Contribution au développement de l'interopérabilité en entreprise : vers une approche anticipative de détection de problèmes d'interopérabilité dans des processus collaboratifs

Sihem Mallek Daclin

ACADÉMIE DE MONTPELLIER

UNIVERSITÉ MONTPELLIER II

SCIENCES ET TECHNIQUES DU LANGUEDOC

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ MONTPELLIER II

Discipline : Génie Informatique, Automatique et Traitement du signal

ÉCOLE DOCTORALE : Information, Structure et Systèmes

présentée et soutenue publiquement

par

Sihem MALLEK

Le 14 octobre 2011

Titre :

Contribution au développement de l'interopérabilité en entreprise : vers une approche anticipative de détection de problèmes d'interopérabilité dans des processus collaboratifs

---

JURY

Mr. Jean-Pierre Bourey	Professeur, Centrale Lille	Rapporteur
Mr. David Chen	Professeur, Université Bordeaux I	Rapporteur
Mr. Michaël Petit	Professeur, Facultés Universitaires Notre Dame de la Paix	Examinateur
Mr. Christian Braesch	Maître de Conférences, Université de Savoie	Examinateur
Mr. Nicolas Daclin	Ingénieur de Recherche, École des Mines d'Alès	Co-encadrant
Mr. Vincent Chapurlat	Professeur, École des Mines d'Alès	Directeur de thèse
Mr. Didier Crestani	Professeur, Université Montpellier II	Président

- 2011-

Remerciements

Je souhaite adresser, ici, tous mes remerciements aux personnes qui m'ont apporté leur aide, permis et contribué à l'aboutissement de cette thèse.

Je tiens à exprimer, en premier lieu, mes vifs remerciements à mon directeur de thèse, Mr. Vincent Chapurlat, Professeur à l'Ecole des Mines d'Alèse, dont la direction avisée a contribuée largement à l'aboutissement de ce travail de thèse. Les connaissances et les conseils rigoureux qu'il m'a apportés tout en restant à l'écoute ont contribués à me former en tant que chercheur. Je le remercie pour la confiance qu'il m'a accordée pour l'élaboration de cette thèse. Travailler sous sa direction fut, pour moi, un plaisir.

Je tiens à exprimer mes sincères remerciements, à mon co-encadrant de thèse, Mr Nicolas Daclin, Ingénieur de Recherche à l'Ecole des Mines d'Alès, pour l'aide et le temps qu'il m'a accordé pour le développement de ce travail de thèse. Les longues heures de travail et de discussion ont renforcé mon goût pour la recherche. Je le remercie de m'avoir guidée tout au long de ces trois années de thèse et de m'avoir initié à la recherche au niveau de l'interopérabilité qui fut, pour moi, un nouveau domaine et challenge à relever. Le travail avec lui fut agréable et riche en connaissances.

Je remercie Mr. Jean-Pierre Bourey, Professeur à l'Ecole Centrale de Lille et Mr. David Chen, Professeur à l'Université de Bordeaux 1, pour l'intérêt qu'ils ont manifesté pour ce travail de recherche et d'avoir accepté d'être rapporteurs de ce mémoire de thèse. Les remarques et questions pertinentes de leurs rapports me permettent d'envisager plusieurs perspectives intéressantes pour ce travail de recherche.

Je remercie Mr. Didier Crestani, Professeur de l'Université de Montpellier 2, d'avoir accepté d'examiner mes travaux de thèse et de m'avoir fait l'honneur de présider mon jury de thèse. Je remercie également Mr. Michaël Petit, Professeur aux Facultés Universitaires Notre Dame de la Paix, et Mr. Christian Braesch, Maître de Conférences à l'Université de Savoie pour le temps et l'attention qu'ils ont consacré à l'examen de ce travail de thèse.

Je tiens à remercier les membres du laboratoire LGI2P qui m'ont apporté leur aide et leurs encouragements quand j'en avais besoin. Plus particulièrement, je tiens à présenter mes

remerciements aux thésards avec qui j'ai passé d'agréables moments pendant ces trois années de thèse.

Enfin, je tiens à présenter mes remerciements les plus sincères à ma famille. A mes parents, pour leurs soutiens et encouragements de tous les instants et sans qui cette aventure n'aurait jamais vu le jour. A mon frère Amir, qui m'a soutenu de manière infaillible dans les moments les plus délicats. Sois assuré, en retour, du soutien sans conditions de ta grande sœur dans les projets que tu entreprendras. A mes grands-parents, mes oncles, tantes, cousins et cousines dont les encouragements ont traversé la mer Méditerranée, m'ont réchauffé le cœur et apporté un peu de cette douceur algérienne qui me manque tant parfois. A mes amis, qu'ils trouvent dans ces quelques mots ma gratitude pour m'avoir aidé d'une façon ou d'une autre à traverser ces trois années de travail.

Nîmes, le 1er novembre 2011

Sihem

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

L'ouverture des marchés, l'âpreté de la concurrence et la recherche constante d'innovation sont autant de phénomènes émergents et liés à la mondialisation auxquels les entreprises doivent faire face. En conséquence, elles cherchent à améliorer, de manière continue, leurs capacités de réaction et d'adaptation pour rester à l'écoute de leurs clients. Une des solutions est de développer des collaborations afin de devenir aptes à répondre à des opportunités commerciales éventuellement complexes auxquelles une entreprise isolée ne peut faire face seule. Ce mode de fonctionnement partenarial lui permet de se consacrer à son cœur de métier, d'optimiser l'usage et l'utilisation de ses ressources et souvent d'éviter la dispersion potentielle de ses compétences. Elle accepte ainsi de fournir les biens et les services nécessaires pour couvrir une partie de la demande et ce, dans un climat de confiance et de partage avec ses partenaires devant se construire et lui-même s'améliorer quelquefois pour des durées plus ou moins longues.

Dans ce contexte, plusieurs caractéristiques du partenariat doivent être prises en compte. On évoque aujourd'hui le concept d'interopérabilité comme un des enjeux majeurs devant être pris en compte et maitrisé lors de toute tentative de collaboration inter entreprises, voire au sein même d'une entreprise. En effet, un défaut d'interopérabilité peut induire l'apparition de problèmes, de dysfonctionnements, de ralentissements ou plus globalement de pertes de performance pouvant induire une baisse de confiance entre les partenaires. Ainsi, la caractérisation et la détection de ces problèmes avant toute collaboration effective, puis l'analyse et la recherche de solutions adaptées à chaque partenaire concerné, revêt maintenant une importance stratégique pour chaque entreprise prise séparément et lorsqu'elle est impliquée dans un processus collaboratif.

Ces travaux de recherche tentent de répondre au problème de caractérisation et de détection des problèmes d'interopérabilité. Pour cela, l'interopérabilité est vue comme une exigence globale que des entreprises en situation de partenariat doivent satisfaire. La première contribution de ces travaux est donc de définir et de formaliser cette exigence. Ensuite, la détection se doit d'être anticipative et d'être effective avant même que la collaboration ne débute. Il faut donc d'une part travailler à partir d'une représentation de cette collaboration c'est-à-dire de modèle des processus collaboratifs qui doivent alors être mis en œuvre. Il faut ensuite devenir capable d'analyser si les exigences d'interopérabilité sont effectivement satisfaites par les partenaires impliqués dans ce ou ces processus collaboratifs.

La deuxième contribution de ces travaux est donc de proposer et de développer une approche qui permet de détecter des problèmes d'interopérabilité en se basant sur la formalisation puis la vérification d'exigences d'interopérabilité sur un modèle de processus collaboratif.

Le premier chapitre positionne le contexte et présente la problématique de cette thèse. Le travail de recherche qui a été mené à bien se positionne dans le contexte particulier des processus collaboratifs qu'ils soient privés (intra entreprise) ou public (inter entreprises). En conséquence, nous nous attachons dans ce chapitre à définir, d'une part, le concept d'interopérabilité et les notions qui lui sont associées et, d'autre part, à la définition de processus collaboratif (privé et public). Par la suite, nous montrons que la nécessité de détecter et d'anticiper des problèmes d'interopérabilité de façon locale, puis globale, est une problématique qu'il ne faut pas négliger.

Le deuxième chapitre présente un état de l'art sur les travaux portant sur le développement de l'interopérabilité. Plus précisément, nous nous intéressons aux cadres d'interopérabilité qui permettent de structurer ce concept, aux modèles qui permettent de l'évaluer et de le mesurer, et enfin à la nécessité d'envisager des méthodes plus formelles pour l'évaluer. De ce fait, la deuxième partie de ce chapitre se focalise sur la présentation des certaines techniques de modélisation et de vérification d'exigences. Nous nous intéressons également à la modélisation d'entreprise et, plus particulièrement, à la modélisation de processus collaboratifs.

Le troisième chapitre présente l'approche proposée pour l'anticipation des problèmes d'interopérabilité par détection préalable des causes de ces problèmes au moyen de modèles. Nous montrons qu'un problème d'interopérabilité peut refléter un besoin d'interopérabilité non pris en compte pendant le partenariat. Ce besoin d'interopérabilité est alors traduit sous forme d'exigences devant être satisfaites par chaque partenaire. Nous définissons donc la notion d'exigence d'interopérabilité ainsi que des classes d'exigences d'interopérabilité. Par la suite, nous proposons un modèle pour structurer ces exigences et permettre la détection de problèmes à un niveau local puis à un niveau global avec l'utilisation de mécanismes de propagation.

Le quatrième chapitre est consacré à la modélisation du processus collaboratif et à la vérification des exigences d'interopérabilité. Pour la modélisation du processus collaboratif,

nous montrons que le langage utilisé doit être enrichi pour prendre en considération les concepts liées à l'interopérabilité. Nous présentons donc les enrichissements apportés au langage choisi et utilisé dans nos travaux. Nous nous attachons, par la suite, à présenter les étapes nécessaires pour accomplir la vérification. La première étape concerne la transformation de modèles pour passer du modèle de processus collaboratif relativement informel à un modèle formel et adéquat sur lequel des techniques de vérification formelles peuvent alors être appliquées. La seconde étape est relative à la formalisation des exigences d'interopérabilité en propriétés pour permettre leur vérification.

Le cinquième chapitre présente enfin deux cas d'application de notre approche. Nous présentons deux exemples de partenariat où la mise en œuvre de l'interopérabilité est un enjeu important pour le bon déroulement du partenariat.

Chapitre 1

L’interopérabilité dans les processus collaboratifs : contexte et problématique

Table des matières du chapitre 1

1. Introduction

Les travaux présentés dans ce manuscrit s'inscrivent dans le cadre de l'amélioration de la collaboration et considère celle-ci aussi bien au sein d'une entreprise (intra-entreprise) qu'entre plusieurs entreprises (inter entreprises). Plus précisément, nous concentrons notre recherche sur une meilleure prise en compte et une meilleure exploitation du concept d'interopérabilité dans les processus qui caractérisent la relation de collaboration : **les processus collaboratifs**.

Pour cela, nous définissons et caractérisons dans la suite le concept d'**interopérabilité**. Le concept de **processus collaboratif** est ensuite présenté afin de mettre en évidence l'enjeu et la nécessité d'une meilleure prise en compte de l'interopérabilité au sein de ce type de processus. Enfin, nous exposons les difficultés à percevoir un déficit d'interopérabilité, puis à évaluer son impact sur le fonctionnement d'un processus collaboratif. Cela nous permet de préciser la problématique de ces travaux de recherche.

2. L'interopérabilité dans le contexte du processus collaboratif

2.1. Interopérabilité : concepts et définitions

La mondialisation[1] s'est concrétisée par l'ouverture des marchés à la fin du 20ème siècle. Cela a créé un environnement concurrentiel accru pour les entreprises (Dollfus, 2001). L'effet de la mondialisation sur les entreprises peut se traduire par un besoin de compétitivité[2]. Cette compétitivité engendre un besoin d'amélioration de performances qui peut trouver une réponse opportuniste dans le partenariat. En effet, le partenariat apparaît comme une solution porteuse de nombreux avantages pour les entreprises (*e.g.* meilleures performances, réalisation de nouveaux projets…) et peut être exploité afin d'assurer la survie de l'entreprise dans le marché mondial (Lecerf, 2006).

Cependant, la réalisation d'un partenariat nécessite la prise en compte de relations telles que la *communication*, la *coordination*, la *coopération* ou encore la *collaboration*, au

[1] On retrouve également le terme globalisation dans la littérature.
[2] La compétitivité est la capacité qu'a l'entreprise pour faire face à la concurrence tant sur les marchés externes que sur les marchés internes (Larousse, 2010).

travers des différentes interactions établies entre les partenaires (Lauras, 2004), (Forest, 2003), (Seguy, 2008), (Winer *et al.*, 1994).

- La communication se limite à la transmission de données et de connaissances entre les partenaires.
- La coordination s'attache à l'organisation des activités déployées par les différents acteurs dans un partenariat.
- La coopération représente un travail conjoint où un partage des activités est défini pour accomplir une mission.
- La collaboration représente un travail collectif où les activités sont réalisées par tous pour accomplir une mission.

Au travers des définitions présentes dans la littérature et relatives à la collaboration et la coopération, il est parfois difficile de différencier ces deux notions. Nous retiendrons que la coopération est réalisée par la division du travail entre les participants comme une activité où chaque personne est responsable d'une partie de la résolution du problème. La collaboration consiste à l'engagement mutuel des participants dans un effort coordonné pour résoudre le problème ensemble (Roschelle *et al.*, 1995). La coopération se caractérise par des relations informelles qui existent sans mission commune définie alors que la collaboration connote une relation plus durable et généralisée pour un engagement total en faveur d'une mission commune (Mattessich *et al.*, 1992). De plus, (Touzi, 2007) montre que la collaboration ne peut se détacher des relations de communication, de coordination et de coopération mais propose des niveaux de collaboration. Nous nous attachons donc dans ces travaux de recherche à améliorer les conditions permettant d'aboutir à une **collaboration** fructueuse lors d'un partenariat. Parmi ces conditions, nous nous intéressons à l'interopérabilité nécessaire et requise au cours de cette collaboration. Dans la suite, nous définissons ce concept d'interopérabilité et les notions qui lui sont associées. L'objectif, ici, n'est pas d'établir une liste exhaustive des définitions liées à l'interopérabilité mais d'en présenter les principales.

Une des premières définitions qui clarifie le concept d'interopérabilité est apparue dans les années 90 (IEEE, 1990) comme : « *l'habilité pour deux (ou plusieurs) systèmes à échanger des informations et à utiliser les informations qu'ils ont échangées* ». A l'origine, cette première définition concerne exclusivement les domaines liés à l'informatique. Au fil des années, cette définition s'est étendue à d'autres domaines, notamment à celui des systèmes industriels. Depuis, le développement de l'interopérabilité fait l'objet de nombreuses

initiatives (Européenne, Nord Américaine…). Plus particulièrement, plusieurs projets européens ont été initiés afin de répondre à la problématique de l'interopérabilité au sein de l'entreprise (IDEAS, 2003) (ATHENA, 2003) (INTEROP, 2007). A titre d'exemple, nous pouvons citer le projet IDEAS (*Interoperability Development for Enterprise Application and Software - Roadmaps, IST-2001-37368*), qui définit l'interopérabilité comme « *la capacité d'interaction entre les applications d'entreprises. L'interopérabilité est considérée comme acquise si cette interaction peut, au moins, se réaliser à trois niveaux : donnée, application et processus*».

L'interopérabilité a également fait l'objet de travaux de normalisation, aussi bien dans les domaines techniques (ISO17933, 2000) que dans les domaines liés aux systèmes industriels (ISO14258, 1999), (ISO16100-1, 2009). En ce qui concerne l'interopérabilité d'entreprises, la norme ISO11354-1 (ISO11354-1, 2009) définit celle-ci comme « *la capacité des entreprises et des entités au sein de ces entreprises à communiquer et à interagir efficacement* ».

Sans prétendre à une étude exhaustive, les différentes définitions et travaux susmentionnés illustrent bien les enjeux importants que revêt la mise en œuvre de l'interopérabilité dans les différents domaines où elle est considérée et développée. Dans le cadre plus restreint de nos travaux, nous retenons la définition proposée dans (Pingaud, 2009) - spécifiquement adaptée aux entreprises (*e.g.* un réseau d'entreprises, tout ou partie d'une entreprise…) - qui définit l'interopérabilité comme : « *une capacité de systèmes, nativement étrangers les uns par rapport aux autres, à* ***interagir*** *afin d'établir des comportements collectifs* ***harmonieux*** *et finalisés,* ***sans avoir à modifier*** *en profondeur leur structure ou leur* ***comportement individuel*** ». Cette vision de l'interopérabilité est cohérente avec les définitions proposées dans le cadre d'autres travaux (Vernadat, 1996) (Wegner, 1996). Le terme « ***interagir*** » marque l'action d'agir ensemble de manière réciproque. Dans le contexte de l'interopérabilité, il s'agit dans un premier temps, de permettre à des partenaires de réaliser ces interactions. Il faut alors s'assurer que les partenaires soient **compatibles** *i.e*. qu'ils peuvent réaliser une (ou des) interactions sans effort d'interfaçage particulier. Ensuite, le fait d'établir un comportement « ***harmonieux*** » évoque la notion **d'interopération.** Elle sous-entend la nécessité de partager, d'échanger et d'exploiter des flux de données, d'informations, d'énergie, de ressources humaines, de ressources matérielles ou encore de produits. Le fait de « ***ne pas avoir à modifier*** *en profondeur les systèmes en interaction* » met en évidence la notion de **réversibilité** afin d'assurer aux systèmes le retour à un état identifié et stable

souvent considéré comme l'état original des systèmes avant l'interaction. Enfin, cette même expression évoque également la notion d'**autonomie** c'est-à-dire la faculté pour un système donné de pouvoir continuer à agir et décider afin d'assurer ses propres missions indépendamment des autres systèmes.

Au-delà de ces définitions de l'interopérabilité, le développement de celle-ci dans une entreprise est confronté à trois catégories de problèmes identifiés et définis comme suit par (INTEROP, 2007) :

- **Les problèmes dits conceptuels.** Ils concernent les aspects syntaxiques et sémantiques. Ils sont liés principalement à la modélisation des informations à un niveau d'abstraction relativement haut, comme par exemple un modèle d'entreprise, ainsi qu'à la façon de structurer les informations en vue d'un échange.
- **Les problèmes dits technologiques.** Ils concernent l'utilisation de matériels informatiques pour communiquer et échanger des informations. Ces problèmes sont relatifs, par exemple, à des incompatibilités au niveau des plates-formes, des infrastructures, du système d'exploitation, *etc*. D'un point de vue purement technique, ces problèmes concernent les normes pour présenter, stocker, échanger, traiter et communiquer des données et des informations grâce à l'utilisation de systèmes logiciels.
- **Les problèmes dits organisationnels.** Ils sont liés aux structures organisationnelles et aux techniques de management utilisées dans des entreprises différentes. Plus précisément, ces problèmes concernent les aspects de responsabilité, d'autorisation, de confiance, légaux, de propriété intellectuelle, ...

Ces différents problèmes peuvent être étendus à des flux de matière, d'énergie, de produits ou encore de ressources comme proposés dans plusieurs travaux de recherche. Notamment, les travaux présentés dans (Baina, 2006) s'intéressent à l'interopérabilité des produits via une approche dirigée par les modèles.

Ainsi, l'interopérabilité est un sujet abordé dans différents domaines et, notamment, l'entreprise, les organisations et les systèmes d'informations. Dans la suite, nous nous

intéressons à l'interopérabilité dans les processus collaboratifs intra et inter entreprises tels que définis dans la suite.

2.2. Processus collaboratifs : concepts et définitions

Le passage d'une vision cloisonnée à une vision transversale de l'organisation a été rendue possible par la mise en œuvre de ce que l'on appelle communément l'approche processus. D'abord développée au sein même de l'entreprise (processus intra entreprise), cette approche s'est étendue pour décrire et gérer des relations avec des partenaires externes (processus inter entreprises). Dans ce cas là, nous parlons plus précisément de **processus collaboratifs**.

A la base, un **processus** est vu comme « *un enchaînement ordonné de faits ou de phénomènes, répondant à un certain schéma et aboutissant à quelque chose* » selon le (Larousse, 2010). C'est selon (ISO9000, 2000) un « *ensemble d'activités corrélées ou interactives qui transforme des éléments d'entrée en éléments de sortie* ». Cette définition, *a priori* simple, ne donne aucune information sur la notion d'objectif du processus. Nous pouvons donc citer (Vernadat, 1999) qui précise qu'un processus est « *un ensemble partiellement ordonné d'étapes exécutées en vue de réaliser au moins un objectif* ».

Les **processus collaboratifs** représentent, de nos jours, une nouvelle vision de l'organisation dans le contexte partenarial. Nous adoptons pour cela la définition proposée par (Touzi, 2007) : « *un ensemble partiellement ordonné d'activités spécifiquement organisées chez les partenaires de la collaboration et leur exécution est hébergée chez ces derniers* ». Il existe deux types de processus collaboratif :

- **Les processus collaboratifs privés** représentent des processus internes spécifiquement réalisés au sein de l'organisation.
- **Les processus collaboratifs publics** représentent des processus menés en collaboration avec d'autres en dehors des frontières de l'organisation.

2.2.1. Processus collaboratifs privés : vision intra entreprise

Cette catégorie de processus est par définition un processus interne à l'entreprise (*cf.* Figure 1), et peut être assimilée à la définition du processus métier donnée par (Esper, 2010) : « *un ensemble d'activités ordonnées selon un ensemble de règles procédurales pour réaliser*

un objectif précis au sein d'une organisation et réalisé par un groupe de personne ». Le processus collaboratif privé offre donc une vision transversale orientée métier de l'entreprise.

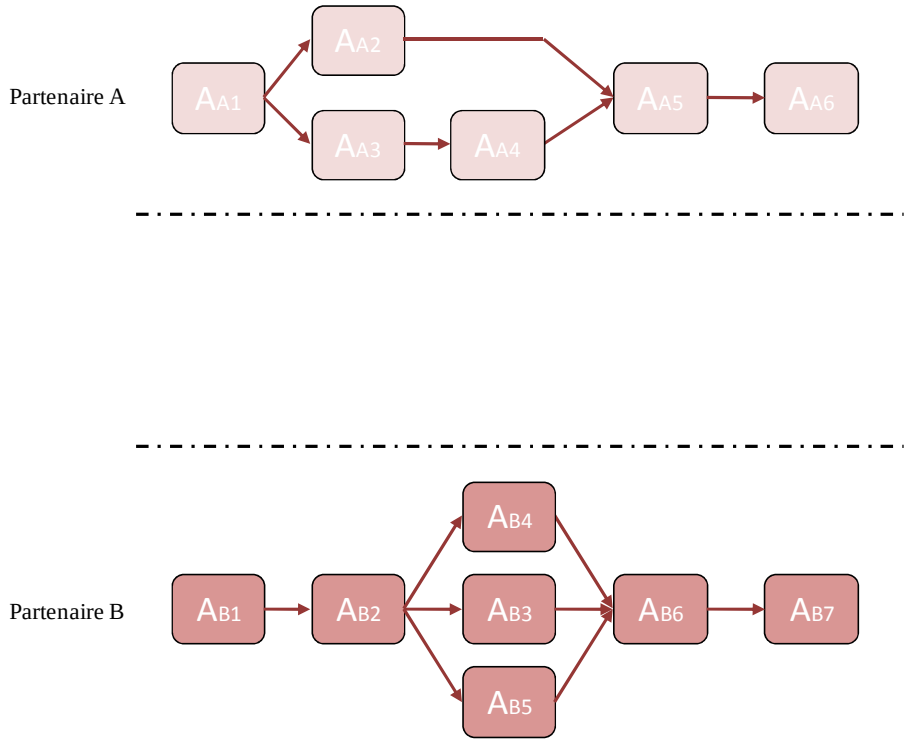


Figure 1. Illustration de deux processus collaboratifs privés

2.2.2. Processus collaboratifs publics : vision inter entreprises

Un processus collaboratif public implique différentes entreprises et peut se définir comme : « *un processus dont les activités appartiennent à différentes organisations* » (Aubert *et al.*, 2002).

La mise en œuvre de processus collaboratifs publics permet à une entreprise de décrire comment elle doit s'impliquer dans des activités comme le montre la Figure 2. Cette implication s'effectue en dehors de ses propres frontières, voire en dehors même de son réseau de clients, pour atteindre un objectif commun (*e.g.* de conception, de production…). Il est à noter que les activités du processus collaboratif public peuvent être réalisées par une ou plusieurs unités organisationnelles (*i.e.* des acteurs, équipes, départements) distinctes qui restent contraintes à la fois par l'atteinte de leurs objectifs propres et de l'objectif commun qui prévaut lors de la collaboration. Il est donc clair qu'une entreprise - impliquée dans un processus collaboratif - doit réfléchir, améliorer et optimiser ses capacités à collaborer et à interagir avec d'autres entreprises.

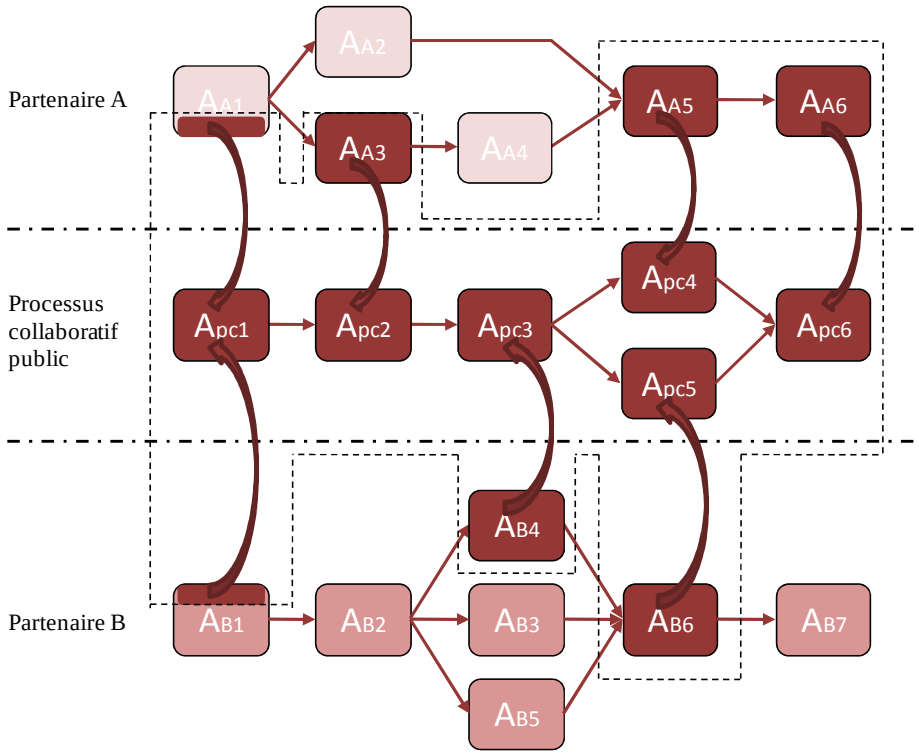


Figure 2. Illustration d'un processus collaboratif public

2.3. De la mise en œuvre d'une collaboration à l'interopérabilité

Un processus collaboratif met en œuvre, éventuellement de manière temporaire au regard de la durée nécessaire de la collaboration, des mécanismes de coordination, d'interaction et d'échange entre les différentes unités organisationnelles impliquées (Touzi, 2007). En conséquence, la réalisation effective de ce processus repose (1) sur la synchronisation et la coordination des unités organisationnelles, (2) sur la définition des rôles au regard des objectifs internes et communs, ainsi que celle des capacités et des ressources des unités organisationnelles et, (3) sur les capacités des ces unités organisationnelles à interagir avec les autres c'est-à-dire leur capacité à être interopérables. C'est ce troisième point qui est l'objet de ces travaux de recherche.

Nous allons montrer dans ce qui suit les différents aspects à considérer pour garantir une certaine interopérabilité à chaque partenaire lorsqu'il souhaite s'impliquer dans des processus collaboratifs. Pour faciliter la lecture, et afin d'adresser simultanément plusieurs niveaux de détail dans l'entreprise, nous allons parler d'**interopérabilité des systèmes** en définissant un système de manière générique comme « *quelque chose (n'importe quoi, présumé identifiable) qui dans quelque chose (environnement), pour quelque chose (finalité*

ou projet), fait quelque chose (activité = fonctionnement) par quelque chose (structure = forme stable) qui se transforme dans le temps (évolution) » (Le Moigne, 1977).

2.3.1. Interopérabilité et compatibilité

Le premier aspect que nous pouvons mettre en avant dans le développement de l'interopérabilité au niveau d'un processus collaboratif est la **compatibilité**. En effet, la compatibilité consiste à harmoniser les systèmes entre eux. Si deux systèmes sont compatibles, il est possible de les faire fonctionner ensemble sans ajouter, en théorie, aucun élément supplémentaire et donc sans nécessaire effort d'interfaçage. Cette notion de compatibilité est une conséquence du concept d'interopérabilité mais la contre apposée n'est pas toujours vraie. Ainsi, des systèmes interopérables sont forcément compatibles, cependant des systèmes qui sont compatibles ne sont pas forcément interopérables (Panetto, 2006), (Kasunic *et al.*, 2004).

2.3.2. Interopérabilité et interopération

Au delà de la compatibilité, l'interopérabilité peut être caractérisée est étudié lors de la phase opérationnelle du processus. Nous parlons alors d'**interopération**, c'est-à-dire que l'acte d'échange/partage et d'exploitation est « physiquement » réalisé. Il peut s'agir, par exemple, de s'assurer qu'un échange de donnée est bien réalisé, qu'une ressource sollicitée par une activité du processus est bien disponible ou encore que des activités sont correctement synchronisées… Il est également possible de considérer l'interopération en termes de performance. Dans ce cas, nous nous intéressons à la performance de l'échange et de l'exploitation en accord avec des critères identifiés (Kasunic *et al.*, 2004) (Daclin, 2007).

2.3.3. Interopérabilité et autonomie

L'interopérabilité est déployée lorsqu'une opportunité de partenariat apparait. Pour le bon déroulement de la collaboration tout au long du partenariat, les partenaires doivent assurer, d'une part, les objectifs (de performance par exemple) communs et relatifs à la mission à accomplir, et d'autre part, leurs objectifs propres (par exemple de performance interne). En effet, l'autonomie ne peut être dissociée de l'interopérabilité comme le précise le projet INTEROP en indiquant que des partenaires « *peuvent échanger des services, tout en continuant leur propre logique de fonctionnement* » lorsque l'interopérabilité est mise en œuvre (INTEROP, 2007). Nous retrouvons également cette notion d'autonomie dans les

systèmes de systèmes où « *un système donné doit opérer de manière autonome tout en continuant de fournir les capacités fonctionnelles requises par le système de systèmes* » (Luzeaux, 2008). En conséquence, arriver à mettre en œuvre l'interopérabilité sans pour autant dépendre de la collaboration et oublier ou négliger d'assurer les propres missions de chaque partenaire devient un enjeu important à prendre en considération

Cette caractéristique d'autonomie et liée à deux aspects. Le premier aspect concerne l'**autonomie de gouvernance**. La gouvernance désigne l'ensemble des mesures à prendre, des règles à appliquer et des décisions à adopter pour un partenaire afin d'assurer son **bon fonctionnement et son contrôle**. Plus généralement, l'autonomie de gouvernance est liée à la capacité d'un partenaire à prendre ses propres décisions tout en prenant en considération les décisions prises dans le cadre du partenariat. Le second aspect représente l'**autonomie opérationnelle** qui consiste, pour un partenaire, à garder sa **liberté dans l'exécution de son propre travail**.

2.3.4. Interopérabilité et réversibilité

Nous avons vu qu'un processus collaboratif public peut impliquer une partie (ou l'ensemble) de l'activité d'un processus collaboratif privé d'une entreprise. Cependant, à la fin de la collaboration, les entreprises doivent être en mesure de retrouver leur performance originale (avec des variations positives ou négatives acceptables). Cela signifie que, si la mise en œuvre de l'interopérabilité peut conduire à des adaptations ou des modifications (*e.g.* méthodes de travail, outils, structure organisationnelle...), les entreprises doivent s'assurer que l'implantation de solutions, permettant la mise en œuvre de l'interopérabilité, n'impacte pas de façon négative leur propre fonctionnement à la fin de la collaboration.

3. Problématique d'anticipation et de détection de problèmes d'interopérabilité

Les définitions et premiers travaux évoqués plus haut laissent penser que l'interopérabilité est réellement un enjeu majeur qui n'est certes pas nouveau mais qui a gagné en maturité en termes de définition, de formalisation et de propositions de travaux. Nous tentons dans cette dernière partie de nous focaliser sur la contribution attendue de ce travail. L'objectif est en effet de proposer une **démarche anticipative** (section 3.1) permettant une

détection locale (section 3.2) et enfin la **formalisation** de la notion d'interopérabilité (section 3.3).

3.1. Nécessité de développer une démarche anticipative

Pour diverses raisons (temps de réaction, temps de préparation, disponibilité des ressources, ou encore une confiance excessive) les entreprises sont souvent amenées à devoir solutionner des problèmes seulement quand ils apparaissent, c'est-à-dire durant l'exécution d'un processus collaboratif dans lequel elles sont impliquées. Nous nous focalisons, ici, seulement sur des problèmes liés à des défauts de compatibilité, d'interopération, de réversibilité ou d'autonomie. Pour s'assurer de leurs capacités à s'impliquer correctement et avec des retours positifs dans ce processus, une entreprise ne peut se permettre d'attendre et de gérer au coup par coup ces éventuels problèmes liés au final à sa propre incapacité à être et à rester interopérable durant la collaboration. Elle doit pouvoir détecter, le plus tôt possible, ces problèmes et en analyser les causes. L'hypothèse principale de nos travaux de recherche est qu'une identification des problèmes d'interopérabilité avant l'exécution du processus collaboratif est un des leviers d'action permettant de maîtriser l'efficacité de la collaboration. Nous nous positionnons donc clairement dans une démarche basée sur des modèles de processus collaboratif.

Il faut noter que peu de méthodes et d'outils sont disponibles sur le marché pour envisager l'interopérabilité de cette manière c'est-à-dire *a priori*. De plus, ceux-ci se concentrent essentiellement sur l'étude de l'interopérabilité séparée des partenaires et ne se focalisent pas réellement sur la réalisation du processus dans son ensemble.

3.2. Nécessité de détecter les problèmes d'interopérabilité

La détection de problèmes peut être envisagée à un **niveau local**. Il s'agit ici de localiser précisément les éléments - engagés dans le processus - initiateurs d'un défaut d'interopérabilité et la ou les causes qui font de cet élément une source potentielle de dysfonctionnement. Il peut s'agir de causes liées aux comportements, aux rôles, aux capacités, aux fonctions de cet élément. Plus l'identification d'un problème est fine, plus des actions correctives efficaces (*e.g.* temps de mise en œuvre et coût de la solution) peuvent être trouvées et implantées. Cette localisation nécessite donc de clarifier puis de vérifier quelles sont les **exigences d'interopérabilité** devant être respectées par les éléments du processus.

Cependant, la détection locale de problèmes d'interopérabilité ne saurait être, à elle seule, une condition nécessaire et suffisante pour permettre à un système de remplir sa mission et d'atteindre ses objectifs. Il est en effet nécessaire d'envisager aussi la détection de défauts à un **niveau global** car un problème identifié localement peut avoir des effets se propageant dans tous le processus et impactant d'autres éléments. Il s'agit, ici, de constater l'effet d'un problème, à un moment donné, sur l'ensemble de la structure, aussi bien en termes (inspirés des définitions proposée par l'approche SAGACE (Penalva, 1993)) :

- **D'intégrité.** Dans ce cas, les éléments impactés peuvent-ils continuer à inter opérer ?
- **De stabilité.** On s'intéresse ici à voir si les éléments impactés peuvent s'adapter pour revenir à leur niveau d'interopérabilité initialement prévu ?
- **De performance.** Cet aspect s'attache à savoir si les performances d'interopérabilité initiales des éléments impactés peuvent toujours être atteintes ?

3.3. Nécessité de formaliser l'interopérabilité

La détection de manière anticipative - sur des modèles de processus - de défauts d'interopérabilité est donc un levier d'action nécessaire, et bien sûr non suffisant, pour s'assurer de l'efficacité de réalisation d'un processus collaboratif supporté par un collectif de partenaires.

Pour l'instant, les entreprises s'appuient essentiellement sur leur expertise pour déterminer leurs rôles dans un processus et présument souvent de leurs capacités réelles à interopérer avec les autres. Ce type d'approche, non remis en cause dans nos travaux, présente cependant les faiblesses liées à l'implication de l'acteur humain, à savoir la mauvaise interprétation d'un défaut, la non détection de défauts, la remise en cause (non véracité) du défaut constaté… D'autres techniques, plus formelles, pourraient permettre de pallier certains défauts de l'expertise bien que leur mise en œuvre nécessite, *de facto*, un niveau de formalisation accrue. Dans le contexte restreint de nos travaux, il s'agit de formaliser le concept d'interopérabilité et plus précisément les exigences d'interopérabilité à respecter pour détecter d'éventuels problèmes dans un processus collaboratif. La difficulté réside alors dans le fait de proposer une vision formelle appliquée à des modèles de systèmes qui restent « *faiblement formalisables* » (Vallespir, 2003).

La question de formaliser l'interopérabilité pour prouver que des systèmes interagissent correctement pour accomplir leur mission commune n'est pas nouvelle. Des approches s'attachent à mettre en avant l'interopérabilité sous un angle formel (Castanet *et al.*, 1994). Cependant, ces approches restent encore marginales et ne considèrent, bien souvent, que les aspects techniques de l'interopérabilité et sous-estime donc les aspects conceptuels et organisationnels.

En conséquence, il apparaît nécessaire de proposer un **modèle formel** qui considère les trois aspects de l'interopérabilité (conceptuel, technique et organisationnel) pour permettre une détection anticipative des problèmes, de manière précise, la plus exhaustive qui soit et « *non discutable* ».

4. Conclusion

Le développement de l'interopérabilité est devenu un enjeu majeur pour les entreprises désirant évoluer sereinement dans un environnement de plus en plus collaboratif. L'interopérabilité est une caractéristique et, au final, une aptitude essentielle à maîtriser pour l'amélioration et la facilitation des interactions entre des entreprises en situation de collaboration. Dans ce premier chapitre, nous avons mis en évidence la nécessité de développer l'interopérabilité au sein des entreprises et plus particulièrement dans les processus collaboratifs. Nous avons ensuite démontré que cette maîtrise passe par l'anticipation des problèmes liés à des défauts de compatibilité, d'interopération, de réversibilité ou d'autonomie. Par anticipation, nous sous-entendons logiquement une démarche d'analyse *a priori* des problèmes d'interopérabilité venant à la fois des systèmes qui sont impliqués mais aussi du processus tout entier. Par hypothèse, nous supposons que cette démarche repose naturellement sur un travail de modélisation des processus, un travail de formalisation de l'interopérabilité sous forme d'exigences puis un travail de détection et d'analyse avant même la mise en œuvre du processus collaboratif en question. Cette détection ne peut cependant se faire que sur des modèles et doit être envisagée à des niveaux de détail différents pour assurer l'intégrité, la stabilité et la performance du processus collaboratif au regard de la mission commune et des objectifs propres de la collaboration. Cette détection doit donc être basée sur une formalisation des exigences d'interopérabilité et permettre de s'assurer que ces exigences sont bien respectées au moyen de technique de vérification outillée.

Dans le chapitre suivant, nous présentons une analyse des travaux concernant le développement de l'interopérabilité au travers d'un état de l'art. Cet état de l'art couvre, d'un côté, des travaux liés à l'évaluation et la mesure d'interopérabilité, et de l'autre, des travaux autour de techniques de vérification.

Chapitre 2

Interopérabilité et vérification : état de l’art

Table des matières du chapitre 2

1. Introduction

Nous avons établi dans le chapitre précédent le besoin potentiel des entreprises de s'impliquer de plus en plus dans des processus collaboratifs. Il apparait alors pertinent de pouvoir aider ces entreprises à connaître et essayer de résoudre d'éventuels problèmes. En effet, ceux-ci peuvent être à l'origine de défauts, de dysfonctionnements ou de pertes de performance de l'entreprise lorsqu'elle prend part à des processus collaboratifs. Nous nous intéressons particulièrement ici à la **détection** et à la **caractérisation** de problèmes liés à des défauts d'interopérabilité en soulignant que cette détection doit être :

- **Anticipative,** c'est-à-dire avant que toute exécution du processus collaboratif ne soit effectuée.
- **Locale** puis **globale,** c'est-à-dire limitée dans un premier temps la détection aux seules parties prenantes de l'entreprise impliquées dans le processus collaboratif. Par la suite, cette détection peut être généralisée à l'ensemble du processus, par exemple, en propageant les effets des problèmes potentiels pour en détecter de nouveaux non plus au niveau local mais au niveau global de la collaboration.
- **Basée sur des concepts éprouvés,** c'est-à-dire prenant en compte les concepts nécessaires à la définition de l'interopérabilité de manière adéquate et pertinente au regard des objectifs du processus collaboratif.
- **Outillée techniquement,** c'est-à-dire fournissant un support technique pour détecter ces problèmes.

De fait, ce travail de recherche vise à développer une démarche d'ingénierie de processus collaboratifs permettant de détecter localement - puis globalement – et de manière anticipative un problème d'interopérabilité. Il est construit autour d'une hypothèse globale stipulant que : « *la détection anticipative d'éventuels problèmes d'interopérabilité consiste à s'assurer que certaines exigences d'interopérabilité sont vérifiées sur un modèle de processus collaboratif afin d'indiquer si des améliorations sont nécessaires, avant que l'entreprise ne participe au processus* ».

Avant d'introduire les éléments de cette démarche, ce chapitre présente un état de l'art sur les connaissances et les travaux concernant, d'une part, la caractérisation et la mesure d'interopérabilité, et d'autre part, la vérification.

2. Interopérabilité : travaux existants

2.1. Les cadres d'interopérabilité

Plusieurs travaux de recherche se sont intéressés à structurer, catégoriser et caractériser le concept d'interopérabilité. A cet effet, la plupart de ces travaux proposent la mise en place de **cadres d'interopérabilité**. En effet, le terme cadre se réfère à un mécanisme pour structurer et classer quelque chose par rapport à un domaine (ATHENA, 2007). Cette notion de cadre est apparue dans les années 80, plus particulièrement, dans le domaine de la modélisation d'entreprise afin d'homogénéiser les différents modèles existants. Les premiers cadres apparus tels que le cadre MERISE (Tardieu, 1983) et le cadre ZACHMAN (ZACHMAN, 1987) sont principalement destinés aux systèmes d'information. Plus récemment, dans le domaine de la modélisation d'entreprise, les cadres tels que CIMOSA (AMICE, 1993) et GIM (*GRAI Integrated Methodology*) (Doumeingts, 1993) ont été développés pour permettre de représenter l'entreprise selon différents points de vue. Bien que ces nombreux cadres permettent d'identifier et de structurer les concepts liés aux systèmes d'information ou de production, ils n'ont pas vocation à prendre en compte le concept d'interopérabilité.

Ainsi, des cadres spécifiquement dédiés à l'interopérabilité, ont été développé ces dernières années pour structurer, catégoriser et caractériser le concept d'interopérabilité. Dans la suite, nous présentons, les principaux cadres d'interopérabilité : le cadre IDEAS, le cadre AIF (*ATHENA Interoperability Framework*), le cadre INTEROP et le cadre EIF (*Enterprise Interoperability Framework*), qui nous permettent, par la suite, de statuer sur le types de problèmes d'interopérabilité qui peuvent apparaître lors d'une collaboration.

2.1.1. Le cadre IDEAS

Le cadre d'interopérabilité développé dans le projet IDEAS (IDEAS, 2003) a été élaboré dans le but de structurer, collecter et identifier les visions sur l'interopérabilité des applications d'entreprise. Trois niveaux d'interopérabilité ont été identifiés : le niveau

organisation (*Business*), le niveau **connaissance** (*Knowledge*) et le niveau **application** (*ICT systems*).

- **Le niveau business.** Il est relatif aux problèmes liés à l'organisation et aux processus métiers des entreprises. Il est plus précisément décomposé en trois sous-niveaux : le modèle décisionnel, le processus industriel et le modèle business.
- **Le niveau connaissance (*knowledge*).** Il concerne la structuration des connaissances de l'entreprise selon trois sous niveaux : l'organisation des rôles, les compétences et la représentation des connaissances de l'entreprise.
- **Le niveau application (*ICT systems*[3]).** Il concerne les solutions technologiques pour la collaboration des ressources d'une entreprise avec d'autres. Il est décomposé en trois sous niveau : application, données et infrastructures de communication.
- **La vue sémantique (*semantic*).** Elle couvre les trois niveaux susmentionnés pour assurer une compréhension mutuelle entre eux.

Le cadre IDEAS indique les problèmes à résoudre pour établir l'interopérabilité au travers d'une sémantique commune en précisant les vues (sous-niveaux) à prendre en considération pour chaque niveau. Nous pouvons constater que trois points de vue sont clairement identifiés pour structurer les problèmes d'interopérabilité : un point de vue **organisationnel** (niveau organisation), un point de vue **conceptuel** (niveau connaissance) et un point de vue **technique** (niveau application). Ces points de vue correspondent à la catégorisation classique des problèmes d'interopérabilité. Le projet IDEAS est l'un des premiers à proposer un cadre afin de structurer l'interopérabilité d'entreprise selon différents niveaux. Il a posé les premiers fondements de l'interopérabilité et est à l'origine d'autres cadres développés dans divers projets européens.

2.1.2. Le cadre AIF

Le cadre AIF a été développé dans l'objectif de synthétiser les résultats de recherche du projet ATHENA (ATHENA, 2007). Trois niveaux concernant l'intégration sont identifiés : **l'intégration conceptuelle** (*conceptual integration*), **l'intégration applicative** (*applicative integration*) et **l'intégration technique** (*technical integration*).

[3] Information and Communication Technology

- **L'intégration conceptuelle.** Elle met l'accent sur les concepts, les méta-modèles et les langages. Cette intégration fournit une base de modélisation pour conceptualiser les divers aspects de l'interopérabilité.
- **L'intégration applicative.** Elle met l'accent sur les méthodologies, normes et modèles de domaine. Cette intégration fournit des lignes directrices, principes et modèles qui peuvent être utilisés pour résoudre les problèmes d'interopérabilité.
- **L'intégration technique.** Elle se concentre sur le développement technique et l'environnement des applications (*system ICT*). Cette intégration fournit des outils et des plateformes pour développer et exécuter des applications d'entreprise et de systèmes logiciels.

Le cadre AIF fournit un ensemble de concepts, de pratiques et de solutions pour répondre à la problématique d'interopérabilité selon trois niveaux : conceptuel, applicatif et technique. Nous pouvons constater que nous retrouvons le point de vue technique (niveau applicatif et niveau technique) et le point de vue conceptuel (niveau conceptuel). Cependant nous ne retrouvons pas le point de vue organisationnel qui est clairement identifié comme un problème d'interopérabilité à part entière. Enfin, ce cadre reste générique et ne permet pas de localiser les problèmes identifiés dans les niveaux car il ne propose pas des vues à prendre (*e.g.* processus) en considération.

2.1.3. Le cadre INTEROP

Le cadre d'interopérabilité développé dans le projet INTEROP (INTEROP, 2007) vise à structurer les notions de bases pour l'interopérabilité d'entreprise. Ce cadre a deux dimensions de base : les barrières d'interopérabilité, et les niveaux d'interopérabilité. Par la suite, une troisième dimension a été ajoutée pour permettre la prise en compte des approches d'interopérabilité. La Figure 3 illustre les trois dimensions de ce cadre d'interopérabilité.

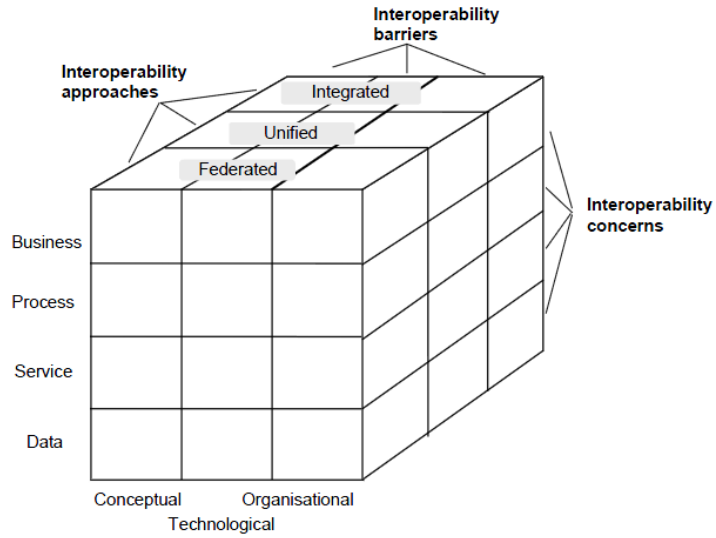


Figure 3. Le cadre d'interopérabilité INTEROP (INTEROP, 2007)

- **Les barrières d'interopérabilité (*interoperability barriers*).** Elles représentent les problèmes (obstacles) d'interopérabilité identifiés. Le terme « barrière » signifie incompatibilité ou non-concordance qui empêche le partage et l'échange d'informations. Trois catégories de barrières sont identifiées : conceptuelle, technologique et organisationnelle.
- **Les niveaux d'interopérabilité (*interoperability concerns*).** Ils représentent les niveaux de l'entreprise où les barrières d'interopérabilité peuvent apparaître. Quatre niveaux sont identifiés : données, services, processus et business.
- **Les approches d'interopérabilité (*interoperability approaches*).** Elles représentent les approches proposées par la norme ISO14258 (ISO14258, 1999) pour développer l'interopérabilité, c'est-à-dire l'intégration, l'unification et la fédération.

Le cadre INTEROP sert à structurer les connaissances sur l'interopérabilité ainsi que les solutions susceptibles de résoudre les problèmes d'interopérabilité. Ce cadre ne propose pas seulement d'identifier les barrières d'interopérabilité - qui correspondent à la catégorisation classique - à un niveau de l'entreprise et de proposer des solutions pour les résoudre, mais d'étudier également la façon dont ces barrières sont levées. Ainsi, une solution est considérée comme satisfaisante si elle contribue à lever une barrière à un niveau considéré et, en respectant une des trois approches fournit par la norme.

2.1.4. Le cadre EIF

Le cadre EIF (EIF, 2004) développé dans le contexte du PEGS (*Pan-European eGovernment Services*) a pour objectif de fournir des services en ligne en facilitant l'interopérabilité des services afin de soutenir la stratégie de l'Union Européenne. Pour ce faire, trois dimensions sont déterminées pour faire face à toute la problématique de l'interopérabilité comme présenté en Figure 4 (EIF, 2008):

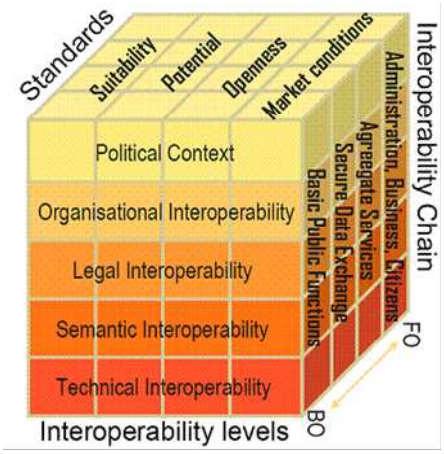


Figure 4. Le cadre d'interopérabilité EIF (EIF, 2008)

- **Les niveaux d'interopérabilité (*interoperability levels*).** Cette dimension classe les problèmes d'interopérabilité, qu'ils soient d'ordre technique, sémantique, légal, organisationnel ou encore politique.
- **La chaîne d'interopérabilité (*interoperability chain*).** Cette dimension perçoit l'interopérabilité comme un phénomène, comme quelque chose de construit progressivement au fil du temps par l'intermédiaire de « blocs de construction ».
- **Les normes d'interopérabilité (*standards*).** Cette dimension est concernée par le cahier des charges et/ou les décisions qui régissent en détail la façon dont l'interopérabilité est mise en œuvre. L'évaluation et la sélection de ces standards facilite les échanges d'informations et l'intégration des composants.

Le cadre EIF examine la possibilité de coopérer et d'échanger des informations sur les plans politiques, organisationnels, juridiques, sémantiques, et technologiques. Nous constatons que, ce cadre couvre les problèmes d'ordre organisationnels, conceptuels et techniques avec la dimension des niveaux d'interopérabilité. Il offre également un point de

vue temporel quand à la construction progressive de l'interopérabilité en utilisant des normes pour faciliter sa mise en œuvre.

2.1.5. Synthèse

Les différents cadres que nous venons de présenter sont exclusivement dédiés au domaine de l'interopérabilité. Ils permettent de décliner l'interopérabilité selon différents points de vue et perceptions à prendre en compte dans l'optique de l'implémenter. Ainsi, bien qu'ils considèrent des dimensions différentes, leur primo-objectif est commun et s'attache à permettre à des systèmes de faire face à des problèmes d'interopérabilité. Nous pouvons également noter que chacun d'eux s'articule autour des trois problèmes majeurs de l'interopérabilité *i.e.* les problèmes conceptuels, organisationnels et techniques. Ensuite certains de ces cadres (IDEAS, INTEROP) considèrent les différents niveaux où les entreprises souhaitent améliorer leurs capacités à l'interopérabilité (*e.g.* décisionnel, processus, donnée…) avec la résolution des problèmes identifiés.

Pour le travail de recherche proposé, un cadre d'interopérabilité est nécessaire pour structurer les exigences d'interopérabilité aussi finement que possible. Ce cadre doit prendre en considération les deux aspects (problèmes et niveaux) qui ne doivent pas être négligés pour autoriser une localisation précise d'un ou de plusieurs problèmes d'interopérabilité. Il doit également nous permettre d'élaborer une liste de besoins en interopérabilité à partir desquels des exigences peuvent, par la suite, être spécifiées.

Plusieurs études sur les cadres d'interopérabilité ont été proposées dans divers travaux de recherche (Chen, 2011). Une étude comparative est présentée dans le Tableau 1 où nous nous intéressons à l'apport de ces cadres, pour nos travaux, en termes (1) d'expression de besoins d'interopérabilité, (2) d'identification des problèmes d'interopérabilité de base (organisationnel, technique et conceptuel), (3) de prise en compte de vues qui permettent de structurer les problèmes identifiés et (4) de proposition de solution pour résoudre les problèmes identifiés.

Tableau 1. Etude comparative des cadres d'interopérabilité[4]

	Besoins	Vues	Problèmes	Solutions
IDEAS	+	++	+++	-
AIF	+	-	++	++
INTEROP	++	+++	+++	+++
EIF	+	-	+++	++

Enfin, ces cadres d'interopérabilité ne permettent pas de mesurer et d'évaluer l'interopérabilité. Leur vocation principale est de fournir à des systèmes un ensemble de solutions à des problèmes d'interopérabilité identifiés et adaptés à leurs besoins. Là encore, dans le cadre de nos travaux, cette dimension - de mesure et d'évaluation - doit être pleinement intégrée pour pouvoir indiquer à des partenaires leur réelle capacité au regard de l'interopérabilité. Cette dernière constatation nous conduit naturellement à nous intéresser dans la section suivante aux différents travaux liés à l'évaluation et à la mesure d'interopérabilité.

2.2. Les modèles pour la mesure et l'évaluation d'interopérabilité

Divers travaux ont été développés pour permettre aux entreprises de connaître leurs points forts et leurs points faibles au regard du concept d'interopérabilité. Dans ce cadre, deux paradigmes pour la mesure et l'évaluation de l'interopérabilité ont émergés. D'une part, les modèles de maturité, ont pour objectifs de décrire des niveaux pour évaluer l'interopérabilité des systèmes. Un niveau caractérise l'état du système ou de l'organisation - à un instant donné - vis-à-vis de son interopérabilité. Ces modèles fournissent également des recommandations pour passer d'un niveau à l'autre afin d'atteindre la pleine interopérabilité. D'autre part, des modèles plus formels proposent de mesurer l'interopérabilité de manière quantitative. Ces modèles offrent des outils et des métriques basés sur les différents critères de l'interopérabilité.

[4] Nous adoptons les notations suivantes pour l'évaluation que nous proposons.
- « +++ » signifie que le cadre répond le mieux au critère.
- «++» désigne que le cadre répond au critère.
- « +» indique que le cadre répond faiblement au critère.
- «-» signifie que le cadre ne répond pas ou n'est pas adapté pour répondre au critère.

Dans les sections suivantes, nous présentons les modèles de maturité majeurs ainsi que des outils de mesure quantitative de l'interopérabilité.

2.2.1. Les modèles de maturité

La question de mesurer et/ou d'évaluer le niveau de maturité en interopérabilité d'une entreprise s'est posée, depuis quelques années déjà, avec l'apparition de modèles de maturité pour l'interopérabilité. Un modèle de maturité a pour objectif de décrire les niveaux par lesquelles passe un système ainsi que des recommandations pour progresser à travers les différents niveaux. Les modèles de maturité ont été initiés, dans les années 1980 avec le développement du modèle *Capability Maturity Model* (CMM) (Paulk *et al.,* 1995). Ce modèle est à l'origine d'autres modèles de maturité tels que le *Capability Maturity Model Integration* (CMMI) (CMMI, 2002) (Basque, 2004), le modèle de maturité COBIT (Sallé *et al.,* 2005) (Simonsson *et al.,* 2006) ou encore le modèle de maturité SOAMM (Bachman, 2005). Plus récemment, des modèles de maturité spécifiquement dédiés à l'interopérabilité ont été développés. Nous présentons, dans la suite, les principaux modèles de maturité pour l'interopérabilité au travers des différents niveaux qu'ils proposent.

a. Le modèle de maturité LISI

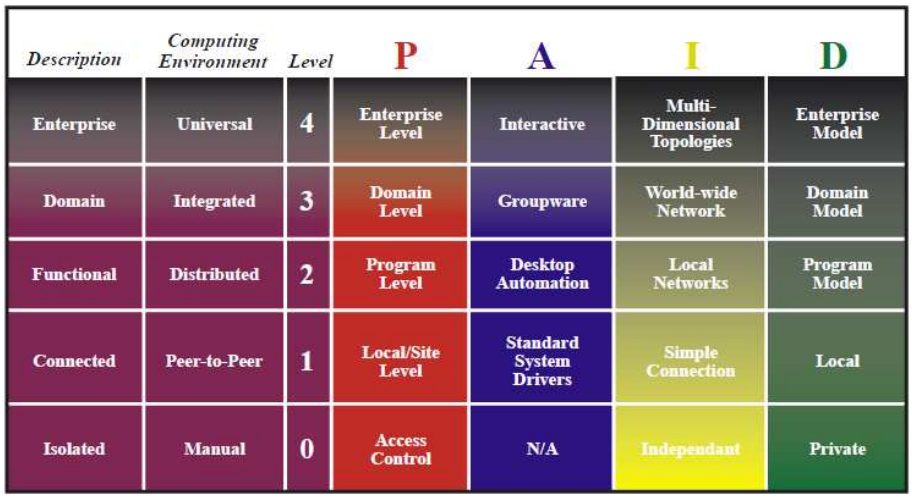
Le modèle de maturité LISI (*Levels of Information Systems Interoperability*) est le premier modèle proposé pour évaluer l'interopérabilité des systèmes d'information (C4ISR, 1998). Le modèle LISI identifie cinq niveaux au travers desquels les systèmes progressent afin d'améliorer leurs capacités à interagir.

- **Niveau 0 : Isolé (*Isolated*).** L'interopérabilité est reliée à un environnement manuel. En effet, aucun lien direct électronique n'est autorisé, ni disponible. La seule interface entre ces systèmes est manuelle ou *via* des médias communs.
- **Niveau 1 : Connecté (*Connected*).** L'interopérabilité est reliée à un environnement peer-to-peer[5]. Les systèmes sont susceptibles d'être reliés électroniquement et de simples échanges électroniques sont permis. Ces systèmes ont une capacité limitée. Ils permettent l'échange de données uniquement homogènes et leurs applications sont séparées.

[5] Pair-à-pair.

- **Niveau 2 : Fonctionnel (*Functional*).** L'interopérabilité est dite fonctionnelle dans un environnement distribué. Les systèmes se trouvent sur des réseaux locaux qui permettent la transmission de l'ensemble des données d'un système à l'autre. Des échanges de données hétérogènes sont possibles.
- **Niveau 3 : Domaine (*Domain*).** Les systèmes évoluent dans un environnement intégré. Les systèmes sont capables d'être connectés *via* des réseaux étendus qui permettent à plusieurs utilisateurs d'accéder aux données. A ce niveau les données sont partagées entre des applications indépendantes.
- **Niveau 4 : Entreprise (*Enterprise*).** Les systèmes fonctionnent dans un environnement universel. Les systèmes sont capables de fonctionner en utilisant un espace d'information distribué mondial dans de multiples domaines. Plusieurs utilisateurs peuvent accéder et interagir avec des données complexes en même temps. Les données et les applications sont entièrement partagées et peuvent être distribués dans tout cet espace.

Le modèle LISI est organisé en niveaux de maturité qui représentent des capacités de plus en plus évoluées ainsi que les environnements informatiques associés qui les soutiennent. Cependant, dans chacun de ces niveaux de maturité, de nombreux autres facteurs influencent la capacité d'interaction des systèmes d'information. En conséquence, le modèle LISI classe ces facteurs en quatre attributs PAID pour : *Procédures*, *Applications*, *Infrastructure* et *Données*. Ces attributs sont suffisamment larges pour englober, par définition, une gamme complète de facteurs où l'interopérabilité est considérée. Le modèle de référence complet LISI est présenté dans la figure suivante.



Description	*Computing Environment*	*Level*	P	A	I	D
Enterprise	Universal	4	Enterprise Level	Interactive	Multi-Dimensional Topologies	Enterprise Model
Domain	Integrated	3	Domain Level	Groupware	World-wide Network	Domain Model
Functional	Distributed	2	Program Level	Desktop Automation	Local Networks	Program Model
Connected	Peer-to-Peer	1	Local/Site Level	Standard System Drivers	Simple Connection	Local
Isolated	Manual	0	Access Control	N/A	Independant	Private

Figure 5. Le modèle de maturité LISI (C4ISR, 1998)

Ce modèle présente, à chaque niveau, un mot ou une phrase qui met en évidence l'aspect le plus important pour atteindre ce niveau. Cependant, ce modèle n'évalue que l'aspect technique de l'interopérabilité. Nous nous intéressons dans la section suivante à un modèle de maturité destiné plus particulièrement à l'aspect organisationnel de l'interopérabilité.

b. Le modèle de maturité OIM

Le modèle OIM (*Organisational Interoperability Model*) (Clark *et al.*, 1999) est une « extension » du modèle LISI afin de prendre en compte l'aspect organisationnel de l'interopérabilité. Cinq niveaux de maturité sont alors identifiés :

- **Niveau 0 : Indépendant.** Ce niveau décrit l'interaction entre des organisations indépendantes. Ce sont des organisations qui travaillent normalement sans aucune intervention autre que celle prévue par le contact direct entre les personnes.
- **Niveau 1 : Ad hoc.** A ce niveau, très peu de cadres organisationnels sont mis en place pour appuyer des arrangements *ad hoc*. Des lignes directrices pour décrire les interactions sont mises en œuvre, mais les dispositions spécifiques sont toujours imprévues.
- **Niveau 2 : Collaboratif.** A ce niveau, des cadres sont mis en place pour soutenir l'interopérabilité, les objectifs communs sont identifiés et les rôles et les responsabilités sont attribués. Cependant les organisations restent encore distinctes.

- **Niveau 3 : Combiné.** A ce niveau, il y a partage des systèmes de valeurs et des objectifs communs. Une compréhension commune et une préparation à l'interopérabilité sont mises en place.
- **Niveau 4 : Unifié.** A ce niveau, les objectifs organisationnels, les systèmes de valeurs, la structure de commandement, et les bases de connaissances sont partagées par les organisations. Les organisations sont capables d'interopérer sur une base continue.

De plus, quatre attributs ont été identifiés comme permettant l'interopérabilité organisationnelle : *Preparedness*, *Understanding*, *Command Style* et *Ethos*. Enfin, les niveaux de maturité du modèle OIM sont alignés avec les niveaux du modèle LISI comme schématisé ci-dessous.

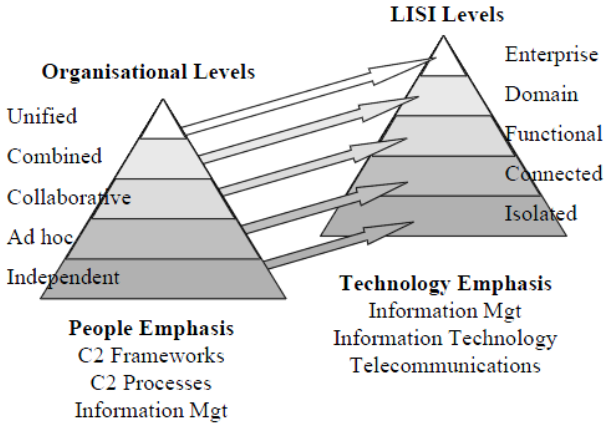


Figure 6. Alignement du modèle OIM avec le modèle LISI

Le modèle OIM est une extension du modèle LISI dans le contexte de l'organisation. L'objectif principal de ce modèle est d'indiquer un niveau d'interopérabilité organisationnel adéquat permettant aux systèmes d'interagir.

c. Le modèle de maturité LCIM

Le modèle LCIM (*Levels of Conceptual Interoperability Model*) (Turnitsa, 2005) (Tolk *et al.*, 2007) a été développé pour combler le fossé entre les méthodes de mise en œuvre ciblées technologie et les modèles conceptuels. Ce modèle a été élaboré de manière à être aussi simple que possible dans le but de faciliter les discussions entre toutes les communautés. L'accent est mis sur les données à échanger et la documentation de l'interface, qui est

disponible. Dans un premier temps, cinq niveaux sont identifiés (Tolk *et al.,* 2003). Par la suite, un modèle à sept niveaux de maturité a été proposé comme représenté en Figure 7.

- **Niveau 0 : Pas d'interopérabilité.** Aucune interopérabilité entre les systèmes. Les données sont utilisées au sein de chaque système d'une manière exclusive et sans partage.
- **Niveau 1 : Interopérabilité technique.** Les données sont documentées en utilisant un protocole commun et sont accessibles *via* des interfaces.
- **Niveau 2 : Interopérabilité syntaxique.** Une structure commune est introduite pour l'échange d'information *i.e.* un format de donnée commun est appliqué. A ce niveau, un protocole commun pour structurer les données est utilisé, le format de l'information échangée est défini de façon non ambigüe.
- **Niveau 3 : Interopérabilité sémantique.** Les données sont documentées grâce à un modèle de référence commun reposant sur une ontologie commune, à savoir, la signification des données est clairement décrite.
- **Niveau 4 : Interopérabilité pragmatique.** A ce niveau l'utilisation des données - ou le contexte de leur application – est compris par tous les systèmes participants. Le contexte dans lequel l'information est échangée est défini de façon non ambigüe.
- **Niveau 5 : Interopérabilité dynamique.** A ce niveau les systèmes sont capables de comprendre les changements d'état qui se produisent, au fil du temps, au niveau des systèmes avec lesquels ils interagissent, et sont capables de tirer parti de ces changements.
- **Niveau 6 : Interopérabilité conceptuelle.** A ce niveau, les modèles conceptuels sont alignés. Ces modèles sont documentés sur la base de méthodes d'ingénierie permettant leur interprétation et leur évaluation.

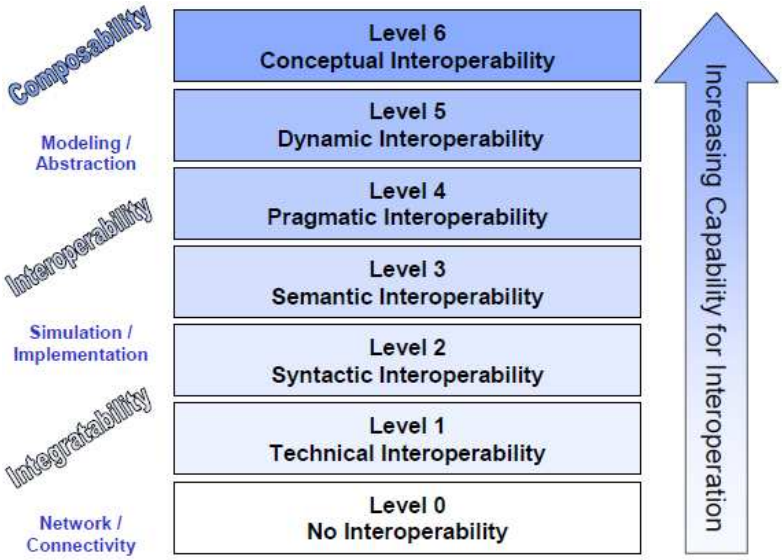


Figure 7. Les niveaux du modèle de maturité LCIM (Tolk *et al.*, 2007)

L'intérêt du modèle LCIM réside dans l'évaluation et la considération de l'interopérabilité au niveau conceptuel et non plus technique ou organisationnel.

Les modèles de maturité précités évaluent l'interopérabilité avec la proposition de niveaux de maturité. Cependant, ils ne permettent pas de donner une vision précise sur l'existence ou non de problèmes d'interopérabilité, ni même de mesurer quantitativement l'interopérabilité. Dans les sections suivantes, nous nous intéressons à des modèles développés pour mesurer l'interopérabilité de façon quantitative.

2.2.2. Le modèle pour la mesure du degré d'interopérabilité

L'objectif de la mesure d'interopérabilité est de permettre une analyse et une évaluation quantitative de l'interopérabilité d'un système donné. En effet, « *l'interopérabilité ne sera jamais un champ utile pour l'analyse et l'étude jusqu'à ce qu'elle soit définie de manière quantitative* » (Presson, 1983). La mesure de l'interopérabilité est largement étudiée et présentée dans les travaux de recherches proposés dans (Daclin, 2007) pour l'interopérabilité des entreprises. Dans ces travaux, la mesure de l'interopérabilité est envisagée sous deux aspects : **la mesure de compatibilité** et **la mesure des performances d'interopération**.

La mesure de compatibilité se fait par le biais d'une matrice de compatibilité présentée en Figure 8. Cette mesure est effectuée en tenant compte des problèmes d'interopérabilité précédemment identifiés (*i.e.* d'ordre conceptuel, technologique ou organisationnel). Ainsi,

pour chaque problème d'interopérabilité et pour chaque niveau, il s'agit de regarder s'il y a compatibilité ou incompatibilité entre les partenaires. Une liste de questions simples (*e.g.* les responsabilités, les autorités et les compétences des acteurs de l'interopérabilité sont-elles clairement identifiées ?) aux réponses binaires est alors établie. Nous pouvons constater que cette matrice se base sur le cadre d'interopérabilité développé dans le projet INTEROP précédemment présenté (INTEROP, 2007).

Barrières / Niveaux	Conceptuelle		Technologique		Organisationnelle		Barrières / Niveaux
Business	0	1	0	1	1	1	Business
Processus	0	0	1	1	0	0	Processus
Services	1	1	0	0	1	0	Services
Données	1	0	0	1	1	1	Données

Figure 8. Matrice de compatibilité

La mesure de compatibilité est alors un outil qui permet de renseigner les partenaires sur l'existence des problèmes d'interopérabilité qu'il faut annihiler. Un degré de compatibilité égal à 0 signifie qu'il n'existe pas d'incompatibilité entre les partenaires (tous les coefficients dans la matrice sont égaux à 0). Un degré de compatibilité supérieur à 0 signifie que des incompatibilités existent entre les partenaires et qu'il faut les éliminer pour le bon fonctionnement du partenariat.

La mesure de performance d'interopération permet aux entreprises de mesurer et d'évaluer l'interopérabilité selon des critères de performances. Ainsi, la mesure de la performance est réalisée durant la phase opérationnelle du partenariat, c'est-à-dire lors de l'échange et de l'exploitation des informations. Les domaines de performances d'interopération évalués ici, sont liés aux trois critères de performances primordiaux pour les entreprises, c'est-à-dire le coût, le temps et la qualité. Cette mesure comprend donc, une mesure du **coût de l'interopération**, du **temps de l'interopération** et de la **qualité de l'interopération.**

En définitive, ces travaux de recherche proposent des **outils de mesure** permettant aux entreprises d'évaluer, pour ensuite améliorer, leur capacité à interopérer. Cependant, cette mesure permet une détection locale des problèmes d'interopérabilité qui se limite à ceux identifiés dans le cadre d'interopérabilité d'INTEROP.

2.2.3. Le modèle pour la mesure de l'efficacité opérationnelle

Les travaux proposés dans (Ford, 2008), proposent également une méthode pour mesurer l'interopérabilité dans le domaine militaire. Ces travaux admettent (1) que, l'interopérabilité doit être mesurée dans un contexte d'une mission opérationnelle, (2) qu'une opération peut être mise en œuvre par des systèmes de différents types et la mesure d'interopérabilité doit tenir compte de tous ces systèmes, (3) qu'une parfaite interopérabilité n'est pas toujours souhaitable ou possible, et enfin (4) que ce n'est pas le nombre d'interopérations qui est important mais la qualité de l'interopération.

En conséquence, il est admis que la mesure d'interopérabilité peut être liée mathématiquement à la mesure de l'efficacité opérationnelle. Afin de déterminer la mesure de l'efficacité opérationnelle à laquelle la mesure d'interopérabilité sera corrélée, un processus opérationnel est utilisé. Pour effectuer cette mesure d'interopérabilité, plusieurs modes, ainsi que leurs métriques associées, sont définis tels que :

- **Le mode directionnel.** Ce mode s'intéresse aux interopérations en termes de directions. Dans ce sens, il considère que celles-ci peuvent être unidirectionnelles (émetteur vers récepteur) ou bidirectionnelles (émetteur vers récepteur et vis-versa)
- **Le mode auto.** Ce mode considère l'interopérabilité propre d'un système c'est-à-dire que l'interopération part du système considéré et « revient » sur ce même système.
- **Le mode pure.** Ce mode s'intéresse à l'interopérabilité entre deux systèmes, même si ceux-ci interopèrent avec d'autres.
- **Le mode contextuel.** Ce mode s'intéresse à l'interopérabilité entre deux systèmes, mais considère également les interopérations de ceux-ci avec d'autres systèmes.
- **Le mode temporel.** Ce mode s'intéresse à la variation de l'interopérabilité dans le temps. Plus précisément, il considère le temps sur l'échelle continue et discrète.
- **Le mode contrainte « limite supérieure ».** Ce mode part du principe que l'interopérabilité ne peut jamais être parfaite. Dans ce sens, ce mode définit une limite réaliste de performance d'interopérabilité à atteindre.

- **Le mode collaboration et confrontation.** Ce mode est particulier au domaine militaire puisque il considère les interopérations ami/ami (collaboration) et les interopérations ami/ennemi (confrontation).

Là encore, ces travaux proposent un ensemble de métriques qui permet d'évaluer l'interopérabilité. Cependant, ces travaux ne s'attachent pas détecter des problèmes précis d'interopérabilité et s'applique dans un domaine très particulier qui est le domaine militaire.

2.2.4. Synthèse

Les modèles de maturité permettent d'évaluer l'interopérabilité au regard des niveaux qu'ils définissent et de donner des recommandations pour évoluer d'un niveau à l'autre afin d'atteindre la pleine maturité. Cependant, chacun d'eux se focalise sur un seul aspect de l'interopérabilité (technologique, conceptuel et organisationnel) et ne permettent pas d'intégrer tous ces aspects à la fois. Pour pallier ce manque, les travaux de (Guédria *et. al*, 2009) proposent un modèle de maturité englobant ces trois aspects. De plus, les modèles de maturité ne permettent pas de donner une notation quantitative de l'interopérabilité ni même de la mesurer. En conséquence, nous nous sommes intéressés à deux approches différentes qui permettent de mesurer l'interopérabilité.

La première approche se focalise sur la mesure du degré d'interopérabilité par la mesure de la compatibilité et de la performance d'interopération. La seconde approche se concentre sur la mesure d'interopérabilité par le biais d'une mesure de l'efficacité opérationnelle. Ces deux approches permettent effectivement de donner une quantification de l'interopérabilité sans pour autant s'attacher à détecter des problèmes d'interopérabilité. Dans ce sens, les travaux de (Blanc, 2006) représente une première tentative. La méthode proposée s'attache à la détection de problèmes d'interopérabilité avec la proposition de règles qui restent cependant génériques et difficilement applicables.

Dans le cadre de nos travaux de recherche, les modèles de maturité nous permettent d'extraire des exigences d'interopérabilité au travers des besoins implicitement présentés dans les différents niveaux. De plus, les différents modèles présentés précédemment ne permettent pas de détecter localement des problèmes d'interopérabilité. En conséquence, afin d'avoir une idée sur les avantages que nous apportent chaque modèle pour le développement de notre approche, une étude comparative est présentée dans le Tableau 2. Dans ce tableau, nous nous intéressons (1) à l'expression des besoins d'interopérabilité, (2) à la mesure d'interopérabilité,

(3) à la formalisation de l'interopérabilité, (4) à l'utilisation de méthode pour la vérification, (5) à la détection locale et enfin (6) à l'aspect opérationnel de la collaboration.

Tableau 2. Etude comparative entre les différents modèles

	Besoins	**Mesure**	**Formalisation**	**Vérification**	**Détection locale**	**Aspect opérationnel**
Modèles de maturité	+++	-	-	-	-	-
Degré d'interopérabilité	-	+++	-	-	+	++
Efficacité opérationnelle	-	+++	-	-	-	+++
Règles d'interopérabilité	-	-	++	-	+	-

Ces modèles permettent donc effectivement d'évaluer ou de donner une indication sur l'existence ou non de problèmes d'interopérabilité sans pour autant s'attacher à localiser précisément les problèmes ou à donner une preuve formelle de leur existence, qui puisse être irréfutable. En analysant le Tableau 2, nous pouvons constater essentiellement (1) un manque de formalisation du concept d'interopérabilité, (2) un manque de moyens d'analyse, (3) un manque de précision quand à la détection de problèmes et enfin (4) un manque de considération de l'aspect opérationnel de l'interopérabilité. Ces manques sont, au final, liés au niveau de formalisation à la fois des problèmes à détecter et des concepts nécessaires pour les détecter. Ils sont également liés aux techniques qui permettraient formellement et non plus qualitativement ou quantitativement, et toujours en suivant les principes d'une approche basée sur des modèles de processus, de déceler des problèmes. Nous introduisons alors la notion de technique de vérification formelle qui fait l'objet de la partie suivante.

3. La vérification

La vérification est "*la confirmation par examen et apport de preuves tangibles que les exigences spécifiées ont été satisfaites*" (ISO8402, 1994). Dans le cadre qui nous intéresse, la vérification est employée pour s'assurer qu' « un modèle est bien construit ». Le but dans ces travaux est donc de prouver la cohérence du modèle en vérifiant que des exigences (des règles de modélisation, d'origine règlementaire, des parties prenantes, induites ou encore d'interopérabilité) sont respectées lors de la construction du modèle (Chapurlat, 2007).

Dans ce travail de recherche, la vérification doit mettre à disposition des **preuves tangibles** de l'existence ou de l'absence de problèmes d'interopérabilité. Nous adoptons donc une définition de la vérification un peu plus large, en ce sens qu'elle cherche à s'assurer des critères de qualité d'un modèle tel que (ElMansouri, 2009) :

- **La cohérence.** Un modèle ne contient pas d'informations contradictoires.
- **La pertinence.** Un modèle représente le système avec une qualité suffisante et répond aux objectifs du modeleur.
- **La complétude.** Un modèle contient toutes les informations nécessaires qui peuvent être utilisées pour démontrer la véracité d'une affirmation.

Enfin, la vérification nécessite (1) la construction et la mise à disposition d'un ou de plusieurs modèles d'un système, ici, il s'agit d'un modèle de processus collaboratif, (2) la formulation des exigences à vérifier qui sont élaborées à partir des besoins, ici, nous nous intéressons seulement aux besoins en interopérabilité, et enfin (3) l'utilisation d'une technique de vérification formelle pour s'assurer que ces exigences sont bien respectés par le modèle. Nous nous attachons, dans cette partie de l'état de l'art, à présenter quelques travaux paraissant essentiels sur chacun de ces trois points.

3.1. La modélisation de processus collaboratif

L'entreprise est un système complexe qui suscite depuis longtemps de nombreux travaux de recherche. Par exemple, le domaine de la modélisation d'entreprise a pour objet « *la construction de modèles d'une partie déterminée d'une entreprise pour en expliquer la structure, le fonctionnement et en analyser le comportement* » (Vernadat, 1999). Les travaux dans ce domaine ont permis, entre autres, de voir émerger des cadres de modélisation (ou architectures de référence) et des méthodes opérationnelles (Braesch *et al.*, 1995) (Vernadat, 1999) (Cattan *et al.*, 2001) sur lesquels des méthodologies ont été proposées.

Un cadre de modélisation est « *une structure de modèles partiels reposant sur des dimensions, au sein duquel on chemine afin de comprendre le tout* » (Pingaud, 2005). Nous pouvons citer les cadres de modélisation CIMOSA (AMICE, 1993), GERAM (Bernus *et al.*, 1996) et PERA (Williams *et al.*, 1994).

Une méthodologie est définie par (Pingaud, 2005) comme « *un ensemble de méthodes qui contribuent à résoudre un problème en les utilisant selon des règles établies* ». Les

méthodologies rassemblent des méthodes opérationnelles telles que MERISE (Tardieu *et al.*, 1983), GRAI (Doumeingts, 1984), ARIS (Scheer, 1994) et OLYMPIOS (Braesch *et al.*, 1995) en se basant sur des cadre de modélisation d'entreprise. Parmi les méthodologies existantes dans la littérature, nous pouvons par exemple citer GIM (Doumeingts, 1993).

La modélisation d'une entreprise nécessite plusieurs points de vue afin de mettre en évidence *les aspects de l'entreprise à prendre en compte lors de la modélisation* (Vernadat, 2002). Ces travaux se sont donc évidemment largement inspirés des vues classiques proposées par la systémique pour représenter tout ou partie d'une entreprise (Le Moigne, 1977) : **fonctionnelle**, **comportementale** et **structurelle** (Vallespir, 2003). Ils mettent ainsi en avant les vues **fonctionnelle**, **informationnelle**, **ressources** et **organisationnelle** de l'entreprise (Vernadat, 1996). Nous nous intéressons dans la suite à la vue fonctionnelle (Figure 9) et donc à la description du fonctionnement d'une entreprise au moyen de la modélisation de processus. D'après (Vernadat, 1999), cette vue de modélisation est incontournable et entretient des liens forts avec les autres vues comme le montre la Figure 9. La suite présente rapidement un panorama des langages de modélisation susceptibles d'être utilisés pour élaborer un modèle dans cette vue.

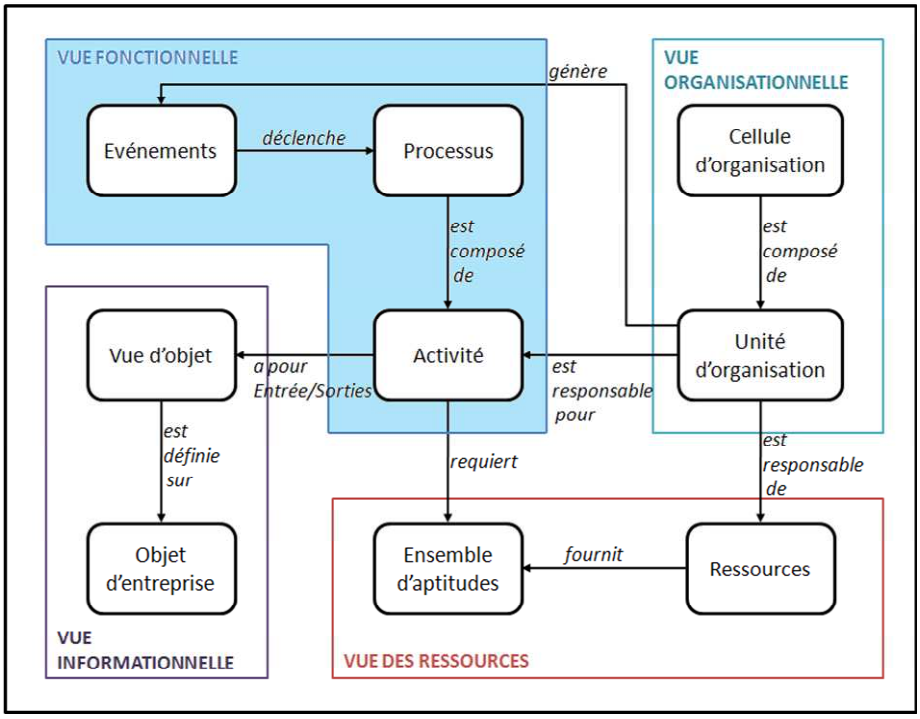


Figure 9. Modélisation d'entreprise

Le premier langage de modélisation de processus fut sans doute les **organigrammes** qui offrent une représentation graphique d'une séquence logique d'activités. Nous pouvons également citer la méthode **SADT** (*Structured Analysis and Design Techniques*) (Ross, 1977) qui a été initiée pour la modélisation de systèmes logiciels, a évoluée puis a été standardisée sous le nom d'**IDEF0** (Marca *et al.*, 1988). Pour représenter la synchronisation des activités ainsi que la dynamique de flux, les **Réseaux de Petri** sont aussi utilisés (Reisig, 1985). Le langage **eFFBD** peut être utilisé pour la description des processus client, support et de pilotage des entreprises (Long, 2002) (Aloui, 2007).

Il existe également d'autres langages plus évolués qui peuvent inclure une représentation informationnelle et/ou organisationnelle comme le langage **BPMN** (Business Process Modeling Notation) (BPMN, 2009). Ce langage couvre essentiellement la vue fonctionnelle mais permet aussi de décrire une partie de la vue organisationnelle, de la vue informationnelle et de la vue ressources (Touzi, 2007). C'est un langage semi-formel développé par l'OMG[6] afin de proposer une spécification pour définir et représenter des processus métier. Dans ce contexte, ce formalisme est utilisé dans le but d'offrir une notation explicite, facile d'emploi et accessible à tous les utilisateurs métiers. Le langage BPMN n'est pas destiné à la simple modélisation organisationnelle ou structurelle mais propose des structures adaptées à la modélisation des processus métiers (BPMN, 2009). Nous pouvons également citer la méthode ARIS qui propose, pour la description de la vue fonctionnelle un diagramme des enchainements de processus **PCD** (*Process Chain Diagram*) qui offre une vue générale de la composition structurelle des processus et de leurs relations avec les autres vues de l'entreprise (Scheer, 1994). Enfin, certains **diagrammes d'UML** représentent, d'une part, la structure du système et d'autre part, le comportement du système. Dans le cas de la modélisation de processus, les diagrammes UML sollicités sont le diagramme d'activités, le diagramme de séquence et le diagramme d'états-transitions (Grangel Seguer *et al.*, 2006).

Les différents langages énumérés précédemment, sont décrits et étudiés de manière plus précise dans (Petit, 2002). En définitive, ces langages sont des langages semi-formels possédant une représentation graphique rendant plus aisée leur manipulation. Cependant, ces langages restent pauvres, voire inadaptés, en ce qui concerne la représentation de l'interopérabilité.

[6] Open Management Group

3.2. Des besoins aux exigences

Une exigence est définie comme « *un énoncé qui prescrit une fonction, une aptitude ou une caractéristique à la quelle doit satisfaire un produit ou un système dans un contexte donné »* (Scucanec *et al.,* 2008). C'est un concept fondamental pour guider toute activité de conception et de développement *i.e.* d'ingénierie de systèmes. Dans ce domaine, il existe deux catégories d'exigences (INCOSE, 2007) à savoir les exigences fonctionnelles et les exigences non fonctionnelles. Les exigences fonctionnelles précisent ce que doit faire le système. Les exigences non fonctionnelles précisent au final comment le système « le fait » en termes de performance, sécurité, fiabilité, ergonomie, … et de contraintes.

La conception d'un système nécessite donc de préciser quelles sont les exigences auxquelles ce système doit répondre puis de s'assurer à chaque étape du projet de conception, que chaque exigence a bien été prise en compte et que le résultat de chaque étape la respecte. Différents problèmes peuvent être rencontrés au cours de ces activités (volatilité[7], modification, expression des exigences, *etc.*). Pour cela, l'ingénierie des exigences permet de gérer les exigences d'un système durant tout le cycle de vie de ce système. Les objectifs de l'ingénierie d'exigences sont les suivants :

1. **L'identification des parties prenantes.** Qui fait quoi ? Identifier les parties prenantes c'est-à-dire l'ensemble des personnes ou des groupes qui sont mises en relation avec le produit ou le système à traiter au cours de son cycle de vie.
2. **L'élicitation des besoins.** Que doit faire le système ? Trouver et formuler les besoins exprimés par les parties prenantes. Des méthodes telles que le brainstorming (séance de réflexion) et l'interview sont souvent employées.
3. **La description des exigences.** Traduire chaque besoin en une série d'exigences puis étudier l'ensemble des exigences pour les analyser, les compléter, les arranger, les tracer et éliminer les possibles ambiguïtés, redondances, non dits, blancs ou omissions classiques entre les exigences. Le résultat est un document spécifiant clairement toutes les exigences souvent décrites en langage naturel.

[7] La volatilité d'exigences est la mesure de l'instabilité des exigences. Une volatilité des exigences entraîne toujours le changement des exigences. (Eljamal, 2006)

4. **La vérification et la validation[8] des exigences.** Il s'agit alors de vérifier que les exigences sont bien écrites et possèdent les qualités nécessaires d'une exigence à savoir la non ambiguïté, la lisibilité, la faisabilité, … puis de valider la liste des exigences par rapport aux besoins des parties prenantes. Ceci permet d'obtenir un ensemble d'exigences cohérent et complet appelé *« spécification »* notamment lorsqu'il est validé et agréé par les parties prenantes (IEEE, 1999).
5. **La modification des exigences.** Il s'agit enfin de savoir gérer et tracer les demandes de modification d'exigences.

Nous pouvons citer deux méthodes d'ingénierie d'exigences : la méthode KAOS et la méthode REGAL. La méthode KAOS (*Keep All Objectives Satisfied*) (Van Lamsweerde *et al.*, 1991) part du principe qu'une exigence peut s'exprimer sous forme d'objectifs à atteindre. Ces objectifs peuvent ensuite être raffinés en sous objectifs jusqu'à obtention d'exigences prouvables et vérifiables sur un modèle. En complément, la démarche REGAL[9] (*Requirement Engineering Guide for All*) liste et structure un certain nombre de bonnes pratiques en ingénierie des exigences.

Après l'établissement des exigences par les parties prenantes, le système doit respecter ces exigences. Nous nous intéressons donc, dans la prochaine section aux méthodes de vérification de ces exigences par le système i.e. par un modèle du processus collaboratif.

3.3. Les méthodes pour la vérification

Pour réaliser une vérification, il existe plusieurs techniques. Certaines sont considérées comme informelles ou seulement structurées comme l'**expertise**, la **simulation** (Zaigler *et al.*, 2000), le **test** ou la **démonstration par l'application** (DoD, 2003). D'autres sont considérées comme des techniques formelles comme le ***model-checking*** (Edmund *et al.*, 1999)(Bérard *et al*, 2001) ou le ***theorem-proving*** (Duffy, 1991)(Gallier, 1986).

[8] La Validation cherche à répondre à la question « Construisons-nous le bon modèle ? ». Elle se définit comme la "confirmation par examen et apport de preuves tangibles que les exigences particulières pour un usage spécifique prévu sont satisfaites. Plusieurs validations peuvent être effectuées s'il y a différents usages prévus" (ISO8402, 1994).

[9] REGAL : Requirements Engineering Guide for All, http://www.incose.org/REGAL (accessible à la date du 11 janvier 2012)

3.3.1. Les méthodes informelles pour la vérification

L'expertise est menée par un expert d'un domaine ou d'un métier. La simulation se fait sur un modèle théorique dont le comportement est considéré comme similaire au comportement du système cible. Elle peut être utilisée pour aider à la conception du système ou en cours d'exploitation du système auquel cas, elle est une aide au pilotage à plus ou moins court terme. La simulation est maintenant une technique bien connue, outillée et utilisée dans l'entreprise. Toutefois, du fait du niveau de détail du modèle de simulation, de sa pertinence réelle ou d'autres facteurs liés à la qualité de ce modèle, la simulation n'est pas en mesure de décrire tous les scénarios possibles d'évolution du système, de prendre en compte toutes ses entrées ou de rendre compte de l'évolution de toutes ses sorties. Cette technique exige enfin des compétences humaines pour établir le modèle de simulation, pour analyser les résultats et formuler la démonstration. Enfin, les tests et les essais sont effectués grâce à des simulations ou directement sur un système existant, sur un prototype ou une maquette de ce système. Cette technique permet par exemple d'évaluer la conformité ou un niveau de performance.

3.3.2. Les méthodes formelles pour la vérification

Les techniques de vérification formelle sont basées sur l'exploitation de modèles formels *i.e.* obtenus avec un langage de modélisation utilisant une sémantique formelle basée sur un modèle mathématique sous-jacent. Dans ce cas, il est possible de fournir une preuve formelle du respect (ou non) d'une exigence (formalisée avec un langage formel sous forme de propriété[10]), indépendamment de toute interprétation humaine ce qui représente l'intérêt essentiel de l'usage de ce type de technique. On distinguera ici deux techniques de preuve de propriétés sur des modèles formels : le *theorem-proving* et le *model-checking*.

Le *theorem-proving* - ou preuve automatique de théorème - consiste à prouver directement ou à assister une personne pour prouver sur un modèle (un modèle du processus décrit *via* un langage formel est donc nécessaire) un théorème (décrivant alors l'exigence) au moyen d'axiomes et de règles d'inférences (Duffy, 1991). Nous pouvons citer des outils appelés alors *theorem prover* comme PVS (Owre *et al.*, 1996), Coq (Huet *et al.*, 1995), HOL (Melham *et al.*, 1993) et Isabelle (Paulson, 1994) qui permettent de mettre en œuvre cette technique.

[10] Une propriété est « *une connaissance que l'on a du système ou d'un modèle qui traduit une exigence, une finalité ou une caractéristique à satisfaire* » (Lamine, 2001)

Le *model-checking* (Edmund *et al.*, 1999) (Bérard *et al.*, 2001) (Alur *et al.*, 1993) est une technique de vérification formelle de propriétés sur des modèles comportementaux. Le principe est d'explorer exhaustivement tous les états atteignables du système calculés à partir du modèle comportemental de ce système (un modèle comportemental du processus collaboratif est donc nécessaire) et de vérifier la véracité de la propriétés dans certains de ces états. Dans le cas où une propriété s'avère non vérifiée, un contre exemple est proposé par exemple sous la forme d'une séquence d'état ou chemin permettant d'arriver à l'état du système dans lequel la propriété n'est pas respectée. Nous pouvons citer des outils appelés alors *model checker* comme HyTech (Henzinger *et al.*, 1995), SMV (McMillan, 1992), Kronos (Yovine, 1997)(Daws *et al.*, 1996) ou encore UPPAAL (Behrmman *et al.*, 2004) qui permettent de mettre en œuvre cette technique.

Le *theorem-proving*, à l'opposé du *model checking*, offre l'avantage d'être indépendant de la taille de l'espace des états atteignables et peut pour cette raison s'appliquer sur des modèles avec un très grand nombre d'états, ou même, selon les stratégies de preuve employées, sur des modèles dont le nombre d'états n'est pas déterminé *a priori*. En effet, le *model checking* est confronté au phénomène bien connu **d'explosion combinatoire** qui restreint la vérification à des systèmes de petite taille. Par contre, le *model-checking* permet une vérification du comportement et peut ainsi prendre en compte la dynamique d'évolution du processus *i.e.* la dimension temporelle qui n'est pas facile à prendre en compte avec le *theorem proving*.

Dans tous les cas, le *model-checking* comme le *theorem proving* reposent sur trois phases.

- **La modélisation du système.** Cette première phase fournit une représentation formelle du système à étudier. Dans le cas du *theorem proving*, le modèle est statique *i.e.* ne permettant pas de prendre en compte le temps. Il s'agit alors d'équation ou de graphe dont le parcours permet de décider du respect d'un théorème. Dans le cas du *model checking* cette formalisation est basée sur le principe de systèmes de transitions où les nœuds représentent les états du système et les transitions représentent l'évolution du système d'un état à l'autre (*e.g.* structures de Kripke, automates temporisés, Réseaux de Petri…).

- **La spécification de la propriété.** Cette seconde phase permet de fournir une formalisation de l'exigence à vérifier sous forme de théorème ou de propriété. Dans le cas du *theorem proving*, le théorème est exprimé dans le même langage que celui utilisé pour formaliser le modèle du système. Dans le cas du *model checking* cette formalisation se fait par le biais de formules écrites souvent à partir de logiques temporelles (*e.g.* LTL, CTL, TCTL…).
- **La vérification.** Cette dernière phase consiste à appliquer une stratégie de parcours et un algorithme de vérification qui prend en entrée le modèle du système et le théorème ou la propriété à vérifier.

La suite présente comment ces phases sont mises en œuvre dans le cas de certaines techniques qui nous intéressent particulièrement dans ces travaux.

3.3.3. Les Graphes Conceptuels pour la vérification

L'utilisation des Graphes Conceptuels pour la vérification est une thématique largement étudiée et proposée dans différents travaux de recherche (Chapurlat *et al.,* 2003) (Kamsu-Foguem, 2004) (Kamsu-Foguem *et al.,* 2006) (Aloui, 2007).

Les Graphes Conceptuels ont été initiés par (Sowa, 1984) pour représenter la sémantique contenue dans une phrase, et non sa syntaxe, dans une forme qui soit précise, lisible et utilisable par un ordinateur. D'une manière formelle, un graphe conceptuel est un graphe composé de deux sortes de nœuds, les concepts et les relations conceptuelles comme illustré dans la Figure 10.

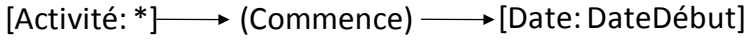


Figure 10. Exemple de graphe conceptuel

Ce graphe conceptuel peut être lu comme « *toute* (*) *activité* (concept) *commence (relation) à une date* (concept nommé ici *DateDébut* qui est un référent ou encore marqueur ou représentant) ». Les nœuds concepts représentent des entités, des attributs, des états, des événements… Les nœuds relations conceptuelles symbolisent les liens existants entre deux concepts. Un graphe conceptuel est orienté, fini (*i.e.* nombre fini de nœuds), connexe (*i.e.* si deux parties ne sont pas connectées entre elles, on obtient deux graphes conceptuels) et bipartites (*i.e.* deux sortes de nœuds : concepts et relations) (Chein *et al.,* 1992).

Dans un graphe conceptuel, un concept est formé par deux éléments principaux, le type du concept et le référent du concept selon la notation : [Type : Référent]. Le type du concept est une abstraction de l'ensemble des référents du concept. Sur l'exemple précédent, « Activité » est un type qui représente la classe de toutes les activités. Le référent peut indiquer un concept générique ou individuel ("*" signifie concept générique sur la figure ci-dessus).

Les concepts et les relations sont décrits dans des structures hiérarchiques appelé les **Treillis de concepts** et les **Treillis de relations.** Le rôle de ces treillis (ou hiérarchies) est de permettre de généraliser ou au contraire de spécialiser les concepts ou les relations. Le treillis de concept possède un concept général ou universel (noté T) qui recouvre tous les objets ainsi qu'un concept absurde qui n'en recouvre aucun (noté ⊥). Dans le treillis des relations, il existe une relation binaire universelle. Chaque relation est caractérisée par un concept source et un concept cible (Espinasse, 2008), (Chawk, 2000).

Les graphes conceptuels possèdent un fondement mathématique abouti qui permet d'effectuer de la vérification à l'aide d'opérations sur les graphes. En effet, des mécanismes formels de projection permettent de s'assurer de la véracité ou plus simplement de la simple présence (ou non) d'une connaissance donnée dans un graphe. Ce mécanisme est appliqué pour vérifier des propriétés sous forme de contraintes. Il existe des outils comme COGUI[11] et COGITANT (COGITANT, 2009) pour permettre de manipuler des graphes conceptuels et d'appliquer les mécanismes de projection.

Le but de la projection est de rechercher une information, un motif dans le modèle du graphe et de vérifier ainsi la présence (ou non) dans ce graphe d'un sous graphe représentant le modèle de la propriété. En effet, la projection consiste à localiser dans un graphe conceptuel un sous-graphe donné ; ainsi elle permet le filtrage sur un ensemble de graphes conceptuels. En termes de vérification, il s'agit donc de comparer un modèle de graphes conceptuel représentant le système avec un modèle représentant la propriété. Si la projection échoue, alors la propriété modélisée ne peut être vérifiée et les causes sont mises en évidence.

Le mécanisme de projection avait comme but premier de prouver la véracité ou non de propriétés dites positives (projection du graphe contrainte sur le graphe du modèle du système). Cependant, il est parfois nécessaire de démontrer la non existence de certaines

[11] COGUI Version 1.2. Tutoriel disponible sur : http://www.lirmm.fr/cogui/tutorial.php (accessible à la date du 11 janvier 2012)

propriétés. Dans cette perspective, les travaux de (Baget, 2001) ont étendu le modèle de base des graphes conceptuels en définissant des contraintes qui permettent la prise en compte d'une forme de négation. De ce fait, une propriété à vérifier peut être représentée par une contrainte positive ou négative. Une contrainte positive est décrite avec une cause et une conclusion et sa projection est réalisée en fonction de l'interprétation suivante : «*Si la cause est vraie, alors la conclusion doit aussi être vraie*". Une contrainte négative est un graphe conceptuel unique et sa projection est interprétée comme : « *Si une contrainte négative n'est pas projeté sur le modèle, la propriété est vérifiée* ».

Les graphes conceptuels offrent donc la possibilité d'effectuer la vérification de contraintes positives et négatives sur des modèles statiques ne prenant pas en compte l'aspect temporel. Leur utilisation pour la vérification, largement étudiée et présentée, n'est plus remise en cause.

3.3.4. Synthèse

L'étude des différentes méthodes de vérification nous montre que certaines d'entre elles sont informelles (expertise, test et simulation), d'autres formelles (*theorem proving* et *model checking*) ou encore basées sur les graphes conceptuels. Ces différentes méthodes ont chacune des inconvénients et des avantages comme présenté en Tableau 3.

Les méthodes informelles comme le test et l'expertise ne nécessitent pas forcément de recourir à un modèle du système à vérifier mais peuvent s'utiliser directement sur le vrai système. Ces techniques sont sollicitées lorsqu'on ne dispose pas d'un modèle du système à analyser ou encore de la description de l'exigence à vérifier sur le modèle, ce qui leur donne un avantage considérable. Cependant, ces méthodes exigent des compétences humaines pour leur utilisation et peuvent toujours se confronter à l'erreur humaine.

Les méthodes formelles telles que le *theorem proving* et le *model checking* sont basées sur l'utilisation de modèles et procurent une preuve formelle du respect ou non respect d'une exigence. Ces méthodes utilisent des mécanismes mathématiques, permettent de disposer d'un modèle représentant le comportement du système et offrent une vision dynamique du système. Cependant, ces méthodes peuvent paraître « lourdes » à mettre en œuvre (e.g. besoin de formalisation accru) et sont peu ou pas utilisées, dans certains domaines comme la modélisation d'entreprise (difficulté de formalisation des modèles).

La vérification par les graphes conceptuels repose sur des fondements mathématiques aboutis tels que les mécanismes formels de projection permettant de s'assurer de la véracité ou plus simplement de la présence d'une connaissance donnée dans un graphe. Cependant, les graphes conceptuels ne permettent pas de représenter le comportement d'un système, ils n'offrent donc qu'une vision statique du système.

Eu égard aux avantages et inconvénients de chaque méthode de vérification, nous pouvons constater qu'elles peuvent être utilisées de façon complémentaire offrant une modélisation complète du système et donc une vérification complète.

Tableau 3. Etude comparative des méthodes de vérification

	Modèles	Preuve formelle	Mécanismes mathématiques	Explosion combinatoire
L'expertise	+	-	-	-
La simulation	+++	++	-	-
Le test	-	++	-	-
Le theorem proving	+++	+++	++	-
Le model checking	+++	+++	+++	+++
Les graphes conceptuels	+++	+++	+++	-

4. Conclusion

Dans ce chapitre, nous avons présenté un état de l'art couvrant, d'un côté, des travaux sur l'interopérabilité et, de l'autre côté, des travaux sur la vérification.

Nous avons vu qu'il existe des cadres d'interopérabilité pour structurer ce concept. Cependant, ces différents cadres ne permettent pas l'évaluation de l'interopérabilité. Pour cela nous avons présenté des modèles de maturité qui permettent de répondre à cette problématique sans pour autant en donner une mesure quantitative. D'autres travaux ont vu le jour afin de remédier à cela. Ces différents travaux mettent l'accent sur la nécessité de formaliser ce concept d'interopérabilité ou du moins lui donner une vision mathématique. Néanmoins, cette vision mathématique et cette mesure d'interopérabilité n'informe pas de la présence locale de problème d'interopérabilité et ne proposent pas non plus une approche anticipative de l'interopérabilité. Ainsi, les travaux autour de l'interopérabilité permettent de

structurer et d'évaluer l'interopérabilité sans pour autant fournir une preuve formelle de la présence ou non de problèmes d'interopérabilité

Nous nous sommes donc intéressés, dans la deuxième partie de ce chapitre, à la vérification qui pourrait répondre à cette problématique de détection et d'anticipation de problèmes d'interopérabilité. Ainsi, nous avons vu qu'il existe plusieurs méthodes de vérification qui sont complémentaires.

Dans la suite de ce document, nous présentons l'utilisation de la vérification pour détecter des problèmes d'interopérabilité en suivant une démarche d'ingénierie d'exigences permettant de structurer des exigences d'interopérabilité et de les formaliser. Cette démarche nous permettra par la suite d'utiliser des mécanismes afin de raisonner pour prouver la véracité ou non d'une propriété. Plus précisément, cette démarche permet de :

- Formuler des exigences d'interopérabilité en langage naturel à partir de besoins récoltés.
- Modéliser le processus collaboratif avec un langage de modélisation dédié. Ce modèle sera réécrit avec des règles de transformation de modèles pour permettre la vérification.
- Vérifier les exigences d'interopérabilité à l'aide de techniques complémentaires afin de représenter l'aspect statique et dynamique d'un processus collaboratif et permettre la vérification de propriétés à la fois atemporelles et temporelles.

Chapitre 3

Exigences d’interopérabilité : formulation et classification

Table des matières du chapitre 3

1. Introduction

L'état de l'art a présenté et commenté plusieurs travaux de recherche sur la structuration (cadres d'interopérabilité), l'évaluation (modèles de maturité) et la mesure de l'interopérabilité. Cependant, nous avons démontré dans ce qui précède que ces moyens de permettent pas de détecter des problèmes d'interopérabilité de façon anticipative, besoin à l'origine de ce travail de recherche.

Dans ce chapitre, nous allons présenter et construire une approche qui se propose, d'une part, d'être **anticipative** et, d'autre part, de caractériser et de **détecter localement** puis **globalement** des problèmes d'interopérabilité.

2. Démarche scientifique

L'approche proposée est **guidée par des modèles** de processus collaboratifs. Elle cherche à localiser des problèmes d'interopérabilité et s'inspire pour cela de l'ingénierie des exigences. Précisément, l'ingénierie des exigences permet de décrire et de structurer des exigences d'interopérabilité dont le non respect est la cause potentielle de problèmes pouvant freiner voire rendre impossible la collaboration. Enfin, elle repose sur l'usage de techniques de formalisation puis de vérification de modèle par preuve de propriétés pour s'assurer que le modèle respecte bien ces exigences. Ainsi, la détection locale de problèmes redoutés est permise par l'utilisation de techniques de modélisation, de formalisation, de structuration et de vérification. Cette approche est mise en œuvre suivant les étapes présentées Figure 11.

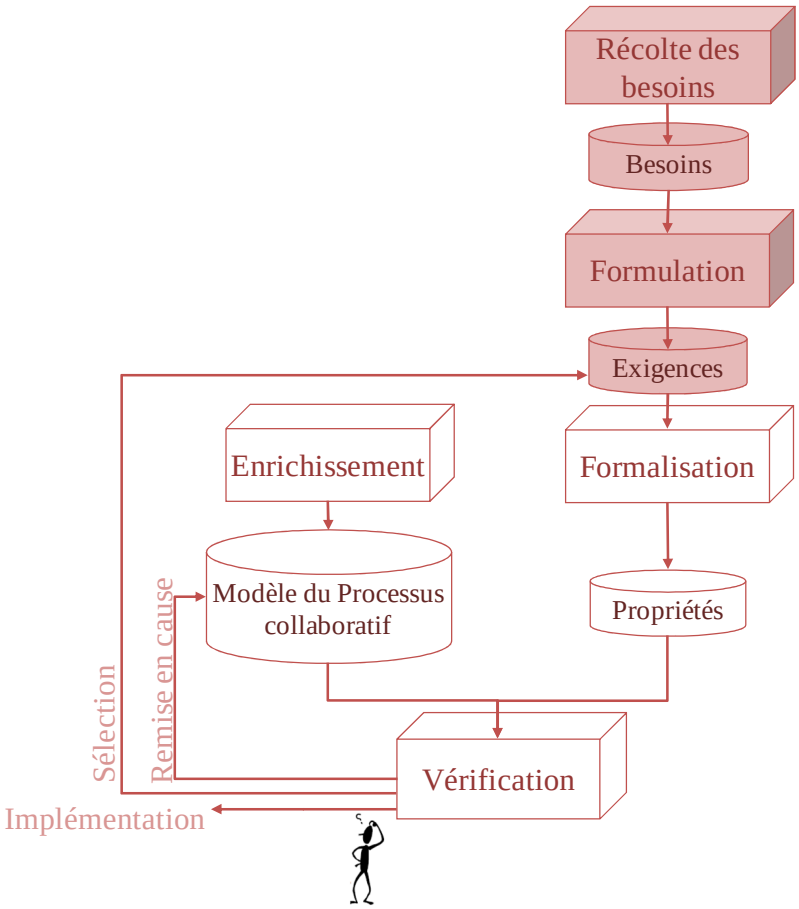


Figure 11. Approche anticipative de détection de problèmes d'interopérabilité dans des processus collaboratifs.

Dans un premier temps, les besoins en termes d'interopérabilité sont récoltés et exprimés en employant un langage proche des acteurs du processus collaboratif. L'objectif d'une telle étape est de développer un **référentiel des besoins d'interopérabilité** aussi exhaustif et générique que possible. D'autres besoins, plus spécifiques au processus collaboratif concerné, peuvent être exprimés par les utilisateurs au cours de l'utilisation de l'approche. Dans ce cas, ces nouveaux besoins viennent compléter le référentiel des besoins existant.

Ces besoins sont alors clarifiés, structurés et formalisés sous forme d'exigences dites d'interopérabilité décrites plus loin. Il s'agit ici d'obtenir un **référentiel d'exigences d'interopérabilité** permettant alors de guider le processus d'analyse des problèmes d'interopérabilité et ce en employant un langage plus formel ou en tout cas plus structuré. Le développement d'un tel référentiel doit permettre à l'utilisateur en charge de ce processus d'analyse d'avoir une vision claire de l'ensemble des exigences d'interopérabilité à disposition. Cet utilisateur peut ensuite sélectionner aisément celle(s) qu'il souhaite vérifier en

fonction de son objectif d'analyse. Il peut s'agir, par exemple, de s'intéresser à la performance ou seulement à l'interopérabilité des outils employés au cours du processus collaboratif. Une fois ces exigences sélectionnées, elles sont alors formalisées en **propriétés** *via* l'utilisation d'un langage formel pour permettre leur vérification effective sur le modèle de processus collaboratif par des **techniques de vérification** formelle. Dans cette optique, le modèle, doit, d'abord, pouvoir prendre en considération les exigences d'interopérabilité. Dans ces travaux, la modélisation est réalisée avec le langage BPMN auquel certains **enrichissements conceptuels** ont permis d'intégrer les éléments nécessaires (classes, attributs et relations) qui permettent la vérification des exigences d'interopérabilité. Il doit ensuite pouvoir être traduit dans un formalisme supportant la technique de vérification proposée. Pour cela, des mécanismes de **transformation de modèle** sont nécessaires.

Cependant, nous devons garder à l'esprit que la mise en œuvre de techniques de vérification dépend de deux critères, à savoir, la qualité du modèle et la qualité de la propriété à vérifier. En ce sens, la qualité d'un modèle peut être bonne ou mauvaise, de même que pour la propriété. De ces qualités, dépendront, d'une part, l'interprétation de la vérification et d'autre part, les exigences qui peuvent être mises en œuvre. Le tableau suivant présente les différentes combinaisons de qualité (modèle/propriété) et leur interprétation. Dans nos travaux, nous essayons de nous positionner dans le cas idéal, à savoir, le modèle est « bon » et la propriété est « bien écrite ».

Tableau 4. Interprétation du résultat de vérification[12]

Modèle	Propriété	Interprétation
Bon	Bonne	•Résultat de la vérification correct • Prise en compte d'exigence d'interopérabilité
Bon	Mauvaise	•Résultat de la vérification incorrecte •Remise en cause de l'exigence d'interopérabilité
Mauvais	Bonne	•Résultat de la vérification incorrect •Prise en compte d'exigence modèle
Mauvais	Mauvaise	•Résultat de la vérification incorrect •Remise en cause du modèle •Remise en cause de l'exigence d'interopérabilité

Ainsi, si les propriétés sont effectivement respectées par le modèle du processus

[12] Bon modèle : complet, pertinent, et cohérent.
Bonne propriété : complète, pertinente, formelle et vérifiable.

collaboratif, alors ce dernier peut être implémenté. Dans le cas où des problèmes sont constatés *i.e.* il existe une ou plusieurs propriétés non respectées, alors (1) le modèle du processus collaboratif sera remis en cause afin d'apporter les solutions adéquates aux problèmes détectés, et (2) l'utilisateur aura le choix de sélectionner d'autres exigences à respecter *i.e.* parfaire la modélisation du processus et/ou de l'exigence avant de refaire une vérification.

Nous nous attachons, dans ce chapitre, à la présentation des étapes grisées sur la Figure 11 ainsi que les concepts qui leurs sont associés.

3. Les besoins d'interopérabilité

Ce n'est bien souvent qu'en situation de collaboration effective que les entreprises constatent et doivent faire face à des problèmes dont l'origine est liée à une lacune en termes d'interopérabilité. En effet, ces entreprises n'anticipent pas ou très peu sur les besoins effectifs d'interopérabilité qu'une collaboration peut entrainer. Il existe pourtant des solutions d'interopérabilité qui permettent soit de résoudre un problème d'interopérabilité identifié à un instant donné, soit de répondre à un besoin exprimé et ce de manière anticipative. Ainsi, sans une prise en compte explicite de l'ensemble des besoins d'interopérabilité, l'utilisation d'approche de type « problèmes identifiés – solutions » peut être préjudiciable au partenariat (*e.g.* émergence de besoins apparemment nouveaux et perte de temps pour trouver la solution adéquate). En conséquence, la première partie de notre approche se focalise sur la collecte des besoins d'interopérabilité.

Comme dit plus haut, cette collecte consiste d'abord à exprimer des besoins en analysant des travaux sur l'interopérabilité. Elle consiste ensuite à aider des parties prenantes susceptibles d'être impliquées dans des processus collaboratifs à exprimer et faire remonter leurs propres besoins. Pour cela, un questionnaire à destination d'industriels est utilisé.

3.1. Extraction des besoins par analyse de l'existant

L'étude de l'état de l'art présenté dans le chapitre 2, couvrant les différents travaux autour de l'interopérabilité, a permis de dégager dans un premier temps des besoins d'interopérabilité. Par exemple, les niveaux proposés dans les modèles de maturité pour atteindre la pleine interopérabilité permettent d'obtenir un premier référentiel des besoins. Ces

modèles de maturité couvrent les trois aspects de l'interopérabilité présentés précédemment : conceptuel, organisationnel et technologique. De ce fait, les besoins récoltés de cette étude bibliographique englobent également ces trois aspects. Citons par exemple :

- **Besoin 1 :** « *Les données sont documentées en utilisant un protocole commun et sont accessibles via des interfaces* » (LCIM – niveau 1).
- **Besoin 2 :** « *Les organisations sont encore distinctes, des cadres sont mis en place pour soutenir l'interopérabilité et des objectifs communs sont définis, les rôles et les responsabilités sont également attribuées* » (OIM – niveau 2).
- **Besoin 3 :** « *Les systèmes en connexion sont identifiés par leur capacité à fournir une compréhension des données échangées. Les e-mails incluent des pièces jointes et contribuent à la réussite de l'échange* » (LISI – niveau 2)
- **Besoin 4 :** « *La gouvernance des processus d'approvisionnement relie explicitement les exigences opérationnelles à l'architecture technique à travers le financement de projets*». (IMM – niveau 4) (Nehta, 2007)

3.2. Extraction des besoins par enquête auprès des industriels

Afin de consolider la liste des besoins récoltés à partir de l'analyse de l'existant et ainsi d'enrichir le premier référentiel, une enquête a été réalisée auprès d'industriels sous forme de questionnaire (*cf.* Annexe A). L'objectif de cette enquête est d'offrir aux industriels la possibilité de décrire un certain nombre de problèmes auxquels ils sont confrontés puis d'exprimer dans leur langage quels sont alors les besoins auxquels il semble nécessaire d'accorder une attention particulière

Ce questionnaire se focalise sur plusieurs aspects. Le premier aspect met en avant des besoins de collaboration intra et inter entreprises : cela permet de différencier, si il y a lieu, les besoins propres à un processus collaboratif privé et ceux propres à un processus collaboratif public. Le deuxième aspect se focalise sur les problèmes d'interopérabilité (conceptuelle, organisationnelle et technologique). Enfin, l'aspect performance est également considéré pour permettre aux industriels de faire un bilan de leurs expériences passées. Les renseignements sur cet aspect sont exprimés à l'aide des critères de performance classiques et aisément manipulables : coût, temps et qualité.

Les besoins présentés ci-après sont extraits des réponses fournies par les industriels au questionnaire :

- **Besoin 5 :** « *Il faut pouvoir envoyer et recevoir des données exploitables, et être sûr de leur réception par l'autre partenaire pendant le déroulement de l'activité* ».
- **Besoin 6 :** « *Homogénéiser la communication quelle qu'en soit la forme, donc disposer d'une sémantique comprise et partagée entre les deux partenaires* ».
- **Besoin 7 :** « *Rester aussi performant en terme de coût, de qualité (produit et service) et de délais après le partenariat que ce que l'on était performant avant même que nous n'y soyons impliqués* ».
- **Besoin 8 :** « *Une ressource doit être disponible au moment où l'on fait appel à elle* ».

Comme nous pouvons le constater – aussi bien pour les besoins issus de l'analyse de l'existant que pour les besoins issus de l'enquête - ces besoins sont exprimés en langage naturel et peuvent être difficiles à manipuler de par leur nature et la complexité de leur expression. Dès lors, nous faisons face à un ensemble de difficultés liées aux besoins extraits, à savoir :

- **Le blanc.** Un besoin est purement et simplement omis.
- **La redite.** Un besoin est exprimé plusieurs fois mais en des termes différents.
- **L'ambigüité.** Un besoin peut avoir différentes interprétations possibles.
- **L'imprécision.** Un besoin parait flou et/ou insuffisamment précis. Une solution classique est soit de le décomposer en plusieurs besoins plus précis soit de le raffiner en améliorant le niveau de détail des informations utilisées pour le décrire.
- **L'incohérence.** Un besoin exprimé peut contredire un autre besoin.

Pour lever ces problèmes essentiellement liés à l'expressivité du langage naturel, les besoins sont formalisés et structurés sous forme d'exigences comme le propose (INCOSE, 2007) et (ISO15288, 2008) quand on aborde le problème de l'ingénierie des exigences.

4. Les exigences d'interopérabilité

Sur la base de la définition d'une exigence proposé dans (Scucanec *et al.*, 2008), nous définissons une exigence d'interopérabilité comme « *un énoncé qui prescrit une fonction, une aptitude ou une caractéristique liée à la capacité qu'a un partenaire à assurer son partenariat au niveau de sa compatibilité, son interopération, sa réversibilité et son autonomie, à laquelle il doit satisfaire* »

Une exigence d'interopérabilité traduit donc un besoin du référentiel des besoins décrit plus haut. Elle se doit d'être exprimée de manière claire (*i.e.* sous la forme « sujet – verbe – complément »), d'être identifiable, d'être traçable, d'être vérifiable, non ambigüe, non contradictoire avec une autre exigence du même référentiel, *etc*. Toutes les exigences sont de fait regroupées dans un **référentiel d'exigences d'interopérabilité**. Ce référentiel guide le travail de l'analyste qui va rechercher des problèmes d'interopérabilité (quel est le problème et où se situe-t-il ?) mais aussi, qui va devoir renseigner les acteurs du processus sur leur interopérabilité propre. Ce référentiel permet de structurer, classer et tracer les exigences d'interopérabilité traduisant chacun des besoins issus du référentiel des besoins. Sa formalisation permet ensuite d'adopter des techniques de vérification formelle pour s'assurer de la véracité de tout ou partie de ces exigences. Pour cela des règles de traduction et de propagation sont définies pour aboutir à un résultat de vérification exploitable par les partenaires.

Pour ce faire, un modèle d'exigences d'interopérabilité permettant (1) de structurer et de classer les exigences et, (2) de mettre en œuvre des mécanismes de propagation est proposé dans la suite de ce chapitre.

4.1. Classification des exigences d'interopérabilité

Le chapitre 1 a montré que l'interopérabilité est liée d'abord à la compatibilité. Il s'agit alors d'harmoniser les systèmes souhaitant collaborer (*e.g.* leur comportement, organisation, méthodes de travail, règles de communication et d'échange,…) afin que la mise en œuvre des interfaces ne nécessite pas d'effort particulier.

Dans un second temps, l'interopérabilité fait appel à la notion d'interaction (aussi appelée interopération). En effet, pendant une collaboration, des problèmes d'autres natures

que la compatibilité peuvent apparaître à certains moments, de manière fugace ou plus ou moins longue avec des effets plus ou moins néfastes sur la qualité et l'efficience de la collaboration. Par exemple, certains moyens ou ressources peuvent devenir indisponibles ou s'avérer au final incapables de mener à bien leur mission en fonction de leur état et des conditions du moment. Dans ce cas, certaines exigences doivent considérer toutes les interactions qui peuvent être nécessaires entre des partenaires durant toute la durée du processus collaboratif (*e.g.* ressources, contrôle, activité…).

De même, l'interopérabilité est liée à la préservation de l'autonomie des partenaires d'une collaboration. Il s'agit pour ce type d'exigence de s'assurer, d'une part, qu'un système donné est bien un acteur efficace du processus collaboratif en terme d'interopérabilité et, d'autre part, que ce même système - malgré la mise en jeu dans le processus collaboratif de certains de ses éléments - peut poursuivre sa propre logique de fonctionnement (en termes de gouvernance et de rendu opérationnel).

Enfin, l'interopérabilité est également liée à la notion de réversibilité. Dans ce cas, chacun des partenaires doit être en mesure de revenir à son état initial au terme du partenariat et ce, malgré les modifications ou adaptations qui ont pu être engendrées par la mise en œuvre de solutions d'interopérabilité.

L'étude de ces quatre notions primordiales de l'interopérabilité nous conduit naturellement à définir quatre classes d'exigences d'interopérabilité, à savoir : les **exigences de compatibilité**, les **exigences d'interopération**, les **exigences d'autonomie** et les **exigences de réversibilité**.

Afin d'avoir une vision plus concrète de la classification des exigences d'interopérabilité que nous proposons et développons par la suite, il est possible de positionner celle-ci dans le cycle de vie de l'Entreprise Virtuelle où l'interopérabilité revêt un enjeu important (*e.g.* prise de marché sur opportunité, relation temporaire, cessation de l'EV après disparition ou satisfaction des conditions liées à cette opportunité, présence d'objectifs communs/propres) (Camarinha-Matos *et al.*, 2003). Ainsi, des partenaires qui ont pour but de travailler ensemble doivent, dans un premier temps, créer la collaboration (phase de création), c'est-à-dire s'assurer ou réaliser les interfaces nécessaires au bon déroulement de la collaboration. Ensuite, pendant la phase d'opération, les partenaires doivent, d'une part, s'assurer que les interactions entre eux sont efficaces et, d'autre part, s'assurer que leur

autonomie (de gouvernance et opérationnelle) est effectivement préservée. Enfin, la phase de dissolution - après la collaboration – nécessite de s'assurer que l'arrêt du partenariat n'affecte pas l'intégrité (ou l'affecte dans des limites de tolérances définies), la stabilité et la performance d'un partenaire donné. Le positionnement de ces quatre classes d'exigences au sein du cycle de l'entreprise virtuelle est mis en évidence dans la Figure 12.

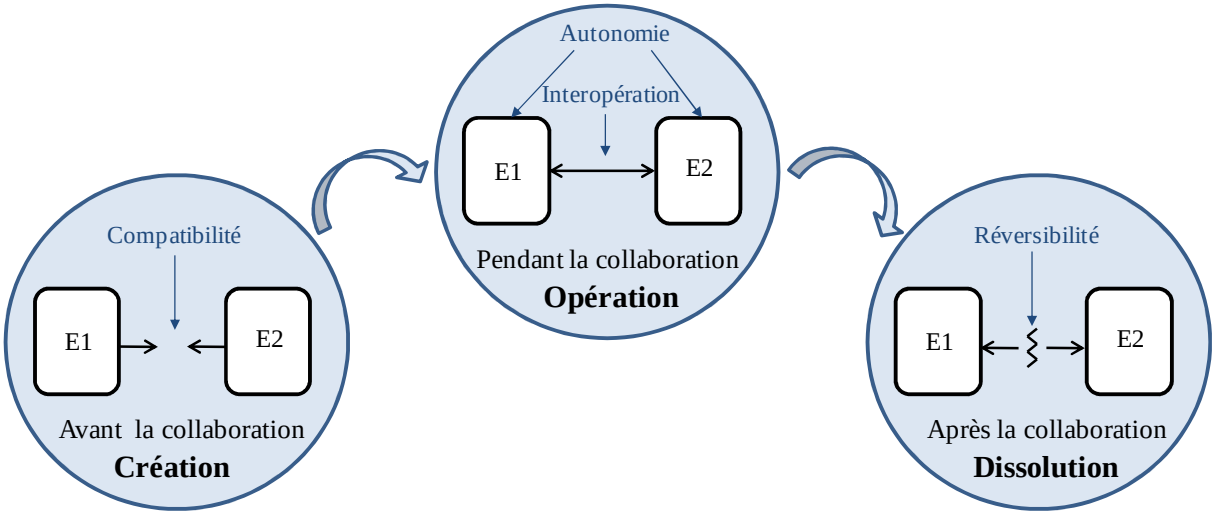


Figure 12. Les classes d'exigences d'interopérabilité liées au cycle de vie de l'entreprise virtuelle [adapté de (Camarinha-Matos *et al.*, 2003)]

Bien entendu, dans le cas de la mise en œuvre de l'approche anticipative, ces exigences seront vérifiées, par la suite, avant même que la collaboration ne devienne réellement effective.

Ainsi, pour bien structurer les exigences de chaque classe, nous nous basons (1) sur le principe de l'élaboration d'une relation de décomposition d'exigence, (2) sur la classification des exigences proposée précédemment et, (3) sur le cadre d'interopérabilité développé au sein du Réseau d'Excellence INTEROP (INTEROP, 2007). L'ensemble de ces considérations nous permet de proposer un **GRAphe de Décomposition d'Exigences d'Interopérabilité** (**GRADEI**) sur lequel sont positionnées les classes d'exigences définies et les primo-exigences qui les décomposent (*cf.* Figure 13). Le premier nœud de ce graphe représente l'exigence d'interopérabilité la plus abstraite (N_0) qui exprime le premier besoin des entreprises *i.e.* satisfaire à une exigence globale et abstraite d'interopérabilité. Cette première exigence est décomposée par les quatre classes d'exigences d'interopérabilité qui sont la compatibilité, l'interopération, la réversibilité et l'autonomie. Les exigences de chaque classe sont alors décomposées à leur tour en considérant les niveaux business, processus, services et

données[13]. Ensuite, afin de cibler et de préciser les défauts d'interopérabilité, nous considérons les trois catégories de problèmes d'interopérabilité (conceptuels, organisationnels et techniques). Enfin, le principe de décomposition adopté permet de peupler cette classification avec des exigences extraites des travaux existants et des résultats de l'enquête effectuée auprès d'industriels (Mallek *et al.*, 2010a).

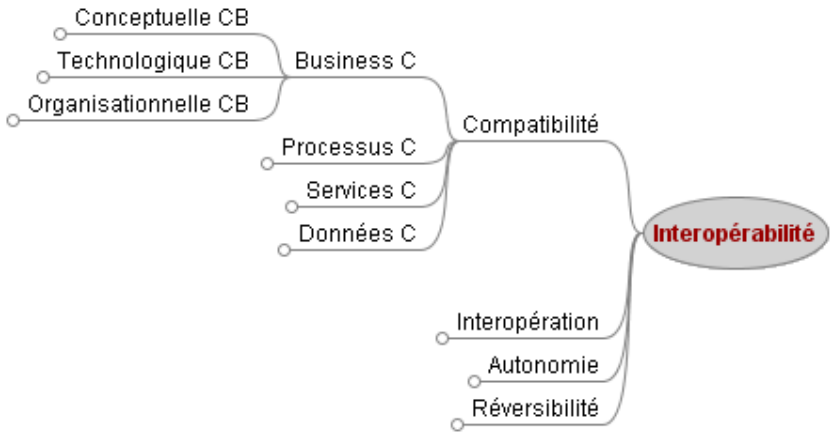


Figure 13. Les quatre classes structurées sur le modèle GRADEI

4.2. Principe du modèle GRADEI

Le modèle GRADEI permet de décomposer, manipuler et enfin, formaliser les exigences à des fins d'analyse sur un modèle d'un processus collaboratif public ou privé. Il repose sur une relation de décomposition d'exigence permettant d'associer et de combiner entres elles des exigences d'un niveau de détail (k+1) plus fin à une exigence de plus haut niveau (k). GRADEI est ainsi un graphe orienté schématisé Figure 14 (Mallek *et al.*, 2010b).

[13] Selon (INTEROP, 2007) :

- L'interopérabilité des **données** consiste à faire travailler ensemble différents modèles de données. Ce domaine doit permettre l'échange d'informations entre des bases de données différentes.
- L'interopérabilité des **services** concerne l'identification et la mise en relation d'applications différentes. Le terme « service » ne se limite pas aux applications informatiques mais peut aussi définir, par exemple les fonctions d'une entreprise.
- L'interopérabilité des **processus** vise à faire travailler ensemble divers processus : un processus définit une séquence de services. Il est nécessaire d'étudier comment connecter les processus internes des entreprises pour créer un processus commun.
- L'interopérabilité au niveau **business** a trait à l'organisation globale des entreprises. Il s'agit de faire travailler en « harmonie » différents modes de prise de décision, différentes méthodes de travail, différentes législations *etc.*

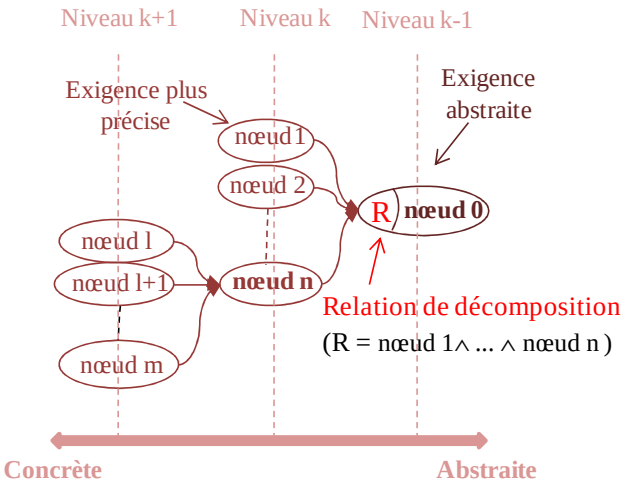


Figure 14. Décomposition des exigences

Chaque nœud représente une exigence d'un niveau de détail donné. Les arcs liant un nœud d'un niveau de détail à d'autres nœuds d'un niveau de détail moins élevé représentent la relation de décomposition.

Cette relation qui lie un nœud cible (*i.e.* représentant une exigence abstraite d'un niveau k) à ses nœuds sources (*i.e.* représentants des exigences plus précises à un niveau k+1) est contrainte par une fonction logique qui permet d'exprimer l'influence de la véracité des nœuds sources sur l'exigence représentée par le nœud cible.

Dans la suite, pour décrire le modèle GRADEI, les notations suivantes sont adoptées :

- p = nombre de niveaux de détail appelé profondeur du graphe, $p \in \mathbb{N}^*$.
- m = nombre de nœuds du graphe, $m \in \mathbb{N}$.
- n = nombre d'arcs du graphe, $n \in \mathbb{N}$.
- $\prod$ ensemble des instants selon le type d'échelle temporelle choisie. Par hypothèse, $\prod \subseteq \mathbb{N}$ dans le cas d'une échelle de temps discret ou $\prod \subseteq \Re$ dans le cas d'une échelle de temps continue.
- *String* = {chaîne_de_caractères | chaîne_de_caractères respecte la règle BNF suivante chaîne_de_ caractères::= ('a'..'z' | 'A'..'Z') ('a'..'z' | 'A'..'Z' | '1'..'9' | '_' | '-')*}
- *Fait* = {fait | fait ∈ $VM \cup PM \cup Pr$} avec :
    - *VM = {variables de modélisation extraites du modèle de processus}*
    - *PM = {paramètres de modélisation extraits du modèle de processus}*

- *Pr = {prédicats de modélisation extraits du méta modèle de BMPN enrichi}*

Le modèle GRADEI est formalisé sous la forme d'un graphe **G** défini comme suit :

$$G ::= (E, N, N_0)$$

Où :

- E est l'ensemble des arcs du graphe :

$$E = \{E_j | E_j = (sourceE_j, cibleE_j); j \in [0, n]\}$$

Avec ($SourceE_j$, $CibleE_j$) $\in$ N x N : la source et la cible de l'arc sont des nœuds du graphe en respectant l'hypothèse suivante :

- $niveauSourceE_j > niveauCibleE_j$ *i.e.* le niveau de détail (variant de 0 qui est par convention le niveau de détail du nœud N_0 ou nœud racine jusqu'à p qui est la profondeur du graphe) du nœud source de l'arc est strictement supérieur au niveau du nœud cible de l'arc. En conséquence, un nœud ne peut pas être source et également cible d'un même nœud.

- N est l'ensemble des nœuds du graphe :

$$N = \{N_i | N_i ::= (nom_i, description_i, relation_i, fait_i, niveau_i, valeur_i); i \in [0, m]\}$$

Avec :

- (nom_i, $description_i$) $\in$ String x String, le nom et la description informelle de l'exigence formalisée par le nœud N_i,
- $fait_i \subseteq Fait$
- $relation_i$ ::= (T_i, θe, θc) avec :
  - $T_i \subseteq \prod$
  - θe : $fait_i \cup T_i \rightarrow \{0, 1\}$ est la fonction logique qui décrit la condition nécessaire, mais non suffisante, pour que l'exigence modélisée par le nœud N_i soit vérifiée. Cette condition s'exprime seulement sur les variables de modélisation, les paramètres de modélisation et les prédicats de modélisation extraits du modèle de processus collaboratif.

- θc : $NN_i \rightarrow \{0,1\}$ *i.e.* $\{\text{valeur}(N_1),\ldots, \text{valeur}(N_k)\} \rightarrow \{0,1\}$ est la fonction logique qui décrit la condition nécessaire, mais non suffisante, pour que l'exigence modélisée par le nœud N_i soit vérifiée. Cette condition permet d'interpréter les résultats (valeurs) des nœuds sources. L'ensemble NN_i est l'ensemble des nœuds sources de N_i défini comme suit :

$$NN_i = \{N_j | \exists E_k(N_j, N_i)\ pour k \in [0,n], j \in [0,m]\ avec\ i \neq j\}$$

θe et θc sont des fonctions logiques booléennes qui sont par défaut définies comme θe = vrai et θc = vrai et dont la syntaxe est décrite par les règles BNF données en Annexe B.

- $niveau_i \in [0, p]$ indique le niveau de détail de l'exigence. Par convention, la profondeur maximale du graphe est égale à p et le niveau du nœud racine est égal à 0
- $valeur_i \in$ {vrai, faux} est le résultat de la vérification pour un nœud considéré in absentia 0 (faux). La valeur d'un nœud i est telle que :

$$valeur_i = \theta e \wedge \theta c$$

- N_0 est par convention le nœud racine du graphe représentant l'exigence la plus abstraite (niveau de détail = 0) défini donc par :

$$\exists! N_0 = ('interopérabilité', description_0, relation_0, fait_0, 0, valeur_0) \in N$$

Nous pouvons noter qu'une exigence peut être caractérisée par une hypothèse temporelle. Elle peut être atemporelle, c'est-à-dire considérée comme indépendante du temps. Dans ce cas, elle est vérifiable à chaque instant (l'ensemble *T* est vide). Elle peut être temporelle, c'est-à-dire devant être vérifiée seulement à certaines phases du cycle de vie de la collaboration. L'hypothèse temporelle donnée à une exigence revêt un caractère important pour sa vérification. En effet, de cette hypothèse dépend, par la suite, le choix de la technique de vérification à utiliser présentée au chapitre 4.

Nous pouvons également constater - au travers de la formalisation de l'arbre - que le respect (ou non) d'une exigence abstraite (nœud cible N_i) dépend de toutes les exigences plus élémentaires (représentées par des nœuds N_j dits nœuds sources de N_i) qui la décomposent (pour j := 1 à *n*, *n* = nombre d'exigences décomposant l'exigence représentée par le nœud N_i)

ainsi que de la relation qui lie l'exigence représentée par le nœud N_i aux exigences représentées par les nœuds sources N_j. Le choix de la fonction logique qui va contraindre l'interprétation de cette relation est laissé à l'appréciation de l'utilisateur. La figure suivante synthétise la formalisation du modèle GRADEI proposé.

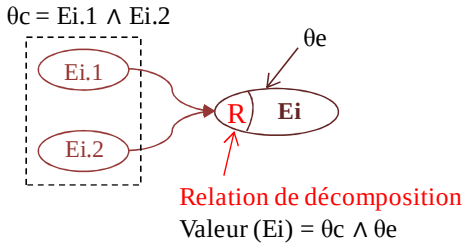


Figure 15. Positionnement des éléments de la formalisation du modèle GRADEI

4.3. Les classes d'exigences d'interopérabilité

Nous avons établi précédemment quatre classes d'exigences d'interopérabilité : les exigences de compatibilité, les exigences d'interopération, les exigences de réversibilité et les exigences d'autonomie. La définition et les concepts liés aux différentes classes d'exigences d'interopérabilité définies précédemment, sont présentés et développés dans les sections suivantes[14].

4.3.1. La classe des exigences de compatibilité

Nous définissons une exigence de compatibilité comme « *un énoncé qui prescrit une fonction, une aptitude ou une caractéristique - considérée comme invariable durant toute la collaboration et liée aux barrières d'interopérabilité et à chaque niveau d'interopérabilité - que les partenaires doivent satisfaire avant toute collaboration* ».

Les exigences de compatibilité sont principalement dérivées du cadre d'interopérabilité d'entreprise développé par le Réseau d'Excellence INTEROP (INTEROP, 2007) d'une part et, par la matrice de compatibilité proposée dans (Daclin, 2007) d'autre part. Ainsi, pour chaque barrière d'interopérabilité et à chaque niveau d'interopérabilité, il s'agit de déterminer et de

[14] Pour exprimer les exigences de compatibilité et d'interopération, présentées dans la suite, nous adoptons la convention suivante :

- Première lettre : classe d'exigence ;
- Seconde lettre : niveau concerné par l'exigence ;
- Troisième lettre : catégorie concerné par l'exigence ;
- Indice : nom de l'exigence.

dégager des exigences qui peuvent par la suite, et suivant leur satisfaction ou non, nous renseigner sur la présence d'incompatibilités entre des partenaires.

Ainsi, les exigences de compatibilité sont liées aux trois catégories mentionnées précédemment (conceptuelle, technologique et organisationnelle). Pour la catégorie conceptuelle les exigences considèrent, plus précisément, les aspects liés à la sémantique et à la syntaxe. Pour la catégorie technologique, les exigences sont exprimées au regard des plateformes, des infrastructures informatiques et des protocoles de communication déployés chez les partenaires. Cependant, cette catégorie ne se limite pas au couplage des applications informatiques ou des technologies de l'information. Elle peut considérer également d'autres aspects techniques (*e.g.* compatibilité d'un outil utilisé chez un partenaire avec la machine outil d'un autre partenaire…). Enfin, pour la catégorie organisationnelle, les exigences s'intéressent au couplage des partenaires au niveau des personnes et des structures organisationnelles. Il s'agit d'établir des exigences relatives à la responsabilité, l'autorité, aux compétences, aux autorisations, *etc*. des personnes qui sont impliquées dans le processus collaboratif ainsi qu'à la connaissance des structures et des organisations internes des parties en présence.

Un ensemble d'exigences de compatibilité est établi en tenant compte de ces concepts. La figure suivante représente les concepts liés aux exigences de compatibilité structurées sur le modèle GRADEI.

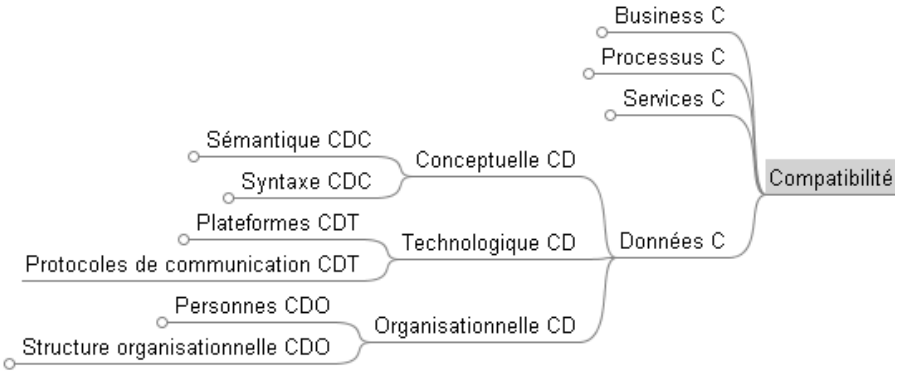


Figure 16. Exigences de compatibilité structurées sur le modèle GRADEI

Les exigences de compatibilité se concentrent sur l'interfaçage entre différents partenaires. L'ensemble de ces exigences vise donc à être vérifié lors de la phase de couplage des partenaires. En conséquence, une fois ces exigences satisfaites, elles peuvent être supposées comme invariables durant toute la durée de la collaboration.

Ainsi, plusieurs exigences de compatibilité peuvent être formulées à partir des besoins présentés en section 3. Par exemple le besoin numéro 5 « *Il faut pouvoir envoyer et recevoir des données exploitables, et être sûr de leur réception par l'autre partenaire pendant le déroulement de l'activité* » est traduit en trois exigences. La première est une exigence au niveau des données pour la catégorie technologique. Elle est liée à l'exploitation des données transmises par un émetteur vers un récepteur et s'exprime comme suit :

$CDT_{Traduction}$[15] : « *le partenaire récepteur dispose des moyens nécessaires de traduction des données* ».

Cette exigence traduit le besoin du partenaire récepteur au regard de la traduction de données homogènes ou hétérogènes et donc de leur exploitation. La seconde exigence est liée à la compatibilité au niveau des données. Elle concerne la catégorie organisationnelle et s'intéresse aux autorisations pour l'échange de données :

$CDO_{Autorisation}$: « *le partenaire émetteur possède les autorisations requises pour effectuer des échanges avec le récepteur* ».

Cette exigence revêt un caractère important dans le cas de transmission de données sensibles par exemple. Enfin, la troisième exigence est exprimée au niveau services pour la catégorie organisationnelle et concerne la définition des responsabilités :

$CSO_{Responsabilité}$: « *chaque service possède un responsable clairement défini qui a l'autorité sur le reste de l'équipe* »

Elle traduit le fait qu'un service possède un responsable clairement identifié. L'objectif de la satisfaction de cette exigence est d'éviter, par exemple, des pertes de temps susceptibles de porter préjudice au processus lors de son exécution.

Ensuite, une exigence de compatibilité au niveau des données pour la barrière conceptuelle, et liée à la sémantique des données échangées, est formulée à partir du besoin numéro 6 « *Homogénéiser la communication quelle qu'en soit la forme, donc disposer d'une*

[15] Convention adoptée :
- Première lettre : classe d'exigence ;
- Seconde lettre : niveau concerné par l'exigence ;
- Troisième lettre : catégorie concerné par l'exigence ;
- Indice : nom de l'exigence.

sémantique comprise et partagée entre les deux partenaires ». Une exigence peut être exprimée au niveau des données pour la catégorie conceptuelle comme suit :

$CDC_{Sémantique}$: « *Le partenaire récepteur maîtrise le sens de l'information émise par l'émetteur* ».

4.3.2. La classe des exigences d'interopération

Nous définissons une exigence d'interopération comme « *un énoncé qui prescrit une fonction, une aptitude ou une caractéristique – considérée comme variable pendant la collaboration - liée aux interactions entre des partenaires* (*i.e. interopérations qui peuvent être présentes au sein d'un processus collaboratif*) *- que les partenaires doivent satisfaire* ».

En d'autres termes, cette classe vise à s'assurer que l'échange et/ou l'exploitation entre des objets (*e.g.* activité, ressource, contrôle, sous-processus…) du processus sont réalisés correctement. Il s'agit donc de déterminer et de formuler des exigences liées à la phase opérationnelle du partenariat. Cette classe couvre donc les exigences concernant aussi bien les interactions entre activités que les interactions entre des ressources et des activités. Cela peut concerner la disponibilité, l'aptitude… d'une ressource au regard d'une ou plusieurs activités. Cela peut concerner les interactions entre différentes activités, *e.g.* une activité attend la confirmation d'un échange, une activité souhaite débuter à une date précise…. Enfin, cela peut concerner également les performances d'interopération (Daclin, 2007). Dans ce dernier cas, nous nous intéressons à exprimer des exigences vis-à-vis des critères comme le coût d'interopération, le temps d'interopération et la qualité d'interopération. Pour le premier critère, il peut s'agir, par exemple, de s'assurer que le coût d'un échange ne dérive pas par rapport au coût initialement prévu à supposer que ce coût initial puisse être estimé. Pour le second critère, il peut être envisagé de considérer le temps d'exploitation (*e.g.* temps nécessaire à une activité d'un partenaire pour exploiter une machine outil fournie par un autre partenaire). Pour le dernier critère, on peut s'assurer, par exemple, que le nombre de pièces fournies par une activité correspond à celui attendu par une autre activité.

En conséquence, un ensemble d'exigences d'interopération est établi en fonction des interactions possibles et structurées dans le modèle GRADEI comme le montre la figure ci-dessous.

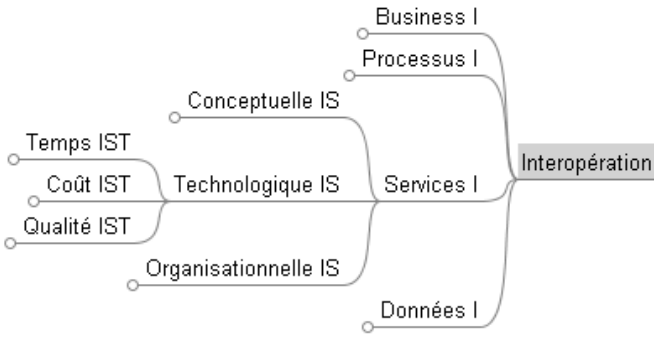


Figure 17. Exigences d'interopération structurées sur le modèle GRADEI

Les exigences d'interopération concernent la phase d'exécution du processus collaboratif. Aussi, leur véracité peut être considérée comme variable au cours de la collaboration eu égard aux interactions elles-mêmes.

Si nous reprenons le besoin numéro 1 à partir duquel nous avons précédemment extrait des exigences de compatibilité, il est également possible de dégager une exigence d'interopération située au niveau des services pour la catégorie organisationnelle et exprimée comme suit :

$ISO_{réception}$[16] : « *le récepteur accuse réception des données à l'émetteur* »

Considérant le besoin numéro 8 qui s'intéresse plus précisément aux interactions entre une ressource et une activité « *Une ressource doit être disponible au moment où l'on fait appel à elle* », il est possible de dégager deux exigences liées au niveau service et pour la catégorie organisationnelle. La première exigence se focalise sur la disponibilité de la ressource à partir du moment où celle-ci est sollicitée par une activité et s'exprime comme suit:

$ISO_{TempsDisponibilité}$: « *la ressource est disponible lorsque l'activité la sollicite* ».

La seconde exigence vise à s'assurer qu'une ressource participera bien à l'exécution complète d'une activité donnée. Le but est par la suite d'éviter que cette ressource soit allouée

[16] Convention adoptée :
- Première lettre : classe d'exigence ;
- Seconde lettre : niveau concerné par l'exigence ;
- Troisième lettre : catégorie concerné par l'exigence ;
- Indice : nom de l'exigence.

à une autre activité durant toute la durée de la première activité qui la sollicite. Ainsi, cette deuxième exigence est exprimée telle que :

$ISO_{TempsExécution}$: « *la ressource est active sur toute la durée d'exécution de l'activité* ».

4.3.3. La classe des exigences d'autonomie

Nous définissons une exigence d'autonomie comme « *un énoncé qui prescrit une fonction, une aptitude ou une caractéristique - liée à la capacité qu'ont les partenaires à garder leur propre gouvernance et leurs propres capacités opérationnelles lors de la collaboration - que les partenaires doivent satisfaire.* »

Cette classe d'exigences se rapporte à la faculté qu'ont les partenaires à garder leur autonomie pendant qu'ils sont impliqués dans un processus collaboratif. Cette autonomie se caractérise par, l'**autonomie de gouvernance**, et l'**autonomie opérationnelle**. L'autonomie de gouvernance est nécessaire pour que les partenaires puissent toujours garder leur liberté de choix de décision sans que cela n'engendre des conflits et des désaccords pouvant, au mieux, freiner la collaboration. L'autonomie opérationnelle est nécessaire pour que les partenaires puissent garder une certaine liberté d'action pour atteindre leurs propres objectifs. Ces deux types d'autonomie doivent être présents aux niveaux business, processus, et services de l'entreprise et pour chaque barrière d'interopérabilité (conceptuelle, technologique et organisationnelle) comme le schématise la figure suivante.

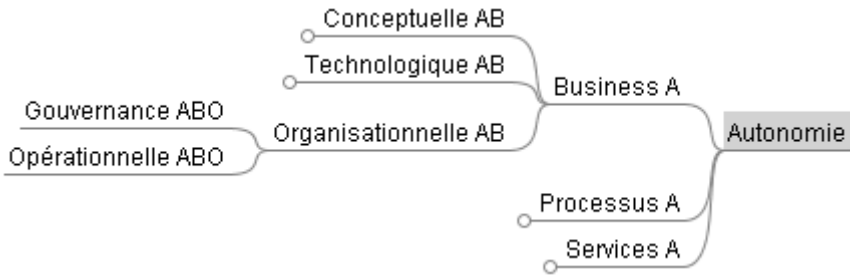


Figure 18. Exigences d'autonomie structurées sur le modèle GRADEI

A partir d'un des besoins exprimés précédemment, et plus précisément le besoin numéro 4, « *La gouvernance des processus d'approvisionnement relie explicitement les exigences opérationnelles à l'architecture technique à travers le financement de projets*»., nous pouvons extraire deux exigences d'autonomie au niveau business et pour la barrière organisationnelle. Elles permettent de s'assurer que chaque partenaire assure la maîtrise du choix de ses fournisseurs et des coûts associés.

$ABO_{Gouvernance}$[17] : « *un partenaire du processus collaboratif reste responsable du choix de ses fournisseurs* ».

$ABO_{GouvernanceCoût}$: « *Le partenaire maîtrise les coûts relatifs au choix des fournisseurs* ».

Une troisième exigence concerne plus particulièrement l'autonomie opérationnelle au niveau des processus et pour la barrière technologique. Elle s'exprime comme suit.

$APT_{Opérationnelle}$: « *un processus garde un flux physique nécessaire d'approvisionnement* ».

4.3.4. La classe des exigences de réversibilité

Une exigence de réversibilité est définie comme « *un énoncé qui prescrit une fonction, une aptitude ou une caractéristique - liée à la capacité qu'a un partenaire donné à revenir à son état initial - que l'entreprise doit satisfaire à la fin de la collaboration* ».

A la fin de la collaboration, un système donné doit être en mesure de poursuivre à nouveau sa propre finalité, réaliser sa mission et atteindre ses objectifs. Les exigences de réversibilité s'assurent donc que le système retrouve bien son état initial dans le cas où la mise en œuvre de l'interopérabilité a mené à des adaptions et/ou des modifications (*e.g.* organisation, méthode, comportement, aptitudes…) du système lui même. Il s'agit essentiellement pour ces exigences de constater l'impact (si impact il y a) sur l'intégrité, la stabilité et la performance du système. En ce qui concerne l'intégrité, le système doit revenir à un mode de fonctionnement connu et ce, malgré une possible configuration différente de l'original. En ce qui concerne la stabilité, le système doit être en mesure de maintenir sa viabilité. Pour la performance, le système doit remplir à nouveau - si ce n'est mieux - les objectifs de performance qui étaient les siens avant collaboration moyennant éventuellement une tolérance *i.e.* le système peut être amené à admettre une perte de performance connue et définie au préalable.

[17] Convention adoptée :
- Première lettre : classe d'exigence ;
- Seconde lettre : niveau concerné par l'exigence ;
- Troisième lettre : catégorie concerné par l'exigence ;
- Indice : nom de l'exigence.

En conséquence, les exigences de réversibilité sont formulées par la considération de ces trois critères que sont l'intégrité, la stabilité et la performance. Une série d'exigences de réversibilité sont structurées sur le modèle GRADEI comme le montre la figure suivante.

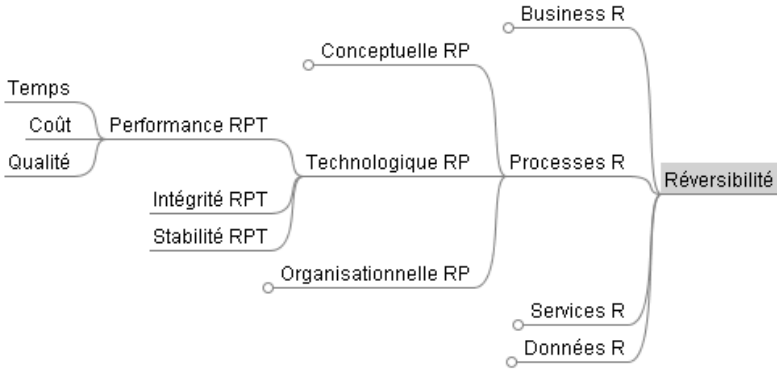


Figure 19. Exigences de réversibilité structurées sur le modèle GRADEI

Considérons à présent le besoin numéro 7 issu de l'enquête auprès des industriels et défini comme « *Rester aussi performant en terme de coût, de qualité (produit et service) et de délais après le partenariat que ce que l'on était performant avant même que nous n'y soyons impliqués* ».

Il est possible d'en extraire trois exigences, liées à la réversibilité, qui se focalisent chacune sur un critère de performance précis. Ainsi, la première exigence considère le critère de coût :

$RST_{PerformanceCoût}$[18] : « *le coût d'une activité post-collaboration correspond – tolérances comprises - au coût ante-collaboration.* »

La deuxième exigence de réversibilité concerne le critère de performance de délai :

$RST_{PerformanceTemps}$: « *le temps d'exécution d'une activité post-collaboration correspond – tolérances comprises - au temps d'exécution ante-collaboration.* »

Enfin, la troisième exigence s'intéresse au critère de performance de qualité :

[18] Convention adoptée :
- Première lettre : classe d'exigence ;
- Seconde lettre : niveau concerné par l'exigence ;
- Troisième lettre : catégorie concerné par l'exigence ;
- Indice : nom de l'exigence.

$RST_{PerformanceQualité}$: « *le nombre de produits conformes fournis par une activité post-collaboration correspond - tolérances comprises - au nombre de produits conformes anté-collaboration.* »

4.4. Influence des classes des exigences d'interopérabilité, les unes sur les autres

Les classes d'exigences d'interopérabilité que nous avons définies sont associées aux notions relatives à l'interopérabilité. Toutes ces notions sont liées entre elles. En conséquence, lorsqu'une une exigence d'une des classes n'est pas satisfaite par le modèle du processus collaboratif, elle peut influencer l'interprétation d'exigences provenant des autres classes comme présenté sur le Tableau 5.

Tableau 5. Influence des classes d'interopérabilité les unes sur les autres

Influence →	**Exigences de compatibilité**	**Exigences d'interopération**	**Exigences d'autonomie**	**Exigences de réversibilité**
Exigences de compatibilité	--	Oui	Non	Non
Exigences d'interopération	Oui	--	Oui	Oui
Exigences d'autonomie	Non	Oui	--	Oui
Exigences de réversibilité	Non	Non	Non	--

A titre d'exemple, lorsque nous effectuons la vérification de l'exigence de compatibilité $CSO_{Aptitude}$ exprimée telle que : « *Une ressource possède l'aptitude que requiert l'activité la sollicitant* », sa non satisfaction peut influencer l'interprétation d'une exigence d'interopération. En effet, l'exigence d'interopération $ISO_{DisponibilitéRessource}$ qui s'exprime telle que : « *Chaque service utilise la ressource nécessaire pendant le temps d'exploitation* » est directement impactée par la première ($CSO_{Aptitude}$). Dans ce cas, même si cette dernière exigence est satisfaite, l'interopération (partage) ne peut être effectuée puisque la ressource allouée ne possède pas l'aptitude requise. Ceci montre bien la relation existante entre les classes d'exigences d'interopérabilité et, de ce fait, leur influence mutuelle. Cependant, il est à noter que cette influence n'entrave en rien l'interprétation du modèle GRADEI. En effet, dans le cadre de nos travaux, nous ne considérons pas pour l'instant cette influence lors de

l'interprétation des résultats obtenus par le modèle GRADEI, mais nous évaluons chaque classe d'exigence indépendamment les unes des autres. Il s'agit maintenant de permettre l'interprétation de ce modèle. Celle-ci repose sur le mécanisme de propagation.

4.5. Mécanisme de propagation à travers la structure du modèle GRADEI

L'objectif est maintenant de vérifier (*i.e.* de trouver la valeur de véracité $valeur_0$) de l'exigence modélisée par le nœud N_0. Deux cas sont possibles pour vérifier la véracité d'une exigence d'un niveau *k* (*k*=0 à *p*).

- **L'exigence n'est pas décomposée.** Dans ce cas, sa valeur est donnée par la mise en œuvre d'une des techniques de vérification telles que l'expertise, la vérification par *model checking* ou la vérification en utilisant des graphes conceptuels présentées dans le chapitre suivant. La mise en œuvre de ces techniques dépend de la classe de l'exigence visée comme le précise le tableau suivant.

Tableau 6. Impacts et techniques de vérification pour chaque classe

	Impacts	Vérification			
		Expertise	Simulation	Model checking	Graphe conceptuel
Compatibilité	•Interfaçage •Performance de la collaboration	Oui	Oui	Oui	Oui
Interopération	•Interaction •Performance de la collaboration	Oui	Oui	Oui	Oui
Réversibilité	•Reprise du fonctionnement initial •Performances propres	Retour d'expérience	Oui	Non	Non
Autonomie	•Gouvernance •Capacité à être opérationnelle	Oui	Oui	Oui	Oui

- **L'exigence à un niveau k est décomposée en une série d'exigences plus précises *i.e.* de niveau de détail k+1**. Dans ce cas, le principe repose sur l'utilisation de la relation associée au nœud modélisant l'exigence et sur l'usage d'un mécanisme de propagation. Rappelons que cette relation met en avant deux fonctions logiques θc et θe.

A titre d'exemple, pour le modèle GRADEI présenté en Figure 20, il est proposé d'utiliser les fonctions logiques θc suivantes :

$$Interopérabilité = Compatibilité \wedge Interopération \wedge Réversibilité \wedge Autonomie \quad (1)$$

$$Compatibilité = Business\ C \wedge Processus\ C \wedge Services\ C \wedge Données\ C \quad \text{..... (2)}$$

$$Services\ C = Conceptuelle\ CS \wedge Technologique\ CS \wedge Organisationnelle\ CS \text{.... (3)}$$

$$Conceptuelle\ CS = Syntaxe\ CDC \vee Sémantique\ CSC \text{..... (4)}$$

En d'autres termes, pour que l'exigence abstraite d'interopérabilité modélisée par le nœud N_0 soit satisfaite, il faut bien évaluer d'abord puis propager les valeurs de véracité des exigences de compatibilité, d'interopération, de réversibilité et d'autonomie. Ces exigences doivent être respectées comme indiqué par l'équation (1). A son tour, pour que l'exigence de compatibilité soit respectée, il faut évaluer et propager la valeur de véracité de chacun des nœuds modélisant les exigences des niveaux d'interopérabilité comme représenté par l'équation (2). Par itération, les exigences de chaque niveau d'interopérabilité sont respectées si et seulement si chaque exigence présente pour ce niveau est respectée comme le montre l'équation (3) avec le niveau services. L'opération d'évaluation et de propagation est répétée jusqu'aux exigences primaires (non décomposées). Nous pouvons noter que ces trois premières équations représentent les trois premiers niveaux du modèle GRADEI (interopérabilité, niveaux, catégories) et sont invariants. Enfin, il s'agit de vérifier l'exigence dite « *conceptuelle* » comme le montre l'équation (4). Cette exigence sera satisfaite si et seulement si une des deux exigences qui la décomposent est respectées (opérateur logique « OU »).

Rappelons que le choix des fonctions logiques θc et θe utilisées pour évaluer la valeur de véracité d'un nœud est laissé à la discrétion de l'utilisateur. De plus, certaines des exigences présentes dans le référentiel et certaines exigences spécifiées par l'utilisateur peuvent ne pas être satisfaites et donc en apparence gêner la collaboration. Cette gêne peut cependant être considérée comme négligeable ou représenter un risque acceptable par l'utilisateur. Dans ce sens, l'utilisateur peut *de facto* admettre et imposer la véracité d'une exigence et cela sans faire appel à une technique de vérification.

La figure suivante montre un exemple de propagation des valeurs de véracité des nœuds du modèle GRADEI modélisant chacun une exigence plus ou moins abstraite d'après l'exemple précédent et les choix de l'utilisateur pour ce qui est des fonctions logiques θc utilisées.

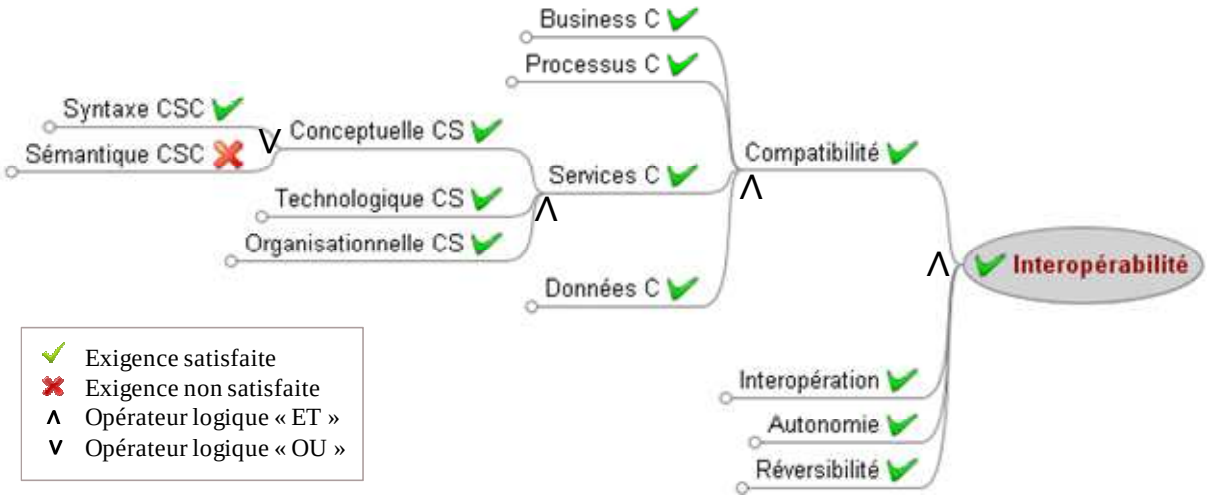


Figure 20. Propagation d'exigences initiales à travers la structure du modèle GRADEI

Pour une meilleure compréhension, l'algorithme de propagation est présenté dans ce qui suit :

```
Début
    /* initialisation du tableau des valeurs de vérité des nœuds du graphe */
    /* valeur(x) contient la valeur de véracité du nœud x du graphe */
    n := card(E) ;
    m := card(N) ;
    Dim valeur(m) ;
    /* rechercher les nœuds connectés au nœud racine */
    NN0 := construitEnsembleNN(0) ;
    /* algorithme de propagation des valeurs de vérité dans l'ensemble du graphe */
    propagation (NN0,0) ;
Fin
```

fonction propagation (NN,j)

```
Début
    taille := card(NN) ;
    si taille = 0 alors
        valeur(j) := noeudj.θe ;
        retourner (valeur(j)) ;
    fin si
    Dim vérité(taille) : true ;
    pour k = 0 à taille - 1 faire
        NNk := construitEnsembleNN(NN(k)) ;
        Vérité(k) := propagation(NNk,k)
    fin pour k
    /* la valeur de vérité du nœud courant est donnée */
    /* par l'interprétation des fonctions θe et θc */
    valeur(j) := interpreteTetaE(noeudj.θe, vérité) AND interpreteTetaC(noeudj.θc,
    vérité) ;
    retourner (valeur(j)) ;
Fin
```

Avec :

- propagation(tableauNN, indice) est la fonction récursive permettant de calculer la totalité des valeurs de vérités du graphe G.
- construitEnsembleNN(i) est la fonction qui retourne un tableau contenant les nœuds connectés au nœud N_i
- interpreteTetaC(noeudj.θc, vérité) est la fonction qui interprète la fonction logique θc et retourne la valeur booléenne correspondante.
- interpreteTetaE(noeudj.θe, vérité) est la fonction qui interprète la fonction logique θ_e et retourne la valeur booléenne correspondante.
- (noeudj.θe) est la fonction logique θe du noeudj.
- (noeudj.θc) est la fonction logique θc du noeudj qui lie l'exigence représentée par le nœudj aux exigences le décomposant qui sont de niveau supérieur.

4.6. Proposition d'un référentiel des exigences d'interopérabilité sur le modèle GRADEI

Le référentiel d'exigences d'interopérabilité bâtie selon le modèle GRADEI que nous proposons dans cette thèse est représenté partiellement en Figure 21. Le nœud racine (ellipse grise) représente l'exigence d'interopérabilité globale la plus abstraite. Exprimer et prouver cette exigence revient à exprimer et à prouver l'aptitude d'une unité organisationnelle (d'une entreprise ou d'une équipe) à pouvoir interopérer lorsqu'elles sont impliquées dans un processus collaboratif. Cette exigence est décomposée afin d'obtenir des exigences précises, claires, formalisables et donc vérifiables sur le modèle du processus collaboratif. Le graphe G du modèle GRADEI est donc complété sur les dernières branches avec des exigences précises extraites de l'étude bibliographique et des résultats de l'enquête effectuée auprès d'industriels. Par conséquent, d'éventuels enrichissements avec de nouvelles exigences d'interopérabilité se font sur les derniers niveaux d'abstraction de l'arbre causal. Ce référentiel dans sa version actuelle est présenté en Annexe C.

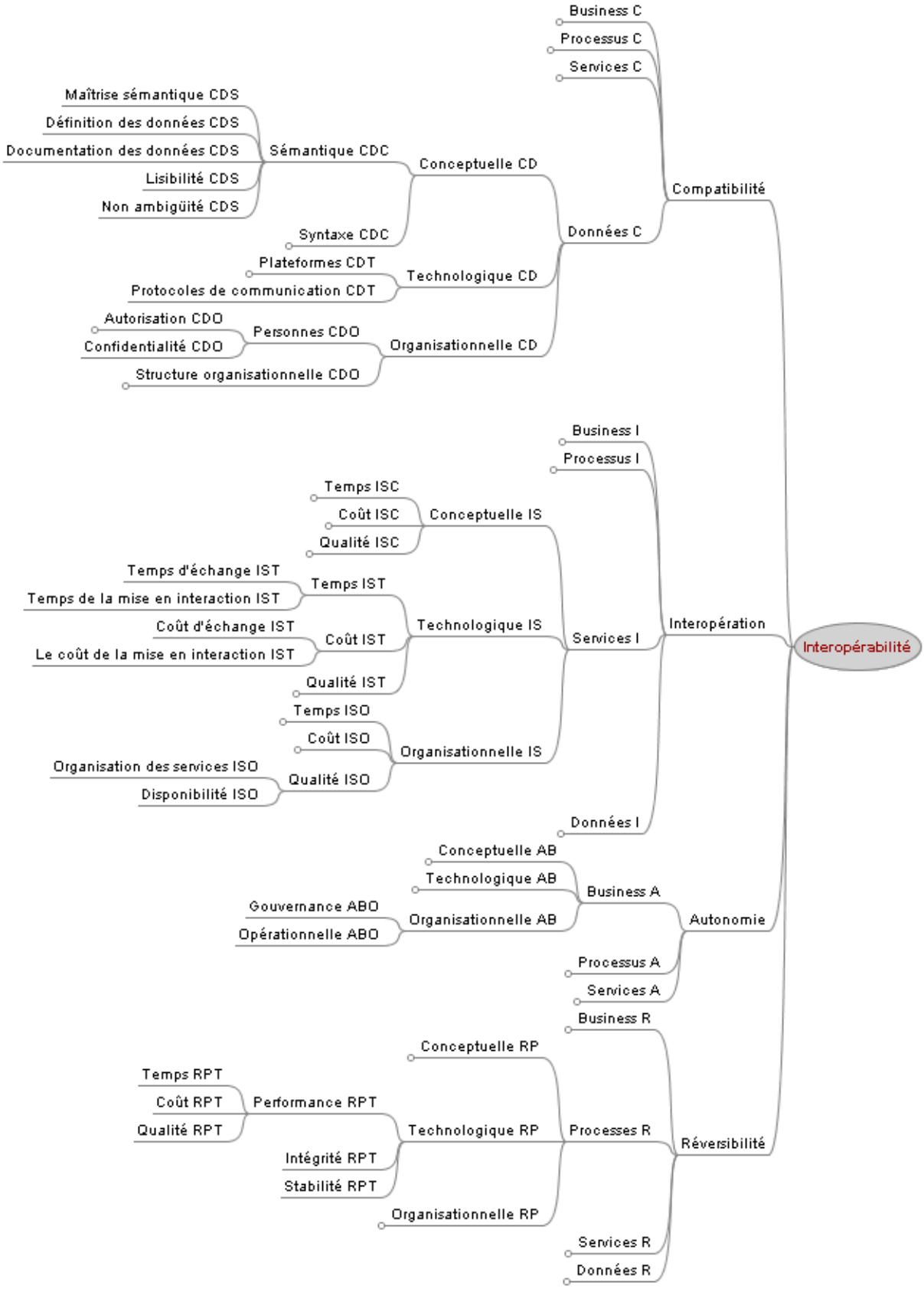


Figure 21. Modèle partiel du modèle GRADEI

5. Conclusion

Dans ce chapitre, nous avons présenté une approche anticipative pour la détection de problèmes d'interopérabilité. Pour ce faire, nous avons mis l'accent sur la nécessité d'utiliser des techniques de vérification formelle qui permettent d'anticiper et de donner une indication sur l'existence et la localisation d'éventuels problèmes d'interopérabilité. Nous avons établi que ces problèmes sont localisés et détectés en procurant une preuve formelle de la satisfaction ou non d'exigences d'interopérabilité qui reflètent les problèmes.

Nous nous sommes focalisés sur la présentation d'un modèle d'exigence reflétant les problèmes d'interopérabilité qui peuvent apparaitre lors de l'exécution d'un processus collaboratif. Ces exigences sont formulées à partir de besoins récoltés (étude bibliographique et enquête auprès d'industriels). Nous nous sommes attachés, par la suite, à structurer et classifier les exigences d'interopérabilité.

Dans un premier temps, nous avons classé les exigences d'interopérabilité en quatre classes liées aux notions relatives à celle-ci : les exigences de compatibilité, les exigences d'interopérabilité, les exigences de réversibilité et les exigences d'autonomie. Dans un second temps, nous avons établi un modèle de GRAphe de Décomposition des Exigences d'Interopérabilité (GRADEI) pour la structuration des exigences. Ce modèle offre une représentation claire et structurée des exigences et permet l'utilisation de mécanismes de propagation. En effet, le modèle GRADEI permet de lier une exigence abstraite à des exigences plus claires avec une fonction logique. Les mécanismes de propagation permettent de démontrer la satisfaction d'une exigence abstraite par le biais des exigences la décomposant.

Les exigences plus précises doivent donc être claires, explicites et formalisables pour permettre leur vérification. Ces exigences sont décrites au moyen d'un langage naturel structuré (sujet, verbe, complément) afin de permettre leur vérification par :

- Des techniques de vérification par preuve formelle. Dans ce cas précis, ces exigences doivent être formalisées en propriétés à l'aide d'un langage formel.

- Des techniques d'expertise car la description formelle de ces exigences n'est pas possible avec les techniques proposées ou impossible avec les données présentes dans le modèle du processus collaboratif.

Dans cette thèse, des techniques de vérification formelles sont employées pour prouver la véracité d'une exigence. L'utilisation de ce type de techniques requiert au préalable la transformation du modèle du processus collaboratif et la formalisation des exigences d'interopérabilité en langage formel. Nous nous attachons, dans le chapitre 4 à la présentation de ces étapes de notre approche.

Chapitre 4

Vérification des exigences d’interopérabilité : modélisation et formalisation

Table des matières du chapitre 4

1. Introduction

L'objectif de ces travaux est d'aider à détecter de manière anticipative des problèmes d'interopérabilité. Ce quatrième chapitre présente la démarche adoptée pour assurer la vérification des exigences d'interopérabilité définies et établies dans le chapitre précédent.

Nous avons présenté dans le chapitre 2 un panorama des techniques de vérification essentielles existantes dans la littérature. Rappelons qu'il existe des techniques de vérification non formelle (expertise, simulation et test), des techniques de vérification formelles (*model checking* et *theorem proving*) ainsi qu'une technique de vérification se basant sur les Graphes Conceptuels. Nous avons montré que ces trois types de vérification peuvent être complémentaires sachant que le choix d'une de ces techniques est fonction du caractère temporel ou atemporel de l'exigence et de son niveau de détail. Nous proposons donc d'utiliser :

- le ***model checking*** comme technique de vérification formelle pour analyser le comportement d'un processus collaboratif vis-à-vis de l'interopérabilité. Nous nous intéressons, dans ce cas, à la vérification des exigences d'interopérabilité temporelles.
- la technique de vérification basée sur les **graphes conceptuels** pour analyser essentiellement la structure et la cohérence du modèle vis-à-vis de l'interopérabilité. Nous nous consacrons, ici, à la vérification des exigences d'interopérabilité atemporelles.
- l'**expertise** comme technique de vérification informelle, dans les cas où l'exigence (qu'elle soit temporelle ou atemporelle) ne peut être suffisamment décomposée ou décrite pour pouvoir être prouvée formellement. Cette dernière technique n'est pas étudiée, plus en détail, dans ces travaux de recherche.

2. Vérification des exigences d'interopérabilité

La vérification des exigences d'interopérabilité repose sur l'application de techniques de vérification. Cela nécessite de prendre en compte plusieurs besoins à commencer par la modélisation du processus qui se doit de respecter certains principes. Dans nos travaux, nous

utilisons le langage de modélisation de processus collaboratif BPMN[19]. Ce langage, initialement développé pour modéliser les processus métier, fournit une notation standardisée facilement compréhensible par tous les acteurs impliqués dans la conception, le développement et le suivi d'un processus collaboratif. Cependant, ce langage ne permet de manipuler les concepts liés aux exigences d'interopérabilité tel que proposés dans le modèle GRADEI. En effet, et à titre d'exemple, considérons l'exigence : « *la ressource est disponible lorsque l'activité la sollicite* ». Dans ce cas, il faut s'assurer que le modeleur dispose des concepts « *ressource* », « *activité* » et d'une relation signifiant la sollicitation d'une ressource dès le démarrage ou en pré condition du lancement d'une activité. Or, le langage BPMN, tout du moins dans sa version 1.2, s'avère insuffisant pour pouvoir manipuler le concept de ressource par exemple.

Il est donc proposé, dans un premier temps, d'enrichir **conceptuellement** et **opérationnellement** cette version du langage BPMN en tenant compte des besoins suivants :

1. Un enrichissement doit préserver la **sémantique conceptuelle** de BPMN *i.e.* il ne peut introduire de modifications dans la dénomination, la définition usuelle et l'interprétation des concepts et des relations existantes à l'origine dans le langage. Plus globalement, il se doit de respecter la cohérence globale du méta modèle de BPMN.
2. Un enrichissement doit préserver la **sémantique opérationnelle** du langage BPMN quand elle existe, ou la promouvoir, quand elle est absente.

En d'autres termes, tout enrichissement doit préserver la cohérence globale et l'interprétation d'un modèle élaboré au moyen de la version enrichie de BPMN. Ce modèle peut être traduit sans perte ni ajout en un modèle établi avec la version initiale de BPMN en élaguant seulement les concepts, attributs et relations relevant de l'enrichissement.

Ensuite, la vérification des exigences d'interopérabilité nécessite de considérer le type d'exigence à prouver. Il est alors indispensable de disposer (1) d'un modèle de processus collaboratif décrit avec le langage utilisée par la technique de vérification et (2) de formaliser les exigences d'interopérabilité.

Dans le cas d'une exigence atemporelle, il s'agit de décrire le modèle du processus collaboratif à l'aide d'un graphe conceptuel unique intégrant tous les concepts et toutes les

[19] Business Process Modeling Notation Version 1.2.

relations présents dans le modèle. Ensuite, il faut décrire chacune des exigences d'interopérabilité sous forme de graphes conceptuels de type contrainte. Enfin, il faut pouvoir utiliser les mécanismes mathématiques de projection présentés au chapitre 2 pour vérifier les exigences.

Dans le cas d'une exigence temporelle, il est nécessaire de disposer d'un modèle comportemental du processus modélisé *i.e.* de la sémantique opérationnelle qui se définit comme l'ensemble des règles d'évolution à mettre en œuvre pour pouvoir interpréter un modèle. Ensuite, il est important de formaliser les exigences temporelles en propriétés *via* un langage formel.

Il est alors nécessaire, dans un premier temps, de **transformer le modèle** de processus collaboratif - grâce à des règles de transformation de modèles - dans des modèles sur lesquels ces techniques peuvent alors être appliquées comme le montre la figure suivante. La technique basée sur les graphes conceptuels est présentée dans (Roque et al, 2009). Dans ce cas, la vérification est effectuée avec l'outil COGITANT (Genest, 2010). La technique de *model checking* est basée sur l'utilisation des Réseaux d'Automates Temporisés proposés comme langage de modélisation comportementale du model checker UPPAAL (Behrmann et al., 2004). Les règles de transformation du modèle de processus, d'une part, vers les graphes conceptuels et, d'autre part, vers les réseaux d'automates temporisés, sont développées avec ATL (Atlas Transformation Language) (ATL, 2005).

Par la suite, il est nécessaire de s'attacher à la **formalisation des exigences** d'interopérabilité soit sous forme de graphes contraintes dans le cas où la technique basée sur les graphes conceptuels avec COGITANT est proposée, soit sous forme de propriétés en utilisant la logique TCTL (*Timed Coputation Tree Logic*) dans le cas où *le model cheking* avec UPPAAL est utilisée.

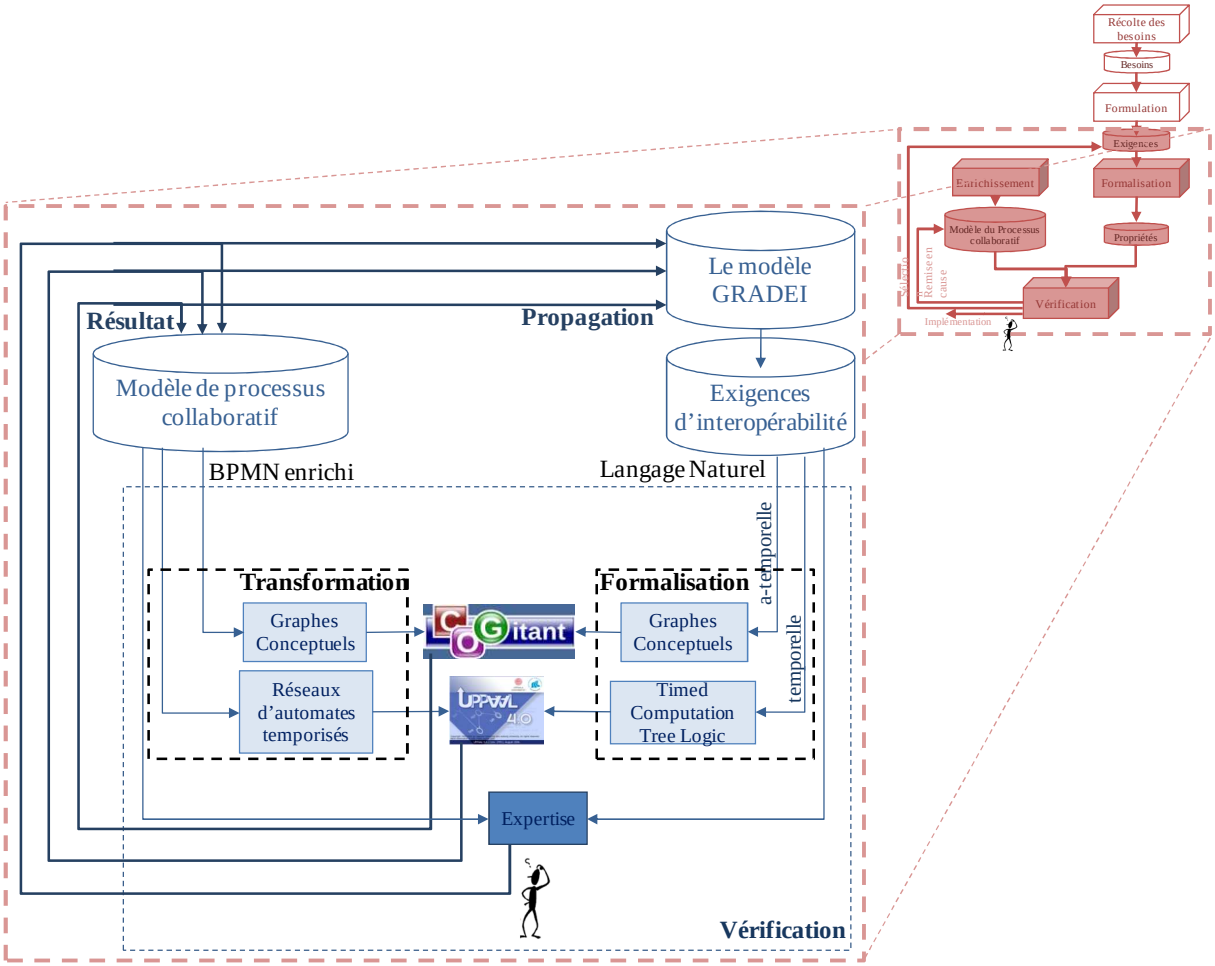


Figure 22. Techniques de vérification employées

Les résultats de vérification obtenus avec les trois techniques employées doit enfin être visible sur le modèle de processus collaboratif modélisé avec le langage BPMN enrichi. Ces résultats sont enfin utilisés pour effectuer la propagation sur le modèle GRADEI et permettre enfin de vérifier des exigences d'interopérabilité abstraites.

3. Modélisation du processus collaboratif : proposition d'enrichissements

Les enrichissements conceptuels présentés et développés ont été initiés dans la cadre du projet CARIONER (Roque *et al.*, 2009) sur la version 1.2 du langage BPMN (BPMN, 2009). D'autres enrichissements propres à nos travaux de recherche, qu'ils soient conceptuels ou opérationnels, viennent compléter à nouveau cette même version. Le résultat est un langage de modélisation de processus collaboratifs appelé dans la suite **BPMN enrichi** dont le méta modèle UML donné en Annexe D.

3.1. Enrichissements conceptuels

Il est d'abord proposé d'ajouter les concepts et les relations nécessaires pour exprimer des exigences d'interopérabilité.

Ils concernent d'abord la prise en compte des ressources utilisées par les tâches comme le montre la Figure 23. En effet, il est important dans un modèle de processus collaboratif de disposer d'une description des ressources impliquées dans le cadre de la collaboration. Pour plus de précision, la nature des ressources a été spécifiée (personne, énergie, software, matériel, unité organisationnelle et machine) ainsi qu'une description de leurs aptitudes et de leurs capacités.

L'enrichissement apporté au niveau des ressources ne contredit en rien la syntaxe ni même la sémantique de la version 1.2 initiale du langage BPMN. Il est à noter que la version 2.0 du langage BPMN, parue en Janvier 2011 (BPMN, 2011), propose de fait une représentation des ressources.

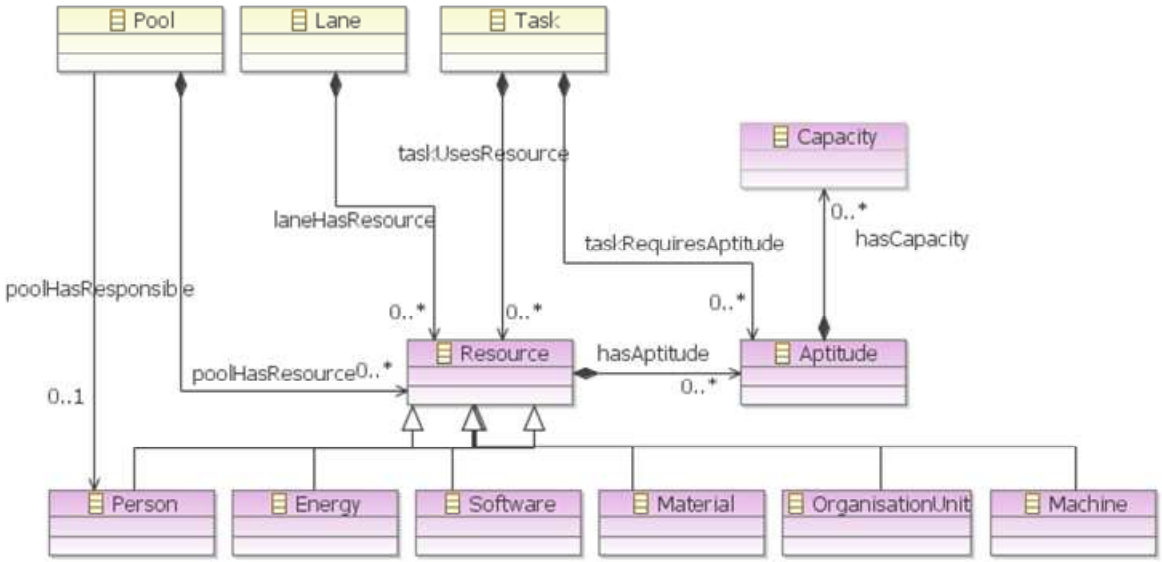


Figure 23. Enrichissement conceptuel du langage BPMN au niveau des ressources

D'autres enrichissements sont ensuite proposés. Ils concernent plus particulièrement la nature des *Data objects* utilisés. En effet, pour préciser la nature du flux transporté dans un *Data object*, il est proposé de préciser la nature de ce flux (*FlowObject*) qui peut être un flux de personne, un flux de matériel, un flux d'énergie ou encore un flux d'information comme le montre la figure suivante.

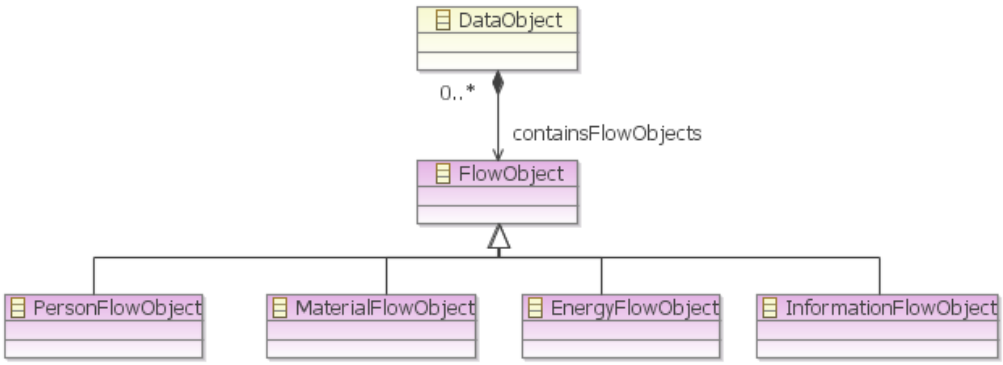


Figure 24. Enrichissement conceptuel du langage BPMN au niveau des flux d'objet

Il est proposé enfin d'ajouter et de typer de nouveaux attributs pour enrichir la description des concepts existant dans le langage BPMN comme proposé par exemple dans la Figure 25.

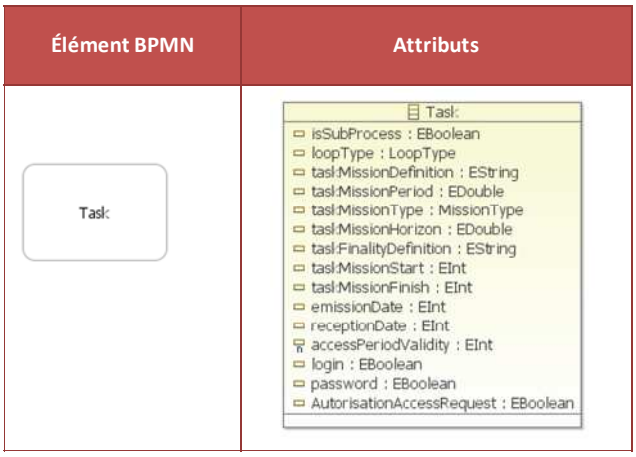


Figure 25. Exemple d'enrichissement sur les attributs (ici pour une tâche)

Nous nous intéressons également dans ce travail de recherche au comportement des éléments du langage BPMN dont la sémantique opérationnelle n'est pas fournie explicitement dans la version 1.2 et sur laquelle ces travaux de recherche sont nécessairement basés.

3.2. Enrichissements opérationnels

Une des techniques de vérification préconisée dans la suite est basée sur le *model checking* qui nécessite de disposer d'un modèle comportemental du système dont on veut vérifier les propriétés. Nous proposons donc de définir une sémantique opérationnelle pour certains éléments de base de BPMN. Plus précisément, les éléments considérés sont l'activité,

la ressource et les *gateways*. Naturellement, cette sémantique opérationnelle est décrite en suivant une logique de modélisation comportementale basée sur des modèles à états.

Il existe dans la littérature plusieurs modèles à états pour représenter une activité. A titre d'exemple, le modèle de Rodde présenté en Figure 26 (a) propose quatre états pour modéliser le comportement d'une activité tout comme le modèle proposé par (Trinquet, 1997) montré en Figure 26 (b).

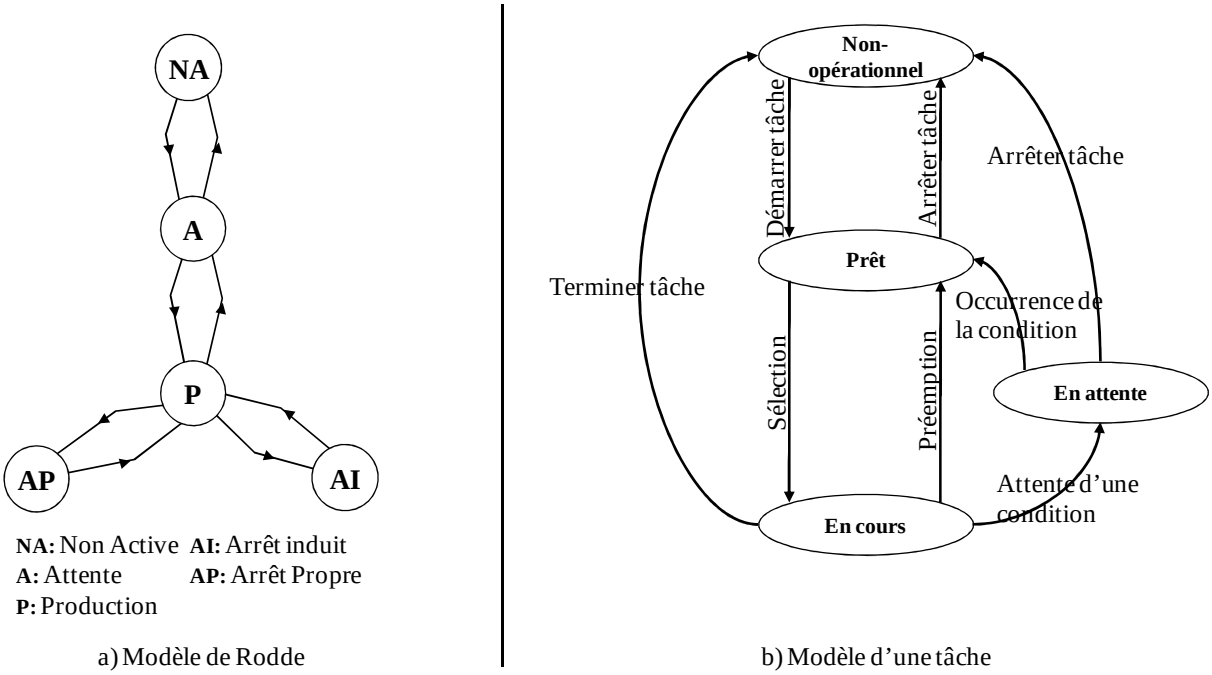


Figure 26. Différents modèles à état pour représenter une activité

Ces deux modèles à états représentent une activité de façon différente mais possèdent certains points communs, notamment, au niveau des états qu'ils proposent. Nous pouvons retrouver les états principaux que peut avoir une activité, c'est-à-dire un état d'arrêt, un état actif ou encore un état d'attente. Dans notre cas, pour un souci d'implémentation, nous considérons les trois états fondamentaux présentés en Figure 27.

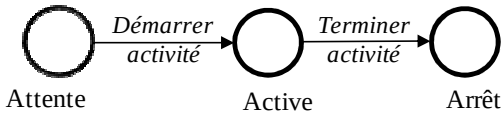


Figure 27. Proposition d'un modèle à états pour une activité

De plus, nous nous intéressons à la sémantique opérationnelle des *gateway*. De ce fait, il est proposé de modéliser un ET logique qui peut représenter le comportement d'un

« *gateway data based parallel* », un OU Exclusif logique qui peut représenter le comportement d'un « *gateway data based exclusive* » et un OU Inclusif qui peut représenter le comportement d'un « *gateway data based inclusive* ». Dans ce sens, nous utilisons la description comportementale de ces éléments fournie dans (BPMN, 2009).

Enfin, nous proposons une sémantique opérationnelle pour l'élément ressource du langage BPMN enrichi comme le montre la Figure 28. Nous supposons qu'une ressource ne possède que deux états. Elle peut être disponible c'est-à-dire en attente d'une sollicitation par une activité ou bien active lorsqu'elle est utilisée par une activité. Enfin, elle repasse à l'état disponible dès que l'activité a fini de l'exploiter.

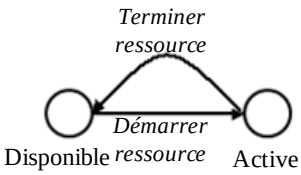


Figure 28. Proposition d'un modèle à états pour l'élément ressource de BPMN enrichi

Par la suite, les exigences d'interopérabilité spécifiées dans le chapitre précédent vont être vérifiées après une traduction adéquate du modèle du processus collaboratif modélisé en BPMN enrichi vers des modèles sur lesquels des techniques de vérification peuvent être mises en œuvre.

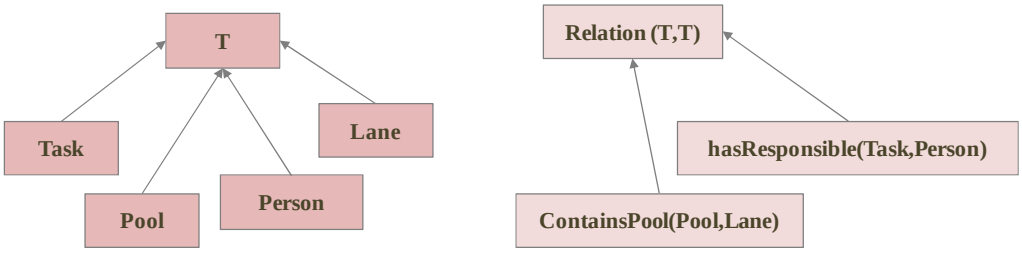
4. Vérification des exigences d'interopérabilité atemporelles de processus collaboratif

L'outil COGITANT choisi pour vérifier les exigences d'interopérabilité atemporelles permet de manipuler les graphes conceptuels et d'appliquer, un mécanisme formel basé sur le principe de la projection de graphe. Ce principe nécessite un graphe conceptuel dans lequel est décrite une *contrainte* reflétant l'exigence d'interopérabilité et un graphe conceptuel qui décrit le modèle du système (un processus collaboratif ici). L'opération de projection consiste à s'assurer que le graphe représentant le modèle respecte bien la contrainte décrite dans le graphe contrainte. Si l'opération de projection échoue, nous considérons que l'exigence n'est pas vérifiée. Les causes peuvent être mises en évidence en analysant le graphe conceptuel résultant de la projection.

Il est donc nécessaire d'obtenir, dans un premier temps, le modèle équivalent à celui du processus collaboratif modélisé en BPMN enrichi. Dans un second temps, les exigences exprimées en langage naturel doivent être réécrites en graphes conceptuels pour pouvoir les vérifier.

4.1. Vers les graphes conceptuels : principes

Le modèle du processus collaboratif est transformé en deux modèles à savoir (1) le « modèle support » et (2) le « modèle de faits ». Le modèle support contient l'ensemble des classes, de leurs attributs et des relations du langage BPMN enrichi. Il est lui-même décomposé en un treillis de concepts et un treillis de relations comme présenté au chapitre 2. Le treillis des concepts représente la hiérarchie entre les concepts et le treillis des relations représente les relations entre ces concepts. Cette transformation inclut également les marqueurs individuels qui décrivent les instances des concepts (Figure 29).



a) Treillis de concepts

b) Treillis de relations

Marqueur	Concept
Gérer production	Task
Mr. T	Person

c) Marqueurs

Figure 29. Exemple d'un modèle support avec treillis de concept, treillis de relations et marqueur individuel

Le modèle de fait représente le modèle du processus collaboratif. C'est un graphe conceptuel construit et conforme aux concepts et aux relations décrits dans les treillis des concepts et des relations. Il utilise soit des marqueurs génériques noté '*' soit des marqueurs individuels présents dans le modèle support.

Figure 30. Exemple de graphe conceptuel

En conséquence, il est nécessaire d'obtenir le modèle de fait correspondant au modèle de processus décrit en BPMN. Pour ce faire, deux transformations sont nécessaires. La Figure 31 représente le principe de la première transformation afin d'obtenir le modèle support (la seconde transformation est basée sur le même principe et n'est pas détaillée).

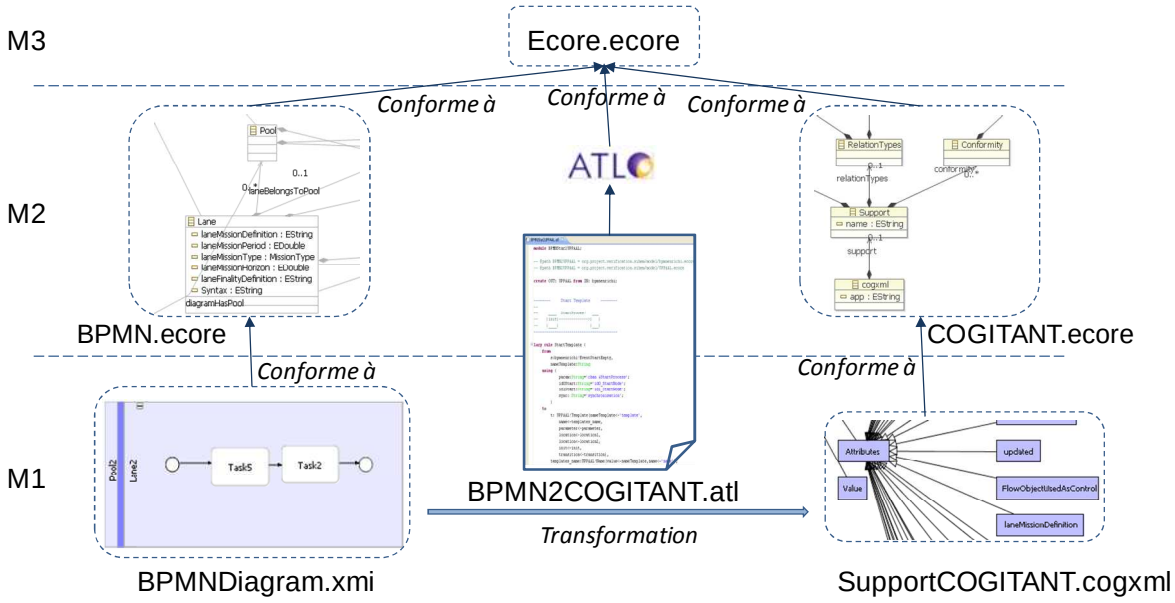


Figure 31. Transformation de BPMN enrichi vers des graphes conceptuels

Les deux transformations permettent de disposer du graphe conceptuel équivalent, selon les hypothèses limitatrices et des hypothèses simplificatrices de transformation, en respectant le format d'entrée de l'outil COGITANT. Ces hypothèses sont les suivantes.

4.2. Vers les graphes conceptuels : hypothèses

Afin d'aboutir à un modèle du processus collaboratif en graphe conceptuel à partir du modèle représenté avec le langage BPMN enrichi qui soit correcte, il est proposé d'adopter des hypothèses limitatrices et des hypothèses simplificatrices.

4.2.1. Hypothèses limitatrices

Une hypothèse limitatrice permet de restreindre la transformation aux seuls concepts et aux seules relations dont nous avons besoin pour faire de la vérification.

Hypothèse 1. Les éléments d'un « *sub process* » sont considérés comme des éléments d'un processus.

Hypothèse 2. Pour effectuer des opérations sur les valeurs des attributs, seuls l'opérateur d'infériorité stricte et l'opérateur de supériorité stricte sont créés au cours de la transformation.

4.2.2. Hypothèses simplificatrices

Une hypothèse simplificatrice permet de faciliter la transformation.

Hypothèse 1. Les attributs des classes du langage BPMN enrichi sont considérés comme des concepts.

Hypothèse 2. Pour faire le lien entre un attribut et sa classe, une relation « *hasAttribute* » est créée au cours de la transformation.

Hypothèse 3. Pour prendre la valeur d'un attribut en compte, un concept « *value* » est créé ainsi qu'une relation « *hasValue* » au cours de la transformation.

4.3. Vers les graphes conceptuels : transformation

Comme nous l'avons mentionné précédemment, la transformation vers des graphes conceptuels a été initiée dans les travaux de (Roque *et al.*, 2009). Dans le cadre de notre thèse, nous avons eu recours à l'automatisation de cette approche. Nous présentons donc ici, les transformations réalisées pour chaque élément de BPMN enrichi.

En se référant à l'hypothèse simplificatrice 1, il est proposé de transformer chaque élément de BPMN enrichi (la classe UML le représentant dans le méta modèle) en un concept. Ensuite, en accord avec l'hypothèse simplificatrice 2, il est proposé de transformer chaque attribut de chaque élément (*i.e.* attribue de chaque classe) également comme un concept. Enfin, les relations entre les classes (dans le méta modèle du langage BPMN enrichi) sont transformées en relations sur le graphe conceptuel comme le montre la figure qui suit.

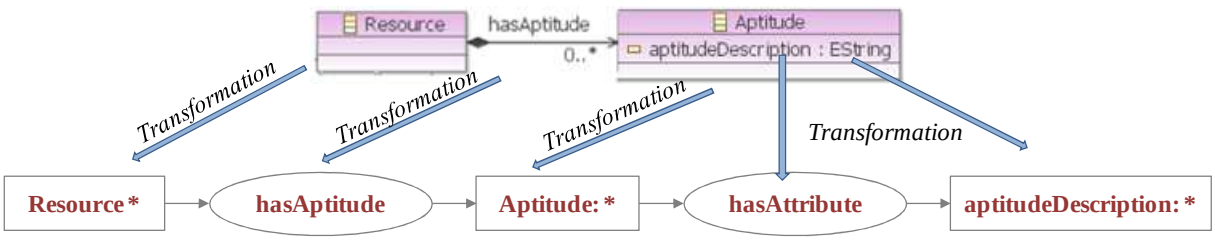


Figure 32. Exemple de transformation de BPMN enrichi vers les graphes conceptuels

De plus, afin de réaliser des comparaisons entre différentes valeurs numériques présentes dans le modèle du processus collaboratif (*e.g.* la capacité sur un modèle de ressource), nous proposons de rajouter deux concepts **isGreaterThan** et **isLessThan** sur lesquels un traitement pour permettre la comparaison de valeurs d'attributs est établie.

4.4. Validation des transformations

La transformation proposée est validée en s'assurant de la cohérence et de la conformité du modèle de processus obtenu en graphe conceptuel. Il s'agit de prouver que chaque entité de modélisation utilisée dans BMPN enrichi est effectivement transformée dans le modèle de graphe conceptuel et de garantir que le modèle du processus collaboratif soit bien représenté et complet. Pour ce faire, il est proposé de recourir à des jeux de test présentés en Annexe E.

4.5. Formalisation des exigences atemporelles en propriétés

Le mécanisme de projection permet de formellement établir la relation entre le modèle du processus (*i.e.* modèle de faits) et une exigence considérée (i.e. un graphe conceptuel représentant l'exigence). Cependant, les exigences atemporelles sont initialement exprimées en langage naturel. L'utilisation de COGITANT nécessite donc, dans un premier temps, d'exprimer ces exigences en graphes conceptuels. Une fois l'exigence réécrite, COGITANT permet de considérer deux types de projections :

- **La contrainte positive.** Elle est composée d'une cause et d'une conclusion. Pour vérifier une contrainte positive, toute projection de la cause sur le graphe du modèle doit permettre de projeter aussi la conclusion sur le graphe du modèle.
- **La contrainte négative.** Ce type de contrainte est un simple graphe conceptuel. Si ce graphe est projeté sur le graphe qui représente le modèle alors la contrainte n'est pas vérifiée.

Par exemple, intéressons-nous à l'exigence de compatibilité décrite dans le chapitre 3 et exprimée comme suit :

$CSO_{Responsabilité}$[20] : « *chaque service possède un responsable clairement défini qui a l'autorité sur le reste de l'équipe* »

Cette exigence peut être réécrite par une contrainte positive comme le montre la Figure 33.

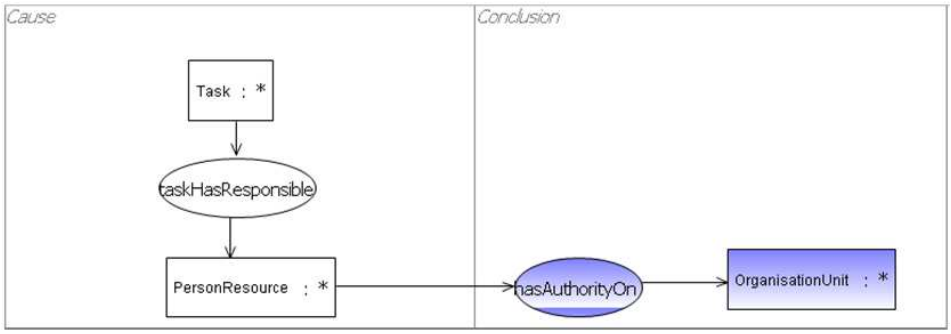


Figure 33. Exemple d'une contrainte positive représentant une exigence de compatibilité

La vérification de cette contrainte en utilisant la projection est réalisée en projetant la cause (concept non coloré sur le côté gauche) sur le modèle de fait obtenu par la transformation. Si la cause est projetée sur le modèle, la conclusion (concept et relation en couleur sur le côté droit) doit être projetée aussi dans le but de respecter l'exigence comme le montre la Figure 34.

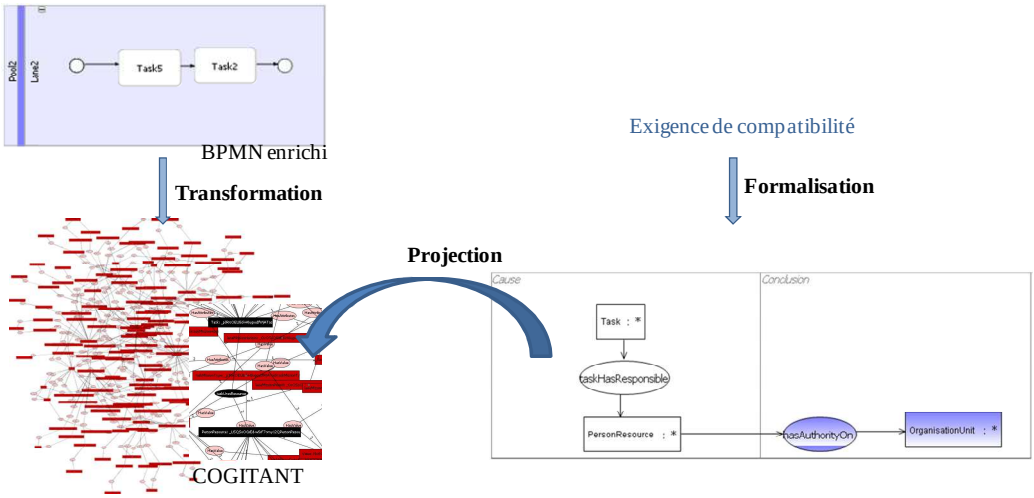


Figure 34. Exemple de vérification avec la projection

[20] Convention adoptée :

- Première lettre : classe d'exigence ;
- Seconde lettre : niveau concerné par l'exigence ;
- Troisième lettre : catégorie concerné par l'exigence ;
- Indice : nom de l'exigence.

4.6. Limites de la technique de vérification

Cette technique de vérification permet d'obtenir un modèle unique de graphe conceptuel représentant le modèle du processus collaboratif en prenant en compte tous les concepts et toutes les relations présents dans le modèle et cela quelle que soit sa complexité. Cependant, l'utilisation des graphes conceptuels n'est pas destinée, à l'origine, à la vérification mais au raisonnement sur la connaissance. Ainsi, la vérification des graphes conceptuels ne s'appuie pas sur les stratégies (Abed, 2009) propres aux algorithmes de vérification dédiés et déployés à cet effet. De plus, des opérations sur des valeurs numériques ne sont pas possibles. En effet, nous avons implémenté des opérateurs d'infériorité et de supériorité qui ne sont utilisables qu'après établissement d'une relation d'ordre de toutes les valeurs.

Il est important de noter qu'une autre limite des graphes conceptuels réside dans la modélisation du temps. Ceci est dû à la manipulation des graphes basés sur la logique du premier ordre et non pas la logique temporelle (Sowa, 1984). En conséquence, il est difficile de décrire des exigences dans lesquels des notions temporelles sont nécessaires. Cette technique de vérification ne permet pas de vérifier des exigences d'interopérabilité temporelles. Nous présentons donc dans ce qui suit la technique de *model checking* adoptée pour vérifier des exigences temporelles.

5. Vérification des exigences d'interopérabilité temporelles

Rappelons que le principe d'un *model checker* est de vérifier des propriétés sur des automates à états et temporisés qui décrivent le comportement d'un système, ici, le modèle du processus collaboratif. Pour réaliser la vérification avec un *model checker*, il est donc nécessaire de réaliser deux étapes. La première étape consiste à définir un ensemble de modèles (*templates*) qui permettent de décrire le comportement du modèle de processus collaboratif à l'aide d'un automate temporisé. Il s'agit dans ce cas de définir les règles qui vont nous permettre de passer du modèle de processus aux automates temporisés *i.e.* de respecter la sémantique opérationnelle fixée précédemment. La seconde étape consiste à réécrire les exigences - qui seront vérifiées – dans le langage adopté par le *model checker*, ici une logique temporelle TCTL (Schnoebelen, 2002).

5.1. Vers les réseaux d'automates temporisés : principes

L'outil choisi, UPPAAL, permet de gérer un modèle de comportement défini sous la forme d'un Réseau d'Automates Temporisés[21] (Behrmann *et al.,* 2004) qui communiquent par une synchronisation binaire utilisant des canaux et une syntaxe du type émission/réception. L'automate possède des états et des transitions. Un état de l'automate peut comporter une condition sur les horloges, appelée **invariant**, qui doit être satisfaite pendant toute la durée passée dans cet état. Une transition est étiquetée par :

- une **garde**, qui exprime une condition sur les valeurs des variables. Cette condition doit généralement être compatible avec l'invariant de l'état origine de la transition et elle doit être satisfaite pour franchir la transition,
- une **synchronisation** des canaux de la forme c ! (émetteur) et c ? (récepteur), l'absence de synchronisation indiquant une action interne de l'automate,
- une remise à zéro (**update**) de certaines horloges et une mise à jour de certaines variables entières.

Les états sont de trois types : ***normal***, ***urgent*** et ***committed*** (Duflot-Kremer, 2010) :

- ***Normal.*** C'est un état où l'on peut attendre jusqu'à ce qu'une transition l'en fasse sortir.
- ***Urgent.*** Un état est dit urgent lorsqu'on ne peut pas attendre dans cet état. Le système ne peut donc faire que des transitions instantanées tant qu'on est dans cet état.
- ***Committed.*** C'est un état urgent et la prochaine transition doit faire sortir de l'état.

La différence principale entre un état dit *urgent* et un état dit *committed* se caractérise dans la priorité de l'état *committed* par rapport à l'état *urgent* comme illustré en Figure 35.

[21] Networks of Timed Automata (NTA).

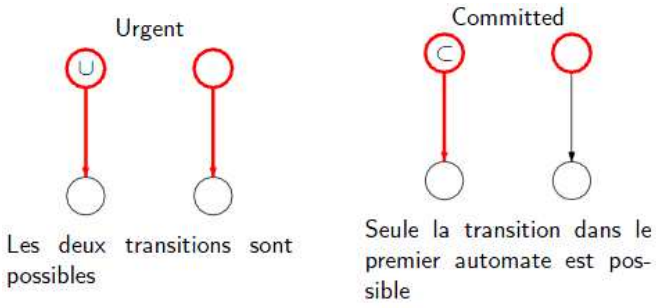


Figure 35. Différence entre un état dit *urgent* et un état dit *committed*

Les modèles à états sur lesquels repose la sémantique opérationnelle des entités de modélisation de BPMN, présentée plus haut, sont donc transformés sous forme d'un réseau d'automates temporisés comme schématisé dans la figure suivante.

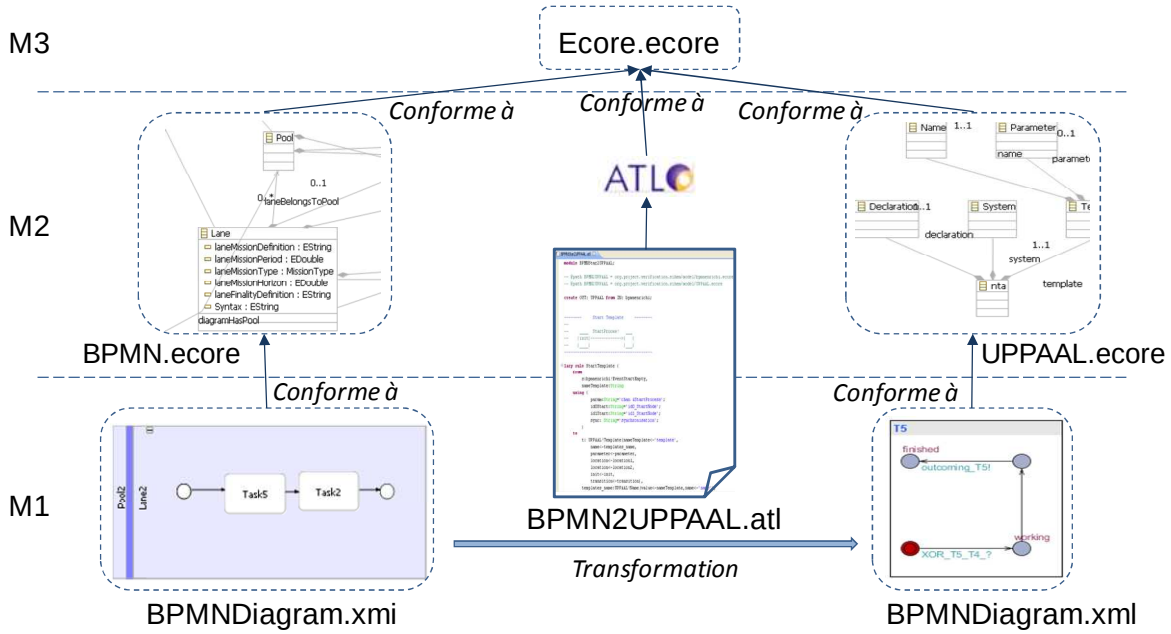


Figure 36. Transformation de BPMN enrichi vers un réseau d'automates temporisés

Chaque classe nécessaire pour la vérification et décrite dans le méta-modèle (*cf.* Annexe D) se traduit par un modèle (*template*) d'automate temporisé. De ce fait, les modèles résultant rassemblent toutes les connaissances décrites dans le modèle du processus collaboratif et permettent d'en établir un modèle de comportement.

5.2. Vers les réseaux d'automates temporisée : hypothèses

Comme pour la transformation précédente, il est proposé d'adopter les hypothèses limitatrices et simplificatrices suivantes.

5.2.1. Hypothèses limitatrices

Hypothèse 1. Tous les « *events* » possèdent la même sémantique opérationnelle. Il n'est donc pas nécessaire de les transformer tous. Nous nous limitons, ici, à la transformation du« *start event empty* » et du « *end event empty* »

Hypothèse 2. Le « *gateway complex* » ne possède pas de sémantique opérationnelle. Il ne sera donc pas pris en compte dans notre transformation.

Hypothèse 3. Une « *pool* » et une « *lane* » n'interviennent pas dans la sémantique opérationnelle du processus collaboratif. De ce fait, leur représentation avec des automates temporisés n'est pas pertinente.

Hypothèse 4. Un « *gateway event based exclusive* » est considéré comme un « *gateway data based exclusive* ».

5.2.2. Hypothèses simplificatrices

Hypothèse 1. Une « *pool* » possède au moins une « *lane* ».

Hypothèse 2. Une « *pool* » possède au moins un « *start event* » et un « *end event* ».

Hypothèse 3. Un « *start event* » ne possède pas de « *sequenceflow* » entrant et possède un seul « *sequenceflow* » sortant.

Hypothèse 4. Un « *end event* » ne possède pas de « *sequenceflow* » sortant et possède un seul « *sequenceflow* » entrant.

Hypothèse 5. Une « *activity* » possède au moins un « *flow* » entrant et un « *flow* » sortant.

Hypothèse 6. Une « *activity* » ne possède pas de boucle de retour (feedback).

Hypothèse 7. Les relations entre les « *pools* » se font par des « *message flows* » entre des « *activities* ».

Hypothèse 8. Un « *gateway* » permettant de faire diverger les flux, nécessite la présence d'un « *gateway* » permettant de les faire converger sauf si l'élément final de chaque branche du « *gateway* » qui les diverge est un « *end event* ».

Hypothèse 9. Un « *gateway* » pour la divergence de flux possède un seul « *sequence flow* » entrant et au moins deux « *sequence flows* » sortants.

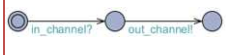
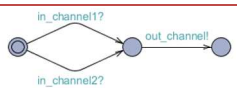
Hypothèse 10. Un « *gateway* » pour la convergence de flux possède au moins deux « *sequence flows* » entrants et un seul « *sequence flow* » sortant.

5.3. Vers les réseaux d'automates temporisés : transformation

Il s'agit, ici, de définir un modèle d'automate temporisé qui représente l'ensemble des états possibles ainsi que les conditions associées aux transitions d'états pour chaque élément de BPMN enrichi, afin de fournir tous les éléments nécessaires pour une vérification sous UPPAAL. En prenant en considération les hypothèses vues ci-dessus, nous nous sommes basés, dans un premier temps, sur les travaux de (Gruhn *et al.*, 2005) qui proposent des modèles en automates temporisés des éléments principaux qui composent un processus. Ces travaux ont permis de développer les règles de transformation qui permettent d'obtenir les modèles proposés, à savoir : le « s*tart event* », le « *end event* », le « *gateway parallel* » (AND), le « *gateway data based exclusive* » (XOR) et un modèle d'activité simple. Cependant, ces travaux ne prennent pas en compte certains éléments résultant de l'enrichissement proposé pour le langage BPMN. En particulier il n'existe pas de modèles qui s'intéressent aux ressources, à la considération de plus d'un flux sur une tâche ou encore au « OU inclusif » comme présentée sur le Tableau 7. Dans un premier temps, et sur la base de la sémantique opérationnelle de ces éléments (*cf.* section 2.3, enrichissements opérationnels), nous avons établi les règles pour leurs transformations.

Il s'agit, ici, de définir un modèle d'automate temporisé qui représente l'ensemble des états possibles ainsi que les conditions associées aux transitions d'états pour chaque élément de BPMN enrichi, afin de fournir tous les éléments nécessaires pour une vérification sous UPPAAL. Les différents flux du modèle sous BPMN enrichi sont représentés comme des synchronisations de canaux afin de modéliser les transitions.

Tableau 7. Modèles existants et modèles réalisées

Elément BPMN enrichi	Modèle NTA existant	Templates	
Start event	✓		
End event	✓		
Task	Modèle de base		
AND (divergent/convergent)	✓		
XOR (divergent/convergent)	✓		
OR (divergent/convergent)	✗	--	--
Resource	✗	--	
Plus d'un flux (entrant/sortant)	✗	--	

Les éléments de BPMN dont les modèles d'automates temporisés sont déjà disponibles dans la littérature (Gruhn *et al.*, 2005) sont directement transformés avec ATL. Dans la suite, nous présentons uniquement les éléments de BPMN enrichi qui ont nécessité au préalable la réalisation (ou l'extension) d'un modèle d'automate temporisé (*template*) avant l'implémentation des transformations, à savoir : l'activité, la ressource, le « *gateway data based inclusive* (OU) » et la prise en compte de « *message flow* ».

5.3.1. Transformation d'une « *Task* » : extension du modèle de base

L'élément « *Task* » dans le langage « BPMN enrichi » décrit un travail qui doit être réalisé par un (ou plusieurs) partenaire au sein d'un un processus collaboratif. Le modèle présenté en Figure 37 est inspiré du modèle de tâche à quatre états présenté dans (Gruhn *et al.*, 2005) et de la sémantique opérationnelle proposée. Dans ce modèle, une tâche se déclenche avec la réception d'une synchronisation de type « *in_channel?* » pour commencer puis elle devient active dans l'état « *Working* ». Enfin, la tâche se termine par l'émission d'une synchronisation de type « *out_channel!* » pour déclencher l'élément suivant dans le modèle du processus collaboratif modélisé par BPMN enrichi.

Figure 37. Modèle d'une « *Task* »

Nous avons vu précédemment que le langage de modélisation est enrichi afin de considérer certains éléments nécessaires à la vérification des exigences d'interopérabilité. Un de ces enrichissements est lié aux ressources qui vont être déployées tout au long du processus collaboratif. En conséquence, les activités auxquelles est allouée une ressource, doivent également posséder un modèle équivalent en automate temporisé. A cet effet, le modèle de la Figure 37 est étendu afin de prendre en considération les ressources utilisées par une tâche pendant le temps d'exécution de cette dernière. Le modèle d'automate temporisé d'une tâche – sollicitant une ressource - que nous proposons est présenté dans la figure suivante.

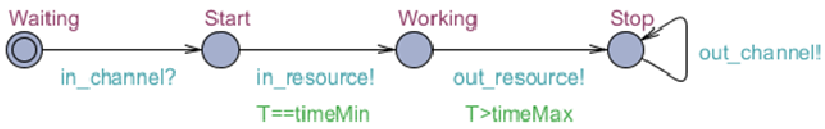


Figure 38. Modèle d'une « *Task* » prenant en compte les ressources sous UPPAAL

5.3.2. Transformation d'une ressource

L'élément « *Resource* » dans BPMN enrichi est modélisé en modèle à état comme nous l'avons présenté en section 2.2. L'automate temporisé correspondant à ce modèle est présenté en Figure 39. Initialement, la ressource est dans l'état disponible (*Available*). Elle passe dans l'état « *active »* quand elle reçoit une synchronisation – provenant d'une tâche - de type « *in_resource?* ». Par la suite la ressource retourne à son état initial en recevant une synchronisation de type « *out_resource ?* » quand la tâche aura terminé de l'utiliser.

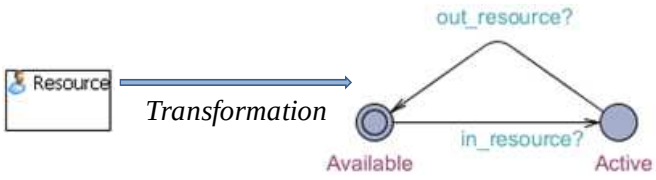
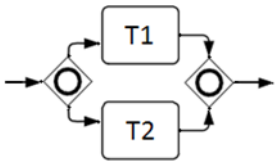


Figure 39. Modèle d'une ressource

5.3.3. Transformation d'un « gateway data based inclusive »

Lors d'une divergence de flux à l'aide de l'élément « *gateway data based inclusive* », un ou plusieurs embranchements sont activés (BPMN, 2009). De ce fait, cet élément possède la sémantique opérationnelle d'un OR logique à ceci près qu'il doit être conçu de telle manière qu'au moins un chemin d'accès soit emprunté (BPMN, 2009). Afin d'effectuer la transformation de cet élément, nous émettons l'hypothèse que le comportement de cet élément est équivalent à la mise en relation entre les opérateurs logiques XOR et AND comme le montre la Figure 40 pour deux activités.



T1	T2	T1 OR T2	T1 AND T2	T1 XOR T2 XOR (T1 AND T2)
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Figure 40. Interprétation de la sémantique opérationnelle du « *gateway data based inclusive* » qui a deux branches (sortie/entrée)

De ce fait, afin d'effectuer la transformation du « *gateway data based inclusive* » (OR), nous nous basons sur les modèles des éléments « *gateway data based exclusive* » (XOR) et « *gateway parallel* » (AND). Une mise en relation entre ces deux modèles est donc nécessaire pour modéliser le comportement de l'élément « *gateway data based inclusive* » comme le montre la Figure 41.

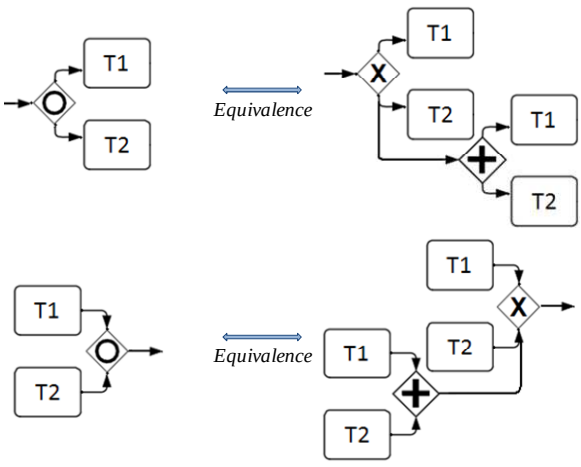


Figure 41. Comportement d'un « *gateway data based inclusive* » pour deux entrées

En généralisant, pour modéliser un « *gateway data based inclusive* » qui diverge les flux et qui a « N » branches de sortie, il est normalement nécessaire de représenter « 2^N » cas différents.

Cependant, dans notre cas, au moins un chemin d'accès doit être pris en compte. Nous ne considérons donc pas le cas où « *gateway data based inclusive* » ne permet pas l'avancement du processus (tous les états sont à 0). Ainsi, le nombre de cas possibles et nécessaires à modéliser est égal à 2^N-1. Si l'on considère un « *gateway data based inclusive* » à deux branches de sortie, nous considérons donc « 2^4-1 = 3 » cas possibles d'évolution. D'après la Figure 38, le « *gateway data based inclusive* » enclenche, soit l'activité T1, soit l'activité T2, soient les deux en même temps. De ce fait, nous modélisons le « *gateway data based inclusive* » (OR) par un « *gateway deta based exclusive* » (XOR) qui a 3 branches de sorties, (« T1 », « T2 », « T1 AND T2 ») et un « *gateway parallel* » (AND) qui a 2 branches de sorties (« T1 » et « T2 »).

Généralisons maintenant le comportement d'un « *gateway data based inclusive* ». Soit N le nombre de branches de sortie du « *gateway data based inclusive* ». Nous obtenons ;

- 2^N-1, le nombre de branches de sorties à prendre en compte pour le « *gateway data based exclusive* » (XOR)
- C_N^p , le nombre de « gateway parallel »(AND) où p est le nombre de branches de sortie du « gateway parallel »(AND), $p \in [2, N]$

Le Tableau 8 donne la construction d'un « *gateway data based inclusive* » à partir d'un « *gateway data based exclusive* » et d'un « *gateway parallel* ».

Tableau 8. Formalisation des sorties du OR logique

N branches de sortie / Opérateurs	2	3	4	N
XOR	3	7	15	$2^N - 1$
AND à 2 branches de sortie	1	3	6	C_N^2
AND à 3 branches de sortie	--	1	4	C_N^3
AND à 4 branches de sortie	--	--	1	C_N^4
AND à i branches de sortie	--	--	--	C_N^i
AND à N branches de sortie	--	--	--	C_N^N

Enfin, nous pouvons noter que cette généralisation est utilisée de la même façon pour la transformation d'un « *gateway data based inclusive* » qui converge des flux.

5.3.4. Transformation avec prise en compte de plus d'un flux

Dans le langage BPMN enrichi, une activité peut, par exemple, envoyer ou recevoir deux flux, un « *sequence flow* » et un « *message flow* ». Dans ce cas-là, le modèle d'activité sous UPPAAL doit pouvoir envoyer deux synchronisations différentes de canaux. Le modèle à états présenté précédemment pour représenter une activité ne permet pas la prise en compte de deux synchronisations en même temps. Ainsi, nous proposons de simuler l'existence de deux « *gateway parallel* » l'un pour la convergence des flux et l'autre pour la divergence. Ces « *gateway parallel* » posséderont - après transformation - autant de branches entrantes ou sortantes qu'il y a de flux entrants ou sortants d'une tâche.

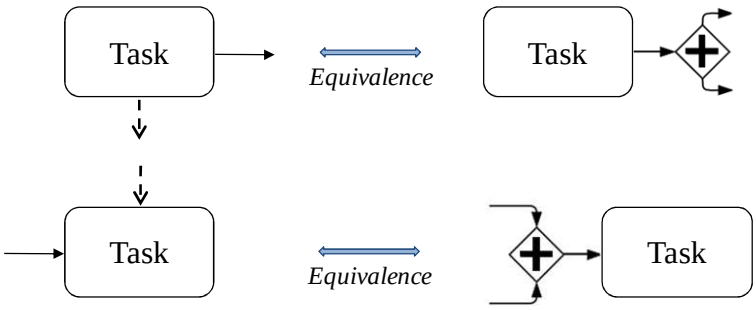


Figure 42. Comportement pour un envoi et une réception de deux flux

5.4. Validation des transformations

Les transformations que nous avons présentées ont été établies sur une équivalence entre le comportement des entités de BPMN enrichi - décrit par la sémantique comportementale proposée plus haut - et un modèle d'automate temporisé. Les transformations effectuées doivent donc assurer la préservation de la sémantique du comportement de la version enrichie de BPMN. Afin de s'assurer de la pertinence de notre transformation et de la bonne représentation de chaque élément de BPMN enrichi en modèle d'automates temporisés, il est proposé d'avoir recours à des jeux de test présentés en Annexe E. En effet, le *model checker* UPPAAL possède un éditeur et un simulateur. Le simulateur nous offre la possibilité d'effectuer les tests correspondants à ces jeux d'essai afin de voir si le modèle équivalent de chaque entité sous BPMN enrichi représente bien le comportement attendu de ce dernier.

5.5. Formalisation des exigences temporelles en propriétés

Les exigences d'interopérabilité écrites en langage naturel sont manuellement réécrites en utilisant un fragment de la logique TCTL en respectant la syntaxe suivante :

- E <> p : il existe un chemin le long duquel p est vrai un jour.
- E [] p : il existe un chemin le long duquel p est toujours vrai.
- A <> p : le long de tout chemin p est vrai un jour.
- A [] p : le long de tout chemin p est toujours vrai.
- p $\rightarrow$ q : quand p est vrai, alors q est forcément vrai un jour.

Avec p et q des propriétés, <> et [] des opérateurs temporels, E et A les quantificateurs de chemin.

Considérons l'exigence temporelle « $ISO_{TempsDisponibilité}$ » décrite telle que :

$ISO_{TempsDisponibilité}$[22] : « *La ressource allouée est active sur toute la durée d'exécution (5<T<10) de l'activité (état Working)* ».

[22] Convention adoptée :
- Première lettre : classe d'exigence ;
- Seconde lettre : niveau concerné par l'exigence ;
- Troisième lettre : catégorie concerné par l'exigence ;
- Indice : nom de l'exigence.

Cette exigence doit permettre de s'assurer qu'une ressource est bien utilisée durant toute la durée de l'activité à laquelle elle est allouée. Pour vérifier cette exigence, il est nécessaire de la formaliser par une propriété utilisant la logique TCTL telle que :

E <> Resource.Active and Activity.Working and T> 5 and T <10

Cette propriété indique qu'il existe un chemin le long duquel la ressource est dans l'état « *Active* » et l'activité est dans l'état « *Working* » entre 5 et 10 unités de temps. Cette propriété peut être vérifiée sur le modèle d'activité présenté en Figure 38. Ensuite, le *model checker* répondra par vrai ou faux selon que l'exigence est vérifiée ou non.

5.6. Limites de la technique de vérification

Le *model checking* est basé sur l'exploration de tous les chemins ou comportements atteignables d'un système. Il permet en outre de prendre en considération l'aspect temporel. Cependant, son utilisation et par là même, la mise en œuvre de *model checker* tels que UPPAAL, est limité par le phénomène d'explosion combinatoire du nombre d'états et donc de chemins à parcourir qui peut apparaitre. Cela entraine au mieux une lenteur de la vérification et dans le cas le plus défavorable, l'impossibilité de disposer d'une preuve tangible.

6. Mécanismes de propagation à travers la structure du modèle GRADEI

Nous avons vu que les exigences vérifiables avec les deux techniques présentées précédemment sont les exigences du plus bas niveau donc les plus précises présentes sur le modèle GRADEI. Nous avons également vu que des exigences non vérifiables avec ces deux techniques peuvent être vérifiées par expertise. Dans l'optique de vérifier les exigences abstraites présentes sur le modèle GRADEI et essentiellement d'aboutir à la vérification de l'exigence la plus abstraite (« interopérabilité » qui est la racine du modèle GRADEI), nous avons proposé dans le chapitre 3 de recourir à un mécanisme de propagation.

A cette fin, la fonction logique (θc) est utilisée afin de lier une exigence abstraite aux exigences qui la décomposent. Cette fonction logique utilise indifféremment les opérateurs logiques (AND, OR et NOT) afin de laisser l'utilisateur le type de décomposition qu'il souhaite.

Les exigences les plus précises sont vérifiées par l'un des outils proposés (UPPAAL ou COGITANT) ou par expertise. En conséquence, il est nécessaire de propager ces résultats en interprétant les fonctions θc de chaque nœud intermédiaire jusqu'au nœud racine du graphe.

Il est proposé sur l'exemple de la Figure 43 de vérifier l'exigence de compatibilité $CSO_{Responsabilité}$ avec l'outil COGITANT et l'exigence d'interopération $ISO_{TempsDisponibilité}$ avec l'outil UPPAAL. Ces exigences sont donc formalisées en propriétés et sont représentées, sur le modèle GRADEI par les exigences les plus concrètes (extrême gauche). Enfin, l'expert indiquera la satisfaction (en vert) ou non (en rouge) des exigences non vérifiables formellement. Dans cet exemple, l'opérateur logique utilisé, sur tout le modèle, pour lier les exigences précises aux exigences abstraites est l'opérateur AND.

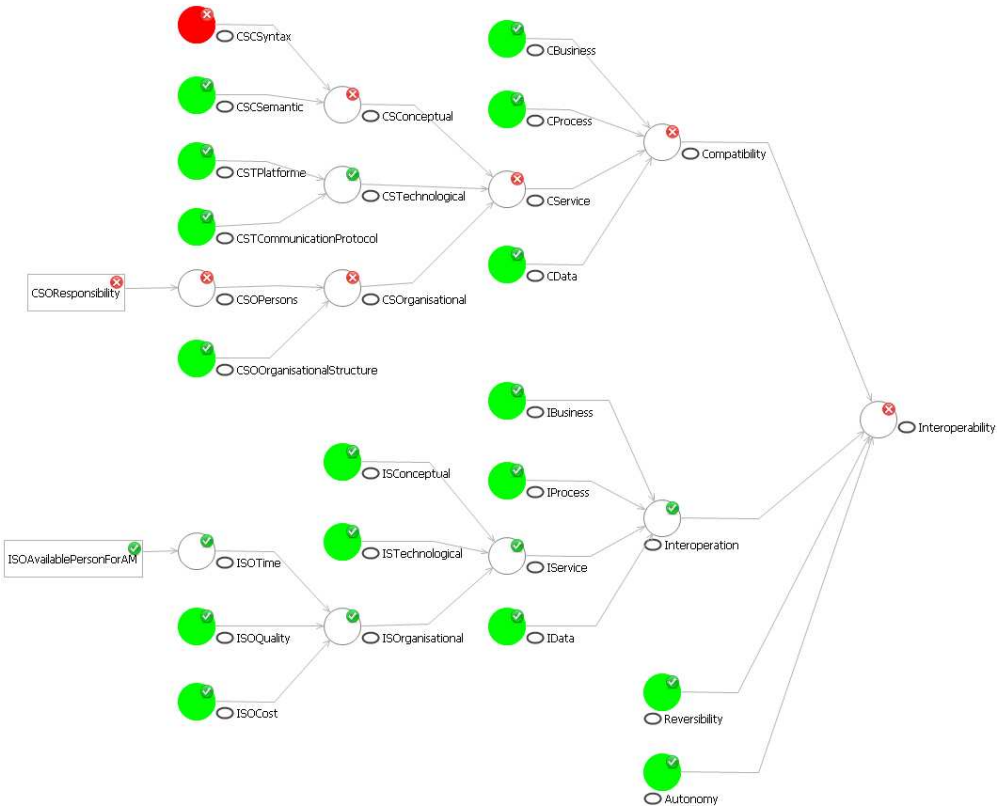


Figure 43. Propagation avec le modèle GRADEI

Le résultat de la vérification montre que l'exigence de compatibilité $CSO_{Responsabilité}$ n'est pas satisfaite, et l'exigence d'interopération $ISO_{TempsDisponibilité}$ est satisfaite. De plus, l'expertise indique que l'exigence $CSC_{Syntaxe}$ n'est pas satisfaite. Enfin la mise en oeuvre des mécanismes de propagation nous montre que l'exigence « interopérabilité » n'est pas satisfaite.

7. Conclusion

Nous nous sommes intéressés, dans ce chapitre, aux techniques de vérification que nous adoptons dans un premier temps pour aider à la détection de problèmes d'interopérabilité sur un modèle de processus collaboratif et aux besoins que l'usage de ces techniques a entrainé.

Nous avons montré que le modèle du processus collaboratif doit posséder les concepts nécessaires pour représenter tous les concepts d'interopérabilité. Pour cela, nous avons établi que le langage de modélisation doit être enrichi et nous avons proposé des enrichissements conceptuels et sémantiques pour aboutir au langage BPMN enrichi.

Par la suite, la vérification des exigences d'interopérabilité utilise des techniques de vérification complémentaires. La première technique utilise des mécanismes mathématiques appliqués sur les graphes conceptuels pour vérifier des exigences d'interopérabilité atemporelles. Le seconde technique est une technique de vérification formelle basée sur le *model checking* et permet de vérifier des propriétés temporelles sur des automates temporisés. Cette technique est utilisée pour la vérification des exigences temporelles. Enfin, les exigences non vérifiables sur un modèle ou non formalisables sont destinées à l'expertise.

Nous avons montré que l'utilisation des deux premières techniques requiert deux phases pour effectuer la vérification. La première phase consiste à transformer le modèle du processus collaboratif modélisé en des modèles équivalents sur les quels la vérification est effectuée. Cette transformation est réalisée pour obtenir, d'une part, le modèle du processus collaboratif en graphe conceptuel pour la vérification des exigences atemporelle. D'autre part, la transformation est effectuée pour obtenir le modèle du comportement en réseau d'automates temporisés pour la vérification des exigences d'interopérabilité temporelles.

La deuxième phase consiste à formaliser les exigences d'interopérabilité en propriétés pour permettre leur vérification. De ce fait, les exigences atemporelles sont formalisées en contraintes avec l'utilisation de graphes conceptuels pour pouvoir employer le mécanisme de projection et vérifier l'exigence sous COGITANT. Les exigences temporelles sont formalisées en propriétés avec un fragment de la logique TCTL utilisée par le *model checker* UPPAAL.

L'utilisation de ces deux techniques permet donc de donner une indication sur la présence ou non de problèmes d'interopérabilité à un niveau local. Notre approche propose, par la suite, de détecter des problèmes à un niveau global grâce à l'utilisation de mécanismes de propagation sur le modèle GRADEI.

Afin de montrer l'application de notre approche d'anticipation de problèmes d'interopérabilité, un cas applicatif est présenté dans le chapitre suivant.

Chapitre 5

Application de l’approche anticipative de détection de problèmes d’interopérabilité

Table des matières du chapitre 5

1. Introduction

Afin d'illustrer les concepts de l'approche proposée dans ces travaux ce chapitre présente deux cas d'application dans lesquels la recherche de problèmes d'interopérabilité revêt un enjeu particulier. Le premier cas d'application est un processus interne d'un organisme de recherche pour l'organisation d'une conférence nationale. Le second cas d'application est un processus public, vu à un haut niveau d'abstraction, entre plusieurs entreprises pour la conception et la mise en production d'un véhicule.

Dans ces deux cas, nous nous intéressons à la vérification de plusieurs exigences de compatibilité, d'interopération, réversibilité et d'autonomie extraites du référentiel d'exigences donné en annexe C et détaillées dans la suite.

2. Exigences d'interopérabilité

Nous retenons trois **exigences de compatibilité**, relatives à la catégorie organisationnelle. Elles sont exprimées comme suit :

- $CSO_{Responsabilité}$[23] : « *Chaque service possède un responsable clairement identifié* ».
- $CSO_{Aptitude}$: « *Une ressource possède l'aptitude requise par l'activité la sollicitant* ».
- $CSO_{Autorisation}$: « *L'émetteur possède les autorisations nécessaires pour échanger avec le récepteur* ».

Ces exigences de compatibilité possèdent un caractère important pour des partenaires pour assurer la bonne marche du processus collaboratif lors de son exécution. En effet, l'exigence $CSO_{Responsabilité}$ reflète le besoin qu'ont les partenaires à connaître le responsable du service pour éviter de perdre du temps au cours des échanges. L'exigence $CSO_{Aptitude}$ exprime le besoin des partenaires vis-à-vis des aptitudes des ressources employées par les différents

[23] Convention adoptée :
- Première lettre : classe d'exigence ;
- Seconde lettre : niveau concerné par l'exigence ;
- Troisième lettre : catégorie concerné par l'exigence ;
- Indice : nom de l'exigence.

services. Enfin, l'exigence $CSO_{Autorisation}$ représente le besoin d'avoir les autorisations nécessaires pour chaque échange ou partage entre partenaires.

Les **exigences d'interopération** retenues pour cette étude sont liées à la barrière organisationnelle et exprimées telles que :

- $ISO_{DisponibilitéRessource}$: « *Chaque service utilise la ressource nécessaire pendant le temps d'exécution de ce service.* »
- $ISO_{TempsEchange}$: « *Le temps d'échange entre les services est strictement inférieur à n unités de temps (UT).* »
- $ISO_{AcccuséRéception}$: « *Le récepteur accuse réception à l'émetteur.* »

La pertinence de ces exigences d'interopération s'illustre comme suit. Lors de la collaboration, le non respect de l'exigence $ISO_{DisponibilitéRessource}$ peut entrainer des retards et des dysfonctionnements au niveau du processus collaboratif. De même, l'exigence $ISO_{TempsEchange}$ reflète le besoin des partenaires à respecter un temps d'échange défini entre leurs services pour limiter l'impact de l'interopération sur les délais de la collaboration. Enfin, l'exigence $ISO_{AcccuséRéception}$ permet de s'assurer que tout échange est contrôlé d'une certaine manière et en temps réel.

Nous nous intéressons, également, à une exigence de réversibilité qui concerne la barrière organisationnelle. Cette exigence représente la capacité qu'ont les partenaires à retrouver leurs performances au niveau des ressources employées dans le processus collaboratif. Elle s'exprime comme suit :

$RSO_{PerformanceRessource}$: « *Les ressources employées dans le processus collaboratif possèdent les mêmes performances - tolérances comprises – qu'avant la collaboration.* »

Enfin, une exigence d'autonomie qui concerne la barrière organisationnelle est retenue pour l'étude. Cette exigence a une importance pour les partenaires dans le cas où une ressource, d'habitude employée par un service dans le processus collaboratif privé, est employée par un autre service dans le processus collaboratif public. De fait, l'unité organisationnelle à laquelle appartient cette ressource doit posséder la capacité de la remplacer pour ne pas freiner le travail interne de l'entreprise. Cette exigence est exprimée comme suit :

$ASO_{Ressource}$: « *Un service peut remplacer une ressource utilisée par un autre service du processus collaboratif public* ».

L'ensemble des exigences utilisées est représenté sur le modèle GRADEI comme le montre la figure suivante.

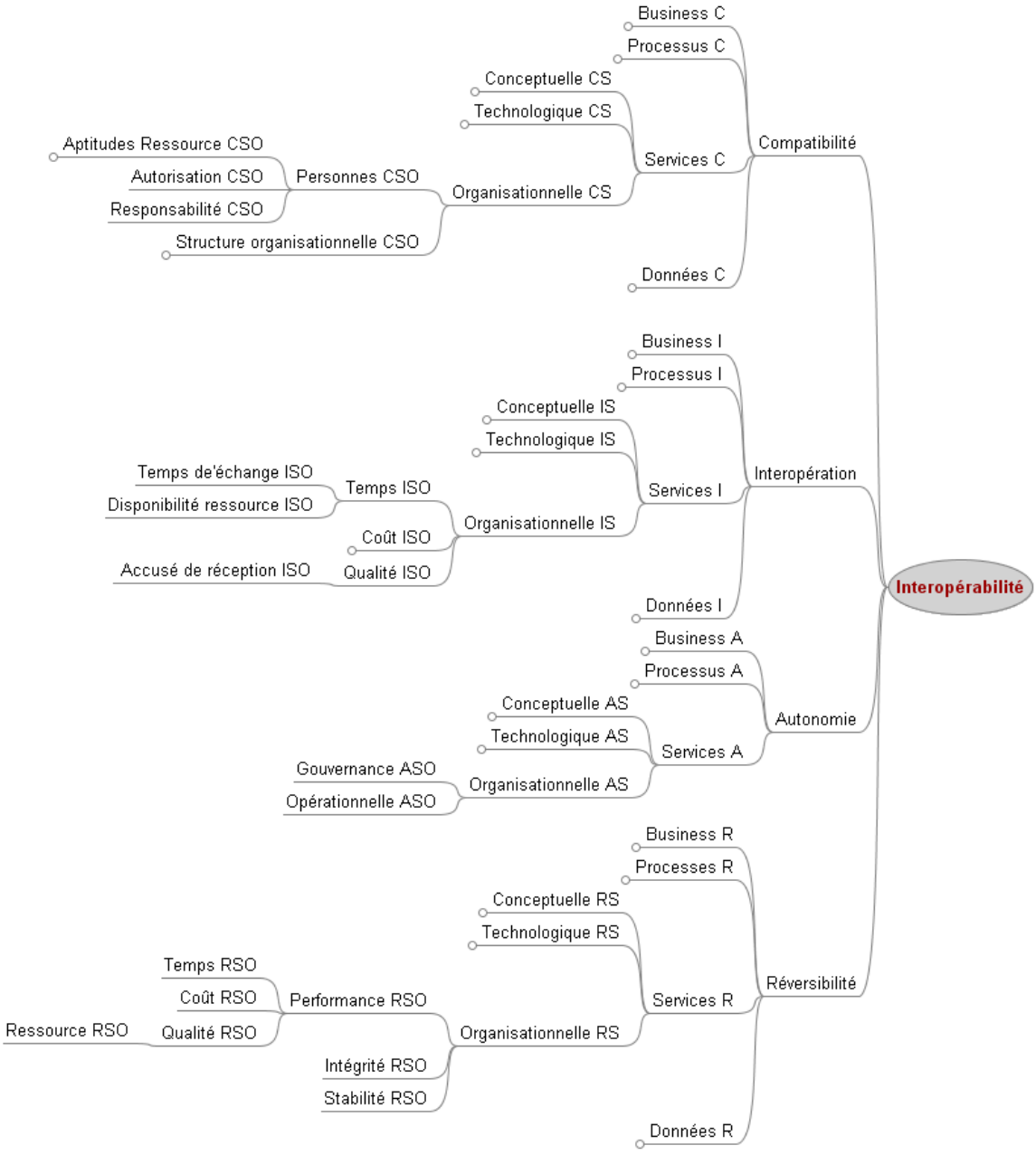


Figure 44. Représentation des exigences de l'étude sur le modèle GRADEI

3. Formalisation et vérification des exigences d'interopérabilité

Pour détecter d'éventuels problèmes d'interopérabilité, pour les deux cas d'application proposés, nous devons, dans un premier temps, formaliser ces exigences par des propriétés, et dans un second temps les vérifier avec les techniques de vérification adéquate.

Comme présenté dans le chapitre 4, le choix de la technique de vérification dépend du caractère temporel de l'exigence. En effet, si l'exigence est atemporelle, elle sera formalisée en graphe conceptuel et vérifiée avec l'outil COGITANT. Dans le cas où l'exigence est temporelle, elle sera formalisée par la logique TCTL et vérifiée avec l'outil UPPAAL sur un modèle comportemental considéré comme représentant le comportement du processus collaboratif.

De plus, les exigences non formalisables ou non vérifiables sur le modèle du processus collaboratif seront destinées à une vérification par expertise.

3.1. Formalisation et vérification des exigences de compatibilité

Les exigences de compatibilité à vérifier ont un aspect atemporel. En conséquence, nous avons recours à la technique de vérification utilisant les graphes conceptuels. De ce fait, les exigences seront formalisées sous forme de contraintes avec des graphes conceptuels.

Afin de procéder à la vérification de l'exigence $CSO_{Responsabilité}$, nous l'avons formalisée avec la contrainte positive de la Figure 45. Cette contrainte exprime le fait que chaque service (tâche) possède un responsable. Un concept de référence, est un concept sur lequel nous nous basons pour voir l'insatisfaction d'une exigence sur le modèle de processus collaboratif. Ce concept de référence est indiqué par un point bleu sur le concept.

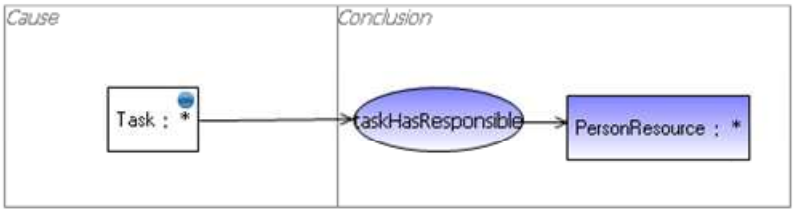


Figure 45. Formalisation de l'exigence $CSO_{Responsabilité}$ avec une contrainte positive

Cependant, cette contrainte ne fournit aucune indication sur l'identification de ce responsable. En conséquence, il est proposé d'avoir plus d'information sur son nom, son

numéro de téléphone ou son email par exemple. De ce fait, pour une détection plus précise, l'exigence $CSO_{Responsabilité}$ peut être décomposée avec les trois exigences suivantes.

- $CSO_{ResponsabilitéNom}$[24] : « *Un responsable de service possède un nom* ».
- $CSO_{ResponsabilitéPhone}$: « *Un responsable de service possède un numéro de téléphone* ».
- $CSO_{ResponsabilitéMail}$: « *Un responsable de service possède un email* ».

Ces exigences donnent plus d'indication sur l'identification du responsable. Elles sont formalisées par trois contraintes négatives comme le montre la Figure 46.

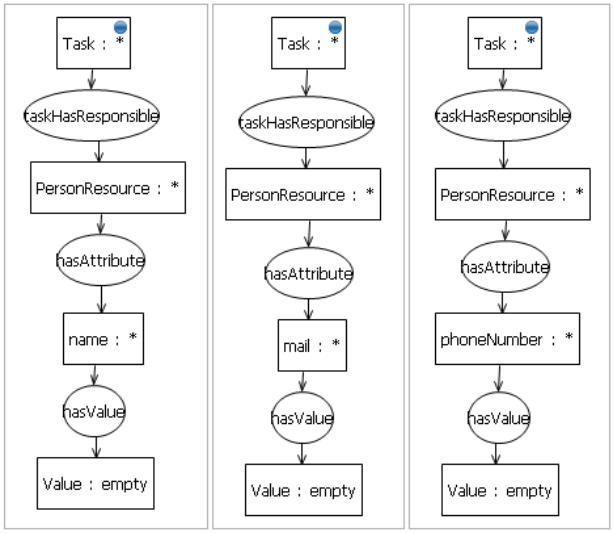


Figure 46. Formalisation des exigences décomposant l'exigence $CSO_{Responsabilité}$ avec trois contraintes

L'exigence $CSO_{Aptitude}$ est formalisée par la contrainte positive en Figure 47. Cette contrainte se traduit comme suit : pour chaque service (tâche), les ressources employées par cette tâche doivent posséder la ou les aptitudes requises par celle-ci.

[24] Convention adoptée :
- Première lettre : classe d'exigence ;
- Seconde lettre : niveau concerné par l'exigence ;
- Troisième lettre : catégorie concerné par l'exigence ;
- Indice : nom de l'exigence.

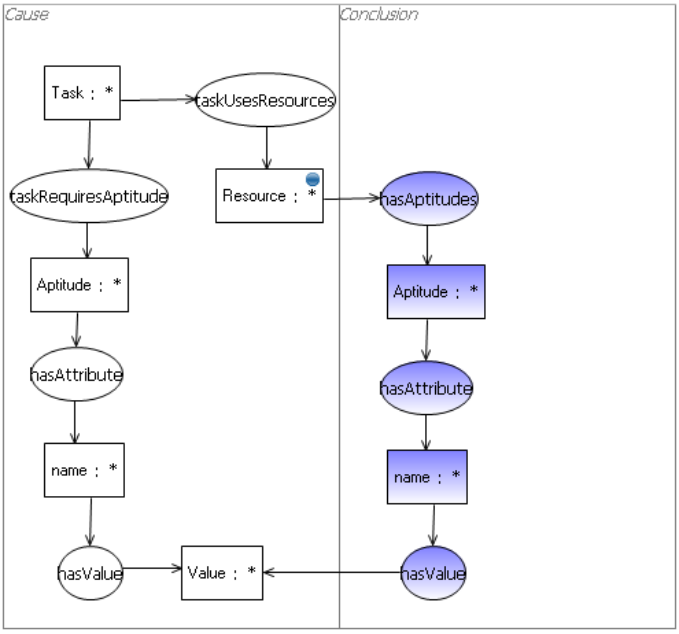


Figure 47. Formalisation de l'exigence $CSO_{Aptitude}$ en contrainte positive

Pour accomplir la vérification de l'exigence $CSO_{Autorisation}$, cette dernière est formalisée par la contrainte positive en Figure 48. Cette contrainte reflète le fait que chaque service (tâche) qui est en interaction avec un autre service (tâche) qui requiert une autorisation, doit avoir les autorisations d'accès nécessaires pour pouvoir les diffuser à ses ressources pour effectuer les échanges et les partages entre les deux.

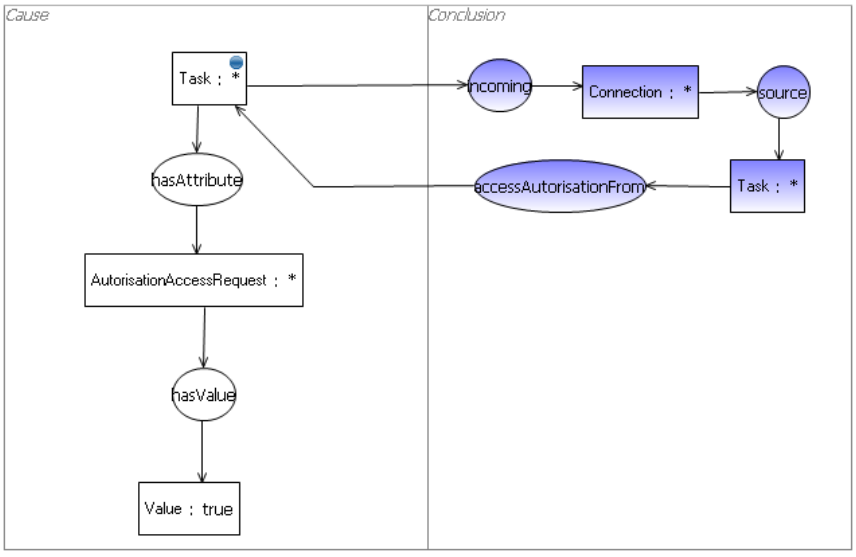


Figure 48. Formalisation de l'exigence $CSO_{Autorisation}$ avec une contrainte positive

Cependant, la notion d'autorisation peut concerner différents aspect. Cela peut être la possession d'un login, d'un mot de passe ou encore une période de validité pour les accès.

Ainsi, pour aller plus en précision pour la détection de problème, trois autres exigences peuvent compléter l'exigence précédente.

- $CSO_{AutorisationLogin}$: « *L'émetteur possède un login pour échanger avec le récepteur.* »
- $CSO_{AutorisationMotPasse}$: « *L'émetteur possède un mot de passe pour échanger avec le récepteur.* »
- $CSO_{AutorisationValidité}$: « *La validité temporelle des autorisations est définie.* »

Ces exigences sont formalisées par deux contraintes positives et une contrainte négative schématisées en Figure 49.

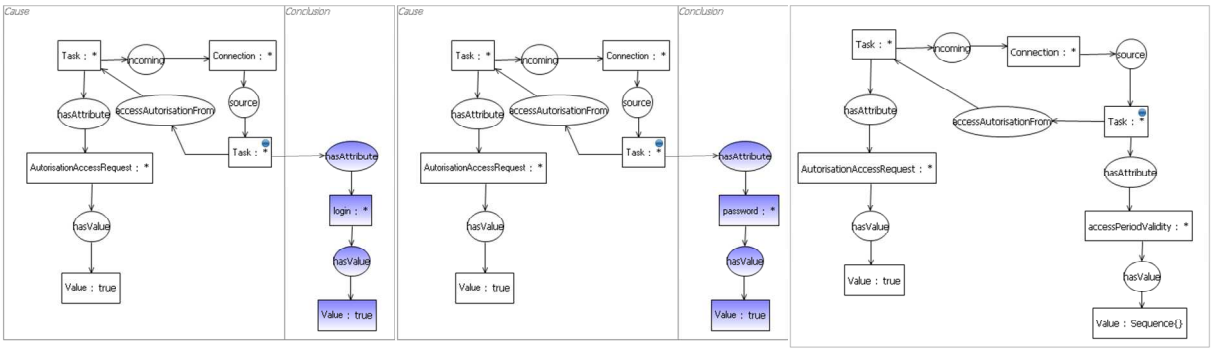


Figure 49. Formalisation des exigences décomposant l'exigence $CSO_{Autorisation}$ avec trois contraintes

La vérification de ces exigences de compatibilité se fera sur un modèle de processus collaboratif par le biais du mécanisme de projection des contraintes sur ce modèle.

3.2. Formalisation et vérification d'exigences d'interopération

L'exigence d'interopération à caractère atemporel est formalisée avec un graphe conceptuel pour pouvoir la vérifier par projection. Les exigences d'interopération qui possèdent un caractère temporel vont être vérifiées avec le *model checker* UPPAAL. Elles sont donc formalisées avec un fragment de la logique TCTL.

L'exigence $ISO_{DisponibilitéRessource}$ est une exigence temporelle. Elle est formalisée sous forme de propriété avec l'expression suivante :

« *E<> Task.Working and Ressource.Active and T<Task.timeMin and T>Task.timeMax* »

Cette propriété traduit l'existence d'un chemin où une tâche exploite une ressource pendant sa durée d'exécution.

L'exigence $ISO_{TempsEchange}$ possède un caractère temporel. La formalisation de cette exigence se fait par l'expression suivant :

« *StartNode.Start → forall (i :nbrTask) Reception[i]-Emission[i] <=nUT* »

Cette propriété exprime que dès le commencement du processus collaboratif, le temps d'échange (Réception – Emission) ne dépasse pas 5 unités de temps entre toutes les tâches qui se succèdent.

La vérification de ces deux exigences d'interopération se fait sur un modèle de processus collaboratif transformé en réseau d'automates temporisés.

L'exigence $ISO_{AcccuséRéception}$ est une exigence atemporelle. En conséquence, sa vérification peut être effectuée avec l'outil COGITANT. Elle est formalisée en contrainte positive présentée sur la figure suivante.

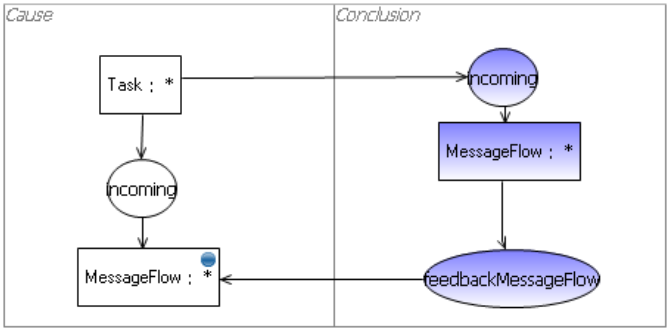


Figure 50. Formalisation de l'exigence d'interopération $ISO_{AcccuséRéception}$

3.3. Formalisation et vérification de l'exigence de réversibilité et l'exigence d'autonomie

Il est impossible de vérifier l'exigence de réversibilité sur le modèle de processus collaboratif. En effet, pour comparer les performances avant et après le partenariat, il est nécessaire de disposer du modèle de processus collaboratif privé avant et après la collaboration. Puisque nous ne disposons pas de ce modèle, nous ne pouvons vérifier cette exigence que par expertise et *a posteriori*.

Pour vérifier l'exigence d'autonomie, il est nécessaire de disposer, par hypothèse, du modèle de processus collaboratif ainsi que d'un modèle de processus privé. En effet, lorsqu'une entreprise est impliquée dans un processus collaboratif public, cela peut impacter les performances de son processus collaboratif privé. Ne disposant pas du processus collaboratif privé, cette exigence sera également destinée à être vérifiée par expertise.

Pour les deux cas d'application choisis, nous réalisons les étapes suivantes :

1. **Modélisation** du processus collaboratif avec le langage BPMN enrichi.
2. **Détection locale** de problèmes d'interopérabilité. Pour cela, nous appliquons les techniques de vérification proposées dans le chapitre 4.

- **Propagation** sur le modèle GRADEI pour la détection globale des problèmes d'interopérabilité.

4. Cas d'application 1 : préparation d'une conférence nationale

Le premier cas d'application, auquel nous nous intéressons, concerne l'organisation, le suivi et la clôture d'une conférence nationale par un organisme de recherche. Un tel événement nécessite l'interaction de nombreux acteurs avant, pendant et après le déroulement de la conférence. Ces acteurs peuvent être internes, appartenant à l'organisation qui s'occupe de la conférence (il s'agit, ici, d'un laboratoire de recherche appartenant à une école d'ingénieurs), ou externes *i.e.* n'appartenant pas au laboratoire de recherche mais participant à l'organisation de la conférence. Les acteurs internes assurent, par exemple, les fonctions dévolues aux comités d'organisation et scientifique, l'administration des inscriptions, le support technique... Les acteurs externes représentent sont les membres des comités, les fournisseurs, les auteurs présentant leurs travaux à la conférence…

4.1. Modélisation du processus collaboratif avec BPMN enrichi

Pour réaliser l'analyse du processus collaboratif pour la préparation d'une conférence, il est nécessaire de le modéliser. Le modèle complet de ce processus collaboratif est fourni en Annexe F. Nous nous intéressons, ici, à un extrait de ce modèle, comme présenté en Figure 51.

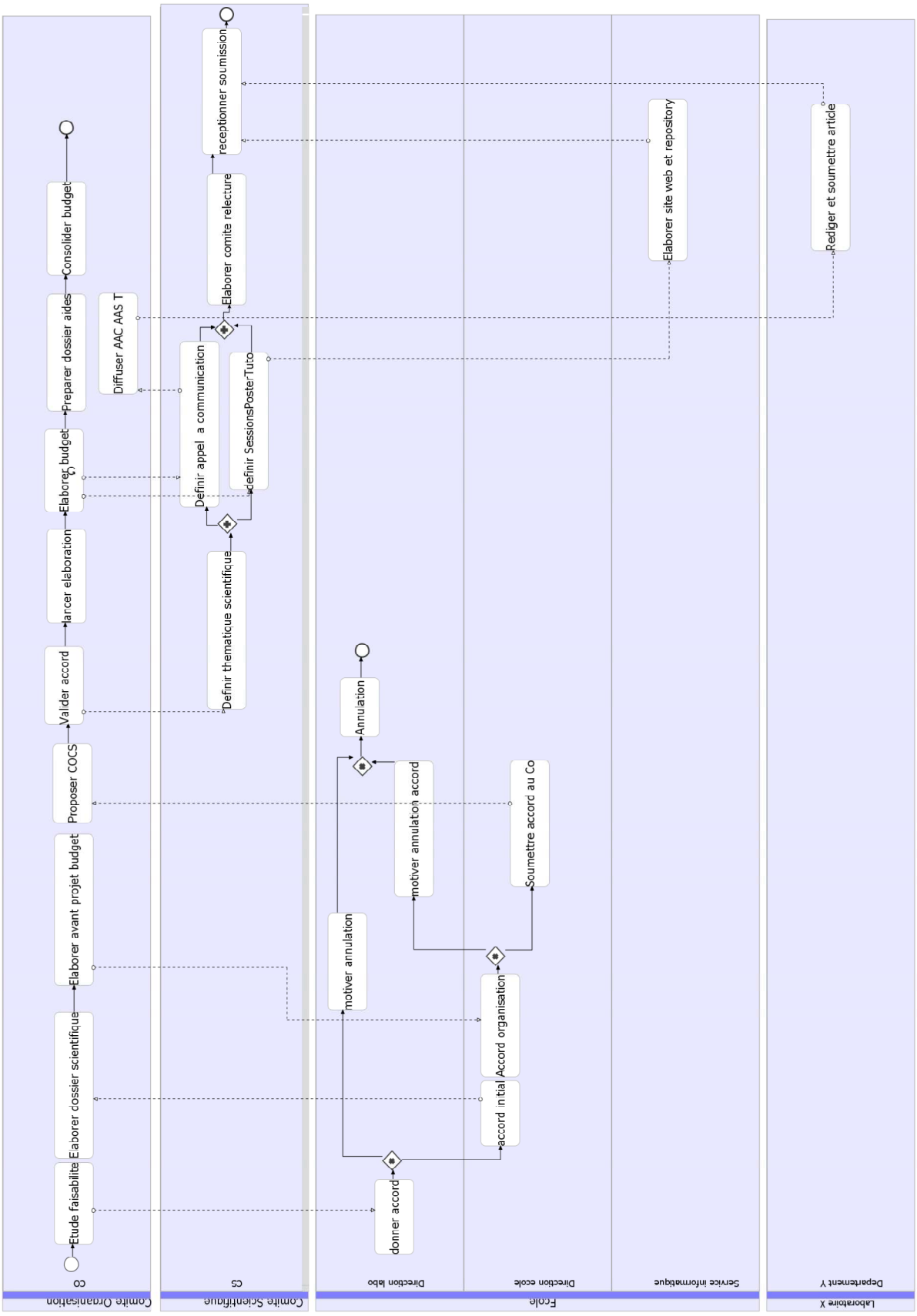


Figure 51. Extrait du modèle de processus d'organisation de conférence

Dans ce modèle, nous nous intéressons aux interactions entre les acteurs du laboratoire organisateur, le comité scientifique et le comité d'organisation. De ce fait, ces acteurs impliqués sont représentés à l'aide d'objets de type « *pool* » et « *lane* » comme suit :

- Une « *pool* », qui représente l'**école** organisatrice. Elle est composée de trois « *lanes* » représentants trois départements différents dont les rôles sont distincts.
    - Une « *lane* » représente la **direction de l'école**. Les acteurs, ici, s'occupent de donner leur accord pour autoriser une conférence au laboratoire. Cet accord se déploie au travers de trois activités : **accord initial**, **accord organisation et soumettre accord au comité d'organisation**.
    - Une « *lane* » représente la **direction du laboratoire**. Le rôle des acteurs, ici, se distingue au travers des activités suivantes : **donner accord** (afin de donner un accord de principe au comité d'organisation), **motiver annulation**, **motiver annulation accord** et **annulation** (afin d'annuler l'accord de principe).
    - Une « *lane* » représente le **service informatique**. Les acteurs de ce service se préoccupent de l'**élaboration du site web**.
- Une « *pool* » représente le **comité scientifique** dont le rôle est de s'occuper de la thématique scientifique. Afin d'accomplir son rôle dans le processus collaboratifs, les acteurs du comité d'organisation sont impliqués dans les activités suivantes : **définir thématique scientifique**, **définir appel à communication**, **définir les sessions**, **élaborer le comité de relecture** et **réceptionner les soumissions**.
- Une « *pool* » représente le **comité d'organisation**. Le rôle des acteurs, ici, est de s'occuper de toute l'organisation de la conférence *via* les activités suivantes : **étude faisabilité**, **élaborer dossier scientifique**, **élaborer avant projet budget**, **proposer comité d'organisation/comité scientifique**, **valider accord du comité d'organisation**, **élaborer budget**, **préparer dossier aide**, **consolider budget**, **définir appel à communication** et **diffuser appel à communication**.

4.2. Détection locale de problèmes d'interopérabilité

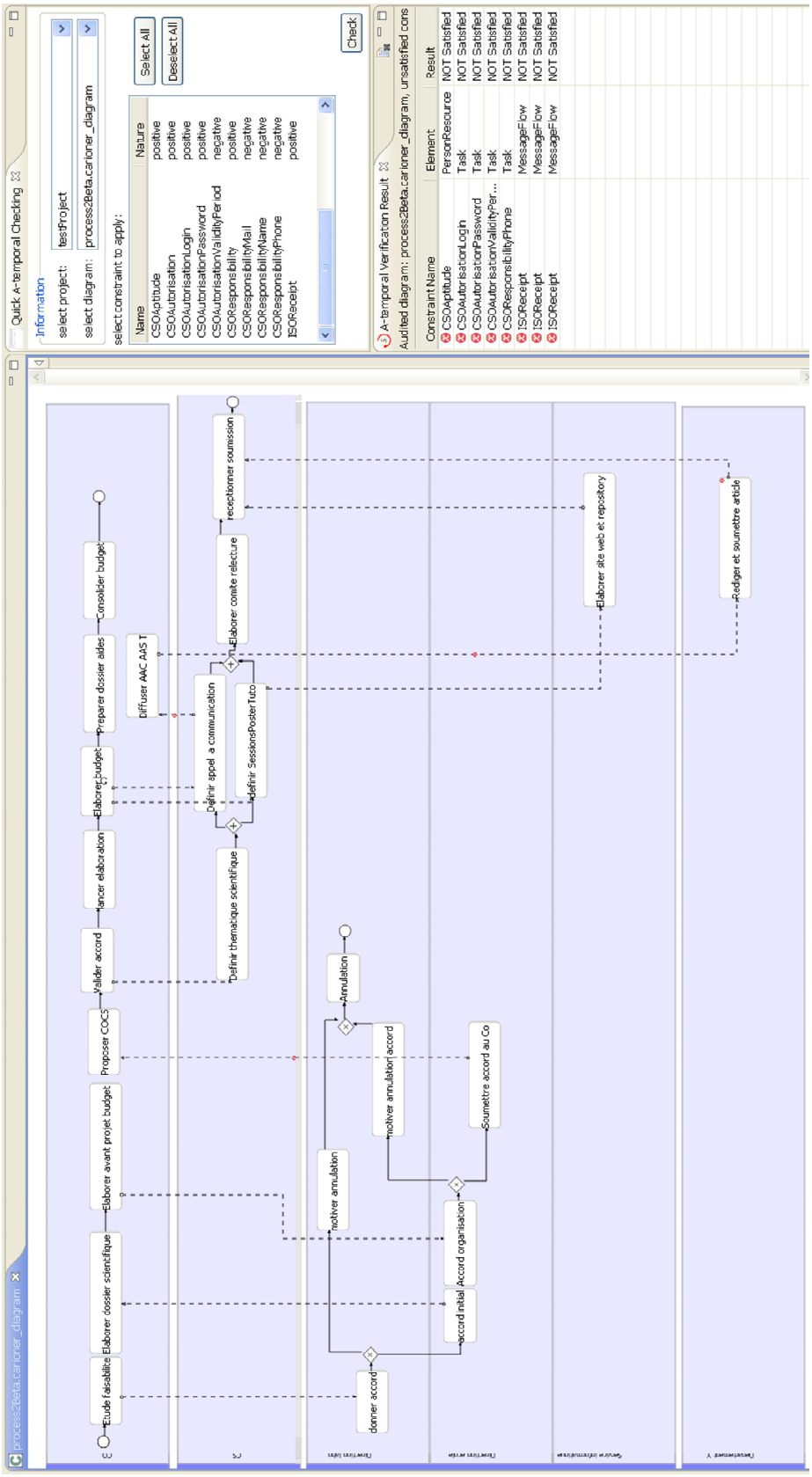


Figure 52. Résultat de la vérification des exigences d'interopérabilité atemporelles

Dans un premier temps, nous nous attachons à la vérification des exigences d'interopérabilité atemporelles. Il s'agit de la vérification des exigences de compatibilité suivantes : $CSO_{ResponsabilitéNom}$, $CSO_{ResponsabilitéPhone}$, $CSO_{ResponsabilitéMail}$, $CSO_{Aptitude}$, $CSO_{AutorisationLogin}$, $CSO_{AutorisationMotPasse}$ et $CSO_{AutorisationValidité}$ et de l'exigence d'interopération $ISO_{AcccuséRéception}$. L'outil COGITANT effectue la projection adéquat selon la nature de la contrainte et réponds par vrai ou faux. Le résultat de la vérification de ces exigences est schématisé sur la Figure 52.

Nous pouvons remarquer que l'exigence $CSO_{ResponsabilitéMail}$ n'est pas satisfaite par la tâche « Rédiger et soumettre un article ». En effet, cette contrainte, de nature négative, est projetée sur le modèle de fait représentant le modèle de processus collaboratif. L'email d'un auteur qui est responsable de rédiger et de soumettre un article à la conférence n'est pas référencé dans une base de données. En conséquence, il ne pourra jamais recevoir l'appel à communication afin de participer à la conférence. Dans le cas où cette information doit être rajoutée, l'auteur est informé de son omission.

Le résultat de la vérification de l'exigence $CSO_{Aptitude}$, nous informe que celle-ci n'est pas satisfaite par une des ressources employées dans une tâche. En effet, nous pouvons remarquer sur la Figure 53 (partie haute gauche) que la tâche « réceptionner soumission » requiert de ces ressources de posséder les aptitudes « Evaluer article » et « Connaissance en interopérabilité ». Or la ressource « Relecteur1 », ne possède pas une de ces aptitudes (*cf.* Figure 53 sur la partie haute droite). L'aptitude non assurée par cette ressource est « connaissances en interopérabilité » comme présentée sur la partie basse droite de la Figure 53.

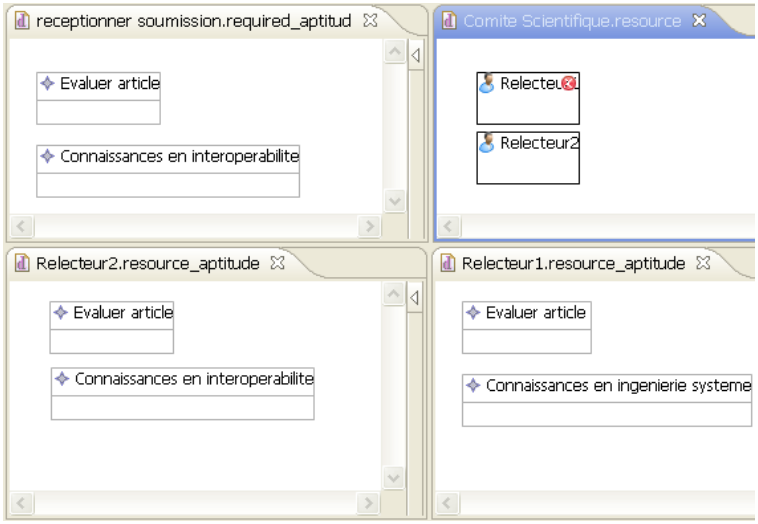


Figure 53. Résultat de la vérification de l'exigence $CSO_{Aptitude}$

Après vérification de cette exigence le comité scientifique peut statuer sur les aptitudes de chaque relecteur susceptible d'être engagé dans la relecture des articles réceptionnés et cela avant même le début de la conférence afin d'éviter de mauvaises évaluations.

La vérification de l'exigence $CSO_{Autorisation}$, nous indique qu'elle est satisfaite par toute activité qui requiert une autorisation. Cependant, pour aller plus en détail et vérifier si des login, mot de passe et période de validité ont bien été fourni par cette activité, nous pouvons nous attacher à la vérification des exigences $CSO_{AutorisationLogin}$, $CSO_{AutorisationMotPasse}$ et $CSO_{AutorisationValidity}$. Le résultat de cette vérification nous montre que ces trois exigences ne sont pas satisfaites par l'activité « Rédiger et soumettre article ». En effet, lorsque des auteurs rédigent un article, ils doivent pouvoir l'envoyer sur un serveur de la conférence avec un login et un mot de passe et cela dans un délai limité.

Enfin, le résultat de la vérification de l'exigence d'interopération $ISO_{AcccuséRéception}$, nous indique qu'un accusé de réception n'est pas renvoyé par la tâche « Diffuser Appel à communication » vers la tâche « Définir appel à communication » qui lui envoie l'appel à communication. Si le comité d'organisation ne renvoie pas un accusé de réception au comité scientifique, ce dernier peut perdre du temps à renvoyer l'appel à communication et négliger ou devoir ralentir ses tâches suivantes induisant du retard dans l'organisation de la conférence.

Intéressons nous maintenant à la vérification des exigences temporelles $ISO_{DisponibilitéRessource}$ et $ISO_{TempsEchange}$. Cette vérification est effectuée sur le modèle du

processus collaboratif équivalent en réseau d'automates temporisés et obtenu par transformation de modèles. Le résultat de vérification de ces exigences, après leur formalisation, est schématisé sur la Figure 54.

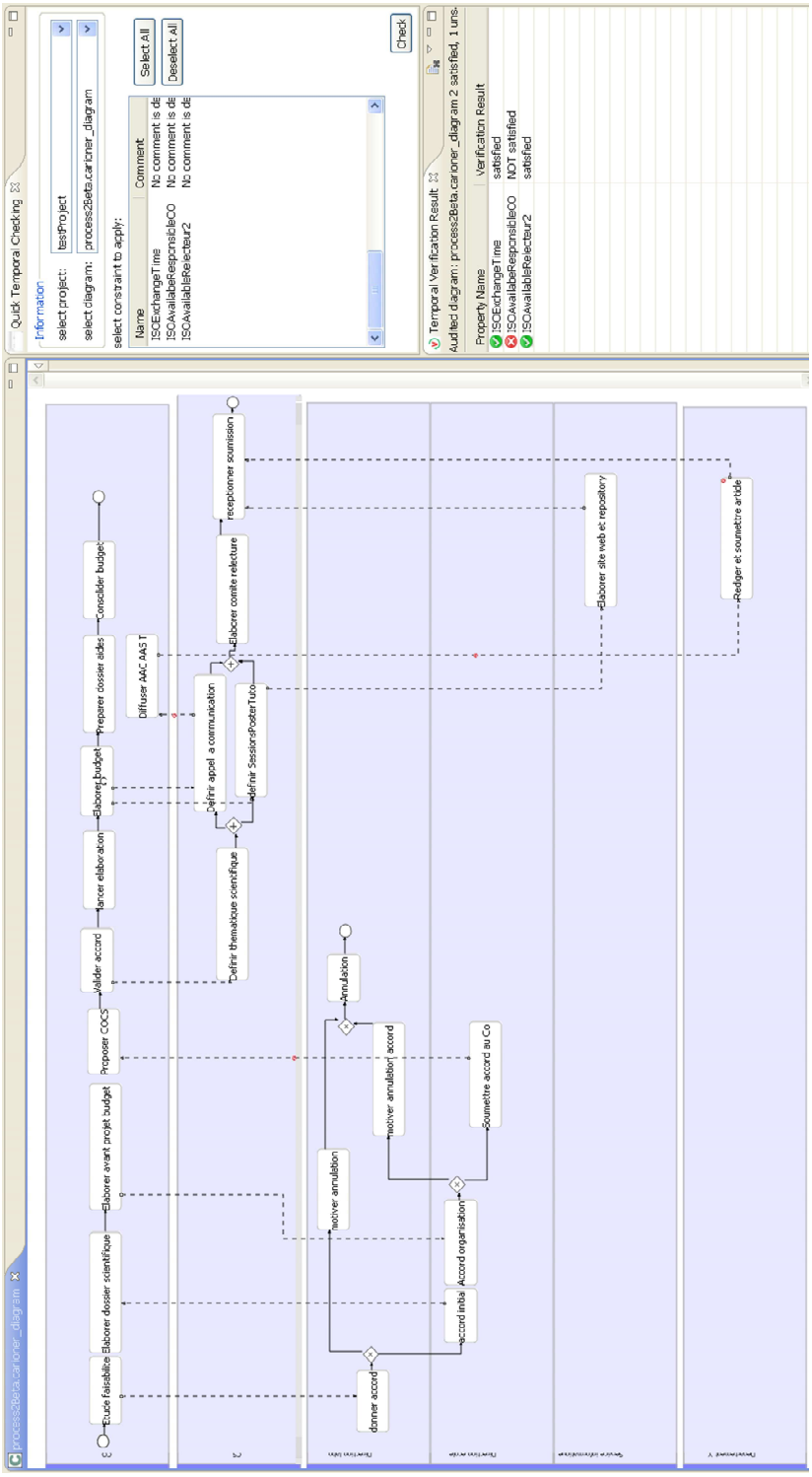


Figure 54. Résultat de la vérification des exigences d'interopérabilité temporelles

Nous cherchons, dans un premier temps, à vérifier la disponibilité du relecteur qui s'occupe d'évaluer les soumissions réceptionnées. De ce fait, la propriété $ISO_{DisponibilitéRelecteur2}$ s'exprime de la manière suivante :

$ISO_{DisponibilitéRlecteur2}$: « *E<> receptionnersoumission_CS_ComiteScientifique.Working and Relecteur2.Active and T<receptionnersoumission_CS_ComiteScientifique.timeMax and T>receptionnersoumission_CS_ComiteScientifique.timeMin* »

Cette propriété exprime le fait qu'il existe un chemin le long duquel le relecteur 2 est employé dans l'activité « receptionner soumission » pendant son temps d'exécution. Nous pouvons remarquer sur la Figure 54 que cette exigence est satisfaite ce qui implique que le comité scientifique peut demander au relecteur 2 de réaliser cette tâche.

Par la suite, nous nous sommes attachés à voir si le responsable du comité d'organisation peut être impliqué dans deux activités différentes qui le sollicitent. En effet, le responsable est sollicité par l'activité « élaborer budget » et l'activité « définir thématique scientifique ». Ces deux activités se déroulent pratiquement en même temps, de ce fait, le résultat de la vérification de la propriété $ISO_{DisponibilitéResponsableCO}$, nous indique qu'elle n'est pas satisfaite et informe le responsable pour qu'il s'organise afin de remplir ces deux tâches correctement.

$ISO_{DisponibilitéResponsableCO}$=« *E<>(Definirthematiquescientifique_CS_ComiteScientifique .WorkingandResponsableCO.ActiveandT<Definirthematiquescientifique_CS_ComiteScientifique.timeMax and T>Definirthematiquescientifique_CS_ComiteScientifique.timeMin) and (Elaborerbudget_CO_ComiteOrganisation.Working and ResponsableCO.Active and T>Elaborerbudget_CO_ComiteOrganisation.timeMinandT<Elaborerbudget_CO_ComiteOrganisation.timeMax)* »

Enfin, nous réalisons la vérification de l'exigence $ISO_{TempsEchange}$ qui stipule que le temps d'échange entre toutes les activités du processus collaboratif ne doit pas dépasser les 5 unités de temps. Cette exigence est formalisée de la façon suivante :

$ISO_{TempsEchange}$: « *Start_Etudefaisabilite.Start → forall(i: NbTask) Reception[i]-Emission[i] <=5* »

Le résultat de cette vérification nous indique que les temps d'échange sont respectés entre toutes les activités et que ceux-ci n'affectent pas le processus collaboratif.

4.3. Détection globale de problèmes d'interopérabilité

Pour voir l'impact global des exigences sur l'interopérabilité au sein du processus collaboratif, nous avons recours à la propagation des résultats sur le modèle GRADEI comme présenté en Figure 55.

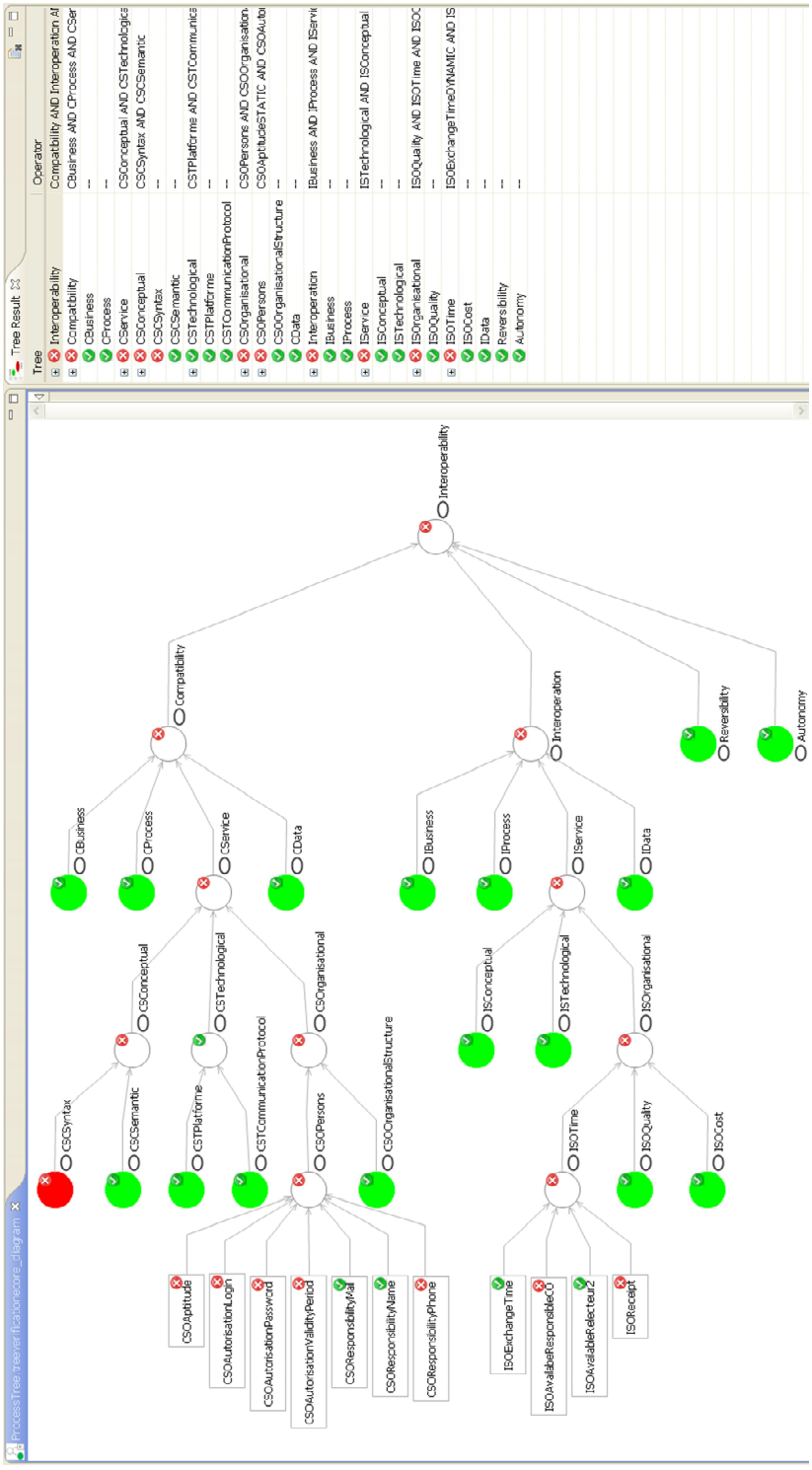


Figure 55. Propagation des résultats de vérification à l'aide du modèle GRADEI

Sur le modèle GRADEI, nous utilisons le résultat des vérifications effectuées avec les outils COGITANT et UPPAAL pour les exigences atemporelles et temporelles respectivement (exigences représentées à l'extrême gauche). Les exigences modélisées en vert et en rouges représentent le résultat de la vérification par expertise. Pour celles-ci, l'expert a la possibilité de les valider, ou non, directement sur le modèle GRADEI.

Comme nous pouvons le remarquer sur la Figure 55, le résultat de la propagation indique que l'exigence globale « interopérabilité » n'est pas satisfaite (exigence la plus abstraite à l'extrême droite). Cela est dû, d'une part, à la fonction logique (ET) qui lie les exigences abstraites avec les exigences qui les composent, et d'autre part à la non satisfaction de certaines des exigences vues plus haut.

5. Cas d'application 2 : conception et mise en production d'un véhicule

Le deuxième cas d'application est un processus collaboratif pour la conception et de production d'un véhicule. Plusieurs partenaires géographiquement répartis sur le territoire européen souhaitent anticiper certains défauts inhérents à l'interopérabilité.

5.1. Modélisation du processus collaboratif avec BPMN Enrichi

Un modèle complet du processus est fourni en Annexe F. Il est modélisé avec le langage BPMN enrichi comme présenté en Figure 56.

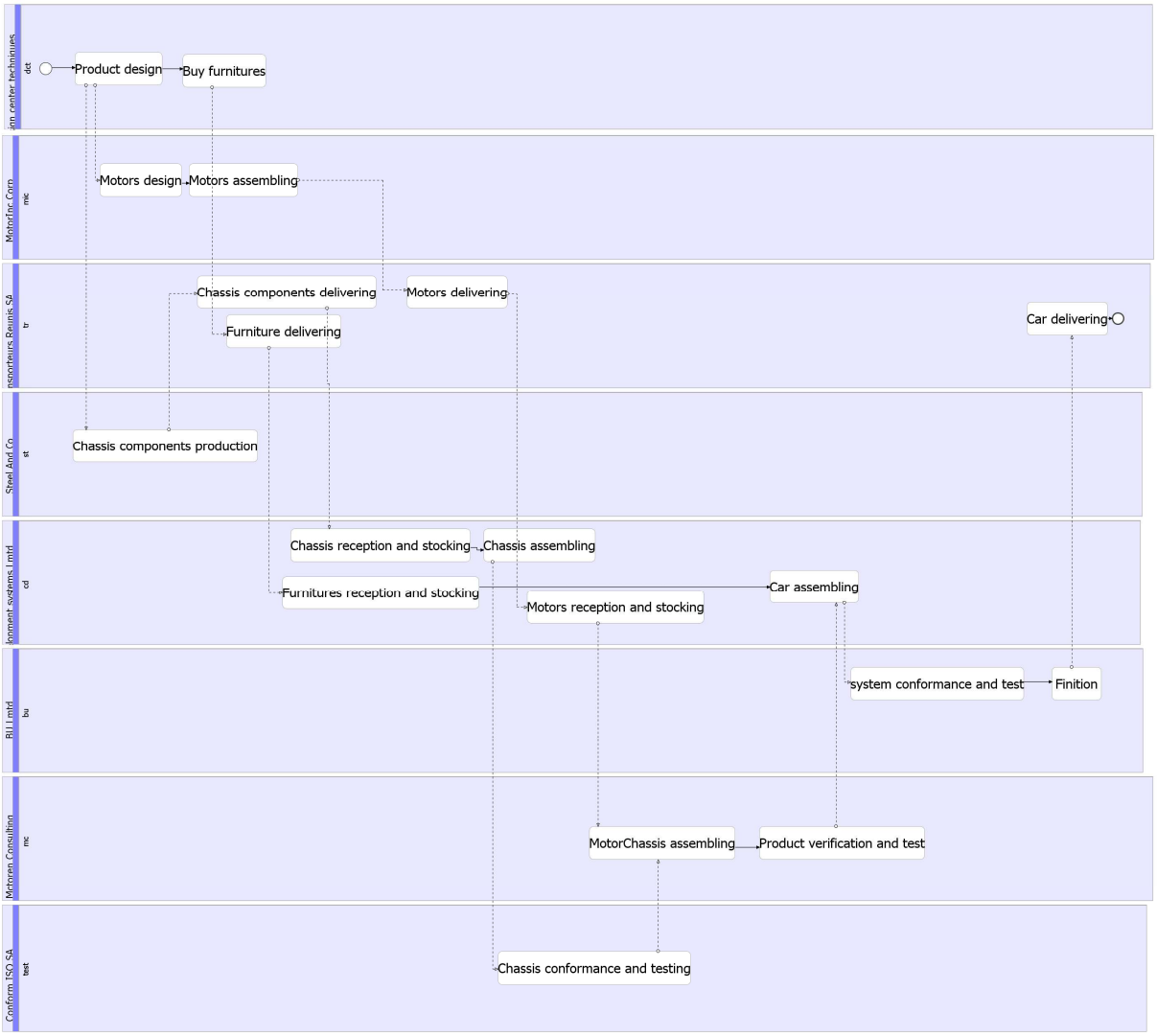


Figure 56. Extrait du modèle de processus de conception et de mise en production

Dans ce modèle, nous nous intéressons aux interactions entre les acteurs des différentes entreprises impliqués dans le processus collaboratif. De ce fait, chaque entreprise est représentée à l'aide de « *pool* » possédant chacune une « *lane* » du même nom et leurs acteurs sont impliqués dans les activités comme suit :

- Une « *pool* », représente l'entreprise ***Design center techniques***. Elle s'occupe de la conception du produit (***product design***) et de l'achat des fournitures (***buy furnitures***).
- Une « *pool* » représente l'entreprise ***MotorInc Corp***. Les acteurs, ici, s'occupent du design des moteurs (***motors design***) et de leur assemblage (***motors assembling***).

- Une « *pool* » représente l'entreprise ***Transporteurs Reunis SA***. Le rôle de ses acteurs est de s'occuper de toutes les livraisons. Il se distingue au travers des activités suivantes : ***chassis components delivering***, ***furniture delivering***, ***motors delivering*** et ***cars delivering***.
- Une « *pool* » représente l'entreprise ***Steel and Co***. Les acteurs de cette entreprise s'occupent de la production des composants du châssis (***chassis components production***).
- Une « *pool* » représente l'entreprise ***Car development Systems***. Cette entreprise s'occupe de l'assemblage des pièces d'un véhicule. Afin d'accomplir son rôle dans le processus collaboratifs, les acteurs sont impliqués dans les activités suivantes : ***chassis reception and stocking***, ***chassis assembling***, ***furnitures reception and stocking***, ***motors reception and stocking*** et ***car assembling***.
- Une « *pool* » représente l'entreprise ***BU Lmtd***. Le rôle des acteurs, ici, est d'effectuer des tests et des vérifications et il est représenté au travers des activités suivantes : ***motor chassis assembling*** et ***product verification and teste***.
- Une « *pool* » représente l'entreprise ***Confor ISO SA***. Le rôle des acteurs, ici, est de voir la conformité du châssis par rapport aux normes existantes (***chassis conformance and testing***).

Ce processus collaboratif peut être confronté à de sérieux problèmes d'interopérabilité qui peuvent provoquer, par exemple, une perte considérable de temps et d'argent entrainant, à terme, la dissolution de la collaboration.

5.2. Détection locale de problèmes d'interopérabilité

L'objectif, ici, est de détecter et de porter à la connaissance des partenaires les problèmes d'interopérabilité susceptibles d'apparaitre au cours de la collaboration. Pour ce faire, la même procédure de vérification que celle réalisée pour le cas précédent est appliquée.

Le résultat de la vérification des exigences d'interopérabilité atemporelles est donné en Figure 57.

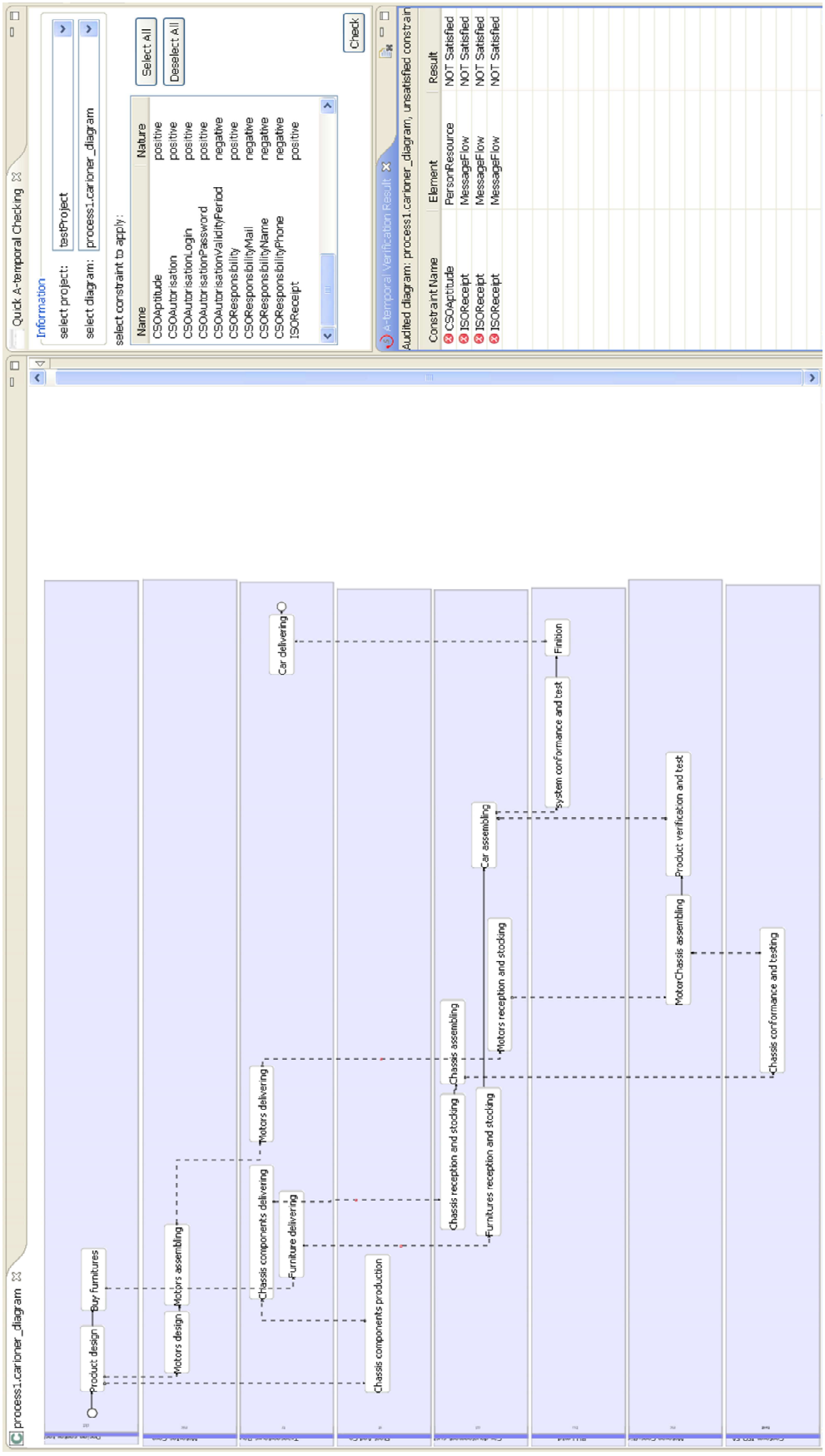


Figure 57. Résultat de la vérification des exigences d'interopérabilité atemporelles

Nous pouvons remarquer que les exigences $CSO_{ResponsabilitéNom}$, $CSO_{ResponsabilitéPhone}$ et $CSO_{ResponsabilitéMail}$ sont satisfaites par toutes les activités (aucun retour d'insatisfaction des exigences). Ce qui implique que les partenaires se connaissent et arrivent à s'identifier.

L'exigence $CSO_{Aptitude}$ n'est pas satisfaite par une tâche. En effet, nous pouvons remarquer que la tâche « motor assembling » ainsi que la tâche « chassis assembling » font appel à la même ressource. La première tâche requiert l'aptitude « Assembling Components Motors » (*cf.* Figure 58 partie haute droit) alors que la seconde tâche requiert l'aptitude « Assembling Components » (*cf.* Figure 58 partie haute gauche). Cependant, la ressource employée ne possède que l'aptitude « Assembling Components Motors » (*cf.* Figure 58 partie basse droite). En conséquence, l'entreprise doit envisager de changer de ressource. Grâce à la détection de ce problème *via* l'approche que nous proposons, ce changement peut être fait avant l'exécution du processus. Ne pas effectuer ce changement avant l'exécution du processus collaboratif, risque de sanctionner ce dernier en termes de temps d'exécution pour réaliser ce changement suite à sa détection soudaine.

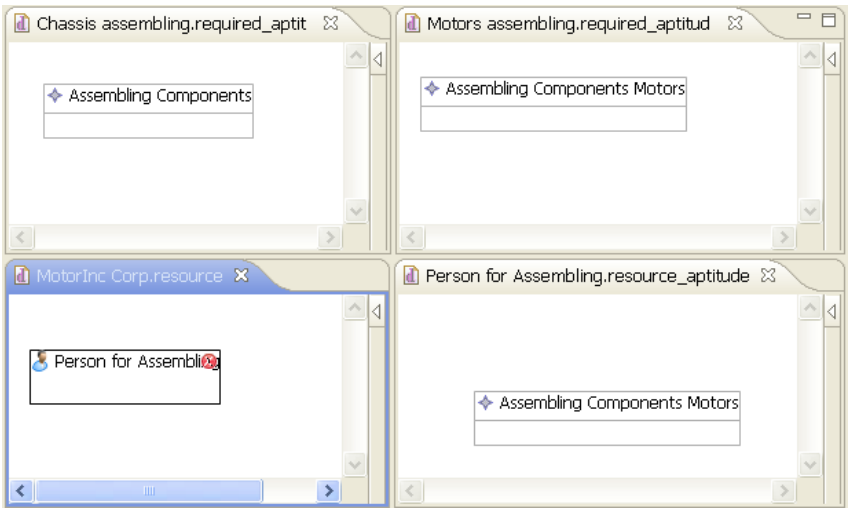


Figure 58. Résultat de la vérification de l'exigence $CSO_{Aptitude}$

La vérification de l'exigence $CSO_{Autorisation}$, nous indique qu'elle est satisfaite par toute activité qui requiert une autorisation. Enfin, la vérification de l'exigence d'interopération $ISO_{AcccuséRéception}$, nous indique que les activités « *chassis reception and stocking* », « *furniture reception and stocking* » et « *motors reception and stocking* » ne renvoient pas d'accusés de réception aux activités qui leurs fournissent leur stock. L'entreprise s'attachant à livrer tous les composants doit prendre soin de s'assurer que la livraison de chaque composant est bien effectuée pour éviter des coûts et des délais supplémentaires.

Pour la vérification des exigences temporelles, nous nous intéressons, dans un premier temps, à voir si les activités « *motor assembling* » et « *chassis assembling* » peuvent utiliser la même ressource. Pour ce faire, nous nous attachons à la vérification des propriétés $ISO_{AvailableResourceForAM}$ et $ISO_{AvailableResourceForAC}$ pour savoir si la ressource peut bien être employée par les deux activités. Ces Exigences sont formalisées en propriétés avec un fragment de la logique TCTL comme suit.

$ISO_{AvailableResourceForAC}$:«*E<>Chassisassembling_cd_CardevelopmentsystemsLmtd.WorkingandPersonforAssembling.ActiveandT<Chassisassembling_cd_CardevelopmentsystemsLmtd.timeMax and T>Chassisassembling_cd_CardevelopmentsystemsLmtd.timeMin* »

$ISO_{AvailableResourceForAM}$: « *E<> Motorsassembling_mic_MotorIncCorp.Working and PersonforAssembling.Active and T<Motorsassembling_mic_MotorIncCorp.timeMax and T>Motorsassembling_mic_MotorIncCorp.timeMin* »

Enfin, nous nous attachons à vérifier si tous les temps d'échanges entre toutes les activités ne dépassent pas les 6 unités de temps. Pour cela, nous vérifions l'exigence $ISO_{TempsEchange}$ qui est formalisée par la propriété $ISO_{ExchangeTime}$.

$ISO_{ExchangeTime}$: « *Start_Productdesign.Start --> forall(i: NbTask) Reception[i]-Emission[i] <6* »

Le résultat de vérification de ces propriétés est donné en Figure 59.

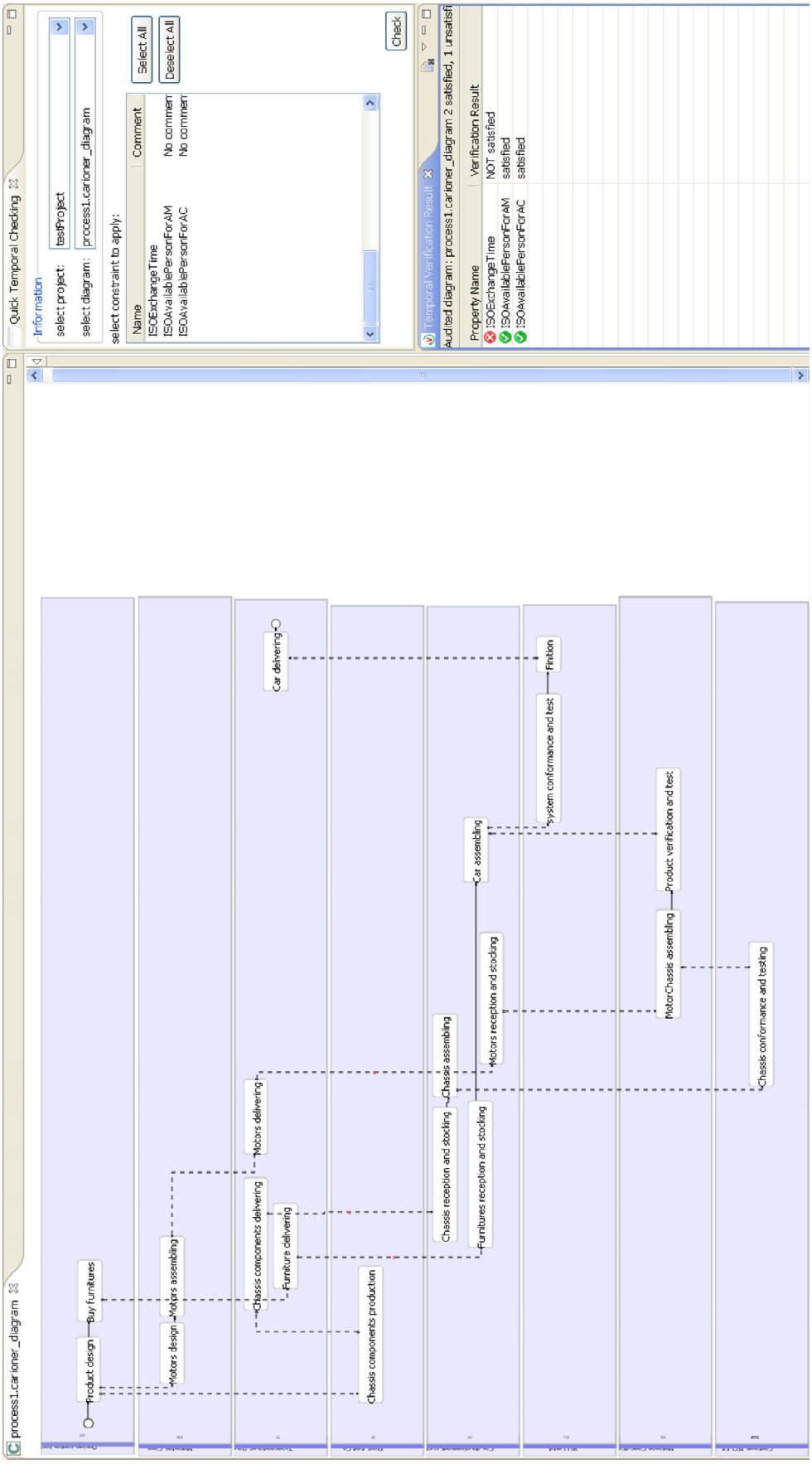


Figure 59. Résultat de la vérification des exigences d'interopérabilité temporelles

La vérification des deux premières propriétés par le *model checker* UPPAAL, nous indique qu'elles sont satisfaites par le modèle du processus collaboratif. En effet, la ressource est employée par les deux activités dans des périodes différentes. Cependant, il ne faut pas oublier que cette ressource ne possède pas l'aptitude requise par l'une des activités et le fait qu'elle soit bien disponible ne veut pas dire que l'on puisse l'utiliser. En conclusion, il est important de prendre toutes les exigences d'interopérabilité en considération et statuer sur le fait que les deux vérifications sont complémentaires.

La vérification de la dernière propriété, nous permet de statuer sur tous les temps d'échange rapidement et nous indique qu'au moins une tâche ne respecte pas le temps d'échange imposé.

5.3. Détection globale de problèmes d'interopérabilité

La détection locale de ces problèmes d'interopérabilité n'est pas suffisante. En effet, ces problèmes peuvent impacter la collaboration à un niveau plus globale. De ce fait, nous nous intéressons à la détection globale vis-à-vis de l'exigence d'interopérabilité (*cf.* Figure 60).

La propagation, pour ce cas d'application, suit le même principe que celui présenté pour le cas d'application précédent. La fonction logique qui lie les exigences abstraites à des exigences plus précises qui les composent est également un opérateur « ET ». Le résultat de cette propagation est représenté sur la figure suivante.

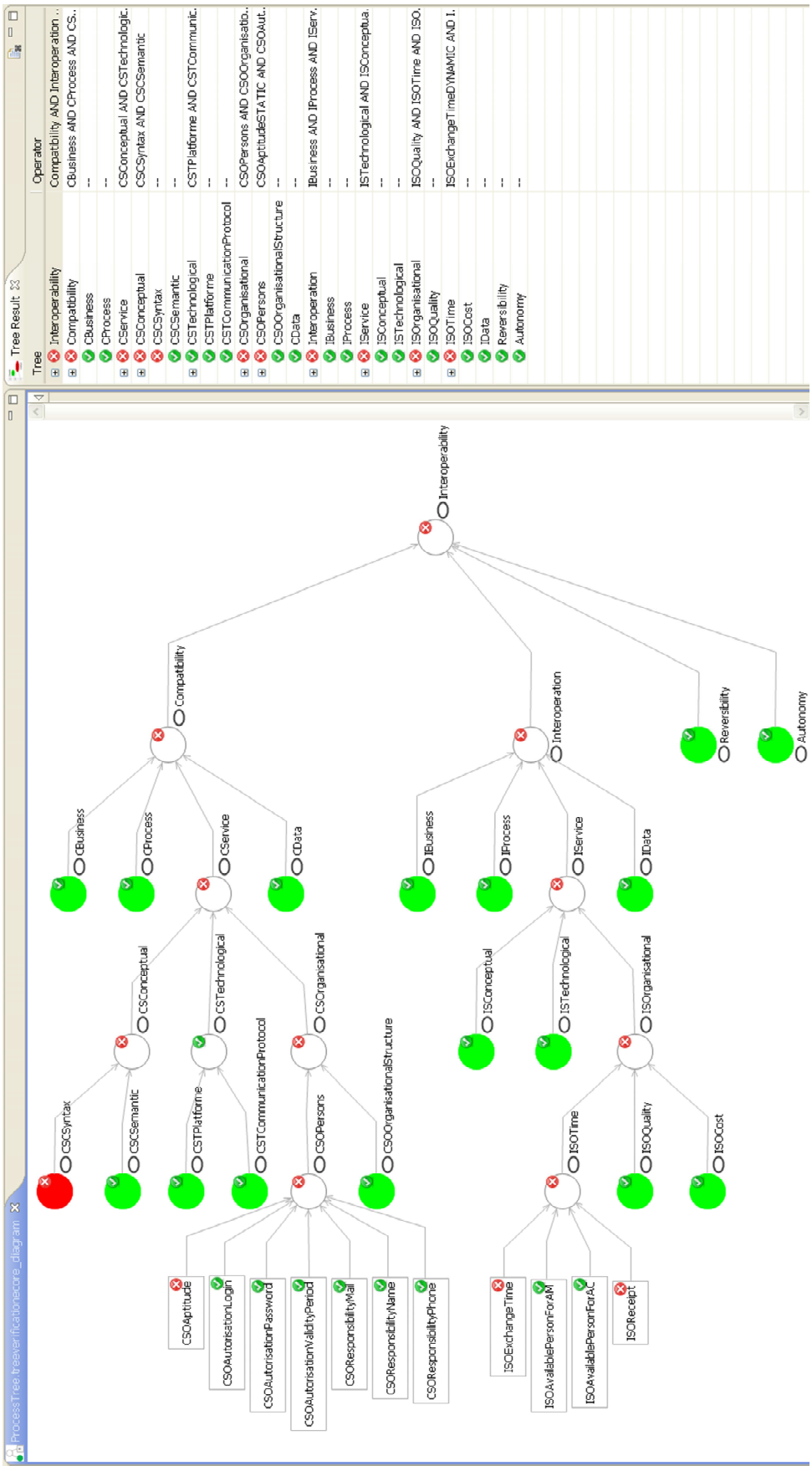


Figure 60. Propagation des résultats de vérification à l'aide du modèle GRADEI

A partir du résultat de la propagation, nous pouvons constater que même si une ressource est disponible pour effectuer une tâche quelconque, si elle ne possède pas l'aptitude requise, le processus peut être freiné. De plus, la non vérification de ces exigences implique que l'exigence globale « interopérabilité » n'est pas satisfaite.

6. Retours et commentaires

D'après les résultats obtenus par la vérification des exigences d'interopérabilité, les acteurs impliqués dans le processus collaboratif d'organisation de conférence peuvent statuer sur les problèmes qu'ils peuvent rencontrer. Ainsi, l'approche fournie des indications sur la présence ou non de problèmes d'interopérabilité ainsi que sur leur localisation. De cette façon, les acteurs peuvent solutionner les problèmes détectés directement là où ils peuvent apparaitre avant même qu'ils ne se posent.

De plus, il est possible pour les acteurs de statuer sur l'importance et sur la nécessité ou non de résoudre les problèmes détectés. Il s'agit de voir l'impact de la non satisfaction d'une exigence d'interopérabilité sur tout le processus avec la propagation obtenu grâce au modèle GRADEI proposé.

Pour appliquer notre approche sur ces deux cas d'application, il était nécessaire de modéliser le processus collaboratif puis de formaliser les exigences d'interopérabilité. Lorsque nous disposons de toutes les informations nécessaires pour la modélisation du processus collaboratif, cette étape peut être rapidement accomplie. Cependant, la formalisation des exigences d'interopérabilité nécessite de connaitre parfaitement le formalisme avec lequel il est possible de les vérifier.

7. Conclusion

Nous avons présenté l'utilisation de notre approche au travers de deux cas d'application. Ceci nous permet de montrer l'applicabilité de notre approche d'anticipation et de détection de problèmes d'interopérabilité. Nous avons montré, dans un premier temps, la mise en œuvre de la détection locale des problèmes d'interopérabilité. Dans un second temps, nous avons montré la mise en œuvre de la détection globale au travers de mécanismes de propagation.

Dans cette étude, nous nous sommes limités à vérifier quelques exigences d'interopérabilité. En effet, nous nous sommes attachés à vérifier les exigences d'interopérabilité qui nous paraissent les plus pertinentes pour les deux cas d'application. Ainsi, nous avons vérifié des exigences de compatibilité et des exigences d'interopération en utilisant des techniques de vérification formelle. La vérification de l'exigence de réversibilité et celle de l'exigence d'autonomie se sont avérées impossibles sur le modèle de processus collaboratif dont nous disposions. Nous les avons donc destinées à une vérification par expertise.

Nous avons montré que cette approche apporte des indications sur l'existence de problèmes d'interopérabilité à un niveau local (positionnement du dysfonctionnement sur le processus, message à destination des acteurs). Ceci peut aider à solutionner les problèmes détectés et, à terme, améliorer l'interopérabilité dans un processus collaboratif. De plus, avec la détection globale - à l'aide de mécanisme de propagation - nous avons montré l'impact que peut avoir un problème d'interopérabilité à un niveau local sur l'exigence globale d'interopérabilité.

Cependant, l'applicabilité de cette approche reste limitée lorsque nous considérons des modèles très abstraits où encore conséquent car cela peut entrainer une augmentation des temps de vérification inhérents aux outils et à leur mode de vérification. Enfin, l'approche reste, pour l'instant, destinée à des utilisateurs qui ont les connaissances nécessaires sur les techniques de vérification employées pour ce qui est de la formalisation des exigences en propriétés ainsi que sur la modélisation des processus.

Conclusion, limites et perspectives

1. Conclusion

Nous nous sommes attachés, dans ce travail de recherche, à proposer une approche pour améliorer l'interopérabilité d'entreprise dans les processus collaboratifs. Cette approche se propose de répondre à la problématique de détection et d'anticipation de problèmes d'interopérabilité. Pour ce faire, notre contribution s'articule autour de trois points majeurs.

Tout d'abord, nous avons proposé que l'interopérabilité soit vue comme une exigence. A partir de ce constat, nous avons défini un ensemble d'exigences d'interopérabilité que nous avons positionné au sein d'une classification. Ces classes représentent les notions primordiales de l'interopérabilité à savoir, la compatibilité, l'interopération, l'autonomie et la réversibilité. Par la suite, à partir des besoins d'interopérabilité récoltés et liés à chaque classe d'exigences, nous avons développé un premier **référentiel d'exigences d'interopérabilité**.

Dans le but de structurer ce référentiel d'exigences d'interopérabilité, nous avons ensuite proposé le **modèle GRADEI** basé sur la notion de graphe et de décomposition. Ce modèle permet de décrire précisément les causes d'un problème d'interopérabilité (détection locale) et de statuer sur l'exigence globale d'interopérabilité (détection globale), à l'aide de mécanismes de propagation. Il est à noter que ce modèle peut être utilisé comme un guide aidant à obtenir rapidement un modèle de processus possédant par défaut les propriétés attendues. De plus, le modèle GRADEI est extensible et peut être enrichi par des exigences d'interopérabilité propres à l'utilisateur. En conséquence, ce modèle est également un guide pour l'ingénierie d'exigences d'interopérabilité.

Enfin, l'approche proposée met en œuvre des techniques de vérification formelle pour fournir une preuve tangible de l'existence d'un problème d'interopérabilité. L'utilisation de ce genre de techniques reste difficile à appliquer sur des modèles de processus qui sont issus de la modélisation d'entreprise, ni même sur des concepts informels telle que l'interopérabilité des entreprises. En conséquence, une dernière contribution de ces travaux de recherche réside dans l'**application de techniques formelles**. Dans ce sens, nous avons proposé, dans un premier temps, de représenter les éléments d'un processus collaboratif de façon formelle, et nous intéressons plus particulièrement au comportement de ce dernier. Dans un second temps, nous avons proposé une formalisation du concept d'interopérabilité à travers la formalisation des exigences d'interopérabilité.

Pour terminer, nous avons développé un outil qui montre l'applicabilité de notre approche.

2. Limites et perspectives

L'approche de détection et d'anticipation de problèmes d'interopérabilité que nous avons développée possède des limites auxquelles nous pouvons envisager des solutions à travers des perspectives proposées ici.

Premièrement, le référentiel d'exigences d'interopérabilité que nous avons proposé n'a pas la prétention d'être exhaustif. Cependant, ce premier référentiel est flexible et extensible en fonction des besoins de l'utilisateur.

Ensuite, le modèle GRADEI que nous avons proposé permet la détection locale et globale de problèmes d'interopérabilité. Cependant, la notion d'influence que nous avons évoquée au chapitre 3 entre les différentes classes d'exigences d'interopérabilité n'est pas considérée pour l'instant. Pour ce faire, nous proposons d'intégrer cette influence dans l'algorithme de propagation présenté au chapitre 3. De plus, seule la fonction logique θc (fonction logique d'interprétation de la décomposition) est actuellement considérée par le modèle GRADEI. En effet, bien que la fonction logique θe est présente au sein de l'algorithme de propagation, nous l'avons toujours considérée comme vraie puisqu'elle est difficile à appréhender et non liée à l'interopérabilité.

Une autre limite de nos travaux est liée à l'utilisation de techniques de vérification formelle. Par exemple, la technique reposant sur le *model checking* est souvent confrontée au phénomène d'explosion combinatoire. Cette limite se traduit par l'impossibilité de vérifier des exigences d'interopérabilité. En conséquence, nous proposons d'étudier des mécanismes de partitionnement de modèle afin de lever ce problème.

Notre approche se limite à formaliser les exigences d'interopérabilité en propriétés directement avec les graphes conceptuels ou la logique TCTL. Cependant, il reste difficile de s'assurer de la conformité et de la qualité de l'expression d'une propriété. De plus, l'utilisateur doit avoir la maitrise explicite des langages de représentation des propriétés. En conséquence, il est proposé d'avoir recourt à un langage pivot et unique de description de propriétés tel que le langage LUSP (Lamine, 2001).

Ensuite, nous avons vu que les exigences de réversibilité sont toujours destinées à une vérification par expertise avec des retours d'expérience. Cette analyse est souvent confrontée aux problèmes de précision et d'erreurs humaines. Pour pallier ce problème, nous pouvons envisager d'utiliser une technique de vérification basée sur les systèmes multi-agents afin de simuler la collaboration.

Enfin, nous nous sommes limités dans ces travaux de recherche à constater et à détecter des problèmes d'interopérabilité sans rentrer dans le domaine de la solution. Nous proposons d'établir un référentiel des solutions en accord avec le référentiel des exigences permettant à un utilisateur de sélectionner et de comparer diverses alternatives de solutions. Les techniques de vérification proposées dans notre approche peuvent alors être utilisées pour aider l'utilisateur à comparer ces alternatives.

Bibliographie

A

(Abed, 2009) Abed N., « *Exploration randomisée de larges espaces d'états pour la vérification* ». Thèse de doctorat, Université Grenoble 1, juin 2009.

(Aloui, 2007) Aloui S., « *Contribution à la modélisation et l'analyse du risque dans une organisation de santé au moyen d'une approche système* ». Thèse de doctorat, Ecole des Mines de Paris, novembre 2007.

(Alur *et al.*, 1993) Alur R., Courcoubetis C., Dill D., « *Model-Checking in Dense Real-Time* ». Information and Computation Journal, volume 104(1), pp.2-34, 1993.

(AMICE, 1993) AMICE, « *CIMOSA : Open Systems Architecture for CIM* ». Springer-Verlag, Berlin, 1993.

(ATHENA, 2003) ATHENA Integrated Project, « *Advanced Technologies for Interoperability of Heterogeneous Enterprises Networks and their Applications* ». Integrated Project description of work, 2003.

(ATHENA, 2005) ATHENA Integrated Project, « *Framework for the establishment and management methodology* ». ATHENA deliverable A1.4, 2005.

(ATHENA, 2007) ATHENA Integrated Project, « *Guidelines and best practices for applying the ATHENA interoperability framework to support SME participation in digital ecosystems* ». ATHENA deliverable A8.2, 2007.

(ATL, 2005) ATLAS Groupe INA & INRIA Nantes, « *ATL Atlas Transformation Language. Specification of the ATL Virtual Machine* ». Version 0.1, 2005

(Aubert *et al.*, 2002) Aubert B., Dussart A., « *Système d'Information Inter-Organisationnel* ». Rapport Bourgogne, CIRANO, mars 2002.

B

(Bachman, 2005) Bachman J., « *Service-Oriented Architecture Maturity Model* ». http://www.sonicsoftware.com (accessible à la date du 11 janvier 2012), 2005.

(Baget, 2001) Baget J.F., « *Représenter les connaissances et raisonner avec des hypergraphes: de la projection à la dérivation sous contraintes* ». Thèse de doctorat, Université de Montpellier 2, 2001.

(Baïna, 2006) Baïna S., « *Interopérabilité dirigée par les modèles : Une Approche Orientée Produit pour l'interopérabilité des systèmes d'entreprise* ». Thèse de doctorat, Université Henri Poincaré, Nancy 1, décembre 2006.

(Basque, 2004) Basque R., « *CMMI : Un itinéraire fléché vers le Capability Maturity Model Integration* ». Dunod Paris, 2004.

(Behrmann *et al.*, 2004) Behrmann G., David A., Larsen K.G., « *A tutorial on Uppaal* ». Department of Computer Science, Aalborg University, Denmark, 2004.

(Bérard *et al.*, 2001) Bérard B., Bidoit M., Finkel A., Laroussinie F., Petit A., Petrucci L., Schnoebelen Ph., McKenzie P., « *Systems and Software verification: model checking techniques and tools* ». Springer, 2001.

(Bernus *et al.*, 1996) Bernus P., Nemes L., « *A Framework to Define a Generic Enterprise Reference Architecture and Methodology* ». Computer Integrated Manufacturing Systems Journal, volume 9(3), pp. 179-191, 1996.

(Bézivin *et al.*, 2005) Bézivin J., Jouault F., Touzet D., « *Principles, Standards and Tools for Model Engineering* », Proceedings of 10th Int. Conf. on Engineering of Complex Computer Systems, ICECCS2005, Shanghai, Chine, 16-20 juin 2005.

(Blanc, 2006) Blanc S., « *Contribution a la caractérisation et a l'évaluation de l'interopérabilité pour les entreprises collaboratives* ». Thèse de doctorat, Université de Bordeaux 1, décembre 2006.

(BPMN, 2009) BPMN, Business Process Modeling Notation, Version 1.2, http://www.bpmn.org/ (accessible à la date du 11 janvier 2012), 2009.

(BPMN, 2011) BPMN: Business Process Modeling Notation, Version 2.0. http://www.bpmn.org/ (accessible à la date du 11 janvier 2012), 2011.

(Braesch *et al.*, 1995) Braesch C., Haurat A., « *La modélisation systémique en entreprise* », Hermès Science Publications, 1995.

C

(C4ISR, 1998) C4ISR Architecture Working Group, « *Levels of Information Systems Interoperability (LISI)* ». United States of America Department of Defense, Washington DC, USA, 30 mars 1998.

(Camarinha-Matos *et al.*, 2003) Camarinha-Matos L.M., Afsarmanesh H., Rabelo R.J., « *Infrastructure developments for agile virtual enterprises* ». Journal of Computer Integrated Manufacturing, ISSN 0951-192X, Vol. 16, N. 4-5, 2003

(Castanet *et al.*, 1994) Castanet R., Koné O., « *Deriving coordinated testers for interoperability* ». In Proceedings of the IFIP TC6/WG6.1 Sixth International Workshop on Protocol Test systems VI, pages 331–

346, Amsterdam, The Netherlands, North-Holland Publishing Co, 1994.

(Cattan *et al.*, 2001) Cattan M., Idrissi N., Knockaert P. « *Maîtriser les processus de l'entreprise : guide opérationnel* ». Paris, Editions d'organisation, 2001.

(Chapurlat, 2007) Chapurlat V., « *Vérification et validation de modèles de systèmes complexes: application à la Modélisation d'Entreprise* ». Habilitation à Diriger des Recherches, Université des Sciences et Techniques du Languedoc (USTL) Montpellier 2, 2007.

(Chapurlat *et al.*, 2003) Chapurlat V., Kamsu-Foguem B., Prunet F., « *Enterprise model verification and validation: an approach* ». Annual Review in Control, Volume 27, Issue 2, pp 185-197, 2003.

(Chawk, 2000) Chawk M., « *Modèle de graphes conceptuels et representation sémantique du langage naturel* ». Rapport de recherche de l'équipe ERSICO, Université Jean Moulin Lyon 3, 2000.

(Chein *et al.*, 1992) Chein M., Mugnier M.L., « *Conceptual graphs: fundamental notions* ». Revue d'intelligence artificielle, vol.6, n°4, pp. 365- 406, 1992.

(Chen, 2011) Chen D., « *CEN/ISO 11354 – Framework and Maturity Model for Enterprise Interoperability* ». In Workshop of the International IFIP Working Conference on Enterprise Interoperability, IWEI 2011, Stockholm, Suède, 22 mars 2011.

(Clark *et al.*, 1999) Clark T., Jones R., « *Organisational Interoperability Maturity Model for C2* ». Proc of Command and Control Research & Techn. Symposium, Newport, USA, 1999.

(CMMI, 2002) CMMI, « *CMMI for systems Engineering, Software Engineering, Integrated Product and Process Development, and Supplier Sourcing (CMMI- SE/SW/IPPD/SS, V1.1)* ». Staged Representation,

publié par SEI, CMU/SEI- 2002-TR-012, CMMI Product Team, mars 2002.

(COGITANT, 2009) COGITANT, « *CoGITaNT Version 5.2* ». Reference Manual, http://cogitant.sourceforge.net (accessible à la date du 11 janvier 2012), 2009.

D

(Daclin, 2007) Daclin N., « *Contribution au développement d'une méthodologie pour l'interopérabilité des entreprises* ». Thèse de doctorat, Université de Bordeaux 1, décembre 2007.

(Daws *et al.*, 1996) Daws C., Olivero A., Tripakis S., YovineS., « *The Tool KRONOS* ». In Proc. Hybrid Systems III : Verification and Control, volume 1066 of Lecture Notes in Computer Science, pp. 208–219. Springer, 1996.

(DoD, 2003) Department OF Defense, « *Standard Practice Defense and Program-Unique Specification Format and Content* ». MIL-STD-961E, 1 aout 2003.

(Dollfus, 2001) Dollfus O. « *La Mondialisation* ». Presses de Sciences po, 2e éd., 2001.

(Doumeingts, 1984) Doumeingts G., « *Méthode GRAI, méthode de conception des systèmes en productique* ». Thèse de Doctorat, Université Bordeaux I, 1984.

(Doumeingts *et al.*, 1993) Doumeingts G., Chen D., Vallespir B., Fénié P., « *GIM (Grai Integrated Methodology) and its evolutions – A methodology to design and specify advanced manufacturing systems* ». Proceedings of IFIP WG5.3 Workshop on the design of information infrastructure systems for manufacturing (DIISM), 1993.

(Duffy, 1991) Duffy D.A., « *Principles of Automated Theorem Proving* ». John Wiley & Sons, 1991.

(Duflot-Kremer, 2010) Duflot-Kremer M., « *Méthodes et outils de vérification : Présentation du model checker UPPAAL* ». Cours de Master 1, Université Paris XII, 2010.

$\mathcal{E}$

(Edmund *et al.*, 1999) Edmund M., Clarke Jr., Grumbereg O., Doron A.P., « *Model checking* ». The MIT Press, 1999.

(EIF, 2004) EIF, « *European Interoperability Framework For Pan-European eGovernment Services* ». European Commission, version 1.0, European Communities Ed., 2004.

(EIF, 2008) EIF, « *Pan-European eGovernment Services* », Draft Document as basis for EIF 2.0, 2008.

(Eljamal, 2006) Eljamal M.H., « *Contribution à l'évolution des exigences et son impact sur la sécurité* ». Thèse de doctorat, Université du Paul Sabatier, Toulouse, décembre 2006.

(ElMansouri, 2009) ElMansouri R., « *Modélisation et vérification des processus métiers dans les entreprises virtuelles : une approche basée sur la transformation de graphes* ». Thèse de doctorat en science en informatique, Université Mantouri Constantine, Algérie, 2009.

(Esper, 2010) Esper A., « *Intégration des approches SOA et orientée objet pour modéliser une orchestration cohérente de services* ». Institut National des Sciences Appliqués de Lyon, septembre 2010.

(Espinasse, 2008) Espinasse B., « *Représentation des connaissances : introduction aux graphes conceptuels* ». Cours de l'Université d'Aix Marseille, 2008. http://www.lsis.org/espinasseb/Supports/MR-

TC/GraphesConceptuels-oct08-4p.pdf (accessible à la date du 11 janvier 2012)

(Estefan, 2007) Estefan, J.A., « *Survey of Model-Based Systems Engineering (MBSE) Methodology* ». INCOSE MBSE Focus Group Report, 2007.

F

(Ford, 2008) Ford C.T., « *Interoperability measurement* ». Thèse, Department of the Air Force Air University, Air Force Institute of Technology, 2008.

(Forest, 2003) Forest C., « *Empowerment skills for family workers: A worker handbook* ». Cornell University, 2003.

G

(Gallier, 1986) Gallier, J.H., « *Logic for Computer Science: Foundations of Automatic Theorem Proving* ». Harper & Row Publishers, 1986.

(Grangel Seguer *et al.*, 2006) Grangel Seguer R., Bourey J.P., Chalmeta R., Bigand M., « *UML for Enterprise Modelling: a basis for a Model-Driven Approach* ». Enterprise Interoperability: New Challenges and Approaches- Springer Verlag edition, ISBN-10 : 1846287138, 2006.

(Gruhn *et al.*, 2005) Gruhn, V., Laue, R., « *Using Timed Model Checking for Verifying Workflows* ». Computer Supported Activity Coordination 2005: 75-88, 2005.

(Guédria *et al.*, 2009) Guédria W., Chen D., Naudet Y., « *A Maturity Model for Enterprise Interoperability* ». IFAC-IFIP EI2N, Enterprise

Integration, Interoperability and Networking, Portugal, 4-5 novembre 2009.

H

(Henzinger *et al.*, 1995) Henzinger T., Ho P.H., Wong-Toi H., « *A User Guide to HYTECH* ». In Proc. 1st International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'95), volume 1019 of Lecture Notes in Computer Science, pp 41–71. Springer, 1995.

(Huet *et al.*, 1995) Huet G., Khan G., Paulin-Mohring C., « *The Coq proof assistant – a tutorial* ». Technical Report 178, Inria, 1995.

I

(Luzeaux, 2008) Luzeaux D., « *Systèmes de systèmes : du concept à la réalisation effective* ». Système de Systèmes - concepts et illustrations pratiques, chapitre 1, pp 21-110,.Hermes Lavoisier Edition, 2008.

(IDEAS, 2003) IDEAS Project Deliverables, « *WP1-WP7* ». Public reports, 2003.

(IEEE, 1990) IEEE, « *A compilation of IEEE standard computer glossaries* ». Standard computer dictionary, New York, 1990.

(IEEE, 1999) IEEE, « *Standard for application and management of the systems engineering process* ». IEEE 1220, janvier 1999.

(INCOSE, 2007) INCOSE, « *System Engineering (SE) Handbook Working Group, System Engineering Handbook, A « How To »* ». Version 3.1, Guide For All Engineers, http:///www.incose.org (accessible à la date du 11 janvier 2012), 2007.

(INTEROP, 2007) INTEROP, « *Enterprise Interoperability-Framework and knowledge corpus - Final report* ». INTEROP NoE, FP6 – Contract n° 508011, Deliverable DI.3, 21 mai 2007.

(ISO11354-1, 2009) ISO/DIS 11354-1, 2009, « *Advanced automation technologies and their applications. Part 1: Framework for enterprise interoperability* ». 2009

(ISO14258, 1999) ISO 14258– Industrial Atomation Systems – « *Concepts and Rules for Enterprise Models* ». ISO TC 184/SC 5/WG1, 1999.

(ISO15288, 2008) ISO/IEC 15288:2008(E), « *IEEE Standards 15288.2008 – Systems engineering – System life cycle processes (2nd edition)* ». Février 2008.

(ISO16100-1, 2009) ISO 16100-1:2009, « *Systèmes d'automatisation industrielle et intégration - Profil d'aptitude du logiciel de fabrication pour interopérabilité - Partie 1: Cadre* ». TC184/SC5, 2009.

(ISO17933, 2000) ISO 17933 : 2000, « *GEDI - Échange standard de documents électroniques* ». ISO TC 46/SC 4, 2000.

(ISO8402, 1994) ISO 8402, « *Quality management and quality assurance* ». Vocabulary, Second edition 1994-04-01, International Standard Organization, 1994.

(ISO9000, 2000) ISO 9000 (2000), « *Organisation internationale de normalisation* ». Normes internationales ISO 9000 relatives au management de la qualité, ISO, 2000.

$\mathcal{K}$

(Kamsu-Foguem, 2004) Kamsu-Foguem B., « *Modélisation et Vérification des propriétés de systèmes complexes : Application aux processus*

d'entreprise ».Thèse de doctorat, Université de Montpellier 2, 2004.

(Kamsu-Foguem *et al.*, 2006) Kamsu-Foguem B., Chapurlat V., « *Requirements modelling and formal analysis using graph operations* ». International Journal of Production Research 44(17): 3451-3470, 2006.

(Kasunic *et al.*, 2004) Kasunic M., Anderson W., « *Measuring systems interoperability: challenges and opportunities* ». Software engineering measurement and analysis initiative, Technical note CMU/SEI-2004-TN-003, 2004.

(Kingston *et al.*, 2004) Kingston G., Fewell S., Richer W., « *An Organisational Interoperability Agility Model* ». Proc. 10th ICCRTS, 2004.

$\mathcal{L}$

(Lamine, 2001) Lamine E., « *Définition d'un modèle de propriété et proposition d'un langage de spécification associé : LUSP* ». Thèse de doctorat, Université de Montpellier 2, décembre 2001.

(Larousse, 2010) Larousse, « *Le petit Larousse illustré* ». Dictionnaire de la langue française, 2010.

(Lauras, 2004) Lauras M., « *Méthodes de diagnostics et d'évaluation de performance pour la gestion de chaînes logistiques : application à la coopération maison mère – filiales internationales dans un groupe pharmaceutique et cosmétique* ». Thèse de doctorat, Institut Nationale Polytechnique de Toulouse, juillet 2004.

(Le Moigne, 1977) Le Moigne J.L., « *La théorie du système générale – théorie de la modélisation* ». Presse Universitaire de France, 1977.

(Lecerf, 2006) Lecerf M., « *Les petites et moyennes entreprises face à la mondialisation* ». L'Harmattan, 2006.

(Long, 2002) Long J., « *Relationships between Common Graphical Representations in Systems Engineering Vitech Corporation* ». 2002.

M

(Mallek *et al.*, 2010a) Mallek S., Daclin N., Chapurlat V., « *Toward a conceptualisation of interoperability requirements* ». IESA 2010, Interoperability for Enterprise Software & Applications, Coventry, 14-15 avril 2010.

(Mallek *et al.*, 2010b) Mallek S., Daclin N., Chapurlat V., « *Catégorisation et formalisation des exigences d'interopérabilité dans les processus collaboratifs* ». Ingénierie des Systèmes d'Information, Modélisation d'entreprise et interopérabilité. Revue des Sciences et Technologies de l'Information, RSTI série ISI Volume 15 – numéro 5, octobre 2010.

(Mallek *et al.*, 2011) Mallek S., Daclin N., Chapurlat V., « *An approach for interoperability requirements specification and verification* ». International IFIP Working Conference on Enterprise Interoperability, IWEI 2011, Stockholm, Suède, 23-24 mars 2011.

(Marca *et al.*, 1988) Marca D.A., Mc GOWAN C.L., « *Structured analysis and design technique* ». USA: Mc Graw-Hill, 216p, 1988.

(Mattessich *et al.*, 1992) Mattessich P.W., Monsey B.R., « *Collaboration: What Makes It Work ?* ». Amherst H. Wilder Foundation, St. Paul, MN, 1992.

(McMillan, 1992) McMillan K. L., « *Symbolic Model Checking: An approach to the state explosion problem* ». Thèse de doctorat, Carnegie Mellon University, USA, mai 1992.

(Melham *et al.*, 1993) Melham T.F., Gordon M.J.C., « *Introduction to HOL: A theorem proving Environment for higher-order logic* ». Cambridge University Press, 1993.

$\mathcal{N}$

(Nehta, 2007) Nehta, « *Interoperability maturity model* ». Version 1.0, 26 mars 2007.

$\mathcal{O}$

(Ouimet, 2008) Ouimet M., « *Formal Software Verification: Model Checking and Theorem Proving* ». Embedded Systems Laboratory Technical Report ESL-TIK-00214, Massachusetts Institute of Technology, 2008.

(Owre *et al.*, 1996) Owre S., Rajan S., Rushby J., Shankar N., Srivas M.K., « *PVS: combining specification, proof checking and model checking* ». Computer Aided Verification 1996, LNCS 1102, pp 411-414. Springer-Verlag, 1996.

$\mathcal{P}$

(Panetto, 2006) Panetto H., « *Meta-modèles et modèles pour l'intégration et l'interopérabilité des applications d'entreprises de production* ». Habilitation à Diriger des Recherches, Université de Nancy 1, 2006.

(Paulk *et al.*,1995) Paulk, M.C., Weber, C.V., Curtis, B., Chrissis, M.B., « *The capability maturity model : guidelines for improving the software Process* », Carnegie Mellon University Software Engineering Institute, Addison Wesley Professional, the SEI series in software Engineering, 1995.

(Paulson, 1994) Paulson L.C., « *Isabelle: A Generic Theorem Prover* ». Lecture Notes in Computer Science, vol. 828. Springer-Verlag, New York, 1994.

(Penalva, 1993) Penalva J. M., « *SAGACE method: the modelling of human designed systems* ». COMETT'93, Rome, Italie, 1993.

(Petit, 2002) Petit M., « *UEML Project WP1 - Enterprise Modelling State of the Art-* ». 2 Octobre 2002.

(Pingaud, 2005) Pingaud H., « *Modélisation d'entreprise* ». Cours Master Productique de l'Université Paul Sabatier de Toulouse, 2005.

(Pingaud, 2009) Pingaud H., « *Rationalité du développement de l'interopérabilité dans les organisations* ». 8ème congrès international de Génie Industriel, CIGI09', 2009.

(Presson, 1983) Presson P., « *Software Metrics and Interoperability* ». Proceedings of the 4th Computers in Aerospace Conference. Hartford, CT, 24-26 Octobre 1983.

$\mathcal{R}$

(Reisig, 1985) Reisig W., « *Petri Nets : an introduction* ».Springer-Verlag, 161p, Berlin, 1985.

(Roque *et al.*, 2009) Roque M., Chapurlat V., « *Interoperability in collaborative processes: requirements characterisation and proof approach* ».

PRO-VE'09, 10th IFIP Working Conference on VIRTUAL ENTERPRISES, Thessaloniki, Grèce, 7-9 Octobre 2009.

(Ross, 1977) Ross E.T., « *Structured analysis for requirements definition* ». IEEE Transactions on Software Engineering, Volume 3, pp. 6-15, 1977.

(Roschelle *et al.*, 1995) Roschelle J., Teasley S., « *The construction of shared knowledge in collaborative problem solving* ». In Computer supported collaborative learning, C.E. O'Malley (Eds), pp. 69-97, Springer-Verlag, Heidelberg, 1995.

S

(Sallé *et al.*, 2005) Sallé M., Rosenthal S., « *Formulating and Implementing an HP IT Program Strategy Using Cobit and HP ITSM* ». Proceedings of the 38th Hawaii International Conference on System Sciences, Hawaii, 2005.

(Scheer, 1994) Scheer A.W., « *ARIS Toolset: A software product is born* ». Information Systems, volume 19(8), pp. 607-624, 1994.

(Schnoebelen, 2002) Schnoebelen Ph., « *The Complexity of Temporal Logic Model Checking* ». Advances in Modal Logic, Volume 4, pp. 1-44, 2002.

(Scucanec *et al.*, 2008) Scucanec S.J., Van Gaasbeek J.R., « *A day in the life of a verification requirement* ». U.S Air Force T&E Days, Los Angeles, Californie, février 2008.

(Seguy, 2008) Seguy A., « *Décision collaborative dans les systèmes distribués* ». Thèse de doctorat, Université de Toulouse, décembre 2008.

(Simonsson *et al.*, 2006) Simonsson M., Johnson P., « *Assessment of IT Governance – A Prioritization of Cobit* ». Proceedings of the Conference on Systems Engineering Research. Los Angeles, USA, 2006.

(Sowa, 1984) Sowa J.F., « *Conceptual structures: information processing in mind and machine* ». New York (U.S.A.), Addison-Wesley Longman Publishing, 1984.

(Sowa *et al.*, 1992) Sowa J.F., Zachman J.A., « *Extending and Formalising the Framework for Information Systems Architecture* ». IBM Systems Journal, volume 31, n°3, pp. 590 – 616, 1992.

$\mathcal{T}$

(Tardieu *et al.*, 1983) Tardieu, Rochfeld, Colletti, « *La méthode MERISE – tome 1 et 2* », Editions d'organisation, 1983.

(Tolk *et al.*, 2003) Tolk A., Muguira J.A., « *The Levels of Conceptual Interoperability Model* ». Proceedings of Fall Simulation Interoperability Workshop (SIW), Orlando, USA, 2003.

(Tolk *et al.*, 2007) Tolk A., Diallo S.Y., Turnitsa C.D., « *Applying the Levels of Conceptual Interoperability Model in Support of Integratability, Interoperability, and Composability for System-of-Systems Engineering* ». Proceeding Systemics, Cybernetics and informatics, volume 5 - number 5, 2007.

(Touzi, 2007) Touzi J., « *Aide à la conception de Système d'Information Collaboratif : support de l'interopérabilité des entreprises* ». Thèse de doctorat, Institut National Polytechnique de Toulouse, novembre 2007.

(Trinquet, 1997) Trinquet Y., « *Un aperçu sur quelques exécutifs temps réel du marché* ». Ecole d'été, Temps Réel'97 Applications, Réseaux et Systèmes, 1997.

(Turnitsa, 2005) Turnitsa C.D., « *Extending the Levels of Conceptual Inter-operability Model* ». Proceedings IEEE Summer Computer Simulation Conference, IEEE CS Press, 2005.

V

(Vallespir, 2003) Vallespir B., « *Modélisation d'entreprise et architecture de conduite des systèmes de production* ». Habilitation à Diriger les Recherches, Université Bordeaux 1, 19 décembre 2003.

(Van Lamsweerde *et al.*, 1991) Van Lamsweerde A., Dardenne A., Delcourt B., Dubisy F., « *The KAOS Project: Knowledge Acquisition in Automated Specification of Software* ». Proceedings AAAI Spring Symposium Series, American Association for Artificial Intelligence, Stanford University, pp. 59-62, mars 1991.

(Vernadat, 1996) Vernadat, F.B., « *Enterprise Integration Modeling and Integration : Principles and Applications* ». Capman & Hall, London, 1996.

(Vernadat, 1999) Vernadat F.B., « *Techniques de modélisation en entreprise : application aux processus opérationnels* ». Economica, 1999.

(Vernadat, 2002) Vernadat F.B., « *Enterprise modeling and integration (EMI): Current status and research perspectives* ». Annual Reviews in Control, volume 26(1), pp. 15-25, 2002.

W

(Wegner, 1996) Wegner P., « *Interoperability* ». ACM Computing Survey, volume. 28(1), pp 285–287, 1996.

(Williams *et al.*, 2001) Williams T.J., Rathwell G.A., Li H., « *A handbook on master planning and implementation for enterprise integration programs* ». Purdue Laboratory for Applied Industrial Control, 2001.

(Winer *et al.*, 1994) Winer M., Ray K., « *Collaboration handbook: creating, sustaining, and enjoying the Journey* ». Ed. Amherst H. Wilder Foundation, St. Paul, USA, ISBN 0-940069-03-2, 1994.

(Yovine, 1997) Yovine S., « *Kronos: A verification tool for real-time system* ». Springer international journal of Software Tools for technology, octobre 1997.

Z

(Zachman, 1987) Zachman, J., « *A framework for information systems architecture* ». IBM Systems Journal, volume 26, 1987.

(Zeigler *et al.*, 2000) Zeigler B., Kim T. G., Praehofer H., « *Theory of Modeling and Simulation* » Academic Press, New York, 2000.

Annexe A

Enquête sur les besoins de l'entreprise en termes d'interopérabilité

Présentation

L'objectif de cette enquête est de lister et de formaliser les besoins d'une entreprise souhaitant améliorer ses aptitudes et sa capacité à travailler de manière harmonieuse et performante en situation de collaboration c'est-à-dire son interopérabilité:

- D'un point de vue inter entreprise dans le cadre de collaborations avec des partenaires industriels externes.
- D'un point de vue intra entreprise dans le cadre de son fonctionnement interne.

Les réponses à cette enquête permettront d'élaborer une liste de besoins génériques d'interopérabilité. Cette liste servira de référentiel de base à un travail de recherche dont l'objectif est de caractériser, détecter puis analyser des problèmes d'interopérabilité dans l'entreprise. Ces besoins peuvent être de nature :

- **Technique :** communiquer entre des outils supports utilisés dans l'entreprise ou durant la collaboration, …
- **Organisationnel :** partager des données, faire travailler plus efficacement les acteurs et/ou équipes impliquées dans la collaboration ou au sein de l'entreprise, …
- **Conceptuel :** respecter la nécessaire hétérogénéité des métiers et des savoirs faire de l'entreprise, prendre en compte des sémantiques différentes des données, …

Ce questionnaire s'adresse à des responsables d'entreprises et à des responsables fonctionnels (responsables de projets de R&D, de relation client / fournisseur, d'applicatifs, DSI, DRH). Il est **confidentiel** et toutes les données qui y sont relatives feront l'objet d'un **traitement anonyme**.

Responsables de l'étude

Cette étude est menée dans le cadre du travail de Doctorat de Mlle Sihem Mallek (sihem.mallek@mines-ales.fr) sous la direction de M. Nicolas Daclin (nicolas.daclin@mines- ales.fr) et de M.Vincent Chapurlat (vincent.chapurlat@mines-ales.fr)

de l'équipe ISOE (Interoperable System and Organisation Engineering) du Laboratoire de Génie Informatique et d'Ingénierie de Production (LGI2P) de l'Ecole des Mines d'Alès (http://www.ema.lgi2p.fr/).

Renseignements généraux

1. Quel est l'effectif global de l'entreprise ?

   ☐ -de 50 ☐ 51-250 ☐ +250

2. L'entreprise occupe-t-elle plusieurs sites géographiquement distincts ?

   ☐ Oui ☐ Non

   Si oui, la répartition géographique des sites est- elle ?

   ☐ Nationale ☐ Internationale

   Est-ce qu'une langue de travail unique est imposée dans l'entreprise ?

   ☐ Oui ☐ Non

3. Quel est le secteur d'activité de l'entreprise (voir ici la liste des codes APE) ?

4. L'entreprise est-elle certifiée ?

   ☐ Oui ☐ Non ☐ En cours

   Si oui, quel(s) type(s) de certification est (ont été) visé(s) ? Précisez :

Renseignements sur les collaborations inter entreprises

Dans cette partie du questionnaire, l'entreprise peut être vue soit comme **Cliente** d'une autre entreprise, soit comme un **Fournisseur** ou un **Sous-traitant** d'une autre

entreprise, soit, enfin, en tant que **Partenaire** dans le cadre d'un projet commun avec d'autres entreprises.

5. Quels sont pour vous les besoins de l'entreprise pour pouvoir collaborer avec d'autres entreprises (plusieurs réponses sont possibles) ?

☐ **Besoins d'ordre technique** (exemples : utiliser des standards, utiliser des outils communs, imposer un standard, mettre en place des outils de communication, respecter une norme de compatibilité, …). Précisez :

☐ **Besoins d'ordre organisationnel** (exemples : remettre en cause les aptitudes et pratiques courantes, les processus internes, la structure organisationnelle, les compétences des personnes et leurs responsabilités, la structure des équipes, les autorisations ou droits d'accès, améliorer la capacité, prendre en compte une législation du travail différente selon les sites, …). Précisez :

☐ **Besoins d'ordre conceptuel** (exemples : définir et comprendre la sémantique, comprendre un langage donné, utiliser plusieurs langages, respecter des usages pour l'interprétation de données, …). Précisez :

☐ **Autres besoins** : précisez

☐ **Aucun besoin** n'est perçu: selon vous, pourquoi ne percevez-vous pas de besoins ?

6. Pour répondre à ces besoins, quelles modifications ou améliorations de l'organisation, des moyens et/ou du fonctionnement de l'entreprise ont été mises en œuvre ou proposées ? (exemples : formation de personnel, acquisition de

matériels / de logiciels, mise en place d'une démarche de certification, d'une démarche d'amélioration continue, …).

7. Les modifications et améliorations effectivement mises en œuvre ont-elles :

- **Apporté objectivement un gain de performance** durant la collaboration ?

  ☐ Oui ☐ Non

  Si oui, lesquels ?

  ☐ Délais ☐ Coûts ☐ Qualité ☐ Autres: précisez

- **Entrainé une perte de performance de l'entreprise considérée comme acceptable** au regard de la collaboration elle-même ?

  ☐ Oui ☐ Non

  Si oui, lesquels ?

  ☐ Délais ☐ Coûts ☐ Qualité ☐ Autres: précisez

- **Entrainé une perte de performance de l'entreprise considérée comme non acceptable** au regard de la collaboration elle-même ?

  ☐ Oui ☐ Non

  Si oui, lesquels ?

  ☐ Délais ☐ Coûts ☐ Qualité ☐ Autres : précisez

- **Influencé d'autres collaborations dans lesquelles l'entreprise est aussi impliquée** ?

  ☐ Oui ☐ Non

  Si oui, lesquels ?

  ☐ Délais : trouvez-vous que cela a été

  ○ Plutôt bénéfique ○ Bénéfique ○ Plutôt néfaste ○ Néfaste

☐ Coûts : trouvez-vous que cela a été

○ Plutôt bénéfique ○ Bénéfique ○ Plutôt néfaste ○ Néfaste

☐ Qualité : trouvez-vous que cela a été

○ Plutôt bénéfique ○ Bénéfique ○ Plutôt néfaste ○ Néfaste

☐ Autres : précisez

8. **Dans tous les cas précédents**, qu'avez-vous alors constaté comme nouveaux besoins ?

Renseignements sur les besoins intra entreprise

Dans cette partie du questionnaire, nous considérerons qu'une **unité** peut être une équipe, un département ou un processus devant collaborer avec une autre équipe, département ou processus de l'entreprise. De même comme précédemment, l'unité peut être vue soit comme **Cliente** d'une autre unité, soit comme un **Fournisseur** ou un **Sous-traitant** d'une autre unité, soit, enfin, en tant que **Partenaire** dans le cadre d'un projet commun avec d'autres unités de l'entreprise.

9. Quels sont pour vous les besoins de l'unité pour pouvoir collaborer avec d'autres unités (plusieurs réponses sont possibles) ?

☐ **Besoins d'ordre technique** (utiliser des standards, utiliser des outils communs, imposer un standard, mettre en place des outils de communication, respecter une norme de compatibilité, …). Précisez :

☐ **Besoins d'ordre organisationnel** (remettre en cause les aptitudes et pratiques courantes, les processus internes, la structure organisationnelle, les compétences des personnes et leurs responsabilités, la structure des équipes, les autorisations ou droits d'accès, améliorer la capacité, prendre en compte une législation du travail différente selon les sites, …). Précisez :

☐ **Besoins d'ordre conceptuel** (définir et comprendre la sémantique, comprendre un langage donné, utiliser plusieurs langages, connaitre les usages pour l'interprétation de données, des flux, …). Précisez :

☐ **Autres besoins** : précisez

☐ **Aucun besoin** n'est perçu : selon vous, pourquoi ne percevez-vous pas de besoins ?

10. Pour répondre à ces besoins, quelles modifications ou améliorations de l'organisation, des moyens et/ou du fonctionnement de l'unité ont été mises en œuvre ou proposées ? (exemples : formation de personnel, acquisition de matériels / de logiciels, mise en place d'une démarche de certification, d'une démarche d'amélioration continue, …).

11. Les modifications et améliorations effectivement mises en œuvre ont-elles :

- **Apporté objectivement un gain de performance** durant la collaboration ?

☐ Oui ☐ Non

Si oui, lesquels ?

☐ Délais ☐ Coûts ☐ Qualité ☐ Autres : précisez

- **Entrainé une perte de performance de l'unité considérée comme acceptable** au regard de la collaboration elle-même ?

☐ Oui ☐ Non

Si oui, lesquels ?

☐ Délais ☐ Coûts ☐ Qualité ☐ Autres : précisez

- **Entrainé une perte de performance de l'unité considérée comme non acceptable** au regard de la collaboration elle-même ?

☐ Oui ☐ Non

Si oui, lesquels ?

☐ Délais ☐ Coûts ☐ Qualité ☐ Autres : précisez

- **Influencé d'autres collaborations dans lesquelles l'unité est aussi impliquée**

par l'unité dans cette collaboration ?

☐ Oui ☐ Non

Si oui, lesquels ?

☐ Délais : trouvez-vous que cela a été

○ Plutôt bénéfique ○ Bénéfique ○ Plutôt néfaste ○ Néfaste

☐ Coûts : trouvez-vous que cela a été

○ Plutôt bénéfique ○ Bénéfique ○ Plutôt néfaste ○ Néfaste

☐ Qualité : trouvez-vous que cela a été

○ Plutôt bénéfique ○ Bénéfique ○ Plutôt néfaste ○ Néfaste

☐ Autres : précisez

12. **Dans tous les cas précédents**, qu'avez-vous alors constaté comme nouveaux besoins ?

Synthèse

13. Quelles sont les principales problématiques d'interopérabilité que vous percevez dans votre entreprise ?

14. Quelles sont les raisons essentielles qui, pour vous, sont à l'origine de problèmes d'interopérabilité ?

- ☐ Robustesse de l'organisation en place (suffisante / insuffisante)
- ☐ Maturité de l'organisation (suffisante / insuffisante)
- ☐ Refus du changement des acteurs
- ☐ Coûts cachés ou non prévus
- ☐ Risques mal perçus ou mal anticipés
- ☐ Notion de performance non partagée
- ☐ Autres : précisez

Fiche d'identité

L'équipe de recherche vous remercie de votre collaboration. Les résultats de cette enquête vous seront communiqués si vous le souhaitez. Pour cela, merci de remplir le tableau suivant:

Nom Prénom	
Entreprise	
Adresse Messagerie	

Veuillez, s'il vous plaît, envoyer votre questionnaire complété en cliquant sur le bouton ci- dessous ou en nous le faisant parvenir à l'adresse sihem.mallek@mines-ales.fr.

Annexe B

Règles BNF d’écriture des θe et θc

θe	::=	(quantifier)? condition
θc	::=	(quantifier)? condition
quantifier	::=	(‘∀’ | ‘∃’) IDVARIABLE ‘,’ quantifier ?
condition	::=	conditionalAndExpression (assignmentOperator condition)? ;
assignmentOperator	::=	'=' | ':=' ;
conditionalAndExpression	::=	inclusiveOrExpression ('AND' inclusiveOrExpression)* ;
inclusiveOrExpression	::=	exclusiveOrExpression ('OR 'exclusiveOrExpression)* ;
exclusiveOrExpression	::=	andExpression (XOR' andExpression)* ;
andExpression	::=	equalityExpression ('&' equalityExpression)* ;
equalityExpression :	::=	relationalExpression (('==' | '!=') relationalExpression)* ;
relationalExpression	::=	additiveExpression (relationalOp additiveExpression)* ;
relationalOp	::=	'<' ('=')? | '>' ('=')? ;
additiveExpression	::=	multiplicativeExpression (('+' | '-') multiplicativeExpression)* ;
multiplicativeExpression	::=	unaryExpression (('*' | '/') unaryExpression)* ;
unaryExpression	::=	'+' unaryExpression
		| '-' unaryExpression
		| unaryExpressionNotPlusMinus ;
unaryExpressionNotPlus Minus	::=	'(' unaryExpression (conditionalAndExpression)? ')'
		| IDPREDICAT
		| NOMBRE
		| IDVARIABLE
		| 'TRUE’
		| 'FALSE' ;
OPUNARYLOGICALOP	::=	'!' ;
IDPREDICAT	::=	IDVARIABLE '(' IDVARIABLE (',' IDVARIABLE)* ')';
IDVARIABLE	::=	LETTRE_MINUS
		(LETTRE_MINUS
		| LETTRE_MAJUS

| INT

| '_')* ;

IDENSEMBLE ::= LETTRE_MAJUS (LETTRE_MINUS| LETTRE_MAJUS| INT | '_')* ;

NOMBRE ::= ('0' .. '9')+ ('.' ('0' .. '9')+)? EXPONENT? ;

EXPONENT ::= ('e' | 'E') ('+' | '-')? INTETOILE ;

INTETOILE ::= '1'..'9' ('0'..'9')* ;

INT ::= '0'..'9' ('0'..'9')* ;

LETTRE_MINUS ::= ‘a'..'z';

LETTRE_MAJUS ::= 'A'..'Z';

Annexe C

Le référentiel des exigences d'interopérabilité

Exigences de compatibilité

Une exigence de compatibilité est définie comme suite : « *un énoncé qui prescrit une fonction, une aptitude ou une caractéristique - considéré comme invariable durant toute la collaboration - lié aux barrières d'interopérabilité et à chaque niveau d'interopérabilité, que les partenaires doivent satisfaire avant toute collaboration* ».

Au niveau données

Exigences d'ordre conceptuel

Syntaxe :

- $CDC_{Vocabulaire}$: Le récepteur et l'émetteur connaissent le vocabulaire des données.
- $CDC_{ReprésentationDonnées}$: Les données des partenaires sont représentées.

Sémantique :

- $CDC_{MaîtriseSémantique}$: Les partenaires maîtrisent le sens des données.
- $CDC_{DéfinitionDonnées}$: Les données sont définies.
- $CDC_{Documentation}$: Les données sont documentées.
- $CDC_{NonAmbigüité}$: Les données sont non ambigües.

Exigences d'ordre technologique

Plateformes

- $CDT_{OutilsConnexion}$: Les partenaires disposent d'interface de connexion.
- $CDT_{FormatEchange}$: Le format des données est non ambigu.
- $CDT_{FormatTraduction}$: Les partenaires possèdent un format de traduction/mapping.
- $CDT_{PointsConnexion}$: Les bases de données possèdent des points de connexion.
- $CDT_{Traduction}$: Les partenaires disposent des moyens de traduction des données.
- $CDT_{Exploitation}$: Les partenaires disposent des moyens d'exploitation des données.

Exigences d'ordre organisationnel

Personnes

- $CDO_{Confidentialité}$: Les données internes confidentielles sont sécurisées.

- $CDO_{Autorisation}$: Les partenaires possèdent les autorisations requises pour effectuer les échanges.
- $CDO_{AutorisationModification\_Ajout}$: Les partenaires fournissent une autorisation pour l'ajout de données.
- $CDO_{AutorisationModification\_Suppression}$: Les partenaires fournissent une autorisation pour la suppression de données.
- $CDO_{AutorisationModification\_Changement}$: Les partenaires fournissent une autorisation pour la modification de données.
- $CDO_{AutorisationModification\_Miseàjour}$: Les partenaires fournissent une autorisation pour la mise à jour de données.

Structure organisationnelle

- $CDO_{BaseDonnées}$: les données sont regroupées dans une base de données.
- $CDO_{Organisation}$: Les données sont organisées dans une base de données.

Au niveau services

Exigences d'ordre conceptuel

Sémantique

- $CSC_{MaîtriseSémantique}$: Les partenaires maîtrise la sémantique des services.
- $CSC_{SémantiqueNoms}$: les noms des services possèdent une sémantique.
- $CSC_{Description}$: les services sont décrits.

Exigences d'ordre technologique

Plateformes

- $CDT_{PointsConnexion}$: Les services possèdent des points de connexion.

Protocole de communication

- $CDT_{ProtocoleCommunication}$: Un protocole de communication est défini.

Exigences d'ordre organisationnel

Personnes

- $CSO_{Aptitude}$: Une ressource possède l'aptitude requise par le service la sollicitant.

- $CSO_{Fonctions}$: Les fonctions d'un service sont définies.
- $CSO_{AttributionRôles}$: Les rôles de chaque service sont attribués.
- $CSO_{Autorisation}$: Un service émetteur possède les autorisations nécessaires pour échanger avec un service récepteur.
- $CSO_{AutorisationLogin}$: L'émetteur possède un login pour échanger avec le récepteur.
- $CSO_{AutorisationMotpasse}$: L'émetteur possède un mot de passe pour échanger avec le récepteur.
- $CSO_{AutorisationValidité}$: La période de validité des autorisations est définie.
- $CSO_{Responsabilité}$: Chaque service possède un responsable clairement identifié.
    - $CSO_{ResponsabilitéNom}$: Un responsable de service possède un nom.
    - $CSO_{ResponsabilitéPhone}$: Un responsable de service possède un numéro de téléphone.
    - $CSO_{ResponsabilitéMail}$: Un responsable de service possède un email.

Structure organisationnelle

- $CSO_{Structure}$: Un service possède une structure organisationnelle définie.

Au niveau processus

Exigences d'ordre conceptuel

Syntaxe

- $CPC_{Syntaxe}$: La syntaxe des langages de description des processus est connue.
- $CPC_{Représentation}$: Les processus sont représentés.

Sémantique

- $CPC_{Sémantique}$: Les partenaires maîtrise la sémantique des langages de description des processus.

Exigences d'ordre organisationnel

Personnes

- $CPO_{Procédures}$: Les processus disposent de procédures de travail.
- $CPO_{ModeOpération}$: Les processus disposent de modes d'opération.

Structure organisationnelle

- $CPO_{Stratégies}$: Les processus ont une stratégie.

Au niveau business

Exigences d'ordre organisationnel

Personnes

- $CBO_{Décision}$: Le responsable est le seul habilité à prendre des décisions.
- $CBO_{MéthodesTravail}$: Les méthodes de travail sont définies.
- $CBO_{Législation}$: Les aspects légaux sont connus
- $CBO_{Fiscalité}$: Les aspects fiscaux sont connus.

Structure organisationnelles

- $CBO_{Structure}$: Les structures décisionnelles sont définies.

Exigences d'interopération

Une exigence d'interopération est définie comme suite : « *un énoncé qui prescrit une fonction, une aptitude ou une caractéristique – considéré comme variable pendant la collaboration - lié aux interactions entre des partenaires et que ceux-ci doivent satisfaire* ».

Au niveau données

Exigences d'ordre conceptuel

Temps

- $IDC_{TempsExploitation}$: Le temps pour l'exploitation effectif est inférieur ou égal au temps d'exploitation défini.

Coût

- $IDC_{CoûtExploitation}$: Le coût effectif d'exploitation des données échangées est inférieur ou égal au coût défini.

Qualité

- $IDC_{Conformité}$: Les données reçues sont conformes à celles requises.
- $IDC_{Complétude}$: Les données reçues sont complètes.

- $IDC_{DistributionDonnées}$: Un espace d'information distribué est mis en place.

Exigences d'ordre technologique

Temps

- $IDT_{TempsExploitation}$: Le temps pour l'exploitation effectif est inférieur ou égal au temps d'exploitation défini.

Coût

- $IDT_{CoûtExploitation}$: Le coût effectif d'exploitation des données échangées est inférieur ou égal au coût défini.

Qualité

- $IDT_{Débit}$: Le débit de transfert est suffisant.
- $IDT_{Quantité}$: La quantité de données reçue est égale à la quantité de données requise.
- IDTProtocole : Un protocole commun existe pour structurer les données.

Exigences d'ordre organisationnel

Temps

- $IDO_{Disponibilité}$: Les données à échanger sont disponibles.
- $IDO_{Adresse}$: L'émetteur a connaissance du récepteur.

Qualité

- $IDO_{DroitAccès}$: Les données échangées sont accessibles uniquement par les personnes autorisées.
- $IDO_{PartageDonnées}$: Les données sont partagées entre les partenaires.

Au niveau services

Exigences d'ordre technologique

Temps

- $IST_{TempsEchange}$: Le temps d'échange est inférieur ou égal au temps défini.
- $IST_{TempsExploitation}$: Le temps pour l'exploitation effectif est inférieur ou égal au temps d'exploitation défini.

- $IST_{partageApplication}$: Les applications sont partagées.

Coût

- $IST_{CoûtsEchange}$: Le coût d'échange est inférieur ou égal au coût défini.

Exigences d'ordre organisationnel

Temps

- $ISO_{TempsExecution}$: Chaque service utilise la ressource nécessaire pendant le temps d'exécution de ce service.
- $ISO_{TempsDisponibilité}$: Une ressource est disponible lorsqu'un service la sollicite.
- $ISO_{TempsExploitation}$: Le temps pour l'exploitation effectif est inférieur ou égal au temps d'exploitation défini.

Qualité

- $ISO_{AccuséRéception}$: Le récepteur accuse réception à l'émetteur.

Au niveau processus

Exigences d'ordre organisationnel

Qualité

- $IPO_{Procédure}$: Une procédure commune est définie.

Au niveau business

Exigences d'ordre organisationnel

Qualité

- $IBO_{PérimètreDécisionnel}$: Les périmètres décisionnels sont délimités.
- $IBO_{Stratégie}$: Une stratégie commune est définie.
- $IBO_{MéthodeTravail}$: Une méthode de travail commune est définie.

Exigences de réversibilité

Une exigence de réversibilité est définie comme suite : « *un énoncé qui prescrit une fonction, une aptitude ou une caractéristique - lié à la capacité qu'a un partenaire donné à revenir à son état initial - que l'entreprise doit satisfaire à la fin de la collaboration* ».

Au niveau données

Exigences d'ordre conceptuel

Intégrité

- $RDC_{qualité}$: Le sens des données partagées n'est pas altéré.
- $RDC_{TempsExploitation}$: Les données restent compréhensibles, malgré une modification.

Exigences d'ordre organisationnel

Intégrité

- $RDO_{BaseDonnée}$: L'organisation des données au sein d'une base de données n'est pas altérée.

Au niveau services

Exigences d'ordre technologique

Performance

- $RST_{PerformanceCoût}$: Le coût d'une activité *post*-collaboration correspond – tolérances comprises - au coût *ante*-collaboration.
- $RST_{PerformanceTemps}$: Le temps d'exécution d'une activité *post*-collaboration correspond – tolérances comprises - au temps d'exécution *ante*-collaboration.
- $RST_{PerformanceQualité}$: Le nombre de produits conformes fournit par une activité *post*-collaboration correspond - tolérances comprises - au nombre de produits conformes *ante*-collaboration.

Exigences d'ordre organisationnel

Intégrité

- $RSO_{Ressource}$: Le service rempli ses fonctions malgré la perte d'une ressource.

- $RSO_{Structure}$: Le service retrouve sa structure d'origine.

Au niveau processus

Exigences d'ordre organisationnel

Intégrité

- $RPO_{Mission}$: Le processus réalise sa mission malgré la perte d'un service.

Au niveau business

Exigences d'ordre organisationnel

Stabilité

- $RBO_{MéthodeTravail}$: Les méthodes de travail adoptées avant la collaboration sont reprises, même si elles ont été perturbées au cours de la collaboration.

Exigences d'autonomie

Une exigence de réversibilité est définie comme suite : « *un énoncé qui prescrit une fonction, une aptitude ou une caractéristique, lié à la capacité qu'ont les partenaires à garder leurs propres décisions et leurs propres opérationnalités, tout au long de la collaboration, que les partenaires doivent satisfaire.* »

Au niveau services

Exigences d'ordre organisationnel

Opérationnelle

- $ASO_{Ressource}$: Un service peut remplacer une ressource utilisée par un autre service du processus collaboratif public.

Au niveau processus

Exigences d'ordre technologique

Opérationnelle

- $APT_{Opérationnelle}$: Un processus garde un flux physique d'approvisionnement.

Au niveau business

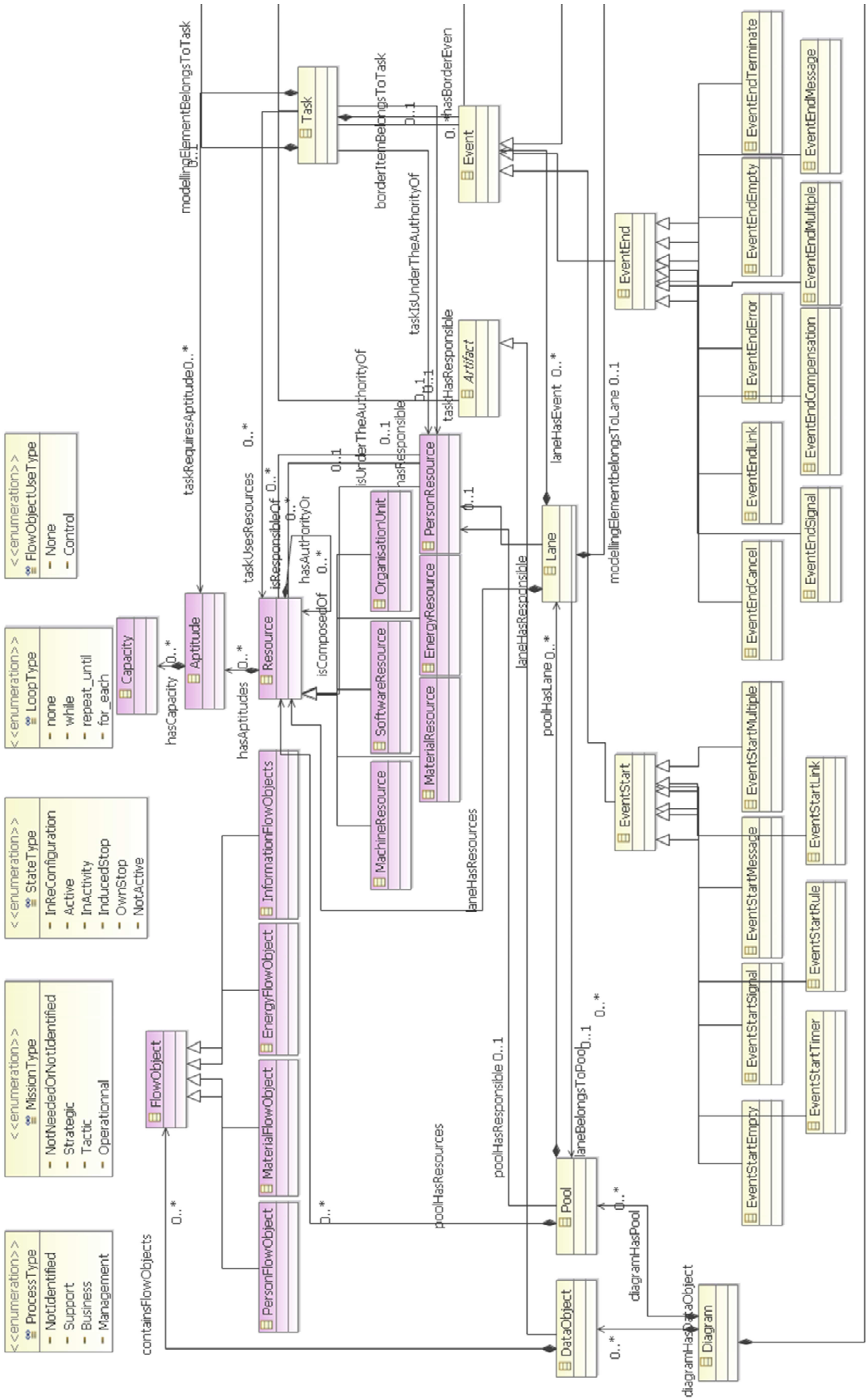
Exigences d'ordre organisationnel

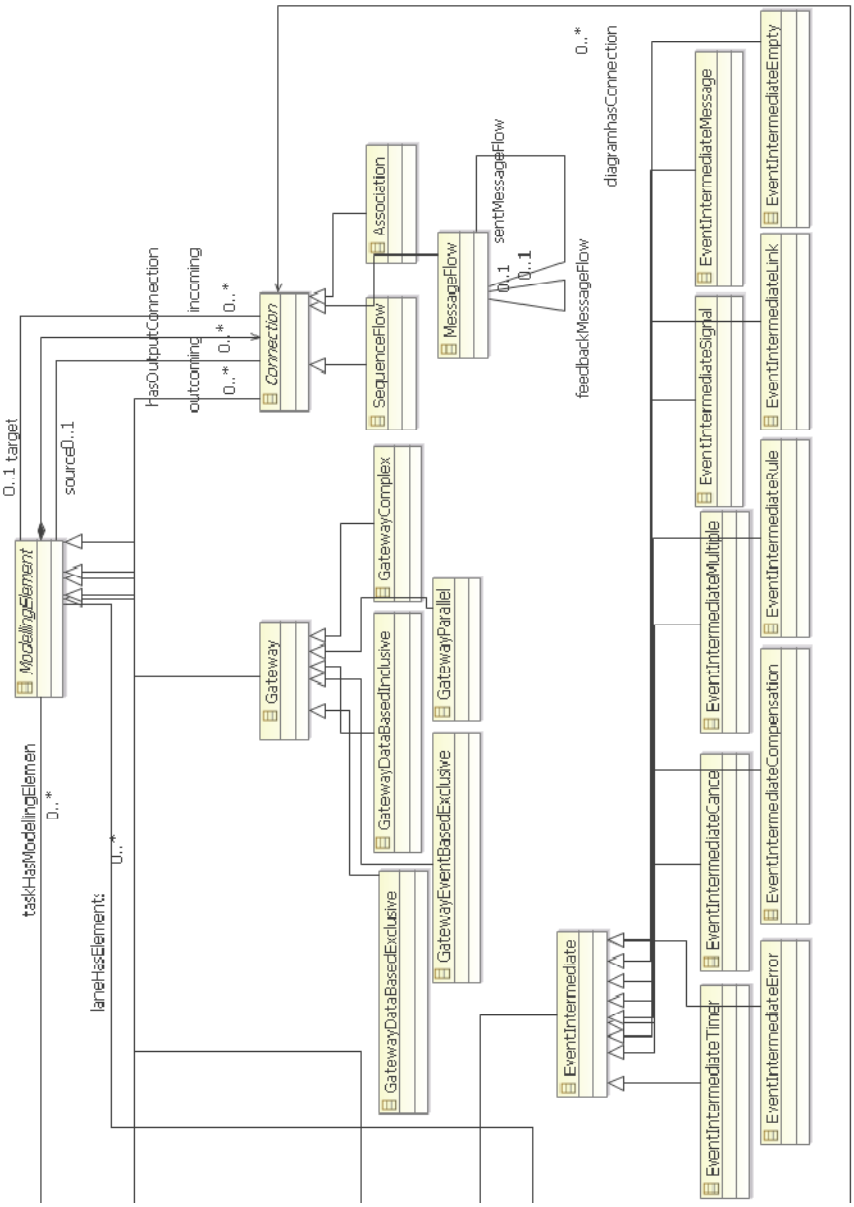
Gouvernance

- $ABO_{Gouvernance}$: Un partenaire du processus collaboratif reste responsable du choix de ses fournisseurs.
- $ABO_{Gouvernancecoût}$: Le partenaire maîtrise les coûts relatifs au choix des fournisseurs.
- $ABO_{Stratégie}$: Les stratégies propres d'un processus ne sont pas altérées par les stratégies communes.

Annexe D

Le méta modèle de BPMN enrichi


<<enumeration>>
ProcessType
- NotIdentified
- Support
- Business
- Management
<<enumeration>>
MissionType
- NotNeededOrNotIdentified
- Strategic
- Tactic
- Operationnal
<<enumeration>>
StateType
- InReConfiguration
- Active
- InActivity
- InducedStop
- OwnStop
- NotActive
<<enumeration>>
LoopType
- none
- while
- repeat_until
- for_each
<<enumeration>>
FlowObjectUseType
- None
- Control
FlowObject
PersonFlowObject
MaterialFlowObject
EnergyFlowObject
InformationFlowObjects
Capacity
Aptitude
Resource
PersonResource
EnergyResource
MaterialResource
SoftwareResource
MachineResource
OrganisationUnit
Task
Event
Artifact
Lane
Pool
DataObject
Diagram
EventStart
EventStartMultiple
EventStartMessage
EventStartSignal
EventStartTimer
EventStartEmpty
EventStartLink
EventStartRule
EventEnd
EventEndTerminate
EventEndEmpty
EventEndMessage
EventEndMultiple
EventEndError
EventEndCompensation
EventEndLink
EventEndSignal
EventEndCancel
containsFlowObjects
modellingElementBelongsToTask
taskRequiresAptitude
taskUsesResources
isResponsibleOf
hasAuthorityOf
isComposedOf
hasCapacity
hasAptitudes
isUnderTheAuthorityOf
hasResponsible
taskIsUnderTheAuthorityOf
taskHasResponsible
borderItemBelongsToTask
hasBorderEven
laneHasEvent
modellingElementBelongsToLane
laneHasResponsible
poolHasLane
laneBelongsToPool
poolHasResponsible
poolHasResources
laneHasResources
diagramHasPool
diagramHasDataObject


ModellingElement
Connection
Association
SequenceFlow
MessageFlow
sentMessageFlow
feedbackMessageFlow
diagramhasConnection
HasOutputConnection
incoming
outcoming
target
source
taskHasModellingElemen
laneHasElement
Gateway
GatewayComplex
GatewayDataBasedInclusive
GatewayParallel
GatewayDataBasedExclusive
GatewayEventBasedExclusive
EventIntermediate
EventIntermediateMessage
EventIntermediateSignal
EventIntermediateMultiple
EventIntermediateRule
EventIntermediateLink
EventIntermediateEmpty
EventIntermediateCompensation
EventIntermediateCance
EventIntermediateTimer
EventIntermediateError

Annexe E

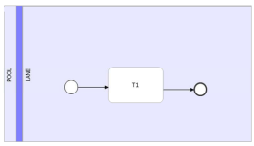
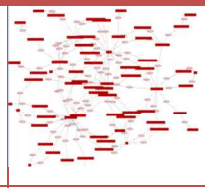
Jeux de test pour valider les transformations

Transformation de BPMN enrichi vers un graphe conceptuel

Afin de valider les transformations effectuées pour passer d'un modèle de processus collaboratif, modélisé avec BPMN enrichi, vers un modèle en graphe conceptuel, nous avons eu recours à des jeux de test.

L'outil utilisé pour effectuer la vérification n'offre pas d'éditeur graphique, de ce fait, nous avons eu recours à l'outil COGUI[25] qui nous permet de voir l'aspect graphique d'un modèle de processus collaboratif sous forme de graphe conceptuel (aussi appelé modèle des faits), obtenu après transformation. L'objectif, ici, est de s'assurer de la présence, dans le modèle des faits, de l'ensemble des concepts et des relations utilisés dans le modèle du processus collaboratif. En d'autres termes, il s'agit de s'assurer que le modèle de processus collaboratif en graphes conceptuels est équivalent au modèle de processus collaboratif en BPMN enrichi. De ce fait, une première vérification peut être effectuée de manière visuelle directement sur le modèle des faits pour vérifier la présence des concepts et des relations du modèle BPMN. Cependant, cette vérification visuelle peut être très longue sur des modèles importants. En conséquence, il est proposé d'effectuer ces tests à l'aide de « query » avec l'outil COGUI. Une query permet, à l'aide du mécanisme de projection, de s'assurer de la présence (ou non) de concepts, de relation ou encore d'un graphe, sur le modèle de faits. A titre d'exemple, le tableau suivant présente un jeu de test, par le biais de la projection d'une query, pour montrer l'équivalence d'un modèle de processus collaboratif en BPMN enrichi et en graphes conceptuels.

Tableau 9. Jeu de test sous COGUI

Jeux de test	Transformations	Query	Remarques
			• 3 concepts présents • 2 relations présentes • La tâche existe bien sur le modèle des faits et a comme nom T1

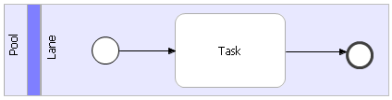
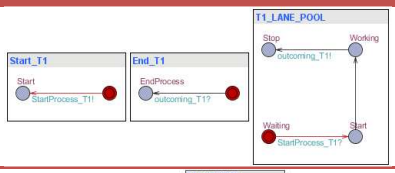
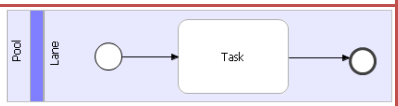
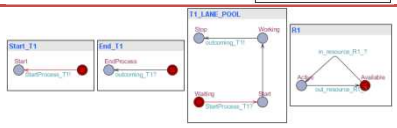
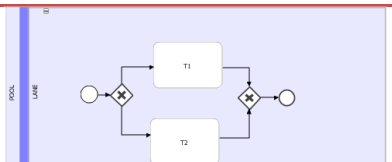
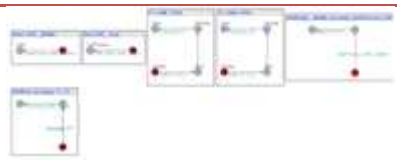
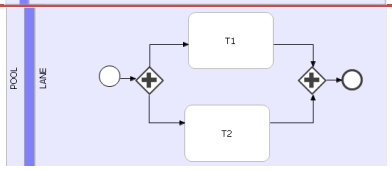
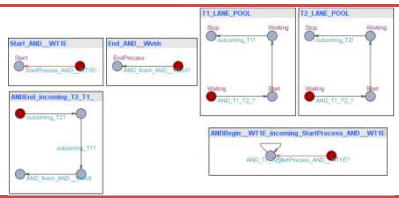
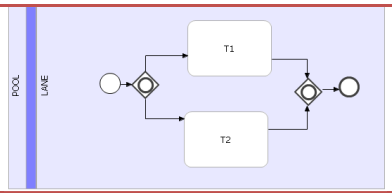
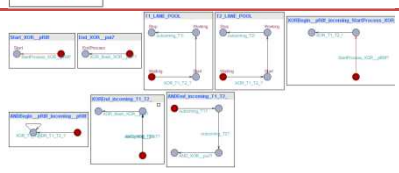
[25] COGUI Version 1.2. Tutoriel disponible sur : http://www.lirmm.fr/cogui/tutorial.php (accessible à la date du 11 janvier 2012)

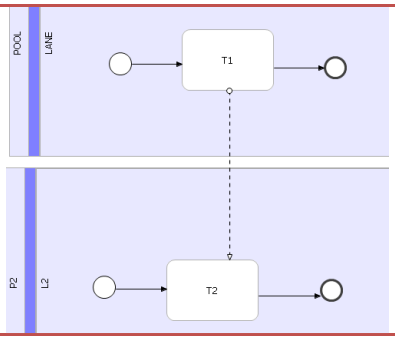
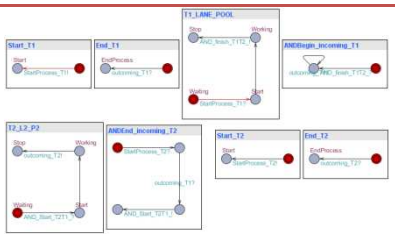
Transformation de BPMN enrichi vers un réseau d'automates temporisés

Le *model checker* UPPAAL possède un éditeur et un simulateur. Le simulateur nous offre la possibilité d'effectuer des tests en forme de jeux d'essai afin de voir si le modèle équivalent de chaque entité sous BPMN enrichi représente bien le comportement attendu de ce dernier. En prenant en considération l'hypothèse simplificatrice 2 qui oblige la présence d'un « *start event* » et d'un « *end event* » pour chaque processus collaboratif, des jeux de test sont effectués et présentés en Tableau 10.

Pour chaque jeu de test, nous nous intéressons (1) au nombre d'automates (templates) obtenu qui doit correspondre au nombre d'éléments présents dans le modèle du processus collaboratif, (2) au blocage du processus, qui ne doit se bloquer qu'à la fin du processus et enfin (3) au comportement que doit avoir le processus sous UPPAAL.

Tableau 10. Jeux de test sous UPPAAL

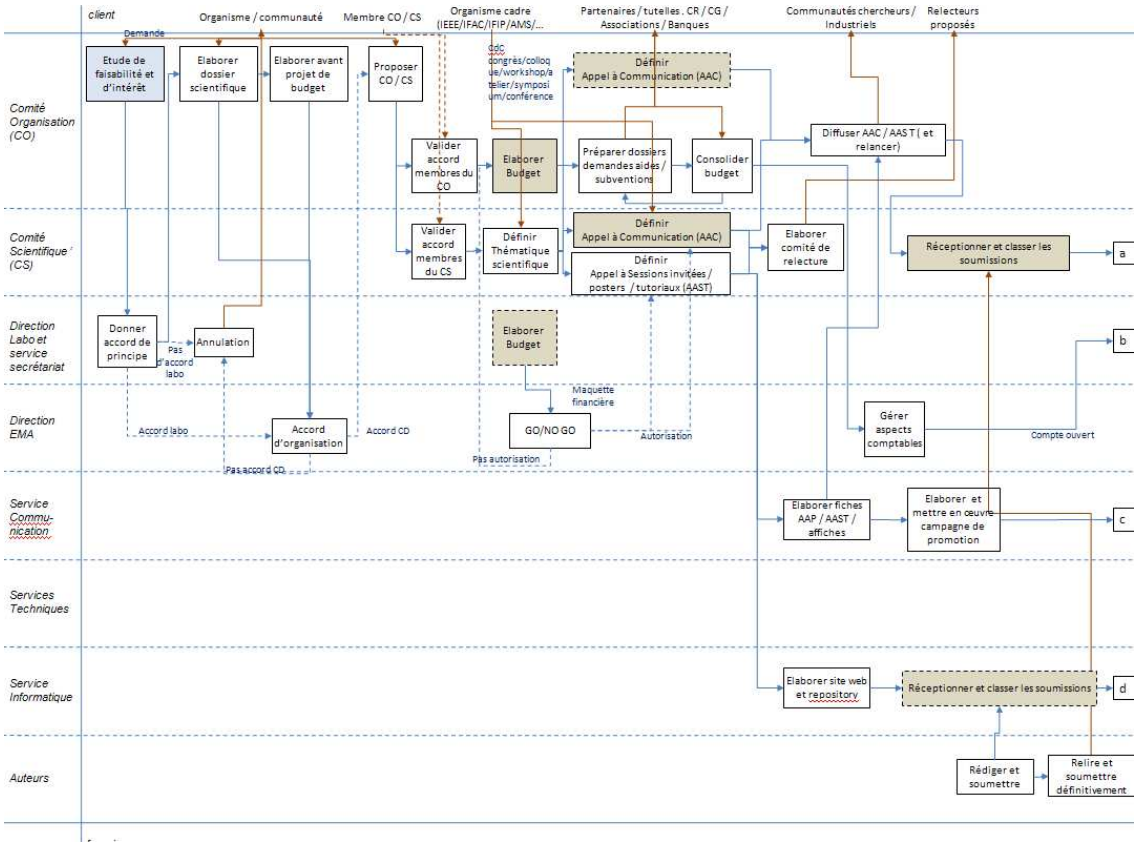
Jeux de test	Transformations	Remarques
		• 3 templates. • Blocage à la fin. • Start → Tâche → End
		• 4 templates (ressource). • Blocage à la fin. • Start→Tâche→Ressource→ Tâche→End
		• 6 templates. • Blocage à la fin. • Start→(T1ouT2)→End
		• 6 templates. • Blocage à la fin. • Start→(T1etT2) →End
		• 8 templates. • Blocage à la fin. • Start→(T1ouT2ou(T1etT2))

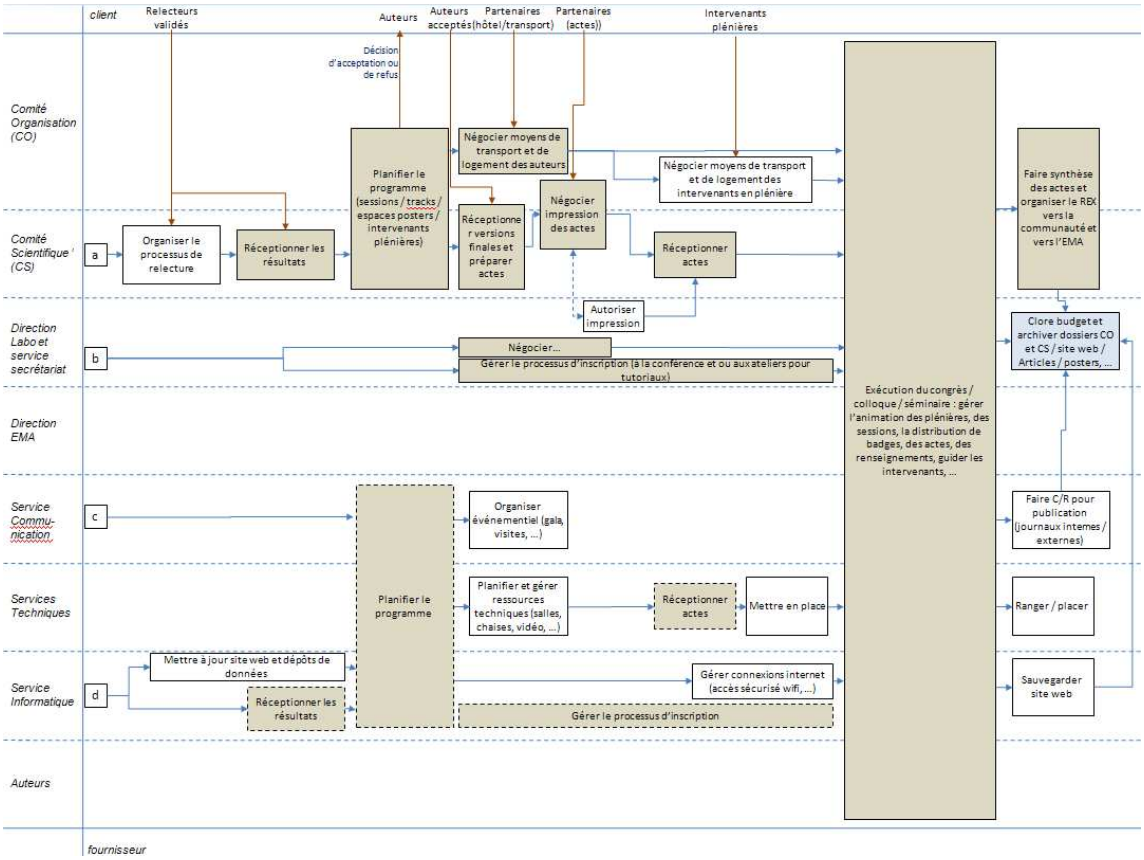
		→End
		• 8 templates. • Blocage à la fin. • Start→T1→(T2 et End) • (Start et T1)→T2→End

Annexe F

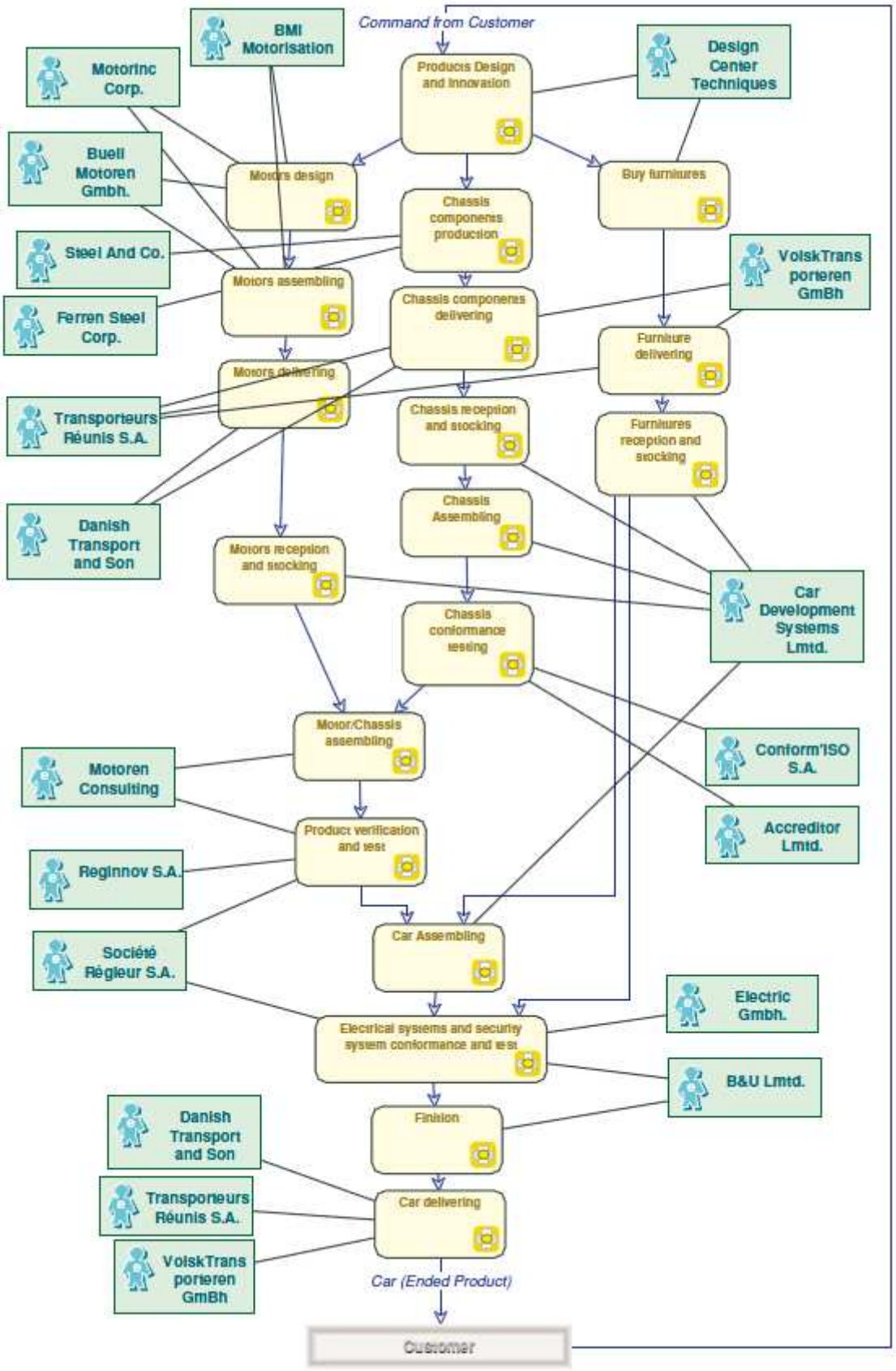
Les modèles des cas d’application

Cas d'application 1 : préparation d'une conférence nationale





Cas d'application 2 : conception et production d'un véhicule



Résumé

L'interopérabilité revêt un enjeu majeur pour l'industrie et son absence peut être vue comme un des principaux freins à un travail collaboratif, et plus particulièrement dans les processus collaboratifs aussi bien publics (inter-entreprises) que privé (intra-entreprise). Il parait donc pertinent d'analyser et de détecter d'éventuels manques ou défauts d'interopérabilité dans des entreprises impliquées dans un processus collaboratif. Les recherches en interopérabilité ont montré l'intérêt de mesurer et d'évaluer l'interopérabilité avec la proposition de cadres et de modèles de maturité dans le but d'éviter d'éventuels problèmes d'interopérabilité. Cependant, des approches de détection et d'anticipation de problèmes d'interopérabilité n'existent pas à notre connaissance. Les travaux de recherche proposés dans cette thèse se développent dans un contexte d'ingénierie de processus guidée par les modèles et se proposent d'utiliser des techniques de vérification formelle pour détecter différents types de problèmes ou de présomption de problèmes d'interopérabilité. Ceci implique, dans un premier temps, de définir les besoins particuliers en interopérabilité devant être pris en compte dans un contexte collaboratif. Dans un second temps, il est nécessaire de formaliser ces besoins en un ensemble d'exigences d'interopérabilité, de manière aussi formelle que possible. Ceci a abouti à quatre classes d'exigences d'interopérabilité respectant le cycle de vie d'un processus collaboratif : les exigences de compatibilité, les exigences d'interopération, les exigences d'autonomie et les exigences de réversibilité. Enfin, ces exigences doivent être vérifiées en se référant aux modèles du ou des processus étudiés.

Mots clés : interopérabilité, exigences d'interopérabilité, compatibilité, interopération, autonomie, réversibilité, vérification, processus collaboratif.

Abstract

Interoperability is becoming a crucial issue for industry and a lack of interoperability can be seen as an important barrier to a collaborative work, particularly on collaborative process both public (inter-enterprise) and private (intra-enterprise). Indeed, interoperability characterises the ability of any enterprises to interact within a collaborative process. Prior to any effective collaboration, it is necessary to inform enterprises that aim to work together, if they are able to interoperate. Researches in interoperability have shown the benefits of having a measurement and an evaluation of interoperability by the proposition of several frameworks and maturity models. However, approaches to detect and anticipate interoperability problems do not exist as we know. The research works proposed in this thesis are developed in a context of process engineering-driven models and propose to use formal verification techniques to detect different types of problems or suspected problems of interoperability. On the one hand, this induces to be able to define the particular needs of interoperability to consider. On the other hand, it requires to formalise these needs as a set of unambiguous and, as formal as possible, requirements. Four classes of interoperability requirements are defined: compatibility requirements, interoperation requirements, autonomy requirements and reversibility requirements. Finally, interoperability requirements must be checked thanks to target process model.

Keywords: interoperability, interoperability requirements, compatibility, interoperation, autonomy, reversibility, verification, collaborative process.