également si les négociations se font séquentiellement ou en parallèle afin d'éviter les deadlock. Comme la décomposition d'un service peut être faite sur plusieurs niveaux, la finalisation d'une négociation sur un service complexe doit attendre la finalisation de plusieurs négociations aux niveaux inférieurs. Afin de limiter le temps d'attente et pour conduire en parallèle les négociations plusieurs suppositions sont faite : *i)* l'agent responsable est supposé à faire toujours ses services, de cette manière un agent peut finaliser les négociations avec les agents du même niveau sans attendre les finalisations des négociations avec ses sous-agents ; *ii)* le temps n'est pas pris en compte, de cette manière si une négociation échoue les agents sont pénalisés mais une renégociation est toujours possible et donc toute la chaîne des négociations peut recommencer.

Bien sûr, d'autres problèmes de coordination plus complexes existent, par exemple si l'agent négocie dans le même temps le service avec plusieurs agents ou si les attributs des services ont des dépendances entre eux**.** Ces types des problèmes ne sont pas abordés dans ce projet.

Selon notre grille sur la négociation décrite dans le chapitre 2, le projet ADEPT est dédié à la *négociation bilatérale* sur un objet de *négociation multi-attribut*. La coordination est *distribuée* sur plusieurs agents responsables qui se synchronisent et coopèrent entre eux afin d'offrir un service *global*. La coordination est gérée au niveau *stratégique*, les agents ont des stratégies bien définies pour interagir et se coordonner.

Le projet ADEPT a été un des premiers qui attaque le problème de la coordination entre des négociations bilatérales avec des dépendances entre elles et qui se porte sur des objets multi-attribut en proposant une solution distribuée de la coordination sur plusieurs agents. Par contre, bien que les négociations soient sur des objets de négociation multi-attribut les dépendances entre les négociations ne prennent en compte aucune information sur les valeurs des objets : les dépendances entre négociations sont réduites aux informations sur les

finalisations des négociations - une négociation se finalise avec un contrat seulement si les autres négociations sur les sous-service se finalisent avec des contrats.

Egalement, l'architecture hiérarchique proposée contraint l'autonomie d'un agent et fait que de notre point de vue sur la négociation les interactions entre agents peuvent être vues de la manière suivante : i) si la demande d'un service vient d'un agent de même niveau, les deux agents commencent une négociation et ii) si la demande d'un service vient d'un agent d'un niveau élevé, l'agent doit exécuter le service s'il peut et donc ce type d'interaction est plus une commande qu'une négociation.

Finalement, pour les négociations entre les agents du même niveau le système propose la possibilité de choisir entre plusieurs participants afin de commencer une négociation, mais toujours un agent est impliqué dans une seule négociation bilatérale à la fois. Donc, la résolution des problèmes liés aux dépendances entre les négociations est assez primitive sans permettre d'identifier plus clairement les problèmes dans le cas où un agent serait impliqué simultanément dans plusieurs négociations bilatérales.

3.3.1.2. MACIV, SMACE et ForEV

La complexité du processus de négociation et les nombreux problèmes liés à la coordination des plusieurs négociations sont soulignés et certaines résolues dans les projets MACIV, SMACE et ForEV développés par E. Oliveira, A.P. Rocha, H.L. Cordoso et J.M. Fonseca dans le laboratoire de l'Université de Porto [Oliveira et al.'97] [Oliveira et Rocha'00] [Cordoso et Oliveira'00] [Oliveira'01].

Plusieurs caractéristiques sur la négociation sont communes dans ces projets : les agents sont autonomes, chacun dispose de mécanismes de décision et de stratégies de négociation locales, les négociations sont faites sur des objets de négociation ou des tâches décrites par *plusieurs attributs*, chaque négociation consiste en *plusieurs échanges des propositions* entre les participants. Les participants peuvent avoir différents rôles en fonction de leur implication dans une négociation : *vendeur*, *acheteur* ou *médiateur*. Les propositions échangées pendant le processus de négociation sont utilisées par les agents dans un processus d'apprentissage afin d'augmenter la qualité de leurs propositions. Finalement, chaque agent peut être impliqué dans *plusieurs négociations* dans le même temps.

Par la suite, nous allons présenter quels sont les différents mécanismes de coordination proposés dans les trois projets en commençant, du point de vue de la coordination, avec le plus simple vers le plus complexe.

Dans le projet ForEV la coordination est implémentée avec un protocole de négociation issue du Contract Net Protocol (CNP) [Smith'80]. Mais contrairement au CNP, où l'agent coordinateur se contente de choisir la meilleure proposition reçue, le protocole proposé en ForEV est itéré sur plusieurs étapes. L'objectif principal de ces itérations est de permettre à chaque participant de connaître les défauts de sa proposition. En effet, les informations fournies à chaque étape par un agent coordinateur (« Market Agent ») guident chacun des participants à connaître les défauts de sa proposition et les choix possibles afin d'augmenter le plus possible ses chances de gagner dans la prochaine étape avec une nouvelle proposition. Cette approche est très performante dans la mesure où elle améliore efficacement les propositions à chaque étape, mais elle est très limitative au niveau de l'autonomie des agents et des interactions possibles. Un exemple est le fait qu'à chaque étape l'agent gagnant est obligé de ne pas faire des offres pendant l'étape suivante.

La négociation dans SMACE est basée sur une interaction caractérisée par un échange de propositions et de contre-propositions entre les agents acheteurs et vendeurs. Une

caractéristique principale de la négociation dans SMACE est le fait que chaque agent peut conduire une négociation multilatérale comme un ensemble de négociations bilatérales. Par contre, de la même manière que dans le projet ADEPT, la coordination entre les négociations bilatérales est limitée seulement aux contraintes soulevées par la finalisation des négociations avec la mise en place d'un accord. A la différence de ADEPT, ici les agents sont en compétition et donc la finalisation d'une négociation avec un contrat empêche les autres négociations sur le même objet de continuer. Pour régler ce problème deux moyens de coordination sont proposés :

1) un protocole de coordination qui spécifie qu'à l'envoi d'une acceptation l'agent receveur doit confirmer qu'il peut toujours honorer son engagement. De cette manière sont résolues les problèmes liés à la synchronisation et à la concurrence entre plusieurs processus (négociations bilatérales) sur la même ressource (objet de négociation). L'agent coordonnateur qui reçoit les acceptations décide d'accepter seulement une des propositions.
2) une police de coordination générale spécifiant qu'un agent qui envoie une acceptation à un autre agent doit suspendre toutes les négociations avec les autres et rejeter toutes les acceptations qu'il reçoit de la part de ces derniers. De cette manière sont résolues les mêmes problèmes de synchronisation qu'auparavant mais cette fois pour les agents qui envoient les acceptations.

Une approche plus complexe sur la coordination des négociations est faite dans le projet MACIV. Dans MACIV les agents sont considérés autonomes et ils essaient de satisfaire leurs buts sans tenir compte d'un possible gain global du système et sans avoir aucune information globale sur le système. Mais à la différence des autres projets, dans MACIV les agents ont la possibilité de former de coalitions virtuelles afin de gérer de manière commune une même tâche.

Partant de cette caractéristique, la coordination des négociations est envisagée sur deux niveaux. Le premier niveau est géré par un agent coordinateur qui en plus des fonctionnalités présentées auparavant, il a deux autres attributions : *i)* de former les possibles coalitions en fonction des premières propositions envoyées par les agents participants; et *ii)* de nommer un agent intra-coordonnateur dans chaque coalition. Le deuxième niveau est géré par l'agent nommé comme intra-coordonnateur dans une coalition.

Par la suite, le processus de négociation est divisé sur ces deux niveaux par deux protocoles de négociation.

Dans le premier, l'agent coordonnateur reçoit les propositions envoyées par les agents désignés comme coordonnateurs d'une coalition et les répond avec la meilleure proposition reçue jusqu'au moment considéré. Par la suite, les agents intra-coordinateurs essaient de négocier dans le cadre de leur coalition une meilleure proposition en utilisant un nouveau protocole. Donc, nous pouvons remarquer que l'agent intra-coordinateur gère deux types de conversations pilotées avec des protocoles de négociation différents. Cette coordination sur deux niveaux avec deux protocoles de négociation différents nous l'avons trouvée aussi dans ADEPT, la différence ici est que les agents dans les sous-groupes ou coalitions sont au même niveau d'autonomie et donc les processus de négociation sont plus compliqués.

Afin des soutenir cette approche sur la coordination, une politique globale est définie sur la manière dont les deux protocoles de négociations sont imbriqués.

Une négociation est considérée comme une succession des différentes étapes :

- dans la première étape l'agent coordinateur collecte les différentes propositions et il décide comment les agents peuvent s'organiser dans des coalitions. Par la suite, l'agent coordinateur a aussi le pouvoir de choisir quels seront les agents intra-coordinateurs dans chacune des alliances.
- dans la deuxième étape, le processus de négociation commence avec l'agent coordonnateur qui envoie la meilleure proposition qu'il a au moment respectif. Par la suite, la négociation continue avec plusieurs itérations d'échange des propositions et de contre-propositions.
- dans la troisième étape, afin d'essayer de construire une contra-proposition, chaque agent intra-coordinateur commence dans son alliance un processus de négociation. Si l'agent intra-coordinateur réussit de construire une proposition meilleure que celle envoyée par l'agent coordinateur, dans ce cas il envoie la contra-proposition, si non il se retire de la négociation.
- la dernière étape consiste dans une agrégation, au niveau de l'agent coordinateur, des toutes les propositions reçues de la part des agents intra-coordinateurs. L'agent coordinateur évalue les propositions et communique la meilleure réponse à tous les participants présents dans la négociation.

Ces deux dernières étapes sont répétées jusqu'à ce que une seule alliance reste ou que le temps disponible pour négocier soit écoulé.

Selon notre grille sur la coordination, dans les trois projets, la coordination d'une négociation est faite par une approche *stratégique* et *globale*, principalement par des protocoles de négociations complétés dans certains cas avec des différentes polices de coordination.

Egalement, la coordination est gérée de manière *centralisée* par un agent coordinateur (dans le projets ForEV et SMACE) et de manière *distribuée* par plusieurs agents coordinateurs (dans le projet MACIV). L'agent coordinateur a trois fonctionnalités : i) la première est de collecter les propositions envoyées par les agents participants ; ii) la deuxième est de répondre à chacune d'entre elles en lui spécifiant quelle est la manière de faire évoluer sa proposition et finalement iii) de décider qui est le gagnant en fonction des propositions reçues et du temps limite fixé pour la finalisation de la négociation.

Donc, dans MACIV le processus de coordination gère plusieurs négociations et il est distribué sur plusieurs agents. En revanche, le système envisage la coordination seulement au niveau des négociations faites sur une même tâche de négociation et donc des ensembles limités des dépendances sont envisagés et gérés. Egalement, un problème intéressant soulevé par cette approche est le fait que pour la même négociation et dans le même moment de temps, un agent peut être présent dans plusieurs coalitions. Dans MACIV ce problème a été considéré et résolu au niveau du système qui doit empêcher que ça se produit. Par contre, en se positionnant du coté de l'agent omniprésent d'autres questions différentes sur la coordination peuvent être envisagées, comme : quelles sont les dépendances entre les propositions que l'agent omniprésent peut faire dans les différentes coalitions et comment il peut essayer de coordonner ces propositions ?

3.3.1.3. Multi-linked negotiation in multi-agent systems

Les travaux sur la négociation du Zhang et Lesser [Zhang et al.'01a] [Zhang et al.'01b] [Zhang et Lesser'02] sont également sur des négociations multilatérales, mais par rapport au projet MACIV, ils proposent des mécanismes afin de coordonner plusieurs négociations faites sur des *tâches différentes*. Ils décrivent une situation où un agent peut se trouver engagé dans plusieurs négociations avec plusieurs agents au sujet de tâches différentes et où une négociation sur une tâche influence les négociations sur d'autres tâches.

Dans ce type d'environnement où un agent négocie dans le même temps plusieurs tâches (tâches contractées par l'agent ou tâches proposées par l'agent), l'agent doit gérer les liens entre ces différentes tâches et en conséquence entre les négociations associées. Les liens entre les tâches qu'ils essaient à coordonner sont de deux types : i) *lien-direct (directly-linked*) représentant une relation de type producteur - consommateur entre deux tâches, une tâche B affecte directement une tâche A parce que la tâche B est une ressource nécessaire (ou une sous-tâche) pour la tâche A ; et ii) *lien-indirect (indirectly-linked)* représentant une relation de ressource partagée entre deux tâches, « une tâche B est indirectement liée à la tâche A parce que les deux sont en compétition pour une même ressource commune ».

Afin de coordonner ces types de liens ils ont proposée une approche *stratégique* et *distribué* basée sur une technique de planification d'ordre partiel où chaque agent gère de manière *locale* les dépendances entre les négociations dans lesquelles il est impliqué.

La planification d'ordre partiel est l'outil principal de raisonnement utilisé par l'agent pour coordonner les différentes négociations et il dispose aussi d'un moyen de mesurer son raisonnement, nommée *flexibilité*. Par la suite, nous allons présenter rapidement cette approche afin de montrer à la fin les limitations en fonction de notre vue sur la négociation.

Une planification d'ordre partiel structure dans un graphe acyclique un groupe de tâches avec des relations de précédences entre elles: G = {V, E} avec l'ensemble des nœuds V représentant l'ensemble des tâches et l'ensemble des arcs E={<u,v> | u,v∈V} représentant les relations des précédences entre deux tâches (la tâche *u* doit se finir afin que la tâche *v* puisse commencer).

Figure 5. Graphe de planification

Dans la Fig. 5 il y a la représentation de la planification faite par un agent sur ses tâches où chaque nœud est une tâche. Ces tâches peuvent être accomplies localement ou par d'autres agents et pour lesquelles l'agent considéré doit négocier. Faisant sa planification, un agent fixe non seulement l'ordonnancement des tâches mais il fixe également des contraintes sur ces tâches. Les contraintes sont des valeurs pour les temps d'exécution des tâches (*pt* : processe-time) et pour les intervalles de temps dans lesquels l'exécution de la tâche doit commencer (*est* : earliest-start-time et *lst* : latest-start-time) ou doit se finaliser (*eft* : earliest-finish-time et *dl* : deadline).

L'efficacité de la planification d'une tâche est donnée par la liberté de mouvement de la tâche dans l'ordonnancement de l'agent représentée par la flexibilité de la tâche :

$$F(t) = \frac{t.dl - t.est - t.process\_time}{t.process\_time}$$

L'efficacité de l'ordonnancement d'un agent est la somme des flexibilités de toutes ses tâches rapportées au temps d'exécution d'une tâche par rapport au temps d'exécution global de l'ordonnancement.

En fonction de la valeur de flexibilité d'un ordonnancement, l'agent peut raisonner sur l'influence qu'un engagement peut avoir sur les autres négociations en cours et quels sont les intervalles des valeurs dans lesquels l'engagement peut être fixé. Une fois que l'agent construit de manière locale et autonome l'ordonnancement de ses tâches il a deux possibilités de fixer les intervalles de temps. Il peut fixer des valeurs qui excluent les possibles conflits et de cette manière les négociations peuvent se dérouler concurremment. La deuxième alternative est d'ordonner les négociations en fonction de leur importance.

Dans les deux cas, une limitation est que l'agent est supposé connaître dès le début quelles sont les tâches et les négociations dans lesquelles il est impliqué. Cette approche suppose également que le contexte dans lequel l'agent évolue est complètement déterministe où rien ne change et rien de nouveau n'intervient pendant le déroulement d'une négociation.

Ainsi l'aspect dynamique inféré par des nouvelles négociations possibles ou par les incertitudes liées à l'exécution et la finalisation des tâches n'est pas pris en compte.

Du fait que l'agent a à priori toutes les informations sur ses négociations, il est capable de fixer et de gérer des dépendances entre les états initiaux et finaux des négociations. Egalement, en prenant en compte les informations sur les tâches négociées et les dépendances entre ces tâches, l'agent peut faire la distinction entre les négociations qui sont en compétition (lien-indirect) ou en coopération (lien-direct).

La coordination dans ce système est complètement *distribuée* et chaque agent gère *localement* les dépendances entre ses négociations. En revanche, pour fixer certaines contraintes sur les évolutions de ses négociations, notamment les meta-informations sur les difficultés ou la durée d'une négociation**,** l'agent est supposé avoir une coopération totale de la part des autres agents qui lui fournissent ces informations. Donc la coordination locale est soutenue par le fait que chaque agent a une connaissance complète sur la globalité du système. Cette approche n'est donc pas du tout applicable dans les systèmes où il y a une forte compétition entre les agents et qui ne permettent pas à chaque agent d'avoir une connaissance complète et générale de toutes les négociations en déroulement.

3.3.1.4. Système de planification de réunions

Continuant nos présentations progressives des systèmes de coordination des négociations, les travaux de Sen et Durfee sur la planification des réunions reprend les caractéristiques du système présenté auparavant. Dans leurs travaux sur la planification des réunions [Sen et Durfee'91] [Sen et Durfee'93] [Sen et Durfee'94], Sen et Durfee proposent aussi la négociation comme le seul moyen distribué de coordination tout en conservant l'autonomie des agents participants. Egalement, chaque agent peut être simultanément impliqué dans les planifications des plusieurs négociations et il essaie de les coordonner de manière *locale*. Mais contrairement aux travaux antérieurs, cette nouvelle approche suppose que l'information et le contrôle sont complètement décentralisés et que les agents échangent assez d'informations au cours des négociations afin de pouvoir les coordonner efficacement.

Dans ce type de scénario, la planification des réunions est un exemple typique de ce problème d'allocation des ressources, avec le temps comme ressource principale. L'approche *distribuée* fait que les agents ont seulement des connaissances sur leurs intervalles de temps et de cette manière les agents sont supposés être complètement coopératifs afin d'arriver à planifier des réunions.

L'approche sur la coordination est de type *stratégique* où les différentes contraintes sont gérées par un protocole de négociation est complétée avec des politiques de coordination qui sont appelées stratégies de coordination.

Pour permettre l'échange de l'information, les agents ont à leur disposition un protocole de communication commun pour négocier les temps de déroulement des réunions. Le protocole proposé est un protocole en plusieurs étages de négociation [Conry et al.'92] [Meyer'88] et qui est une généralisation du CNP [Smith'80].

Pour chaque réunion il y a un seul agent responsable qui initie la négociation (*host agent*). Le protocole est composé de quatre pas :

- dans la première étape, l'agent responsable ou *initiateur* commence la négociation en envoyant une ou plusieurs propositions avec les intervalles de temps qu 'il préfère.
- dans la deuxième étape, après avoir reçu les propositions, les agents invités vont essayer de trouver les intervalles libres dans leur agenda. Leurs réponses peuvent être des sous intervalles de ceux reçus ou des intervalles complètement différents.

- dans la troisième étape, les contra propositions sont reçues et évaluées par l'agent responsable. Si les contra propositions sont avec des intervalles de temps qui sont libres dans son agenda, l'agent responsable annonce un accord. Si non, il envoie des autres intervalles de temps.
- dans la quatrième étape, si les agents invités reçoivent des nouvelles propositions, la négociation continue avec l'itération des pas antérieurs. Si les agents invités reçoivent des annonces d'un accord, si l'intervalle est encore libre la réunion est planifiée. Si non ils envoient des refus et la négociation est reprise dès le début.

Ce processus pourrait être répété plusieurs fois pendant qu'ils recherchent par différentes parties de leurs calendriers respectifs. En général, un agent peut simultanément être impliqué dans la planification des plusieurs réunions, agissant en tant qu'*initiateur* pour certains et *invité* pour d'autres.

Les problèmes liés aux conflits entre deux ou plusieurs négociations qui essaient de programmer des réunions sur des intervalles de temps superposés sont gérés par différentes stratégies de coordination. Plusieurs stratégies sont présentées mais du point de vue de la coordination seulement deux stratégies sont à retenir : *commitment strategy* et *non-commitment strategy*.

Dans *commitment strategy*, les agents bloquent les intervalles de temps qui sont proposés dans une négociation jusqu'une réponse est reçue. Ce type de stratégie exclue la possibilité d'avoir plusieurs réunions superposées, mais le temps de computation est très grand et aussi le résultat risque de ne pas être optimale. Avec une stratégie de type *non-commitment* le temps de computation est plus court, mais l'agent se trouve souvent dans la situation d'annuler une proposition que lui a été faite parce que l'intervalle a été pris par une autre négociation. Une annulation peut avoir aussi des effets en cascade sur d'autres négociations et donc le temps de computation dévient à nouveau grand.

Cette approche *stratégique* résout les problèmes de la coordination des négociations seulement dans un cadre très général et simple. Pour un agent toutes les réunions sont vues sur le même niveau et donc la première négociation qui trouve un accord prend l'intervalle de temps. Normalement, les préférences au niveau de l'importance ou la durée de l'intervalle de temps de la réunion doivent être prises en compte. Mais avec la prise en compte de ces préférences le temps de computation croît de manière significative. Afin d'améliorer la qualité et l'efficacité de la planification des réunions, des nouvelles stratégies de négociation sont proposées, comme l'annulation et la re-planification des réunions pour favoriser les réunions considérées importantes.

3.3.2. Coordinations génériques de négociations

Jusqu'à maintenant, avec les quatre types des systèmes de coordination des négociations, nous avons présenté seulement l'approche stratégique où le processus de coordination est assimilé dans la description globale du processus de négociation. Avec les deux systèmes suivants nous serons capables de présenter une délimitation plus visible entre le processus de décision, le processus de communication et le processus de coordination composant le processus de négociation.

3.3.2.1. Framework générique de négociation automatique

Aux laboratoires HP ils ont constaté que pour automatiser le mécanisme de négociation entre plusieurs agents le protocole de négociation ne suffit pas et que plusieurs contraintes liées aux mécanismes de marché et aux négociations associées sont hard-codées dans des agents. Le protocole de négociation indiquant l'écoulement des messages peut fixer des contraintes de coordination sur les possibles actions que chaque partie peut faire et sur la finalisation de la négociation. Mais, les protocoles ne spécifient pas quelles sont les contraintes liées aux contenus des messages ou au temps de négociation.

Donc, en partant de la constatation que le protocole de négociation est insuffisant pour coordonner les interactions dans une négociation, ils ont proposé un framework *générique* dédié à la négociation automatique dans les places de marché *centralisée* [Bartolini et Priest'01] [Bartolini et al.'02]. Ce framework est générique dans la mesure où il ne prend pas en compte la structure de l'objet à négocier, les caractéristiques des participants impliqués ou leurs stratégies de négociation. Le système coordonne les messages faisant abstraction du contenu ou des caractéristiques spécifiques à l'émetteur ou au receveur.

Une autre caractéristique de ce framework de négociation est qu'il propose une séparation entre le *processus de décision* mis en œuvre par des stratégies de négociation décidées de manière locale et autonome par chaque participant et le *processus de coordination* mis en oeuvre par des règles publiques partagées par tous les participants à une négociation. Cette séparation permet une description flexible du protocole et des règles de coordination utilisées pour la mise en œuvre d'une négociation.

L'architecture du framework est structurée en couches (voir Fig. 6). La couche inférieure est une plate-forme orientée agent, proposant des services de base comme la communication entre agents ou la gestion de cycles de vie de la négociation. Au-dessus de la plate-forme orientée agent ils ont définit : (i) un protocole général de négociation, (ii) une taxonomie des règles de négociation, (iii) un langage pour définir les règles de négociation et (iv) un langage pour exprimer des propositions de négociation.

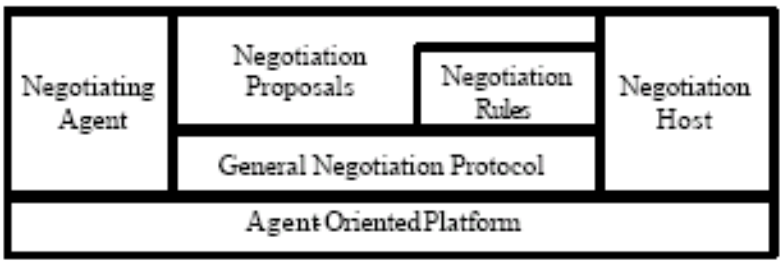

Figure 6. Architecture du framework de négociation

Le protocole général de négociation est composé par des règles de coordination indiquant le moment où un agent peut envoyer un message et quel message il peut envoyer en tant que réponse valide à un message indiqué.

Dans ce type de processus de négociation les agents peuvent avoir deux rôles, qui selon notre grille sont : agent participant (*negotiating agent*) et agent médiateur (*negotiation host*). Afin de négocier, les participants envoient leurs propositions à un espace commun (*the negotiation locale*), qui est contrôlé par l'agent médiateur. Cet espace de négociation est un type de tableau noir auquel chaque agent – participant ou médiateur – a accès pour écrire. Par contre, seul l'agent médiateur a une visibilité totale sur l'information. Il a le rôle de créer et de faire respecter les règles de coordination qui peuvent porter des contraintes sur l'exécution du protocole de négociation, sur la validation des propositions, sur la finalisation de la négociation et aussi sur la communication entre les participants. Donc le processus de négociation est modélisé comme une place de marché centralisée où l'agent médiateur a le rôle de coordonner toutes les négociations.

En fonction des types de règles, le rôle de l'agent médiateur est décomposé dans plusieurs sous-rôles attachés à une taxonomie des règles déclaratives qui peuvent être employées pour capturer une grande variété de mécanismes de négociation. Les sous-rôles et les types des règles attachés sont les suivants:

- *Gatekeeper :* gère les règles pour l'admission des participants dans la négociation ;

- *Proposal validator :* gère les règles de validité, imposant que n'importe quelle proposition soumise soit conforme avec un format de négociation fixant les contraintes sur les attributs négociés et leurs valeurs ;
- *Protocol enforcer :* gère plusieurs types de règles portant sur l'exécution du protocole de négociation : les règles de signalisation – fixant quand un participant peut envoyer une proposition, les règle d'amélioration – fixant quelles nouvelles propositions peuvent être envoyées, et les règles de retrait – fixant quand les propositions peuvent être retirées et quelle est la politique sur la période d'expiration des propositions ;
- *Agreement maker :* gère plusieurs types de règles, notamment pour mettre à jour les statuts des participants et l'information que chacun des participants peut accéder : les règles de mise à jour – indiquant comment les paramètres de la négociation changent sur l'occurrence de certains événements, les règles de visibilité – indiquant quels participants peuvent accéder une certaine proposition, les règles d'affichage – indiquant si une certaine information (nouvelle proposition ou nouvel accord) est visible pour un participant et aussi la manière dont cette information sera partagée aux participants ;
- *Information updater :* gère les règles d'acceptation d'un accord – cet ensemble de règles détermine quels sont les accords qui devraient être formés, étant donné un ensemble de propositions parmi lesquelles au moins deux sont compatibles;
- *Negotiation terminator :* fixe les règles pour le cycle de vie du processus de négociation, indiquant quand la négociation doit s'arrêter.

.

L'agent médiateur avec son rôle et ses sous-rôles est mis en application comme un système d'agents coopératifs qui communiquent à l'aide d'un tableau noir. Du point de vue des participants ces agents sont vus comme un seul agent médiateur qui coordonne la négociation. Cette centralisation du processus de coordination fait qu'il n'y a pas une distribution du contrôle sur les interactions entre les agents participants et l'agent médiateur. Chaque agent participant à une négociation envoie ses propositions à l'agent médiateur et c'est celui-ci qui, utilisant les ensembles des règles de coordination comme des filtres, décide si la proposition est valide et qui peut accéder à cette proposition. Ainsi, différemment des systèmes présentés auparavant, dans ce système le processus de coordination ne gère pas seulement des dépendances entre les états initiaux et finaux des négociations, mais l'agent médiateur est capable de vérifier des contraintes à chaque envoi d'une proposition. Par contre ces contraintes ont seulement le but de synchroniser la communication dans les différentes négociations. Bien que toutes les propositions envoyées soient disponibles au niveau de l'agent médiateur, celui-

ci ne peut pas fixer de dépendances sur les valeurs de ces propositions sans avoir aucune information sur les buts et les stratégies des différents agents participants.

Finalement, le système permet de modéliser et de coordonner des négociations de type *n vers n* participants comme un ensemble de plusieurs négociations de type *un à un*, mais toujours avec l'agent médiateur comme un des participants. Cette approche peut avoir des qualités sur l'optimisation de la coordination des interactions, mais elle limite l'autonomie et le pouvoir des agents participants au travers des différentes règles imposées et gérées par l'agent médiateur.

3.3.2.2. Interaction Oriented Programming

Pour mettre en place un système multi-agent, avec des agents autonomes qui peuvent interagir entre eux, le concepteur doit manipuler non seulement des aspects spécifiques à l'application des divers agents, mais également des aspects sur les comportements des agents et sur les interactions possibles entre eux. Les approches présentées jusqu'ici réalisent ce type de coordination en se basant sur les protocoles de communication ou sur les caractéristiques organisationnelles du système. Ces outils sont rigides et parfois peuvent limiter l'autonomie de l'agent. Singh a saisi ce problème et, afin que les agents gardent leur autonomie, il a proposé que la coordination d'un système multi-agent soit séparée dans un service distinct [Singh'97a] [Singh'97b] [Singh'98].

L'approche sur la coordination du Singh cherche à préserver l'autonomie de l'agent sur deux niveaux. Premièrement, au niveau de la programmation, où les agents peuvent être implémentés de façons différentes et leurs détails internes peuvent être indisponibles. Et deuxièmement, au niveau du comportement de l'agent, où les agents agissent de façon autonome et peuvent effectuer certaines actions qui sont dans leur portée.

La coordination porte sur les actions que l'agent fait à l'extérieur et qui sont aperçues comme potentiellement significatives pour le service de coordination. Ces actions sont appelées événements et peuvent être de quatre types :

- *flexible* : un événement indiquant que l'agent est disposé à retarder ou à écarter l'action;
- *inevitable* : un événement indiquant que l'agent est disposé seulement à retarder l'action;
- *immediate* : un événement indiquant que l'agent ne veut ni retarder ni écarter l'action;
- *triggerable* : un événement indiquant que l'agent peut exécuter l'action mais seulement sur demande du système.

Les événements d'un agent sont ensuite organisés dans un skeleton pour fournir un modèle simple de la coordination au niveau d'un agent. Ce modèle est typiquement un automate à états finis. En suite, chaque événement a attaché une *garde* qui exprime les conditions dans le système pour que l'événement soit exécuté. Les conditions, exprimées dans une logique temporelle, fixent des contraintes sur l'exécution éventuelle de l'événement et de quels autres événements il dépend pour pouvoir s'exécuter.

La Fig. 7 montre comment le service de coordination interagit avec les agents. Les agents l'informent des événements immédiats (*immédiate*) et demandent la permission pour les événements inévitables (*inevitable*) et *flexibles*. Le service accorde ou nie les permissions sur les événements flexibles, retarde les événements inévitables ou déclenche les événements *triggerable* en fonction des gardes associées.

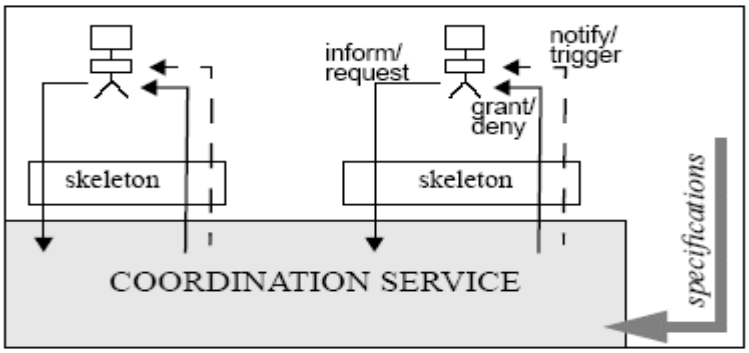

Figure 7. I.O.P. : vue sur l'interaction entre service et agents

Afin d'arriver à une coordination efficace entre les événements exécutés par plusieurs agents l'approche proposée par Singh exige les caractéristiques suivantes : *a)* les dépendances entre les actions (événements) sont connues au début de l'implémentation du système et *b)* les agents communiquent entre eux les informations relevantes sur l'exécution de leurs actions. Cela permet de calculer les gardes sur les actions qui permettront des prises de décisions sur leur occurrence et aussi de coordonner l'exécution de ce workflow distribué.

Donc, l'approche propose un service *générique* de coordination qui est implémenté de manière *distribuée,* entre les agents composant le système, l'exécution d'un seul workflow décrivant les actions nécessaires afin d'accomplir une seule tâche. Chaque agent peut être vu comme une entité qui englobe une partie du service de coordination et qui prend les décisions sur les exécutions des événements basées sur des informations *locales*. Finalement, nous avons choisi de présenter en dernier ce système parce que Singh montre aussi une séparation claire entre le processus de décision modélisé au niveau agent et le processus de coordination modélisé par un service extérieur. Cette caractéristique est similaire à un des nos objectifs, par contre, ce type de service est seulement intéressé à synchroniser les interactions entre les agents sans prendre en compte aucune information sur le contenu des messages, sur le but de la conversation ou sur les agents. Ainsi, supposant le cas particulier que les conversations entre les agents sont des négociations, les seules dépendances que ce service peut gérer sont des

dépendances entres les exécutions des différentes actions (i.e. envoi des propositions) composant une négociation afin de gérer le flux cohérent de la conversation.

3.4. Synthèse

Nous avons présenté dans ce chapitre les différentes approches de coordination de négociations ou d'interaction existantes. Au fil de leur présentation, nous avons souligné leurs limitations vis-à-vis de notre problématique. Cette étude nous permet de mettre en évidence un ensemble de besoins à satisfaire par le modèle de coordination de négociations à développer. La synthèse présentée dans le tableau 2 présente ces différentes approches en fonction de la manière dont la coordination est abordée (*stratégique ou générique*), des dépendances prises en compte, de la gestion *centralisée ou distribuée* de la coordination, et, enfin, de la vision *locale ou globa*le développée par la coordination.

	Stratégique/ Générique	**Centralisé/ Distribué**	**Locale/ Globale**	**Type de dépendances gérées**
ADEPT	Stratégique	Distribué	Globale	- Dépendances entre états finaux de négociations (coopération) *: dépendances de statuts*
MACIV	Stratégique	Distribué	Globale	- Dépendances entre états finaux de négociations (compétition) *: dépendances de statuts*
Multi-link négociation	Stratégique	Distribué	Locale	- Dépendances entre les états initiaux et finaux de négociations (coopération et compétition) *: dépendances de statuts* - Dépendances entre objets de négociations *: dépendances d'issues*
Planification des réunions	Stratégique	Distribué	Locale	- Dépendances entre états finaux de négociations (coopération et compétition) *: dépendances de statuts* - Dépendances entre objets de négociations et dépendances entre valeurs des attributs composant les objets de négociations*: dépendances d'issues*

HP	Générique	Centralisé	Locale	- Dépendances entre les états des actions faites au cours des négociations *: dépendances de statuts*
I.O.P.	Générique	Distribué	Globale	- Dépendances entre les états des actions faites au cours des négociations *: dépendances de statuts*

Tableau 2. : Synthèse sur les systèmes de coordination

Reprenant les différentes dimensions du processus de négociation que nous avons mises en évidence dans le chapitre 2, nous pouvons noter que la coordination stratégique est directement influencée par les dimensions telles que : *les participants et leurs rôles*, *le temps*, *l'objet de négociation* et *le protocole de négociation*. Egalement, comme ce type de coordination est mis en œuvre dans le processus de décision, la coordination stratégique est donc fortement influencée par *la stratégie de négociation*. La coordination stratégique fixe des dépendances entre *les états initiaux et finaux des négociations, entre les objets négociés* ou *entre les attributs décrivant ces objets*. En ce qui concerne la coordination générique, celle-ci ne prend en compte au niveau du processus de négociation que les événements liés à la gestion des interactions entre activités, actions ou messages. Ni l'objet à négocier, ni les participants à la négociation ne sont considérés. Ainsi, de manière générale, la coordination générique n'est influencée par aucune des dimensions du processus de négociation.

4. Conclusion et objectifs

Après cet état de l'art détaillant le processus de négociation et les différentes approches de coordination de négociations, nous allons préciser les objectifs et les contraintes qui sous-tendent le modèle que nous allons présenter dans la partie suivante.

L'état de l'art sur la négociation (voir chapitre 1 et 2) nous a permis de mettre en lumière que la négociation que nous voulons modéliser implique *plusieurs participants* négociant sur un objet de négociation décrit par *plusieurs attributs inter-dépendants* et chaque agent gère ses propres informations sur *les stratégies* qu'il utilise ou *les buts* qu'il essaie de satisfaire. Ainsi, pour conduire une ou plusieurs négociations de manière efficace, en prenant en compte toutes leurs dimensions, des mécanismes de coordination avec de fonctionnalités bien ciblées sont nécessaires.

Dans les différents systèmes de coordination présentés dans le chapitre 3 de l'état de l'art nous pouvons noter qu'aucun des systèmes propose une coordination entre les phases intermédiaires de négociations afin de corriger dynamiquement les différentes propositions faites au cours des négociations. Cette limitation peut s'expliquer dans une coordination générique par le fait que les données nécessaires sont indisponibles. Dans une coordination stratégique, bien que les données soient disponibles, la coordination par un processus de décision n'est pas le moyen optimal pour implémenter la gestion de ce type de dépendances. En effet, les approches stratégiques qui essaient de prendre en compte des données dynamiques sont lourdes du fait que chaque changement dans les données au cours de route suppose un redémarrage du processus de décision dès le début, car le processus de décision s'appuie sur l'historique.

Certaines caractéristiques des systèmes présentés sont similaires à nos objectifs, mais notre problème est plus complexe par le fait que nous voulons coordonner des actions (négociations) qui sont exécutées par des agents en compétition pour accomplir non seulement une tâche mais plusieurs tâches différentes. Une différence supplémentaire est que les dépendances à gérer concernent les valeurs des attributs des propositions envoyées au cours des négociations. Ainsi, au lieu de coordonner seulement les états initiaux et finaux des négociations, nous coordonnons aussi les états intermédiaires en regardant et en modifiant le contenu des messages échangés au cours des négociations.

Ainsi, notre approche de la coordination de négociations est structurée selon les deux modèles mis en exergue dans les systèmes existants : *coordination stratégique* gérée au *niveau agent* et *coordination générique* gérée au *niveau intergiciel*. Cette approche, multi-agent avec intergiciel, permet ainsi de déployer facilement le système multi-agent sur une architecture distribuée et concurrente. Les aspects synchronisation et distribution spatiale du processus de négociation sont déportés au niveau intergiciel. Dans le contexte de l'approche hybride système multi-agent et intergiciel, que nous avons envisagé dans le cadre du projet E-Alliance, le processus de négociation peut être décomposé en trois processus: i) un processus de coordination, ii) un processus de décision et iii) un processus de communication.

Cette structuration du processus de négociation en trois niveaux se justifie par la complexité du processus de négociation impliquant de multiples dimensions et plusieurs mécanismes.

Nous allons maintenant décliner les objectifs à satisfaire pour chacun de ces processus.

4.1. Processus de coordination

Dans les différents systèmes s'intéressant à la coordination de négociations, nous avons pu constater que le processus de négociation est souvent structuré en seulement un processus de communication et un processus de décision. Le processus de décision concerne la phase de délibération et de construction de l'offre. Le processus de communication (souvent construit autour d'un seul protocole) est impliqué dans la communication de l'offre élaborée. Dans le même temps les systèmes présentés proposent aussi des mécanismes de coordination utilisant soit des stratégies de négociation soit des protocoles de communication. Avec les approches stratégiques, *le processus de décision* essaie de coordonner la négociation en prenant en compte les dimensions d'une négociation comme les participants, l'historique ou la structure de l'objet. Avec les approches plus génériques, *le processus de communication* fixe des contraintes et propose une coordination en fonction des événements mais ne prenne pas en compte beaucoup de caractéristiques d'une négociation.

Ainsi, le *processus de coordination* n'est pas toujours bien délimité au sein de ces deux processus.

Par exemple, en définissant les règles de conversation comme un niveau d'abstraction au-dessus du protocole (« conversation policy ») c'est le processus de communication qui gère la coordination. Par ailleurs, en définissant des plans globaux de conversation, c'est le processus de décision qui choisit quelles sont les actions qui peuvent être exécutées. Les protocoles peuvent être utilisés pour maintenir une cohérence locale dans un dialogue impliquant un petit nombre de messages. En revanche la cohérence du dialogue sera très difficile à maintenir si la négociation devient plus complexe (augmentation du nombre de participants, du nombre de protocoles utilisés dans le même processus de négociation, du nombre des conversations appartenant à une même négociation). Par exemple, dans les négociations de type « un à plusieurs » le protocole de type enchère n'est pas suffisant pour assurer la coordination des propositions si un des participants est engagé dans plusieurs enchères en même temps [Ito et al.'00]. De la même façon, un seul protocole ne peut pas suivre deux fils de conversations concurrentes et dépendantes dans une même négociation: 1) une dans laquelle le vendeur accepte de vendre l'article, et 2) l'autre dans laquelle le vendeur demande un paiement à l'acheteur [Flores et Kremer'02]. Un autre cas dans lequel le protocole ne peut pas gérer la totalité de la coordination du processus de négociation est la modélisation de la négociation sur plusieurs niveaux d'engagement [Sandholm et Lesser'95]. Dans ces

négociations, un agent peut à tout moment abandonner un contrat pour s'engager dans un autre jugé plus rentable.

Egalement, dans le cas où le processus de décision assure une coordination des négociations, la lourdeur et l'efficacité sont vite rédhibitoires lorsque le nombre de négociations, le nombre de participants et le nombre d'attributs impliqués dans les propositions s'accroissent.

Mais, dans tous ces cas, la mise en œuvre d'une négociation suppose l'existence d'un mécanisme ou d'un processus de coordination bien structuré. Dans ce cas, on appelle ***processus de coordination***, le processus qui a une vision globale sur la négociation afin de gérer le parallélisme et la dépendance entre les actions exécutées dans une négociation complexe.

De cette manière, nous abordons la négociation comme un processus qui conserve l'autonomie des participants - chacun d'entre eux gère ses propres négociations et ses propres informations (description des tâches, des contrats, des partenaires, … etc.). De plus, deux aspects importants sur le processus de négociation doivent être soulignés: *i)* pour une seule tâche, un participant est impliqué dans plusieurs négociations bilatérales – une pour chaque participant intéressé par la contractualisation de la tâche et *ii)* au même moment, le même participant se trouve également engagé dans d'autres négociations bilatérales sur d'autres tâches différentes.

Donc, *le processus de coordination* est le processus qui gère pour chaque participant trois aspects importants : 1) la coordination des différentes propositions reçues dans une *négociation bilatérale (un à un)* afin de modéliser une négociation *multi-phase* sur un objet de négociation *multi-attribut*, 2) la coordination des *plusieurs négociations bilatérales* afin de modéliser une négociation *multi-participant*, et 3) également, la coordination des *plusieurs négociations* où le participant se trouve engagé.

Objectif : L'objectif est de proposer des mécanismes de base pour la coordination de négociations venant s'interfacer entre le processus générique de communication et le processus de décision, afin de mettre en place des schémas de négociation qui gèrent l'évolution concurrente de plusieurs négociations. Ces mécanismes de coordination doivent maintenir la consistance sur la rationalité de l'agent en termes des actions qui peuvent être exécutées (initialisation ou finalisation d'une négociation) et des propositions qui peuvent être envoyées.

Cette consistance doit être aussi garantie au niveau *local* (pour une seule négociation) qu` au niveau *global* (pour la totalité des négociations dans lesquelles un atelier est impliqué).

4.2. Processus de décision

Le processus de décision s'intéresse à la création et à l'évaluation des offres échangées au cours d'une négociation. Son degré de complexité est lié à *la stratégie de négociation* suivie par l'agent et qui elle-même est liée à la nature de *l'objet de négociation*, au *nombre des participants*, au *contexte de négociation* et même au *protocole de négociation*.

La stratégie de négociation d'un agent représente la modalité utilisée par l'agent afin de mettre en ordre les actions qu'il projette à faire pendant le processus de négociation. La stratégie suivie par un agent dépend de ses connaissances et d'une *fonction d'utilité.* Cette fonction permet à un agent de calculer le niveau de son engagement suite à chacune de ses propositions.

Il y a une dépendance directe évidente entre les données utilisées par une stratégie et le temps associé à cette stratégie afin d'évaluer ou de créer une proposition. A une extrémité, nous avons les stratégies *locales* dans lesquelles un agent impliqué dans une négociation construit sa proposition courante en se basant seulement sur sa dernière proposition et en ne faisant aucune supposition sur les fonctions d'utilité des autres participants [Barbuceanu et Lo'00]. A l'autre extrémité, nous avons les stratégies *globales* dans lesquelles les agents construisent leurs propositions courantes utilisant des stratégies qui se basent sur la totalité de la séquence des offres et contre-offres dans toutes les négociations en cours, et aussi sur leurs connaissances vis à vis des stratégies employées par leurs opposants. Ces stratégies calculent et prévoient tous les mouvements futurs et elles sont efficaces pour trouver le meilleur accord, mais elles sont aussi très lourdes en temps de calcul [Rosenschein et Zlotkin'94].

Evidement ces stratégies peuvent offrir aussi, jusqu'à un certain niveau, des fonctionnalités de coordinations entre les propositions faites dans une seule négociation (*stratégie locale*) ou entre les propositions faites dans plusieurs négociations (*stratégie globales*).

Comme nous l'avons vu dans les systèmes existants ce processus de décision peut être très complexe et, selon la stratégie utilisée (*locale* ou *globale*), il peut incorporer également des aspects attachées à la coordination des négociations. Dans notre travail, la négociation est utilisée afin de permettre à un Manager d'un atelier d'impression de sous-traiter ou de contractualiser une ou plusieurs tâches dans l'alliance. Pour cela, c'est au niveau décisionnel que le processus de négociation doit être capable de représenter et de manipuler des propositions *multi-attributs* afin d'offrir au Manager la possibilité de faire progresser et de

conclure les négociations. C'est donc le processus de décision qui doit élaborer et évaluer les propositions faites dans toutes les négociations dans lesquelles un atelier est impliqué. *Le processus de décision* doit donc permettre au manager de répondre aux questions suivantes: - quelle est la première proposition, quelle est la proposition suivante si la première n'a pas été acceptée, quand abandonner la négociation, quand conclure avec un contrat ? Dans ses réponses à ces questions, il doit tenir compte, principalement, des contraintes et des préférences de l'utilisateur (le Manager), mais il doit également tenir compte des règles de fonctionnement fixées par les autres participants impliqués dans le processus de négociation.

Objectif : Notre objectif est de proposer un ensemble de modèles et de structures de données afin de permettre à un participant (humain ou agent) de spécifier la structure d'un objet de négociation avec les contraintes et les dépendances entre ses multiples attributs, les participants avec lesquels une négociation est envisageable et les stratégies à utiliser pendant la négociation. Notre objectif n'est pas de proposer un processus de décision ou une nouvelle stratégie, mais de délimiter quelles sont les informations et les données qui peuvent être définies à ce niveau et par la suite utilisées ou modifiées au sein du processus de coordination et du processus de communication.

4.3. Processus de communication

La négociation que nous modélisons est une négociation *multi-participants* et *multi-phases*. Comme nous l'avons vu dans ce qui précède, le processus de communication mis en œuvre est *asynchrone* : tous les participants peuvent envoyer des propositions à n'importe quel moment de temps et à partir de n'importe quelle phase intermédiaire de la négociation. Mais le processus de communication va également au-delà et maintient une conversation structurée en une séquence de messages bien définie. Les messages sont construits, la plupart du temps, selon un schéma issu de la théorie des actes de langage, proposant des primitives de communication pour la négociation, intégrant leur sémantique et leur usage dans les protocoles de négociation. Généralement, ces langages donnent seulement des indications pour construire les conversations (autorisation de certaines séquences de messages), mais ne font pas d'hypothèses sur le contenu des messages. En revanche, les protocoles d'interaction mis en œuvre gèrent des règles qui sont plus figées et également plus strictes afin de réduire les inconsistances dans le processus de communication en fixant des conditions temporelles et/ou contextuelles sur :

1. *la durée de la négociation* – des règles fixant la finalisation de la négociation en fonction d'un indicateur spécifié auparavant (i.e. temps ou nombre des messages échangés);
2. *la structuration de la négociation* – des règles spécifiant, en fonction des messages reçues, quelles sont les réponses possibles;
3. *la visibilité des offres* – des règles indiquant quels sont les agents qui peuvent recevoir une certaine proposition ;
4. *l'identification d'une offre* – des règles fixant la modalité par laquelle chaque proposition ou message envoyé dans une négociation est identifié de manière unique.

Ainsi, de manière générale, dans les approches existantes, le processus de communication manipule des structures des données qui précisent quels sont les actes de langage qui peuvent être utilisés, quand les employer, à qui la communication doit s'adresser et les actions à mettre en œuvre lors de la réception des réponses prévues. Le processus de communication manipule des préoccupations attachées aux protocoles et à la sémantique des langages de communication entre agents. Le processus de communication a donc aussi des attributs de coordination.

Objectif : Notre objectif est de modéliser le processus de communication comme un processus générique de propagation et d'échange d'alternatives qui fait abstraction du contenu des messages ou du protocole de négociation utilisé. Les informations relatives aux attributs de la tâche négociée associées aux propositions échangées et aussi celles relatives aux primitives de communication utilisées, leurs sémantiques et leur enchaînement devront donc être transparentes pour ce processus. *Le processus de communication* doit s'intéresser seulement aux aspects transactionnels du processus de négociation et s'intéresser à la synchronisation des différentes propositions faites par les participants dans une négociation : cohérence sur la succession des propositions et contre-propositions pour chaque participant dans une négociation, cohérence entre les images aperçues par tous les autres partenaires concernés. Libérant ainsi le processus de communication de ces dimensions attachées à la coordination, nous pourrons mettre en œuvre différents modèles de négociation, allant des enchères aux négociations intégratives, impliquant un ou plusieurs partenaires et des objets de négociations complexes, structurés selon différentes étapes de contractualisation

III. Modèle pour la Coordination de Négociations

Afin de définir un processus de coordination s'inscrivant dans notre vision du processus de négociation développée dans le chapitre précédent, nous allons proposer tout d'abord un modèle de description de la négociation entre participants d'une alliance. A partir de celui-ci, nous pourrons mettre en avant les aspects qui peuvent faire l'objet d'une coordination, les dépendances qui peuvent être définies et la manière dont celles-ci peuvent être gérées au niveau du modèle de coordination envisagé.

Modèle de négociation : le modèle de négociation sera utilisé pour décider et coordonner une négociation *multi-bilatérale* sur un objet *multi-attribut*. Il doit donc inclure les structures permettant de représenter et de modifier tout au long du processus de négociation :

- un *objet de négociation complexe* décrit par un *ensemble d'attributs* de négociation avec des valeurs associées ;
- la possibilité que *plusieurs participants* soient impliqués dans une même négociation où chacun peut avoir *une* ou *plusieurs vues différentes* sur cette négociation ;
- la gestion d'un *ensemble de propositions* participant à la construction de la négociation. Chaque proposition est une *phase de négociation active* à partir de laquelle la négociation peut continuer à tout moment.

Comme nous avons précisé, le principal objectif de ce travail est de proposer un modèle de coordination de négociations dans une système dynamique avec des agents autonomes où chacun gère ses propres négociations.

Ainsi, au *local,* ceci exige une description formelle des règles de coordination qui régissent le comportement de l'agent dans une négociation et, au *global*, le modèle doit assurer une coordination globale de la totalité des négociations d'un agent.

Nous visons à proposer un modèle de coordination structuré dans des briques (*schémas de coordination*) qui regroupent des relations basiques de dépendance. En faisant cette structuration nous essayons de garantir les propriétés suivantes :

- *flexibilité :* le modèle de coordination doit garantir ses fonctionnalités dans des situations de dépendances prédéfinies, mais aussi dans des situations de dépendances définies dynamiquement ;
- *décentralisation :* la coordination n'est pas faite dans un seul module, mais dans plusieurs. Cette propriété implique que les modules peuvent fonctionner simultanément afin d'augmenter l'*efficacité* de l'exécution du modèle.
- *réutilisabilité:* définition des modules spécialisés pour des cas basiques de coordination afin de pouvoir être utilisés dans n'importe quelle négociation.
- *émergence*[8] *:* le modèle de coordination n'est pas un mécanisme centralisé qui garantit la coordination entre les négociations de plusieurs agents (ateliers), mais un modèle de coordination des négociations fondé sur un mécanisme "conceptuel distribué" [Ossowski'99]. La coordination globale des négociations pour un participant de l'alliance, va émerger comme une conséquence de la synchronisation des différents modules de coordination disponibles pour chaque participant.

Nous allons maintenant présenter nos modèles de négociation et de coordination des négociations qui veulent répondre à toutes les exigences présentées précédemment. La description du modèle pour la coordination des négociations faite dans cette partie est divisée en deux: *un modèle générique de coordination* et *un modèle stratégique de coordination*.

Cette partie est structurée en cinq chapitres. Dans le chapitre 5, nous décrivons les éléments de base du modèle de négociation au sein d'une alliance présentant les différentes caractéristiques détaillées auparavant : système décentralisé et environnement distribué et ouvert. Dans le chapitre 6, nous nous appuyons sur ces différents éléments pour proposer un modèle de processus de négociation. La mise en place de ce modèle s'appuie sur la métaphore

[8] Comme dans [Ossowski'99], nous sommes concernés par " layered emergence " : une propriété est émergée si elle peut être attribuée à une entité à un certain niveau de complexité ou d'abstraction, mais elle ne peut pas être détectée à des niveaux plus bas (Conte & Gilbert 1995 Computer simulation for social theory)

Interaction Abstract Machines (IAM) afin de pouvoir modéliser l'évolution au cours du temps d'une négociation *multi-attribut, multi-participant, multi-phase*. Ayant clairement défini le processus de négociation, nous pouvons présenter notre modèle de coordination constitué d'un modèle générique de coordination (chapitre 7) et d'un modèle stratégique de coordination (chapitre 8).

5. Modèle de négociation

Comme nous l'avons précisé en introduction à ce mémoire, la problématique abordée dans cette thèse s'intéresse au processus de négociation dans le cadre d'une alliance inter-organisationnelle. Ainsi, dans ce chapitre afin de mettre en place les fondements de notre travail, nous allons présenter le modèle de négociation au sein d'une telle alliance. Dans la première partie nous proposons une description formelle d'une alliance qui fixe le cadre général dans lequel les participants peuvent négocier, afin de détailler par la suite les participants, leurs négociations et les dépendances possibles entre les négociations gérées dans le cadre d'une telle alliance.

5.1. Fondements

L'objectif principal du système informatique proposé dans le projet E-Alliance est de préserver l'*ouverture* de l'alliance mise en place, d'une part, et l'*autonomie* des organismes groupés dans une alliance d'autre part, tout en permettant la *simultanéité* de leurs activités, la *flexibilité* de leurs négociations et le *dynamisme/évolution* de leur environnement. Dans ce contexte général, la description de l'alliance que nous proposons prend en compte les caractéristiques suivantes :

- chaque participant (organisation) peut entrer et sortir de l'alliance à tout moment. Le nombre et l'identité des participants sont donc fortement dynamiques dans l'alliance ;
- chaque participant est responsable de sa propre charge de travail, de ses propres clients et de ses contrats avec les clients extérieurs ou avec les autres participants de l'alliance. Dans ce type d'alliance, il est réaliste de supposer qu'à tout moment de nouvelles tâches peuvent être proposées ou de nouveaux contrats peuvent être établis. A tout moment des négociations peuvent donc être démarrées, arrêtées ou finalisées ;
- à l'intérieur d'une seule organisation, il est possible et même probable que plusieurs négociations se déroulent au même moment et que l'organisation essaie de gérer les différentes influences qui peuvent exister entre ces

négociations. Comme dans les systèmes présentés à la section 3.3, dans notre système nous essayons aussi de représenter et de coordonner des dépendances entre négociations.

Ces trois caractéristiques sont modélisées dans la description formelle de l'alliance par trois ensembles $\mathcal{P}$, $\mathcal{N}$ et $\mathcal{R}$ qui représentent respectivement les ensembles complets des participants, des négociations et des règles de gestion des dépendances définies dans l'alliance :

DEFINITION **Alliance** : Une alliance est définie comme un quintuplet $\mathcal{A} = <\mathcal{T}, \mathcal{P}, \mathcal{N}, \mathcal{R}, O>$ où :

1. $\mathcal{T}$ désigne *le temps du système*. Nous supposons qu'il est discret, linéaire [Emerson'90] et uniforme.
2. $\mathcal{P}$ désigne *l'ensemble des participants de l'alliance*. Ces participants peuvent être impliqués dans une ou plusieurs négociations au sein de l'alliance.

   Nous noterons $\mathcal{P}=\{p_i \mid i \in \mathbb{N}\}$ cet ensemble, p_i désigne le i -éme participant de l'alliance.
3. $\mathcal{N}$ désigne *l'ensemble des négociations* prenant place au sein de l'alliance.

   Nous noterons $\mathcal{N}=\{N_j \mid j \in \mathbb{N}\}$ cet ensemble, $\mathbf{N}_j$ désigne la j -éme négociation.
4. $\mathcal{R}$ désigne *l'ensemble des politiques de coordination des négociations* se déroulant au sein de l'alliance.

   Nous noterons $\mathcal{R}=\{R_k \mid k \in \mathbb{N}\}$ cet ensemble, R_k désigne la k-éme politique de coordination.
5. O désigne l'ontologie commune constituée de l'ensemble des définitions des attributs utilisés dans une négociation.

Une alliance $\mathcal{A}$ est ainsi décrite à un instant *t* par l'ensemble de négociations en cours ou qui se sont déroulées en son sein, l'ensemble de participants membres de l'alliance, impliqués ou non dans ces négociations, l'ensemble de politiques de coordination que ces participants utilisent pour gérer les négociations dans lesquelles ils sont impliqués et l'ensemble des attributs pouvant être négociés dans l'alliance. Dans les approches de négociation basées sur une succession linéaire de propositions (cf. système ADEPT, MACIV – section 3.3.1.), une négociation ne peut se poursuivre qu'à partir de la dernière proposition

réalisée. Dans notre modèle, nous choisissons de modéliser une *négociation* sous forme d'une négociation multi-phase pouvant se poursuivre et se ramifier à partir de n'importe quel moment de temps (phase) et de n'importe quelle proposition. Dans cette optique, comme nous le verrons par la suite, une négociation sera représentée sous forme d'un graphe.

Avant de présenter plus en détail chacun des composants de l'alliance (*participant*, *négociation*, *politique de coordination*), ainsi que les relations qui existent entre eux, nous introduisons l'exemple suivant que nous utiliserons tout au long de ce chapitre (cf. Fig. 8).

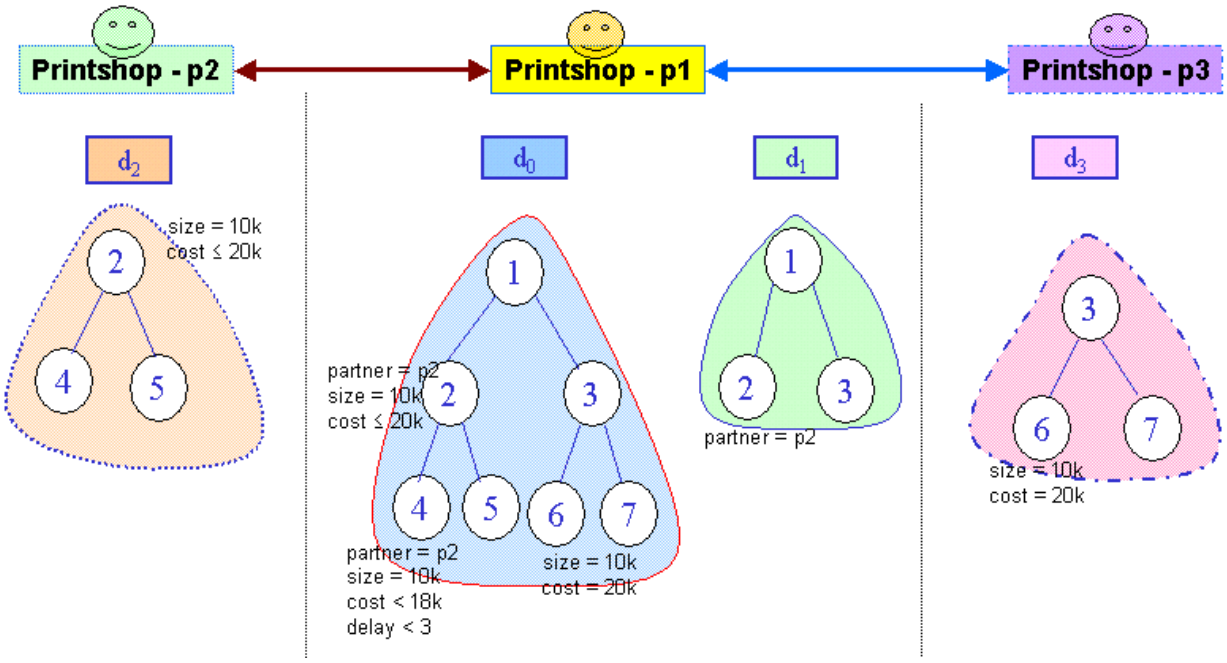

Figure 8. Description de la négociation N_1 à l'instant de temps t1

d_0 description de la négociation N_1 vue par le partenaire p_1 selon la politique R_0
d_1 description de la négociation N_1 vue par le partenaire p_1 selon la politique R_1
d_2 description de la négociation N_1 vue par le partenaire p_2 selon la politique R_2
d_3 description de la négociation N_1 vue par le partenaire p_3 selon la politique R_3

Considérons la négociation N_1 qui implique un donneur d'ordre potentiel (l'atelier p_1) et différents contractants possibles (les ateliers p_2 et p_3). Les trois ateliers p_1, p_2 et p_3 sont impliqués dans la même négociation, *composition de deux négociations bilatérales (entre p_1 et p_2 d'une part et entre p_1 et p_3 d'autre part)*. Chaque participant a ses propres connaissances sur la négociation courante. Il fixe et gère en propre les politiques de coordination qui lui sont attachées. Ainsi, chacun des participants a des descriptions différentes (graphes sur la Fig. 8) de cette négociation N_1. La figure 8 représente la négociation à l'instant de temps t_1. La

négociation étant un processus évolutif, ces descriptions vont évoluer en accord avec les politiques de coordination fixées par chaque participant.

Dans ce qui suit, nous allons présenter les ensembles $\mathcal{P}$, $\mathcal{N}$ et $\mathcal{R}$ ainsi que les relations qui les lient. Nous détaillerons ensuite la manière dont les négociations sont décrites dans chaque partenaire. Nous terminerons par la définition des politiques de coordination.

5.3. Participant

Nous avons défini $\mathcal{P}$ comme *l'ensemble des participants* présents dans l'alliance. Cet ensemble est variable dans le temps dans la mesure où de nouveaux participants peuvent entrer ou sortir de l'alliance à tout moment. Un participant peut être décrit par une structure de données complexe correspondant à une description complète de l'organisation comme le nom, l'adresse, le type des services qu'il peut fournir ou le type des services qu'il peut chercher à sous-traiter[9] (ex.: Printshop1, 14 rue de la Libération, impression noir-blanc, impression couleur). Cette description est transparente pour la mise en place du modèle proposé. Dans cette thèse, nous utilisons seulement un identifiant unique du participant dans l'alliance qui sert à la designer dans les négociations auxquelles il participe.

DEFINITION **Participant** : Un *participant* $p_i \in \mathcal{P}$ est une organisation autonome de l'alliance tel que : $\forall p_j \in \mathcal{P},\ i \neq j \Rightarrow p_i \neq p_j$.

Dans notre exemple les trois ateliers d'impressions considérés sont identifiés respectivement par p_1, p_2 et p_3.

Entre les éléments des ensembles $\mathcal{P}$, $\mathcal{N}$ et $\mathcal{R}$ plusieurs relations existent. Nous les avons modélisées par des graphes de relations n-aire.

a) Relations entre $\mathcal{P}$ et $\mathcal{N}$

Soit le produit cartésien $\mathcal{T} \times \mathcal{P} \times \mathcal{N}$ et une relation Φ_1 définie sur $\mathcal{T} \times \mathcal{P} \times \mathcal{N}$ telle que :

[9] Une description complète du cycle de vie de l'alliance, avec la représentation des organisations participantes dans l'alliance et de leurs contrats d'adhésion à l'alliance est faite dans [Rap'01].

$\forall(t,p,N) \in \mathcal{T} \times \mathcal{P} \times \mathcal{N}$, $\Phi_1(t,p,N)$ si et seulement si à l'instant de temps t, le participant p est impliqué dans la négociation N.

Soit $E_{\Phi 1}$ désigne le graphe de la relation Φ_1, sous-ensemble de $\mathcal{T}$ x $\mathcal{P}$ x $\mathcal{N}$. Les requêtes réciproques associées à cet ensemble sont :

- *participants* requête définie sur $E_{\Phi 1}$ vers une partie de l'ensemble $\mathcal{P}$ qui retourne l'ensemble des participants impliqués dans la négociation N_j à un instant de temps t avec $t \in \mathcal{T}$ et $N_j \in \mathcal{N}$.
- *negotiations* requête définie sur $E_{\Phi 1}$ vers une partie de l'ensemble $\mathcal{N}$ qui retourne l'ensemble des négociations dans lesquelles un participant p_i est impliqué à un instant de temps t.

Dans l'exemple précédent, à l'instant t_1 nous avons $participants(t_1,N_1) = \{p_1, p_2, p_3\}$ et $negotiations(t_1, p_1)=\{ N_1\}$.

Nous appuyant sur notre recherche sur *la dimension participant* de la négociation (voir sections 1.1. et 2.1.) et sur les différents types de rôles possibles dans une négociation, nous considérons dans notre modèle deux types de rôles: 1) *initiator* – si le participant initie la négociation *N* en proposant une tâche à sous-traiter dans l'alliance ; et 2) *guest* -si le participant est invité dans la négociation *N* comme candidat possible à la sous-traitance de la tâche négociée.

DEFINITION **Role** : *role* est une fonction définie sur $E_{\Phi 1}$ vers l'ensemble *Role={initiator, guest}* :

$$role : E_{\Phi 1} \rightarrow Role,$$

avec la propriété que un participant a un seul rôle dans une négociation et que celui-ci ne change pas dans le temps:

$\forall t_1, t_2 \in \mathcal{T} \wedge \exists N \in \mathcal{N} \wedge \forall p \in \mathcal{P}$: $p \in participants(t_1,N) \wedge p \in participants(t_2,N) \Rightarrow role(t_1,p,N)=role(t_2,p,N)$.

Comme nous modélisons une négociation de type un-à-plusieurs, un seul participant pourra avoir le rôle d'initiator dans une négociation - $length(initiator\ (t,N)) = 1$; et un ou plusieurs participants pourront avoir le rôle de guest - $length(guest\ (t,N)) \geq 1$.

Les fonctions *guest(t,N)* $\subset \mathcal{P}$ et *initiator(t,N)* $\subset \mathcal{P}$ retournent l'ensemble des participants impliqués dans la négociation N à l'instant t ayant le rôle de *guest* et d'*initiator* respectivement. La fonction *length()* retourne le cardinal d'un ensemble.

Les relations entre ces différents ensembles sont les suivantes :

guest(t,N) $\cap$ *initiator(t,N)* $= \varnothing$

guest(t,N) $\cup$ *initiator(t,N) = participants(t,N).*

Dans notre exemple le participant p_1 a le rôle de *initiator* et les participants p_2 et p_3 ont les rôles de *guest*s. Comme nous l'avons précisé ci dessus, le rôle d'un participant dans une négociation ne varie pas au cours du temps, mais les participants p_1, p_2 ou p_3 peuvent avoir d'autres rôles dans d'autres négociations.

b) Relations entre $\mathcal{P}$ et $\mathcal{R}$

Utilisant le même modèle de notation présenté auparavant, nous définissons pour le produit cartésien $\mathcal{T} \times \mathcal{P} \times \mathcal{R}$ et le sous-ensemble $E_{\Phi 2}$ désignant le graphe de la relation Φ_2 :

$\forall$(t,p,R) $\in \mathcal{T} \times \mathcal{P} \times \mathcal{R}$, *$\Phi_2$(t,p,R) si et seulement si à l'instant de temps t, le participant p gère la politique de coordination R.*

Les requêtes réciproques associées à cet ensemble sont :

- *coordinator* requête définie sur $E_{\Phi 2}$ vers une partie de l'ensemble $\mathcal{P}$ qui retourne les participants gérant la politique de coordination R_k à un instant de temps t avec $t \in \mathcal{T}$ et $R_k \in \mathcal{R}$.
- *politiques_p* requête définie sur $E_{\Phi 2}$ vers une partie de l'ensemble $\mathcal{R}$ qui retourne l'ensemble des politiques de coordination que le participant p_i gère à un instant de temps t avec $t \in \mathcal{T}$ et $p_i \in \mathcal{P}$.

Dans notre exemple nous avons : coordinator(t,R_1) = {p_1} et politiques_p(t,p_1)={ R_0, R_1}.

5.4. Négociation

Nous avons défini $\mathcal{N}$ *l'ensemble des négociations* qui ont lieu dans l'alliance. Comme dans le cas de l'ensemble des participants, le contenu de cet ensemble varie au cours du temps, à tout moment dans l'alliance, de nouvelles négociations peuvent démarrer et d'autres peuvent s'arrêter. Mais dans notre modèle de l'alliance une négociation est associée à un identificateur unique : $\forall N_i, N_j \in \mathcal{N}$ avec $i \neq j$ nous avons $N_i \neq N_j$.

Avant de définir plus en détail le modèle de négociation nous pressentons premièrement les relations entre $\mathcal{N}$ et $\mathcal{R}$.

Relations entre $\mathcal{N}$ et $\mathcal{R}$

Dans la section précédente nous avons vu que chaque participant peut être impliqué dans plusieurs négociations et qu'une négociation peut impliquer plusieurs participants. De manière similaire nous définissons les relations entre les ensembles $\mathcal{N}$ et $\mathcal{R}$.

Ainsi, nous définissons pour le produit cartésien $\mathcal{T} \times \mathcal{N} \times \mathcal{R}$ le sous-ensemble $E_{\Phi 3}$ comme étant le graphe de la relation Φ_3 telle que:

$\forall(t,N,R) \in \mathcal{T} \times \mathcal{N} \times \mathcal{R}$ $\Phi_3(t,N,R)$ si et seulement si à l'instant de temps t, la négociation dont l'identificateur est N fait l'objet de dépendances gérées par la politique de coordination R.

Les requêtes réciproques associées à cet ensemble sont :

- *manage* requête définie sur $E_{\Phi 3}$ vers une partie de l'ensemble $\mathcal{N}$ qui retourne l'ensemble des négociations dépendantes considérées par la politique de coordination R_k à un instant de temps t avec $t \in \mathcal{T}$ et $R_k \in \mathcal{R}$.
- *politiques_n* requête définie sur $E_{\Phi 3}$ vers une partie de l'ensemble $\mathcal{R}$ qui retourne l'ensemble des politiques de coordination dans lesquelles la négociation N_j est impliquée à l'instant de temps t avec $t \in \mathcal{T}$ et $N_j \in \mathcal{N}$.

Dans notre exemple les différentes requêtes retourneront les valeurs suivantes : $manage(t,R_1) = \{N_1\}$ - en supposant aussi que le participant p_1 est impliqué seulement dans la négociation N_1 à ce moment de temps et $politiques\_n(t,N_1)=\{ R_0, R_1, R_2, R_3\}$ – avec la même supposition que les participants p_2 et p_3 sont aussi impliqués seulement dans la négociation N_1 à ce moment de temps.

Avec ce dernier couple de deux relations nous avons complété le cycle de dépendance réciproque entre les trois ensembles $\mathcal{P}$, $\mathcal{N}$ et $\mathcal{R}$. La cohérence de ce cycle peut être illustrée en partant de n'importe lequel des trois ensembles et dans n'importe laquelle de directions. Par exemple en partant de l'ensemble $\mathcal{P}$ vers l'ensemble $\mathcal{R}$ nous pouvons avoir les relations suivantes :

- soit un participant p_x impliqué à l'instant t dans la politique de coordination R_x

$\exists p_x \in \mathcal{P}\ \exists R_x \in \mathcal{R}$: $R_x \in$ politiques_p$(t,p_x) \Rightarrow$

$\exists N_x \in \mathcal{N}$: $N_x \in$ manage$(t,R_x) \wedge p_x \in$ participants(t,N_x)

Afin de modéliser le fait que chaque participant a une description propre de la négociation dans laquelle il est impliqué, nous définissons une négociation comme suit.

DEFINITION **Négociation** : Une *négociation* est décrite à un instant de temps par un ensemble de descriptions de négociation.

Par la suite, nous considérons $\mathcal{D}$ l'ensemble des *descriptions de négociations* présentes dans l'alliance.

Soit $\mathcal{D} = \{d_i \mid i \in \mathbb{N}\}$ désigne l'ensemble des descriptions de négociations telles que : $\forall$ $d_i, d_j \in \mathcal{D}$ avec $i \neq j$ nous avons $d_i \neq d_j$.

Une première relation peut être définie entre les ensembles $\mathcal{D}$ et $\mathcal{N}$. Selon notre définition une négociation est un ensemble de descriptions de négociation.

Ainsi, soit la fonction *neg_desc* de $\mathcal{T}$ x $\mathcal{N}$ vers une partie de $\mathcal{D}$ qui retourne l'ensemble des descriptions attachées à une négociation à un instant donné.

$$neg\_desc : \mathcal{T}x\ \mathcal{N} \rightarrow P(\mathcal{D}).$$

Dans l'exemple présenté sur la Fig.8, $neg\_desc(t,N_1)=\{d_0,d_1,d_2,d_3\}$ représente l'image de la négociation N_1 à l'instant de temps t. Par la suite, afin de simplifier les notations, nous utilisons $N(t)$ à la place de $neg\_desc(t,N)$.

Afin de définir précisément une négociation nous allons définir les différents éléments qui la composent. Nous introduisons par la suite les concepts de *description de négociation, graphe de négociation, phase de négociation* et *aperçu de négociation*.

5.4.1. Description de négociation

DEFINITION **Description de négociation** : Une *description de négociation* d_i, $d_i \in \mathcal{D}$ telle que $d_i \in N(t)$, est une séquence de graphes de négociation qui décrivent la négociation N de sa création au moment du temps *t*.

Utilisant les notions et les propriétés de la séquence comme le type de donnée abstrait utilisée en informatique, nous pouvons décrire plus précisément une description de la manière suivante. Nous considérons que dans le cadre de ce type d'ensemble l'indexation est fait en fonction du temps $\mathcal{T}$ et ainsi nous pouvons définir une fonction partielle *itemAt* sur P($\mathcal{D}\times\mathcal{T}$) qui retourne un graphe de négociation :

$$\forall t_i \in [t_0,t] \quad itemAt(d,t_i) = G_{t_i} \Leftrightarrow \exists N \in \mathcal{N} : d \in N(t_i)$$

De cette manière, chaque description de négociation *d* est une succession finie des graphes de négociation, une pour chaque instant de temps t_i dans lequel la négociation *N* décrite est en déroulement et la description considérée est active ($d \in N(t_i)$)

Par la suite, afin de simplifier les notations, nous utilisons *d(t)* à la place de *itemAt(d,t)*.

L`exemple de la Fig. 8 représente les différentes descriptions de négociation attachées à la négociation N_1. Chacune de ces descriptions est une séquence de graphes de négociation (Par souci de clarté nous n'avons fait apparaître sur la figure, que le graphe de négociation à l'instant courant). Remarquons que l'atelier p_1 a dans la description d_0 une représentation totale de la négociation et que les deux autres ateliers p_2 et p_3 n'en ont qu'une représentation partielle dans leurs descriptions respectives d_2 et d_3 : en effet, les propositions faites dans les nœuds 2, 4 et 5 ne sont vues que par les ateliers p_1 et p_2, les propositions faites dans les nœuds 3, 6 et 7 ne sont vues que par les ateliers p_1 et p_3. Notons également que les ateliers p_2 et p_3 ne sont pas « conscients » de leur participation mutuelle dans la négociation N_1. Seul l'atelier p1 a cette connaissance. Il a donc la tâche de gérer les dépendances entre les propositions dans les deux négociations bilatérales. De même, dans cet exemple nous avons fait apparaître une deuxième description pour le participant p_1 : le graphe d_1 contenant les nœuds 1, 2 et 3. Cette description résulte de la stratégie de négociation du participant p_1 qui a décidé de fixer en premier qui seront les participants avec lesquels il veut négocier la même tâche en parallèle.

Cet exemple montre les relations entre $\mathcal{D}$ et $\mathcal{P}$. De la même manière $\mathcal{D}$ est lié à $\mathcal{R}$ par le biais de la relation entre $\mathcal{N}$ et $\mathcal{R}$. Ainsi l'ensemble des descriptions de négociation $\mathcal{D}$ est lié d'une manière précise avec les ensembles $\mathcal{N}$, $\mathcal{P}$ et $\mathcal{R}$. Nous détaillerons ceci plus loin.

5.4.1.1. Graphe de négociation

Nous avons défini une description de négociation comme une séquence de graphes *de négociation.* Le choix de la structure du graphe est motivé par le besoin de modéliser une *négociation multi-phase*. Une phase de négociation est créée à partir d'une proposition ou d'une contre-proposition exprimées au cours de la négociation. Ainsi en associant les nœuds du graphe aux différentes phases, la négociation peut être poursuivie à partir de n'importe laquelle de ces phases. Les différentes propositions effectuées dans chaque phase sont associées aux nœuds. Un *graphe de négociation* est ainsi défini de la manière suivante :

DEFINITION **Graphe de négociation** : Un *graphe de négociation, G=(A,E)* est un graphe orienté dont les nœuds sont attachés aux phases de négociation (avec *A l'ensemble des nœuds*) et les arcs expriment la relation de précédence entre les phases de négociation apparaissant dans les nœuds du graphe (avec *E un ensemble d'arcs*).

Dans notre exemple, les graphes de la Fig. 8. représentent à un instant donné les différentes propositions (associées aux nœuds du graphe) qui ont été échangées dans cette négociation.

5.4.1.2. Phase de négociation

DEFINITION **Phase de négociation** : Une *phase* de la négociation N_i est un identificateur unique au sein de la négociation N_i dénotant une étape de la négociation considérée.

Soit $\mathcal{Ph}_{N_i} = \{ph_k, k \in \mathbb{N}\}$ -l'ensemble d'identificateurs des phases des négociations et chaque phase est identifiée de manière unique : $\forall ph_i, ph_j \in \mathcal{Ph}_{N_i}$ avec $i \neq j$ nous avons $ph_i \neq ph_j$

Pour simplifier, dans l'exemple nous avons utilisé des entiers comme identificateurs des phases de négociation N_1. Donc, ici l'entier 2 écrit dans un nœud est un identificateur de phase de négociation N_1.

Nous pouvons remarquer que le même identificateur est présent dans plusieurs graphes décrivant une même négociation. Ceci est du au fait que les descriptions d'une même négociation peuvent partager la même phase de négociation.

Soit *link*, fonction définie sur $\mathcal{T}$ x $\mathcal{D}$ x $\mathcal{Ph}$ vers une partie de l'ensemble $\mathcal{D}$ qui retourne l'ensemble des descriptions avec lesquelles une description d_i partage, à un instant t, la phase de négociation ph_k.

$$link : \mathcal{T} \times \mathcal{D} \times \mathcal{Ph} \rightarrow P(\mathcal{D}).$$

Dans notre exemple la description d_0 partage la phase de négociation 2 à l'instant t avec les descriptions d_1 *et* d_2 : $link(t, d_0, 2) = \{d_1, d_2\}$.

Ainsi, selon nos définitions des nœuds de négociation et phase de négociation nous pouvons constater qu'une phase de négociation peut être associée à plusieurs nœuds et qu'un nœud est associé à une seule phase. Par ce biais nous sommes capables de modéliser le fait que dans une négociation chacun des participants a une représentation locale de la proposition communiquée mais ils partagent aussi un identificateur commun pour cette proposition. Nous y reviendrons par la suite (voir section 5.4.2.).

Par la suite, afin d'accéder à partir d'un nœud a à l'identificateur de la phase attachée ph_i nous utilisons la notation *a.ph*.

5.4.1.3. Vue d'une description de négociation

Selon notre modélisation d'une négociation, nous voulons que chaque participant impliqué dans une négociation puisse avoir ses propres descriptions sur cette négociation et qu'il puisse gérer ses propres politiques de coordination. Ainsi, soit la relation *Vue* entre les éléments de $\mathcal{D}$ et de $\mathcal{N}$, $\mathcal{P}$ et $\mathcal{R}$.

Considérons la relation Φ_{vue} définie sur $\mathcal{T} \times \mathcal{P} \times \mathcal{N} \times \mathcal{R} \times \mathcal{D}$ telle que:

$\forall(t,p,N,R,d) \in \mathcal{T} \times \mathcal{P} \times \mathcal{N} \times \mathcal{R} \times \mathcal{D}$, $\Phi_{vue}(t,p,N,R,d)$ si et seulement si à l'instant de temps t, le participant identifié par p, participe à la négociation N ($p \in$ participants(t,N)) en construisant la description de négociation d selon la politique de coordination R (coordinator(R)=p).

Pour cette relation $\Phi_{vue,}$ une dépendance fonctionnelle réciproque existe entre l'ensemble $\mathcal{D}$ et l'ensemble $\mathcal{P} \times \mathcal{N} \times \mathcal{R}$, en d'autres termes, la connaissance de l'élément de $\mathcal{D}$ détermine un seul élément dans $\mathcal{P} \times \mathcal{N} \times \mathcal{R}$ et vice-versa.

DEFINITION **Vue** : La fonction *vue* retourne le participant, la négociation ainsi que la politique de coordination concernés par une description de négociation :

$$vue: \mathcal{D} \rightarrow \mathcal{P} \times \mathcal{N} \times \mathcal{R}$$

Ainsi, nous pouvons constater une relation entre $\mathcal{P}$ et $\mathcal{D}$ selon laquelle une description négocie pour un seul participant, mais un participant peut être impliqué dans plusieurs négociations et donc plusieurs descriptions négocient pour lui :

$$descriptions : \mathcal{T} \times \mathcal{P} \rightarrow P(\mathcal{D}).$$

Finalement, comme une conséquence au fait qu'une description est attachée à un seul participant et à une seule négociation et aussi selon notre modèle où le rôle du participant p dans la négociation N est constant en temps nous pouvons déterminer le rôle d'un participant dans une description de négociation. Ainsi, nous définissons la fonction *role_d* : $\mathcal{T}$ *x* $\mathcal{D} \rightarrow$ *Role* pour retourner le rôle du participant *p* dans la négociation à qui la description appartient:

$$\forall d \in \mathcal{D}, \exists p \in \mathcal{P}, \exists N \in \mathcal{N}, \exists R \in \mathcal{R}: vue(d)=(p,N,R) \Rightarrow$$
$$\forall t \in \mathcal{T}, \; role\_d(t,d)=role(t,p,N)$$

5.4.2. Aperçu de négociation

Comme nous venons de voir la négociation est *un processus évolutif au cours du temps* consistant en une succession de nouvelles propositions mais aussi en acceptations ou en refus de ces propositions. Notre négociation se déroule en plusieurs phases et implique plusieurs participants, qui comme nous l'avons vu développent chacun une ou plusieurs descriptions. Afin de représenter les liens entre descriptions manipulées par les participants, phases de négociation, propositions échangées et état d'une négociation nous définissons les ensembles suivants : *Statut*, *Issues* et *Aperçu* et les fonctions permettant d'y accéder - *aperçu()*, *statut()* et *issues()*.

Statut : Le *Statut* est l'état possible d'une négociation. Cet état prend une des valeurs suivantes (*Statut* $\in$*{initiated, undefined, success, failure}*):

- *initiated* – la négociation décrite dans une description vient d'être initiée;
- *undefined* – le processus de négociation pour la description considérée est en déroulement;
- *success* – le processus de négociation modélisé par la description considérée vient de fixer un accord;
- *failure* – le processus de négociation modélisé par la description considérée vient de fixer un refus;

Issues : *Issues* est l'ensemble des attributs avec leurs valeurs associées décrivant les propositions faites dans une phase de négociation.

Dans ce système de négociation nous avons supposé l'existence d'une ontologie commune, notée *O.* Les définitions des attributs associent chacun des noms des attributs à leur domaine maximal de valeurs et à une grammaire pour l'utilisation de ces valeurs (ex : (size, (self.size≥10) (self.size≤10K) and) ; (quality, self.quality enum {high low}) ; …). Une description plus détaillée de l'utilisation de cette ontologie est faite en section 8.1.

Etant donné *att* un attribut de l'ontologie *O* et *lim(att)* l'ensemble maximal des valeurs possibles pour l'attribut *att*, nous définissons l'ensemble *Issues* de la manière suivante :

$$Issues = \{(att, val) \mid att \in O \wedge val \subseteq lim(att)\}.$$

Nous définissons l'ensemble *Aperçu* comme l'ensemble des combinaisons entre un état de négociation (*Statut*) et les informations négociées (*Issues*) :

$$Aperçu \subset Statut \times P(Issues).$$

Etant données ces trois ensembles nous définissons par la suite les fonctions utilisées afin d'accéder aux informations attachées à une phase de négociation.

Soit le produit cartésien $\mathcal{T} \times \mathcal{D} \times \mathcal{Ph} \times \mathcal{D}$, nous définissons l'ensemble $E_{\Phi 4}$ désignant le graphe de la relation Φ_4 telle que :

$\forall(t,d_i,ph,d_j) \in \mathcal{T} \times \mathcal{D} \times \mathcal{Ph} \times \mathcal{D}$, $\Phi_4(t,d_i,ph,d_j)$ *ssi* $d_j \in link(t,d_i,ph)$

DEFINITION **Aperçu de négociation** : L'aperçu de négociation d'une description d_i en lien avec une description d_j au sein de la phase de négociation ph est le couple *statut* et *issues* qui sont partagées à un instant *t* par ces deux descriptions de négociation dans la phase considérée :

$$aperçu : E_{\Phi 4} \rightarrow Aperçu.$$

Ex : $aperçu(t,d_0,2,d_2) = (undefined, \{(size,10),(cost, \leq 20)\})$ aperçu de négociation géré par la description d_0 pour la description d_2 qui est liée à la description d_0 à l'instant t selon la phase de négociation 2 ($d_2 \in link(t,d_0,2)$).

Les fonctions *statut* et *issues* retournent respectivement le statut d'un aperçu de négociation et l'ensemble des issues attachées à un aperçu de négociation.

- la fonction *statut* définie sur $E_{\Phi 4}$ vers l'ensemble *Statut*:

  $statut : E_{\Phi 4} \rightarrow Statut.$

Ex : $statut(t,d_0,2,d_2)$ retourne le statut de l'aperçu de négociation des description d_0 et d_2 à l'instant *t* dans la phase de négociation *2*.

- la fonction *issues* définie sur $E_{\Phi 4}$ vers l'ensemble *Issues* :

  $issues : E_{\Phi 4} \rightarrow Issues.$

Ex : $issues(t,d_0,2,d_2)$ retourne l'ensemble des attributs négociés de l'aperçu de négociation des descriptions d_0 et d_2 à l'instant t dans la phase de négociation *2*.

Chaque nœud *a* d'une description de négociation *d* contient non seulement *l'aperçu local* de la description courante – $aperçu(t,d,a.ph,d)$ –, mais aussi une *représentation externe*

constituée des *aperçus* partagés avec les autres descriptions liées à *d* pour la phase de négociation du nœud *a* – ex : *aperçu(t,d,a.ph,d_i)*.

L'*aperçu local* et les *aperçus externes* représentés dans un même nœud de négociation, ne sont pas redondantes. Notamment, les valeurs du *Statut* peuvent ne pas être les mêmes. Cette différence est normale dans la mesure où chaque participant est autonome au niveau de ses décisions et il gère de manière différente l'évolution de la négociation dans chacune de ses descriptions.

Les différences peuvent exister aussi au niveau des *Issues*. Dans la mesure où la phase d'un nœud peut être partagée entre plus de deux descriptions, il n'est pas nécessaire que toutes les informations contenues dans le nœud soient partagées de manière uniforme. Les différences peuvent être au niveau des attributs négociés ou/et de leurs valeurs.

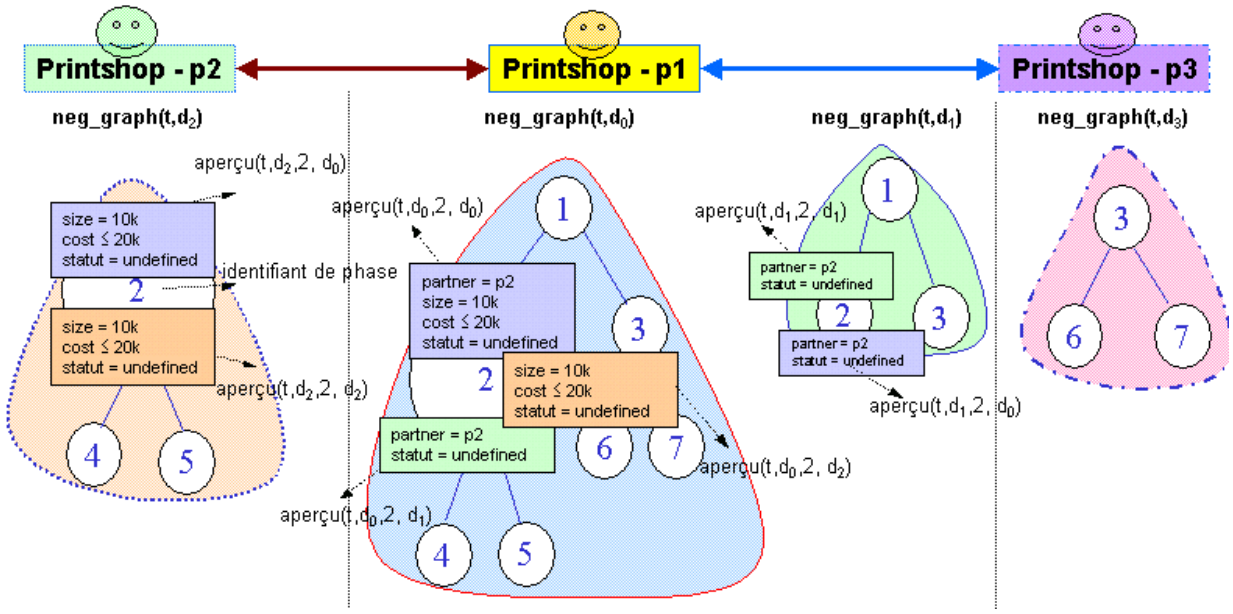

Figure 9. Descriptions de la négociation N_1 à un instant de temps t

d_0 description de la négociation N_1 vue par le partenaire p_1 selon la politique R_0

d_1 description de la négociation N_1 vue par le partenaire p_1 selon la politique R_1

d_2 description de la négociation N_1 vue par le partenaire p_2 selon la politique R_2

d_3 description de la négociation N_1 vue par le partenaire p_3 selon la politique R_3

Afin d'illustrer ceci, considérons la phase 2 de notre exemple à l'instant *t* (voir Fig.9):

- la description d0 du partenaire p1 est liée aux descriptions d1 et d2 à l'instant *t*, *link(t,d_0,2) = {d_1,d_2}*. Les aperçus du nœud associé à la phase de négociation *2* sont: l'aperçu local *aperçu(t,d_0,2,d_0) = (undefined, {(partner, p_2), (size,10),(cost, ≤20)})* et des aperçus de négociation en commun avec *d_1*

aperçu(t,d_0,2,d_1) = (undefined,{(partner, p_2)}) et en commun avec d_2 *aperçu(t,d_0,2,d_2) = (undefined,{(size, 10), (cost, ≤20)})}*.

Le participant p1 cherchant à proposer une tâche de taille 10, construit des sous-propositions séparées pour les partenaires p2 et p3. La décision de négocier avec le partenaire p2 a été prise après le déroulement d'une partie du processus de négociation, visible seulement dans les descriptions d_0 et d_1. Ce type de décision sera facilement visible en présentant les descriptions aux moments successifs de temps permettant ainsi de constater quelles descriptions évoluent ou non. Ceci peut ce voir cependant au travers de la numérotation incrémentale de l'identificateur des nœuds en fonction de leur création au fil du temps Remarquons que le graphe de négociation attaché à la description d_2 (voir Fig. 9) débute avec la phase de négociation 2. Ceci résulte de l'invitation du participant p2 dans la négociation N1 en cours de route. L'aperçu du participant p2 dans cette phase est (*size =10, cost≤20)* du fait de la proposition envoyée. De cette manière l'attribut *partner* est partagé entre d_0 et d_1 et les attributs *size* et *cost* sont partagés seulement entre d_0 et d_2.

- la description d_1 du partenaire p1 n'est liée à l'instant *t* qu'à la description d_0 (*link(t,d_1,2) = {d_0})*. Les aperçus de la phase *2* sont *aperçu(t,d_1,2,d_0)=(undefined, {(partner, p_2)}* et de l'aperçu local *aperçu(t,d_1,2,d_1)=(undefined, {(partner, p_2)}*. Les aperçus montrent le fait que la description d_1 est en interaction uniquement avec la description d_0 et que leur négociation a été sur l'attribut *partner*.

- la description d_2 du partenaire p2 n'est liée à l'instant *t* qu'à la description d_0 (*link(t,d_2,2) = {d_0})*. Les aperçus de la phase *2* au sein du graphe correspondant sont de *aperçu(t,d_2,2,d_0) =(undefined, { (size, 10), (cost, ≤20)})* et de l'aperçu local *aperçu(t,d_2,2,d_2)=(undefined, { (size,10), (cost, ≤20)}).* Pour le participant p2 dans la description d_2 le nœud correspond à une proposition faite par son partenaire de négociation p1. Notons que le partenaire p2 n'a aucune information sur l'existence de la description d1 ou de la description d_3.

Egalement, le partenaire p2 n'a aucune information sur le fait que p1 a décidé de négocier avec lui après une interaction entre les descriptions d_0 et d_1.

Finalement, pour la phase 2 le partenaire p3, selon sa description d_3, n'a aucune information sur l'existence de la phase de négociation 2 et aussi sur l'existence de la description d_1 ou de la description d_2.

5.4.3. Propriétés du Graphe de Négociation

La structuration d'une négociation en un graphe ne souligne pas uniquement la succession des phases, mais elle exprime aussi *l'affinement* ou *extension* des propositions réalisées.

Affinement :

L'affinement concerne les valeurs des attributs présents dans les deux propositions faites dans deux nœuds successifs d'un graphe.

Donc, un attribut x_c dans une phase de négociation $a_c.ph$ est l'affinement de la proposition de la phase de négociation $a_s.ph$ si pour les attributs présents simultanément dans les deux propositions, les valeurs données par la proposition $a_c.ph$ appartiennent aux valeurs données par la proposition $a_s.ph$:

$is\_refinement(t,d_i, a_s.ph, a_c.ph, x_c) \Leftrightarrow$

$\exists\, d_j \in link(\, t, d_i\, , a_c.ph\,) \cup \{\, d_i\,\} \wedge x_c \in issues(t, d_i\, , a_c.ph, d_j)\ :$

$\exists\, x_s \in issues(t, d_i\, , a_s.ph,d_j)\colon x_c.attribut = x_s.attribut\ \wedge x_c.value \subseteq x_s.value$

Dans la figure 4 nous avons fait apparaître dans le nœud 2 une proposition (*size=10k, cost≤20k*) et dans le nœud suivant 4, la proposition (*size=10k,cost<18k*) qui est un affinement au niveau de l'attribut *cost* de la proposition du nœud 2.

Extension :

L'extension concerne les ensembles des attributs négociés dans deux propositions faites au niveau des deux nœuds successifs d'un graphe.

Donc, une proposition dans une phase de négociation $a_c.ph$ est une extension de la proposition de la phase de négociation $a_s.ph$ si parmi les attributs de la proposition $a_c.ph$ il y en a au moins un qui n'existe pas parmi les attributs de la proposition $a_s.ph$:

$is\_extension(t,d_i, a_s.ph, a_c.ph, x_c) \Leftrightarrow$

$\exists d_j \in link(t,d_i,a_c.ph) \cup \{ d_i \} \wedge x_c \in issues(t, d_i , a_c.ph,d_j) : (d_j \notin link(t, d_i ,a_s.ph) \vee (d_j \in link(t, d_i , a_s.ph) \wedge \forall x_s \in issues(t, d_i , a_s.ph,d_j): x_c.attribut \neq x_s.attribut)$

La proposition faite dans le nœud 4 (*size=10k, cost<18k, delay<3*) est, par rapport à la proposition faite dans le nœud 2, un affinement pour les attributs *size* et *cost*, et dans le même temps une extension de l'ancienne proposition avec l'introduction du nouvel attribut *delay*.

Etant données ces deux définitions, nous allons préciser les propriétés d'un graphe de négociation: 1)persistance du lien de négociation, 2)cohérence du processus de négociation, 3)continuité d'une description de négociation et 4)continuité du processus de négociation.

1. Persistance d'un lien de négociation

A un instant de temps *t*, une description de négociation partage ses phases de négociation avec une ou plusieurs descriptions. La propriété de persistance d'un lien de négociation énonce que si l'ensemble de descriptions partage la phase de négociation attachée au nœud source à un instant *t*, il partagera aussi la phase de négociation attachée au nœud cible :

$link(t, d_i , a_s.ph) \subseteq link(t, d_i, a_c.ph)$

Une conséquence de cette propriété est que si pour la phase attachée au nœud source on trouve un aperçu pour une description de négociation, on trouvera également un aperçu pour la même description de négociation dans la phase attachée au nœud cible:

Soit l'arc de négociation $e \in d_i(t)$ avec $e=(a_s,a_c)$ et $\exists aperçu(t,d_i,a_s.ph,d_j) \Rightarrow$

$\exists aperçu(t, d_i , a_c.ph, d_j).$

2. Cohérence du processus de négociation

La propriété de cohérence du processus de négociation est conservée par les arcs de négociation $\forall e \in E$ avec $e=(a_s,a_c)$ de la manière suivante :

$\forall x_c \in issues(t,d_i,a_c.ph,d_j) \Rightarrow$

$is\_ extension(t, d_i, a_s.ph, a_c.ph,x_c) \vee is\_ refinement(t,d_i, a_s.ph, a_c.ph,x_c)$

- pour n'importe quel élément x_c de type *(attribut value)* négocié dans le nœud cible a_c, l'attribut n'est pas présent dans le nœud source a_s (- *extension de la négociation*) ou, s'il est présent, sa valeur est plus générale que sa valeur dans le nœud a_c (- *affinement de la négociation*).

Ainsi prenant en compte la *propriété de persistance*, nous pouvons affirmer que dans le graphe d'une description de négociation, le nœud cible est une *extension* ou un *affinement* des informations contenues dans les aperçus des descriptions avec lesquels la description courante partage la phase de négociation considérée dans le nœud source.

Nous avons illustré dans l'exemple, qu'à partir du nœud 2 - $aperçu(t,d_0,2,d_0)=($ *undefined, {(partner, p2), (size, 10), (cost,* $\leq 20)\})$), la proposition faite dans le nœud 4 - $aperçu(t,d_0,2,d_0)=($ *undefined, {(partner, p2), (*size,10k), (cost,<18k), (delay,<3)}) est par rapport à la proposition faite dans le nœud 2, un affinement (les valeurs des attributs *size* et *cost* vérifient les contraintes du nœud 2), et dans le même temps une extension de l'ancienne proposition (l'attribut *delay* est nouvellement introduit dans le nœud 4 et il n'est pas présent dans le nœud 2).

3. Continuité d'une description de négociation

Tenant compte de deux propriétés introduites ci-dessus, nous pouvons définir la propriété de continuité cohérente (notée avec ~>) d'une description de négociation de la manière suivante :

$\forall t_1, t_2 \in \mathcal{T}$ *avec* $t_1 < t_2, \forall d \in \mathcal{D}$ $d(t1) \sim> d(t2) \Leftrightarrow$

i) A tout moment des nouvelles phases de négociation peuvent être introduites dans une description de négociation. Les nouvelles phases respectent le principe de cohérence (cf. 2)

$\exists a' \in d(t_2) \wedge \exists e'=(b, a') \in d(t_2) : b \in d(t_1) \wedge \forall a \in d(t_1)\ a.ph \neq a'.ph$

ii) Les phases présentes dans une description à un instant du temps et les arcs leurs nœuds associés seront aussi présentes dans tous les instants futurs.

$\forall e=(a_1, a_2) \in d(t_1) \Rightarrow \exists e'=(a'_1, a'_2) \in d(t_2)$: $a_1.ph = a'_1.ph \wedge a_2.ph = a'_2.ph$

iii) La continuité d'une description de négociation respecte le principe de persistance d'un lien de négociation (cf. 1)

$\exists a \in d(t_1)\ \exists a' \in d(t_2) : (\ a.ph = a'.ph\ ) \wedge (\forall d_j \in link(t_1, d, a.ph) \Rightarrow d_j \in link(t_2, d, a'.ph))$

4. Continuité du processus de négociation

Une négociation est essentiellement un processus dynamique. Ainsi entre deux moments successifs de temps une négociation peut évoluer. Cette évolution doit être cohérente à la fois du point de vue du processus de négociation (cf. 1) et des descriptions qui la composent (cf. 3).

Une négociation N est initiée à t_0 par la création d'une première description de négociation dont le graphe est composé d'un seul nœud:

$N(t_0) = \{d_0\} \wedge d_0(t_0) = G(\{a\}, \varnothing)$ où $a.ph = 1$ et $aperçu(t_0, d_0, 1, d_0) = (initiated, \varnothing)$.

Par la suite, l'image de la négociation se développe progressivement en termes de descriptions de négociation (voir règle 1. ci-dessous) et des graphes associés (cf. 2) :

$\forall t_1, t_2 \in \mathcal{T}$ *avec* $t_1 < t_2$, $\forall N \in \mathcal{N}$ $N(t1) \sim> N(t2) \Leftrightarrow$

i) $\exists d_i \in N(t_1) \wedge d_i \notin N(t_0)$

ii) $\forall d \in N(t_1) \wedge \forall t_i \in [t_1, t_2] \Rightarrow d \in N(t_i)$

iii) $d(t1) \sim> d(t2)$

Pour une négociation, la propriété de continuité du processus de négociation garantit la cohérence de l'évolution en termes de descriptions des négociations présentes dans une négociation. L'évolution de la négociation est garantie par le fait qu'à tout moment de temps de nouvelles descriptions de négociation peuvent être introduites dans une négociation (voir la règle *i*). La consistance est garantie par le fait que toutes les descriptions présentes dans une négociation à un instant de temps, se retrouvent par la suite dans tous les instants de temps suivants (voir la règle *ii*). Les différents graphes introduits dans la séquence d'une description de négociation doivent avoir aussi une évolution cohérente (règle *iii* – cf.3).

Donc, l'évolution d'une négociation est faite par l'introduction de nouvelles descriptions pour le processus de négociation ou par la création de nouveaux nœuds dans les graphes associés aux descriptions de négociation.

Le nombre de descriptions pour une même négociation dépend des caractéristiques globales sur le processus de négociation (nombre de participants, tâches négociées, stratégies ou tactiques employées - voir section 2.4.). Par contre, la construction du graphe d'une description de négociation dépend essentiellement des informations disponibles aux moments antérieurs de temps (la succession des graphes jusqu'au moment considéré) et de la politique de coordination vérifiée dans la description considérée. *Les politiques de coordination* sont la mise en œuvre *des relations des dépendances* qui peuvent être fixées entre les descriptions des négociations. Comme les politiques de coordination sont les structures de base de notre modèle de coordination, elles seront détaillées dans le chapitre introduisant le modèle. Par la suite, nous décrivons seulement quelles sont les relations de dépendance qui peuvent être fixées

entre des négociations et quelles doivent être les propriétés de ces relations afin d'être gérées par le processus de coordination en manipulant des politiques de coordination.

5.5. Dépendances entre négociations

En utilisant la décomposition d'une négociation en différentes descriptions de négociations, nous pouvons fixer des relations de dépendance entre des descriptions de négociation. Ces descriptions peuvent provenir de la même négociation ou de négociations différentes. Cette approche permet par exemple de fixer des relations qui existent entre les négociations bilatérales composant une négociation faite sur une même tâche. De la même façon, mais à un autre niveau, nous pouvons fixer des relations entre des négociations faites sur des tâches différentes. Dans les deux cas, les évolutions des négociations ainsi liées doivent maintenir la consistance des relations fixées et cette consistance doit être maintenue par un processus de coordination gérant l'évolution des graphes associés aux négociations considérées.

Par la suite, nous présentons premièrement les types de dépendances que nous voulons modéliser. Nous identifions les caractéristiques des graphes de négociation qui nous permettent de coordonner les relations fixées. Ensuite, une définition formelle des relations de dépendances et les propriétés de ces relations seront introduites. Finalement nous donnons une définition pour le dernier ensemble présent dans notre modèle d'alliance, $\mathcal{R}$ - l'ensemble des politiques de coordination.

5.5.1. Types de Dépendances

De manière générale dans la synthèse sur la coordination de négociations (voir 3.3.3.) nous avons présenté les types de dépendances gérées dans différents systèmes de négociation : dépendances entre états initiaux et finaux de négociations, dépendances entre objets de négociation et dépendances entre valeurs des attributs composant les objets de négociation. Notre objectif est d'élargir la coordination sur les étapes intermédiaires de la négociation et de prendre en compte les dimensions du processus de négociation, comme le participant, le temps et l'objet de négociation.

C'est la raison pour laquelle nous considérons trois types de dépendances entre négociations au sein de notre modèle d'alliance:

- *évolution synchronisée* : l'évolution des valeurs des attributs proposés dans une des négociations influe sur l'évolution des valeurs des attributs proposés dans les

autres négociations. Par exemple, la négociation d'un travail divisé en deux lots peut impliquer une négociation bilatérale sur chaque lot. La négociation globale ne sera satisfaite que si le travail dans sa globalité est accepté : une proposition reçue dans une des négociations bilatérales sur un des lots, influence ainsi la création de propositions dans la négociation bilatérale dédiée à l'autre lot.

- *une relation d'évolution parallèle concurrente* : les négociations peuvent évoluer de manière asynchrone au niveau de la tâche négociée, seulement le démarrage ou/et la finalisation d'une des négociations peut entraîner des contraintes sur les démarrages ou/et les finalisations des autres négociations. Par exemple, si nous essayons de négocier un travail avec différents participants dans le même temps, nous pouvons envisager une négociation bilatérale pour chaque participant. Ainsi, dans cet exemple chaque négociation peut évoluer de manière indépendante des autres, la seule contrainte fixée est qu'on cherche à avoir un seul contrat à la fin. Ce type de relation fait que si une des négociations fixe un contrat les autres doivent être arrêtées.
- *une relation d'évolution parallèle* : les négociations évoluent de manière complètement asynchrone l'une par rapport à l'autre. Nous avons défini aussi cette évolution indépendante comme une troisième relation, parce que si au début les négociations sont complètement libres, en fonction de leurs évolutions, des dépendances peuvent être fixées de manière dynamique. Par exemple, si on trouve que nous avons engagé deux négociations différentes avec un même partenaire, nous pouvons chercher un échange en fixant dynamiquement une relation d'évolution parallèle concurrente : les deux négociations doivent se finaliser avec des contrats.

En regardant les différents types de dépendances présentées ci-dessus, nous pouvons observer que les caractéristiques des négociations sur lesquelles sont fixées des relations sont : les attributs négociés, les statuts des négociations et les participants impliqués. Ainsi, en fonction des caractéristiques d'une négociation sur lesquels les contraintes sont portées, nous avons trois types de dépendances : 1) *dépendance d'issues;* 2) *dépendance de statut;* et 3) *dépendance de rôles.* Ces caractéristiques sont visibles au niveau d'une description et elles peuvent être accédées par les fonctions *issues()*, *statut()* et *role_d()*.

5.5.2. Relations de dépendance

Pour gérer une ou plusieurs dépendances d'évolution entre des négociations, nous définissons une *relation de dépendance* portant sur les descriptions qui décrivent les négociations considérées. Cette évolution est visible au niveau des valeurs des caractéristiques présentes dans un graphe de négociation: i) valeurs des attributs, ii) statut de la négociation et iii) identités et rôles des participants impliqués dans une négociation.

DEFINITION **Relation de dépendance** :Nous définissons une *relation de dépendance* notée φ, comme une expression composée de deux formules logiques *condition*, et *conclusion* reliées par une *relation d'évolution* :

$$\varphi = \textbf{<Condition> <Relation> <Conclusion>.}$$

La condition (**<Condition>**) et la conclusion (**<Conclusion>**) d'une relation de dépendance sont des expressions d'un seul terme, noté $\mathcal{L}_i$, ou d'un ensemble de termes liés par des opérateurs logiques : « et » $\wedge$, « ou « $\vee$ et « non » $\neg$. Soit $\mathcal{L}$ l'ensemble total des termes présents dans des relations de dépendance, ces termes sont pris dans l'ensemble des fonctions suivantes définies ci-dessus : *issues(), statut()* et *role_d()*.

De manière générale, dans les types des dépendances décrites ci-dessus, nous fixons des relations de dépendance entre des négociations qui devront être satisfaites au fil de l'évolution des négociations. Comme nous l'avons vu précédemment selon la description considérée, certaines issues, statuts ou rôles ne sont pas décrites dans certaines descriptions. Les caractéristiques sur lesquelles sont fixées les relations doivent être visibles dans les descriptions considérées afin de pouvoir gérer leurs évolutions futures. Etant donnés les trois types de valeurs pour un terme (i.e. *issues(), statut()* et *role_d()*), nous définissons premièrement les notions de visibilité d'un terme dans un nœud et dans un graphe de négociation afin de définir par la suite la notion de visibilité en temps d'un terme. Ces notions de visibilité nous aideront à caractériser les types de dépendances que nous voulons modéliser.

Visibilité instantanée d'un terme au sein d'un graphe de négociation

La notion de visibilité au sein d'un graphe est défini en deux pas par deux fonctions booléennes: une visibilité au niveau d'un nœud *visible_a()* et ensuit la visibilité au sein d'un graphe *visible_g()*.

- Soit A l'ensemble total des nœuds présents dans des graphes de négociation, dans ce cas nous avons :

$$visible\_a : \mathcal{L} \times A \rightarrow Boolean, \text{ où}$$

$$\exists t \in \mathcal{T}, \exists d_i \in \mathcal{D}, \exists a \in d_i(t): visible\_a(\mathcal{L}_i, a) = true \quad \Leftrightarrow$$

$$\exists d_j \in link(t, d_i, a.ph) \cup \{ d_i \} \wedge (\mathcal{L}_i = statut(t, d_i, a.ph, d_j) \vee \mathcal{L}_i = issues(t, d_i, a.ph, d_j) \vee \mathcal{L}_i = role\_d(t, d_j))$$

- à l'instant t, terme $\mathcal{L}_i$ est visible dans le nœud a du graphe de négociation de la description d_i, ssi le terme est relatif au statut ou aux issues ou au rôle attachés à une description d_j avec laquelle la description d_i partage la phase de négociation $a.ph$.

- Soit $\mathcal{G}$ l'ensemble total des graphes de négociation, dans ce cas nous avons :

$$visible\_g : \mathcal{L} \times \mathcal{G} \rightarrow Boolean, \text{ où}$$

$$\exists t \in \mathcal{T} \; \exists d \in \mathcal{D} : visible\_g(\mathcal{L}_i, d(t)) = true \Leftrightarrow$$

$$\exists a \in d(t) \wedge visible\_a(\mathcal{L}_i, a) = true.$$

- étant donnés un terme $\mathcal{L}_i$ et un graphe de négociation attaché au moment de temps t à la description d, le terme est visible dans le graphe ssi le terme est visible dans un des nœuds composant le graphe de négociation considéré.

Visibilité

Avec ces deux fonctions ci-dessus nous avons défini pour les expressions composant une relation de dépendance des propriétés au niveau d'un nœud ou d'un graphe. Comme ces deux structures ne sont pas évolutives dans le temps, les propriétés sont donc statiques. Afin de pouvoir modéliser des relations entre les négociations qui sont des processus dynamiques, nous allons définir également la propriété de visibilité prenant en compte le temps.

- Selon un ensemble des termes considérés, la visibilité en fonction du temps est:

$$visible : P(\mathcal{L}) \times \mathcal{T} \rightarrow Boolean, \text{ ou}$$

$$visible(Exp, t)=true \Leftrightarrow \forall \mathcal{L}_i \in Exp \;\; \exists d \in \mathcal{D} : visible\_g(\mathcal{L}_i, d(t)).$$

- étant donnée une expression composée des plusieurs termes, l'expression est visible à un instant du temps t ssi tous les termes $\mathcal{L}_i$ qui la composent sont visibles dans les graphes de négociation attachés aux différentes descriptions au même moment de temps t.

Types de relations

A partir de ces fonctions nous allons préciser maintenant la notion **<Relation>** qui lie la condition et la conclusion d'une relation de dépendance.

Une caractéristique principale de notre modèle de coordination et le fait que nous cherchions premièrement une cohérence locale, au niveau d'un participant qui peut avoir sur les négociations dans lesquelles il est impliqué, des descriptions (graphes) qui ne sont pas forcement complètes. Donc, prenant en compte la dynamique de la négociation et la partialité de l'information, nous avons essentiellement défini deux types de relations de coordination: relations de *dépendances-fortes* et relations de *dépendances-faibles*.

Les relations de dépendances-fortes (notées $\rightarrow\bullet$) garantissent que s'il existe un instant de temps t_1 où les conditions sont satisfaites, la conclusion de la relation sera satisfaite à l'instant de temps suivant (t_1+1).

Soit deux expressions logiques *Exp_c* et *Exp_r*, la règle *Exp_c(t1)* $\rightarrow\bullet$ *Exp_r(t2)* avec $\forall t_1,t_2 \in \mathcal{T}$: $t_2 = t_1 + 1$, garantit que si l'expression *Exp_c* est vraie et visible à l'instant de temps t_1, (*visible*(Exp_c, t1)=true), dans ce cas l'expression *Exp_r* sera vraie et visible à l'instant de temps t_2, (*visible*(Exp_r, t2) = *true* pour $t_2=t_1+1$) .

Dans l'exemple de règle suivant:

$\exists\, a \in d_i(t), \forall b \in d_j(t+1)$

$statut(t,d_i,a.ph,d_i)=success \cdot\rightarrow\bullet\ statut(t+1,d_j,b.ph,d_j)=failure$

la dépendance entre les statuts visibles dans les descriptions d_i et d_j des deux négociations différentes N_X et N_Y avec un même participant p1 (N_x, N_y, $N_x \neq N_y$, $d_i \in N_x(t)$, $d_j \in N_y(t)$, $vue(d_i).p$[10] $= vue(d_j).p = p1$) fait qu'au moment où le participant p1 trouve dans le graphe attaché à $d_i(t)$ un nœud dans lequel le statut est *success*, la relation fixée fait que dans le graphe attaché à $d_j(t+1)$ les statuts dans les nœuds seront automatiquement dans *failure*.

Ceux-ci sont traduits de manière informelle par le fait que le participant p1 cherche à coordonner les négociations N_X et N_Y afin d'arrêter la négociation N_Y s'il trouve un accord dans la négociation N_X.

---

[10] Vue(d) retourne un tuple de type (p,N,R), nous utilisons l'opérateur « . » afin d'accéder à la valeur de chaque paramètre – ex. pour vue(d1)=($p1,N_1,R_1$) la vue(d1).p=p1 retourne la valeur du paramètre p dans le tuple ($p1,N_1,R_1$).

Les relations de dépendances-faibles (notées $\rightarrow$), garantissent que s'il existe un instant de temps t_2 au moment où le résultat est satisfait, les conditions ont été satisfaites à un instant $t_1 < t_2$ et qu'elles sont restées vraies jusqu'à l'instant de temps t_2-1.

L'expression, $Exp\_c(t_1) \rightarrow Exp\_r(t_2)$ avec $\forall t_1,t_2 \in \mathcal{T}$,garantit que si l'expression *Exp_r* est vraie et visible à l'instant t_2, l'expression *Exp_c* a été vraie et visible à partir d'un instant $t_1 \leq t_2$-1 jusqu'à t_2-1, ($visible(Exp\_c,ti)=true$ pour $\forall ti \in \mathcal{T}: t1 \leq ti < t2$).

Donc, si à un moment donné seules les conditions sont satisfaites, on ne peut pas garantir que le résultat sera obtenu. Ce sont des relations dans lesquelles un participant peut apercevoir tous les termes fixés dans la condition, mais ces conditions ne sont pas suffisantes afin d'arriver au résultat de la relation. Un participant cherche à satisfaire les conditions d'une dépendance faible, parce qu'il sait qu'elles sont nécessaires, mais pas suffisantes, pour obtenir le résultat. Ces relations fixent un cadre dans lequel un participant doit gérer le processus de coordination local afin de garantir une coordination globale du système.

Dans l'exemple suivant :

N_X, N_Y : $N_X \neq N_Y$, $\exists\, t_1,t_2 \in \mathcal{T}$ $t_2 > t_1$, $d_i \in N_x(t_1)$, $dj \in N_y(t_1)$, $vue(d_i).p = p_1$, $vue(d_j).p = p_2$, $\exists a \in d_i(t_1)$, $\exists b \in d_j(t_2)$,

$$statut(t_1,d_i,a.ph,d_i)=success \cdot\!\rightarrow statut(t_2,d_j,b.ph,d_j)=success$$

La description de manière informelle de la relation est que p1 sache que pour avoir une chance d'arriver dans la négociation N_Y dans un statut de *success*, il doit premièrement arriver dans la négociation N_X dans un statut *success*. Donc, le succès dans la négociation N_Y ne dépend pas seulement du participant p1 mais aussi du participant p2.

Etant donnés ces relations de dépendances nous définissons les propriétés qu'elles doivent satisfaire afin de pouvoir être gérées par un processus de coordination.

5.5.3. Propriétés d'une relation de dépendance

Soit ϑ l'ensemble des relations de dépendance que nous pouvons définir dans le cadre d'une alliance. Cet ensemble peut se structurer selon deux dimensions correspondant aux deux propriétés : *localité* et *déterminisme*.

5.5.3.1. Localité

Soit $\varphi \in \vartheta$ une relation de dépendance, cette relation sera visible localement si tous ses termes sont visibles au sein d'une même description de négociation à des moments différents de temps:

is_local(φ) $\Leftrightarrow$

$\exists d \in \mathcal{D}, \exists t_1,t_2 \in \mathcal{T}\ t_2 > t_1 \wedge (\forall \mathcal{L}_i \in \varphi.\text{condition} :\ visible\_g(\mathcal{L}_i, d(t_1))\ ) \wedge (\forall \mathcal{L}_j \in \varphi.\text{conclusion} :\ visible\_g(\mathcal{L}_j, d(t_2))\ )$.

Inversement, nous pouvons définir q'une relation de dépendance est visible globalement si elle n'est pas visible localement dans une description mais elle est visible en prenant un ensemble fini de descriptions à deux moments différents de temps :

is_global (φ) $\Leftrightarrow$

$\exists t_1,t_2 \in \mathcal{T} t_2 > t_1\ \wedge (\forall d \in \mathcal{D}\ \ \neg is\_local(\varphi, d)) \wedge (\forall \mathcal{L}_i \in \varphi.\text{condition} :\ visible(\mathcal{L}_i, t_1)) \wedge (\forall \mathcal{L}_j \in \varphi.\text{conclusion} :\ visible(\mathcal{L}_j, t_2))$.

Soit ϑ_l l'ensemble des relations visibles localement à l'instant du temps t et ϑ_g l'ensemble des relations visibles globalement, nous avons les expressions suivantes: $\vartheta_l \cap \vartheta_g = \varnothing$ et $\vartheta_l \cup \vartheta_g = \vartheta$.

5.5.3.2. Déterminisme

Une relation de dépendance est composée d'une condition et d'une conclusion, expressions logiques de plusieurs termes. Selon la structure logique de la conclusion d'une relation nous distinguons deux types de relations de dépendances :

i) une relation de dépendance est *déterministe* si la conclusion de la relation est formée d'un seul terme ou d'une conjonction des termes $\mathcal{L}_i$ (liés par $\wedge$ - *and logique*) :

<Conclusion> ::= <Exp> | <Exp> $\wedge$ <Exp>.

ii) une relation de dépendance est *non-déterministe* si la conclusion de la relation est formée d'une disjonction de termes $\mathcal{L}_i$ liés au moins une fois par $\vee$ (*or logique)* ou des négations ($\neg$ *non logique*):

<Conclusion> ::= $\neg$<Exp> | <Exp> $\vee$ <Exp> | <Exp> | <Exp> $\wedge$ <Exp>.

La satisfaction de la relation de dépendance déterministe suppose seulement des mécanismes d'observation pour les conditions et des permissions d'actionner sur les conclusions de la règle. Par contre, l'exécution d'une relation non-déterministe suppose aussi des mécanismes de décision ayant à choisir entre plusieurs possibilités d'exécuter la conclusion. Donc la différence entre les deux types de relations est le niveau auquel elles peuvent être gérées (voir partie I). Les relations déterministes peuvent être traitées au niveau du processus de coordination et les relations non-déterministes peuvent être traitées seulement au niveau du processus de décision.

Etant données ϑ_d l'ensemble des relations déterministes et ϑ_n l'ensemble des relations non-déterministes, nous avons les expressions suivantes : $\vartheta_d \cap \vartheta_n = \varnothing$ et $\vartheta_d \cup \vartheta_n = \vartheta$.

Selon ces deux dimensions nous pouvons avoir quatre ensembles différents de relations de dépendance :

	Non-déterministe	**Déterministe**
Globale	$\vartheta_g \cap \vartheta_n$	$\vartheta_g \cap \vartheta_d$
Locale	$\vartheta_l \cap \vartheta_n$	$\vartheta_l \cap \vartheta_d$

Tableau 3. Ensembles des relations de dépendances

5.5.4. Politique de coordination

Considérant, pour le moment de manière informelle, que les règles de coordination sont l'expression des relations de dépendance au niveau du processus de coordination, nous allons introduire la notion de politique de coordination.

Considérant l'ensemble $\mathcal{R}$ des politiques de coordination, nous définissons une fonction *policy* sur l'ensemble cartésien $\mathcal{T}$ x $\mathcal{R}$ vers un ensemble $\vartheta_l \cap \vartheta_d$ des règles coordonnables (voir Tableau 4):

$$policy : \mathcal{T}x\ \mathcal{R} \rightarrow P(\vartheta_l \cap \vartheta_d).$$

DEFINITION **Politique de coordination** : Une *politique de coordination*, est l'ensemble des règles de coordination à suivre à un instant de temps donné pour qu'un ensemble de relations de dépendance attachées à une description de négociation soient satisfaites. Les relations de dépendance considérées sont ainsi des relations de dépendance déterministes et visibles localement pour la description de négociation considérée.

	Non-déterministe	**Déterministe**
Globale	$\vartheta_g \cap \vartheta_n$	$\vartheta_g \cap \vartheta_d$
Locale	$\vartheta_l \cap \vartheta_n$	$\vartheta_l \cap \vartheta_d$

Tableau 4. Ensembles des relations de dépendances

Dans notre exemple (voir Fig. 9.), nous avons introduit quatre politiques de coordination identifiées par R_0, R_1, R_2 et R_3 et qui modélisent les évolutions des graphes des négociations attachées aux descriptions d_0, d_1, d_2 et respectivement d_3.

5.4. Conclusion

Nous avons proposé un modèle d'alliance dans lequel chacun des participants interagit avec les autres uniquement à travers une ou plusieurs négociations. Le modèle de négociation que nous proposons offre la possibilité de spécifier une négociation *multi-phases* (une négociation à plusieurs fils de négociation actifs modélisés comme un graphe de négociation) et *multi-attributs* (chaque nœud d'un graphe de négociation contient une phase de négociation décrite par plusieurs attributs).

Dans le type de négociation que nous modélisons, chacun des ateliers impliqués gère de manière indépendante le processus de négociation comme une construction d'une ou plusieurs descriptions de négociation. L'*indépendance* est vue non seulement au niveau des décisions prises mais aussi au niveau des informations et des données manipulées. Chaque description contient dans ses graphes de négociation des données qui sont partagées seulement avec certaines descriptions des négociations existantes.

Les différentes descriptions de négociation font apparaître une nouvelle caractéristique du modèle de négociation : *la distribution* du processus de négociation sur *plusieurs graphes différents* construits de manière *coordonnée*. Ces descriptions construisent en collaboration leurs graphes de négociation afin de satisfaire leurs politiques de coordination. Donc, en attachant à une description une politique de coordination, la description a la possibilité *d'observer*, *de gérer* et *de vérifier* les relations de dépendance imposées pendant la construction des ces graphes de négociation.

Ces différentes caractéristiques seront prises en compte dans la description du processus de négociation faite dans les sections suivantes.

Le modèle de coordination générique que nous proposons dans les sections suivantes (voir chapitre 7) sera structuré dans des briques qui correspondent aux descriptions des négociations. Donc, la modélisation de la coordination des négociations correspond à coordonner les évolutions des processus de négociation vus à travers les constructions des graphes des descriptions des négociations impliqués dans les négociations considérées. Ainsi, avant de présenter le modèle de coordination, dans le chapitre suivant, nous allons présenter premièrement la modélisation de l'évolution d'une négociation à travers les descriptions qui la composent.

6. Modèle du processus de négociation

Découlant de notre vision de la négociation au sein d'une alliance, le modèle du processus de négociation doit pouvoir mettre en œuvre un traitement parallèle des différentes phases actives d'une négociation. Une autre contrainte pour ce modèle est de proposer des mécanismes génériques qui puissent être facilement composés et exécutés dans une démarche de coordination générique. Afin de présenter le modèle du processus de négociation que nous proposons (présenté en section 6.2.) nous nous appuyons sur le formalisme s'inspirant de la métaphore Interaction Abstract Machines (IAMs) (présenté en section 6.1.).

6.1. Interaction Abstract Machines

La métaphore des Interaction Abstract Machines (IAMs) [Andreoli'96a] est conceptuellement similaire à un système multi-agent ouvert au sein duquel se déroulent des communications entre des agents-composants évoluant de manière indépendante. Le modèle computationel des IAMs, appelé Linear Objects [Andreoli'96b], intègre les concepts de la programmation orientée objet au sein de la logique linéaire [Girard'87].

La métaphore IAMs représente une entité[11] comme un espace de calcul où les ressources utilisées sont des particules en mouvement Brownien continu pouvant entrer en collision de manière aléatoire. L'évolution du système est modélisée par des lois physiques (appelées aussi *méthodes*). Ces lois constituées de deux parties (appelées tête et corps) expriment des collisions entre particules et les changements en résultant : les particules présentes dans la tête de la méthode et qui entrent en collision disparaissent pour laisser place à de nouvelles particules décrites dans le corps de la méthode. Cette métaphore modélise deux niveaux de parallélisme : au sein d'une entité où plusieurs activités peuvent se dérouler en parallèle, au sein du système où plusieurs entités peuvent évoluer en parallèle. Si au niveau entité le choix non-déterministe des lois exclue tout type de coordination entre des activités, par

[11] Le terme d'agent est habituellement utilisé dans cette métaphore. Cependant comme le sens ne correspond pas exactement aux définitions classiques d'agent dans le domaine multi-agent [Wooldridge et Jennings'95], nous utilisons le terme *entité* pour désigner le modèle conceptuel d'agent proposé dans IAMs

contre, au niveau du système une coordination peut être mise en place avec un échange explicite d'informations entre entités en utilisant le mécanisme de communication proposé.

6.1.1. Entité, Méthode

Dans IAMs, un système est constitué de plusieurs *entités*, chaque *entité* est caractérisée par un état et correspond à un ensemble de *ressources*. L'évolution de l'état est décrite par différentes lois appelées *méthodes* portant sur les ressources:

A1@…@An <>- B1@…@Bm

Une méthode est exécutée si l'état de l'entité contient toutes les ressources de la partie gauche (appelée « tête ») et dans ce cas l'entité passe dans un nouvel état où les anciennes ressources (*A1, ...,An*) sont remplacées par les ressources (*B1, ...,Bm*) de la partie droite (appelée « corps »). Toutes les autres ressources de l'entité qui n'interviennent pas dans l'exécution de la méthode se retrouvent dans le nouvel état.

Les deux opérateurs présents dans une méthode sont :

- l'opérateur @ groupe les ressources présentes dans le même état d'une entité,
- l'opérateur <>- traduit le changement d'état d'un entité.

Une caractéristique de l'entité dans IAMs est que si deux méthodes existent et elles ont les têtes composées de deux ensembles de ressources disjointes, dans ce cas les deux méthodes peuvent s'exécuter en parallèle. Par contre, si les deux méthodes ont des ressources communes, dans ce cas une seule méthode s'exécutera et le choix entre elles est fait de manière non-déterministe.

6.1.2. Evolution d'une entité

L'évolution d'une entité dans IAMs correspond à l'exécution d'une ou de plusieurs méthodes. L'exécution d'une méthode pour une entité dans un état dépend exclusivement de l'état de l'entité et de la méthode elle-même. Aucune supposition ne peut être faite : *i)* sur la sélection d'une méthode parmi des méthodes applicables (Fig. 10) l'entité trouvée dans l'état

A peut évoluer de manière aléatoire soit dans l'état B1 soit dans l'état B2 à cause de la compétition sur la ressource *b* ; ou *ii)* sur le temps d'exécution d'une méthode[12].

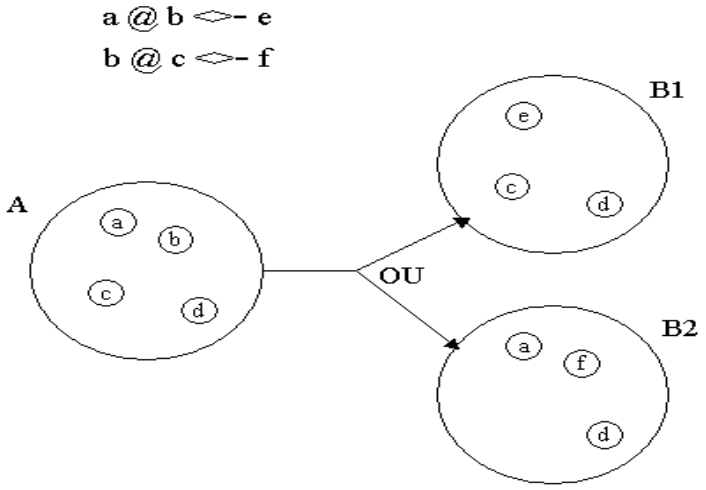

Figure 10. IAM : exemple d'évolution d'une entité

Dans IAMs les méthodes peuvent modéliser quatre types de transitions qu'une entité peut subir : transformation, duplication, destruction et communication. Les trois premières opérations (*transformation d'état, clonage* et *destruction d'état*) permettent : 1) premièrement, de modéliser l'évolution d'une entité indépendamment des autres entités existantes; et puis 2) de modéliser un système dynamique et ouvert, dans lequel des entités apparaissent et disparaissent à tout moment. Le dernier type de méthode permet la communication entre les entités d'un même système.

6.1.2.1. Transformation d'état

Avec les méthodes de type *transformation*, l'état d'une entité est simplement transformé dans un nouvel état. Si l'état de l'entité contient toutes les ressources de la tête d'une méthode de transformation, l'entité passe dans un nouvel état où les ressources de la tête sont remplacées par les ressources du corps de la méthode (Fig.11 l'entité trouvée dans l'état A évolue dans l'état B).

[12] La seule supposition faite est qu'une méthode qui peut être exécutée n'est pas ignorée à l'infini.

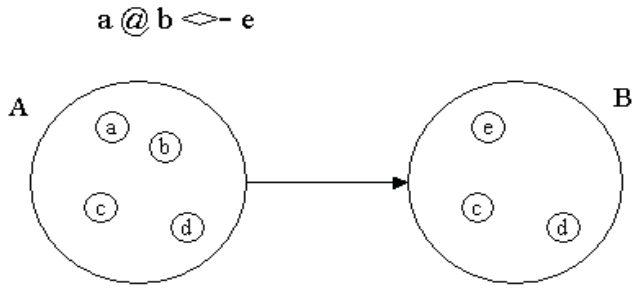

Figure 11. IAM : exemple de transformation d'état

6.1.2.2. Clonage

Avec les méthodes de type *clonage* une entité est clonée dans un nombre fini d'entités, ayant le même état. Si l'état de l'entité contient toutes les ressources de la tête d'une méthode de clonage et si le corps de la méthode contient plusieurs ensembles distincts de ressources, l'entité est clonée plusieurs fois en fonction du nombre des ensembles distincts et chaque clone obtenu subit ensuite une transformation en remplaçant la tête de la méthode par le corps correspondant (Fig.12 l'entité1 trouvée dans l'état A est clonée une fois en créant l'entité2, l'entité1 passe dans l'état B et l'entité2 passe dans l'état C).

L'opérateur *&* est utilisé dans le corps d'une méthode pour connecter plusieurs ensembles de ressources.

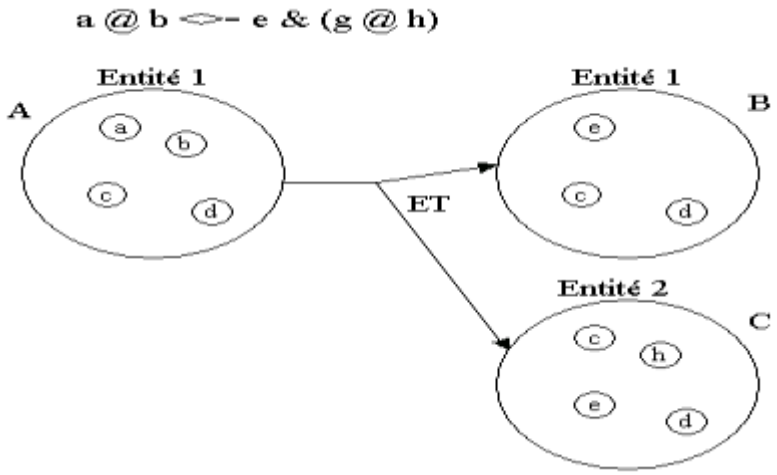

Figure 12. IAM : exemple de duplication d'état

6.1.2.3. Destruction d'état

Dans le cas de destruction d'état l'entité disparaît. Si l'état de l'entité contient toutes les ressources de la tête d'une méthode de transformation et si le corps de la méthode est la ressource T, l'entité disparaît (Fig. 13 l'entité trouvée dans l'état A évolue vers l'état final).

Le symbole T est utilisé pour indiquer un corps vide.

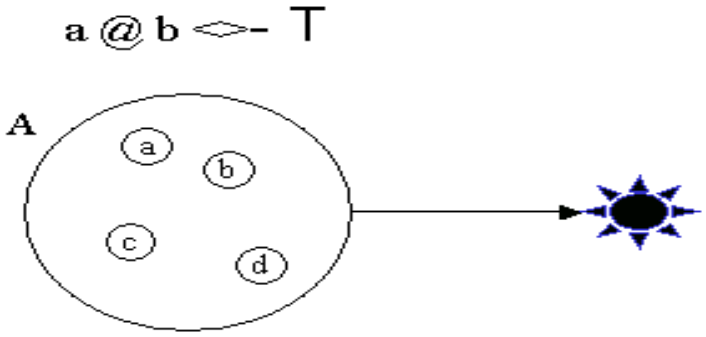

Figure 13. IAM : exemple de destruction d'état

6.1.2.4. Communication

En IAMs, la communication entre différentes entités est de type diffusion (broadcasting), représenté par le symbole « ^ ». Ce symbole est utilisé dans les têtes des méthodes et préfixe les ressources concernées par la diffusion. Ces dernières seront insérées dans l'entité courante et diffusées à toutes les entités existantes dans le système exceptée l'entité courante. Ce mécanisme de communication exécute ainsi deux opérations synchrones :

i) une transformation : si toutes les ressources non-préfixées de la tête d'une méthode entrent en collision, les ressources préfixées sont insérées dans l'entité et immédiatement consommées en appliquant la méthode (Fig.14 l'entité1 trouvée dans l'état A passe dans l'état B) ;

ii) une communication : insertion des copies des ressources préfixées dans toutes les autres entités présentes à ce moment dans le système (Fig.14 une copie de la ressource h est insérée dans l'entité 2).

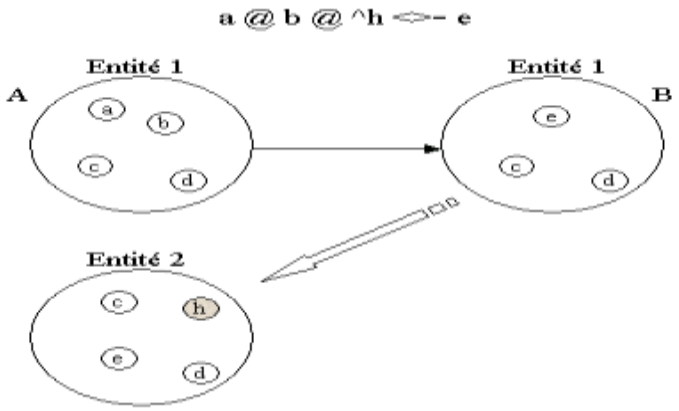

Figure 14. IAM : exemple de communication

Les opérations sans importance au niveau d'interaction entre entités comme les opérations arithmétiques ou les manipulations des strings ne sont pas analysées dans le modèle IAMs. Ces types d'opérations (appelées **« raw » computations**) sont des ressources implicites qui peuvent être récupérées par toutes les entités dans n'importe quel état.

Dans l'exemple suivant une adition simple est mise en place. Les accolades autour de la ressource *sum(A,B,C)* signifient que cette ressource peut être trouvée dans la partie implicite de l'état de l'entité :

val(A) @ inc(B) @ {sum(A,B,C)} <>- val(C)

Par la suite, nous allons utiliser la métaphore IAMs afin de modéliser le parallélisme entre plusieurs activités qui peuvent se dérouler au même moment au sein d'une négociation.

6.2. Processus de négociation

Dans le chapitre précédent (voir chapitre 5), nous avons fait apparaître au niveau des graphes attachés à une description de négociation, la possibilité de décomposer une négociation en plusieurs phases de négociation gérées dans des nœuds différents. Nous allons nous attacher dans ce qui suit à la description du processus de négociation manipulant et faisant évoluer en parallèle ces différentes phases de négociation en nous appuyant sur la métaphore IAMs (voir section 6.1.).

Dans IAMs, nous avons vu que chaque entité est caractérisée par un état correspondant à un ensemble de ressources et que l'état d'une entité peut évoluer en fonction d'un ensemble de méthodes. Nous désignerons par *Program Formulae* un ensemble de méthodes modélisant un processus de négociation et par *atome*, une entité des IAMs que nous enrichissons d'un ensemble de ressources et de propriétés spécifiques à notre modèle de négociation.

6.2.1. Program Formulae

La métaphore de l'IAMs permet dans un système multi-entité de modéliser et de gérer l'indépendance du processus d'évolution pour chaque entité composante. Nous avons vu également que chaque entité peut faire évoluer son état de manière indépendante des autres, en s'appuyant seulement sur ses ressources et les *méthodes* de son espace computationel. Cette approche nous permet des modéliser l'évolution parallèle de plusieurs phases de négociation. Utilisant la métaphore de l'IAMs, l'évolution des phases de négociation attachées aux nœuds d'une description de négociation *d*, et par transition l'évolution de la négociation *N* à laquelle d est attachée, seront ainsi gouvernées par différentes *méthodes* que nous regroupons dans ce que nous appelons une Program Formulae (PF).

DEFINITION **Program Formulae** : Une *Program Formulae* d'une description de négociation *d* - *PF(d)* est l'ensemble des méthodes qui gèrent l'évolution de la description *d*.

Une *Program Formulae* est donc la représentation d'une politique de coordination. L'ensemble des méthodes qui la constitue est obtenu à partir de la politique de coordination *R*,

de la négociation *N*, du participant *P*, concernés par la description de négociation *d* (tels que *vue(d) = (P,N,R)*).

6.2.2. Atome de négociation

Dans le modèle de négociation défini précédemment, une phase de négociation est liée à l'ensemble des aperçus sur le statut de la négociation et sur les instanciations des attributs négociés présents dans un nœud d'un graphe de négociation. Dans les approches classiques sur la négociation dans les systèmes multi-agent, le processus de négociation consiste en un échange de messages qui décrivent non seulement la proposition sur l'objet de négociation en termes d'attributs et de valeurs. A partir du message il est également possible d'accéder aux participants impliqués, au protocole et l'ontologie utilisés. Il est même possible d'obtenir les valeurs illocutoires, perlocutoires ou locutoires du message en termes d'actes de langage. Toutes ces informations sont utilisées afin de définir clairement l'évolution de la négociation dans une phase créée par un nouveau message.

De la même manière, afin de garder non seulement la description d'une *phase de négociation* en termes des aperçus vus dans le nœud courant, mais aussi de pouvoir spécifier les actions qui vont faire évoluer la négociation par changement de la phase, nous modélisons les *nœuds* d'un *graphe* de *description de négociation* comme des ensembles de particules,.

DEFINITION **Atome de négociation** : UN *atome de négociation* – noté *atome(d,ph)* – est un ensemble de ressources appelées *particules* décrivant l'état de la négociation selon la description de négociation *d* pour la phase de négociation *ph*. Les particules sont distinguées en particules de *représentation* (description de la négociation) en particules *événement*, *message*, *contrôle*, *computationnelle* (description et gestion de la dynamique de cette négociation) (cf. ci-dessous).

Ainsi, un *atome de négociation* est l'ensemble des différents aperçus caractérisant une phase de négociation (*instanciations des attributs négociées, statut de la négociation, participants impliqués*) et également la partie dynamique d'une négociation via la modélisation des différentes actions ou messages impliqués dans l'évolution d'une phase de négociation (*nouveaux messages, nouveaux participants* ou *nouvelles descriptions invitées dans la négociation courante*). Nous avons ainsi défini cinq types de particules qui seront détaillées par

la suite : particules de *représentation*, particules *événement*, particules *message*, particules de *contrôle* et particules *computationelles*.

6.2.2.1. Particules de représentation

Dans le modèle de négociation présenté dans le chapitre précédent, une description de négociation garde dans les nœuds de ses graphes des ensembles d'aperçus, images qu'un participant a sur le statut de la négociation et sur les attributs négociés dans la description courante ainsi que pour les autres descriptions avec lesquelles existe un partage d'informations. Ces informations sont modélisées au sein du processus de négociation comme des particules de représentation. Ces particules sont décrites par trois paramètres : *Name, S,* et *I*

- *Name* est défini par concaténation des identificateurs du participant et de la description considérée (ex.: p_j,d_j). La notation utilisée n'a aucune fonction formelle. Elle aide seulement à s'orienter facilement dans une expression impliquant plusieurs particules.
- *S* prend ses valeurs dans l'ensemble *Statu t= { initiated, undefined, success, failure}*. Cette valeur correspond à la valeur retournée par la fonction *statut()*.
- *I* prend ses valeurs dans l'ensemble *Issues* des attributs négociés avec leurs valeurs associées (par exemple, *I={size =1k, cost =9.5k, delay =5}*). Cette valeur correspond à la valeur retournée par la fonction *issues()*.

U*ne particule de représentation* constitue l'image qu'un participant *p* a dans une de ses descriptions *d* ($d \in description(t,p)$) sur une phase de négociation attachée à un nœud *a* ($\exists a |\ a \in d(t)$ et $a.ph = ph$). Dans notre modèle un nœud contient un ensemble d'aperçues (a.aperçu = $\{aperçu(t,d,ph,d_i) \mid d_i \in link(t,d,ph) \cup \{d\}\}$). Un atome correspondant à la description complète d'un nœud, il contiendra une particule de représentation pour chaque élément de l'ensemble des aperçues attachés au nœud considéré.

Ainsi, une <u>*particule de représentation*</u> d'un atome attaché à une description *d* pour une phase *ph* est un *aperçu* de la description *d* pour la phase *ph*.

Afin d'avoir une présentation plus structurée des descriptions de négociation impliquées dans une phase de négociation, nous définissons les particules suivantes :

- *localr(Name, S, I)* : particule de représentation locale. Elle garde l'aperçu local *aperçu(t,d_i,ph,d_i)* de la description d_i pour laquelle la structure du graphe est créée. Soit *localr(p_id$_i$, S, I)* une particule de représentation locale dans un atome correspondant à un nœud $a \in d_i(t)$. Dans ce cas les paramètres S et I ont les valeurs suivantes : S = *statut(t, d_i, a.ph, d_i)* et I = *issues(t, d_i, a.ph, d_i)*.
- *extr(Name, S, I)* : particule de représentations externe. Elle garde l'aperçu externe décrivant la manière dont une autre description perçoit la même phase de la négociation. Soit *extr(p_id$_i$, S, I)* une particule de représentation externe dans un atome correspondant à un nœud $a \in d_i(t)$. Dans ce cas les paramètres S et I ont les valeurs suivantes : S = *statut(t, d_i, a.ph, d_j)* et I = *issues(t, d_i, a.ph, d_j)* où $d_j \in$*link(t, d_i, a.ph)*.
- *firstr(Name, S, I)*- : particule de représentation externe. Elle garde l'aperçu externe attaché à la description à partir de laquelle la description courante a été créée. Soit *extr(p_id$_i$, S, I)* une particule de représentation externe dans un atome correspondant à un nœud $\exists a \in d_i(t)$. Dans ce cas les paramètres S et I ont les valeurs suivantes : S = *statut(t, d_i, a.ph, d_j)* et I = *issues(t, d_i, a.ph, d_j)* où la description $d_j \in$*link(t, d_i, a.ph)* et $\exists t_0 \in \mathcal{T}$, avec $t_0 < t$, $\forall a \in d_i(t_0)$: *link(t_0,d_i, a.ph)*$=\{ d_j\}$.

De cette manière une *phase de négociation ph* aperçue par un participant p_i dans un nœud *a* d'une de ses descriptions d_i est décrite par un ensemble de particules de représentation présentées dans un même atome.

6.2.2.2. Particules événement

Afin de modéliser les actions d'envoi de messages et de traitement des messages spécifiques à un processus de négociation (i.e. créations et traitement des propositions, acceptation ou refus d'une proposition) au sein d'un atome, nous allons utiliser les types de transitions proposés au sein des IAMs (transformations d'état, clonage, destruction d'état et communication) et définissons deux types de particules : *particule événement* et *particule message*. Les particules *événement* spécifient quels sont les types de transitions IAMs utilisés en fonction des types des messages envoyés dans une négociation.

Une particule *événement* est décrite par trois paramètres :

- *Id* identifie l'atome qui sera cloné,
- *New_id* identifie le nouvel atome créé,
- *Msg* contient le message de négociation avec les données qui vont faire évoluer la négociation dans le nouvel atome créé.

Afin d'identifier plus facilement non seulement l'opération de clonage mais aussi la direction dans laquelle le nouvel atome de négociation évoluera, nous avons défini quatre particules événement auxquelles nous avons associé différentes méthodes: *clone_propose, clone_accept, clone_reject* et *clone_create*.

- *clone_propose(Id, New_ id, Msg):* la particule modélise un événement signalant l'existence d'un nouveau message de négociation de type *propose*. Les méthodes dans lesquelles cette particule apparaît dupliquent l'atome courant et dans le nouvel atome de négociation créé, l'évolution de la nouvelle phase de négociation sera au niveau des attributs négociés.
- *clone_accept(Id, New_ id, Msg):* la particule modélise un événement signalant l'existence d'un nouveau message de négociation de type *accept*. Les méthodes dans lesquelles cette particule participe dupliquent l'atome courant et dans le nouvel atome de négociation créé l'évolution de la nouvelle phase de négociation sera au niveau du changement du statut, d'un statut *initiated* au *undefined* dans un statut de type *accept.*
- *clone_reject(Id, New_ id, Msg):* la particule modélise un événement signalant l'existence d'un nouveau message de négociation de type *reject*. Les méthodes dans lesquelles cette particule participe dupliquent l'atome courant et dans le nouveau atome de négociation créé l'évolution de la nouvelle phase de négociation sera au niveau du changement du statut, d'un statut *initiated* ou *undefined* dans un statut de type *failure.*
- *clone_create(Id, New_ id, Msg):* la particule modélise un événement signalant l'existence d'un nouveau message de négociation qui annonce la création d'une nouvelle description pour la négociation courante. Les méthodes dans lesquelles cette particule participe dupliquent l'atome courant et dans le nouveau atome de négociation créé l'évolution de la nouvelle phase de négociation sera au niveau du nombre des descriptions qui la partagent.

6.2.2.3. Particules message

Les particules messages introduites dans un atome de négociation seront utilisées pour faire évoluer la phase de négociation locale. Les particules *messages* modélisent les messages envoyés afin de les traiter selon leurs interprétations dans un processus typique de négociation. Les différents paramètres des particules messages sont :

- *Rname* et *New_r_name* sont des identificateurs pour la description qui génère le message et respectivement pour une nouvelle description invitée dans la négociation. Dans l'implémentation, afin de faire ressortir qui est le participant gérant la description en question, nous avons noté les identificateurs par la concaténation des deux identificateurs, i.e. p_i,d_j du participant p_i qui est impliqué dans la négociation courante à travers la description d_j.
- *Content* représente le contenu du message, qui est une proposition sur la tâche négociée.
- *Type* représente un identificateur de la nouvelle politique de coordination satisfaisant un certain pattern (voir section 7.2.) et que la description invitée doit gérer.

Les quatre types de particules message sont les suivantes:

- *propose(Rname, Content)*: existence d'une nouvelle proposition dans le processus de négociation.
- *accept(Rname)*: existence d'une acceptation sur une proposition. Cette acceptation a été envoyée par un participant impliqué dans la négociation à travers la description *Rname*.
- *reject(Rname)*: existence d'un refus d'une proposition. Ce refus a été envoyé par un participant impliqué dans la négociation à travers la description *Rname*.
- *create(New_r_name, Type)*: existence d'une nouvelle description qui partage la phase courante de négociation et qui est identifiée par *New_r_name*.

6.2.2.4. Particules de contrôle

En se basant sur le formalisme IAM, l'exécution des différentes méthodes dans un processus de négociation est faite de manière non-déterministe. Il est donc impossible de fixer

un ordre explicite sur les méthodes qui peuvent s'exécuter à un moment donné. Afin de mettre en place une exécution cohérente d'un processus de négociation, nous avons introduit des *particules de contrôle*. Ces particules de contrôle ont différentes fonctions dans l'espace computationel d'une description de négociation :

- une fonction d'identification (Ex. : *name(Id)*). Ce type de particule identifie les atomes de négociation en instanciant le paramètre Id avec une valeur unique pour chaque atome correspondant.
- une fonction de limitation (Ex.: *start(), enable(), freeze(), waiting()).* Ces particules seront utilisées pour introduire la notion de contrôle sur les méthodes qui peuvent ou ne peuvent pas être exécutées.
- une fonction de notification (Ex.: *stop(Accord), ready(Accord), jobToSwap(Name)).*

En s'appuyant sur les mécanismes de communication dans IAMs, ces particules seront utilisées pour introduire la notion de coordination entres les évolutions de plusieurs atomes de négociation. L'ensemble des particules de contrôle présenté n'est pas exhaustif. En fonction du processus de négociation modélisé ainsi que des contraintes de coordination à gérer, des nouvelles particules peuvent être définies. Par la suite, selon les différentes contraintes de coordination envisagées nous introduisons au fur et à mesure d'autres particules de contrôle.

6.2.2.5. Particules computationelles

Les évolutions d'une négociation sont visibles notamment aux niveaux des valeurs attachés aux attributs négociés. Les changements de valeurs se font en prenant en compte les propositions reçues ou envoyées au cours d'une négociation. Pour la manipulation des propositions de négociation nous utilisons la notion de *« raw » computation* introduite au sein des IAMs. Nous supposons que chaque atome contient implicitement dans son état des particules qui font les traitements des différentes propositions en termes d'opérations mathématiques ou de manipulations de chaînes de caractères.

Comme exemple soit la particule *{construct(I1,Content,I)}:* étant donné l'ancienne instanciation des attributs négociés (*I1*) et les nouveaux valeurs des attributs qui font l'objet de la proposition faite dans *Content,* la particule *construct* construit la nouvelle instanciation des attributs (*I*).

6.2.3. Modélisation du processus de négociation

Selon notre approche sur la négociation, les participants impliqués dans une négociation peuvent *proposer* des offres et chaque participant peut décider de manière autonome d'arrêter une négociation soit par *accepter*, soit par *refuser* une offre reçue. Egalement, selon son rôle dans la négociation, un participant peut *inviter* de nouveaux participants dans la négociation. Dans ce qui suit, nous modélisons ce type de négociation en utilisant les particules définies ci-dessus et en définissant les méthodes gérant les évolutions de ces particules.

Comme nous avons vu, une caractéristique de la négociation est son image multi-nœuds permettant une évolution en parallèle de plusieurs phases de négociation. Une possibilité de poursuivre une négociation consiste à créer une nouvelle phase de négociation à partir d'une phase existante. La figure suivante (voir Fig. 15) présente les évolutions possibles d'une phase de négociation ph0 décrite par l'atome *atome(d,ph0)*. Selon les aspects de la négociation sur lesquels les changements surviennent les quatre nouvelles phases possibles sont :

i) évolution des attributs négociés ou/et de leurs instanciations (d'*atome(d,ph0)* à *atome(d,ph1)*) : un participant envoie une nouvelle proposition, *affinement* des attributs en cours de négociation, ou *extension*, introduisant de nouveaux attributs à négocier ;

ii) évolution du statut de la négociation aperçu par une des descriptions partageant la nouvelle phase de négociation : un des participants accepte - *atome(d,ph2)* - ou refuse une proposition - *atome(d,ph3)*);

iii) évolution des participants, des dépendances entre négociations via l'évolution du nombre des descriptions qui partagent la même phase de négociation : une description peut inviter une nouvelle description avec laquelle la nouvelle phase de négociation sera partagée *atome(d,ph4)*.

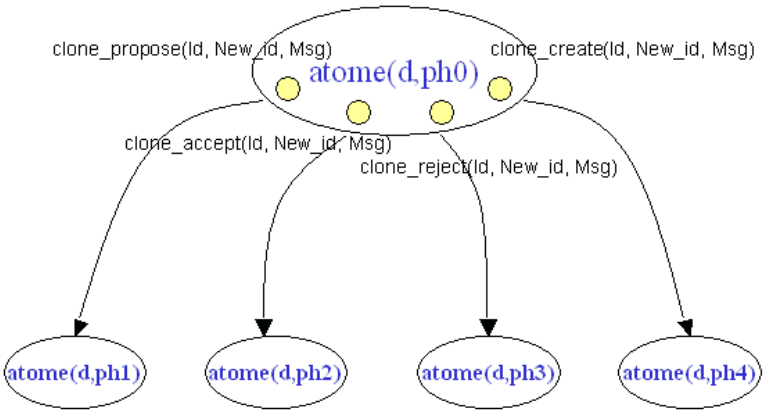

Figure 15. Evolution d'une négociation par clonage d'un atome

Utilisant la métaphore IAMs les évolutions des phases de négociation correspondent à des évolutions au niveau des atomes. Selon les méthodes qui le concernent et les particules qui le composent, un *atome de négociation* peut évoluer et/ou peut faire apparaître d'autres atomes de négociation. Dans les trois cas présentés ci-dessus, l'évolution peut être vue comme un processus composé de deux étapes : opération de *clonage* de l'atome correspondant à la phase initiale, dans notre cas *atome(d,ph0)*, et opération de *transformation* au sein de l'atome cloné pour donner place à la nouvelle phase de négociation.

Opération de clonage

L'opération de clonage est exprimée par un ensemble de méthodes dans lesquelles apparaissent des particules *événement* (cf. ci-dessus). Ces méthodes sont utilisées pour faire évoluer la négociation (voir section 6.1.2.2.). Les particules *événement* apparaissent dans le système de manière asynchrone. Leur apparition peut être le résultat d'un processus de décision, d'un processus de coordination ou d'un processus de communication de l'agent impliqué dans la négociation. Leur création ne fait pas l'objet de ce chapitre. Nous allons nous concentrer ici sur les méthodes qui leur sont associées et la manière dont elles interviennent dans le processus de coordination :

- La méthode *Propose* est associée à la particule événement *clone_propose(Id, New_id, Msg)*. Elle modélise l'introduction d'une nouvelle proposition (*clone_propose*)

par un des participants à la négociation (le traitement de cette proposition sera fait par la suite – voir section 6.2.2.3). L'expression de cette méthode est:

name(Id) @ enable @ clone_propose(Id, New_ id, Msg)<>- (enable @ name(Id)) & (freeze @ name(New_ id) @ propose(Rname, Content))

- l'atome identifié par la particule *name(Id)* est dupliqué. La nouvelle proposition contenue dans la particule *propose(Rname, Content)*.sera introduite dans le nouvel atome *name(New_id)*

- La méthode *Accept* est associée à la particule événement *clone_accept(Id, New_id, Msg)*. Elle modélise le fait qu'un des participants a envoyé un message d'acceptation (*clone_accept*) d'une ancienne proposition. Son expression est:

name(Id) @ enable @ clone_accept(Id, New_ Id, Msg) <>- (enable @ name(Id)) & (freeze @ name(New_ Id) @ accept(Rname))

- l'atome identifié par la particule *name(Id)* est dupliqué. Le message d'acceptation contenu dans la particule *accept(Rname)* sera introduit dans le nouvel atome *name(New_id)* .

- La méthode *Reject* est associée à la particule événement *clone_reject(Id, New_id, Msg)*. Elle modélise le refus d'une ancienne proposition (*clone_reject*) par un des participants. Son expression est:

name(Id) @ enable @ clone_reject(Id, New_ Id, Msg) <>- (enable @ name(Id)) & (freeze @ name(New_ Id) @ reject(Rname))

- l'atome identifié par la particule *name(Id)* est dupliqué. Dans le nouvel atome *name(New_ id)* sera introduite le message de refus contenu dans la particule *reject(Rname)*.

- La méthode *Create* est associée à la particule événement *clone_create(Id, New_id, Msg)*. Elle modélise l'invitation d'une nouvelle description (*clone_create*) par un des participants pour partager la phase de négociation nouvellement créée. Son expression est:

name(Id) @ enable @ clone_create(Id, New_Id, Msg) @ <>- (enable @name(Id)) & (freeze @ name(New_Id) @ create(Rname, Type))

- l'atome identifié par la particule *name(Id)* est dupliqué, et dans le nouvel atome *name(New_ id)* une particule *create(Rname, Type)* est introduite, ce qui va générer par la suite l'apparition d'une nouvelle particule de représentation pour la nouvelle description participant à la négociation.

Ces méthodes sont décrites de manière générique. De nouvelles particules peuvent être ajoutées en fonction de la manière dont la description courante construite les graphes de négociation.

Avec ces méthodes des particules événements nous n'avons modélisé que la duplication d'un atome dans lequel sont introduites de nouvelles particules messages (voir Fig.16). Dans le nouvel atome, les particules de représentations pour la phase de négociation courante restent identiques à celles du premier atome. Dans ce qui suit, nous présentons le processus de transformation permettant de faire progresser la négociation dans l'atome récemment créé en faisant évoluer la phase de négociation (voir section 6.1.2.1.).

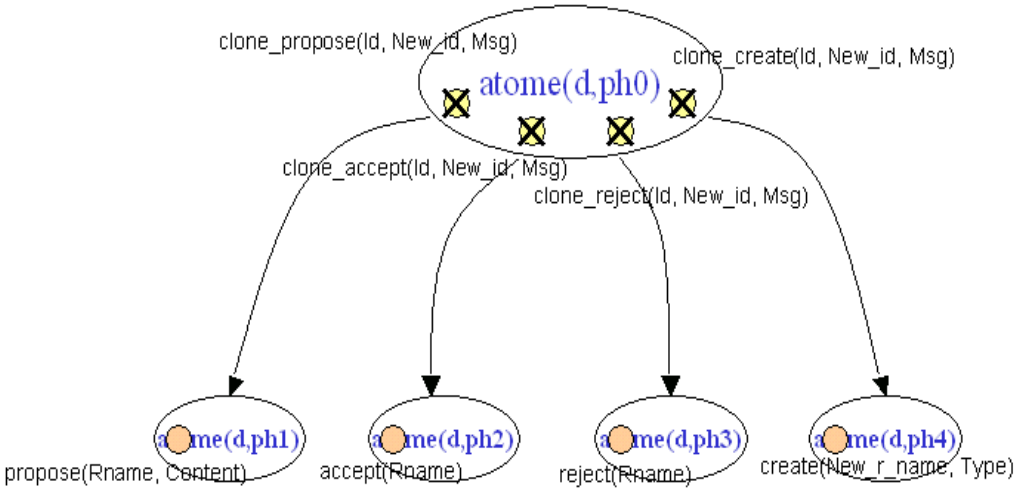

Figure 16. Evolution d'une négociation par transformation de l'état d'un atome

Opération de transformation

Nous avons vu que l'évolution du processus de négociation est faite en changeant ou en créant une nouvelle phase de négociation. Cette phase peut évoluer *i)* en terme de statuts, *ii)* d'attributs négociés et *iii)* du nombre de descriptions participantes dans la négociation :

1. La succession des statuts est la suivante: *i)* à la création d'un premier atome et d'une première phase de négociation, une description *d* se trouve dans l'état *initiated* ; *ii)* au moment d'une émission ou d'une réception d'un message, la description *d* passe dans l'état *undefined* ; *iii)* si un participant accepte ou refuse une proposition, la description associée *d* peut alors passer dans un statut *success* ou *failure.*
2. Concernant les *Issues* négociées, les différents messages font évoluer l'ensemble d'attributs ainsi que leurs valeurs.
3. L'introduction d'une nouvelle description dans la négociation courante est modélisée par l'introduction d'une nouvelle particule de représentation modélisant l'aperçu de la nouvelle description sur la phase courante de négociation.

Afin de modéliser ces évolutions au niveau d'une phase de négociation nous avons défini les particules de type message décrites auparavant. Les particules message participent à des méthodes de transformation qui changent la phase de négociation d'un atome en remplaçant les particules de représentation des descriptions impliquées dans la création ou dans la réception des messages considérés. Dans ce qui suit, nous donnons les formes de base des méthodes de transformations. En fonction des contraintes particulières de négociation d'autres méthodes de transformations peuvent être définies en ajoutant à ces formes de base d'autres particules modélisant les contraintes envisagées.

- Méthode de transformation associée à la particule *propose(Rname, Content)*. Elle fait évoluer localement une phase de négociation en termes de statut et d'attributs négociés. L'évolution est faite en remplaçant, dans l'atome correspondant, toutes les particules de représentation concernées (en fonction de la méthode) avec de nouvelles particules dont les statuts sont changés en *undefined* et où les *Issues* contiennent la nouvelle proposition exprimée dans *Content* de la particule message. La manipulation et la création des propositions est faite avec l'aide des particules computationelles (voir section 6.2.2.5).

  freeze @ localr(Rname1, S1, I1) @ extr(Rname, S2, I1) @ propose(Rname, Content)<>- enable@ localr(Rname1, undefined, I) @ extr(Rname, undefined, I)

- l'atome change d'état en consommant la particule *propose()* ainsi que deux particules de représentation pour créer des nouvelles particules de représentation afin de faire apparaître que la nouvelle proposition reçue a été prise en compte.

- Méthode de transformation associée à une particule *accept(Rname)*. Elle fait évoluer localement une phase de négociation en terme de statut. L'évolution est faite en remplaçant dans l'atome correspondant les particules de représentation concernées avec de nouvelles particules dont les statuts sont changés de *initiated* ou *undefined* en *success :*

freeze @ localr(Rname1, S1, I1) @ extr(Rname, S2, I1) @ accept(Rname) <>- localr(Rname1, success, I1) @ extr(Rname, success, I1)

- l'atome change d'état en consommant la particule *accept()* et deux particules de représentation pour créer des nouvelles particules de représentation ayant les statuts changés en *success*.

- Méthode de transformation associée à une particule *reject(Rname)*. Elle est similaire à la méthode de la particule *accept(Rname)* avec la différence que l'évolution de la phase de négociation est faite en changeant les statuts des particules de représentation concernées de *initiated* ou *undefined* en *failure* :

freeze @ localr(Rname1, S1, I1) @ extr(Rname, S2, I1) @ reject(Rname) <>- localr(Rname1, fail, I1) @ extr(Rname, fail, I1)

- l'atome change d'état en consommant la particule *accept()* et deux particules de représentation pour créer des nouvelles particules de représentation ayant les statuts changés en *success*.

- Méthode de transformation associée à la particule *create(New_r_name, Type)*. Elle fait évoluer localement une phase de négociation en terme du nombre des descriptions qui participent à cette phase de négociation. Cette évolution est faite en ajoutant, dans l'atome correspondant, une nouvelle particule de représentation :

freeze @ create(Rname, type) <>- extr(Rname, init, ∅) @ enable

Comme cette description vient d'être invitée dans la négociation, son statut est *initiated* et son ensemble d'attributs négociés est un ensemble vide.

Pour le moment, nous avons montré qu'avec les particules de représentation, les particules événement et les particules messages nous pouvons décrire les phases de négociation ainsi que les évolutions de ces phases. La métaphore des IAMs met en place une exécution non-déterministe des différentes méthodes, ne garantissant pas l'exécution séquentielle des méthodes qui modélisent le processus de négociation. Afin de palier à cet inconvénient et mettre en place une exécution cohérente du processus de négociation, nous introduisons des *particules de contrôle*. Ces particules ont différentes fonctions dans l'espace computationel d'une description de négociation :

- une fonction d'identification (Ex. : *name(Id)*). Ce type des particules identifie les atomes de négociation en instanciant le paramètre Id avec une valeur unique pour chaque atome correspondant. Cette valeur unique fait par la suite que seulement l'atome spécifié peut consommer les différents événements introduits dans système et qui lui sont destinés. Dans les exemples sur les particules événements, nous avons utilisé la particule *name* afin d'exécuter la méthode seulement si le paramètre *Id* du *name(Id)* partage la même valeur avec le premier paramètre de l'événement.

- une fonction de limitation (Ex. : *start, enable, freeze, waiting*). Ces particules introduisent la notion de contrôle sur les méthodes qui peuvent entraîner des erreurs dans la négociation. Ce type des particules limite, dans un état donné, le nombre des méthodes qui peuvent être exécutées. Donc, on peut garantir une succession dans les exécutions des certaines méthodes. Dans les exemples montrés auparavant, nous avons utilisé les particules *enable* et *freeze* pour favoriser les méthodes de consommation des événements et, respectivement de consommation des messages. Avec la succession de ces deux méthodes nous avons introduit un ordre précis dans le processus de négociation, notamment la création en premier d'un atome de négociation et en suite l'évolution de la phase de négociation dans cet atome nouvellement créé.

- une fonction de notification (Ex. *stop(Accord),ready(Accord), jobToSwap(Name)*). Les évolutions de tous les atomes de négociation et leurs phases de négociation associées sont faites en parallèle. Afin de modéliser une coordination sur l'exécution du processus de négociation aperçu dans une description, nous avons utilisé entre les espaces de négociation composants, le mécanisme de communication. Ce type

des particules, faisant l'objet de l'opération de communication entre les différents atomes de négociation, notifie tous les autres atomes d'une négociation d'un certain résultat.

Dans le processus de négociation, les messages contiennent des meta-informations sur le contenu du message décrivant les propositions en termes de valeurs pour les différents attributs de l'objet négocié. En dehors de l'aspect lié aux ontologies[13], le traitement des propositions suppose des opérations arithmétiques ou des manipulations des strings.

Pour la manipulation des propositions de négociation, en utilisant la notion de *« raw » computation* introduit dans IAMs, nous supposons que chaque atome contient implicitement dans son état des particules qui font les traitements des différentes propositions.

6.2.4. Exemple

Dans ce qui suit, nous présentons un scénario simple de négociation dont le processus de négociation correspond à un échange de propositions débouchant sur un accord.

Considérons un atelier d'impression du participant p1. Le participant p1 décide de sous-traiter un job (une impression de 10K exemplaires, à un coût inférieur à 2€/exemplaire, dans un délai inférieur à 5 jours) à un autre atelier d'impression d'un participant p2. Cette stratégie de négociation est en concordance avec la politique de coordination R_1. La négociation N qui prend place entre p1 et p2 est une négociation bilatérale. Elle est décrite par deux descriptions : $N(t) = \{d_1, d_2\}$ avec $d1 \in descriptions(t,p_1)$, $d2 \in descriptions(t,p2)$ et $role(t,p1,N)=initiator$, $role(t,p2,N)=guest$.

Le scénario que nous modélisons par la suite peut être décomposé en trois étapes distinctes :

- **étape1** : après une première proposition faite par le participant p1 à p2, le participant p2 décide d'envoyer une nouvelle proposition ;
- **étape2** : les participants décident de se mettre d'accord sur la deuxième proposition ;
- **étape3** : l'engagement est pris et la négociation s'arrête.

[13] Nous supposons que tous les participants utilisent le même langage et la même ontologie.

La modélisation est la suivante :

Etape 1

Dans la figure (Fig.17.a.) nous avons la description de négociation d_1, $vue(d_1)=(p_1,N,R_1)$ avec un atome a1 correspondant à une phase de négociation, décrite par deux particules de représentation, une locale et une externe et deux particules de contrôle (*enable* et *name(a1)*). Cet atome peut être considéré comme étant la proposition faite par p1 à p2 au début de la négociation.

Poursuivant notre scénario, nous pouvons supposer que dans l'atome *a1* de la description de négociation d_1 l'événement *clone_propose(a1, a2, cost=18K delay=3)* a été introduit pour annoncer une nouvelle proposition. En utilisant la méthode *Propose* la description de négociation d_1 contiendra deux atomes (Fig. 17. b).:

name(Id) @ enable @ clone_propose(Id, New_ id, Msg)<>- (enable @ name(Id)) & (freeze @ name(New_ id) @ propose(Rname, Content))

L'atome a1 qui a changé en consommant la particule *clone_propose* et un nouvel atome a2 qui est le clone de l'atome a1 (les particules de représentation qui ne participent pas dans la méthode sont présentées inchangées dans les deux atomes). A partir de ce moment la négociation est décrite par deux atomes de négociation dans lesquels la négociation peut évoluer de manière indépendante.

Avec les méthodes suivantes nous allons modéliser le fait que les nouvelles particules (*name, freeze, propose*) introduites dans l'atome a2 feront la phase de négociation attachée à cet atome d'avoir une évolution différente de celle de l'atome a1.

Dans l'exemple donné sur la particule *propose* nous avons utilisé la méthode suivante :

freeze @ localr(Rname1, S1, I1) @ extr(Rname2, S2, I1) @ propose(Rname2, Content)<>- enable@ localr(Rname1, undefined, I) @ extr(Rname2, undefined, I)

Cette méthode change les particules de représentations qui gardent les anciennes valeurs des attributs avec de nouvelles particules de représentation qui prennent en compte la nouvelle proposition reçue.

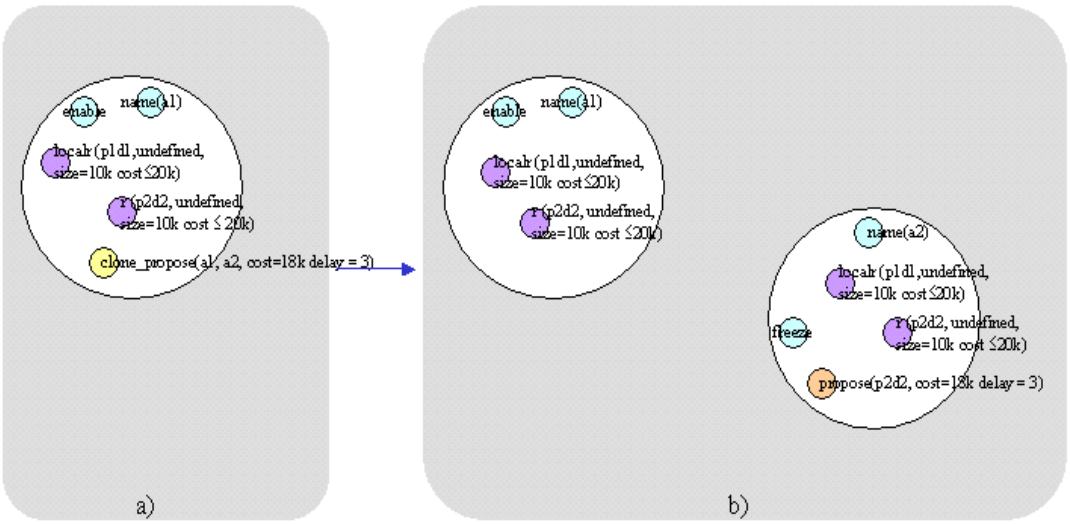

Figure 17

Mais, afin de passer d'une particule de représentation *extr(p2d2, undefined, size=10K cost≤20K)* et d'une particule message *propose(p2d2, cost=18K delay=3)* à une particule de représentation *extr(p2d2, undefined, size=10K cost=18K delay=3)* nous avons supposé l'existence d'une particule de type computationelle appelée *construct(I1,Content,I)* qui calcule cette transformation (Fig.18.c). Et la méthode utilisée est :

freeze @ localr(Rname1, S1, I1) @ extr(Rname2, S2, I1) @ propose(Rname2, Content) @ {construct(I1,Content,I)} <>- enable @ localr(Rname1, undefined, I) @ extr(Rname2, undefined, I)

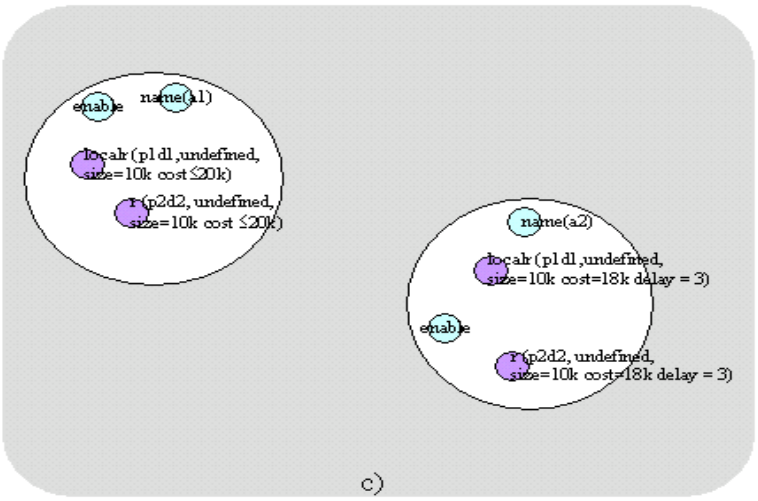

Figure 18

Etape 2

Par la suite nous supposons que le participant p1 analyse les deux propositions et qu'il décide d'accepter la deuxième proposition. L'acceptation est modélisée de la manière suivante : i) l'atome contenant la proposition à accepter est dupliqué selon la méthode *Accept* d'une particule événement de type *clone_accept()*

name(Id) @ enable @ clone_accept(Id, New_ Id, Msg) <>- (enable @ name(Id)) & (freeze @ name(New_ Id) @ accept(Rname))

ii) l'atome clone évolue en consommant la particule message *accept()*

freeze @ localr(Rname1, S1, I1) @ extr(Rname, S2, I1) @ accept(Rname) <>- localr(Rname1, success, I1) @ extr(Rname, success, I1)

Ainsi, dans la figure (Fig. 19.d) la description de négociation du participant d1 est composée de trois phases de négociation. Le troisième atome a3 contient une phase de négociation dans laquelle le statut de la négociation est *success*.

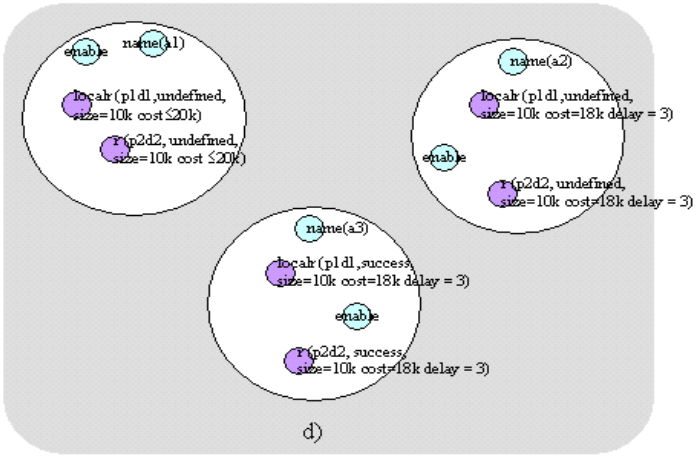

Figure 19

Etape 3

L'atome a3 peut être donc vu comme l'image d'une phase de négociation sur laquelle les deux participants sont mis d'accord. L'existence de cet accord doit aussi impliquer le fait que la négociation doive s'arrêter.

Dans la figure (Fig. 20.e) en utilisant la méthode :

localr(Rname1, success, I1) @ extr(Rname2, success, I1) @ ^stop(I1) <>- ready(I1)

l'atome a3 change d'état et dans le même temps communique aux autres de s'arrêter.

La méthode *stop <>- #t* fera qu'à un moment donné tous les atomes qui contiennent la particule *stop* seront détruits (Fig. 21.f.) et que finalement seulement l'atome avec l'accord reste actif (Fig. 21.g).

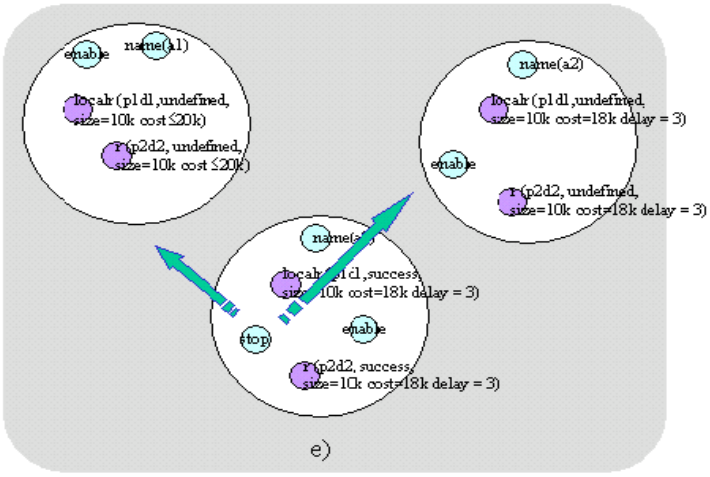

Figure 20

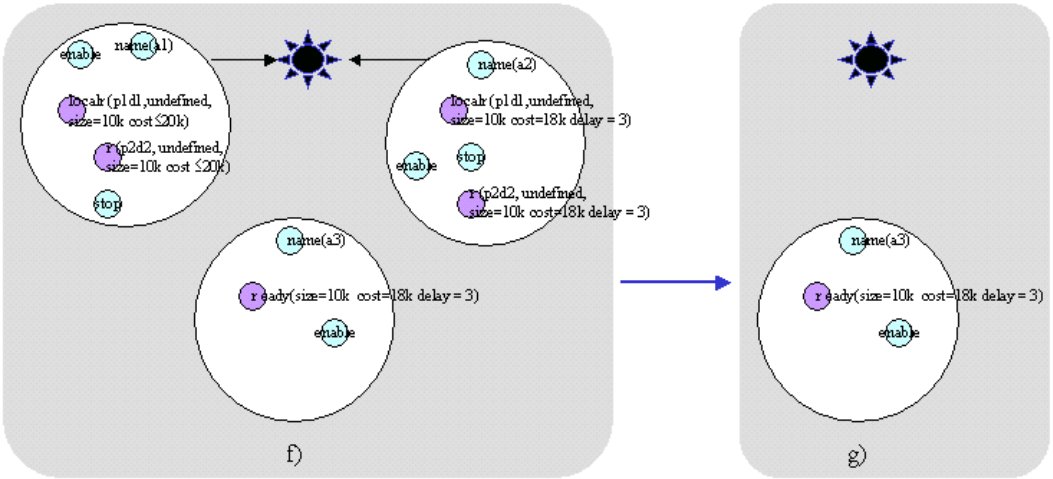

Figure 21

6.3. Synthèse

A partir de notre modélisation de la négociation du chapitre précédent, nous avons défini dans ce chapitre le processus de négociation en nous appuyant sur la métaphore des IAMs. Une négociation est modélisée par un ensemble de descriptions de négociation. Dans ce chapitre nous avons représenté chacune de celles ci par un ensemble d'atomes de négociation où chaque atome gère de manière indépendante une phase de négociation particulière ainsi que l'ensemble des aperçus associés. Un aperçu décrit le statut de la négociation et l'ensemble des attributs négociés. Une négociation est aussi caractérisée par le rôle joué par ses participants et par la politique de coordination qui lui est attachée. Dans ce chapitre nous avons défini comment ces politiques pouvaient être décrites à l'aide de différentes méthodes. Le modèle de processus de négociation proposé dans ce chapitre permet de décrire l'évolution d'une négociation comme l'évolution en parallèle des atomes de négociation associés selon les méthodes modélisation la politique de coordination.

Par la suite nous allons définir le modèle de coordination en utilisant ce modèle de description du processus de négociation en faisant une correspondance entre politique de coordination qu'une description doit satisfaire au niveau du modèle de négociation et l'ensemble de méthodes qui modélise l'évolution d'une description au niveau du modèle de processus de négociation. Comme chaque négociation est composée d'un ensemble de descriptions de négociation, le processus de coordination d'une ou de plusieurs négociations, sera divisé en briques distribuées sur chacune des descriptions participantes dans les négociations impliquées.

7. Modèle générique de coordination

Utilisant le modèle de description d'une négociation présentée antérieurement, nous pouvons définir la coordination d'une ou de plusieurs négociations, comme la gestion des relations de dépendances entre les différentes *descriptions des négociations* qui leur sont attachées. Nous décrivons dans cette section le modèle développé pour représenter le processus de coordination, en se basant sur les dépendances qui peuvent exister entre négociations (voir section 5.3.) et sur le modèle du processus de négociation (voir section 6.2.).

L'objectif est de décomposer le processus de coordination en plusieurs modules qui gèrent des groupes de dépendances. Ces modules correspondent à des *descriptions de négociation* particulières, qui construisent leurs images sur la négociation en accord avec les contraintes de coordination (les règles de coordination) et les contraintes imposées par la structure de type graphe de la négociation (voir section 5.2.4.).

Par la suite, le module de coordination utilise pour faire évoluer les différentes images des négociations, seulement les propositions qui sont en accord avec les contraintes de coordination. Dans certains cas les contraintes peuvent être très figées. C'est le cas avec des relations de dépendance déterministes et où la poursuite d'une négociation est limitée à une seule proposition possible. Dans ce cas le processus de coordination élabore automatiquement cette proposition. Si les contraintes sont assez larges pour ne pas limiter le choix à une seule proposition, c'est le cas avec des relations de dépendances non-déterministes, c'est au processus de décision d'élaborer les propositions en accord avec l'image de la négociation construite par le processus de coordination. Le module de coordination laisse la liberté au processus de décision et au processus de communication d'exécuter n'importe quelle opération tout en s'assurant qu'elle ne viole pas les contraintes de coordination.

Au niveau du processus de coordination des négociations, présenté par la suite, notre objectif n'est pas de décrire a priori toutes les propositions qu'un participant peut faire afin de réaliser une coordination entre ses négociations, mais de définir les contraintes que ces propositions doivent satisfaire. Selon les négociations impliquées et selon les relations de dépendances gérées, nous avons décomposé le modèle de coordination en deux dimensions :

i) coordination en monde fermé où les dépendances sont fixées et non modifiables et où on cherche à coordonner des relations entre des descriptions impliquées dans une même négociation. De cette manière nous pouvons coordonner une négociation de type

un-à-plusieurs ou même une négociation multi-bilatérale (voir section 2.1.2.) en gérant les dépendances entre les négociations bilatérales qui la composent.

ii) coordination en monde ouvert où les dépendances sont dynamiques et modifiables et où on cherche à coordonner des relations entre des descriptions participantes dans des négociations différentes. De cette manière nous pouvons coordonner pour un participant les dépendances existantes entre les négociations dans lesquelles il est impliqué. Egalement en gardant la structure multi-bilatérale d'une négociation, nous pouvons coordonner les dépendances entre des négociations bilatérales appartenant aux différentes négociations.

Dans ce chapitre nous commençons par détailler le langage proposé pour exprimer les *règles de coordination* (section 7.1.), afin de nous intéresser par la suite à décrire *des schémas de coordination* (section 7.2.) comme des ensembles cohérents de règles de coordination. Les deux dernières sections présentent la modélisation du processus de coordination selon deux dimensions : i) coordination en monde fermé (section 7.3.) et ii) coordination en monde ouvert (section 7.4.).

7.1. Règle de Coordination

Les différents types de relations de dépendances introduites en section 5.3., nous permettent de mettre en place des règles de coordination entre des négociations différentes. Ces règles mettent en liaisons des critères liés aux temps d'exécution des négociations, aux descriptions des tâches négociées et aux participants impliqués dans les négociations visées. On utilise ces règles de coordination comme constituants de base pour décrire des liaisons plus complexes entre des négociations.

Nous proposons *une règle de coordination* comme étant la mise en œuvre d'une relation de dépendance entre plusieurs descriptions de négociation et qui gère les évolutions en temps de ces descriptions.

Une *règle de coordination* a la structure suivante :

<Rule_Definition> ':'[<Paramer_Definition >]* ';' <Graphs_conditions> <Condition> <Relation> <Result>.

- La premier partie d'une règle de coordination (**<Rule_Definition>**) sert à définir le nom de la règle et ses paramètres – ex. : *name_rule(v1,v2, ..,vn)*. Les paramètres d'une règle de coordination peuvent être des descriptions de négociations ou des instants du temps - $\forall v_i$ paramètre d'une règle de coordination, $v_i \in \mathcal{T} \cup \mathcal{D}$. Les paramètres peuvent être des constantes (notées avec lettres minuscules) ou des variables (notées avec lettres majuscules).
- Dans **<Paramer_Definition >** chacune des variables est liée à son domaine de valeurs possibles – ex. : $synchronize(T_1,T_2,d1,d2) : T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$.
- **<Graphs_conditions>** fixe des conditions sur graphes et plus particulièrement sur les nœuds qui sont présentés dans les descriptions des négociations impliquées dans la coordination et qui seront concernés par la règle de coordination.
- La deuxième partie d'une règle de coordination est similaire à la structure d'une relation de dépendance (voir section 5.3.2.) et elle est composée d'une partie gauche appelée *condition* **<Condition>** et une partie droite appelée *conclusion* **<Result>** et d'une relation entre les deux (**<Relation>**). Dans la description de

notre model nous avons fixé deux types de relations entre la condition et la conclusion : *relation forte* «$\rightarrow\bullet$» et *relation faibles* «$\rightarrow$» (voir section 5.3.2.). La condition et la conclusion sont définies comme un seul terme ou une composition logique des plusieurs termes. Ces termes sont des fonctions décrivant les caractéristiques d'une négociation. Selon les types des possibles fonctions, le langage peut exprimer des règles de coordination fixant des contraintes sur les statuts d'une négociation, les attributs négociés ou les rôles de participants impliqués.

Par la suite nous allons présenter avec des exemples les différents types des dépendances que nous pouvons modéliser entre des descriptions de négociation.

Dépendances de statuts

Les dépendances de statut fixent des contraintes entre les statuts de deux ou plusieurs descriptions (impliquées dans la même négociation ou dans des négociations différentes).

Le fait que ces relations entre les descriptions des négociations peuvent être complètement indépendantes des tâches négociées, fait que le processus de négociation soit vu comme n'importe quelle activité qui se déroule sur un intervalle de temps. Utilisant la définition de la fonction *statut()* est son domaine des valeurs dans l'ensemble *Statut* définies dans notre modèle (voir section 5.2.3.) nous pouvons fixer l'intervalle de temps dans lequel une négociation se déroule.

Ainsi, le début de l'intervalle est délimité par la valeur *initiated* d'une fonction s*tatut()* et cette valeur correspond au démarrage de la négociation dans une description (par exemple : étant données $\exists t \in T, \exists d \in negotiation(t,N) : statut(t,d,a.ph,d)=initiated$, nous pouvons affirmer que selon la description *d* la négociation *N* a commencé au moment de temps *t*). La fin de l'intervalle, correspondant à la finalisation d'une négociation dans une de ses descriptions, est délimitée par les valeurs *success* ou *failure.* Finalement, une négociation en déroulement correspond à la valeur *undefined.*

Dans le Tableau 4 nous avons décrit des dépendances basées sur les relations d'intervalles temporels de Allen [Allen et Hayes'85]. La majorité des dépendances présentées par la suite, sont composées d'une seule règle modélisée des contraintes ponctuelles entre des négociations, mais nous avons aussi la possibilité de faire des combinaisons pour représenter

des coordinations plus complexes, comme il est présenté dans la dernière dépendance du tableau.

Tableau 4 - Règles de coordination basées sur les relations temporelles de Allen

Règle de coordination $vue(d_i) = (p_k,N_x,R_l)$ $vue(d_j) =(p_m,N_y,R_n)$	Relation de dépendance entre
success_complementary_(hard/soft) (T_1, T_2,d_i,d_j) : $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$; $\exists a \in di(T_1)\ \ \forall b \in dj(T_2)$ $statut(T_1,d_i,a.ph,d_i) = success \rightarrow\!\!\!\!\bullet\!\!\!/\!\!\!\rightarrow\ statut(T_2,d_j,b.ph,d_j) = success$	Un succès dans la description d_i de la négociation N_x entraîne un succès dans la description d_j de la négociation N_y au moment suivant (hard relation) ou dans un moment futur (soft relation).
fail_complementary_(hard/soft) (T_1, T_2, d_i,d_j) : $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$; $\exists a \in di(T_1)\ \ \forall b \in dj(T_2)$ $statut(T_1,d_i,a.ph,d_i) = failure \rightarrow\!\!\!\!\bullet\!\!\!/\!\!\!\rightarrow\ statut(T_2,d_j,b.ph,d_j) = failure$	Un insuccès dans la description d_i de la négociation N_x entraîne un insuccès dans la description d_j de la négociation N_y au moment suivant (hard relation) ou dans un moment futur (soft relation).
ongoing_start_successively (T_1, T_2, d_i,d_j) : $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$; $\exists a \in di(T_1)\ \ \forall b \in dj(T_2)$ $statut(T_1,d_i,a.ph,d_i) = undefined \rightarrow\bullet\ statut(T_2,d_j,b.ph,d_j) = initiated$	Au moment où la négociation N_x est en déroulement dans la description d_i, dans le moment immédiat suivant la négociation N_y démarre dans la description d_j.
end_ongoing_ successively (T_1, T_2, d_i,d_j) : $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$; $\exists a \in di(T_1)\ \ \forall b \in dj(T_2)$ $statut(T_1,d_i,a.ph,d_i) = failure \ \vee\ statut(T_1,d_i,a.ph,d_i) = success \rightarrow\bullet$ $statut(T_2,d_j,b.ph,d_j) = undefined$	Au moment où la négociation N_x est finie dans la description d_i, dans le moment immédiat suivant la négociation N_y est en déroulement dans la description d_j.
end_start_ successively (T_1, T_2, d_i,d_j) : $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$; $\exists a \in di(T_1)\ \ \forall b \in dj(T_2)$ $statut(T_1,d_i,a.ph,d_i) = failure \ \vee\ statut(T_1,d_i,a.ph,d_i) = success \rightarrow\bullet$ $statut(T_2,d_j,b.ph,d_j) = initiated$	Au moment où la négociation N_x est finie dans la description d_i, dans le moment immédiat suivant la négociation N_y démarre dans la description d_j. Cette règles décrive l'opérateur *meets*(d_i,d_j) proposé par Allen.
ongoing_start_successively (T_1, T_2, d_i,d_j) *end_ongoing_ successively* (T_1, T_2, d_i,d_j)	L'ensemble de ces deux règles décrit l'opérateur *overlaps* (d_i,d_j) proposé par Allen.

Dépendances d'issues

Les dépendances d'issues fixent des contraintes entre les valeurs des attributs des tâches négociées dans deux ou plusieurs descriptions. Avec ce type des dépendances nous pouvons maintenir une cohérence entre les propositions existantes à un moment donné et les propositions faites par la suite. Les dépendances entre les valeurs des attributs négociés peuvent fixer plusieurs types des contraintes :

- *contraintes relationnelles*: les valeurs des attributs négociés dans différentes descriptions de négociation doivent satisfaire certaines relations d'égalité ou d'inégalité .
- *contraintes arithmétiques*: les attributs négociés dans différentes descriptions de négociation doivent avoir des valeurs calculées à partir des valeurs des autres attributs.
- *contraintes cumulatives* : les valeurs des attributs négociés sont impliqués dans des contraintes relationnelles et arithmétiques dans le même temps (ex.: la somme des valeurs des attributs négociés par deux descriptions impliquées dans des négociations différentes ne doit pas dépasser une certaine valeur).

Dans l'exemple suivant, la règle de coordination fait que le participant p1 arrive à un accord dans la description d_i attachée à la négociation Nx ($vue(d_i) = (p1,Nx,Ri)$) si 1) il arrive à un accord dans la description d_j attachée à la négociation Ny ($vue(d_j) = (p1,Ny,R_j)$) , et 2) si dans la description d_i existe une proposition égale à la taille de la tâche contractualisée dans Ny, et 3)si cette proposition et acceptée par un autre participant à travers une description d_k attachée aussi à la négociation Nx ($vue(d_k) = (pk,Nx,Rk)$) :

success_and_issues_complementary (T_1, T_2,d_i,d_j,d_k) : $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\exists a \in di(T_1)\ \exists b \in dj(T_1)\ \exists a' \in di(T_2)\ a'.ph = a.ph$

$statut(T_1,dj,b.ph,dj)=success \wedge issues(T_1, di, a.ph, dk).size = issues(T_1, dj, b.ph, dj).size \wedge \Box\Box\Box statut(T_1, di, a.ph, dk) = success\Box \rightarrow\bullet\ statut(T_2,di,a'.ph,di)=success$

Dépendances de rôles

Les dépendances de rôles fixent des contraintes entre les rôles joués par un ou plusieurs participants. Le fait que nous ayons défini seulement deux types de rôles et qu` un participant dans une négociation peut avoir un seul rôle, fait que les relations modélisées ne sont pas très nombreuses.

Toutefois, ce type de dépendance est utilisé dans des règles de coordination qui visent à gérer une relation entre plusieurs aspects décrivant des négociations différentes. Par exemple, un participant p1, impliqué à un moment donné dans plusieurs négociations sur des tâches proposées en alliance (ayant le rôle de *guest*) a la politique locale d'accepter une seule tâche. La règle suivante fixe, entre des descriptions participantes dans des négociations différentes[14], la contrainte spécifiant le fait que si une des descriptions soit dans un statut de *success* cella entraîne toutes les phases de négociation des autres descriptions des passer dans un statut de *failure*:

success_competition(T_1,T_2,D_i,D_j) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ $D_i \in descriptions(T_1,p1)$ $D_j \in descriptions(T_1,p1)$;
$\exists a \in Di(T_1)\ \forall b \in Dj(T_1)\ \exists b' \in Dj(T_2)\ b'.ph = b.ph$

$statut(T_1,Di,a.ph,Di)=success \wedge role\_d(T_1,Di)=guest \wedge \neg state(T_1, Dj,b.ph, Dj)=success \wedge role\_d(Dj,p1)=guest \cdot\!\!\rightarrow\!\!\bullet\ statut(T_2, Dj, b'.ph, Dj)=failure$

Donc, en utilisant ce modèle des règles de coordination nous pouvons fixer des relations entre négociations sur des informations qui peuvent être dispersées entre plusieurs participants et entre plusieurs descriptions (règles globales) ou qui peuvent être toutes retrouvées dans la même description (règles locales). Jusqu'à maintenant, dans notre modèle de négociation nous avons envisagé la possibilité de gérer des règles et de vérifier qu'elles sont satisfaites au niveau d'une seule description, donc seulement des règles locales. Comme le but est d'essayer de coordonner des relations entre plusieurs dépendances, donc des règles globales, par la suite nous allons démontrer le fait que nous pouvons réécrire les règles globales par des règles locales. De cette manière nous gérons au niveau du modèle de coordination générique non

[14] Par la suite nous utilisons l'opérateur « . » pour accéder à la valeur d'un des paramètres du tuple (p,N,R) retourné par une fonction vue() – ex. si $vue(d_i)=(pi,N_j,R_k)$ nous pouvons écrie $vue(d_i).N=N_j$

seulement de règle de coordination locale (voir section 5.3.3.) mais aussi des règles de coordination globales (voir Tableau 5.).

	Non-déterministe	**Déterministe**
Globale	$\vartheta_g \cap \vartheta_n$	$\vartheta_g \cap \vartheta_d$
Locale	$\vartheta_l \cap \vartheta_n$	$\vartheta_l \cap \vartheta_d$

Tableau 5. Ensembles des règles de coordination

7.2. Schéma de coordination

A un moment donné, pour un participant impliqué dans plusieurs négociations, l'infrastructure doit manipuler de nombreuses règles de coordination, qui peuvent être très différentes. Le modèle de coordination que nous proposons, n'est pas modélisé comme un processus centralisé, géré par un module unique, mais distribué sur plusieurs modules de coordination. Donc, il ne s'agit pas de décrire toutes les règles de coordination et toutes les actions possibles, mais de délimiter des ensembles cohérents des règles de coordination gérés par un sous-processus de coordination qui peut avoir une exécution isolée.

Dans la définition du schéma de coordination nous partons des faits suivants : 1) qu` une négociation est vue comme un ensemble des processus répartis sur plusieurs descriptions; 2) que dans une description, l'évolution de la négociation est gérée par des méthodes entre les particules composant les différents atomes de négociations attachés aux phases de négociation visibles dans la description respective; et 3) que les règles de coordination entre les différentes descriptions, fixent des contraintes sur les évolutions des ces phases de négociation. Donc, *un schéma de coordination* va définir un ensemble de règles de coordination utilisé dans la mise en place du mécanisme de coordination en termes des méthodes gérées par des descriptions ayant de comportements particuliers, afin de conduire l'évolution des leurs phases de négociation dans la direction désirée.

DEFINITION **Schéma de coordination** : Un *schéma de coordination* (ou *Coordination Pattern*) est un ensemble de règles de coordination activables dans un contexte donné. Les règles de coordination sont exprimées par autant de *Program Formulae* que de descriptions impliquées dans la formulation des règles de coordination.:

PattCoor = (TriggerSet, RulesSet, ProgramFormulaeSet).

TriggerSet représente l'ensemble des conditions de déclenchement qui doivent être toutes satisfaites dans un même moment du temps afin de pouvoir déployer le pattern de coordination. Ces sont de conditions descriptives sur les descriptions des négociations qui sont impliquées dans un ensemble cohérent des règles de coordination. Dans notre modèle nous avons défini une description de négociation comme faisant partie d'une seule négociation et qui modélise le processus de négociation pour un seul participant selon un ensemble des règles de

coordination - *vue(d) = (N,p,R)* mais une négociation ou un participant sont liés aux plusieurs descriptions.

Selon cette approche, en fixant des conditions sur les négociations et sur les participants, nous pouvons fixer des conditions sur des ensembles des descriptions qui composent la négociation considérée ou qui négocient pour un certain participant. Par la suite nous pouvons fixer des conditions sur des descriptions biens identifiées et aussi sur les caractéristiques de la négociation visibles dans ces descriptions.

Ainsi TriggerSet est formé des trois types des expressions :

(<Exp(t,N,p)>) ';' (<Exp(d)>)* ' ;' (<Exp(t,d)>)**

où

- *<Exp(t,N,p)>* sont les expressions qui identifient les négociations et les participants considérés faisant l'objet d'une ou plusieurs relations de dépendances. Ces expression sont des fonctions définies dans l'espace cartésien $\mathcal{T} \times \mathcal{P} \times \mathcal{N}$ comme les fonctions *participant(), negotiations()* (voir section 5.1.) et *role()* (voir section 5.2.6.) et elles fixent l'implication et les rôles des participants dans les négociations considérées;
- *<Exp(d)>* sont les expressions qui identifient ponctuellement les descriptions qui seront impliquées par la suite dans des relations de dépendance. Ces expressions sont fonctions de type *vue()* qui fixent les liaisons entre les ensembles $\mathcal{P}$, $\mathcal{N}$, $\mathcal{R}$ et l'ensemble $\mathcal{D}$ (voir section 5.2.7.);
- *<Exp(t,d)>* sont les expressions qui fixent quelles sont les conditions sur les caractéristiques des négociations visibles dans les descriptions auparavant identifiées afin de pouvoir déclencher l'exécution des règles de coordination. Selon notre modèle les caractéristiques d'une négociation selon une description considérée sont retournées par les fonctions *statut(), issues()* et *role_d()* (voir section 5.2.3).

RulesSet représente l'ensemble des règles de coordination que le pattern s'engage à synchroniser. Premièrement, les règles de coordination représentées au niveau global sont fixées (visibles au niveau des plusieurs descriptions – voir section 5.3.3.1.). En suite, en utilisant le modèle de négociation proposé (voir section 5.3), ces règles sont décomposées dans des politique de coordination représentées au niveau local (visibles au niveau d'une seule description – voir section 5.3.3.1.) afin d'être manipulées par une seule *description de négociation*.

Dans la modélisation du processus de négociation nous avons introduit *Program Formulae* comme étant un ensemble des méthodes modélisant l'exécution cohérente d'une négociation. Egalement nous avons défini une *description de négociation* représentant l'image de l'évolution d'une négociation en concordance avec les règles que la description doit satisfaire. Ainsi, par la suite nous introduisons **ProgramFormulaeSet** afin de faire la liaison entre l'image d'une négociation à travers une description et les méthodes qui modélisent cette image.

ProgramFormulaeSet est un ensemble des « programs formulae », un pour chaque *description de négociation* impliquée dans le pilotage des règles de coordination spécifiées dans un pattern de coordination. Ainsi un *Program Formulae* modélise les règles de coordination qu'une description doit gérer afin de satisfaire le pattern courrant. La modélisation est faite en termes des méthodes sur les particules présentes dans les atomes de négociations décrivant les phases de négociation visibles dans une description considérée.

Exemple

Pour illustrer ce concept, prenons l'exemple d'un Manager d'atelier d'impression P1 qui décide de sous-traiter un job aux ateliers P2 et P3. Pendant la négociation N_1, il souhaite explorer toutes les propositions faites par ses partenaires en favorisant celles qui sont sur la totalité de la tâche. Le Manager de l'atelier P1 souhaite aussi que la négociation N_1 se termine au moment où il arrive à un accord dans une des deux négociations bilatérales (P1 avec P2 et P1 avec P3), sur la totalité de l'ensemble des attributs qui décrivent la tâche à négocier. Dans le même temps, P1 doit arrêter la négociation avec l'autre participant. Ce type de négociation peut être formalisée par un schéma de coordination de la manière suivante :

TriggerSet

Les conditions de déclenchement du pattern de coordination concernent les participants et leurs descriptions à travers lesquelles ils sont impliqués dans la négociation N_1.

Les ensembles suivants des expressions fixent des contraintes: *i)* sur les participants p1,p2 et p3 qui doivent être impliqués dans la négociation N au même moment de temps ayant les rôles de *initiateur* pour le participant p1 et de *guest* pour les participants p2 et p3 ; *ii)* sur les descriptions impliquées dans ce pattern de coordination qui doivent gérer la négociation N pour les participant p1, p2 et respectivement p3 ; et finalement *iii)* dans le dernier ensemble des expressions sont les contraintes sur les graphes de négociations attachés au descriptions ou

moment t - dans ce cas les conditions sont simples, nous voulons coordonner des descriptions qui sont déjà engagées dans la négociation, ou mois une de leurs phases de négociation est dans un statut de *undefined* -.

$(p1 \in participants(t, N_1))\ (p2 \in participants(t, N_1))\ (p3 \in participants(t, N_1))$
$(role(t,p1, N_1) = initiator)\ (role(t,p2, N_1) = guest)\ (role(t,p3, N_1) = guest)\ ;$

$(vue(d1)=(p1, N_1,R1))\ (vue(d2)=(p2, N_1,R2))\ (vue(d3)=(p3, N_1,R3))\ ;$

$(\exists\ a\ \in d_1(t) : statut(t,d_1,a.ph,d_1) = undefined)$
$(\exists\ b\ \in d_2(t) : statut(t,d_2,b.ph,d_2) = undefined)$
$(\exists\ c\ \in d_3(t) : statut(t,d_3,c.ph,d_3) = undefined).$

RulesSet

$competition(T_1, T_2,d1,d2,d3) :\ T_1 \in \mathcal{T}\ T_2 \in \mathcal{T};$
$\exists a \in d1(T_1)$ et $\exists b\ \in d2(T_1)$ et $\forall c\ \in d3(T_2)$
$statut(T_1,d1,a.ph,d1) = success \wedge statut(T_1,d2,b.ph,d2) = success \wedge\ issues\ (T_1,\ d1,\ a.ph,$
$d1) = issues(T_1,d2,b.ph,d2) \rightarrow\bullet\ statut(T_2,d3,c.ph,d3) = failure;$

$competition(T_1, T_2,d1,d3,d2) :\ \ T_1 \in \mathcal{T}\ T_2 \in \mathcal{T};$
$\exists a \in d1(T_1)$ et $\exists b\ \in d3(T_1)$ et $\forall c\ \in d2(T_2)$:
$statut(T_1,d1,a.ph,d1) = success \wedge statut(T_1,d3,b.ph,d3) = success \wedge\ issues\ (T_1,\ d1,\ a.ph,$
$d1) = issues(T_1,d3,b.ph,d3) \rightarrow\bullet\ statut(T_2,d2,c.ph,d2) = failure;$

Les deux règles de coordination décrivent le fait que si dans la description d1 du partenaire p1 sur la négociation N, il existe une phase de négociation avec le Statut *success* et si dans une description (d2 ou d3) d'un des deux autres participants il existe aussi une phase de négociation avec le Statut *success* et si les deux phases ont la même instanciation des attributs de la tâche négociée, dans ce cas, pour l'autre participant, qui jouent le rôle de *guest*, tous ses phases de négociation vont se trouver dans un Statut *failure*.

Dans ces deux règles la visibilité est globale, les termes des règles sont visibles en prenant plusieurs descriptions de négociation. Comme dans notre approche n'existe par une

entité centrale ayant toutes les informations, nous devons ainsi implémenter la coordination des ces règles globale de manière distribue sur plusieurs descriptions de négociation.

Ces deux règles peuvent être décomposées de la manière suivante dans trois politiques de coordination, chacun visible locale dans une seule description de négociation. Dans ce cas la satisfaction des règles globales sera le résultat de l'émergence de la mise en œuvre des règles locales (voir section 4.5.1.).

- la politique R1 visible localement dans la description d1 :

competition(T_1, T_2,d1,Di) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ $Di \in \{d2,d3\}$;

$\exists a \in d1(T_1)$ et $\forall b \in d1(T_2)$ avec b.ph≠a.ph :

$statut(T_1,d_1,a.ph,d_1) = success \wedge statut(T_1,d_1,a.ph,Di) = success \wedge issues(T_1,d_1,a.ph,d_1) = issues(T_1,d_1,a.ph,Di) \rightarrow\bullet statut(T_2,d_1,b.ph,d_1) = failure;$

La règle fixe le fait que si dans la description *d1* du participant *p1*, il y a un atome *a* dans lequel on trouve un accord avec la description *Di* (l'accord est établi si les deux descriptions sont dans un statut de *success* et leurs instanciations des attributs de la tâches sont les mêmes), dans ce cas au moment suivant de temps tous les autres atomes de la descriptions d1 auront leurs phases dans un statut de *failure*.

Pour les deux autres descriptions les règles sont simples : si la description d1 a refusé la proposition faite dans un atome (l'aperçu pour la description d1 a le statut *failure*), dans le moment suivant de temps la description courante enregistre aussi le refus.

- La politique R2 visible localement dans la description d2 :

fail_complementary_hard(T_1, T_2, d_2,d_1) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$;

$\exists a \in d2(T_1) \wedge \exists a' \in d2(T_2)$: a'.ph = a.ph

$statut(T_1,d2,a.ph,d1) = failure \rightarrow\bullet statut(T_2,d2,a'.ph,d2) = failure$

- La politique R3 visible localement dans la description d3 :

fail_complementary_hard(T_1, T_2, d_3,d_1) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$;

$\exists a \in d3(T_1) \wedge \exists a' \in d2(T_2)$: a'.ph = a.ph

$statut(T_1,d3,a.ph,d1) = failure \rightarrow\bullet statut(T_2,d3,a'.ph,d3) = failure$

Donc avec ces trois politiques compacts des règles de coordination nous pouvons garantir une exécution locale pour les premières deux règles de coordination définies de manière globale.

Par la suite, ces règles sont traduites dans des ensembles des méthodes qui traceront les comportements de chaque description impliquée dans ce schéma de coordination.

ProgramFormulaeSet

Pour ce pattern de coordination nous avons supposé l'existence d'une description pour chaque partenaire impliqué dans la négociation *N.* Dans la partie suivante, nous allons détailler seulement le comportement de la description d_1 qui est plus complexe, les comportements des autres types de descriptions seront présentés dans l'annexe A.

Donc, pour le participant p1 la modélisation de ce pattern de coordination est faite en fixant les méthodes utilisées par la description d_1 afin de garantir les règles imposées dans la politique de coordination R1.

Avec le **program formulae** décrit par la suite nous modélisons non seulement la politique de coordination R1 mais nous définissons aussi toutes les méthodes modélisant le processus entier de négociation que la description d1 doit gérer. Les différentes actions à modéliser sont les suivantes :

- le démarrage de la négociation : la description d1 invite dans la négociation les description d2 et d3 ;
- traitement des propositions de négociation ;
- traitement des messages d'acceptation d'une proposition ;
- mise en œuvre des règles de coordination ;
- traitement des messages de refus d'une proposition ;
- finalisation de la négociation.

Program formulae *pour le comportement de la description d1*

Comme la *description* d1 est supposée voir en totalité les deux négociations bilatérales, elle sera impliquée dans la négociation dès le début, donc du moment de la création des descriptions pour les participants invités. Pour chaque événement **clone_create**(Id, New_Id, Msg) un nouvel atome est créé (1.). La particule {**string_create**(Msg, Rname, Type)}fait appel à une particule computationelle locale qui détermine dans la variable Msg les valeurs des

paramètres de la particule **create**(Rname, Type). La deuxième méthode (2.) fait apparaître une nouvelle particule de représentation **extr**(Rname, initiated, ∅) qui représente l'image qu'une description d'un nouvel participant a sur cette nouvelle phase de négociation. Comme la particule de contrôle **start** est présentée seulement dans le premier atome, par la suite tous les atomes clonés à partir de celui-ci seront des phases de négociation partagées entre la description d_1 du participant p1 et une seule description d'un autre participant. Donc, la première ramification du graphe correspond à la ramification de la négociation dans plusieurs négociations bilatérales.

1. **name**(Id) @ **start** @ **clone_create**(Id, New_Id, Msg) @ {**string_create**(Msg, Rname, Type)} <>- (**start** @ **name**(Id)) & (**freeze** @ **name**(New_Id) @ **create**(Rname, Type))
2. **freeze** @ **create**(Rname, type) <>- **extr**(Rname, initiated, ∅) @ **enable**

A partir de tous les nouveaux atomes créés nous pouvons continuer la négociation avec les nouvelles propositions reçues (3.) introduites dans système par des évènements de type **clone_propose**. La proposition fait évoluer (4.) la nouvelle phase de négociation en construisant premièrement la nouvelle proposition utilisant la particule computationelle {**construct**(I1,Content,I)} et par la suite la avec la nouvelle proposition sont modifiés les Issues des particules de représentation présentes dans l'atome.

3. **name**(Id)@**enable** @ **clone_propose**(Id, New_ Id, Msg) @{**string_propose**(Msg, Rname, Content)} <>- (**enable** @ **name**(Id)) & (**freeze** @ **name**(New_ Id) @ **propose**(Rname, Content))
4. **freeze** @ **localr**(Rname1, S1, I1) @ **extr(**Rname3, S3, I1) @ **propose**(Rname, Content) @ {**construct**(I1,Content,I)}<>- **enable**@ **localr**(Rname1, undefined, I) @ **extr(**Rname3, undefined, I)

Les méthodes de (5.) à (9.) traduisent les dépendances sur les statuts. Dans toutes les phases valides de négociation (celles avec la particule de contrôle **enable**) les partenaires impliqués peuvent accepter la proposition courante (5.). Comme les deux partenaires doivent être d'accord, dans le nouvel atome crée, les particules de représentations pour un seul

participant seront dans un statut de *success* (6. et 7.). Une nouvelle particule de contrôle est introduite (**waiting**) afin de conserver l'atome de négociation seulement pour des événements de type **clone_accept** ou **clone_reject** (8. et 10.). Un accord est trouvé seulement si toutes les deux particules de représentation sont dans un statut *success* (9). Dans ce cas tous les autres atomes de négociation sont annoncés d'arrêter la négociation (la particule **stop** est introduite par le mécanisme de broadcasting dans tous les atomes composant la description de négociation d_1).

5. **name**(Id) @ **enable** @ **clone_accept**(Id, New_ Id, Msg) @ { **string_accept**(Msg, Rname)}<>- (**enable** @ **name**(Id)) & (**freeze** @ **name**(New_ Id) @ **accept**(Rname))
6. **freeze** @ **localr**(Rname1, S1, I) @ **accept**(Rname1)<>- **localr**(Rname1, success, I) @ **waiting**
7. **freeze** @ **extr**(Rname3, S3, I) @ **accept**(Rname3)<>- **extr**(Rname3, success, I) @ **waiting**
8. **name**(Id) @ **waiting** @ **clone_accept**(Id, New_ Id, Msg) @ { **string_accept**(Msg, Rname)}<>- **name**(Id) @ **freeze** @ **accept**(Rname)
9. **localr**(Rname1, success, I) @ **extr**(Rname3, success, I) @ ^**stop** <>- **ready**(I)

Dans les méthodes de (10.) à (11.) nous traitons l'événement de type **clone_reject** et dans la méthode (12.) le message de type **reject**.

La méthode (10.) modélise le traitement du **clone_reject** dans un atome de négociation dans lequel un des participants a fait auparavant un *accept* sur la phase courante de négociation (la particule **waiting** est créée seulement dans une méthode qui traite un message de type **accept** – 7.). La méthode (11.) modélise le traitement du **clone_reject** dans un atome de négociation où la proposition n'est pour le moment ni acceptée ni refusée. Les deux méthodes introduisent la particule message **reject** dans l'atome de négociation courant.

Dans le traitement du message **reject** - la méthode (12.) - nous avons choisi de conserver la phase de négociation associée à l'espace courant, en modifiant tous les statuts en

failure et d'empêcher toutes les évolutions possibles de l'atome en consommant toutes les particules de contrôle.

La dernière méthode (13.) est une loi qui détruit complètement tous les atomes de négociation visibles par la description d1, sauf l'atome de négociation avec l'accord sur la négociation (la particule **stop** a été créée par cet espace – 9.).

10. **name**(Id) @ **waiting** @ **clone_reject**(Id, New_ Id, Msg) @ { **string_reject**(Msg, Rname)}<>- **freeze** @ **name**(New_ Id) @ **reject**(Rname)

11. **name**(Id) @ **enable** @ **clone_reject**(Id, New_ Id, Msg) @ { **string_reject**(Msg, Rname)}<>- **freeze** @ **name**(New_ Id) @ **reject**(Rname)

12. **freeze** @ **localr**(Rname1, S1, I) @ **extr**(Rname3, S3, I) @ **reject**(Rname)<>- **localr**(Rname1, failure, I) @ **extr**(Rname3, failure, I)

13. **stop** <>- **#t**

Donc, de cette manière nous pouvons gérer une coordination entre plusieurs négociations bilatérales menées sur une tâche qui doit être acceptée en totalité par un seul contractant.

Les chapitres suivants proposent plusieurs schémas de coordination afin de montrer comment pour une même négociation, plusieurs relations de dépendance sont gérées dans des modules différents. Pour chaque schéma de coordination nous décrirons premièrement un scénario de négociation en soulignant, de manière informelle, la gestion des différentes dépendances et en suite, de manière formelle, les processus de coordination. Donc, chaque ensemble particulier des dépendances est géré par une ou plusieurs nouvelles descriptions invitées dans la négociation. Ainsi, chaque schéma de coordination modélise un ensemble de règles de coordination cohérent (*RulesSet*) et donne des procédures de coordination (*ProgramFormulaeSet*) en termes d'ensemble des méthodes mises en œuvre au sein d'une ou plusieurs descriptions appartenant aux négociations coordonnées.

7.3. Coordination en monde fermé

Les deux principales caractéristiques de la coordination en monde fermé sont les suivantes :

- les modèles définis prennent en compte seulement des dépendances entre les négociations bilatérales appartenant à une même négociation. Donc, les dépendances prises en compte sont sur des aspects (*Statut, Issues*) caractérisant les *atomes* des *descriptions des négociations* dans la même négociation.
- l'évolution d'une négociation se déroule sans tenir compte des autres négociations dans lesquelles un participant se trouve impliqué. Les descriptions qui géreront les différents schémas de coordination, détaillés par la suite, ont besoin de maintenir des *représentations externes* (voir section 5.2.4.1.) dans lesquelles elles gardent seulement des informations sur des descriptions impliquées dans la négociation courante. De plus, toutes ces informations sont extraites à partir des propositions reçues ou envoyées au cours de la négociation courante.

Le modèle gérera la coordination entre les différentes propositions échangées avec des partenaires différents sur les attributs de l'objet de négociation considéré dans la négociation courante (*dépendances des Issues*). Les règles seront sur : ce que se passe dans les autres négociations bilatérales quand une proposition est envoyée dans une négociation bilatérale ou ce que se passe dans les autres négociations bilatérales quand une proposition est reçue dans une négociation bilatérale.

Le modèle gérera aussi la coordination au niveau des statuts des négociations bilatérales (*dépendances des Statut*). Les règles seront sur: ce que se passe dans les autres négociations bilatérales quand une des négociations bilatérales est dans un Statut *success* ou, ce que se passe dans les autres négociations bilatérales quand une des négociations bilatérales est dans un Statut *failure*.

Par exemple, nous pouvons supposer qu'un Manager d'atelier d'impression décide de sous-traiter un job à des partenaires dans l'alliance, soit pour libérer des ressources en vue de satisfaire une autre requête plus importante, soit pour accepter une requête qu'il ne peut pas réaliser tout seul. Par exemple, le Manager de l'atelier d'impression P1 décide de sous-traiter un job (une impression en noir et blanc de 1000 exemplaires, à un coût inférieur à 10€/exemplaire, dans un délai de 5 jours).

Pour le moment, nous faisons abstraction du processus de décision et de communication, en indiquant seulement que chaque partenaire impliqué dans la négociation doit avoir un support pour une négociation bilatérale multi-attributs en termes de création, d'évaluation et de communication des propositions. Par la suite chaque schéma de coordination gère les règles de coordination décrivant une possible continuation du processus de négociation.

7.3.1. Négociation multi-bilatérale simple

Dans ce cas, le Manager précise qu'il souhaite explorer toutes les propositions faites par les partenaires en favorisant celles qui sont sur la totalité de la tâche. Il précise aussi que la négociation se termine au moment où l'atelier P1 arrive à un accord avec un des partenaires, sur la totalité de l'ensemble des attributs qui décrivent la tâche à négocier et que dans le même temps P1 arrête les négociations avec les autres partenaires.

Ces contraintes peuvent se traduire par le schéma de coordination détaillé par la suite.

TriggerSet

Les conditions du schéma de coordination visent principalement le rôle des ateliers impliqués dans la négociation. Le schéma de coordination est mis en place pour gérer les contraintes au niveau d'un participant p1 qui propose une tâche dans l'alliance, donc qui a le rôle d'*initiator* .

TriggerSet

$(p_1 \in participants(t,N)\ )\ (\ role(t, p_1,N)= initiator)\ ;$

$vue(d_1)=(p_1, N, R_1)\ ;$

$(\exists\ a\ \in d_1(t) : statut(t,d_1,a.ph,d_1) = undefined)$

Le fait que les conditions de déclenchement ne sont pas très nombreuses et qu'elles sont assez faibles permet au pattern de coordination d'être mis en place très tôt dans déroulement du processus de négociation.

C'est par la suite que le pattern de coordination gère le déroulement de la négociation en fonctions des règles de coordination fixées. Et ces règles peuvent être aussi sur les autres participants et les autres descriptions impliquées dans la négociation N.

RulesSet

Pour une négociation, ce schéma de coordination gère des dépendances entre les différentes négociations bilatérales démarrées sur la tâche courante. Ces dépendances sont au niveau des statuts des négociations. L'atelier P1 peut manager, indépendamment pour chaque négociation bilatérale, les propositions qu'il fait. Cette approche favorise la construction d'une proposition en utilisant aussi les informations caractérisant le comportement d'un certain participant. La coordination doit gérer les dépendances au niveau du moment de l'acceptation d'une proposition. Si dans une négociation bilatérale, l'atelier P1 arrive à un accord, il doit arrêter les autres négociations bilatérales faites sur la même tâche. Ces dépendances entre les différentes négociations bilatérales sont fixées par la règle de coordination ***(competition)***. Comme la tâche peut être contractée par un seul participant, cela suppose que la tâche a été acceptée en totalité. Donc, cette contrainte peut être gérée facilement par le module de coordination.

En définissant la règle ***(as_a_block)***, le schéma de coordination garantit que la négociation prend en compte seulement des propositions sur la totalité de la tâche.

competition$(T_1, T_2, d1, Di)$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}\ Di \in negotiation(T1,N)\text{-}\{d1\}$;

$\exists a \in d1(T_1)$; $\forall b \in d1(T_2)$ avec $b.ph \neq a.ph$; $\exists a' \in d1(T_2)$ avec $a'.ph = a.ph$

$statut(T_1,d1,a.ph,d1)=success \wedge statut(T_1,Di,a.ph,Di) \wedge issues(T_1,d1,a.ph,d1) = issues(T_1,Di,a.ph,Di) \wedge role\_d(T_1,Di)=guest \rightarrow\bullet\ statut(T_2,d1,b.ph,d1)=failure$ - si dans la description d1 du participant p1 il existe une phase de négociation dans lequel le statut du *d1* est *success* et elle existe aussi dans une description *di* du participant pi (pi est invité dans la négociation N, donc il a le rôle de *guest*) avec le même statut de *success* et si les ensembles des Issues sont égaux, dans ce cas la négociation s'arrête dans toutes les autres phases présentes dans *d1*.

as_a_block$(T_1, T_2, d1, Di)$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}\ Di \in negotiation(T1,N)\text{-}\{d1\}$;

$\exists a \in d1(T_1)$; $\exists a' \in d1(T_2)$ avec $a'.ph = a.ph$

$\neg(statut(T_1,d1,a.ph,d1)=failure) \wedge \neg(issues(T_1,d1,a.ph,d1).size = issues(T_1,Di,a.ph,Di).size) \wedge role\_d(T_1,Di)=guest \rightarrow statut(T_2,d1,a'.ph,d1).failure$; - la négociation s'arrête dans les propositions où la taille n'est pas celle spécifiée par le participant p1.

Par la suite, ces règles globales peuvent être traduites dans plusieurs politiques de coordination composées de règles visibles local. Une de ces politiques modélise le comportement d'un type particulier de description de négociation (appelé *AsABlock* – voir section 10.1.3.) qui négocie pour le participant p1 *vue(dk)=(p1,N,Rk)*. Les deux règles définies auparavant sont décomposées dans trois types de politiques de coordination : une politique pour une nouvelle description introduite *dk*, une politique pour la description *d1* et un type de politique pour les descriptions des participants invités *di*.

1) politique pour la description dk

Afin de garantir la satisfaction des règles définies par ce schéma de coordination, cette nouvelle description doit avoir la possibilité d'apercevoir et d'agir sur les échanges que le participant p1 a avec chacun des participants invités. Donc, cette description doit gérer les phases de négociation que le participant p1 partage, à travers la description d1, avec les autres participants :

$\exists dk \in N(t1) \wedge \forall a \in d1(t1) : dk \in link(t1, d1, a.ph) \wedge \exists a' \in dk(t1) : a'.ph = a.ph$

Les deux règles suivantes sont la transcription des règles globales définies au début de ce schéma de coordination, mais avec les satisfactions des normes des règles locales en termes de visibilité par rapport à la description dk.

competition$(T_1, T_2, dk, d1, Di) : T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}\ Di \in N(T_1)\text{-}\{d1,dk\};$

$\forall a \in dk(T_1) \wedge \forall b \in dk(T_2)$avec $b.ph \neq a.ph \wedge \exists a' \in dk(T_2)$ avec $a'.ph = a.ph$

$statut(T_1,dk,a.ph,d1)=success \wedge statut(T_1,dk,a.ph,Di)=success \wedge issues(T_1,dk,a.ph,d1) = issues(T_1,dk,a.ph,Di) \wedge role\_d(T_1,Di)=guest \rightarrow\bullet\ statut\ (T_2,dk,b.ph,dk)=failure$

as_a_block$(T_1, T_2, dk, d1, Di) : T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}\ Di \in N(T_1)\text{-}\{d1,dk\};$

$\forall a \in dk(T_1)\ \exists a' \in dk(T_2)$ avec $a'.ph = a.ph$

$\neg(statut(T_1,dk,a.ph,dk)=failure) \wedge \neg(issues(T_1,dk,a.ph,d1).size = issues(T_1,dk,a.ph,Di).size) \wedge role\_d(T_1,Di)=guest \rightarrow statut(T_2,dk,a'.ph,dk)=failure$

2) politique pour la description d1

fail_complementary_hard$(T_1, T_2, d_1, d_k) : T_1 \in \mathcal{T}\ T_2 \in \mathcal{T};$

$\forall a \in d1(T_1) \wedge \exists a' \in d1(T_2) \wedge a'.ph = a.ph :$

$statut(T_1,d1,a.ph,dk)=failure \rightarrow\bullet\ statut(T_2,d1,a'.ph,d1)=failure$

La règle satisfaite au niveau de la description d1 garantit que si la représentation externe de la description dk est dans un statut failure, la description d1 sera dans un statut *failure* .

3) politique pour les descriptions Di

fail_complementary_hard(T_1, T_2, D_i,d_k) : $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}\ Di \in N(T_1)\text{-}\{d1,dk\}$;

$\forall a \in Di(T_1) \wedge \exists a' \in Di(T_2) \wedge a'.ph=a.ph$:

$statut(T_1,Di,a.ph,dk)=failure \wedge role\_d(T_1,Di)=guest \rightarrow\bullet\ statut(T_2,di,a'.ph,Di)=failure$

La règle garantit que chaque fois que dk a refusé de continuer la négociation dans une phase de négociation, les descriptions des participants invités vont arrêter aussi la négociation dans la phase respective.

ProgramFormulaeSet

Ainsi, les politiques présentées seront traduites dans plusieurs *program formulae* composés par des méthodes modélisant le comportement des descriptions pendant le déroulement de la négociation. Les différents programs formules sont présentés dans l'annexe.

Donc, de cette manière nous pouvons gérer une coordination entre plusieurs négociations bilatérales menées sur une tâche qui doit être acceptée en totalité par un seul contractant. Ce type de coordination est pris en charge principalement au niveau d'une seule description de négociation de type *AsABlock*.

7.3.2. Négociation avec une tâche découpée en différents lots

Dans cette continuation du notre scénario, le Manager précise qu'il souhaite explorer les propositions faites sur des parties de la tâche, mais de tel sort que la tâche entière soit contractualisée à un ou plusieurs partenaires. Donc, la tâche est découpée en plusieurs lots en fonction de la taille et afin d'arriver à un contrat, l'infrastructure doit gérer l'acceptation des tous les morceaux. De même façon que dans le scénario précédent, il reste aussi la contrainte qui précise que la négociation se termine au moment où l'atelier P1 arrive à un accord sur la totalité de la tâche à négocier et que dans le même temps toutes les autres propositions de négociation sont arrêtées.

Par la suite, on propose un schéma de coordination qui gère des négociations sur la tâche divisée en deux morceaux pour deux partenaires différents.

TriggerSet

Ce schéma de coordination peut être déployé dans les mêmes conditions que pour le schéma précédent. Le schéma est aussi proposé pour l'atelier qui initie la négociation, dans notre cas atelier P1 et qui peut aussi négocier avec plusieurs participants:

TriggerSet

$(p_1 \in participants(t,N)\ )\ (\ role(t, p_1,N)= initiator)\ ;$

$vue(d_1)=(p_1, N, R_1)\ ;$

$(\exists\ a\ \in d_1(t) : statut(t,d_1,a.ph,d_1) = undefined)$

RulesSet

Dans ce type de schéma de coordination sont plusieurs dépendances entre les différentes négociations bilatérales. Une première caractéristique de ce schéma de coordination est le fait que les négociations bilatérales soient groupées deux par deux. Entre deux négociations bilatérales du même binôme, les dépendances sont : *i)* premièrement sur les attributs négociés – les propositions doivent être complémentaires, et *ii)* deuxièmes sur les statuts – la négociation est finie seulement si les deux négociations bilatérales sont finies. A partir des ces

dépendances entre les négociations bilatérales groupées, nous avons défini la première *règle **split_coordination**)*. Cette règle garantit qu'au moment où une proposition faite par un des participants n'est pas sur la totalité de la tâche, la partie restante est automatiquement proposée à l'autre participant.

Les deux règles suivantes fixent des contraintes au niveau des statuts des négociations bilatérales. *La règle **split_success**)* garantit que si la négociation aboutit à un *success* pour le participant initiateur (dans notre cas le participant p1) et si les deux négociations bilatérales du même binôme aboutissent à un *success* et si les deux morceaux couvrent la totalité de la tâche, dans ce cas les autres négociations sont arrêtées. Ce schéma n'exclut pas le fait qu'un participant décide d'accepter la totalité de la tâche, ce cas est pris en compte par *la règle **competition**)*.

***split_coordination**(T_1, T_2,d1,Di,Dj) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ $Di \in N(T_1)$-{d1} $Dj \in N(T_1)$-{d1};*

$\exists a \in d1(T_1)$; $a' \in Di(T_1)$ et $a'' \in Dj(T_1)$ avec *$a.ph = a'.ph = a''.ph$;* $\forall b \in d1(T_2)$ avec *$b.ph \neq a.ph$*

$statut(T_1,d1,a.ph,d1) = undefined \wedge statut(T_1,Di,a'.ph,Di)=undefined \wedge statut(T_1,Dj,a''.ph,Dj) =undefined \wedge \neg(issues(T_1,d1,a.ph,d1).size = issues(T_1,Di,a'.ph,Di).size) \wedge role\_d(T_1,Di)=guest \wedge role\_d(T_1,Dj)=guest \rightarrow\bullet issues(T_2,Dj,a''.ph,Dj) = (issues(T_1,d1,a.ph,d1) - issues(T_1,Di,a'.ph,Di))$

***split_success**(T_1, T_2,d1,Di,Dj) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ $Di \in N(T_1)$-{d1} $Dj \in N(T_1)$-{d1};*

$\exists a \in d1(T_1)$; $a' \in Di(T_1)$ et $a'' \in Dj(T_1)$ avec *$a.ph = a'.ph = a''.ph$;* $\forall b \in d1(T_2)$ avec *$b.ph \neq a.ph$*

$statut(T_1,d1,a.ph,d1)=success \wedge statut(T_1, Di,a'.ph, Di)=success \wedge statut(T_1, Dj,a''.ph, Dj)=success \wedge (issues(T_1,d1,a.ph,d1).size = issues(T_1, Di,a'.ph, Di).size + issues(T_1, Dj,a''.ph, Dj).size) \wedge role\_d(T_1, Di)=guest \wedge role\_d(T_1, Dj)=guest \rightarrow\bullet statut(T_2,d1,b.ph,d1)=failure;$

- si dans la description d1 du participant p1 existe un atome correspondant à une phase de négociation dans laquelle le statut du d1 est *success* et si cette phase existe aussi dans deux autres descriptions *di* et *dj* pour les participants *pi* et *pj* (pi et pj sont invités dans la négociation N, ils ont les rôles de *guest*) avec le même statut de *success* et si leurs propositions sur la taille de la tâche à négocier sont complémentaires, dans ce cas la négociation s'arrête pour toutes les autres propositions faites dans d1.

competition$(T_1, T_2,d1,Di,Dj)$: $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ $Di \in N(T_1)$-{d1} $Dj \in N(T_1)$-{d1};

$\exists a \in d1(T_1)$; $a' \in Di(T_1)$ et $a'' \in Dj(T_1)$ avec $a.ph = a'.ph = a''.ph$; $\forall b \in d1(T_2)$ avec $b.ph \neq a.ph$

$statut(T_1,d1,a.ph,d1)=success \wedge statut(T_1,d1,a.ph, Di)=success \wedge \neg statut(T_1,d1,a.ph, Dj)=failure \wedge issues(T_1,d1,a.ph,d1) = issues(T_1,d1,a.ph, Di) \wedge role\_d(T_1, Di)=guest \wedge role\_d(T_1, Dj)=guest \rightarrow\bullet statut(T_2,d1,a.ph, Dj)=failure \wedge statut(T_2,d1,b.ph,d1)=failure;$

- la négociation peut aboutir à un accord avec un seul contractant si celui-ci accepte la totalité de la tâche proposée.

Par la suite, les règles peuvent être traduites dans une politique composée de règles bien définies qui traceront le comportement d'un type particulier de description de négociation qui négocie pour le participant p1, (appelée *Split* - voir section 10.1.4.) $vue(dk)=(p1,N,Rk)$. De la même façon que pour la description de type *AsABlock,* cette description doit être capable d'agir sur les phases de négociation dont le participant p1 partage avec les autres participants. Mais, à la différence du premier schéma de coordination où une phase de négociation est partagée entre l'atelier P1 et un seul autre atelier, ici le participant p1 partage une phase de négociation avec deux autres participants. Les trois règles définies auparavant sont décomposées dans trois types de politiques de coordination, une politique pour la nouvelle description introduite *dk*, une politique pour la description *d1* et un type de politique pour les descriptions des participants invités *Di* ou *Dj*:

1) politique pour la description dk

split_coordination$(T_1, T_2,dk,d1,Di,Dj)$: $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ $Di \in N(T_1)$-{d1,dk} $Dj \in N(T_1)$-{d1,dk};

$\forall a \in dk(T_1)$; $\forall b \in dk(T_2)$ avec $b.ph \neq a.ph \wedge \exists a' \in dk(T_2)$ avec $a'.ph= a.ph$

$statut(T_1,dk,a.ph,d1)=undefined \wedge statut(T_1,dk,a.ph, Di)=undefined \wedge \neg(issues(T_1,dk,a.ph,d1).size = issues(T_1,dk,a.ph, Di).size) \rightarrow\bullet issues(T_2,dk,a'.ph, Dj) = (issues(T_1,dk,a.ph,d1) - issues(T_1,dk,a.ph, Di))$

split_success$(T_1, T_2,dk, d1,Di,Dj)$: $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$ $Di \in N(T_1)$-{d1,dk} $Dj \in N(T_1)$-{d1,dk};

$\exists a \in dk(T_1)$ $\forall b \in dk(T_2)$ avec $b.ph \neq a.ph$

$statut(T_1,dk,a.ph,d1)=success \wedge statut(T_1,dk,a.ph, Di)=success \wedge statut(T_1,dk,a.ph, Dj)=success \wedge issues(T_1,dk,a.ph,d1).size = issues(T_1,dk,a.ph,Di).size + issues(T_1,dk,a.ph, Dj).size \rightarrow\bullet statut(T_2,dk,b.ph,dk)=failure$

competition$(T_1, T_2, d_k, d_1, D_i, D_j)$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}\ D_i \in N(T_1)\text{-}\{d_1, d_k\}\ D_j \in N(T_1)\text{-}\{d_1, d_k\}$;

$\exists a \in d_k(T_1)$; $a' \in D_i(T_1)$ et $a'' \in D_j(T_1)$ avec $a.ph = a'.ph = a''.ph$; $\forall b \in d_k(T_2)$ avec $b.ph \neq a.ph$

$statut(T_1, d_k, a.ph, d_1) = success \wedge statut(T_1, dk, a.ph, D_i) = success \wedge \neg statut(T_1, d_k, a.ph, D_j) = failure \wedge issues(T_1, d_k, a.ph, d_1) = issues(T_1, d_k, a.ph, D_i) \rightarrow\bullet\ statut(T_2, d_k, a'.ph, D_j) = failure \wedge statut(T_2, d_k, b.ph, d_k) = failure$

Ces trois règles sont la transcription des règles définies au début de ce schéma de coordination, mai avec les satisfactions des normes des règles bien définies par rapport à la description dk.

Les deux suivantes politiques de coordination garantissent que chaque fois que d_k a refusé de continuer la négociation dans une phase de négociation, la description d_1 ou les descriptions des participants invités vont arrêter aussi la négociation dans leur atome attaché à la phase respective.

2) politique pour la description d_1

fail_complementary_hard(T_1, T_2, d_1, d_k) : $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\forall a \in d1(T_1) \wedge \exists a' \in d1(T_2) \wedge a'.ph = a.ph$:

$statut(T_1, d_1, a, d_k) = failure \rightarrow\bullet\ statut(T_2, d_1, a', d_1) = failure$

3) politique pour les descriptions D_i

fail_complementary_hard(T_1, T_2, D_i, d_k) : $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}\ D_i \in N(T_1)\text{-}\{d_1, d_k\}$;

$\forall a \in D_i(T_1) \wedge \exists a' \in D_i(T_2) \wedge a'.ph = a.ph$:

$statut(T_1, D_i, a, d_k) = failure \rightarrow\bullet\ statut(T_2, D_i, a', d_i) = failure.$

ProgramFormulae

Les règles présentées auparavant sont traduites dans des méthodes modélisant le comportement de la description d_k. Le script avec les explications est détaillé dans l'annexe.

Avec ces deux premiers scénarios les dépendances gérées ont été entre des descriptions participantes dans la même négociation et donc nous avons coordonné des négociations

bilatérales composant une même négociation. L'objectif de la prochaine section est de définir des relations de dépendance entre des descriptions participantes dans des négociations différentes et donc de coordonner pour un participant plusieurs négociations en déroulement dans le même moment du temps.

7.4. Coordination en monde ouvert

La première caractéristique de la coordination en monde ouvert est que, les nouvelles dépendances rajoutées ne sont pas en contradiction avec les différentes dépendances gérées par les schémas de coordination en monde fermé. Ces nouvelles dépendances pour les négociations coordonnées en monde ouvert restreignent les évolutions de ces négociations sans annuler leurs dépendances en monde fermé.

La coordination en monde ouvert peut être faite entre des négociations bilatérales ciblées appartenant à des négociations différentes ou directement entre des négociations composées des plusieurs négociations bilatérales. Comme le rôle d'un participant peut changer d'une négociation à l'autre, les dépendances prises en compte sont sur des aspects (*Statut, Issues*) caractérisant les différentes tâches d'impression négociées et aussi sur les différents *rôles* d'un participant.

Donc, pour un participant impliqué dans plusieurs négociations dans le même temps, les schémas de coordination en monde ouvert vont manager: *i)* des dépendances d'issues, pour modéliser les nouvelles propositions afin de tenir compte des différentes propositions envoyées ou reçues au cours des plusieurs négociations; *ii)* des dépendances de statut, pour décider de continuer ou pas certaines négociations en prenant en compte les statuts des autres négociations ; et *iii)* finalement, des dépendances de rôle qui peuvent influencer les manières dont les différentes négociations peuvent être liées.

7.4.1 Coordination de négociations bilatérales

Nous nous intéressons dans ce scénario au moins à deux négociations bilatérales menées en parallèle. Ces deux négociations impliquent les mêmes participants mais avec des rôles différents dans les sous-traitances envisagées : le participant d'une négociation qui joue le rôle de *initiator* dans la première négociation, joue le rôle de *guest* dans la deuxième négociation et inversement pour le deuxième participant. A un moment donné, entre ces deux négociations, des dépendances peuvent être fixées afin de les gérer comme un échange des tâches.

Dans ce scénario, nous supposons que le Manager de l'atelier P1, engagé déjà dans la négociation N1, précise qu'il souhaite explorer la possibilité de faire un échange. La mise en place de ce scénario nécessite le développement d'un ensemble des règles de décision visant à déterminer les conditions déclenchant la mise en place d'une procédure d'échange (swap) entre les négociations concernées (voir section 10.2.2.). Dans cette partie nous allons présenter seulement les dépendances et la manière dans laquelle elles sont gérées par un schéma de coordination proposé par la suite.

TriggerSet

Les conditions pour la mise en place du schéma de coordination visant un swap entre deux négociations bilatérales sont à la fois sur les rôles de participants impliqués et sur les statuts de leurs deux négociations.

Dans ce scénario, nous supposons que c'est le participant p1 qui décide d'explorer et de mettre en place une coordination entre deux négociations différentes. Dans cette partie on ne détaille pas quel est le mécanisme de trouver les deux négociations, mais seulement quelles sont leurs caractéristiques afin d'être impliquées dans ce schéma de coordination.

Les premières contraintes sont sur les participants impliqués et leurs rôles – les deux négociations bilatérales doivent avoir les mêmes participants, et chaque participant a des rôles différents dans les deux négociations.

Les autres contraintes sont sur le fait que les négociations doivent être en cours de déroulement dans les descriptions considérées et que les descriptions impliquées dans la même négociation partagent au mois une phase de négociation. Ce type de contrainte nous l'avons

modélisé comme des conditions sur les statuts des négociations trouvées dans les nœuds des graphes décrivant la négociation au même moment de temps.

TriggerSet

$(p_1 \in participants(t,N_1))\ (p_2 \in participants(t,N_1))$
$(role(t,p_1,N_1) = initiator)\ (role(t,p_2,N_1) = guest)$
$(p_1 \in participants(t,N_2))\ (p_2 \in participants(t,N_2))$
$(role(t,p_2,N_2) = initiator)\ (role(t,p_1,N_2) = guest)$;

$(vue(d_{11})=(p_1,N_1,R_{11}))\ (vue(d21)=(p_1,N_2,R_{21}))\ (vue(d_{12})=(p_2,N_1,R_{12}))\ \ (vue(d_{22})=(p_2, N_2, R_{22}))$;

$(\exists a \in d_{11}(t),\ \exists a' \in d_{12}(t) :\ a.ph = a'.ph \wedge$
$statut(t,d_{11},a.ph,d_{11})=statut(t,d_{12},a'.ph,d_{12})=undefined\ )$
$(\exists b \in d_{21}(t),\ \exists b' \in d_{22}(t) :\ a.ph = a'.ph \wedge$
$statut(t,d_{21},a.ph,d_{21})=statut(t,d_{22},a'.ph,d_{22})=undefined\ )$

RulesSet

Etant une coordination en monde ouverte, ce schéma de coordination gère des dépendances entre des négociations bilatérales qui font partie des négociations sur des tâches différentes. Dans ce schéma de coordination on ne cherche pas à définir des contraintes sur les propositions échangées dans les deux négociations. Les contraintes sont sur les statuts des deux négociations, et notamment sur les statuts finaux (*success* et *failure*).

La différence entre les deux négociations est que dans une des négociations (N_2), c'est le participant p1 qui est invité dans la négociation, donc il n'a pas forcement toute l'image de cette négociation. Par contre, dans la négociation qu'il a démarrée (N_1), il gère la totalité du processus qui peut être très complexe si plusieurs participants sont impliqués. Donc, dans ce schéma de coordination, nous fixons des règles entre deux négociations bilatérales qui sont vues de manière différente par le participant p_1. La première négociation bilatérale peut faire partie d'un ensemble de plusieurs négociations démarrées sur la même tâche et qui composent la négociation N_1. La deuxième négociation bilatérale représente pour p1 la seule négociation qu'il a sur la tâche associée et qui compose la négociation N_2. Cette caractéristique est prise en compte dans les deux règles de coordination proposées dans ce schéma.

La première règle est une règle de coordination forte par laquelle le participant p_1 décide d'arrêter la négociation sur la tâche qu'il propose au participant p_2 si celui-ci a refusé toutes les propositions du p_1 dans la deuxième négociation – *la règle **fail_complementary_hard)***. Par contre, si la deuxième négociation se termine avec un *success*, cela ne garantit pas que la négociation N_1 sera finie avec un *success*, du fait que la tâche proposée dans cette négociation peut être contractée par un autre participant impliqué dans une autre négociation bilatérale sur la même tâche – *la règle **success_complementary_soft)***.

fail_complementary_hard$(T_1, T_2, d_{21}, d_{11})$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\forall a \in d_{21}(T_1)\ \ \forall b \in d_{11}(T_2)$

$statut(T_1,d_{21},a.ph,d_{21})=failure \rightarrow\bullet\ statut(T_2,d11,b.ph,d11)=failure$;

- si la négociation N2 se termine avec un échec, la négociation N1 se termine aussi avec un échec.

success_complementary_soft$(T_1, T2, d_{21}, d_{11})$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\exists a \in d21(T_1)\ \ \exists b \in d11(T_2)$

$statut(T_1,d21,a.ph,d21) = success \rightarrow statut(T_2, d11, b.ph, d11) = success$;

- si la négociation N2 se termine avec un accord, le succès de la négociation N1 n'est pas garanti.

Ce type de schéma de coordination implique l'existence d'une description, de type *SwapOut,* qui participe dans la négociation N1 $d_{swapout} \in N1(T_1)$est qui est en liaison avec la description d11 et une autre description qui représente la négociation N2, cela de type *SwapIn* $d_{swapin} \in N2(T_1)$. Leurs groups des règles bien définies sont les suivants :

1) politique pour la description $d_{swapout}$:

fail_complementary_hard$(T_1, T_2, d_{swapotu}, d_{swapin})$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\forall a \in d_{swapotu}(T_1)\ \ \forall a' \in d_{swapotu}(T_2)$

$statut(T_1,d_{swapout},a.ph, d_{swapin})=failure \rightarrow\bullet\ statut(T_2,d_{swapout},a.ph, d_{swapout})=failure$

success_complementary_hard$(T_1, T_2, d_{swapotu}, d_{swapin})$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\exists b \in d_{swapout}(T_1)\ \ \exists b' \in d_{swapout}(T_2)$

$statut(T_1,d_{swapout},b.ph, d_{swapin})=success \rightarrow\bullet\ statut(T_2,d_{swapout},b'.ph, d_{swapout})=success$

La description $d_{swapout}$ maintien une liaison avec la négociation N2 à travers description d_{swapin}. Si la description d_{swapin} est dans un statut *failure,* donc la négociation N2 est dans *failure*, dans ces cas $d_{swapout}$ passe dans un statut *failure*. Même chose si la négociation N2 est dans un statut de *success*, dans ce cas d_{swapin} est dans *success* et par la suite $d_{swapout}$ sera dans *success.*

Les dépendances entre d_{swapin} et la négociation N2 sont manipulées par de règles entre d_{swapin} et la description de négociation d_{21} à travers de laquelle le participant p1 négocie dans la négociation N2.

2) politique pour la description d_{swapin} :

fail_complementary_hard$(T_1, T_2, d_{swapin}, d_{21})$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\forall a \in d_{swapin}(T_1)\ \ \exists a' \in d_{swapin}(T_2)\ a'.ph = a.ph$

$statut(T_1, d_{swapin}, a, d_{21}) = failure \rightarrow\bullet\ statut(T_2, d_{swapin}, a', d_{swapin}) = failure$

- si la description d_{21} est dans statut *failure*, signifiant que la négociation N2 pour le participant p1 est dans statut *failure*, par la suite d_{swapin} passe dans un statut *failure*.

success_complementary_hard$(T_1, T_2, d_{swapin}, d_{21})$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\exists b \in d_{swapin}(T_1)\ \ \forall b' \in d_{swapin}(T_2)$

$statut(T_1, d_{swapin}, b, d21) = success \wedge statut(T_1, d_{swapin}, b, d22) = success \rightarrow\bullet\ statut(T_2, d_{swapin}, b', d_{swapin}) = success$

- si un accord est trouvé dans la négociation N2 entre la description *d21* du participant p1 et la description *d22* du participant p2, par la suite d_{swapin} passe dans un statut de *success*.

Les descriptions $d_{swapout}$ et d_{swapin} sont utilisées seulement pour maintenir une interaction entre les négociations N1 et N2 dans lesquelles le participant p1 est impliqué. Les décisions finales et la transcription des règles présentées au début de ce schéma sont au niveau de la description d_{11} :

3) politique pour la description d_{11} :

fail_complementary_hard$(T_1, T_2, d_{11}, d_{swapout})$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\forall a \in d_{11}(T_1)\ \ \exists a' \in d_{11}(T_2)\ a'.ph = a.ph$

$statut(T_1, d_{11}, a.ph, d_{swapout}) = failure \rightarrow\bullet\ statut(T_2, d_{11}, a'.ph, d_{11}) = failure$

- le statut de failure pour la description $d_{swapout}$ signifiant que la négociation N2 est finie avec un échec fait que la description d_{11} se termine avec un échec.

success_complementary_soft(T_1, T_2, d_{11}, $d_{swapout}$) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$;

$\exists b \in d_{11}(T_1)$ $\exists b' \in d_{11}(T_2)$

$statut(T_1, d_{11}, b, d_{swapout}) = success \rightarrow statut(T_2, d_{11}, b', d_{11}) = success$

- par contre, le *success* de la négociation N2 ne garantit pas le *success* immédiat dans la négociation N1.

ProgramFormulae

Comme la règle de coordination forte implique seulement des actions sur la négociation *N1*, dans l'annexe nous allons présenter seulement le comportement de la description de type *SwapOut*. La description de type *SwapIn* qui n 'est pas impliqué dans coordination et qui a des attributions purement de communication sera présentée seulement comme un service (voir section 10.2.2.).

7.4.2. Coordination de négociations multi-bilatérales

Nous nous intéressons dans ce scénario au moins à deux négociations menées en parallèle par le même atelier d'impression.

Dans ce scénario nous supposons qu'à un moment donné, le Manager de l'atelier P1 est impliqué dans deux négociations sur des tâches différentes, mais dans les deux cas, s'il réussissait de trouver des accords, il serait obligé de gérer le transport. Donc, le manager peut décider de créer une liaison entre les deux négociations afin d'explorer la possibilité de faire un seul transport pour les deux tâches qu'il est en train de négocier.

La mise en place de ce scénario nécessite également le développement des mécanismes de décisions et de communications visant à faciliter la mise en place d'une procédure de communication entre deux négociations concernées (voir section 10.2.3.). Dans cette partie nous allons présenter seulement les dépendances et la manière dans laquelle elles sont gérées par un schéma de coordination.

TriggerSet

Pour ce type de schéma de coordination il n'y a pas des contraintes sur les rôles joués par l'atelier dans les négociations explorées. Les contraintes que les deux négociations

différentes doivent respecter sont sur les statuts des négociations et les attributs des tâches négociées. La dernière expression fixe le fait que les deux négociations supposent une livraison dans un espace géographique identique ou très proche, donc il est impérativement nécessaire que l'attribut identifiant la destination du transport soit instancié dans les deux négociations avec la même valeur.

TriggerSet

$(p_1 \in participants(t,N_1))\ (p_1 \in participants(t,N_2))$;

(vue(d11)=(p1, N1, R11)) (vue(d21)=(p1, N2, R21));

$(\exists a \in d_{11}(t1),\ \exists b \in d_{21}(t)$:

$issues(t,d_{11},a.ph,d_{11}).location = issues(t,d_{21},b.ph,d_{21}).location)$

RulesSet

Ce schéma de coordination cherche à gérer, pour un atelier, deux négociations en parallèle afin que les livraisons soient faites dans des lapses de temps proches et que les tailles des deux tâches sont complémentaires pour entrer dans le même transport.

A la différence du scénario Swap présenté auparavant, ici les deux négociations sont gérées au même niveau. Dans le schéma Swap la coordination est faite principalement au niveau d'une seule négociation. Par contre, dans ce nouveau type de schémas de coordination on cherche à satisfaire les mêmes dépendances d'issues et de statuts pour les deux négociations.

Donc les principales règles - *les règles a)* et *b)* – décrivent les dépendances entre les propositions faites dans les deux négociation afin de satisfaire les contraintes d'une possible synchronisation d'un transport commun des deux tâches. Cette synchronisation suppose une coordination entre les assertions faites aux niveaux des attributs *size* et *delayTransp* dans les deux négociations. En définissant *Size1* et *Size2* comme des structures des données fixant des valeurs complémentaires[15] sur les attributs *size,* les règles sont :

[15] Supposons que le transport maximal que l'atelier P1 peut faire est de 1000k, dans deux négociations N1 et N2 la complémentarité pour le transport est le fait que la somme des tailles des deux tâches ne doit pas dépasser le transport maximal.

*a)**transp_coordination**(T_1, T_2,d11,d21) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$;*

$\forall a \in d_{11}(T_1)$ $\forall b \in d_{21}(T_1)$ $\exists c \in d_{21}(T_2)$

issues(T_1,d11,a.ph,d11).size=Size1 $\wedge$ $\neg$issues(T_1,d21,b.ph,d21).size=Size2 $\rightarrow\bullet$ issues(T_2,d21,c.ph,d21).size=Size2 $\wedge$ issues(T_2,d21,c.ph,d21).delayTransp = issues(T_1,d11,a.ph,d11).delayTransp ;

- si dans la description d11 du participant p1 dans la négociation N1, il existe un atome décrit par un ensemble d'attributs avec la valeur *Size1* pour la taille de la tâche et si dans la description d21 du même participant p1 mais dans la négociation N2, il n'existe pas un atome avec un ensemble de valeurs complémentaire *Size2* par rapport aux valeurs nécessaires pour une synchronisation du transport, dans ce cas une nouvel atome sera créé dans la description d21 avec un proposition complémentaire.

*b)**transp_coordination**(T_1, T_2,d21,d11) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$;*

$\forall a \in d_{21}(T_1)$ $\forall b \in d_{11}(T_1)$ $\exists c \in d_{11}(T_2)$

issues(T_1,d21,a.ph,d21).size=Size1 $\wedge$ $\neg$issues(T_1,d11,b.ph,d11).size=Size2 $\rightarrow\bullet$ issues(T_2,d11,c.ph,d11).size = Size2 $\wedge$ issues(T_2,d11,c.ph,d11).delayTransp = issues(T_1,d21,a.ph,d21).delayTransp;

- la même règles qu'auparavant, mais cette fois la proposition automatiquement créée est dans la description d11 de la négociation N1.

Comme les deux négociations peuvent être des négociations complexes (composées des plusieurs négociations bilatérales), le fait de trouver un accord ne doit pas dépendre seulement des dépendances de synchronisation du transport. Par contre, dans les cas qu'une des négociations échoue, dans ce schéma on fixe des règles fortes afin que l'autre sera aussi arrêtée – *les règles c)* et *d)*.

*c) **fail_complementary_hard**(T_1, T_2, d_{11}, d_{21}) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$;*

$\forall a \in d_{11}(T_1)$ $\forall b \in d_{21}(T_1)$ $\exists b' \in d_{21}(t2)$ b'.ph =b.ph

statut(T_1,d11,a.ph,d11)=failure $\wedge$ issues(T_1,d11,a.ph,d11).size = Size1 $\wedge$ issues(T_1,d21,b.ph,d21).size=Size2 $\wedge$ issues(T_1,d21,b.ph,d21).delayTransp = issues(T_1,d11,a.ph,d11).delayTransp $\rightarrow\bullet$ statut(T_2,d21,b'.ph,d21) = failure;

- si la négociation N1 se termine avec un échec dans un atome, la négociation N2 est arrêtée aussi dans les atomes avec les propositions complémentaires.

d) ***fail_complementary_hard***$(T_1, T_2, d_{21}, d_{11})$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\forall a \in d_{21}(T_1)\ \forall b \in d_{11}(T_1)\ \exists b' \in d_{11}(t2)\ b'.ph = b.ph$

$statut(T_1,d21,a.ph,d21)=failure \wedge issues(T_1,d21,a.ph,d21).size = Size1 \wedge issues(T_1,d11,b.ph,d11).size=Size2 \wedge issues(T_1,d21,a.ph,d21).delayTransp = issues(T_1,d11,b.ph,d11).delayTransp \rightarrow\bullet statut(T_2,d11,b'.ph,d11) = failure;$

- si la négociation N2 se termine avec un échec dans un atome, la négociation N1 est arrêtée aussi dans les atomes avec les propositions complémentaires.

Parce qu'au niveau des deux négociations les règles sont les mêmes, ce schéma de coordination sera mis en place par deux descriptions de négociation identiques, une pour chaque négociation impliquée dans cette coordination. Les politiques de coordination des nouvelles descriptions introduites (appelées aussi descriptions de type *Transp*) et impliquées dans ce mécanisme de coordination seront détaillés par la suite.

1) politique pour la description $d_{Transp1}$:

La description $d_{Transp1}$ – présente dans la négociation N1- garde des représentations externes pour la description d_{11} et pour une autre description de type Transp, $d_{Transp2}$ – présente dans la négociation N2.

La transcription des règles présentées auparavant dans des règles bien définies par rapport à la description $d_{Transp1}$ est la suivante :

transp_coordination$(T_1, T_2, d_{Transp1}, d_{Transp2})$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\forall a \in d_{Transp1}(T_1)\ \exists a' \in d_{Transp1}(T_2)$: $a'.ph = a.ph$;

$issues(T_1,d_{Transp1},a.ph,d_{Transp2}).size=Size1 \wedge \neg issues(T_1,d_{Transp1},a.ph, d_{Transp1}).size=Size2 \rightarrow\bullet issues(T_2, d_{Transp1},a'.ph, d_{Transp1}).size=Size2 \wedge issues(T_2, d_{Transp1}, a'.ph, d_{Transp1}).delayTransp = issues(T_1, d_{Transp1}, a.ph, d_{Transp2}).delayTransp$

- une proposition reçue dans la description $d_{Transp2}$ avec une taille de Size1, génère dans la description $d_{Transp1}$ un proposition complémentaire.

transp_coordination$(T_1, T_2, d_{Transp1}, d_{11})$: $T_1 \in \mathcal{T}\ T_2 \in \mathcal{T}$;

$\forall a \in d_{Transp1}\ (T_1) \wedge \exists a' \in d_{Transp1}\ (T_2)$: $a'.ph = a.ph$;

$issues(T_1, d_{Transp1}, a.ph, d_{11}).size=Size1 \wedge \neg issues(T_1, d_{Transp1}, a.ph, d_{Transp2}).size=Size2 \rightarrow\bullet issues(T_2, d_{Transp1}, a'.ph, d_{Transp1}).size=Size1 \wedge issues(T_2, d_{Transp1},$

a'.ph, $d_{Transp1}$).delayTransp = issues(T_1, $d_{Transp1}$, a.ph, d_{11}).delayTransp ∧ issues(T_2, $d_{Transp1}$, a'.ph, $d_{Transp2}$).size=Size2

- une proposition reçue dans la description d_{11} avec une taille de Size1, génère dans la description d_{T1} une proposition complémentaire pour la représentation externe du d_{T2}.

fail_complementary_hard(T_1, T_2, $d_{Transp1}$, $d_{Transp2}$) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$;

$\forall a \in d_{Transp1}$ (T_1) ∧ $\exists a' \in d_{Transp1}$ (T_2) : a'.ph =a.ph;

statut(T_1, $d_{Transp1}$, a.ph, $d_{Transp2}$)=failure →• statut(T_2, $d_{Transp1}$, a'.ph, $d_{Transp1}$)=failure

- un échec dans la description d_{T2} génère automatiquement un échec dans la description d_{T1}.

fail_complementary_hard(T_1, T_2, $d_{Transp1}$, d_{11}) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$;

$\forall a \in d_{Transp1}$ (T_1) ∧ $\exists a' \in d_{Transp1}$ (T_2) : a'.ph =a.ph;

statut(T_1, $d_{Transp1}$, a.ph, d_{11})=failure →• statut(T_2, $d_{Transp1}$, a'.ph, $d_{Transp1}$)=failure

- un échec dans la description d_{11} génère automatiquement un échec dans la description d_{T1}.

La politique de coordination pour la description d_{T2} a le même format que celui présenté ci dessus, avec la seule différence que la description d_{T2} maintient des représentations externes pour les descriptions d_{T1} et d_{21}.

2) politique pour la description d_{11}:

fail_complementary_hard(T_1, T_2, d_{11}, $d_{Transp1}$) : $T_1 \in \mathcal{T}$ $T_2 \in \mathcal{T}$;

$\forall a \in d_{11}(T_1) \wedge \exists a' \in d_{11}$ (T_2) : a'.ph =a.ph;

statut(T_1,d_{11},a.ph, $d_{Transp1}$)=failure →• statut(T_2,d_{11},a'.ph,d_{11})=failure

Pour la description d_{11}, le mécanisme de coordination est réduit à une synchronisation entre d_{T1} et d_{11} au niveau d'un échec.

ProgramFormulae

La traduction des règles en terme des méthodes traçant le comportement des descriptions $d_{Transp1}$ et $d_{Transp2}$ – descriptions de type *Transp* – est détaillée dans l'annexe.

Avec ces scénarios nous avons proposé une modélisation des ensembles de dépendances groupées dans des schémas de coordination. Ces schémas sont gérés par la suite par des modules de coordination séparés. Nous avons défini ces modules comme des descriptions avec des types de comportements particuliers. Le comportement d'une description est caractérisé par un ensemble des méthodes entre les particules composant les atomes de la *description de négociation* considérée. L'ensemble de méthodes représente la traduction de la politique de coordination attachée à la description considérée.

Cette structuration flexible et décentralisée du mécanisme de coordination en plusieurs modules séparés permet la prise en compte des dépendances prédéfinies et également des dépendances définies dynamiquement.

Les fonctionnalités dans la négociation de ces différents modules de coordination sont :

a. la validation des propositions reçues ou envoyées dans une négociation. *Une proposition validée* est une proposition qui satisfait les relations des dépendances gérées par les modules de coordinations impliqués dans la négociation considérée.
b. la création des nouvelles propositions afin de satisfaire les relations de dépendance. Ces propositions sont créées dans la mesure où pour leur élaboration le mécanisme de satisfaction des contraintes est suffisant et aucun choix non-déterministe ne doit pas être fait par les modules des coordinations.

Ainsi, la principale caractéristique de ce mécanisme de coordination est la capacité à maintenir dynamiquement une représentation de la négociation en concordance avec les différentes dépendances dans lesquelles la négociation considérée est impliquée.

7.5. Synthèse

Nous avons défini une négociation comme un ensemble des descriptions, chacun manipulant une image différente de la même négociation. Ensuite, nous avons défini les types des dépendances existantes entre ces descriptions de négociations et qui peuvent être gérées par un processus de coordination. Par la suite nous avons décomposé le processus de coordination en plusieurs modules qui gèrent des *schémas de coordination* bien définis. Chaque module correspond à des *descriptions de négociations*, qui construisent leurs images sur la négociation en accord avec les contraintes de coordination structurées dans des *politiques de coordination*. Finalement, nous avons défini un **schéma de coordination** comme une structuration du processus de coordination dans des règles de coordination utilisées afin de construire dynamiquement les différentes images qu'un participant a sur ses négociations.

Egalement, dans ce chapitre nous avons proposé différents schémas de coordination qui peuvent être distribués sur plusieurs modules de coordination afin de gérer de manière automatique et efficace les différentes relations de dépendance entre les propositions reçues ou envoyées dans une ou plusieurs négociations.

Dans la mise en application du modèle de coordination nous avons supposé l'existence de mécanismes de communication entre les descriptions de négociation afin de se synchroniser sur les phases de négociation partagées. Nous avons également supposé l'existence d'une part de mécanismes de raisonnement afin d'évaluer et de générer les propositions et, d'autre part, d'un utilisateur qui exprime ses contraintes pour les négociations et ses attentes.

Dans le chapitre suivant nous présenterons les contraintes de coordination qu'un utilisateur peut exprimer et qui nécessitent des mécanismes de raisonnement afin de les évaluer et les manipuler. Comme les mécanismes de raisonnement ne font pas l'objet de cette thèse, dans la section suivante nous détaillerons seulement les types des contraintes qu'un utilisateur peut spécifier, afin de souligner les relations de dépendances qui peuvent être représentées et qui peuvent influencer par la suite le processus de coordination.

8. Modèle stratégique de coordination

La principale fonctionnalité du système proposé est d'assister un Manager dans le processus de *négociation* de la sous-traitance d'une tâche et dans la *coordination* de plusieurs sous-traitances concurrentes au sein d'*une alliance*. Nous considérons que la coordination des négociations s'appuie aussi sur des *procédures de négociation stratégique*s afin de coder notamment les informations qui caractérisent les règles de fonctionnement que se fixent les participants humains de l'alliance et qui codent les contraintes auxquelles sont soumis les processus de négociation et leurs relations gérées par la suite du système informatique.

Le modèle proposé permet à un participant de définir le sujet de la négociation (le but de la négociation est spécifié dans une structure nommée *Objet de Négociation* – voir section 8.1) d'une part, et d'autre part comment doit se dérouler la négociation (contraintes spécifiées dans une structure nommée *Cadre de Négociation* – voir section 8.2*,* qui décrit de manière générale le processus d'interaction à suivre pour conduire la négociation). Il offre ainsi à chaque participant de l'alliance la possibilité d'agir non seulement pour faire progresser une négociation mais aussi pour faire évoluer son cadre de négociation et l'objet de négociation.

Ainsi, par la suite, sont proposés deux supports pour spécifier les contraintes de coordination gérées au niveau stratégique :

i) un support, appelé *Objet de négociation,* offrant la possibilité de fixer des contraintes stratégiques pour une coordination dans un monde fermé d'une seule négociation ; et

ii) un support*,* appelé *Cadre de Négociation*, qui permet de fixer des contraintes pour plusieurs négociation afin d'être gérées par un processus de coordination stratégique en monde ouvert. Dans ce deuxième support, une partie détaillée sera sur la description de *la tactique* qui fera la liaison entre les schémas de coordination proposés dans la modélisation générique et la mise en œuvre des différents services de coordination.

8.1. Coordination stratégique en monde fermé

La négociation concerne des demandes des tâches comme les impressions, les relieurs ou les distributions des documents selon des caractéristiques données exprimées par des ensembles d'attributs.

Dans notre approche, nous devons prendre en compte que les propositions faites au cours des négociations correspondent aux tâches d'impression qui peuvent être tout à fait complexes. Cette complexité est non seulement donnée par les attributs décrivant la tâche à négocier mais aussi une tâche peut être une composition des plusieurs sous-tâches. Par exemple l'impression et le relieur sont deux sous-tâches différentes et chacune des sous-tâches pourrait être décrite en tant qu'objet multi-attribut. Par ailleurs, chaque attribut est lié aux contraintes et aux critères locaux d'évaluation (par exemple : les contraintes peuvent être employées pour indiquer des valeurs possibles pour les attributs – *cost <= 12*), mais également aux contraintes globales dessinant des dépendances avec d'autres attributs (par exemple : le coût est dépendant de la qualité du papier - *if quality_paper = low then cost <= 10*). Ceci pourrait avoir lieu dans la description d'une tâche simple. Egalement, dans la description des tâches complexes les contraintes peuvent dessiner des dépendances parmi des attributs appartenant à des sous-tâches différentes (par exemple : nous devons imprimer un certain nombre de livres et puis les relier, ainsi le nombre de livres doit être le même dans les deux tâches - *printing.size = biding.size*).

Afin d'exprimer toutes ces contraintes les attributs d'une tâche à négocier sont exprimés en tant qu'attributs indépendants ou interdépendants (voir section 2.3.1.) dans un *Objet de Négociation* qui a la structure suivante :

<ON> ::= <SON> | '(' <name> <ON>* :dependencies <DDF> ')'
<SON> ::= '(' <name> :type <TDF> :candidates <CDF>
:job <JDF> :dependencies <DDF> ')'
<TDF>::= BWP | CP | B
<CDF>::= '(' <CDF>* [<Relation>]')' | <Name> | ' ?' | ' ?*'
<Relation>::= >> | <<

Un *Objet de Négociation* peut être défini comme une composition d'autres *Objets de Négociation* entre lesquels il existe la possibilité d'exprimer les dépendances (:*dependencies*) -

il est ainsi possible d'exprimer des dépendances parmi des négociations concourantes -, autrement il est un objet simple de négociation. Un objet de négociation simple (<*SON*>) a un type unique (:*type*) défini dans l'ontologie du domaine de l'application considérée. Par exemple, dans le cas du domaine des tâches d'impression[16], le type pourrait être l'impression noir et blanc (BWP) ou l'impression en couleur (CP) ou relieur (B).

Le Manager peut indiquer un ensemble ordonné[17] de candidats possibles (:*candidates*) susceptibles de négocier et de pouvoir accepter la tâche proposée (le choix peut être laissé librement au système informatique - plusieurs *?**, un *?*). La tâche proposée pour la négociation sera spécifiée en tant qu'ensemble de différentes attributs (:*job*) avec des dépendances entre elles (:*dependencies*). Chaque attribut est composée d'un ensemble de contraintes et de préférences. Dans l'alliance, nous ne considérons pas les préférences et les dépendances en tant qu'information négociable. Ils sont seulement employés par le système pour contrôler et valider les propositions qui sont échangées pendant une négociation. Notons que l'*Objet de Négociation* peut avoir un ensemble partiel d'attributs au début de la négociation et aussi pendant le processus de négociation des nouveaux attributs peuvent être ajoutés.

Job Definition Format Une tâche négociable peut être décrite de manière suivante :

<JDF>::='(' '('<Issue> (<Constraint>) * [<Preference>]')' * ')'* [<Relation>]
<Issue> ::= <IssueName> (<value> | ' ?')
<Constraint> ::= '(' (<c Exp> | <Constraint>)* [<c Function>]')'
<c Exp> ::= <IssueName> <Operator> <value>
<c Function> ::= <and> | <or> | <xor>
<Operator> ::= < | <= | > | >= | == | = = | enum

Le travail est défini comme un ensemble d'attributs (<*Issue*>) venant de l'ontologie du domaine de l'application considérée - par exemple pour les tâches du domaine d'impression, les issues peuvent être la quantité, le temps d'exécution, la qualité, le coût, les pénalités, le temps du transport -, ainsi que les contraintes (<*Constraint*>) qui sont imposées aux valeurs

[16] Les types des tâches d'impression sont très nombreux, un travail de recherche détaillé est fait par le concerne AGFA [CD'03]

[17] C'est possible de grouper et d'ordonner les candidats de sorte que pendant la négociation, une tentative soit faite alternativement pour chaque ensemble des candidats.

qu'elles peuvent avoir. Les préférences (<*Preference*>) sont ajoutées pour exprimer comment le directeur veut que la valeur évolue pendant la négociation.

Le langage employé pour exprimer les contraintes est dérivé du OCL, utilisé dans UML. Nous employons certaines de ses spécifications pour décrire les contraintes entre les paramètres ou pour naviguer dans un objet complexe entre les expressions des dépendances. Les contraintes sont employées pour indiquer, pour les attributs, quels sont les valeurs numériques possibles ou des ensembles discrets des valeurs (utilisation d'opérateur *enum*, par exemple : *self.quality enum {high, low}*). Dans le cas de valeurs numériques, les opérateurs sont exprimés sur les bases de la notation mathématique pour des comparaisons numériques. Une attribut peut avoir une (*self.cost <= 10k*) ou plusieurs contraintes (*(self.delay>=2) (self.delay<=4) and*).

La préférence et les relations sont utilisées par le système informatique pour évaluer une proposition ou pour générer une nouvelle offre. Ces contraintes sont personnelles à chaque atelier d'impression et elles ne sont pas connues par les autres participants dans la négociation. Nous exprimons la préférence comme une fonction d'utilité pour chaque attribut afin de calculer la valeur d'une offre, en utilisant par exemple le cadre de la théorie de l'utilité [Keeny et Raiffa'76]. Les relations (<Relation>) sont employées pour ordonner les ensembles d'attributs : les ensembles d'attributs seront considérés un après l'autre pendant la négociation.

Dependencies Definition Format Les dépendances entre les attributs peuvent être exprimées de manière suivante :

<DDF> ::= '(' <Exp>* ')'
<Exp> ::= '(' <val>* [<d Operator>]')'
<val> ::= <IssueName> | <Integer> | <Exp>
<d Operator> ::= <AlgebricOp> | <RelationalOp>
<AlgebricOp> ::= + | - | . | / | =
<RelationalOp> ::= if | < | <= | > | >= | == | / =

Les opérateurs algébriques expriment les opérations possibles sur les valeurs des différents attributs. Les opérateurs relationnels expriment les dépendances entre les attributs et/ou leurs valeurs.

Considérons un *Objet de Négociation* appelé *obj1* dans lequel le travail proposé devrait avoir une valeur pour la pénalité 50 fois plus grandes que le coût (*(self.penalty (self.cost 50k*

+)=)). Comme la valeur de l'attribut *penalty* est une valeur fixe (utilisation de l'opérateur =) et qui dépend uniquement de la valeur de l'attribut *cost*, la valeur de cet attribut est donc non négociable.

Dans un objet complexe (contenant plusieurs tâches) la négociation est faite en satisfaisant les contraintes définies dans chaque objet composant et aussi celles indiquées dans l'objet composé. Dans l'exemple précédent sont spécifié des dépendances entre les attributs du même objet. Considérons maintenant *obj0*, objet complexe composé du *obj1* et *obj2*. La dépendance suivante (obj1.cost (obj2.cost 2 *) >=) dans l'objet de négociation *obj0* déclare que la valeur de l'attribut *cost* pour la tâche définie dans *obj1* doit être plus grande que le double de l'attribut *cost* pour la tâche définie dans *obj2*. Cette approche offre une coordination entre les sous-tâches liées, ajustant la nouvelle proposition selon les états des sous-objets liés pris en considération.

Egalement, ces différentes relations spécifient visiblement des contraintes *non-déterministes* par le fait qu'elles délimitent un espace des possibles valeurs et pas une seule option (voir Tableau 6).

Ces types de relations fixent des contraintes qui réduisent l'espace dans lequel un agent doit chercher un accord pour la négociation dans laquelle il est impliqué. Les relations fixent des dépendances seulement entre les caractéristiques d'une seule négociation – entre les attributs de la tâche négociée ou entre les participants impliques dans la négociation. Si la négociation est conduite en prenant en compte seulement ces relations, l'accord final est pris sans considérer ni les propositions envoyées par les autres participants dans la négociation, ni les autres négociations dans lesquelles le participant courant est engagé. Donc, ces types de relations fixent des contraintes de coordination dans un monde fermé.

	Non-déterministe	**Déterministe**
Globale	$\vartheta_g \cap \vartheta_n$	$\vartheta_g \cap \vartheta_d$
Locale	$\vartheta_l \cap \vartheta_n$	$\vartheta_l \cap \vartheta_d$

Tableau 6

Ces différentes relations ne peuvent pas répondre directement aux questions comme quelle est la première proposition ? ou quelle est la proposition suivante si la première n'a pas été acceptée ? Par contre ces contraintes peuvent répondre à des questions comme : quand abandonner la négociation ? ou quand une négociation peut finir avec un contrat ?

La prise en compte des ces relations suppose l'existence d'un module de raisonnement afin d'ordonner les différentes options selon une fonction d'utilité spécifiée par une certaine stratégie.

Comme nous avons précisé c'est ne pas dans nos intentions des proposer des stratégies ou de modèles de raisonnement, mais de spécifier toutes les contraintes afin d'offrir à l'agent toutes les informations possibles pour conduire la négociation de n'importe quelle manière qu'il veut mais tout en concordance avec les contraintes de coordination.

Avec la structure *Objet de Négociation* nous avons offert la possibilité au Manager de préciser les contraintes sur la tâche à négocier, par la suite nous voulons aussi lui offrir la possibilité d'exprimer des contraintes globales sur la manière dont le processus de négociation doit se dérouler.

8.2. Coordination stratégique en monde ouvert

Afin d'exprimer des contraintes globales sur le processus de négociation et de formaliser les plans pour les différentes négociations dans lesquelles un agent peut se trouver, nous avons proposé une structure de données nommée *Cadre de Négociation*. Le *Cadre de Négociation* est un ensemble de paramètres non-négociables contraignant l'exécution de la négociation. Aucun de ces paramètres n'est négociable pendant le processus de négociation. On pourra envisager dans des développements futurs une étape de négociation de ceux-ci lors de la création de l'alliance ou pendant l'adhésion à l'alliance.

Le *Cadre de Négociation* assemble les conditions du Manager de l'atelier sur la manière dont une négociation doit être contrôlée. Cette structure permet au Manager d'indiquer l'espace logique des actions possibles pendant une négociation. Chaque *Objet de Négociation* a un *Cadre de Négociation* unique, mais un *Cadre de Négociation* peut couvrir plusieurs *Objet de Négociation*. Un *Cadre de Négociation* a la structure suivante :

<CN> ::='(' Context :duration <duration> :messages <integer>
:candidates <integer> :contractants <integer>
:strategies <strategy>* :tactics <tactic>*
:protocols '(':manager <protocolname> :negf <protocolname> ')'

Un Manager peut indiquer quelques paramètres globaux sur un processus de négociation: la durée (:*duration*), la limite supérieure des messages qui peuvent être échangés (:*messages*), la limite supérieure des participants considérés dans la négociation (:*candidate*), la limite supérieure des partenaires impliqués dans le contrat à la fin de la négociation (:*contractants*). Egalement, il peut définir la tactique (:*tactics*) qui doit être employée pour coordonner la négociation courante.

Le *Cadre de Négociation* contient également les protocoles (:*protocols*) à utiliser pendant les différentes négociations. Comme nous avons précisé dans l'état de l'art plusieurs types de protocoles d'interaction existent. Le nouvel aspect de la négociation dans ce projet est le mode de choisir les protocoles du type d'interaction. Dans [Bratu'01] nous avons distingué deux types d'interaction en fonction des participants : interaction entre deux agents implantant chacun d'organisations différentes et interaction entre le Manager (personne humaine) et l'agent de son organisation. Donc, les protocoles utilisés par l'infrastructure informatique d'un

atelier sont pour inter actionner d'une part avec le Manager (:manager) et d'autre part avec les autres ateliers (:negf). Les protocoles qui sont utilisés dans mon travail sont réalisés à partir des protocoles standards proposés par la FIPA. Pour la négociation entre deux agents nous avons proposé des protocoles basés sur les standards FIPA (Iterated Contract Net Interaction Protocol, Query Interaction Protocol).

Le *Cadre de Négociation* peut être vu comme une formalisation de différents degrés d'autonomie dans les décisions et les actions du système. Il est spécifique à un participant donné et il n'est pas communiqué aux autres participants dans le processus de négociation.

En fixant les tactiques, le Cadre de Négociation détaille aussi pour l'agent une formalisation du plan à suivre au cours des négociations.

Dans [Faratin'00], la tactique traite principalement la gestion de la négociation elle-même : les tactiques sont des fonctions qui créent une proposition basée sur un critère donné (temps, ressource ou comportement), ces propositions étant alors combinées en utilisant différents poids dans une stratégie de négociation. Dans notre approche *la tactique* impose des contraintes au-dessus du processus de la négociation. La tactique n'impose pas des contraintes directement sur une proposition mais au-dessus de l'ensemble entier de propositions et de contre-propositions échangées.

La tactique spécifie les schémas de coordination qui fixent des contraintes sur les possibilités de négocier et de synchroniser la propagation des alternatives à d'autres participants dans la négociation. Leur orientation de logiciel de type intergiciel fait que les tactiques traitent aussi les dimensions de l'interaction. Par exemple, une tactique déclarera que la sous-traitance d'une tâche doit être faite comme un seul bloc (voir schéma de coordination *AsABlock* – section 7.3.1.), c.-à-d. pas plus d'un participant pour l'exécution de la tâche. Une autre tactique (voir schéma de coordination *Split* – section 7.3.2.) pourrait permettre la sous-traitance d'une tâche dans plusieurs parties à des différents partenaires. La définition des parties et des partenaires est laissée à la négociation elle-même.

L'exécution de la tactique correspond à l'activation d'une combinaison des schémas de coordination mettant en place une coordination sur les différentes propositions échangées dans la négociation courante. Donc, avec les tactiques le Manager précise seulement quels sont les schémas de coordinations et par transition quels sont les types des règles qui devront gérer le processus de coordination des négociations, par contre les instanciations des paramètres des ces règles (*descriptions des négociations, graphes des négociations* et *temps de négociations* – voir section 5.2.) sont faites par l'agent en fonction de ses connaissances sur les négociations dans lesquelles il est impliqué.

Pour le moment, les tactiques précisent seulement quels sont les schémas de coordination qui doivent être activés pour gérer la coordination, mais sans préciser quand il est vraiment nécessaire de les activer. Un développement futur est de préciser aussi des dépendances entre les schémas de coordinations afin de proposer des stratégies de coordination.

Finalement, ces contraintes sont plus au niveau global de l'interaction en spécifiant des contraintes générales sur les négociations dans lesquelles un agent est impliqué. Ces contraintes sont également non-déterministes parce qu'elles ne précisent pas ponctuellement des contraintes pour chaque négociation, mais des contraintes générales qui serons par la suite instanciées de manières précises et dynamiques en fonction des négociations en cours de déroulement (voire Tableau 7).

	Non-déterministe	**Déterministe**
Globale	$\vartheta_g \cap \vartheta_n$	$\vartheta_g \cap \vartheta_d$
Locale	$\vartheta_l \cap \vartheta_n$	$\vartheta_l \cap \vartheta_d$

Tableau 7

Donc, le modèle de coordination stratégique que nous avons proposé permet de définir **le sujet de la négociation** (le but de la négociation est spécifié dans la structure nommée *Objet de Négociation*) d'une part, et d'autre part **comment la négociation doit se dérouler** (contraintes spécifiées dans la structure nommée *Cadre de Négociation,* qui décrit de manière générale le processus d'interaction à suivre pour conduire la négociation).

Avec cette approche nous proposons non seulement un modèle de structuration des règles de dépendance non-déterministes, mais également nous faisons les premiers pas pour la mise en place d'un processus de décision structuré en spécifiant les informations prises en compte et le niveau où ces informations influence une décision. Par exemple, on doit pouvoir prendre des décisions dans une négociation sans tenir compte des autres négociations en cours, inversement on doit pouvoir prendre des décisions dans une négociation en cohérence avec les autres négociations en cours.

8.3. Conclusions

Typiquement, dans le processus de négociation que nous modélisons, plusieurs parties sont impliquées. Chacune d'entre elles a des intérêts particuliers sur les différentes outsourced ou insourced[18] tâches en cours de négociation. Ces différents intérêts font que dans une négociation, chaque participant peut avoir des critères, des contraintes et des préférences différentes.

La négociation est gérée comme si la tâche sous la négociation était sur une table, visible partiellement ou totalement pour chaque agent participant à la négociation. Chaque agent agit de manière opportuniste aux changements de son environnement (coupure d'une machine, de nouvelles offres des clients...) et à l'évolution des propositions faites par les autres dans la négociation. Dans la même négociation toutes les propositions peuvent progresser en parallèle et de manière asynchrone aussi longtemps qu'aucun engagement n'est pas pris. Ces différentes propositions agissent en tant que contraintes qui réduisent progressivement les différents centres d'intérêt pour chaque participant, de sorte qu'un accord soit conclu dans l'intersection, si elle existe.

Dans ce contexte de modélisation du processus de négociation nous abordons la coordination des négociations comme une structuration de deux modèles: *(i) une partie générique* qui peut répartir et distribuer les différentes relations de dépendances d'une négociation à des instances des différents schémas de coordination, et *(ii) une partie stratégique* qui peut mettre en place différentes stratégies de négociation tout en spécifiant les informations prises en compte. La coordination stratégique identifie le processus de négociation par l'objet à négocier, par les participants et leurs rôles, par les messages et les protocoles utilisés. La coordination générique identifie les abstractions liées au management des interactions entre activités.

Avec cette approche nous proposons également un modèle de *coordination locale* qui peut gérer une négociation sans tenir compte des autres négociations en cours (*coordination en monde fermé*), et inversement un modèle de *coordination global* qui peut gérer une négociation en cohérence avec les autres négociations en cours (*coordination en monde ouvert*).

---

[18] On appelle une négociation de type outsourcing, la négociation vue par le participant qui décide de proposer une de ses tâches dans l'alliance.

On appelle une négociation de type insourcing, la négociation vue par le participant qui décide de contracter une des tâches proposées dans l'alliance.

Finalement, cette décomposition du processus de coordination en deux parties, *coordination stratégique* et *coordination générique*, nous permet de délimiter des contraintes de *coordination non-déterministes* qui nécessitent des mécanismes de décision pour les manipuler et les évaluer et des contraintes de *coordination déterministes* qui nécessitent seulement des mécanismes de coordination pour les manipuler. Cette approche nous a permis d'identifier quels sont les contraintes de coordination qui délimitent le processus de coordination du processus de décision et du processus de communication.

Cette délimitation fait que chacun des processus peut entre implémenté par des mécanismes informatiques différents : agent, services et intergiciel. L'architecture informatique proposée par la suite spécifie comment *1)* en partant d'une description d'une tâche d'impression faite par des structures de type Objet de Négociation et Cadre de Négociation, le processus de décision élabore, au niveau agent, une proposition de négociation et *2)* qui par la suite, au niveau services de coordination, celle-la est transformée dans des actions de manipulation d'un graphe et elle est validée ou non par le processus de coordination, afin *3)* d'être finalement envoyé par le processus de communication implémenté au niveau intergiciel.

La mise en œuvre de l'approche proposée sera détaillée dans la partie suivante du cette mémoire de thèse.

IV. Mise en œuvre de la coordination de négociations

Dans cette partie nous commençons par la présentation de l'infrastructure de négociation mise en place afin de soutenir ***le processus de décision*** et ***le processus de communication*** de notre modèle de négociation (Chapitre 9). Nous poursuivons dans le chapitre 10 par la présentation des mécanismes utilisés pour modéliser ***le processus de coordination*** ainsi que les différentes expérimentations mises en place. Le troisième chapitre (voire chapitre 11) de cette partie présente l'implémentation du logiciel satisfaisant la spécification du processus de négociation global venant se dérouler sur l'architecture et les mécanismes proposés..

9. Infrastructure de Négociation

La suite de ce chapitre commence avec une rapide présentation du projet E-Alliance avec ses différentes contributions scientifiques et avec une présentation de l'ensemble de l'architecture afin de détailler par la suite l'architecture et les mécanismes proposés pour la coordination des négociations.

9.1. Contexte: E-Alliance

Le travail mené dans le cadre de cette thèse s'insère dans le projet E-Alliance. Le projet E-Alliance a été soutenu financièrement par un programme thématique prioritaire de la Région Rhône-Alpes et il a été conduit en collaboration par trois laboratoires : 1) le laboratoire SMA de l'Ecole des Mines de Saint-Étienne, 2) le Centre Européen de Recherche Xerox et 3) le laboratoire LISTIC de l'Université de Savoie**.**

9.1.1. Objectifs

L'objectif du projet E-Alliance a été de fournir un outil informatique d'assistance aux activités de *négociation* et de gestion de contrats entre différents participants d'une *alliance inter-organisationnelle*. L'infrastructure logicielle développée vise à automatiser l'ensemble des échanges d'informations ainsi que la conduite des procédures collaboratives entre les différents participants de l'alliance, laissant à chacun des utilisateurs (gestionnaires d'une organisation, participants de l'alliance) le pouvoir de décision au sein de chacune des étapes de la procédure. Ainsi, le projet a visé de définir et de mettre en place un environnement logiciel qui est à la fois :

- **Ouvert** : permettant l'entrée et la sortie dynamiques des participants dans/de l'alliance;
- **Non intrusif** : permettant/préservant l'autonomie des participants quant à la fourniture ou l'utilisation des services ;
- **Centré négociation** : la négociation étant un moyen privilégié par lequel passe toute fourniture et toute utilisation ou réalisation des services au sein de l'alliance.

Partant de ces trois caractéristiques, au sein du projet E-Alliance, ont été traités deux problèmes principaux qui se posent au sein des alliances inter-organisationnelle :

1. le support informatique pour la gestion du cycle de vie des *alliances* mises en place au sein de ces groupements inter-organisationnelle, avec ce que cela implique comme support pour la mise en place des services de diffusion d'informations, d'authentification, ainsi que des services qui supervisent et contrôlent des comportements,

2. le support informatique pour les comportements collaboratif incluant la *mise en place et la gestion des négociations et des contrats* avec ce que cela implique comme support pour la coordination, la négociation, la prise en compte des autres participants et la gestion des engagements pris au sein de l'alliance par chacun des participants, en tenant compte des aléas de production et des pannes.

Dans le cadre du projet, les participants des alliances auxquelles nous nous sommes intéressées, sont des organisations qui exercent *a priori* le même métier, et peuvent donc être en compétition sur les mêmes marchés. Ces participants se sont constitués en alliance afin d'exploiter leurs complémentarités pour mieux s'adapter à la demande de ce marché. Les activités suivantes peuvent se dérouler à ce niveau, mettant en jeu des opérations complexes au niveau de chaque participant :

– de planifier de manière à optimiser l'utilisation des ressources (matérielles ou humaines) nécessaires à leur réalisation ;

– de mettre en oeuvre de façon coordonnée ;

– de surveiller pour réagir à des événements susceptibles à perturber le déroulement global du processus (panne d'un composant matériel, épuisement d'une ressource consommable comme le papier dans un atelier d'impression) ;

– de comptabiliser en vue de facturer le client (utilisateur des services) ou d'établir des statistiques.

9.1.2. Domaine d'application

Dans ce projet, le cas d'application auquel nous nous sommes intéressés concerne le cas des *alliances des ateliers d'impression*. Dans ce type d'alliance, chaque participant (atelier) est totalement *autonome* au sens qu'il est responsable de sa propre charge de travail et de la gestion de ses ressources. Il veut conserver ses propres outils pour gérer ses contrats, ses ressources, son plan de charge, ses clients, tout en cherchant à coopérer avec les autres via un

échange *minimal* d'informations pour réaliser des tâches globales d'impression. Cet échange est minimal au sens que l'atelier contrôle et sélectionne le type des informations échangées. L'infrastructure logicielle définie au sein de ce projet nous a permis de préserver cette autonomie de chaque participant tout en leur fournissant des règles de coordination communes en vue de mieux s'adapter à la demande des participants utilisateurs des services (clients). Ces règles visent à faciliter le fonctionnement de l'alliance et plus particulièrement la *négociation* entre les différents participants afin de satisfaire de façon collaborative des demandes qui ne peuvent être traitées par un seul participant individuellement. Ce cas d'application nous permet de définir les éléments d'une infrastructure appropriée pour notre projet, de plus que les résultats dans ce cas de "marché électronique" décentralisé sont aisément transportables à d'autres domaines de l'industrie des services, aux applications du commerce électronique et plus particulièrement aux applications de "business to business" (cas de sous-traitances entre entreprises).

Dans le cas spécifique des alliances des ateliers d'impression offrant différents services liés au traitement des documents, il y a des opérations complexes (appelées *jobs* par la suite) que ces ateliers sont amenés à mettre en oeuvre, concernent : (i) l'assemblage électronique du document, avec les problèmes de conversion à partir de formats multiples ; (ii) l'impression et l'assemblage physique du document imprimé, en bloc ou par morceaux, avec des spécifications de qualité et de types variables ; (iii) la distribution du document imprimé vers ses destinataires finaux.

De telles alliances existent actuellement et la Fig. 22 montre un exemple d'interaction entre les partenaires d'une alliance composée de trois ateliers d'impression qui collaborent pour mieux répondre aux appels d'offres provenant des clients extérieurs.

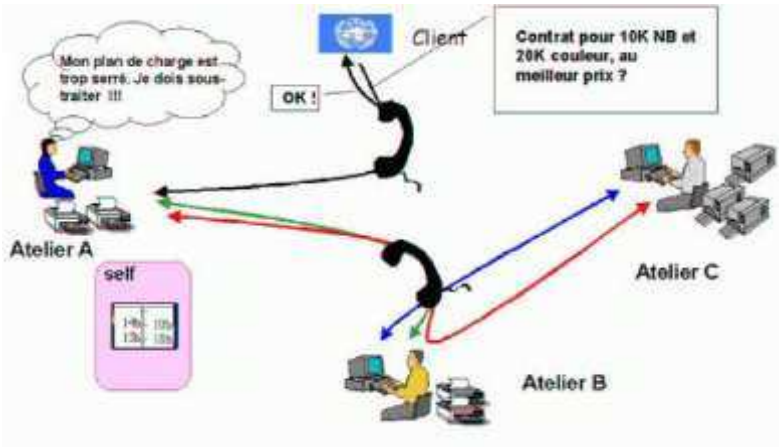

Figure 22 – Contexte du projet E-Alliance : alliances d'ateliers d'impression.

Dans notre exemple, lorsqu'un client extérieur comme l'ONU, contacte un des ateliers de cette alliance pour réaliser des importants travaux d'impression, cet atelier, s'il ne peut (ou ne veut pas) répondre seul à une telle demande, entame une série d'échanges avec un ou plusieurs partenaires de l'alliance pour sous-traiter tout ou une partie du job, et en suite il va se coordonner pour l'exécution des opérations complexes à réaliser. Le problème est qu'actuellement, la majeure partie de ce travail de coordination entre les partenaires se déroule sans un véritable support informatique ou, s'il existe, il impose une uniformisation et une refonte des systèmes informatiques de chacun des partenaires de l'alliance et de facto, une perte d'indépendance de ceux-ci.

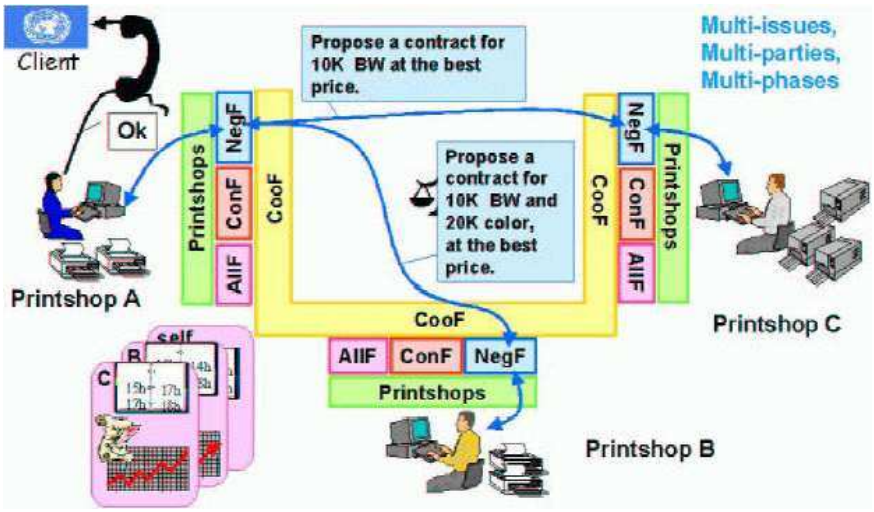

Figure 23 – Objectif du projet E-Alliance : développement d'une infrastructure logicielle pour l'aide à la négociation et à la coordination des alliances inter-organisationnelle.

Dans ce contexte, certaines des objectifs précédents se déclinent en contributions scientifiques proposées par cette thèse, comme : *1)* la définition du(des) *processus de négociation* disponible(s) entre les membres de l'alliance pour établir les différents contrats à assurer; *2)* intégration des ces processus dans un processus de coordination montré sur la Fig. 23. Egalement, ces différents processus satisfont les exigences suivantes : (i) *être flexibles* afin de s'adapter aux évolutions de l'alliance et de l'environnement courant, tout en permettant le découpage/regroupement des travaux, la délégation/adoption des travaux, le choix des partenaires, (ii) *se conformer aux règles* régissant le fonctionnement des membres de l'alliance, (iii) tirer partie des différentes informations représentées et de l'évolution de l'environnement.

9.1.3. Infrastructure E-Alliance

Le modèle de l'infrastructure E-Alliance (cf. Fig. 24) que nous avons défini, propose une structuration des différentes fonctionnalités répondant aux objectifs décrits en introduction selon trois couches logicielles présentes sur chaque site appartenant à une *E-alliance*. Plus précisément, chaque site est équipé *i)* d'une interface facilitant l'interaction avec les acteurs externes au système (*Manager* et *Planning*), *ii)* d'un ensemble de "Facility" gérant les alliances (*AllF*), les contrats (*ConF*) et les négociations (*NegF*) ; et *iii)* ces logiciels s'appuient sur un intergiciel de coordination (*CooF*), couche commune à l'ensemble des sites. Ces différents logiciels sont, a priori, génériques. C'est la raison pour laquelle une couche dédiée à l'application permet d'utiliser l'infrastructure E-Alliance dans différents contextes. Les "Facility" s'échangent différentes informations telles que les contrats, les informations sur chacun des partenaires (représentation des autres), les règles communes à l'alliance, etc.

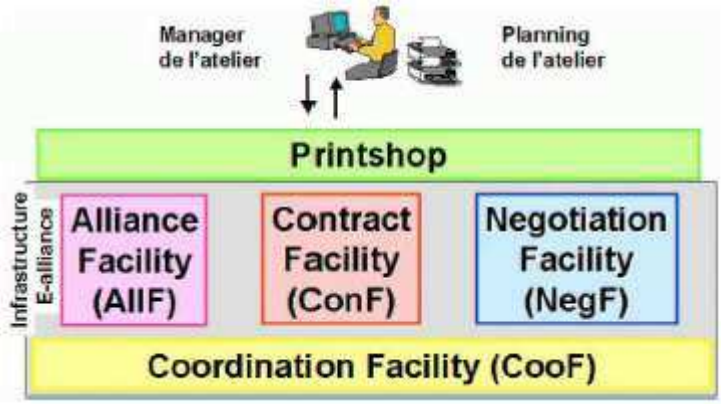

Figure 24 – Infrastructure E-Alliance avec les acteurs externes au système : Manager et Planning.

Chaque **Manager** d'atelier d'impression, ainsi que son système de Planning, sur leur propre site interagissent avec les autres partenaires de l'alliance via un ensemble des services adaptés au domaine de l'application via la couche "Printshop" (cf. fig. 21). Ces services permettent de sous-traiter et de gérer l'exécution de différents travaux d'impression entre plusieurs partenaires sous le contrôle de l'alliance à laquelle ils appartiennent.

– Le **Manager**, un ou plusieurs être humains, est responsable de la prise de décision tout au long de la sous-traitance ;

– Le **Planning** gère les informations sur l'état des ressources (disponibilité des opérateurs, des machines, des ressources "consommables" — papier, encre, etc.), effectue des

réservations de ces ressources, et, éventuellement, permet une forme plus ou moins avancée de planification exploratoire.

La mise en oeuvre des services est réalisée via un ensemble de logiciels appelés "Facility" : gestion de l'alliance **AllF**, gestion de contrats **ConF**, gestion des négociations **NegF**.

L'AllF "Alliance Facility" assure la mise en oeuvre des règles de fonctionnement globales de l'alliance concernant ses différents membres. Il doit donc être capable de contrôler et superviser le comportement d'un ensemble d'entités autonomes distribuées, dans un système ouvert selon ces différentes règles.

Brièvement, afin de maintenir une cohérence au niveau de l'état d'une alliance, les différentes AllFs supervisent les fonctionnements de leurs NegFs et ConFs et vérifient que les règles de l'alliance sont respectées. Chaque AllF représente et gère également l'historique des différents événements qui se produisent dans le système. Egalement, si les événements ont des conséquences sur les négociations ou les contrats en cours d'exécution, l'AllF interagit aussi avec la facilité concernée.

Le ConF "Contract Facility" assure l'exécution correcte de *contrats* exprimant l'entrelacement de processus administratifs et de processus de production (workflow business process) inter-organisationnels entre les partenaires participants aux "jobs" sous-traités.

L'exécution du contrat consiste dans une exécution distribuée du processus de production sur les ConFs des différents participants. Pendant l'exécution du contrat et en fonction des certains événements (i.e. rupture de contrat de la part d'un partenaire) le ConF interagit avec les autres composants de l'infrastructure (AllF, NegF, CooF) et même avec le Manager.

Le NegF "Negotiation Facility" est le maître d'œuvre de la sous-traitance au sein de l'alliance. La principale fonctionnalité du NegF est d'assister le Manager dans le processus de sous-traitance d'une tâche (job) dans l'alliance mais aussi dans la coordination de plusieurs sous-traitances concurrentes. Dont, il gère des négociations concurrentes entre plusieurs partenaires autonomes, en l'interaction avec le Manager et le Planning de son atelier d'impression. Ces trois "Facility" sont en interaction tout au long du fonctionnement de E-Alliance. C'est le cas, par exemple, lorsque l'alliance évolue (i.e. retrait d'un membre), l'AllF interagit avec les NegF et les ConF pour fournir aux Managers concernés, les informations

nécessaires à la maintenance de la cohérence globale du système en terme de négociations et/ou contrats en cours, afin qu'ils puissent prendre des décisions concernant la suite des opérations.

Le NegF est également en interaction permanente avec l'intergiciel CooF qui lui offre un support générique au processus de diffusion et de synchronisation des propositions échangées au cours de la négociation.

Couche intergicielle de coordination et de communication : AllF, ConF, NegF s'appuient sur une infrastructure de coordination et de communication **CooF** (Coordination Facility). Alors que les deux autres couches sont dupliquées sur chacun des sites, la couche intergicielle CooF est commune à l'ensemble des sites sur lesquels les partenaires de l'alliance opèrent. Celle-ci est capable de coordonner plusieurs négociations concurrentes en mettant en œuvre notamment une diffusion d'alternatives entre plusieurs participants à une négociation (en utilisant le modèle *Xplore*). De plus, elle permet d'exécuter et de coordonner plusieurs processus inter-organisationnels (workflow).

9.2. Vision d'ensemble de l'infrastructure de négociation

Dans le cadre de cette thèse, nous nous intéressons au processus de négociation et plus particulièrement au processus de coordination des négociations. Etant données les caractéristiques du projet et nos objectifs, nous avons travaillé plus particulièrement sur la **« facility » NegF** et la couche intergicielle **CooF** afin de fournir un ensemble de services pour assister le **Manager** dans la gestion des *processus de négociation.*

Notre objectif a été de développer une architecture logicielle intégrant les trois couches de l'infrastructure E-Alliance (une couche d'interaction avec les acteurs extérieurs, une couche « facilities » pour la gestion d'actions dans l'alliance et une couche intergicielle offrant les outils génériques de communication et de transaction) et qui sera capable de gérer et de coordonner toutes les négociations dans lesquelles un atelier d'impression est impliqué dans le cadre d'une alliance. Ainsi, en partant de l'axe de modélisation de la négociation Manager – NegF – CooF, notre implémentation sera structurée selon quatre couches (cf. Fig. 25) abordant chacune un aspect particulier de ce processus général :

- **Manager** : c'est une interface qui permet la prise en compte de l'intervention humaine pour la description des besoins ou au ***niveau décisionnel*** ;
- **NegF** : mécanismes d'assistance dans le processus d'interaction dans l'alliance, notamment dans l'élaboration et la prise des décisions automatisées dans une négociation. Ainsi *NegF* modélise les mécanismes composant le ***processus de décision*** et ***le modèle stratégique de coordination***;
- **Services** : ensemble des mécanismes génériques de coordination d'interactions. Ainsi *les services* représentent l'implémentation du ***processus de coordination*** et notamment du ***modèle générique de coordination***;
- **CooF—Xplore** : mécanismes de diffusion et de synchronisation d'alternatives élaborées pendant un processus de négociation. Ainsi *l'intergicielle* modélise notamment le ***processus de communication***.

Dans la présentation de notre modèle de coordination nous avons précisé que l'exécution de la tactique correspond à l'activation d'une combinaison de plusieurs *schémas de coordination.* En dévoilant l'architecture, nous précisons que l'exécution de la tactique correspond à l'activation d'une combinaison des instances des *services* mise en application au-dessus du CooF. En faisant la correspondance entre les deux notions et sans entrer en détail

pour le moment, nous pouvons préciser que chaque *instance de service* représente la mise en œuvre d'une *description de négociation*. Ainsi chaque service a deux principales fonctionnalités: *i)* de construire une vue locale de la négociation globale selon le comportement particulier d'une *description de négociation*, selon notre modèle ce comportement est complètement décrit par un *Program Formulae* faisant partie d'un *schéma de coordination* particulier, et *ii)* de traduire les décisions de négociation et les différents messages de négociation comme modifications sur l'ensemble des graphes de négociation Xplore en utilisant les verbes du protocole utilisé dans le CooF.

Le principe est le suivant: un Manager d'atelier d'impression décide de sous-traiter un job à des partenaires de l'alliance, soit pour libérer des ressources en vue de satisfaire une autre requête plus importante, soit pour accepter une requête qu'il ne peut réaliser seul. Le Manager initie une sous-traitance en communiquant au NegF les propriétés de l'Objet de Négociation et du Cadre de Négociation (voir chapitre 8.).

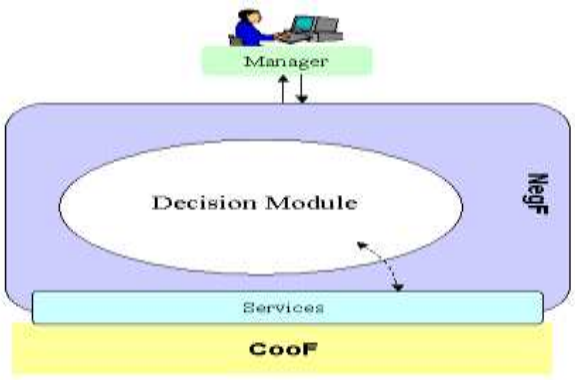

Figure 25 – Architecture du système de négociation

Les différents NegFs, les différents services et le CooF se chargent ensuite de piloter la négociation pour qu'elle débouche sur un contrat, tout en respectant les contraintes définies par le Manager

L'architecture du NegF permet de gérer et de piloter plusieurs négociations en parallèle impliquant des objets de négociation ou des participants différents et le CooF gère une négociation comme un processus de synchronisation de plusieurs copies partielles d'un graphe bicolore. Entre ces deux, la couche des *services* génériques offre la possibilité d'instancier pour une négociation courante un ou plusieurs services. Chaque instance gère, en fonction de ses fonctionnalités, une image locale de la négociation courante comme un graphe bicolore

pour le compte de l'agent auquel il est rattaché. Ces instances interagissent avec l'intergiciel via les différentes primitives Xplore afin de synchroniser les différents graphes bicolores. L'aspect transactionnel est mis en place par le script de coordination (nommé *tactique* – voir section 9.3.) qui met en liaison les différents services impliqués dans une négociation.

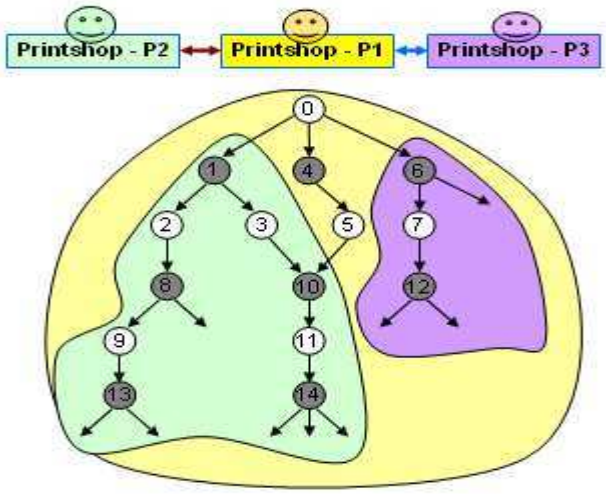

Figure 26 Exemple des graphes de négociation Xplore

Ainsi, dans l'exemple (cf. Fig. 26), une négociation simple impliquant un donneur d'ordre potentiel (l'atelier P1) et différents fournisseurs possibles (l'atelier P2 et l'atelier P3), nécessitera une instance d'un service nommé Outsrc qui gère la négociation pour le donneur d'ordre et deux instances du service nommé Insrc pour chaque fournisseur potentiel. Dans l'exemple on peut voir que l'atelier P1, par le biais de son service Outsrc, voit la totalité du graphe de négociation et que les autres deux ateliers, par le biais de leurs services Insrc, voient seulement une partie de la négociation. Une tactique telle que *outsrc(job) @ insrc(job)* → *contract(job)* permet d'exprimer qu'un contrat est établi si une acceptation sur le même objet *job* est obtenue dans deux graphes gérés par les instances des services Outsrc et Insrc. Donc, chaque instance de service a une vue particulière sur le graphe de négociation en fonction du type de service auquel elle est associée et donc du comportement qu'elle modélise et aussi en fonction de l'endroit où elle a été introduite dans le graphe. Nous reviendrons en détail sur la définition et au développement des services dans la section suivante.

Dans les sections suivantes nous allons présenter plus en détail les mécanismes proposés au niveau de l'intergiciel CooF et de la « facility » NegF afin d'illustrer par la suite comment ces mécanismes sont utilisés pour implémenter les modèles de négociation, de coordination générique et de coordination stratégique que nous avons proposés dans les sections précédentes.

9.3. Processus de communication

Le CooF est une couche intergicielle de coordination orientée négociation qui soutient les différents processus fournis par les trois « facilities » de la deuxième couche de l'infrastructure E-Alliance. Le CooF est une extension de l'intergiciel CLF [Andreoli et al.'99] notamment pour enrichir ses possibilités de soutenir différents types de négociation [Andreoli et Castellani'01].

9.3.1. Principe : CLF

Dans CLF, tous les composants sont regardés comme des ressources. Cette notion de ressource est extrêmement polymorphe, car elle s'applique aux ressources "réelles" traditionnelles, par exemple un copieur, aussi bien qu'aux entités plus virtuelles, comme un processus qui a lieu en dehors du système computationel, par exemple une tâche d'imprimerie. Les composants CLF présentent leurs ressources seulement à travers leurs interfaces. L'interface d'un composant CLF définit des services abstraits par lesquels des opérations sur des ressources sont rendues possibles. L'interaction avec un composant CLF à travers un des services de son interface suit un protocole spécifique qui accroît la structure fondamentale du composant comme directeur de ressource. Le protocole est défini par huit " verbes d'interaction", semblables aux actes de communication et qui ont une signification en termes de manipulations de ressource. Cette approche considère de manière systématique les composants comme des directeurs de ressources et permet la coordination de ces composants avec des ensembles des règles groupées dans des scripts qui cachent complètement le protocole de communication et expriment directement l'interaction par des manipulations de ressources.

Des composants CLF spécifiques, appelés *les coordonnateurs* (*coordinators*), manœuvrent les scripts comme des ressources. Quand une ressource de type script est insérée dans un coordonnateur, les manipulations abstraites des ressources qu'elle indique sont traduites en invocations du protocole relatif aux différents composants à coordonner. Ainsi, les coordonnateurs peuvent être considérés en tant que serveurs génériques et le côté client de l'intergicielle CLF peut être exprimé comme un ensemble de scripts.

Par exemple, un atelier d'impression dans notre scénario d'alliance sera représenté par un composant CLF offrant des services outsourcing et insourcing pour les différentes tâches

d'impression [Andreoli et al.'00]. Les ressources tenues par ce composant sont les décisions respectivement d'outsourcing ou insourcing d'une tâche. Un mécanisme d'outsourcing simple est mis en application dans CLF par la règle suivante :

outsrc (job) @ partner (dest, job) @ insrc(dest, job, offer) @ accept(dest, job, offer) <>- transfer (dest, job, offer)

Afin d'être déclenchée, cette règle exige l'ensemble suivant des ressources:

- une tâche d'impression « *job* » afin d'être proposée (modélisée par la token **outsrc** et qui est proposée par un composant de type atelier d'impression) ;

- une entrée de page jaune liant la tâche proposée et un partenaire potentiel « *dest* » qui a potentiellement la capacité d'exécuter le travail demandé (modélisée par le token **partner** et qui est proposée par un composant de type Yellow Pages) ;

- une proposition « *offre* » pour la tâche proposée faite par ce partenaire potentiel (modélisée par le token **insrc**; et qui est proposée par le nouveau participant déterminé dynamiquement par les ressources précédentes); et

- une acceptation de cette offre (modélisée par le token **accept** et qui est proposée par le même partenaire qui a lancé le outsourcing).

Toutes ces ressources sont recherchées dans leurs composants respectifs en utilisant les possibilités de recherche du protocole CLF. La phase de recherche comprend la construction asynchrone d'un arbre de recherche codant toutes les combinaisons possibles des ressources du type ci-dessus. Quand une branche dans l'arbre est complète, i.e. une ressource a été assignée à chacun des tokens de la partie gauche, dans ce cas le protocole CLF peut effectuer la consommation atomique des ressources correspondantes et il peut informer le succès de la transaction par l'insertion des ressources indiquées dans la partie droite de la règle (dans notre exemple le token **transfert**).

Le protocole de CLF laisse exécuter des multiples recherches dépendantes sur les ressources tenues par plusieurs composants, mais il permet seulement une propagation "*unidirectionnelle*" de l'information entre les composants impliqués. En effet, chaque réponse envoyée par un serveur dans la phase de recherche doit être une spécification "complète" de la ressource réelle (par exemple une tâche donnée) à employer dans la phase d'exécution de la règle. Il n'y a aucun moyen pour que le serveur envoie des réponses "partielles" décrivant certaines caractéristiques communes d'un ensemble des ressources potentielles, laissant ensuite

le client raffiner cet ensemble afin de converger (par des améliorations successives) vers la ressource à employer dans une exécution éventuelle. Ainsi, dans l'exemple ci-dessus, la recherche de la ressource **insrc** créera des branches dans l'arbre de recherche, chaque branche contenant une description complète de l'offre, que l'outsourceur peut accepter ou rejeter en tant que telle, en créant ou pas la ressource **accept**, sans d'autres possibilités de raffiner l'offre.

9.3.2. Protocole XPLORE

L'infrastructure du CooF dans E-Alliance est une extension du CLF qui traite le problème mentionné ci-dessus en soutenant une négociation *multi-participant*, *multi-phase*, *multidirectionnelle*, *multi-attribut* dans la phase de recherche de l'exécution des scripts de coordination. L'arbre de recherche de l'exécution d'une règle de coordination devient ainsi un "graphe de négociation", qui capture les dépendances entre les propositions de négociation. Ainsi cette extension offre les fonctionnalités afin de gérer la synchronisation d'un processus de négociation ayant les caractéristiques imposées par notre modèle. Cette extension porte le nome de **Xplore**

Au niveau de l'intergiciel Xplore un processus de négociation est représenté comme un graphe bicolore. L'évolution d'une négociation en termes des propositions et de contre-propositions envoyées correspond donc à une construction d'un graphe noire-blanc. Dans ce graphe les nœuds blancs contiennent les informations relatives aux attributs de la tâche négociée. Ce type de nœud est caractérisé par un *paramètre* et un ensemble des *attributs* avec leurs valeurs associées décrites par un ensemble de *termes*. Le *paramètre* est la tâche à négocier et qui est décrite, à un moment donné, par les *attributs* et les *termes* négociés selon l'état de la négociation dans le nœud blanc respectif. Dans la Fig. 23 le nœud 2 est représenté par le paramètre *Job* et les attributs et les termes instanciés *cost>100* et *delay<3*.

Au cours d'une négociation plusieurs propositions et contre-propositions peuvent être envoyées et donc plusieurs alternatives pour continuer la négociation existent. Ces alternatives sont modélisées par les nœuds noirs qui sont donc des nœuds de choix exclusifs entre plusieurs nœuds blancs. La notion de choix introduite par le nœud noir impose une restriction au niveau de la construction du graphe Xplore : les sous-graphes qui ont une racine noire commune ne doivent pas avoir aucun autre nœud en commun.

Dans la Fig. 27 un exemple de graphe Xplore est présenté - ici le paramètre unique de la négociation est un travail d'impression *Job* et un des aspects peut être le prix (*cost*) qui peut

assumer une gamme des valeurs possibles. Différentes branches de négociation peuvent être créées dans le graphe Xplore, sur l'initiative des différents participants impliqués dans la négociation, afin d'explorer des solutions de rechange en termes, par exemple, de prix (un prix au-dessous de 100 Euros ou plus de 100 Euros). Les participants peuvent alors continuer la négociation dans chaque branche en spécifiant par exemple différents *delay* (par exemple prix au-dessus de 100 Euros mais dans un temps plus court - le *delay* de 1 jour, ou au-dessous de 100 Euros mais dans un temps plus long - le *delay* de 3 jours). L'interaction indiquant le *delay* (1 jour ou 3 jours) se produirait dans une phase de négociation d'une branche ou une autre, créée par l'interaction au sujet du prix.

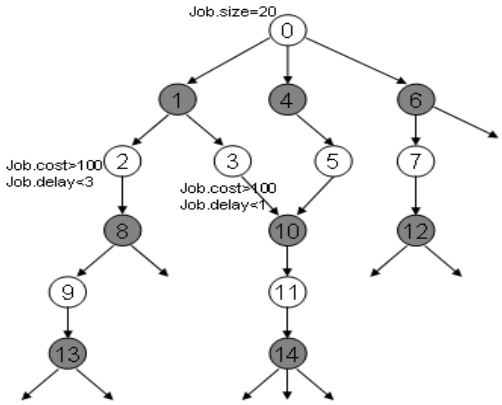

Figure 27.: Graphe Xplore

Comme le processus de négociation est un processus distribué entre les participants impliqués, l'intergiciel Xplore modélise de la même manière la construction du graphe d'une négociation. Chaque participant a sa propre copie (partielle) du graphe Xplore de négociation et il prend des décisions et il agit en manipulant seulement sa copie. L'intergiciel se charge de maintenir pour chaque participant (atelier) l'image de type graphe de ces négociations et de synchroniser cette image avec celle des partenaires concernés. Afin de modéliser cette synchronisation l'intergiciel propose un ensemble des six opérations (connect, open, quit, assert, request and ready) pour agir et manipuler les graphes. Donc, les partenaires ne communiquent pas directement, mais seulement via les opérations (verbes) proposées que chacun d'entre eux utilise sur ses copies de graphes de négociation et qui seront propagés par le CooF. Ces verbes sont :

- **Connect**(*n,m*): informe un component qu'il est impliqué dans une négociation qui a la racine *n* et que le paramètre (tâche) négocié par ce component sera identifié selon le mapping *m*. Cette identification est nécessaire pour faire

clairement la distinction entre les différents graphes Xplore et les tâches négociées au niveau de chaque component impliqué dans la même négociation.

Sans expliquer en détail tous les pas la Fig. 28 montre comment l'atelier d'impression P1 commence une négociation et il invite dans cette négociation deux autres ateliers P2 et P3. En utilisant le verbe connect dans deux nœuds différents les participants P2 et P3 se font introduits par le P1 dans la négociation courante et en fonction de leurs nœuds racine ils auront des images différentes sur la même négociation.

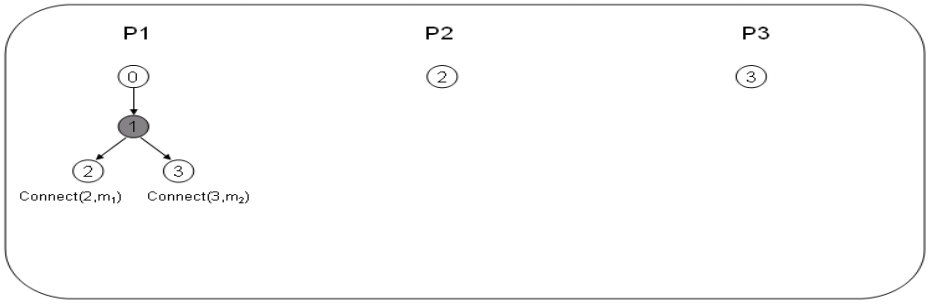

Figure 28. Verbe connect

- **Open**($n,n_1,\ldots, n_p$): crée un nouveau nœud n dans le graphe Xplore ayant comme parents les nœuds $n_1,\ldots, n_p$. Tous les nœuds parents $n_1,\ldots, n_p$ (s'ils existent) doivent être de la même couleur et le nœud n sera de la couleur opposée :
  - si n est noir, alors p doit être au moins 1, et n représente une décision de négociation prise dans une phase de négociation qui est le résultat de la fusion des phases représentées par ses parents blancs pour $p>1$, ou qui est directement représenté par un seul parent blanc si $p = 1$.
  - si n est blanc, alors p doit être au plus 1, et n représente une phase de négociation qui est une alternative dans la décision représentée par son seul parent noir si p = 1, ou une première phase (vide) de négociation si p = 0.
- **Assert**(n, v, a, t) : exprime la décision que, dans la phase de négociation représentée par le nœud n, la valeur de la variable v doit avoir la propriété exprimée par le terme t concernant l'aspect a.

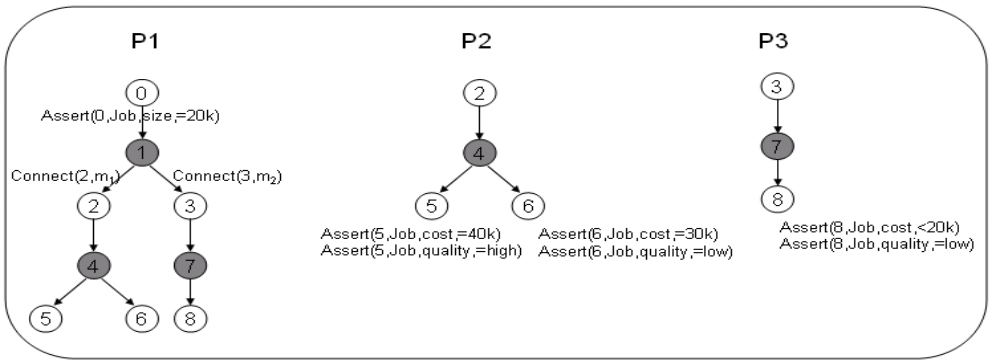

Figure 29. Verbe assert

Continuant l'exemple dans la Fig. 29, nous supposons que le participant P1 veut négocier une tâche des dimensions 20k et il veut faire la même proposition aux deux participants P2 et P3. Comme le nœud 0 est la racine du graphe vue par P1, il peut utiliser la propriété d'héritage du graphe afin de faire la même proposition aux deux participants. Le participant P1 utilise le verbe ***assert****(0,Job,size,=20k)* afin d'introduire dans la racine du graphe de négociation l'attribut *size* avec la valeur *=20k.*

Par la suite, nous supposons que le participant P2 veut faire deux propositions distinctes en ouvrant en premier un nœud noir - ***open****(4,2)* - et en suite à partir de ce nœud noir il fait ses propositions en ouvrant des nœuds blancs - ***open****(5,4)* et ***open****(6,4)* - et en spécifiant ses propositions - ***assert****(5,Job,cost,=40k)* ***assert*** *(5,Job,quality,=high)* et ***assert****(6,Job,cost,=30k)* ***assert*** *(6,Job,quality,=low)* -. En utilisant les mêmes verbes, le participant P3 fait aussi une proposition. Par la suite, l'intergiciel qui gère la communication synchronise les opérations faites par P2 et P3 sur le graphe du P1 afin que l'atelier P1 voit aussi les trois propositions faites par P2 et P3.

- **Request***(n, v, a)* : exprime qu'afin de pouvoir continuer la négociation, un participant fait connaître aux autres dans le nœud blanc *n,* le fait qu'il attende des assertions sur l'aspect *a* de la variable de décision *v*.

Dans la Fig. 30 le participant P1 annonce aux deux autres participants qu'il attend aussi une proposition pour le *delay*, cette annonce est faite avec le verbe ***request****(0,Job,delay)*.

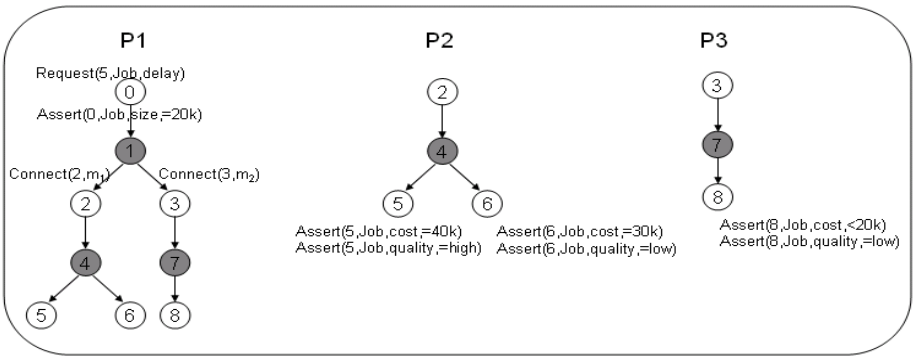

Figure 30. Verbe request

- **Ready***(n)* : exprime le fait qu'un participant soit satisfait de la phase de négociation du nœud blanc *n*. En d'autres termes, le participant a assez d'informations dans le nœud *n* et il est prêt à assigner une action à son pattern d'invocation.
- **Quit***(n)* : exprime le fait qu'un participant ne souhaite pas poursuivre la négociation à partir de la phase de négociation du nœud blanc *n*.

Supposant que le participant P2 répond (cf. Fig. 31) en complétant sa proposition avec une valeur pour l'attribut delay - et que le participant P1 est complètement satisfait avec cette proposition il peut arrêter la négociation dans les nœuds 6 et 8 - *quit(6)* et *quit(8)* - et accepter la proposition du nœud 5 - *ready(5)* -.

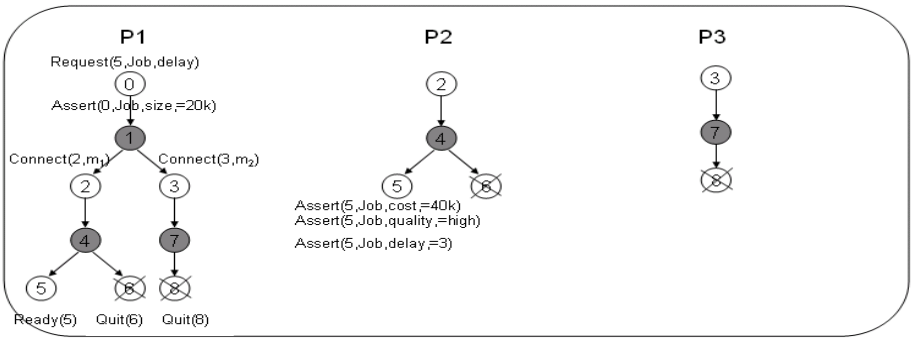

Figure 31. Verbes ready et quit

Donc, le verbe Connect invite dynamiquement un nouveau participant dans une négociation. Les verbes Assert et Open permettent à chaque participant d'une négociation de construire son graphe Xplore, en créant des nœuds et en peuplant les contextes de ces nœuds avec des informations sur les propositions ou les contra-propositions qui vont caractériser les différentes phases des négociations existantes. Le verbe Request évite des situations de

livelock en permettant aux participants d'exprimer leurs besoins pour un certain type d'information afin de continuer la négociation. Les verbes Ready et Quit permettent à un participant de déclarer respectivement qu'il est "préparé pour signer" un contrat définit à partir d'une phase de négociation donnée, ou, au contraire, qu'il souhaite arrêter de poursuivre la négociation à partir de la phase considérée (mais il peut toujours poursuivre la négociation dans d'autres phases).

Le **CooF** est donc une couche intergicielle offrant un ensemble des outils génériques de coordination et de communication pour la mise en oeuvre des « **Facility** » dans un environnement distribué et ouvert.

Plus particulièrement pour le processus de négociation, la couche CooF supporte donc une négociation *multi-attribut, multi-phase* et *multi-participant*. De plus, l'intergiciel peut gérer en parallèle et de manière asynchrone les différentes phases composant une même négociation. Ainsi, comme nous l'avons souligné auparavant, la caractéristique principale de cet intergiciel est l'approche générique qu'il propose. En contre partie, cette approche générique fait que les données englobées dans les différents nœuds sont transparentes à ce niveau et donc des mécanismes visant à construire les graphes de façon consistante avec ces données doivent être proposés aux niveaux supérieurs et notamment au niveau de la « facility » NegF et des Services.

9.4. Processus de décision

La principale fonctionnalité du NegF est d'assister le Manager dans le processus de négociation de la sous-traitance d'une tâche (*job*) dans l'alliance mais aussi dans la coordination de plusieurs sous-traitances concurrentes. Pour cela, il doit être capable de représenter et de manipuler le contexte de chaque négociation afin d'offrir au Manager la possibilité de faire progresser une négociation, de la connecter à d'autres négociations en cours ou à démarrer. Il doit également tenir compte dans le pilotage des négociations des règles de fonctionnement que se fixent les participants de l'alliance ainsi que des contraintes auxquelles les processus de négociation et leurs relations sont soumises. Ces contraintes concernent :

- les éléments constitutifs d'une négociation (rôles, schémas collaboratifs, participants, objets de négociation, stratégies de négociations, etc.) ;
- le type d'information que chaque participant s'engage à délivrer au cours de la négociation ;
- les actions possibles de chaque participant dans la négociation ;
- la manière dont ce processus de négociation peut s'imbriquer dans le processus local de fonctionnement du chacun des participants.

Chaque « facility » NegF est donc en interaction permanente avec un Manager d'un atelier d'impression pour initier et clore des sous-traitances de travaux d'impression. Le manager peut à tout moment faire partie du processus de décision et de pilotage de la négociation mise en œuvre au sein du NegF.

L'architecture du NegF (Fig. 32) permet de gérer et de piloter plusieurs négociations en parallèle impliquant des objets de négociation ou des participants différents. Le NegF fournit un support pour une négociation *multi-attribut* (l'objet sur lequel la négociation se déroule - *Objet de Négociation* (voir section 8.1) - est décrit par plusieurs attributs), *multi-participant* (la négociation implique plusieurs partenaires différents) et *multi-étape* (négociation en plusieurs cycles de propositions et contre-propositions). Afin de contrôler et mettre en place différentes négociations sur ce cycle général, nous avons explicité et regroupé au sein du *Cadre de Négociation,* un ensemble d'informations relatives à la manière dont la négociation doit se dérouler (voir section 8.2.). Le Manager a la possibilité de spécifier non seulement la tâche à négocier, mais aussi *les stratégies*, *les tactiques* ou *les protocoles* à utiliser dans les négociations futures.

Le NegF est également en interaction permanente avec l'intergiciel CooF qui lui offre un support générique au processus de diffusion et de synchronisation des propositions échangées pendant une négociation.

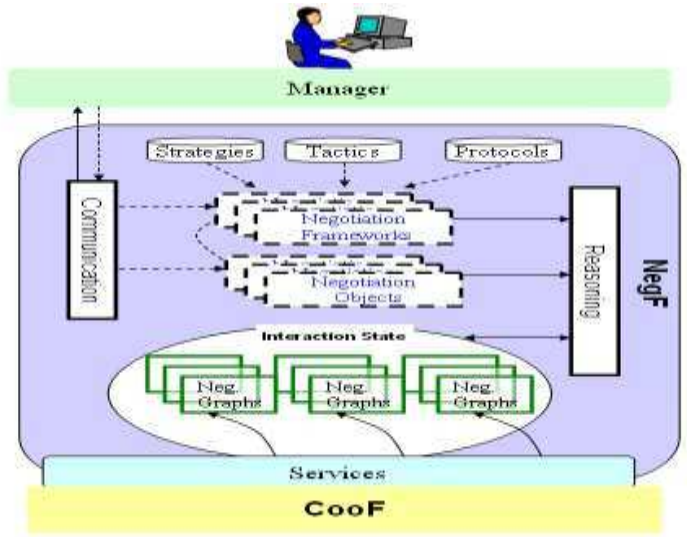

Figure 32 – Architecture du système de négociation

Nous avons vu que le CooF gère une négociation comme un processus de synchronisation des plusieurs copies partielles d'un graphe de négociation. Donc, contrairement au NegF, au niveau du CooF les notions de message, de proposition et de protocole de négociation n'existent pas et les notions d'objet de négociation, d'attributs de négociation et de partenaires de négociation sont présentées comme des paramètres pour le protocole Xplore mais sans retenir aucun concept ou information que ces-ci modélisent.

Etant données les deux couches impliquées dans la négociation (NegF et CooF) et leurs fonctionnalités différentes, une des approches particulières de cette thèse a été d'assurer l'intégration des mécanismes génériques de l'intergiciel avec des mécanismes de pilotage de négociation s'appuyant sur les technologies multi-agent.

Nous proposons un ensemble des *services* comme des mécanismes généraux de mise en œuvre des liaisons entre l'intergiciel CooF et l'agent NegF. Les mécanismes mis en œuvre dans ces services s'intègrent plus précisément entre les mécanismes de diffusion et de synchronisation d'alternatives (i.e. ***processus de communication***) mises en œuvre dans l'intergiciel (CooF) et entre les mécanismes de génération et de gestion des propositions et contre-propositions (i.e. ***processus de décision*** et ***modèle stratégique de coordination***) mis en œuvre au sein de l'agent en charge de la négociation dans le NegF.

Ainsi, au niveau services nous avons choisi d'implémenter les différents mécanismes proposés dans ***le modèle générique de coordination***.

Par la suite, nous allons présenter plus en détail les services proposés et leurs caractéristiques.

10. Services de coordination

A partir de l'architecture présentée dans le chapitre précédent, la composant générique du processus de coordination est mise en œuvre par un ensemble des services. Ces services ont deux fonctions principales :

1) assurer la liaison entre NegF et CooF par la construction du graphe de négociation en manipulant les primitives Xplore sous le contrôle des contraintes en provenance du processus de décision (protocole de négociation, évaluation et création des propositions et stratégies de négociation).
2) mettre en œuvre la coordination générique des négociations dans la construction du graphe de négociation en respectant les contraintes imposées par les comportements des différentes descriptions de négociation ainsi que les différents politiques de coordination qui leur sont attachées.

Dans un premier temps, nous présentons les mécanismes de base d'un service. Nous détaillons ensuite les types de services que nous avons identifiés dans le cadre de ce travail pour implanter différentes politiques de coordination.

10.1. Mécanismes de base d'un service

Dans la présentation du modèle de coordination nous avons détaillé de nombreux scénarios de négociation (voire chapitre 7). La modélisation de ces scénarios comporte non seulement 1) la définition des procédures de coordination que nous avons appelé *schémas de coordination*, mais aussi 2) une définition des procédures de négociation mises en œuvre au sein d'agents NegFs assistant le manager, 3) une définition des mécanismes de synchronisation entre l'évolution de la négociation en termes des informations échangées, 4) ainsi que la construction du graphe de négociation avec les mécanismes génériques proposés par l'intergiciel.

Au fil de la description de ces différents scénarios nous avons mis en évidence des différents *schémas de coordinations* implémentant des mécanismes de base de coordination. Chaque schéma, en fonction de ces fonctionnalités, est composé d'un ensemble des

descriptions de négociation qui ont à leur tour leur propre manière d'interpréter l'image de son graphe et aussi d'agir à son niveau. Le travail présenté par la suite met l'accent sur l'implémentation de mécanismes proposés aux niveaux des services permettant la mise en œuvre des modèles de mécanismes de coordination présentés auparavant (voir chapitre 7.) tout en agençant les ensembles des outils impliqués dans le processus de négociation dans les couches agent et intergiciel.

Comme nous avons précisé auparavant une autre caractéristique de la négociation dans l'E-Alliance est qu'elle est modélisée en tant que construction de collaboration d'un graphe bicolore. Xplore contrôle la négociation à un niveau générique, néanmoins, il y a des informations qui sont transparentes au niveau du logiciel personnalisé. Par exemple, si les attributs qui décrivent le travail sont instantiés ou pas ou si le travail courant fait partie d'une tâche plus grande ou qui sont les participants et quels sont leurs rôles joués dans la négociation ou si la négociation courante est ou peut être liée à d'autres négociations. Ces aspects seront employés au NegF et aux niveaux des services afin de décrire et contrôler le processus de négociation.

Si pour chaque type de service, les mécanismes de coordination sont différents en fonction des ensembles de règles à coordonner, en revanche tous les services ont la même fonctionnalité afin de médiatiser la transition entre l'image de la négociation au niveau du NegF et l'image au niveau du CooF.

Pour arriver à avoir cette fonctionnalité de médiateur, un service doit interpréter le graphe de négociation afin de faciliter le flux d'information du CooF vers NegF (- fonction *d'interprétation* -) et il doit aussi faciliter le flux inverse en traduisant les informations du NegF en termes des actions sur le graphe de négociation (- fonction de *traduction* -).

- *Interprétation du graphe de négociation*

L'intergiciel Xplore fait totalement abstraction de la sémantique des données composantes qu'il coordonne, en revanche au niveau du NegF on parle des propositions de négociation et de protocole de négociation. Alors qu'à un moment donné *le graphe de négociation* représente l'évolution de la négociation en terme d'alternatives concernant les valeurs des attributs en négociation, une *proposition* représente une vue structurée de l'ensemble des attributs négociés à un instant du temps dans une *phase de négociation*. Donc, c'est le service qui doit interpréter le graphe des alternatives manipulées par Xplore afin de traduire en propositions les ensembles des valeurs des différents attributs attachés aux nœuds blancs du graphe de négociation. Comme dans un nœud blanc sont présentes seulement les

informations qui ont généré la création de cette nouvelle alternative, c'est le service qui doit naviguer vers la racine afin de construire une proposition complète.

Dans la Fig. 33 la proposition faite dans le nœud 2 est représentée par les informations sur *cost>100* et *delay<3* présentes dans le nœud courant mais aussi par les informations introduites auparavant dans le nœud 0, comme par exemple *size=20*.

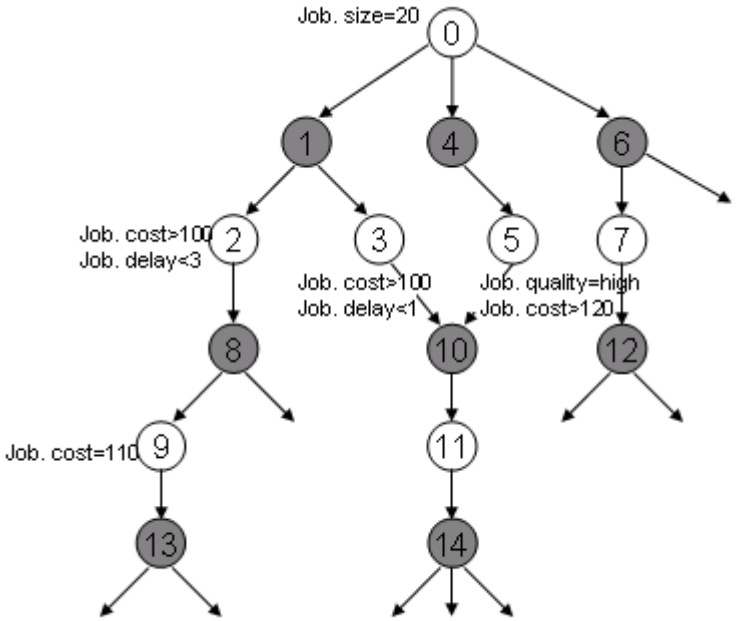

Figure 33.: Graphe Xplore

- *Traduction des actions de négociation*

Au cours d'une négociation un agent NegF peut faire plusieurs propositions ou contre-propositions et ce sont les services impliqués dans la négociation considérée qui doivent *traduire* ces propositions dans les termes du protocole Xplore et selon la structure du graphe qu'il manipule. Le service doit préserver non seulement la représentation et la succession bicolore du graphe Xplore, mail il doit aussi préserver la cohérence du concept d'héritage dans un graphe de négociation.

Selon la *propriété de persistance* et la *propriété de cohérence* introduites par notre modèle de négociation (voire section 5.2.4.2.), le concept d'héritage dans le graphe de négociation est défini par le fait que toutes les valeurs des attributs négociés dans un nœud blanc sont plus spécifiques ou égales aux valeurs des mêmes attributs dans tous les nœuds blancs qui se trouvent sur le trajet en partant du nœud considéré vers la racine du graphe. Donc,

en descendant dans le graphe de négociation on restreint l'espace dans lequel on peut négocier un certain objet afin d'arriver peut-être dans une feuille à une proposition représentant une instantion complète de cet objet. Dans ce cas la traduction d'une nouvelle proposition peut se traduire par trois possibilités : affinement, extension, fusion.

Affinement : Dans notre modèle nous avons défini *l'affinement* et *l'extension* (voire section 5.2.4.2.) comme des propriétés de cohérence du processus de négociation – l'affinement permet de restreindre l'espace de négociation vers une unique option, et l'extension permet d'augmenter l'espace de négociation avec des nouveaux attributs de négociation –. En termes des actions sur le modèle de graphe Xplore l'affinement est traduit par l'ouverture, premièrement d'un nœud noir en partant d'un nœud blanc et ensuite d'un nœud blanc en partant du nœud noir nouvellement créé. C'est par la suite dans ce dernier nœud blanc que la nouvelle proposition est introduite en utilisant des verbes assert. C'est aussi le service qui doit s'assurer que la nouvelle proposition respecte les conditions de l'affinement en fonction du nœud blanc à partir d'où les ouvertures des nœuds ont commencé. Dans la Fig. 33 la proposition faite dans le nœud 9 est un affinement de la proposition faite dans le nœud 2, ce affinement est au niveau de l'attribut *cost* – dans nœud 2 *cost>100*, dans nœud 9 *cost = 110*.

***Extension* :** en termes des actions sur le graphe Xplore elles sont les mêmes que pour l'affinement, avec la seule différence que maintenant un nouvel attribut est introduit et que le service ne doit pas vérifier les contraintes d'héritage entre les nœuds. Dans la Fig. 33 la proposition faite dans le nœud 2 est une extension du nœud 0 en introduisant deux nouveaux attributs *cost* et *delay*.

Fusion : le mécanisme de fusion permet de combiner les alternatives des négociations qui ont été déjà proposées. La fusion permet de continuer la négociation avec une proposition plus complète créée en engluant deux ou plusieurs propositions alternatives auparavant créées. En termes des actions sur le graphe Xplore, la fusion est traduite par la création d'un nœud noir à partir de plusieurs nœuds blancs. Par la suite, tous les futurs nœuds blancs ouverts à partir de ce nœud noir vont hériter les contextes de négociation des nœuds blancs fusionnés. Dans ce type d'action, le service doit préserver aussi les contraintes imposées par l'intergiciel – (les nœuds blancs ayant une racine noire commune ne peuvent être fusionnés) et aussi les contraintes d'héritage – (si un même attribut est négocié dans plusieurs alternatives fusionnées, au niveau des nouvelles propositions créées cet attribut sera associé avec la valeur résultante de

l'intersection des valeurs auparavant proposées). Dans la Fig. 33 la proposition faite dans le nœud 11 est une fusion des nœuds 3 et 5, la proposition dans 11 est : *cost >120, delay<1, quality=high* et *size=20*.

En plus des actions nécessaires pour faire une proposition, il y a aussi les actions pour accepter ou refuser une proposition qui doivent être traduites par des verbes Xplore. Néanmoins, ces actions sont facilement interprétées par les verbes accept et quit

VUE D'ENSEMBLE DES SERVICES DE COORDINATION

La deuxième fonctionnalité des services est d'implémenter les différents schémas de coordination que nous avons proposés dans nos modèles de coordination générique. Dans la définition d'un *schéma de coordination* nous avons fait apparaître des *descriptions de négociation* qui ont des comportements bien délimités selon un ensemble des *règles de coordination* et des *programmes formulaes*. Chacun de ces types de comportements sera par la suite modélisé comme des services. Donc, chaque service représente l'implémentation non-instanciée des ces comportements et chaque instance de service déclenche et gère la mise in œuvre des ces comportements dans le cas particulier d'une négociation donnée.

Nous avons ainsi défini sept services différents :

- ***Outsrc*** : le service principal d'une négociation pour l'atelier initiateur;
- ***Insrc*** : le service principal d'une négociation pour un atelier invité;
- ***Broker*** : le service avec le mécanisme de sélection des possibles partenaires;
- ***AsABlock*** : service assurant que la tâche est sous-traitée en totalité par un seul contractant (voir section 7.3.1);
- ***Split*** : service implémentant les mécanismes de propagation des contraintes et des dépendances entre les deux morceaux de la tâche qui a été divisée afin d'être sous-traitée à deux contractants différents (voir section 7.3.2);
- ***SwapIn/SwapOut*** : services implémentant les mécanismes de liaison d'une négociation courante à une deuxième négociation pour un éventuel changement des tâches (voir section 7.4.1);
- ***Transp*** : service implémentant les mécanismes de liaison d'une négociation courante à une deuxième négociation pour une éventuelle synchronisation sur le transport commun des deux tâches (voir section 7.4.2);

Avec les modèles des règles de coordination que nous avons définis nous sommes capables de générer automatiquement des propositions de négociation. Donc, les services sont capables d'*évaluer* les propositions reçues afin de vérifier les conclusions des règles de coordination et par la suite si celles-ci sont véridiques, les services seront capables de *générer* des propositions en tenant compte des critères (contraintes) de coordination.

Ainsi, la coordination est gérée de manière distribuée par un ensemble des instances des services qui modélise le problème de la coordination comme une construction synchronisée d'un graphe de négociation. Les services sont capables d'utiliser les actions génériques proposées par l'intergiciel et aussi d'interpréter et de comprendre les informations que ces actions apportent en termes des spécifications sur les états de la négociation. Utilisant cette capacité, nous avons défini deux types d'attributs :

- *des attributs des négociations.* Ces attributs sont dérivés de l'ontologie du domaine de l'application considérée (par exemple, dans le domaine de l'impression, les attributs à négocier peuvent être : *size, cost, delay, quality*, ...) et font l'objet des échanges des valeurs entres des services négociant pour des participants différents.
- *des attributs de coordination.* Ces attributs sont utilisés afin de décrire les dépendances entre les négociations et de coordonner par la suite ces dépendances. Les *attributs de coordination* ne sont pas liés aux tâches à négocier mais au processus de coordination des négociations lui-même. Ils sont employés pour exprimer des dépendances précises (par exemple: le fait qu'une négociation attende le résultat d'une autre négociation) et ils font l'objet des échanges des valeurs entre des services négociant pour le même participant (ex. *partner, jobToSwap, readyToSwap, ...*).

Entre des instances des services engagés dans la même négociation et donc collaborant dans la construction d'un même graphe de négociation, les échanges des valeurs pour les attributs de coordination sont très faciles en utilisant le modèle de communication proposé par CooF. En revanche, l'intergiciel n'offre aucun mécanisme de communication ou de synchronisation entre des graphes décrivant des négociations différentes et donc aucun mécanisme pour synchroniser les instances des services engagés dans des négociations différentes. Comme les problèmes de coordination que nous modélisons par des schémas des coordinations gèrent parfois des contraintes entre plusieurs négociations, nous avons ainsi défini deux classes distinctes des services : 1) *services de coordination en monde fermé* : sont les services qui construisent leurs images sur la négociation courante et gèrent les contraintes

de coordination en fonction des informations extraites seulement à partir des actions visibles dans leur graphe de négociation (***Outsrc, Insrc, AsABlock, Split***); et 2) *services de coordination en monde ouverte* : sont les services qui construisent leurs images sur la négociation courante et gèrent les contraintes de coordination en fonction des informations disponibles dans leurs graphes de négociation et également en fonction des informations disponibles dans des structures des données décrivant certaines caractéristiques des négociations où un certain participant est impliqué (***Broker, SwapIn, SwapOut, Transp***).

Par la suite, nous détaillerons chacune des classes des services en nous arrêtant sur la description des fonctionnalités de chaque service qui fait partie de la classe respective.

10.2. Expérimentations

Au fil des descriptions des différents scénarios dans la section sur la coordination en monde fermé nous avons mis en évidence des comportements différents (*program formulae*) implémentant des mécanismes coordination de base pour la mise en place des différents schémas de coordination. En fonction du comportement qu'il implémente, chaque service a ses fonctionnalités et a sa propre manière d'interpréter l'image de son graphe et aussi d'agir sur lui.

10.2.1. Services de coordination en monde fermé

Une caractéristique commune des services de coordination en monde fermé est leur capacité de connecter des autres services dans la négociation en cours. Nous avons défini quatre services qui ont cette capacité : *Outscr*, *Insrc*, *AsABlock* et *Split*.

Entre ses quatre services, deux services sont toujours présentes dans une négociation : *Outsrc* qui démarre la négociation pour le participant initiateur et *Insrc* qui démarre la négociation pour le participant invité.

Si le service *Outsrc*, comme maître du jeu, a la possibilité de voir tout le graphe, et d'utiliser tous les verbes et tous les attributs (issues) de l'objet de négociation, les autres services ont un pouvoir plus restreint sur l'utilisation des verbes et des attributs.

Nous allons par la suite détailler chaque service de coordination en monde fermé.

10.2.1.1. Outsrc

Le service Outsrc, peut être vu comme le service principal d'une négociation. De manière générale, le processus automatique de négociation est initié par la création d'une instance de ce service à partir de l'objet de négociation initial.

Un tel service doit construire le graphe de négociation tout en respectant les aspects de la négociation de haut niveau (évaluation et création des propositions et règles de coordination). Il le fait en manipulant les primitives Xplore.

En plus de ces fonctionnalités, le service Outsrc doit interpréter et vérifier des contraintes sur la négociation qui sont fixées dans les deux structures de données *Objet de Négociation* et *Cadre de Négociation*.

Les informations fournies par la structure *Objet de Négociation* sur les possibles valeurs des attributs à négocier :

*<JDF>::='(' '('<Issue> (<Constraint>) * [<Preference>]')' * ')'* [<Relation>]*

<Issue> ::= <IssueName> (<value> | ' ?')

<Constraint> ::= '(' (<c Exp> | <Constraint>) [<c Function>]')'*

permettent facilement au service Outsrc de vérifier si les différentes propositions reçues sont sur les attributs négociés dans la négociation courante et s'ils sont associés aux valeurs selon les intervalles spécifiés. Supposant brièvement que l'Objet de Négociation impose que le prix est (*cost self.cost <= 10k)*, le service Outsrc peut arrêter la continuation de la négociation dans les phases associées aux nœuds blancs où les propositions sont en dehors de l'intervalle (ex. des nœuds blancs contenant des *assert(12, Job, cost, =11k)*). Egalement, en utilisant l'attribut de coordination *partner*, le service Outsrc peut faire connaître aux autres services quels sont les participants imposés par l'Objet de Négociation dans *:candidates <CDF>* ou si des autres services instancient cet attribut, le service Outsrc peut vérifier facilement si la valeur associée vérifie les contraintes imposées par le Manager.

Finalement, en utilisant le champ *: tactics <tactic>** spécifié dans la structure Cadre de Négociation, c'est aussi le service Outsrc[19] qui peut interpréter la tactique et de l'exécuter en connectant une combinaison des différentes instances des services.

[19] Au niveau intergiciel Outsrc, il a aussi les fonctionnalités de gérer l'aspect transactionnel de la négociation. Il est vue comme un *coordonnateur* et il a le rôle de conclure l'accord entre les instances des services participants dans une même négociation.

Ainsi, le service Outsrc et aussi le service Insrc, décrit par la suite, sont les services qui font la liaison entre les aspects imposés par la coordination stratégique spécifiés au niveau agent et la mise en œuvre des ces aspects au niveau service de coordination.

10.2.1.2. Insrc

Le service Insrc est le service qui gère la négociation du coté de l'atelier qui décide d'accepter une tâche proposée dans l'alliance. Il a des fonctionnalités similaires aux celles du service Outsrc mais avec quelques modifications. Ces modifications sont dues au fait que le service Insrc n'a pas une image complète sur la négociation et qu'au début de la négociation le service Insrc n'a aucune information ni sur quoi il négocie ni quelles sont les contraintes de son Manager.

Premièrement, la vue du service Insrc sur le graphe de la négociation est limitée aux données qui concernent seulement sa négociation directe avec le service Outsrc ou un autre service qui négocie pour l'atelier initiateur de la négociation. Deuxièmement, à la différence du service Outsrc qui a dès le début les contraintes spécifiées par le Manager dans les structures *Objet de Négociation* et *Cadre de Négociation*, le service Insrc a une interaction plus riche avec son Manager en lui demandant au fur et à mesure, ses contraintes sur les nouveaux aspects demandés dans la négociation. Donc, en fonction des attributs qui sont demandés par l'initiateur de la négociation - utilisant le verb `Request` -, le service Insrc est capable de construire progressivement les structures des données décrivant les préférences du Manager sur l'objet de négociation et sur le processus de négociation lui-même [voir Annexe – algorithme pour request].

10.2.1.3. AsABlock

Le service AsABlock est connecté dans les négociations où la tâche doit être sous-traitée en totalité par un seul contractant.

Sa principale fonctionnalité est de médiatiser la négociation entre l'atelier initiateur de la négociation et les autres ateliers invités dans la négociation courante. La médiation est faite avec le but d'arriver à un contrat sur la totalité de la tâche avec un seul participant.

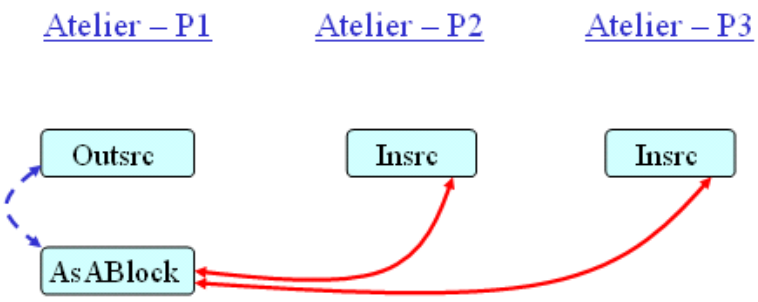

Figure 34. Interactions service AsABlock

La gestion de la contrainte de ne pas décomposer la tâche sous-traitée en différents lots se traduit en opérations spécifiques sur le graphe de négociation. Elle est ainsi déléguée à un service, appelé **AsABlock** et le comportement nous l'avons décri selon notre modèle par un schéma de coordination (voir section 7.3.1.). Celui-ci doit s'assurer tout au long de la négociation que les propositions liées à la tâche gérée par le service Outsrc sont faites sur la totalité de la tâche et non sur des morceaux. Le service AsABlock est lui aussi invité à participer à la négociation courante par le service Outsrc.

C'est par la suite que le service AsABlock qui invite (connect) les services Insrc de chaque partenaire indiqué, afin de négocier avec eux. C'est aussi le service AsABlock qui coordonne les négociations bilatérales créées et qui détecte les nœuds blancs avec des propositions qui ne sont pas faites sur la totalité de la tâche et d'arrêter dans ces cas les alternatives de la négociation, en fermant les nœuds correspondants. Donc, on peut dire que le service AsABlock est un service attaché à l'attribut de coordination «size». Toutes ces contraintes de coordination ont été spécifiées par des règles de dépendances et par un programme formulae délimitant le comportement particulier d'un type de description de négociation et donc implicitement le comportement du service employé pour implémenter ce type de description.

Dans la Fig. 34 nous avons présenté les services impliqués dans une négociation entre un atelier donneur d'ordre A1 et deux autres ateliers A2 et A3. Dans ce cas la négociation commence avec l'instanciation du Outsrc qui par la suite invite le service AsABlock. C'est AsABlock qui invite les services Insrc de chaque partenaire et il va coordonner les deux négociations bilatérales concurrentes avec les deux partenaires A2 et A3. Au niveau des Manager des ateliers A2 et A3, cette mise en concurrence n'est pas forcément connue.

Une fois que tous les services sont connectés, le processus d'interaction entre les participants peut démarrer. Pendant celui-ci, les NegF de chaque atelier impliqué dans cette

négociation commencent la génération et l'échange des propositions et contre-propositions sur la tâche proposée à la sous-traitance en manipulant les valeurs des différents attributs qui la décrivent. Ces interactions se traduisent en manipulations des graphes de manière synchronisée par les services attachés à chaque NegF : service Outsrc et AsABlock pour le NegF de l'atelier A1 et un service Insrc pour chaque atelier invité à participer à la négociation A2 et A3.

La négociation se termine au moment où l'atelier A1 arrive à un accord avec un des partenaires, par exemple avec A2, sur l'ensemble des attributs qui décrivent la tâche à négocier. Dans le même temps A1 arrête la négociation avec A3, cette coordination est aussi assurée par le service AsABlock. Bien entendu, la négociation peut s'arrêter sans aucun accord (par exemple le temps limité de négociation est dépassé ou les deux partenaires A2 et A3 ne veulent pas négocier).

10.2.1.4. Split

Le service Split est connecté dans les négociations où la possibilité de diviser la tâche en deux morceaux pour deux contractants différents est envisageable.

Sa fonctionnalité est de coordonner les liaisons entre la description complète de la tâche faite par une instance d'un service Outsrc et les propositions sur des parties de la tâche, faites dans deux instances du service Insrc.

A partir du moment où l'instance de ce service est connectée, elle devient un intermédiaire entre le service Outsrc et les deux services Insrc.

En effet, le service Split soulève des contraintes. Les contraintes sur la tâche à négocier sont fixées par le service Outsrc. Au moment où un des services Insrc fait une proposition, le service Split est capable de construire la proposition pour le deuxième service Insrc de manière que la tâche proposée par Outsrc soit sous-traitée en totalité. Son comportement est aussi la mise en œuvre d'un schéma de coordination défini dans notre modèle de coordination (voir section 7.3.2.).

Le service Split est invité dans la négociation par le service Outsrc. Exemplifiant aussi par un bref exemple, dans la Fig. 35, la mise en œuvre de la négociation est réalisée au sein d'un service Outsrc et d'un service Split pour l'atelier donneur d'ordre A1 et d'un service Insrc pour chacun des autres ateliers invités dans la négociation. Egalement, comme dans les premiers scénarios, les deux services Insrc coordonnent seulement leurs interactions avec

l'Outsrc du donneur d'ordre A1. De la même manière, les ateliers A2 et A3 ne sont pas forcément au courant de leur mise en concurrence :la synchronisation entre leurs propositions est faite par le service Split.

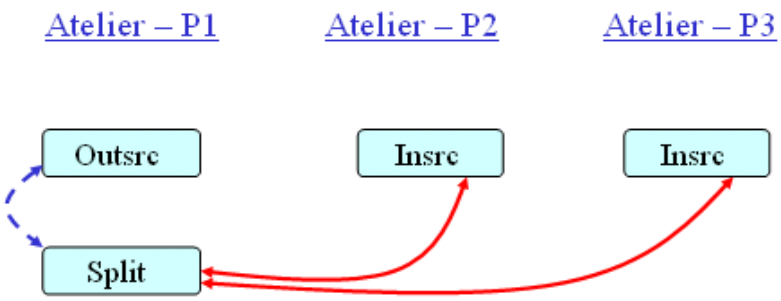

Figure 35. Interactions service Split

De cette manière, la gestion de la résolution des contraintes est déléguée à un service **Split**. Ce service met en place un mécanisme générique de résolution des contraintes, et assure la propagation de celles-ci.

La négociation se termine au moment où l'atelier A1 est satisfait par les propositions faites par A2 etA3, la différence avec la première négociation est qu'ici avec un seul accord nous avons terminé les deux négociations en même temps mais nous avons aussi établi deux contrats.

Donc, dans une négociation et entre ces types des services toutes les communications entre leurs instances sont traduites en verbes Xplore qui manipulent les graphes de la négociation courante. Cette caractéristique limite donc le processus de coordination seulement sur les négociations bilatérales composant une même négociation. Par la suite, nous introduisons des nouveaux types des services proposant des mécanismes de coordination entre différentes négociations.

10.2.2. Services de coordination en monde ouvert

L'objectif de la définition des services de coordination en monde ouvert est de permettre à l'infrastructure E-Alliance de gérer le flux des informations entre plusieurs négociations concurrentes et de coordonner les possibles dépendances entre celles-ci. Donc, la principale fonctionnalité des services est de médiatiser le flux des informations venant de l'extérieur de la négociation courante et en spécial des informations venant des autres négociations dans lesquelles l'atelier considéré est impliqué simultanément.

A la différence des services de coordination en monde fermé qui gèrent un flux des informations inter-ateliers mais intra-négociation, les services de coordination en monde ouvert gèrent un flux des informations intra-atelier mais inter-négociations. Comme l'intergiciel n'offre pas des mécanismes de communication entre les instances des services impliqués dans des négociations différentes, les services de coordination en monde ouvert ne connectent pas d'autres services dans la négociation mais ils se connectent à des structures des données particulières. A partir des ces structures des données les services sont capables de lire des informations sur le monde extérieur et spécialement sur les autres négociations existantes et aussi ils sont capables d'écrire dans ces structures des données afin de mettre à jour les informations sur la négociation dans laquelle ils sont impliqués. Les informations sauf gardées dans ces structures sont susceptibles d'être nécessaires afin d'établir des relations de dépendance entre plusieurs négociations. Un exemple des informations nécessaires sont les identifiants des participants impliqués afin de gérer un possible échange entre les négociations impliquant les mêmes participants (voir section 7.4.1).

Un aspect principal de cette approche est la possibilité de spécialiser des services capables de coordonner ce type des dépendances sans être concernés par tous les aspects du processus de négociation. Envisageant plusieurs scénarios nous avons proposé trois types des approches pour la coordination en monde ouvert et pour chacune nous avons proposé des services différents :

- **Broker** : est le type de service qui peut lire seulement à partir de ces structures des données ;
- **SwapOut/In** : est le type composé par deux services qui peuvent lire et écrire dans ces structures des données, et qui gèrent une dépendance unidirectionnelle entre deux négociations ;

- **Transp** : est le type de service qui peut lire et écrire dans ces structures des données et qui gère une dépendance bidirectionnelle entre deux négociations.

Par la suite, nous allons présenter plus en détail chacun de ces services et leurs fonctionnalités.

10.2.2.1. Broker

L'aspect de la négociation que nous avons intégré comme un service est le mécanisme de sélection des possibles partenaires. La modalité de représentation et de sélection des partenaires possibles s'appuie sur l'approche développée dans [Vercouter'00].

La fonctionnalité du service Broker est de chercher les partenaires qui peuvent faire un certain type de tâche. Sa vue sur le graphe est limitée sur les données du nœud où il a été connecté, mais en plus de ça le service Broker a l'accès à la base des données de la représentation des autres. A partir du graphe, et plus précisément à partir du nœud de la connexion, le service Broker est capable d'extraire les attributs qui caractérisent les attributs cherchés aux possibles partenaires et, en suite, d'utiliser ces attributs comme conditions respectées dans les propositions des partenaires.

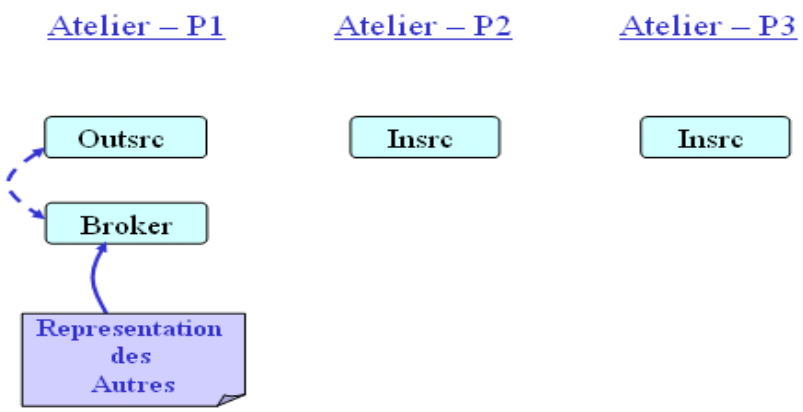

Figure 36. Interactions service Broker

Dans la Fig. 36 nous pressentons le cas d'un possible scénario. Comme dans les scénarios précédents, la négociation envisagée est une négociation multi-bilatérale et multi-attribut. La seule différence est que le Manager ne précise pas les partenaires possibles et délègue cette tâche à l'infrastructure. Les partenaires ne seront ainsi connus qu'en cours de négociation et non à l'initialisation de celle-ci, comme nous pouvions le voir ci-dessus. Ce

processus de sélection des partenaires est délégué à un service appelé Broker. Celui-ci s'appuie sur la description de la tâche à négocier et sur les mécanismes de représentation des autres. Ainsi, dans la Fig. 33 le service Outsrc initie la négociation en demandant au service Broker de lui trouver des ateliers qui peuvent faire un certain type de tâche. Une fois que ce service Broker aurait trouvé des partenaires possibles selon les caractéristiques de la tâche, la négociation se trouvera au point initial des scénarios précédents.

10.2.2.2. Swap

À un moment de temps, un atelier d'impression peut être impliqué dans plusieurs négociations, certaines d'entre elles qu'il a lancées et d'autres auxquelles il a été invité à participer. C'est ainsi possible d'avoir plus d'une négociation en cours entre deux ateliers d'impression. En conséquence, il est possible qu'un des Managers veuille rechercher des dépendances entre ces négociations dans lesquelles les mêmes participants sont impliqués. Par la suite, nous nous concentrerons sur des relations d'échange (*relations swap*) entre ces types des négociations.

Nous définirons une *relation swap* entre deux négociations (*swap(N1, N2)*) comme la relation qui établie une dépendance unidirectionnelle entre les négociations N1 et N2 :

« Si N2 arrive dans un statut de failure, N1 finira aussi dans un statut de failure.»

Ce type de dépendances nous l'avons modélisé par un schéma de coordination décrivant les comportements des plusieurs descriptions de négociation par des politiques de coordination particuliers (voir section 7.4.1.). Par la suite, dans cette section, nous présentons l'architecture des services swap qui visent à contrôler ces politiques de coordination composées des règles de coordination de type swap. Essentiellement, nous offrons aux deux services **SwapOut** et **SwapIn** la possibilité d'avoir accès à toutes les informations sur les négociations dans lesquelles l'atelier d'impression est impliqué et d'être capables de décrire ces négociations en termes de *dépendances swap*.

La gestion des relations swap (cf. Fig. 37) implique deux services appelés **SwapOut** et **SwapIn** et une structure de données partagées entre ces services appelée **Swap Space**.

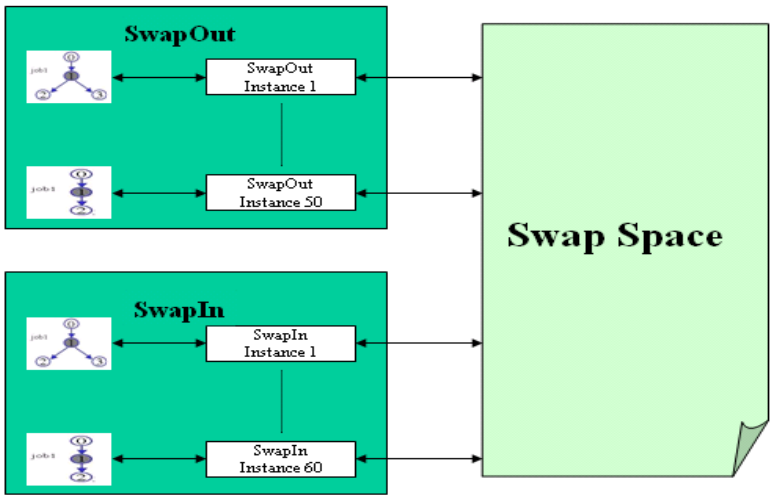

Figure 37. Services SwapOut et SwapIn

SwapIn/SwapOut

Les services SwapOut/In sont connectés dans des négociations déjà commencées afin d'initier une *relation swap* entre ces négociations et aussi de gérer par la suite cette relation.

L'instance du service SwapOut est reliée dans une négociation par les services de coordination en monde fermé (Outsrc, AsABlock ou Split) quand ceux-ci décident de chercher un échange possible. Le service SwapOut a deux fonctionnalités principales: 1.) d'abord, il recherche des négociations qui conviennent à un échange et 2.) ensuite, il participe à la validation et à la coordination de la nouvelle *relation swap* établie.

L'instance du service SwapIn est reliée dans une négociation par le service Insrc. Cette connexion peut être faite du premier nœud blanc sur lequel l'instance du service Insrc peut faire des actions. Le service SwapIn offre, pour une négociation donnée, la possibilité d'annoncer aux autres négociations quelles sont les caractéristiques de la négociation courante et par la suite d'échanger des données avec les autres négociations avec lesquelles des relations swap ont été établies.

Dans la Fig. 38 dans ce scénario les deux négociations entre A1 et A2 se déroulent d'une façon similaire aux scénarios décrits ci-dessus. Lorsque A1 décide de chercher un possible échange (swap) et donc de continuer la négociation en mode ouvert, son service Outsrc invite le service SwapOut. Le service Outsrc demande alors au service Swap de lui trouver d'autres négociations avec l'atelier A2 pour un échange possible. Une fois que ce service SwapOut a trouvé des négociations possibles selon les caractéristiques d'un swap, le service Outsrc décide avec quelle négociation il veut faire l'échange et communique sa décision au Swap. Le service SwapOut a comme tâche de s'assurer que la première

négociation (celle dans laquelle A1 est donneur d'ordre), ne se termine par un contrat que si la deuxième négociation (celle dans laquelle A1 est impliqué en tant que fournisseur) se termine par un contrat. Ce comportement nous l'avons décri selon notre modèle par un schéma de coordination (voir section 7.4.1.).

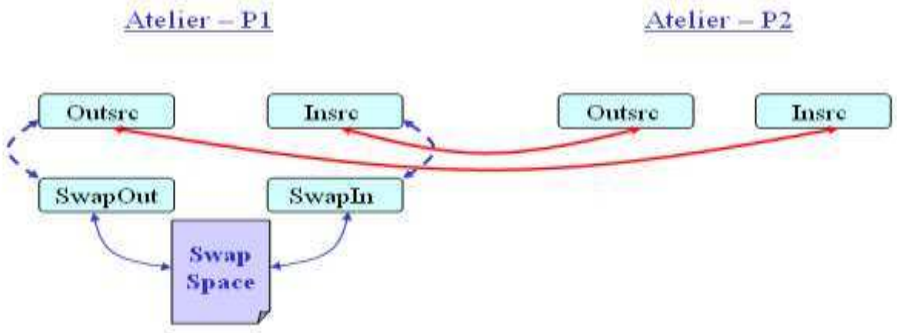

Figure 38. Interactions services SwapOut et SwapIn

Donc, les fonctionnalités de base des services SwapOut et de SwapIn sont :

- *d'interpréter* les graphes de négociation afin d'identifier les *relations swap* établies et de modifier le Swap Space selon celles-ci;
- *de transformer* les modifications faites dans le Swap Space par les autres instances des services Swap, dans des actions sur leur vue locale du graphe de négociation en utilisant les verbes Xplore.
- *de trouver et résoudre* les possibles cycles des dépendances entre les relations swap. Afin de faire ceci, les services Swap ont seulement besoin de connaître les aspects qui caractérisent les dépendances de type swap entre les différents processus de négociation (par exemple les participants impliqués dans la négociation et si d'autres relations swap étaient établies précédemment). Ces aspects sont extraits par toutes les instances des services SwapIn et SwapOut et sont mis dans la structure de données Swap Space. Le Swap Space peut être vu comme un tableau noir auquel chaque instance des services SwapOut/In peut écrire et lire des données liées à la coordination de relations swap. Ces données sont des ensembles pertinents des attributs avec leurs valeurs. Ces attributs sont des *attributs de coordination* et ils décrivent les relations swap.

Pour la dépendance de type swap les attributs de coordination sont : *partner, jobToSwap, askToSwap* et *readyToSwap*.

- *partner* : la valeur de cet attribut est le nom du partenaire avec lequel un échange peut être fait. Il est introduit (utilisant le Xplore verbe assert) par

l'instance du service Outsrc afin de répondre à une demande (utilisant le Xplore verbe request) faite par l'instance du service SwapOut. Le même mécanisme est employé entre les instances du service Insrc et du service SwapIn, afin d'indiquer le nom du partenaire avec qui la négociation courante est faite;

- *jobToSwap* : la valeur de cet attribut est le nom de l'instance du service Insrc et donc de la négociation avec laquelle un échange peut être fait. Il est introduit par l'instance du service SwapOut afin de répondre à la demande faite par l'instance d'un service Outsrc de trouver une négociation appropriée pour un échange. Le même attribut est demandé par l'instance du service SwapIn qui interroge le service Insrc afin d'indiquer avec la quelle des négociations des liens swap sont déjà établis. Dans un contexte où nouvelles négociations commencent et se finissent à tout moment, en fonction de la dynamique du système, l'instance SwapOut ferme les nœuds (utilisant le Xplore verbe quit) avec les noms des instances Insrc impliquées dans les négociations finies et ouvre des nouveaux nœuds (utilisant le Xplore verbe open) pour les instances Insrc impliquées dans les négociations nouvellement démarrées;
- *askToSwap* : le SwapIn introduit cet attribut en identifiant les noms des instances des services avec lesquelles l'instance courante du service Insrc est déjà impliquée dans une relation swap. Le même attribut est demandé par le service SwapOut afin de trouver si la négociation courante est déjà impliquée dans un échange ;
- *readyToSwap* : la valeur de cet attribut est introduite par l'instance du service Outsrc et elle consiste dans une valeur booléenne (ex. *assert(20,readyToSwap, true)* - l'instance d'un service Outscr introduit dans le nœud 20 pour l'attribut *readyToSwap* avec la valeur *true*, donc à partir de ce nœud le processus swap est déclenché). Le même attribut est donc demandé par l'instance du service SwapOut afin de connaître à partir du quel nœud il doit gérer les relations swap.

Ces attributs ne sont pas réellement négociés au niveau de l'agent, mais ils contiennent des informations qui décrivent le processus de coordination et plus particulièrement des

relations de dépendance établies dynamiquement entre les négociations en cours. Ces nouveaux rapports entre les négociations nous pouvons les voir comme une nouvelle négociation appelée *négociation étendue (extended negotiation*). La *négociation étendue* est représentée par les instances des services de coordination en monde ouvert dont les graphes contiennent seulement ces attributs de coordination avec leurs valeurs associées. Dans ce scénario la négociation étendue est représentée par les instances de services SwapOut et de services SwapIn gérant des graphes contenant les attributs de coordination présentés auparavant.

Ainsi nous avons étendu la description de la tâche à négocier avec des nouveaux attributs requis pour maintenir l'information nécessaire aux processus de coordination en monde ouvert. Dans ce cas l'information est liée en particulier aux aspects permettant une *relation swap* directe entre deux négociations différentes, par la suite nous allons présenter un nouveau scénario nécessitant des nouveaux mécanismes de coordination en monde ouvert.

10.2.2.3. Transp

Les services Swap gèrent des dépendances de statuts unidirectionnels entre deux négociations bilatérales. Dans le modèle suivant de service de coordination en monde ouvert, nous allons modéliser des dépendances d'attributs et de statuts bidirectionnels entre deux négociations.

Dans le cas des tâches d'impression le transport fait aussi partie du contrat et il est donc possible qu'un des ateliers ait contracté des tâches qui doivent être livrées dans un certain délai et dans un espace géographique identique ou très proche. En conséquence, le Manager de l'atelier cherche à lier de manière dynamique ces négociations, afin de minimiser le coût du transport, en utilisant une seule de ses voitures de transport pour faire les deux livraisons. L'atelier peut chercher à gérer en parallèle les deux négociations afin que les livraisons soient faites dans des lapses de temps proches et que les tailles des deux tâches se complètent pour entrer dans le même transport et que les deux négociations se finissent par des contrats. Ces types des relations (noté *transp(N1,N2)*) nous les avons modélisées par un schéma de coordination (voir section 7.4.2.) décrivant des règles de coordination particulières et par la suite on va présenter les services et les structures des données qui gèrent ces dépendances.

La gestion d'une *relation transport,* comme pour les *relations swap*, implique l'utilisation d'une même architecture composée par deux instances des services et une structure des données. A la différence du scénario Swap, ici les deux négociations sont gérées au même

niveau donc elles seront représentées par des instances du même service **Transp**. Les deux instances de service *Transp* communiquent entre elles à travers une structure des données partagées appelée *Transp Space* (cf. Fig. 39).

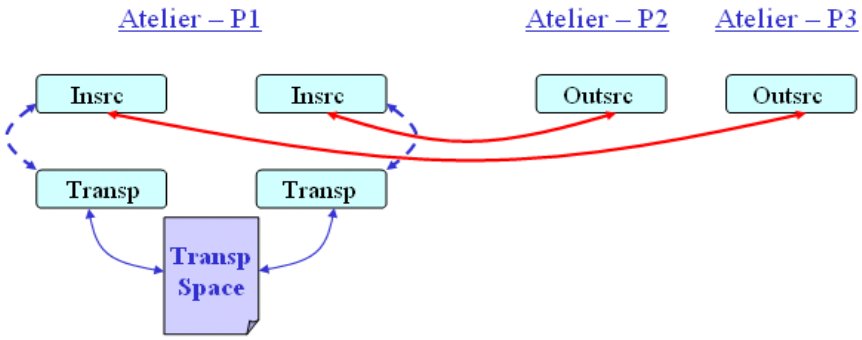

Figure 39. Interactions service Transp

Ayant les mêmes fonctionnalités que les instances des services Swap, les instances du service Transp lisent et écrivent dans la structure *Transp Space* les informations sur les attributs de négociation (- *location, size, delayTransp* -) qui font l'objet des règles de coordination mises en place par le schéma de coordination correspondant et aussi les informations sur les attributs de coordination utilisés dans le processus de coordination de ces dépendances. Ces attributs de coordination sont : *jobToBind, readyToBind* et *acceptToBind*.

Du moment où une instance du service Transp est connectée dans une négociation par une instance des services de coordination en monde fermé, l'instance met à jour la structure Transp Space avec le nouveau contexte de négociation pour lequel une relation de synchronisation sur le transport est envisagée. Par la suite, l'instance Transp cherche aussi au niveau de la structure partagée, avec quels contextes une relation de synchronisation est possible et elle les introduit dans la négociation courante comme des valeurs pour l'attribut de coordination *jobToBind*. Comme dans le cas précédant, la dynamique du système est prise en compte par l'instance Trasnp qui ferme les nœuds avec les contextes qui sont retirés de la structure *Transp Space* (un contexte est retiré si une relation Transp a été mise en place avec ce contexte ou si la négociation proposant ce contexte est finalisée) et ouvre des nœuds pour les nouveaux contextes proposés dans la structure *Transp Space* et qui vérifient les contraintes imposées.

Par la suite, c'est à l'instance du service qui a connecté l'instance Transp de lui annoncer avec quel contexte une relation de synchronisation sur le transport est envisageable.

Ce choix est marqué par l'introduction de l'attribut de coordination *readyToBind* avec une valeur booléenne (ex. *assert(20, readyToBind, true)* dans un des nœuds blancs contenant les contextes proposés par instance Transp. Comme la relation est bidirectionnelle, le service Transp doit s'assurer que la proposition de dépendance peut être encore réalisée et que cella est acceptée. A ce niveau différents algorithmes peuvent être proposés afin d'augmenter l'efficacité en termes des négociations associées (i.e. quelle est la plus courte distance à parcourir ou quelle est la quantité optimale qui peut être obtenue, etc.). Comme nous nous sommes intéressés seulement à des mécanismes de coordination afin de gérer ce type de relation, nous supposons que la demande de mise en relation est acceptée de manière intrinsèque. Ainsi, suite au fait que le choix du contexte est réalisé, le service Transp regarde dans la structure *Transp Space* si le contexte est encore valide et dans ce cas le service Transp annonce l'acceptation de la relation transp avec l'attribut de coordination *acceptToBind* (ex. *assert(20, acceptToBind, true)*).

Donc, en utilisant ces types des services qui gèrent un flux des informations entre les négociations dans lesquelles ils participent et des structures des données décrivant certains aspects sur les autres négociations dans lesquelles un participant est impliqué, nous arrivons à mettre en œuvre les schémas de coordination en monde ouvert.

10.3. Exemple

Les différents services de coordination en monde ouvert ou fermé que nous venons de définir mettent en avant les principales caractéristiques de la coordination de négociations mises en œuvre au sein du projet E-Alliance : *distributivité* et *parallélisme*.

Les schémas de coordination, que nous avons présentés dans le modèle sont mis en œuvre de manière distribuée sur des services différents et ils peuvent être exécutés en parallèle par l'infrastructure proposée. Par la suite, nous allons présenter un exemple schématique de modélisation d'un processus de négociation en utilisant les mécanismes de coordination et de négociation que nous avons proposés dans ce chapitre.

Le processus de négociation que nous présentons ci-dessous (cf. Fig. 40) est structuré en cinq parties (Initialisation, Choix de tactique, Choix des partenaires, Négociation et Adoption). La spécification du processus vise à définir les étapes à suivre et à fixer les services nécessaires pour coordonner le processus de négociation comme une exécution de plusieurs négociations concurrentes.

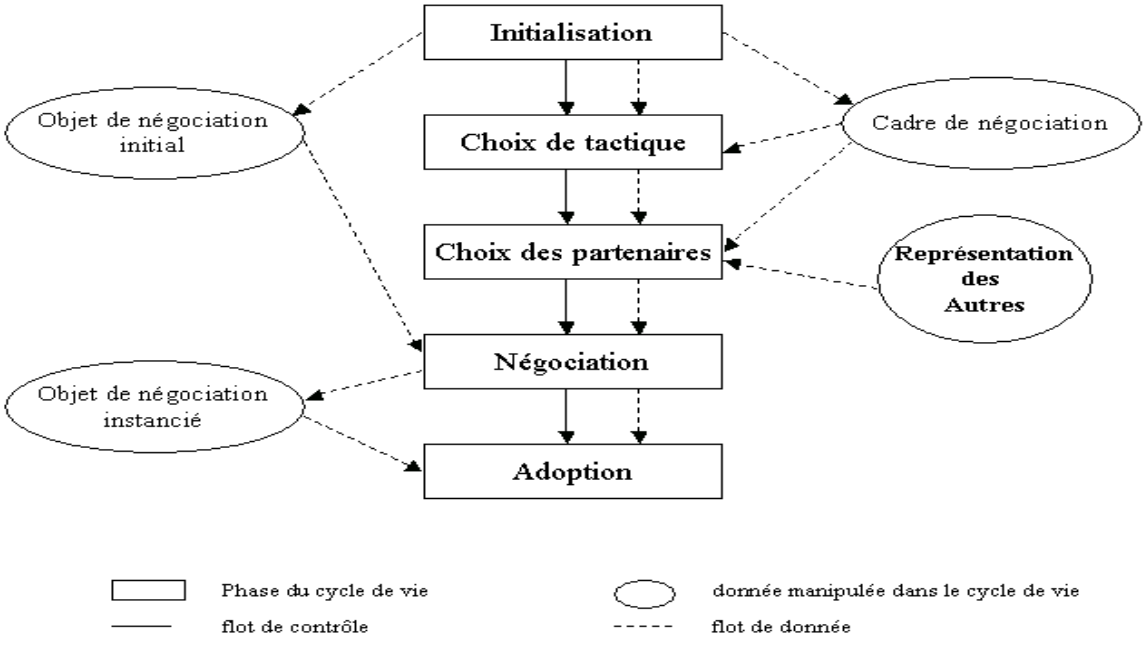

Figure 40. Structuration du processus de négociation

Initialisation. Un Manager initie une sous-traitance en définissant et en communiquant au NegF les propriétés et les contraintes de l'Objet de Négociation et le Cadre de Négociation. Le processus de négociation commence avec la création d'une instance du service Outsrc. Cette

instance va initier les autres étapes du processus de négociation, à partir des contraintes fournies par le Manager : création des nœuds dans le graphe de négociation et invitation des services (Insrc, Broker, etc) comme nous avons pu le décrire dans les scénarios précédents. Egalement, cette instance va conduire la négociation en termes de construction et d'évaluation des propositions pour la tâche proposée en sous-traitance.

Choix de tactique. Utilisant comme données brutes les tactiques spécifiées dans le Cadre de Négociation, la coordination est décomposée en plusieurs schémas de coordination. Pour le moment, nous avons envisagé quatre tactiques qui correspondent aux quatre schémas de coordination: AsABlock, Split, Swap et Transp. La caractéristique de la connexion de ces différents services est que les instances des services AsABlock et Split doivent être connectées de manière à représenter un choix dans la description du graphe de la négociation. Elles doivent être connectées dans des nœuds blancs différents mais ayant le même nœud noir comme père. Par contre, les services Swap et Transp peuvent être connectés n'importe où dans le graphe de la négociation.

Choix des partenaires. Nous avons deux possibilités de choix des partenaires : i) avec les partenaires connus – le Manager qui initie la sous-traitance peut éventuellement spécifier des contraintes sur l'ensemble des contractants possibles au sein de l'alliance. Pour cela, le Manager utilise la description du job à sous-traiter et éventuellement, sa représentation des partenaires dans l'alliance et/ou des différents contrats d'adhésion qu'ils ont signé, et ii) avec des partenaires inconnus – dans ce cas, tout le travail de recherche de possibles partenaires est pris en charge par l'infrastructure et notamment par les services Broker.

Négociation. Dans cette étape, au fil des échanges des propositions, l'infrastructure fait évoluer l'objet de négociation selon les contraintes imposées par le Manager sur les attributs de la tâche à négocier. L'objectif final du processus de négociation est de construire un Objet de Négociation Instancié[20] à partir des spécifications initiales d'Objet de Négociation. Une fois terminé, cet objet sera ensuite utilisé pour établir un Contrat.

[20] Un *Objet de Négociation Instancié* est un objet de négociation dont tous les attributs ont reçu une valeur acceptée par les différentes parties.

Adoption. Finalement, à l'issue de la phase de négociation, les propriétés à négocier ont des valeurs fixées. Le NegF interroge alors le Manager pour valider le résultat de la négociation et contacte les NegFs des autres partenaires. Selon les réponses obtenues, le manager peut : i) relancer ou abandonner la négociation ou ii) activer la phase de contractualisation qui permet de déboucher sur un contrat.

Donc, nous sommes arrivés à proposer une infrastructure qui offre les possibilités de :

- définir d'une manière structurée le processus de négociation : participants, protocole d'interaction – protocole de négociation, tactiques et services de coordination -, objet de négociation, stratégies de négociation ;
- modéliser la totalité des négociations d'un atelier d'impression comme des ensembles de négociations bilatérales où l'agent impliqué dans ces négociations peut les piloter de manière autonome.
- modéliser la coordination entre ces négociations en s'appuyant sur un ensemble des services de coordination et sur les mécanismes de synchronisation de l'intergiciel.

Ainsi, nous proposons une infrastructure qui gère, de manière *décentralisée*, la coordination des négociations *multi-phase*, sur un objet *multi-attribut* et entre *plusieurs participants*.

11. Implémentation

Le travail pratique de la thèse consiste à fournir l'infrastructure logicielle permettant de satisfaire les objectifs du projet en nous appuyant sur les travaux théoriques menés. Notre objectif a été de développer une infrastructure logicielle intégrant les trois couches de l'infrastructure E-Alliance (une couche d'interaction avec les acteurs extérieurs, une couche « facilities » pour la gestion des actions dans l'alliance et une couche intergicielle offrant les outils génériques de communication et de transaction), plus notre couche des services de coordination. Notre partie de l'infrastructure dédiée exclusivement à la négociation est ainsi structurée de la manière suivante : Manager – NegF – Services – CooF

L'implémentation développée à partir du travail théorique est ainsi structurée qu'elle nous permet d'identifier la partie dédiée à l'utilisateur (le Manager) et la partie semi-automatique du processus de négociation que l'infrastructure met à la disposition de l'utilisateur.

11.1. Contraintes d'implémentation

En plus des contraintes imposées par le modèle conceptuel, la mise en place de notre architecture logicielle a été guidée par plusieurs contraintes d'implémentation :

- d'abord, le système devra soutenir le partage de l'information et la prise de décision de manière collaborative à travers plusieurs organisations autonomes qui peuvent être géographiquement distribuées. Dans ce cas, on peut prévoir qu'un serveur central ne pourra pas soutenir l'exécution pour un grand nombre de participants. Ainsi, les services que nous devons fournir doivent être clonés et installés sur plusieurs serveurs Web.
- deuxièmement, les services que nous offrons doivent pouvoir conduire concurremment plusieurs négociations et plusieurs transactions afin de fournir une solution flexible pour les utilisateurs qui veulent négocier leurs travaux.

- troisièmement, les formats des données échangées pendant la négociation doivent être expressifs et assez puissants afin de décrire l'objet de négociation, les attributs composant la tâche à négocier et les valeurs des différentes phases alternatives de négociation proposées au cours d'une négociation.

11.2. Architecture logicielle

L'architecture que nous décrivons par la suite est composée principalement par un côté client et un côté serveur (cf. Fig. 41).

- du côté client nous avons les différents ateliers d'impression appelés *Composant* (*Components)*. Un *composant* peut choisir à négocier dans un service de coordination désiré et également dans une invocation particulière de ce service. L'invocation d'un service créera un graphe de négociation particulier dans lequel le *composant* négociera. Afin de visualiser et agir sur le graphe de négociation, nous avons créé une interface graphique, appelée **Editor**.

- du côté serveur nous avons les différents services de coordination (par exemple Outsrc, Insrc, AsABlock, … etc.). Les différentes instanciations des services seront gérées concurremment. Pour chaque service de coordination, nous fournissons une structure de données capable d'enregistrer toutes ses instances créées. Ainsi, le serveur est capable de fournir les différentes copies des instances existantes aux composants qui sont enregistrés à lui.

Un *composant* est mis en application comme un client de type SOAP RPC et l'infrastructure de négociation (NegF-Services-CooF) est mise en application sur un serveur SOAP. Nous avons fait ce choix parce que : *i)* premièrement, nous voulons que le client soit une application légère et employée seulement pour visualiser une négociation et *ii)* deuxièmement, nous avons à un moment donné, au niveau du client un seul graphe de négociation et en revanche sur le serveur nous gérons tous les graphes de négociations existantes à un moment donné. Afin de mettre en application un système réparti, chaque atelier d'impression aura la même structure software. La communication entre les ateliers d'impression est mise en application comme un lien de type peer-to-peer entre les serveurs correspondants. Le protocole soutenu dans la communication entre les ateliers d'impression est une implémentation du protocole Xplore.

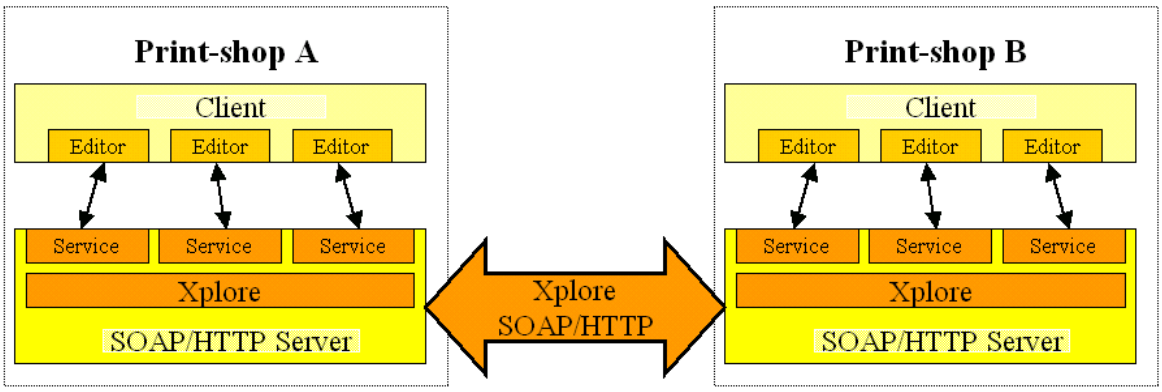

Figure 41. Architecture logicielle

Par la suite, nous décrirons chaque partie de l'architecture (Editor, NegF, les services et le Xplore) et comment ces implémentations emploient les propriétés du protocole SOAP.

11.2.1. Editor

L'Editor est utilisé pour contrôler la négociation d'un participant en lui permettant d'agir sur ces différents graphes de négociation. Donc, à partir de cette interface le Manager d'un atelier dans le cadre de l'architecture développée) est capable de choisir à négocier dans un **Service** particulier et ce service va créer une instance d'un graphe (*Invocation*) pour chaque nouvelle négociation.

L'Editeur représente la vue du NegF sur la totalité des négociations où son Manager est impliqué. A travers l'Editor**,** le Manager est capable d'agir sur un seul graphe de négociation à la fois, mais dans le même temps il a la possibilité de naviguer entre les graphes de négociation existants et de choisir n'importe lequel pour continuer une négociation.

Dans Fig. 42 on peut identifier :

- dans le panel gauche : le nom de l'atelier (A0), le nom de l'instance du service en déroulement (Out1_0) et les services Outsrc et Insrc avec les instances existantes ;
- dans le panel du milieu : l'image du graphe bicolore que l'instance choisie a sur la négociation en cours;
- dans le panel droit : trois fenêtres pour visualiser les différentes données attachées dans un nœud blanc (les attributs instanciés - asserted, les attributs demandés d'être instanciés – requested et les connexions faites à partir du nœud sélecté - connected).

Le panel du milieu est une implémentation graphique qui offre les actions spécifiques à un *viewer* (sélection des nœuds, modifications sur le placement et les dimensions géométriques d'un ou plusieurs nœuds, …etc.) et à un *controler* qui facilite les modifications de la structure du graphe en utilisant le protocole Xplore (la structure du graphe de négociation est modélisé avec les verbes Xplore – open, assert, request, quit, accept, connect).

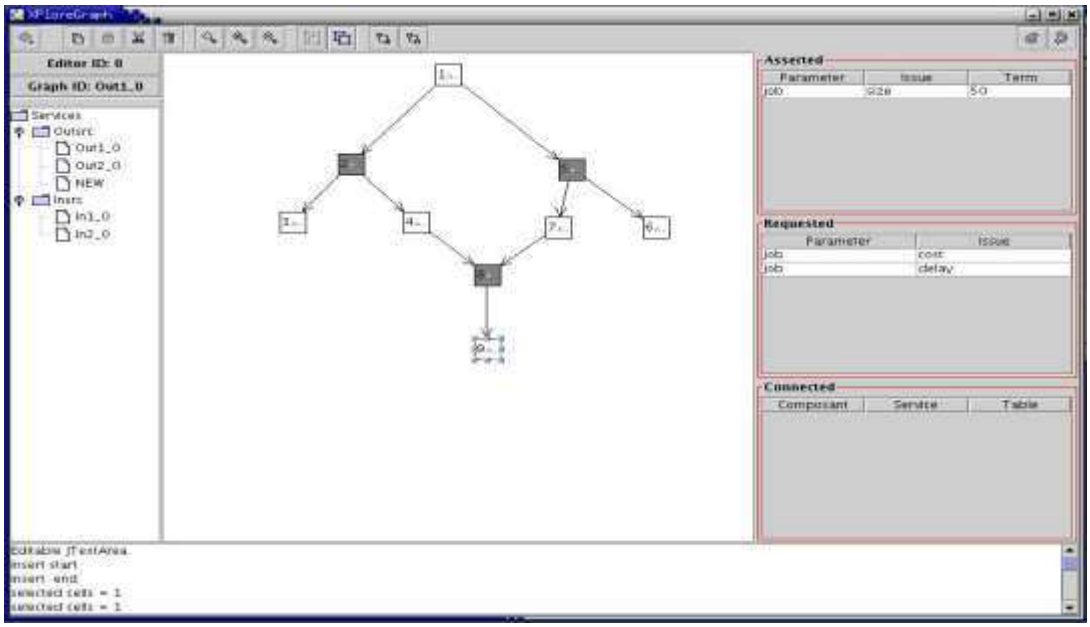

Figure 42.

En plus nous avons enrichi la partie client de l'application, avec l'implémentation des plusieurs interfaces pour la création et la manipulation des différents Cadre de Négociation ou Objet de Négociation qu'un Manager peut spécifier et aussi pour la Représentation des Autres qu'un atelier peut construire.

Pour chaque structure nous avons spécifié et implémenté deux interfaces :

i) une, pour visualiser l'ensemble constitué par toutes les structures du même type (par exemple dans la Fig. 43 nous avons l'image de l'ensemble des objets des négociations) ;

Figure 43

ii) une autre, pour la création ou la modification des champs correspondants au type de structure concernée (dans la Fig. 44 nous avons l'image de la création d'un nouvel objet de négociation).

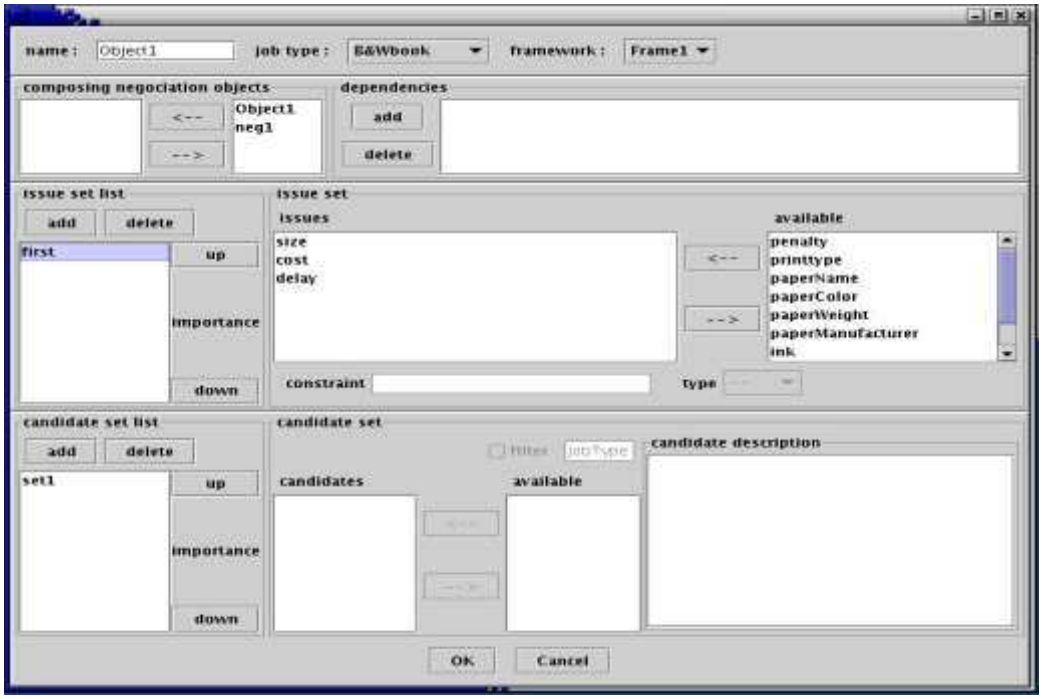

Figure 44

Comme nous l'avons décrit ci-dessus, dans l'implantation actuelle, c'est l'application appelée **Editor**, qui joue le rôle d'interface entre le Manager et les services proposés par

l'infrastructure E-Alliance et mise en œuvre au sein du Serveur. Avec l'introduction de ces interfaces, l'utilisateur (le Manger) peut intervenir dans l'interaction pour introduire ses conditions au niveau de Objet de Négociation et Cadre de Négociation et aussi pour faire des propositions dans une négociation particulière.

Brièvement, l'Editor (cf. Fig. 45) est mis en application par un module de représentation graphique (*Graph Representation Module*) afin de manœuvrer un graphe de négociation et par un module de communication (*Communication Module*) afin de permettre l'interaction avec les autres services via les différentes méthodes proposées dans les **SOAP-Services**.

Figure 45. Architecture Editor

Graph Representation Module

Le *Graph Representation Module* gère un ou plusieurs *Graph View* (un pour chaque conversation :

- Une *conversation* est représentée par un *Negotiation Graph*
- Un *Negotiation Graph* est un graphe bicolore orienté, modélisant la structure topologique du graphe de négociation présentée dans le modèle conceptuel. Le *Negotiation Graph* est implémenté dans l'Editor utilisant la librairie *Jgraph*. Egalement, tous les verbes Xplore - open(n,n1,...,np), assert(n,v,a,t), request(n,v,a), quit(n), ready(n), connect(n,m) - sont disponibles dans la construction d'un graphe de négociation.

Le Graph Representation Module est structuré en trois parties : *Model*, *View* et *Controler*.

Le Model est composé par plusieurs classes implémentant la structure *Negotiation Graph* et les mécanismes de négociation sur le graphe bicolore. Donc, un graphe de négociation est modélisé par un objet *GraphModel* contenant plusieurs instances d'objet *GraphCell* afin de modéliser les nœuds et aussi les arcs (par défaut une instance du *GraphCell* est un nœud). Le Model contient aussi les informations sur les liaisons hiérarchiques entre les nœuds d'un graphe, i.e. liaisons de type parent-enfant. Afin de retrouver ces informations, plusieurs méthodes sont proposées :

- `getSource` et `getTarget`, retourne le nœud source et le nœud cible d'un arc. Un arc est également l'enfant du nœud source, cela permet d'avoir plusieurs arcs partant d'un même nœud.
- `getChild`, `getChildCount`, `getIndexOfChild`, et `getParent` methods. Les objets qui n'ont pas des parents sont des racines et elles peuvent être retrouvées avec les méthodes `getRootAt` et `getRootCount`.

Le View contrôle la disposition de la représentation du modèle géométrique du graphe de négociation et il contrôle aussi la mise à jour du contexte graphique qui fait la liaison entre le modèle (**Model**) et la vue (**View**). Les objets de type *GraphView* et de type *CellView* composent la vue d'un graphe. Les instances de *CellView* sont équivalentes aux cellules du graphe *GraphCell* de la représentation du **Model**. L'objet *GraphView* gère l'ensemble des cellules, une pour chaque nœud ou arc dans le modèle.

Le Controler gère le processus de rendering en spécifiant les étapes de l'édition et la manipulation des cellules et aussi les autres actions de type look-and-feel comme sélection ou déplacement des noeuds. L'interface se compose par :

- un menu supérieur d'où l'utilisateur peut envoyer (Publish button) et recevoir (Update button) les changements pour un graphe de négociation,
- un panneau central pour visualiser un graphe de négociation. À partir de ce panneau les actions autorisées sur le graphe de négociation sont faites par un menu de type pop-up qui devient actif seulement si la souris est sur un noeud.
- un panneau gauche où les asserts/requests déjà faits sont visualisés et les nouveaux assert/request pour un noeud choisi sont édités.

Communication Module

Afin de réaliser les transfères des données de et vers le serveur, chaque **Editor** dispose d'un **Communication Module.** Le module de communication est implémenté par deux classes *CommunicationModule.java* et *SoapServiceCall.java.*

L'objet *CommunicationModule* offre des méthodes utilisées par le **Graph Representation Module** pour envoyer et recevoir différentes données des SOAP-services et aussi d'invoquer les différentes méthodes proposées par ces services. Le *CommunicationModule* contient:

```
// object used to make the SOAP invocations
  private SoapServiceCall soap;

// used to stoke the data received from server
  private Node updateNode;

// thread used in order to listed if there is new data
  private CommunicationHandlerThread comThread;
```

Les méthodes utilisées pour invoquer les SOAP-services sont :

register() // is the first invoked method in order to get the id for the current editor;

getServices() // asks which are the Xplore services existing on the server. This method returns a string whose tokens correspond to the name of the different Xplore services. By default this method returns at the names of Outsrc and Insrc services as the only existing Xplore services;

getInvocations(String serviceName) // asks which are the graphs existing in the service described by serviceName, receives a list of idGraph (String);

getGraph(String idGraph) // demands the description of the graph with the name idGraph. This method returns the description of the graph starting with the root node;

addData(String id_Graph, Node node) // sends the modifications

made by the user, the sent data as Node structure. The node sent is the highest node in the graph hierarchy with a modified data.

`isNewData()` // listens for data changes on the server, the return is blocked until new data arrives on the server.

`getData(String id_Graph)` // gets the new data from the server. The data returned is described as a Node structure.

Pour faciliter l'invocation plus facile des SOAP-services, nous avons implémenté un objet *SoapServiceCall* qui cache les appelles SOAP. Pour faire un appelle d'une méthode d'un SOAP-service nous avons défini deux méthodes :

`void init(String serviceName)` // Initializes the SOAP Service Facade, the encoding style (SOAP envelope) and the URI of the SOAP service

`Object invoke(String methodName, Vector params)` // Remote Method Invocation, needs the name of method to be invoked, list of parameters to be passed and returns the object received from the server

11.2.2. NegF – Services – Xplore

Suivant notre modèle architectural nous avons mis en application la NegF, les services de coordination et l'intergiciel Xplore du côté du serveur. Selon notre modèle conceptuel de processus de négociation, une négociation est modélisée comme une construction d'un graphe de négociation de manière collaborative entre les participants d'une négociation. Chaque participant est représenté par un ou plusieurs services de négociation qui essaient de synchroniser la construction du graphe en utilisant les primitifs Xplore. Pour faire ceci, nous avons dû mettre en application une structure des données afin de décrire le graphe de négociation manœuvré par une instance d'un service de coordination et nous avons dû également mettre en application la communication de cette instance du service, d'une part avec l'Editor et d'autre part, avec les autres services qui peuvent être sur le même serveur ou sur d'autres serveurs. La communication est encapsulée dans des invocations des méthodes proposées par des SOAP-services.

Par la suite, nous présenterons d'abord la structure des données employées pour décrire un graphe de négociation suivi par l'implémentation des SOAP-services.

11.2.2.1. Les structures des données

Un atelier d'impression peut être impliqué, dans le même temps, dans beaucoup de négociations. Selon le nombre d'invocations des services impliqués dans le processus de négociation, chaque négociation est maintenue comme un ensemble des structures de données de type graphe. Finalement, chaque graphe est une structure composée principalement des nœuds noirs ou blancs et des données attachées à ces nœuds. Donc, nous avons proposé deux structures de données : *Graph.java* et *Node.java*.

Un objet de type Graph.java contient la description d'un graphe Xplore. Il est composé d'une instance de l'objet *Node* représentant la racine du graphe et d'un ensemble des objets *Node* représentant les nœuds du graphe dans une relation de type parent-enfant. Cette image de type graphe représente la vue qu'une instance d'un service gère à un moment donné. Egalement cela suppose une synchronisation avec les autres instances des services impliqués dans la négociation courante. Afin de garder la trace sur les autres instances impliquées dans la même négociation, chaque objet *Graph* contient aussi l'ensemble des identifiants des autres instances gérant le même graphe de négociation

La structure de l'objet Graph est la suivante :

```
public class Graph
{
    private Node root;
    private int nodesNumber;

    private String id; // the name of a graph is composed by: the
                       service's name + a number that corresponds to how many graphs
                       were created for the current client + the name of the client for
                       which it was created. (E.g.: Out2_3 – represents the 2-th graph
                       in the Outsrc service for the client 3).

    private Node[] nodesList;   // the list of all the nodes that compose
```

```
                    the currant graph;
        private String[] relatedGraphs; // the name of the graphs which
                    have the same node structure but not the same data in the nodes.
}
```

La structure d'un nœud contient des données identifiant les caractéristiques structurelles du nœud (l'identifiant du nœud, l'identifiant du graphe, …) et aussi les caractéristiques des nœuds Xplore (couleur, données assertées, …) :

```
String idNode;
String idGraph;
String idClient;
String color;
private final  int Max_Related 20;
String parents[Max_Related];
Node offsprings[Max_Related];
private final  int Max_Data 50;
String assertList[Max_Data];
String requestList[Max_Data];
```

Ces structures des données seront utilisées par les SOAP-services dans le processus de coordination et de synchronisation de la communication entre les instances des services impliqués dans différentes négociations.

11.2.2.2. SOAP-services

Les différents SOAP-services mis en application fourniront des méthodes privées afin de traiter les différents ensembles des invocations des services et des méthodes accessibles pour soutenir la communication et la synchronisation entre une instance de service et un client (Editor) ou entre différentes instances.

Nous avons proposé trois SOAP-services représentés par les classes suivantes : *HandlerService, NotificationService* et *DataService*. Par la suite, nous présenterons plus en détail seulement le *DataService*, parce que c'est le SOAP-service où tout le mécanisme de

négociation (NegF-Services-CooF) est mis en application, les deux autres SOAP-services sont utilisés seulement pour maintenir une synchronisation entre l'image d'un graphe de négociation du coté client (Editor) et celle contrôlée par les services de coordination du coté serveur.

Du côté du serveur SOAP, différentes instances de la classe Graphe sont contrôlées en même temps. Le nombre de graphes dépend du nombre des clients enregistrés sur le serveur, donc du nombre des Editors qu'un atelier d'impression utilise, du nombre des négociations dans lesquelles l'atelier participe et finalement du nombre des instances des services de coordination impliquées dans les négociations considérées. Les structures de données utilisées pour enregistrer ces informations sont les suivantes :

- *Client structure*

```
public class Client{ // an instance of this object is created for each
                     component that registered to the server.
   String idClient;
   LinkedList outsrc; //list of the Xplore Outsrc's invocations accessible for
this client;
   LinkedList insrc;  //list of the invocations Xplore Insrc's accessible for
this client;
   LinkedList asABlock; //list of the Xplore AsABlok's invocations
accessible for this client;
   LinkedList broker;  //list of the invocations Xplore Broker's accessible
for this client;
   LinkedList split; //list of the Xplore Split's invocations accessible for this
client;
   LinkedList swapOut;    //list of the invocations Xplore SwapOut's
accessible for this client;
   LinkedList swapIn;  //list of the invocations Xplore SwapIn's accessible
for this client;
   LinkedList transp;  //list of the invocations Xplore Transp's accessible
for this client;
   Boolean newData;
}
```

- *Map of invocations*

`public Hashtable mapGraphs = new Hashtable();` // this hashtable contains all the invocations created by the components registered to this server. The access key for an invocation is the id of the graph structure created during a Xplore service invocation.

- *Vector of clients*

`public Vector vectorClient = new Vector(20,10);` // this vector contains all the registered clients

- *Map of online Graphs*

`public Hashtable mapInvocations = new Hashtable()` // this hashtable contains all the invocations that are currently in use by the clients. The access key is the name of the invocation and the data returned is the id of the client.

Le SOAP-service DataService propose plusieurs types des méthodes afin de faciliter: la communication entre client et serveur, la manipulation des structures du type de graphe de la négociation et les mécanismes de coordination et de synchronisation offerts par les services de coordination et le protocole Xplore.

Dans la Fig 46 les invocations des méthodes sont présentées et proposées par DataService pour enregistrer un client :

`register()` // this is the first method that a client invoke in order to get an unique Id,

`deregister(Id_Client)` // deletes the Id of the client from the server list,

`getServices()` // get the different coordination services that are proposed (e.g.: outsrc, insrc, ...). This method returns a string whose tokens correspond to the name of the different Xplore services.

`getInvocations(String id_Client, String serviceName)` // get the different graphs in each service. This method returns a string whose tokens correspond to the ids of the different Xplore services

`getGraph(String id_Client, String id_Graph)` // get the specified graph.

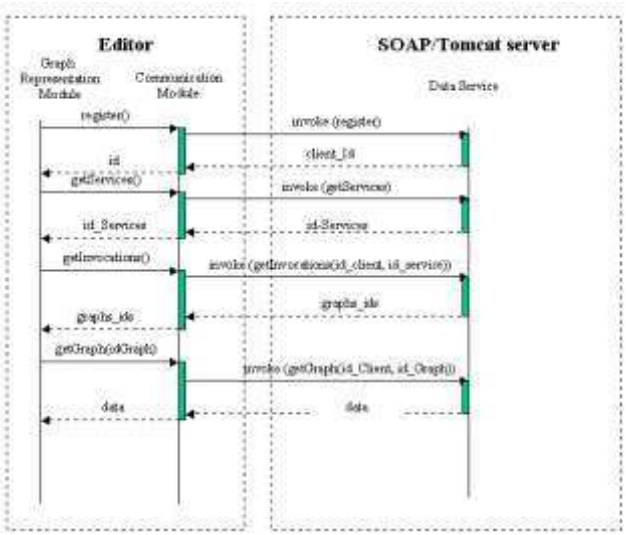

Figure 46

Comme nous avons spécifié dans notre modèle d'architecture, la communication entre les instances des services impliquées dans une négociation est faite seulement en utilisant les verbes Xplore. Donc, le DataService fournit aussi les méthodes afin de faciliter la communication entre les instances qui se trouvent sur le même serveur mais aussi entre des instances qui se trouvent sur des serveurs différents :

```
open (String idGraph, String idNode, String idsParents);
assert(String idGraph, String idNode, String param, String issue,
String term);
request(String idGraph, String idNode, String param, String issue);
quit(String idGraph, String idNode);
ready(String idGraph, String idNode);
connect(String idGraph, String idNode; String idService; String map);
```

La communication entre les instances des services suit les caractéristiques du protocole Xplore. Ainsi, les interactions sont mises en application comme les invocations de RPC qui ont la signature des différents verbes Xplore.

Voici un exemple d'une invocation RPC utilisant le format de message SOAP et qui représente la description des verbes du Xplore en utilisant le XML-Schéma (voir Annexe). Seulement la partie du corps de l'enveloppe de la demande de SAVON est montrée ici :

E.g. The assert verb description is:

```
<element name= "assert" base="tns:assert"/>
<complexType name="assert">
<element name= "node" type="tns:node"/>
<element name="parameter" type="tns:parameter"/>
<element name="issue" type="tns:issue"/>
<element name="term" type="tns:term"/>
 </complexType>
```

The method description will be :

```
<SOAP-ENV:Body>
     <m:assert  xmlns:m="some-URI">
           <node>id_node</node>
           <parameter>parameter_name</parameter>
           <issue>issue_name</issue>
           <term>term_name</term>
     </m:assert>
</SOAP-ENV:Body>
```

Le modèle et l'implémentation que nous avons proposés mettent en application de manière complète le processus de négociation comprenant la description des objets de négociation, les mécanismes d'envoyer des propositions et des contre-propositions, les modules de coordinations et les mécanismes de synchronisation. Cette implémentation ajoute des divers équipements dans la construction et l'exécution des conversations complexes multi-agent. L'intégration des technologies d'enchaînement, par exemple, HTTP, SOAP, XML/XSL et Servlet, permet d'attacher facilement des agents qui effectuent des conversations avec d'autres agents ou utilisateurs humains au-dessus du Web.

12. Conclusion

Nous avons présenté dans ce chapitre la mise en œuvre de notre modèle de coordination des négociations en proposant une architecture structurée en trois couches principales : *la « facility » NegF*, *les services* et *l'intergiciel CooF*. Cette structuration respecte notre approche sur la division du processus de négociation en trois processus distincts : *processus de décision, processus de coordination* et *processus de communication*.

Le processus de communication est géré par la couche intergicielle CooF qui définit des mécanismes génériques de communication et de synchronisation entre plusieurs agents. Les principales qualités qui ont justifié le choix de ce protocole Xplore est le fait qu'il permette de modéliser et de gérer notre modèle de négociation multi-phase où à tout moment les participants peuvent continuer la négociation concurremment sur plusieurs phases de négociation.

Le processus de coordination est géré par une couche des services de coordination. La caractéristique principale de cette approche est le fait que le processus de coordination soit complètement distribué entre des briques basiques de coordination (services). Cela à permis la définition des services spécialisés et qui peuvent être utilisés dans n'importe quelle négociation. Egalement, avec cette distribution des contraintes de coordination, plusieurs services peuvent fonctionner simultanément afin d'augmenter l'efficacité du modèle en termes de nombre des choix de négociation à explorer dans le même temps.

Avec la mise en place des deux classes de services, une pour la coordination en monde fermé et une pour la coordination en monde ouvert, nous avons géré dans cette couche les contraintes de coordination existantes au niveau d'une seule négociation mais aussi les dépendances entre toutes les négociations d'un agent.

Donc, le processus de décision intègre seulement des mécanismes de raisonnement sur le comportement à utiliser lors d'une négociation. Cette couche agent gère donc les choix sur les stratégies à utiliser afin d'évaluer ou générer de nouvelles propositions, les protocoles à suivre afin d'arriver à faire parvenir aux autres agents les propositions ou aussi d'utiliser des informations sur les priorités des possibles partenaires et des attributs à négocier. L'objectif de

l’implémentation du logicielle à ce niveau à été de permettre à l’utilisateur humain de paramétrer et d’intervenir dans le processus décisionnel. Cela nous permet d’abstraire le processus décisionnel des agents ce qui renforce la généricité de notre modèle de coordination.

V. Conclusion et perspectives

Comme nous l'avons indiqué en introduction, les travaux réalisés dans cette thèse visent à supporter la coopération entre les différentes négociations dans une alliance des entreprises. Notre recherche se place dans les domaines des alliances virtuelles, de la négociation automatique et de la coordination. Dans ce cadre large, notre objectif était de fournir des mécanismes de coordination des négociations déroulées entre des entreprises autonomes réunies dans une alliance virtuelle.

Différents travaux ont été réalisés sur le thème de la négociation automatisée et plusieurs systèmes de coordination des négociations ont été proposés. De l'étude des différentes formes de négociation, nous avons dégagé plusieurs caractéristiques groupées selon trois axes – *participant*, *contexte* et *processus de négociation* – ainsi que les différentes propriétés – nombre des participants, nombre des attributs négociés et nombre des phases de négociation – qui en fonction de leurs valeurs influencent les négociations effectuées. Ces caractéristiques et ces propriétés nous ont fourni une base pour la description de notre modèle de négociation de type *multi-bilatérale*, *multi-attribut* et *multi-phase*.

Une autre dimension de la contribution scientifique de cette thèse concerne les modèles de coordination des négociations. En proposant la négociation comme mécanisme de base pour toute activité de collaboration au sein de l'alliance, nous avons été amenés à faire une distinction entre les multiples aspects par le thème de la négociation :

- processus générique de propagation et d'échange d'alternatives comme support à la négociation,
- mise en évidence de mécanismes de base de coordination venant s'interfacer avec ce processus générique pour mettre en place des schémas de coordination sophistiqués et gérer l'évolution concurrente de plusieurs négociations,
- modélisation de la composante stratégique impliquée dans le pilotage de ces mécanismes de base pour une aide à la décision et à la négociation.

Notre modèle de négociation sépare les processus de communication, de coordination et de décision qui sont indépendants les uns des autres au sein d'une application de négociation. L'indépendance du processus de coordination par rapport au processus de décision est le résultat du fait que nous avons séparé deux niveaux de coordination – une coordination

stratégique et une coordination générique -. Ces deux niveaux de coordination nous ont permis d'identifier les contraintes de coordination qui peuvent être traitées sans avoir besoin des mécanismes de raisonnement. L'élimination des mécanismes de raisonnement fait que le processus de coordination est indépendant de la stratégie de négociation ou du protocole de négociation, s'adaptant donc à différents types de négociations.

Les autres qualités principales de notre modèle de coordination, autres que la *généricité*, sont la *décentralisation* et la *flexibilité*. Ces propriétés sont le résultat du fait que le modèle de coordination est structuré dans des briques (schémas de coordination) qui regroupent des cas basiques de coordination basés sur des relations des dépendances statiques ou découvertes dynamiquement.

Finalement, dans les présentations des différents systèmes de coordination des négociations nous avons constaté que la majorité des mécanismes de coordination utilisés essaye de résoudre un problème de manière distribué mais avec complétée coopération entre les entités coordonnatrices (agents), ignorant donc la caractéristique d'autonomie d'un agent. Dans notre modèle nous préservons l'autonomie totale de l'agent pour ces décisions et ces contraintes de coordinations.

Perspectives

Une première perspective qui peut être soulignée est de continuer la modélisation du processus de négociation et du processus de coordination avec la prise en compte du processus de gestion des contrats. De cette manière la coordination peut gérer non seulement des dépendances entre des négociations et les contrats qui se forment mais aussi des dépendances sur l'exécution de ces contrats.

Dans notre approche sur la coordination générique nous avons fait abstraction du protocole de négociation et nous avons essayé de délimiter les dépendances entre des négociations qui ne dépendent pas des protocoles utilisés. Une étude possible est aussi de fournir à l'utilisateur un outil lui permettant de définir le protocole de négociation par des contraintes sur les possibles interactions faites au cours d'une négociation. Sera par la suite un problème de coordination que l'infrastructure doit résoudre afin de construire et gérer le protocole de négociation.

Du point de vue du niveau stratégique, nous envisageons d'étudier de manière plus approfondie les stratégies de négociation. Le but de cette thèse étant de proposer un modèle

générique de coordination des négociations, nous n'avons pas traité plus en détail le niveau stratégique de la négociation. Nous sommes arrivées à proposer des stratégies basiques de négociation (tactiques) basées sur des ensembles des contraintes gérer par le processus de coordination, mais nous n'avons pas proposé des mécanismes de composer efficacement ces tactiques et d'arriver à identifier les stratégies de négociation optimales selon les différents cas de négociation.

Finalement, nous avons conçu l'application actuelle dans le cadre du projet E-Alliance partant d'un cas d'alliance des ateliers d'impression. L'un des objectifs recherchés est celui de généricité et donc de pouvoir utiliser cette infrastructure dans d'autres contextes. C'est pourquoi, une perspective envisagée est celui-là de tester ce modèle et l'infrastructure proposée dans d'autres contextes.

VI. Références

[Allen et Hayes'85] Allen, J.F. and Hayes, P.J., A common-sense theory of time, in: *Proceedings IJCAI-85,* Los Angeles, CA, 528-531, 1985.

[Alty et al.'94] J. A. Alty, D. Griffiths, N. R. Jennings, E. H. Mamdani, A. Struthers, and M. E. Wiegand: "ADEPT - Advanced Decision Environment for Process Tasks: Overview & Architecture", Proc. BCS Expert Systems Conference (Applications Track, ISIP Theme), 359-371,Cambridge, UK, 1994.

[Andreoli'96a] J.-M. Andreoli, "Object Orientation with Parallelism and Persistence", chapter Coordination as negotiation transactions. Kluwer Academic Publishers, 1996.

[Andreoli'96b] J.-M. Andreoli, "Coordination in LO". In Coordination Programming: Mechanisms, Models and Semantics. Imperial College Press, 1996.

[Andreoli'96c] J.-M. Andreoli, J.-L. Meunier and D. Pagani. "Process Enactment and Coordination". In Proc. of European Workshop on software Process Technology (EWSPT'96), France, 1996.

[Andreoli et al.'99] J.-M. Andreoli, D. Arregui, F. Pacull, M. Riviere, J.Y. Vion-Dury and J. Williamowski, "CLF/Mekano: A Framework for Building Virtual-Enterprise Applications". In Proc. of EDOC, Mannheim, Germany, 1999.

[Andreoli et al.'00] J.-M. Andreoli, S. Castellani and M. Munier, "AllianceNet: Information Sharing, Negotiation and Decision-Making for Distributed Organizations". In Proc. of EcWeb, Greenwich, U.K., 2000.

[Andreoli et Castellani'01] J.-M. Andreoli and S. Castellani, "Towards a Flexible Middleware Negotiation Facility for Distributed Components". In Proc. of "E-Negotiation" DEXA, Munich, Germany, 2001.

[Ballmer et al.'81] T. Ballmer and W. Brennenstuhl. "Speech Act Classification : A Study in the Lexical Analysis of Speech Activity Verbs", Springer Verlag, 1981.

[Bamford et al.'03] Bamford James D., Benjamin Gomes-Casseres et Michael S. Robinson, "Mastering Alliance Strategy: A Comprehensive Guide to Design, Management, and Organization", San Francisco: Jossey-Bass, 2003.

[Barbuceanu et Lo'00] Barbuceanu, M. et Lo, W. 2000. A multi-attribute utility theoretic negotiation architecture for electronic commerce. In *Proceedings of the Fourth*

international Conference on Autonomous Agents (Barcelona, Spain, June 03 - 07, 2000). AGENTS '00. ACM Press, New York, NY, 239-246.

[Barbuceanu et Fox'97] M. Barbuceanu and M. Fox. "The design of a coordination language for multi-agent systems". In Intelligent Agents III: Agent Theories, Architectures and Languages (ATAL 96), no. 1193, pg. 341-357, Springer Verlag, 1997.

[Barbuceanu et Fox'98] M. Barbuceanu and M. Fox. "The specification of Cool: A language for representation cooperation knowledge in multi-agent systems". Technical report, Internal Report, EIL, Univ. of Toronto, 1998.

[Bartolini et Priest'01] C. Bartolini and C. Preist, "A framework for automated negotiation", Technical Report HPL-2001-90, HP Laboratories Bristol.

[Bartolini et al.'02] C. Bartolini, C. Preist and N. Jennings, "A generic software framework for automated negotiation", Technical Report HPL-2002-2, HP Laboratories Bristol.

[Bass'03] Jossey Bass, *Bridging Cultural Conflicts: A New Approach for a Changing World. San Francisco*, 2003. available at http://www.beyondintractability.org/m/culture_negotiation.jsp

[Bauer'00] B. Bauer, J. Muller, and J. Odell. "An extension of UML by protocols for multiagent interaction". In 4th International Conference on Multi-Agent Systems (ICMAS 00), pg. 207-214, Boston, USA, 2000.

[Bayardo'97] R.J. Bayardo, "InfoSleuth: Agent-Based Semantic Integration of Information in Open and Dynamic Environments". In Proceedings of the ACM SIGMOD International Conference on Management of Data, 1997.

[Bergstresser et Yu'77] K. Bergstresser and P.L. Yu. Domination Structures and Multicriteria Problems in N-Person Games. *Theory and Decision, vol.8, no. 1, pp 5-48, 1977.*

[Berthet et al.'92] S. Berthet, Y. Demazeau and O. Boissier. "Knowing each other better". In Proc. of the 11th International Workshop on DAI, Glenn Arbor MI, USA, Feb. 1992.

[Boulding'89] Kenneth Boulding, "*Three Faces of Power"*, Newbury Park, CA: Sage Publications, 1989.

[Bradshaw et al.'97] J. Bradshaw, S. Dutfield, P. Benoit, and J. Wooley. "Kaos: Toward an industrial-strength open agent architecture". Technical report, In Software Agents, AAAI Press/MIT press, 1997.

[Bratu'01] Mihnea Ben Bratu, Master Thesis, "Vers une infrastructure multi-agents d'aide à la négociation pour des alliances d'entreprises", Saint-Etienne, 2001.

[Bretier'95] P. Bretier. "La communication orale coopérative : contribution à la modélisation logique et à la mise en œuvre d'un agent rationnel dialoguant". Thèse de doctorat, Université de Paris Nord, LIPM (Paris XIII), 1995.

[Brussel et al.'93] Brussel, H. V. et Valckenaers, P. 1993. Information Infrastructure Requirements for Preserving Flexibility of Flexible Manufacturing Systems. In *Proceedings of the JSPE/IFIP Tc5/Wg5.3 Workshop on the Design of information infrastructure Systems For Manufacturing* (November 08 - 10, 1993). H. Yoshikawa and J. Goossenaerts, Eds. IFIP Transactions, vol. B-14. North-Holland, 209-224.

[Brooks'86] Brooks, R.A., "A Robust Layered Control System for Mobile Robot", IEEE Journal of Robotics and Automation, 1986. RA-2(1).

[Bui et Kowalczyk'01] V. Bui and R. Kowalczyk. On constraint-based reasoning in e-negotiation agents. *In Agent-Mediated Electronic Commerce, Lecture Notes in Artificial Intelligence, Springer Verlag, 2001.*

[Calantone et al.'98] R.J. Calantone, J.L. Graham and A.M. Wimsatt, "Problem Solving Approach in an International Context – Antecedents and Outcome". In International Journal of Research in Marketing, vol. 15, pp. 19-35, 1998.

[Carrieri et Gelernter'92] N. Carriero and D. Gelernter. How to write parallel programs, a first course. *MIT Press, 1992.*

[Carrieri et Gelernter'89] N. Carriero and D. Gelernter. "Linda in Context". In Communications of the ACM, no. 32(4), pg. 444-458, 1989.

[Castefranchi'95] C. Castefranchi: *Guarantees for Autonomy in Cognitive Agent Architecture*. In N. Jennings and M. Wooldridge (eds.) Agent Theories, Architectures, and Languages, Heidelberg, Spinger-Verlag, 1995.

[Chaib-draa et al.'92] Chaib-draa, B., Moulin, B., Mandiau, R. & Millot, P., "Trends in Distributed Artificial Intelligence", *Artificial Intelligence Review* **6**, 35-66(1992).

[Chang et Woo'94] Chang, M. K. et Woo, C. C. 1994. A speech-act-based negotiation protocol: design, implementation, and test use. *ACM Trans. Inf. Syst.* 12, 4 (Oct. 1994), 360-382.

[Chang et Woo'93] M. Chang and C. Woo." SANP : A communication level protocol for negotiation". In Decentralized Artificial Intelligence II – Proceedings of the 2nd European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'91), pg. 31-54, 1992.

[Chen et al.'99]Ye Chen, Yun Peng, Tim Finin, Yannis Labrou and Scott Cost. A negotiation-based Multi-agent System for supply chain management. *Technical Report 1999.*

[Cockburn et Jennings'96] D. Cockburn and N.R. Jennings, "ARCHON: A Distributed Artificial Intelligence System for Industrial Applications". In Foundations of Distributed Artificial Intelligence, O'Hara and N.R. Jennings (eds.), pg. 319-344, 1996.

[Cohen et Perrault'79] P. Cohen and C. Perrault. "Elements of a plan based theory of speech acts". Cognitive Science, (3) : 177-212, 1979.

[Coo'94] Proceedings of the first annual Workshop on Coordination, Imperial College, London, 1994.

[Conry et al.'92] S. Conry, K. Kuwabara, V. Lesser and R. Meyer, "Multistage Negotiation in Distributed Constraint Satisfaction". IEEE Transactions on computers, C-21(6):1462-1477.

[Cordoso et Oliveira'00] Henrique Lopes Cardoso and Eugénio Oliveira, "A Platform for Electronic Commerce with Adaptive Agents", in *AA'2000 Agent-Mediated Electronic Commerce III* , Frank Dignum and Ulises Cortés (eds.), LNAI 2003, pp.96-107,Springer-Verlag, 2000;

[Dammann'03] Catherine Dammann , PhD. Thesis, "Job Definition Format – A Catalyst for Electronic Business Data Exchange in the Commercial Printing Industry", University of Applied Sciences Bielefeld, 2003.

[Decker et al.'97] Keith Decker, Katia Sycara, Mike Williamson, "Middle-Agents for the Internet". In Proceedings of the 15th International Joint Conference on Artificial Intelligence, 1997.

[Demazeau et al.'94] Y. Demazeau, O. Boissier and J.L. Koning. " Using interaction protocols to control vision systems". In IEEE International Conference on Systems, Man and Cybernetics, San Antonio, Texas, Oct; 1994.

[Deutsch'00] Morton Deutsch, "Cooperation and Competition," in *The Handbook of Conflict Resolution: Theory and Practice*, eds. Morton Deutsch and Peter Coleman, San Francisco: Jossey-Bass Publishers, 2000.

[El Fallah et Haddad'01] Amal El Fallah and Serge Haddad. "A recursive model for distributed planning". In Informatique et systèmes d'information, pg. 139-176, 2001.

[Emerson'90] Emerson, E. A. , "Temporal and modal logic". In van Leeuwen, J. (ed.), *Handbook of Theoretical Computer Science*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, 996-1072,1990.

[Ephrati'94] Eithan Ephrati, "Divide and Conquer in Multi-agent Planning ", In Proc. of National Conference on Artificial Intelligence, 1994.

[ebay] eBay: http://pages.ebay.com.

[E. Wertheim] E. Wertheim, "Negotiations and Resolving Conflicts: An Overview," College of Business Administration, Northeastern University. Available at: http://web.cba.neu.edu/~ewertheim/interper/negot3.htm.

[Faratin'00] P. Faratin,(2000)«Automated service negotiation between autonomous computational agents», PhD. thesis, Department of Electronic Engineering Queen Mary & Westfield College;

[Faratin et al.'98] P. Faratin, C. Sierra and N. Jennings, (1998), « Negotiation decision functions for autonomous agents », Journal of robotics and autonomous systems, pp 159-182;

[Fatima et al.'04] S. Fatima, M. Wooldridge and N. R. Jennings, "Optimal negotiation of multiple issues in incomplete information settings". Proc. 3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems , New York, USA, 1080-1087, 2004.

[FIPA'99] FIPA, Agent Communication Language, Specification 2, 1999, http://www.fipa.org.

[FIPA'00] FIPA. "Fipa interaction protocol library specification". Technical Report XC00025D, Foundation for Intelligent Physical Agents, 2000.

[Fisher et Ury] R. Fisher et Ury "Getting to Yes!", eds. *Penguin Books*, 1981.

[Fisher'91] Roger Fisher, "Negotiating Power: Getting and Using Influence," pp. 127-140 in *Negotiation Theory and Practice,* eds. J. William Breslin and Jeffrey Z. Rubin, Cambridge: Program on Negotiation Books, 1991.

[Flores et Kremer'02] R. Flores and R. Kremer, "A Model for Flexible Composition of Conversations: How a simple Conversation got so Complicated", In AAMAS Workshop "Agent Communication Languages and Conversation Policies", Bologna, 2002.

[Foner'96] Leonard N. Foner, "A Security Architecture for Multi-Agent Matchmaking", In Proc. of Second International Conference on Multi-Agent Systems, 1996.

[Fornara et Colombetti'02] Fornara, N. et Colombetti, M. "Operational specification of a commitment based agent communication language". Dans Proceedings of the 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'02), pages 536–542, Bologne, Italie. ACM Press, 2002.

[Fornara et Colombetti'03] Fornara, N. et Colombetti, M. 2003. Defining interaction protocols using a commitment-based agent communication language. In *Proceedings of the Second international Joint Conference on Autonomous Agents and Multiagent Systems* (Melbourne, Australia, July 14 - 18, 2003). AAMAS '03. ACM Press, New York, NY, 520-527.

[Franklin et Grasser'96] Franklin, S. & Grasser, A. (1996). Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. *In Proceedings of the Third International Workshop on Agent Theories*, Architectures, and Languages, Springer-Verlag.

[Gasser'92] L. Gasser, "DAI Approaches to Coordination". In Distributed Artificial Intelligence: Theory and Praxis, Kluwer Academic Publishers, pg. 31-51, 1992.

[Gelernter'85] D. Gelernter. "Generative communication in Linda". In *ACM Transactions on Programming Languages and Systems*, no. 7(1), pg. 80-112, 1985.

[Gerber et al.'99] C. Gerber, J. Siekmann and G. Vierke, "Holonic Multi-Agent Systems", Technical report TR-99-03, DFKI GmbH, 1999.

[Girard'87] Girard, J.-Y. : *Linear Logic*, Theoretical Computer Science, London Mathematical 50:1, pp. 1-102, 1987.

[Governatori et al.'02] G. Governatori, J. Gelati, A. Rotolo, and G. Sartor, "Actions, Institutions, Powers. Preliminary Notes". In *Proceedings of RASTA'02*, pages 131–147. University of Hamburg, Hamburg, 2002.

[Gray'03a] Barbara Gray, "Negotiating With Your Nemesis", *Negotiation Journal October 2003, pg.* 299 – 315.

[Gray'03b] B. Gray, "Framing of environmental disputes". In R. Lewicki, B. Gray, & M. Elliott (Eds.), *Making sense of intractable environmental conflicts: Concepts and cases* (pp. 11-34), Washington, D.C.: Island Press, 2003.

[Greaves'00] M. Greaves. "Issues in agent communication: An introduction.". pg.1-16, Springer Verlag, 2000.

[Greaves et al.'00] M. Greaves, H. Holmback and J. Bradshaw. "What is a conversation policy?" In Issues in Agent Communication, volume 1916 of LNAI, pg. 118-131, Springer Verlag, 2000.

[Hall'93] Lavinia Hall, ed. London: Sage Publications. "*Negotiation: Strategies for Mutual Gain."* 1993.

[Huget'01] M. Huget. "Une ingéniérie des protocoles d'interaction pour les systèmes multi-agents". Thèse de doctorat, Univ. Paris Dauphine (Paris IX), 2001.

[IN] International Negotiation http://interneg.carleton.ca/in/

[Ito et al.'00] Ito, T. Fukuta, N. Shintani, T. Sycara, K. , "BiddingBot: a multiagent support system for cooperative bidding in multiple auctions", In Proceedings. Fourth International Conference, pg.399-400, 2000.

[J.C. Melamed] James C. Melamed, *Communication and Facilitation Skills*. The Mediation Center. Available at: http://www.internetmediator.com/divres/pg54.cfm

[Jennings'93] N.R. Jennings. "Commitments and conventions: The foundations of coordination in multi-agent systems". In The Knowledge Engineering Review, no.8(3), pg. 223-250, 1993.

[Jennings et al.'96] N. R. Jennings, P. Faratin, T. J. Norman, P. O'Brien, M. E. Wiegand, C. Voudouris, J. L. Alty, T. Miah, and E. H. Mamdani (1996) "ADEPT: Managing Business Processes using Intelligent Agents", in: Proc. BCS Expert Systems Conference (ISIP Track), Cambridge, UK 5-23, 1996.

[Jennings et Faratin'98] Jennings N. R., and P. Faratin . "Implementing a business process management system using ADEPT : A real-world case study ", 1998.

[Jennings et al.'00] N. R. Jennings, P. Faratin, T. J. Norman, P. O'Brien, B. Odgers and J. L. Alty "Implementing a Business Process Management System using ADEPT: A Real-World Case Study". Int. Journal of Applied Artificial Intelligence 14 (5) 421-465, 2000.

[Jennings et al.'02] N.R. Jennings, M. He and H.F. Leung, (2002), « A fuzzy logic based bidding strategy for autonomous agents in continuous double auctions ».

[Jones'03] A. Jones, "A Logical Framework". In J. Pitt, editor, Open Agent Societies: Normative Specifications in Multi-Agent Systems. Wiley, Chichester, 2003.

[Keeny et Raiffa'76] R. Keeny and H. Raiffa, "Decisions with Multiple Objectives: Preferences and Value Tradeoffs". Ed. JohnWilley and Sons, 1976.

[Kersten et Noronha'97] Gregory E. Kersten and Sunil J. Noronha, "Supporting International Negotiation with a WWW-Based System", INTERIM REPORT IIASA IR-97-49/August 1997.

[Kersten et Noronha'98] Gregory E. Kersten and Sunil J. Noronha, "Negotiation and the Web: Users' Perceptions and Acceptance", INTERIM REPORT IIASA IR-98-002 /March 1998.

[Kersten'01] Gregory E. Kersten . "Modeling Distributive and Integrative Negotiations. Review and Revised Characterization". In *Group Decision and Negotiation*, 2001, Vol. 10, No. 6, (493-514).

[Kersten et al.'99] G. E. Kersten and S. J. Noronha. Negotiation in Electronic Commerce: Methodological Misconceptions and a Resolution. *INR02 1999.*

[Kersten et al.'02] Gregory E. Kersten, Sabine T. Köszegi et Rudolf Vetschera, "The Effects of Culture in Anonymous Negotiations: Experiment in Four Countries", In Proceedings of the 35th Hawaii International Conference on System Sciences, 2002.

[Koch'98] T. Koch. "Groupware on the Internet". In PhD thesis at Technical University Graz, 1998.

[Koning'01] J. Koning. "Des règles d'interaction comme modèle de controle en univers multi-agent". Habilitation à diriger des recherches, Institut National Polytechnique de Grenoble, 2001.

[Kowalczyk et Bui'01] R.Kowalczyk et Van Bui,(2001) «On Constraint-Based Reasoning in e-Negotiation Agents», in AMEC III, pp. 31-46;

[Krause'95] Krause, P.; Ambler, S.; Elvang-Goransson, M.; and Fox, J., "A logic of argumentation for reasoning under uncertainty", Computational Intelligence 11 pg.113-131, 1995.

[Kraus'01] Sarit Kraus, (2001) « Strategic negotiation in multiagent environments », MIT Press.

[Kumar et al.'02] S. Kumar, M.J. Huber and P.R. Cohen. Representing and executing protocols as joint actions. *In Proc. Of AAMAS'02, Bolognia, Italy, pages. 543-550, 2002.*

[Labrou et Finin'97] Y. Labrou and T. Finin. A Proposal for a New KQML Specification. *Technical report, Univ. of Maryland, USA, 1997.*

[Lax et Sebenius'86] David Lax and James Sebenius in *The Manager as Negotiator: Bargaining for Cooperation and Competitive Gain*, 1986. Available at : http://www.negotiate.com/written-resources.html

[Lewicki et al.'99] Roy J. Lewicki, David M. Saunders and John W. Minton, "*Negotiation, 3rd Edition*", San Francisco: Irwin McGraw-Hill, 1999.

[Lewicki et al.'00] Lewicki, Roy J., David M. Saunders, and John W. Minton. *Essentials of Negotiation.* 2nd ed. Irwin, 2000.

[Lomuscio et al.'01] A.R.Lomuscio, M. Wooldrige et N.R. Jennings, (2001) «A classification scheme for negotiation in electronic commerce», in AMEC II, pp. 19-33.

[Lomuscio et al.'00] A. Lomuscio, M. Wooldrige, and N. Jennings. A classification scheme for negotiation in electronic commerce . *Technical report, In C. Sierra and F. Dignum, editors, A European Respective on Agent-Mediated Electronic Commerce, Springer Verlag, pages: 121-138, 2000.*

[Maes et Chaves'96] P. Maes, A. Chaves, (1996) «Kasbah, An agent marketplace for buying and spelling goods», ICPAIA, London, 1996.

[Mailler et al.'01] Roger Mailler, Regis Vincent , Victor Lesser, Tim Middlekoop et Jiaying Shen, "Soft Real-Time, Cooperative Negotiation for Distributed Resource Allocation",

in AAAI Fall Symposium on Negotiation Methods for Autonomous Cooperative Systems, November 2001.

[Malone et Crowston'94] T.W. Malone and K. Crowston. The Interdisciplinary Study of Coordination. *ACM Computing Surveys, 26(1): 87-119, March 1994.*

[Malone'97] Malone, T. W. "Is "Empowerment" just a Fad? Control, Decision-making, and Information Technology". *Sloan Management Review*, 38 (2), 23-35, 1997.

[Malone et al.99] Malone, T. W., Crowston, K. G., Lee, J., Pentland, B., Dellarocas, C., Wyner, G., Quimby, J., Osborn, C. S., Bernstein, A., Herman, G., Klein, M., & O'Donnell, E. "Tools for inventing organizations: Toward a handbook of organizational processes". *Management Science*, 45, 3 (March), 425-443, 1999.

[Meyer'88] R.A. Meyer, S.Conry and V. Lesser, "Multistage negotiation in distributed planning". In Bond, A. and Gasser, L. editors, "Reading in Distributed Artificial Intelligence", pages 367-386, 1998.

[Minsky et Ungureanu'97] N. Minsky and V. Ungureanu. Regulated coordination in open distributed systems. *In Proc. of Coordination'97, LNCS 1282, pages 81-97, Berlin, Germany, 1997.*

[Moehlman et al.'92] T. Moehlman, V. Lesser and B. Buteau, "Decentralized Negotiation: An Approach to the Distributed Planning Problem", *Group Decision and Negotiation*, 1(2), pg. 161-192, 1992.

[Moore'96] Moore, Christopher W. *The Mediation Process*. 2nd ed. Jossey-Bass, 1996.

[Moor et Woodrow'98] C. Moor and P. Woodrow, "Mapping cultures-strategies for effective intercultural negotiations", appeared in Track Two, vol.7, no. 1 April 1998.

[Mullen et Wellman'98] Tracy Mullen, Michael P. Wellman, (1998), "The auction manager: Market Middleware for Large – Scale Electronic Commerce ". The 3-th USENIX Workshop on Electronic Commerce;

[Muller'96] Muller, J-P. 1996. *The Design of Intelligent Agents: a Layered Approach*. Springer, Berlin.

[Muller'96] H. Muller. Negotiation principles. *In G. O'Hara and N. Jennings, editors, Foundations of Distributed Artificial Intelligence. 1996.*

[Newmann et al.'44] J. von Newmann and O. Morgenstern . Theory of games and economic behavior. *Princeton: Princeton University Press, 1944.*

[NIOTP] "Negotiation," International Online Training Program on Intractable Conflict, Conflict Research Consortium, University of Colorado, [available at: http://www.colorado.edu/conflict/peace/treatment/negotn.htm].

[NR] Negotiation Journal http://www.blackwellpublishers.co.uk

[Nwana et al.'97] H.S. Nwana, L. Lee and N.R. Jennings. Co-ordination in Multi-Agent Systems. *In H. S. Nwana and N. Azarmi, editors, Software Agents and Soft Computing, number 1198 in LNAI, Springer Verlag, 1997.*

[Nwana'99] Hyacinth S. Nwana, Divine T. Ndumu, Lyndon C. Lee, Jaron C. Collis, "ZEUS: A Toolkit for Building Distributed Multi-Agent Systems", (1999), Proceedings of the Third International Conference on Autonomous Agents (Agents'99).

[Oliveira et al.'97] Eugénio Oliveira, José M. Fonseca, Adolfo S. Garçao, "MACIV: A DAI Based Resource Management System", in *Applied Artificial Intelligence Journal* , V.11 N.6, pgs 525-550, Taylor& Francis, September 1997.

[Oliveira et Rocha'00] E. Oliveira and A. P. Rocha. "Agents advanced features for negotiation in Electronic Commerce and Virtual Organisations formation process". *In Agent Mediated Electronic Commerce: the European AgentLink Perspective, LNAI 1991, pp.78-97, Springer Verlag, 2000.*

[Oliveira'01] Eugénio Oliveira, "Agents' Advanced Features for Negotiation and Coordination". In *Proceedings of 9thECCAI Advanced Course, ACAI'01* , Eds. M.Luck, V.Marik, R.Trappl, O.Stepankova, LNAI2086, Springer, June 2001.

[Omicini et al.'03] A. Omicini, A. Ricci, G. Rimassa and M. Viroli. " Integrating Objective & Subjective Coordination in FIPA: A Roadmap to TuCSoN".

[Omicini et al.'04] Omicini, Ricci, Viroli, Castelfranchi, Tummolini "A Conceptual Framework for Self-Organising MAS" WOA 2004 Torino, 1 December 2004.

[Ossowski'99] S. Ossowski. "Co-ordination in Artificial Agent Societies. Social Structure and Its Implications for Autonomous Problem-Solving Agents", no. 1202 in LNAI, Springer Verlag, 1999.

[Parsons et al.'98] S. Parsons, C. Sierra and N. R. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation 8 (3) 261-292, 1998.*

[prodnet] PRODNET, http://www.uninova.pt/~prodnet/

[Putman et Holmer'92] Putnam, L. and M. Holmer, "Framing, Reframing, and Issue Development", in Putnam L. and Roloff, M.E. (Eds.), Communication and Negotiation, Newbury Park, CA: Sage, Vol. 20. pp.128-155, 1992.

[Raiffa'82] H. Raiffa. The Art and Science of Negotiation. *Harvard University Press, 1982.*

[Rap'01] CT/XRCE and SMA/ENSMSE. E-Alliance, Modéles de négociation: rapport intermédiaire. Technical report, E-Alliance consortium, July 2001.

[Rap'02] Rapport Activité, M. Bratu, « Alliances Inter-Organisationnelles ». *Bourse Doctorale, Projet E-Alliance. Mar. 2002*

[Rap'03] Rapport Activité, M. Bratu, « Alliances Inter-Organisationnelles ». *Bourse Doctorale, Projet E-Alliance. Mar. 2003*

[Rap'03b] Rapport Activité intermédiaire, M. Bratu, « Alliances Inter-Organisationnelles ». *Bourse Doctorale, Projet E-Alliance. Sep. 2003.*

[Reed'98] C. Reed. "Dialogue frames in agent communication". In Proc. 3rd International Conference on Multi-Agent Systems (ICMAS 98), pg. 246-253, 1998.

[Robinson et Volkov'98] W. Robinson and V. Volkov. Supporting the negotiation life cycle. *Communications of the ACM, 41(5) :95-102, May 1998.*

[Rosenschein et Zlotkin'94] J. S. Rosenschein and G. Zlotkin. "Rules of Encounter". MIT Press, 1994.

[Rousseau'95] D. Rousseau. "Modélisation et simulation de conversations dans un univers Multi-Agent". PhD thesis, Univ. de Montréal, CA, 1995.

[Sandholm et Lesser'95] T. Sandholm and V. Lesser, "Issues in Automated Negotiation and Electronic Commerce: Existing the Contract Net Framework". In Proc. of ICMAS, AAAI Press, 1995.

[Sandholm et Lesser'97] T. Sandholm and V. Lesser, "Coalitions Among Computational Bounded Agents". In Artificial Intelligence, Special Issue on Principles of Multi-agent System, 1997

[Sandholm'99] T. Sandholm. Distributed rational decision making. *In the textbook Multiagent Systems: A Modern Introduction to Distributed Artificial Intelligence, Weiss G. ed., MIT Press: 201- 258, 1999.*

[Santos et Carmo'96] F. Santos and J. Carmo, "Indirect Action, Influence and Responsibility". In M. Brown and J. Carmo, editors, *Deontic Logic, Agency and Normative Systems*. Springer, Berlin, 1996.

[Santos et al.'97] F. Santos, A. Jones, and J. Carmo. "Action Concepts for Describing Organised Interaction". *In 13th HICSS. IEEE Computer* Society Press, Los Alamitos, 1997.

[Schumacher'01] M. Schumacher. "Objective Coordination in Multi-Agent System Engineering – Design and Implementation". In Lecture Note in Artificial Intelligence, no. 2093, Springer Verlag, 2001.

[Sen et Durfee'91] Sandip Sen and Edmund H. Durfee, "A formal study of distributed meeting scheduling: Preliminary results". In Proc. of the ACM Conference on Organizational Computing Systems, pages 55-68, 1991.

[Sen et Durfee'93] Sandip Sen and Edmund H. Durfee, "Using temporal abstractions and cancellations for efficiency in automated meeting scheduling". In Proc. of the International Conference on Intelligent and Cooperative Information Systems, pages 163-172, 1993.

[Sen et Durfee'94] Sandip Sen and Edmund H. Durfee, "On the design of an adaptive meeting scheduler". In Proc. of the Tenth IEEE Conference on AI Applications, pages 40-46.

[Sian'91] S. Sian. "Adaptation based in cooperative learning in multi-agent systems". In Decentralized Artificial Intelligence II – Proceedings of the 2nd European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'90), pg. 257-270, 1991.

[Singh'97a] M.P. Singh, "Coordinating heterogeneous autonomous agents". TR 97-07, Department of Computer Science, North Carolina State University, Raleigh, July 1997. Available at www.csc.ncsu.edu/ faculty/ mpsingh/ papers/ mas/ coord-tr.ps.

[Singh'97b] M. P. Singh, "Commitments among autonomous agents in information-rich environments". In *Proceedings of the 8th EuropeanWorkshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW)*, pages 141–155, May 1997.

[Singh'98] M. P. Singh. "A customizable coordination service for autonomous agents". In Proc. of MAAMAW'90, pg.257-270, The Netherlands 1998.

[Singh'98] M. P. Singh. "Agent Communication Languages: Rethinking the Principles". *IEEE Computer, pages 40-47, 1998.*

[Smith'80] R.G. Smith, "The Contract Net Protocol: high-level communication and control in a distributed problem solver". IEEE Transactions on computers, C-29(12):1104-1113.

[Smith et Davis'81] R. Smith and R. Davis. "Framework for cooperation in distributed problem solving". IEEE Transactions on Systems, Man and Cybernetics, SMC-11 (1) :61-70, 1981.

[Strobel'00] Strobel, M. "A Framework for Electronic Negotiations Based on Adjusted-Winner Mediation". *Proceedings of the DEXA Workshop on e-Negotiations*, London, UK, 2000, 1020-1028.

[Strobel et Stolze'01] Michael Storbel & Markus Stolze, "A Matchmaking Component for the discovery of agreement and negotiation space in electronic markets", 2001.

[Sycara'91] K. Sycara. "Problem restructuring in negotiation". *In Management Science, 37(10), 1991.*

[Thompson'99] Leigh L. Thompson, "The Mind and Heart of the Negotiator", Upper Saddle River, New Jersey: Prentice Hall;

[Tsvetovat et al.'00] M. Tsvetovat, K. Sycara, Y. Chen and J.Ying. "Customer Coalition in electronic markets". *In AMEC 2000, pp. 121-138, 2000.*

[Tung et Raymund'05] Hung-Wen Tung et Raymund J. Lin. "Automated Contract Negotiation Using a Mediation Service," *cec*, pp. 374-377, Seventh IEEE International Conference on E-Commerce Technology (CEC'05), 2005.

[Vercouter'00] Vercouter L., "A distributed approach to design open multi-agent system". In *Proceedings of ESAW,* 2000.

[Vinoski'97] Steve Vinoski, "CORBA: Integrating Diverse Applications Within Distributed Heterogeneous Environments". *IEEE Communications Magazine*, Vol. 14, No. 2, February, 1997.

[Vitteau et Huget'03] Benjamin Vitteau et Marc-Philippe Huget, "Modularity in Interaction protocols". In Advances in Agent Communication: International Workshop on Agent Communication Languages, ACL 2003, Melbourne, Australia, July 14, 2003, pp. 291 – 309.

[Wacher et Reuter'92] H. Wachter and A. Reuter. The ConTract Model. *In Database Transaction Models for Advanced Applications, A.K. Elmagarmid, pages 219-258, 1992.*

[Walton et Krabbe'95] D. N. Walton and E. C. W. Krabbe. "Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning". SUNY Press, Albany, NY, USA, 1995.

[Wertheim] E. Wertheim. "Negotiations and Resolving Conflicts: An Overview," College of Business Administration, Northeastern University, [available at: http://web.cba.neu.edu/~ewertheim/interper/negot3.htm]

[WMC'95] Workflow Management Coalition. The Workflow Reference Model. *Rapport no. WFMC-TC-1003 Version 1.1, January 1995.*

[Wooldridge et Jennings'95] Wooldridge, M & Jennings, N. R. (1995) Agent Theories, Architectures, and Languages: a Survey. In Wooldridge and Jennings Eds., Intelligent Agents, Berlin:Springer-Verlag.

[Wu et Laws'03] Jenai Wu et David Laws, "Trust and Other-Anxiety in Negotiations: Dynamics Across Boundaries of Self and Culture", in Negotiation Journal 19 (4): 329-367, October 2003.

[Wurman et al.'98] P.R. Wurman, W.E. Walsh, and M.P. Wellman, (1998), « Flexible double auctions for electronic commerce - Theory and implementation », Decision Support Systems 24:17-27;

[Zartman'00] I. William Zartman, "Ripeness: The Hurting Stalemate and Beyond," in *International Conflict Resolution after the Cold War* eds., Paul Stern and Daniel Druckman, Washington: National Academy Press, 2000.

[Zhang et al.'01a] XiaoQin Zhang, Victor Lesser and Rodion Podorozhny "Multi-dimensional, MultiStep Negotiation for Task Allocation in a Cooperative System", National Science Foundation material, July 2001.

[Zhang et al.'01b] XiaoQin Zhang and Victor Lesser "Dealing with Multi-Linked Negotiation with Reasoning", Technical Report of Computer Science Department, UMass., 2001.

[Zhang et Lesser'02] XiaoQin Zhang and Victor Lesser "Multi-linked negotiation in multi-agent systems", In Proc. of AAMAS 2002 July, Bologna, pg. 1207 – 1214.

Annexe A

Program Formulae - AsABlock

Les méthodes détaillées par la suite traduisent la politique R_k attaché pour la description d_k avec $vue(d_k) = (p_1,N, R_k)$. Nous avons identifié ce comportement par un type particulier de description de négociation appelé *ASABlock*.

Nous supposons que le premier atome de la description de négociation $d_{AsABlock}$ contient initialement les particules suivantes: **name**(Id), **start**, **localr**(Rname1,initiated, ∅) – la particule de représentation pour la description *AsABlock*, **firstr**(Rname2,initiated, ∅) – la particule de représentation pour la description *Outsrc*, **test_size**(value) – la particule qui contient la taille de la tâche fixée par le participant p_1.

Comme la description $d_{AsABlock}$ est supposée voir en totalité les différentes négociations bilatérales, elle sera impliquée dans la négociation dès le début, donc du moment de la création des triplets pour les participants invités. Pour chaque événement **clone_create**(Id, New_Id, Msg) un nouvel atome est créé (1.). La particule {**string_create**(Msg, Rname, Type)}fait appel à une particule computationelle locale qui détermine dans la variable Msg les valeurs des paramètres de la particule **create**(Rname, Type). La deuxième méthode (2.) fait apparaître une nouvelle particule de représentation **extr(**Rname, initiated, ∅) qui représente l'image que la nouvelle description a sur cette nouvelle phase de négociation. Comme la particule de contrôle **start** fait que seulement à partir du premier atome nous pouvons introduire des événements **clone_create**, tous les atomes clonés à partir de celui-ci seront des phases de négociation partagées entre le partenaire p_1 et un seul autre partenaire.

1. **name**(Id) @ **start** @ **clone_create**(Id, New_Id, Msg) @ {**string_create**(Msg, Rname, Type)} <>- (**start** @ **name**(Id)) & (**freeze** @ **name**(New_Id) @ **create**(Rname, Type))
2. **freeze** @ **create**(Rname, type) <>- **extr(**Rname, initiated, ∅) @ **enable**

Les deux méthodes suivantes gèrent la dépendance sur la taille de la tâche négociée - *la règle as_a_block.* A partir de tous les nouveaux atomes créés nous pouvons, par la suite, continuer la négociation avec les nouvelles propositions reçues (3.). La proposition fait évoluer (4.) la nouvelle phase de négociation seulement si la contrainte imposée sur la taille de la tâche est respectée - la particule computationelle {**substring**(value,Content, true)} vérifie cette contrainte.

3. **name**(Id)@**enable** @ **clone_propose**(Id, New_ Id, Msg) @{**string_propose**(Msg, Rname, Content)} <>- (**enable** @ **name**(Id)) & (**freeze** @ **name**(New_ Id) @ **propose**(Rname, Content))
4. **freeze** @ **test_size** (value) @ **localr**(Rname1, S1, I1) @ **firstr**(Rname2, S2, I1) @ **extr(**Rname3, S3, I1) @ **propose**(Rname, Content) @ {**substring**(value,Content, true)}@ {**construct**(I1,Content,I)}<>- **enable**@ **localr**(Rname1, undefined, I) @ **firstr**(Rname2, undefined, I) @ **extr(**Rname3, undefined, I)

Les méthodes de (5.) à (9.) traduisent les dépendances sur les statuts – *la règle competition*. Dans toutes les phases de négociation valides (celles avec la particule de contrôle **enable**) les partenaires impliqués peuvent accepter la proposition courante (5.). Comme les deux partenaires doivent être d'accord, dans le nouvel atome crée, les particules de représentations pour un seul partenaire seront dans un statut *success* (6. et 7.). Une nouvelle particule de contrôle est introduite (**waiting**) afin de conserver l'atome de négociation seulement pour des événements de type **clone_accept** ou **clone_reject** (8. et 10.). Un accord est trouvé seulement si toutes les trois particules de représentation sont dans un statut success (9). Dans ce cas tous les autres atomes de négociation sont annoncés d'arrêter la négociation (la particule **stop** est introduite par le mécanisme de broadcasting dans tous les atomes composant la description de négociation $d_{AsABlock}$).

5. **name**(Id) @ **enable** @ **clone_accept**(Id, New_ Id, Msg) @ { **string_accept**(Msg, Rname)}<>- (**enable** @ **name**(Id)) & (**freeze** @ **name**(New_ Id) @ **accept**(Rname))

6. **freeze** @ **localr**(Rname1, S1, I) @ **firstr**(Rname2, S2, I) @ **accept**(Rname2)<>- **localr**(Rname1, success, I) @ **firstr**(Rname2, success, I) @ **waiting**
7. **freeze** @ **extr(**Rname3, S3, I) @ **accept**(Rname3)<>- **extr(**Rname3, success, I) @ **waiting**
8. **name**(Id) @ **waiting** @ **clone_accept**(Id, New_ Id, Msg) @ { **string_accept**(Msg, Rname)}<>- **name**(Id) @ **freeze** @ **accept**(Rname)
9. **localr**(Rname1, success, I) @ **firstr**(Rname2, success, I) @ **extr(**Rname3, success, I) @ ^**stop** <>- **ready**(I)

Dans les méthodes de (10.) à (11.) nous traitons l'événement de type **clone_reject** et dans la méthode (12.) le message de type **reject**.

La méthode (10.) modélise le traitement du **clone_reject** dans un atome de négociation dans lequel un des participants a fait auparavant un accept sur la phase courante de négociation (la particule **waiting** est créée seulement dans une méthode qui traite un message de type **accept** – 7.). La méthode (11.) modélise le traitement du **clone_reject** dans un atome de négociation où la proposition n'est pour le moment ni acceptée ni refusée. Les deux méthodes introduisent la particule message **reject** dans l'atome de négociation courant.

Dans le traitement du message **reject** - la méthode (12.) - nous avons choisi de conserver la phase de négociation associée à l'atome courant, en modifiant tous les statuts en *failure* et d'empêcher toutes les évolutions possibles de l'atome en consommant toutes les particules de contrôle.

La dernière méthode (13.) est une loi qui détruit complètement tous les atomes de négociations visibles par la description $d_{AsABlock}$, sauf l'atome de négociation avec l'accord sur la négociation (la particule **stop** a été créée par cet atome – 9.).

10. **name**(Id) @ **waiting** @ **clone_reject**(Id, New_ Id, Msg) @ { **string_reject**(Msg, Rname)}<>- **freeze** @ **name**(New_ Id) @ **reject**(Rname)

11. **name**(Id) @ **enable** @ **clone_reject**(Id, New_ Id, Msg) @ { **string_reject**(Msg, Rname)}<>- **freeze** @ **name**(New_ Id) @ **reject**(Rname)

12. **freeze** @ **localr**(Rname1, S1, I) @ **firstr**(Rname2, S2, I) @ **extr(**Rname3, S3, I) @ **reject**(Rname)<>- **localr**(Rname1, failure, I) @ **firstr**(Rname2, failure, I) @ **extr(**Rname3, failure, I)

13. **stop** <>- **#t**

Program Formulae - Split

Par la suite, le comportement de la description d_{Split} avec $(d_{Split}) = (p_1,N,R_k)$ sera modélisé .

Nous supposons que le premier atome de la description de négociation d_{Split} contient initialement les particules suivantes: **name**(Id), **start**, **localr**(Rname1,initiated, ∅), **firstr**(Rname2,initiated, ∅) – la particule de représentation pour la description *Outsrc*, **count**(X) – particule utilisée pour la décomposition de la tâche. Dans ce schéma on cherche une décomposition de la tâche en deux, donc X=2.

Pour chaque événement **clone_create**(Id, New_Id, Msg) un nouvel atome est créé (1.). La deuxième méthode (2.) fait apparaître une nouvelle particule de représentation **extr(**Rname, initiated, ∅) qui représente l'image que le triplet d'un nouvel partenaire a sur cette nouvelle phase de négociation. Comme nous avons décidé de chercher une décomposition de la tâche en deux, les atomes dans lesquels nous pouvons commencer à négocier sont ceux avec **count**(0) et donc ayant deux particules de représentation externes pour deux nouveaux partenaires (3.).

1. **count**(X) @ **name**(Id) @ **start** @ **clone_create**(Id, New_Id, Msg) @ {X !=0} @{**string_create**(Msg, Rname, Type)} <>- (**start** @ **name**(Id) @ **count**(X)) & (**start** @ **freeze** @ **name**(New_Id) @ **create**(Rname, Type) @ **count**(X))
2. **count**(X) @ **freeze** @ **create**(Rname, type) <>- **extr(**Rname, initiated, ∅) @ **count**(--X)
3. **count**(0) @ **start** <>- **enable**

Le group suivant des méthodes (4.) à (6.) gère l'échange des propositions et aussi garantit les dépendances entre les propositions faites sur des parties de tâche – *la règle split_competition.* La première méthode est le traitement générique de l'événement **clone_propose**, donc la négociation évolue prenant en compte de propositions seulement dans les atomes qui contiennent la particule de contrôle **enable** (4.).Le traitement de la particule **propose** est différent en fonction de celui qui a fait la proposition. Si la proposition a été faite par le participant initiateur (5.), cette nouvelle proposition faite sur la totalité de la tâche sera découpée par la particule computationelle **construct_split** afin de pouvoir construire des propositions pour chaque participant invité. Si la proposition a été faite par un des participants

invités dans la négociation, cette nouvelle proposition sera manipulée par la particule computationelle **solve_split** afin de pouvoir construire pour l'autre partenaire une proposition sur le reste de la tâche (6.).

4. **name**(Id)@**enable** @ **clone_propose**(Id, New_ Id, Msg) @{**string_propose**(Msg, Rname, Content)} <>- (**enable** @ **name**(Id)) & (**freeze** @ **name**(New_ Id) @ **propose**(Rname, Content))

5. **freeze** @ **localr**(Rname1, S1, I1) @ **firstr**(Rname2, S2, I1) @ **extr(**Rname3, S3, I3)@ **extr(**Rname4, S4, I4)@ **propose**(Rname2, Content) @ {**construct_split**(I1,I3,I4Content,I5,I6,I7)}<>- **enable**@ **localr**(Rname1, undefined, I5) @ **firstr**(Rname2, undefined, I5)@**extr(**Rname3, undefined, I6)@ **extr(**Rname4, undefined, I7)

6. **freeze** @ **localr**(Rname1, undefined, I1) @ **firstr**(Rname2, undefined, I1) @ **extr(**Rname3, undefined, I3)@ **extr(**Rname4, undefined, I4)@ **propose**(Rname2, Content) @ {**solve_split**(I1,I3,I4Content,I5,I6,I7)}<>- **enable**@ **localr**(Rname1, undefined, I5) @ **firstr**(Rname2, undefined, I5)@**extr(**Rname3, undefined, I6)@ **extr(**Rname4, undefined, I7)

La modélisation d'une acceptation est la même que dans l'exemple précédant (7.) à (10.) La seule différence est que maintenant il y a deux modalités de trouver un accord. Dans un premier cas toutes les quatre particules de représentation doivent être dans un statut *success* (11.). C'est la mise en place de *la règle split_success*. Dans le deuxième cas, on trouve l'accord si une seule des représentations externes est dans statut *success,* mais l'accord est sur la totalité de la tâche (12.). C'est la mise en place de *la règle competition*. Dans les deux cas, quand un accord est trouvé les autres atomes sont annoncés avec la particule **stop** par le mécanisme de broadcasting.

7. **name**(Id) @ **enable** @ **clone_accept**(Id, New_ Id, Msg) @ { **string_accept**(Msg, Rname)}<>- (**enable** @ **name**(Id)) & (**freeze** @ **name**(New_ Id) @ **accept**(Rname))
8. **freeze** @ **localr**(Rname1, S1, I) @ **firstr**(Rname2, S2, I) @ **accept**(Rname2)<>- **localr**(Rname1, success, I) @ **firstr**(Rname2, success, I) @ **waiting**
9. **freeze** @ **extr(**Rname3, S3, I) @ **accept**(Rname3)<>- **extr(**Rname3, success, I) @ **waiting**
10. **name**(Id) @ **waiting** @ **clone_accept**(Id, New_ Id, Msg) @ { **string_accept**(Msg, Rname)}<>- **name**(Id) @ **freeze** @ **accept**(Rname)
11. **localr**(Rname1, success, I) @ **firstr**(Rname2, success, I) @ **extr(**Rname3, success, I3) @ **extr(**Rname4, success, I4) @^**stop** <>- **ready**(I)
12. **localr**(Rname1, success, I) @ **firstr**(Rname2, success, I) @ **extr(**Rname3, success, I) @^**stop** <>- **ready**(I)

Dans les méthodes (13.) à (14.) nous traitons l'événement de type **clone_reject** , premièrement dans un atome créé à partir d'une acceptation et deuxièmement dans un atome créé à partir d'une proposition.

13. **name**(Id) @ **waiting** @ **clone_reject**(Id, New_ Id, Msg) @ { **string_reject**(Msg, Rname)}<>- **freeze** @ **name**(New_ Id) @ **reject**(Rname)
14. **name**(Id) @ **enable** @ **clone_reject**(Id, New_ Id, Msg) @ { **string_reject**(Msg, Rname)}<>- **freeze** @ **name**(New_ Id) @ **reject**(Rname)

Le traitement du message de type **reject** est fait en fonction du partenaire qui décide de refuser une proposition. Si c'est un des participants invités qui décide d'arrêter la négociation dans l'atome courant, ceci sera visible dans l'atome géré par la description d_{Split} seulement au niveau de la particule de représentation du participant envoyant le refus (15.). L'atome de négociation est complètement détruit si c'est le participant initiateur qui envoie le refus (16.).

De la même manière, la négociation s'arrête dans les atomes où les deux participants invités sont dans un statut de failure (17.) et dans les atomes qui contiennent la particule **stop** (18.).

15. **freeze** @ **extr(**Rname3, S3, I) @ **reject**(Rname3) <>- **extr(**Rname3, failure, I)

16. **freeze** @ **localr**(Rname1, S1, I) @ **firstr**(Rname2, S2, I) @ **reject**(Rname2) <>- **#t**

17. **extr(**Rname3, failure, I3) @ **extr(**Rname4, failure, I4) <>- **#t**

18. **stop** <>- **#t**

Program Formulae - SwapOut

Nous supposons que le premier atome de la description de négociation $d_{SwapOut}$ contient initialement les particules suivantes: **name**(Id), **start**, **localr**(Rname1,initiated, ∅), **firstr**(Rname2,initiated, ∅).

A partir du premier atome nous allons créer des nouveaux atomes (1.) où on introduit une nouvelle particule de représentation (2.). Les particules **clone_create** sont introduites dans le système à l'issue d'un processus de sélection, qui fait que les nouvelles particules de représentations identifient des triplets de type *SwapIn* appartenant à d'autres négociations. Ces négociations devraient satisfaire les contraintes spécifiées dans **TriggerSet.**

Nous utilisons une nouvelle particule de contrôle **jobtoswap**(Rname1) introduite dans système à l'issues d'un processus de décision et qui identifie la négociation avec laquelle on cherche le swap (3.).

1. **name**(Id) @ **start** @ **clone_create**(Id, New_Id, Msg) @ {**string_create**(Msg, Rname, Type)} <>- (**start** @ **name**(Id)) & (**freeze** @ **name**(New_Id) @ **create**(Rname, Type))

2. **freeze** @ **create**(Rname3, type) <>- **extr(**Rname3, initiated, ∅) @ **enable**

3. **jobtoswap**(Rname3) @ **extr(**Rname3, initiated, ∅) <>- **extr(**Rname3, undefined, ∅) @ **readytoswap**(Rname3)

C'est seulement dans l'atome de la négociation contenant la particule de contrôle **readytoswap** qu'un possible échange (swap) peut être identifié.

Comme dans ce schéma de coordination l'évolution des négociations en termes des propositions et des contra-propositions n'est pas prise en compte, cette description construit l'image de la négociation prenant en compte seulement les changements de statut. Ainsi, on ne traite pas les événements de type **clone_propose,** mais seulement les événements de type **clone_accept** et **clone_reject**. Donc, l'image de la description de négociation $d_{SplitOut}$ est une image très simple sur la négociation N1. En dehors de l'atome racine, nous avons seulement des atomes de négociation qui gardent le statut courant de la négociation.

Sachant qu'à partir d'un statut *undefined* l'évolution d'une négociation est soit dans un statut *success* soit dans un statut *failure* et que les évolutions sur les Issues ne sont pas prises

en compte, dans ce cas les événements de type **clone_accept** (4.) et **clone_reject** (8.) ne nécessitent pas la duplication des atomes qui les reçoivent.

Le mécanisme d'échange se termine avec un accord **swap** seulement si les deux négociations aboutissent à un accord (7.).

4. **readytoswap**(Rname3) @ **name**(Id) @ **enable** @ **clone_accept**(Id, New_ Id, Msg) @ { **string_accept**(Msg, Rname)}<>- **readytoswap**(Rname3) @ **enable** @ **name**(Id) @ **accept**(Rname)

5. **firstr**(Rname2, S2, ∅) @ **accept**(Rname2) <>- **firstr**(Rname2, success, I)

6. **extr(**Rname3, S2, ∅) @ **accept**(Rname3) <>- **extr(**Rname3, success, ∅)

8. **readytoswap**(Rname3) @ **firstr**(Rname2, success, ∅) @ **extr(**Rname3, success, ∅) @ ^**stop** <>- **stop_swap(**Rname2, Rname3**)**

La destruction des atomes de la description SwapOut est faite si n'importe quelle négociation est terminée avec un *failure.* L'échange est impossible et tous les atomes du triplet courant sont détruits (9.) et (10.).

9. **readytoswap**(Rname3) @ **name**(Id) @ **enable** @ **clone_reject**(Id, New_ Id, Msg) @ ^**stop**<>- **#t**

10. **stop**<>- **#t**

Program Formulae - Transp

Nous supposons que le premier atome de la description de négociation d_{Transp} contient initialement les particules suivantes : **name**(Id), **start**, **localr**(Rname1,undefined, I1), **firstr**(Rname2,initiated, I1) – comme la négociation est déjà commencée, les deux particules de représentation ont des valeurs pour le paramètre Issues -*I1* identifiant la phase de négociation à partir de la quelle une synchronisation du transport est cherchée, et **valueTransp**(value) – la particule qui contient la taille maximale d'un possible transport.

A partir du premier atome nous allons créer des nouveaux atomes (1.) dans lesquels on introduit une nouvelle particule de représentation (2.). Les particules **clone_create** sont introduites dans le système à l'issue d'un processus de sélection, qui fait que les nouvelles particules de représentations identifient des triplets de type *Transp* appartenant à d'autres négociations. Ces négociations doivent satisfaire les contraintes spécifiées dans **TriggerSet.**

La nouvelle particule de représentation doit avoir des valeurs en concordance avec les règles sur les dépendances des attributs – *règles transp_coordination*. Comme pour le moment aucune proposition faite dans ces négociations n'a pas été prise en compte, c'est la description Transp qui se charge de coordonner la construction de la première proposition. La particule computationelle **construct_transp**(I1, ∅,value,I) construit la plus large valeur possible *I*, en partant de la proposition faite dans la négociation courante *I1* et de la proposition faite dans la négociation liée (le premier aperçu de cette valeur est ∅) et aussi de la constante décrivant les valeurs maximales pour un transport *value)*.

Afin d'identifier la négociation N_2 avec laquelle on cherche une coordination sur le transport, nous utilisons une nouvelle particule de contrôle **jobtoswap**(Rname3) identifiant le nom de la description participant dans cette négociation (3.). Cette particule de contrôle est introduite dans le système à l'issue d'un processus de décision faisant le choix de la négociation N2.

1. **name**(Id) @ **start** @ **clone_create**(Id, New_Id, Msg) @ {**string_create**(Msg, Rname, Type)} <>- (**start** @ **name**(Id)) & (**freeze** @ **name**(New_Id) @ **create**(Rname, Type))

2. **freeze** @ **firstr**(Rname2, S2, I1) @ **create**(Rname3, type) @ {**construct_transp**(I1, ∅,value,I)} <>- **extr(**Rname3, undefined, I) @ **enable**@ **firstr**(Rname2, undefined, I1)

3. **jobtotransp**(Rname3) @ **extr(**Rname3, undefined, I) <>- **extr(**Rname3, undefined, I) @ **readytotransp**(Rname3)

Le group suivant des méthodes (4.) à (6.) gère l'échange des propositions et garantit les dépendances entre les propositions faites dans les deux négociations sur leurs tâches. Afin de gérer seulement les événements qui font évoluer les négociations impliquées dans la coordination sur le transport, dans le traitement générique de l'événement **clone_propose** nous avons rajouté la particule de contrôle **readytotransp** (4.). Le traitement de la particule **propose** est différent en fonction de celui qui a fait la proposition (5.) et (6.). La différence est essentiellement dans les paramètres des particules computationelles qui construisent les propositions. Dans le (5.) la particule **construct**(I1,Content,I3) construit la proposition *I3* qui est faite dans la négociation courante. La particule **construct_transp**(I3,I2,value,I4) construit la nouvelle proposition *I4* pour la négociation liée. Dans (6.) le mécanisme est inversé, on construit premièrement la proposition faite dans la négociation liée et ensuite la proposition pour la négociation courante.

4. **name**(Id)@**enable** @ **readytotransp**(Rname3) @**clone_propose**(Id, New_ Id, Msg) @{**string_propose**(Msg, Rname, Content)} <>- (**enable** @ **name**(Id) @ **readytotransp**(Rname3)) & (**freeze** @ **name**(New_ Id) @ **propose**(Rname, Content) @ **readytotransp**(Rname3))

5. **freeze** @ **localr**(Rname1, undefined, I1) @ **firstr**(Rname2, undefined, I1) @ **extr(**Rname3, undefined, I2) @ **propose**(Rname2, Content) @ {**construct**(I1,Content,I3)} @ {**construct_transp**(I3,I2,value,I4)} <>- **enable**@ **localr**(Rname1, undefined, I3) @ **firstr**(Rname2, undefined, I3) @ **extr(**Rname3, S3, I4)

6. **freeze**@ **localr**(Rname1, undefined, I1) @ **firstr**(Rname2, undefined, I1) @ **extr(**Rname3, undefined, I2) @ **propose**(Rname3, Content) @ {**construct**(I2,Content,I3)} @ {**construct_transp**(I3,I1,value,I4)} <>- **enable**@ **localr**(Rname1, undefined, I4) @ **firstr**(Rname2, undefined, I4) @ **extr(**Rname3, undefined, I3)

La modélisation d'une acceptation est la même que dans l'exemple précédant (7) à (9.) L'acceptation faite seulement dans une des négociations n'entraîne que le changement dans l'autre négociation. La coordination des deux négociations est finalisée avec un succès seulement si les deux négociations sont dans un statut de *success* (10.). De la même manière les autres atomes sont annoncés avec la particule **stop** par le mécanisme de broadcasting.

7. **name**(Id) @ **enable** @ **readytotransp**(Rname3)@ **clone_accept**(Id, New_ Id, Msg) @ { **string_accept**(Msg, Rname)}<>- **enable** @ **name**(Id) @ **accept**(Rname) @ **readytotransp**(Rname3)

8. **localr**(Rname1, S2, I) @ **firstr**(Rname2, S2, I) @ **accept**(Rname2) <>- **localr**(Rname1, success, I) @ **firstr**(Rname2, success, I)

9. **extr(**Rname3, S3, I) @ **accept**(Rname3) <>- **extr(**Rname2, success, I)

10. **readytotransp**(Rname3) @ **firstr**(Rname2, success, I2) @ **extr(**Rname3, success, I3) @ ^**stop** <>- **stop_transp(**Rname2, Rname3**)**

Le traitement des événements de type **clone_reject** (11.) montre la traduction des règles *fail_complementary_hard* sur les statuts des deux négociations. Donc, si n'importe quelle des négociations introduit dans système un événement **clone_réject**, cela est équivalent à l'impossibilité de continuer la coordination dans l'atome courant (11.).

11. **name**(Id) @ **enable** @ **readytotransp**(Rname3) @ **clone_reject**(Id, New_ Id, Msg) <>- **#t**

La dernière méthode (12.) détruit tous les atomes contenant la particule **stop**, introduite au moment où un accord entre les deux négociations est trouvé.

12. **stop**<>- **#t**

Annexe B

Nous considérons que :

Objet de Négociation (Negotiation Object) représente la description de la tâche à négocie avec les ensembles des possibles valeurs pour chaque attribut négociable sur les propriétés du job. Cette première description de la tâche avec aussi les préférences et les dépendances entre les attributs est décrite par le Manager ;

Contexte de Négociation (Negotiation Context) représente toutes les données qu'on peut extraire à partir d'un nœud blanc (attributs instancies, attributs demandes, position du nœud dans la structure du graphe) ;

Contexte de Négociation possible représente le contexte résultant par la réunion des deux ou plusieurs nœuds blancs, ce contexte peut être considéré comme attaché à un nœud virtuel (un nœud qui n'est pas présent pour le moment dans la structure du graphe);

Issue Set représente l'ensemble des attributs proposes pour être négocies dans un cycle de la négociation ;

Partial_NO_Set représente l'ensemble des contextes de négociations existantes dans lesquelles touts les attributs d'un Issue Set sont instancies ;

Proposals_Set représente l'ensemble des contextes de négociations existantes dans lesquelles touts les attributs d'un Issue Set ne sont pas instancies ;

Possible_NO_Set représente l'ensemble des contextes de négociations possibles dans lesquelles touts les attributs d'un Issue Set sont instancies ;

Possible_Proposals_Set représente l'ensemble des contextes de négociations possibles dans lesquelles touts les attributs d'un Issue Set ne sont pas instancies ;

- *OpenWN(n, pni):*

Graph.open(n,nb)

New negotiation context N1

Where N1.nodes =n

If n is a white node

For every nw parent of nb

Collect negotiation context(nw) in N1

If inconsistencies in N1

Quit(n)

Else

If all attributs are instantiated

Insert the N1 in *Partial_NO_Set*

Else

Insert the N1 in *Proposals_Set*

Generate FindObject(N1)

FindObject(NC)

For every node n in CN.nodes :

Create an empty list L(n) of nodes;

For every white node nw in Graph :

If first common_ancestor(n, nw) is White Node :

Insert nw in L(n);

Create a list L of nodes where $L = \cap_{n \in N1.nodes} L(n)$;

For every nw in L :

New negotiation context CNew ;

CNew.nodes = CN.nodes + nw;

Collect negotiation context (nw) and CN in Cnew;

Check content consistency;

If no inconsistencies in Cnew:

If all attributs are instantiated:

Insert the CNew in *Possible_NO_Set;*

Else:

Insert the CNew in *Possible_Proposals_Set;*

Generate FindObject(CNew).

- *Assert(n,p,i,t):*

Graph.assert(n,p,i,t);

If i in n.NC :

Check consistency;

If inconsistency with the new value t of the attribut i :

Quit(n)

For every NC in *Partial_NO_Set, Proposals_Set, Possible_NO_Set, Possible_Proposals_Set:*

If n∈NC.nodes :

Delete(NC);

Exit();

If all attributs are instantiated :

Insert NC in *Partial_NO_Set;*

For every NC in *Proposals_Set, Possible_NO_Set, Possible_Proposals_Set* :

If n∈NC.nodes :

Delete(Nd);

Else :

For every NC in *Proposals_Set, Possible_NO_Set, Possible_Proposals_Set* :

If n∈NC.nodes:

Update NC with asserted *t*;

Check consistency;

If inconsistency in NC:

Delete(NC);

Exit();

If NC in *Possible_Proposals_Set* and all attributs are instantiated:

Move NC in *Possible_NO_Set*.

- *Request(n,p,i):*

If attribut i not in Issues_Set

Insert i in Issues_Set;

If *Negotiation Object* doesn't contains constraints or possible values for attribut i

Ask higher authority (E.g.: manager, planning);