Détection et résolution de conflits d'autorité dans un système homme-robot

Stéphane Mercier* — **Catherine Tessier*** — **Frédéric Dehais****

* *ONERA, 2 avenue Edouard Belin, F-31055 Toulouse Cedex 4*
stephane.mercier@onera.fr, catherine.tessier@onera.fr

** *ISAE, 10 avenue Edouard Belin, F-31055 Toulouse Cedex 4*
frederic.dehais@isae.fr

RÉSUMÉ. *Dans le cadre de missions réalisées conjointement par un agent artificiel et un agent humain, nous présentons un contrôleur de la dynamique de l'autorité, fondé sur un graphe de dépendances entre ressources contrôlables par les deux agents, dont l'objectif est d'adapter le comportement de l'agent artificiel ou de l'agent humain en cas de conflit d'autorité sur ces ressources. Nous définissons l'autorité relative de deux agents par rapport au contrôle d'une ressource, ainsi que la notion de conflit d'autorité : une première expérience nous montre en effet que le conflit constitue un déclencheur pertinent pour une redistribution de l'autorité entre agents. Une seconde expérience montre qu'au-delà de la modification du comportement de l'agent artificiel, il est effectivement possible d'adapter le comportement de l'opérateur humain en vue de résoudre un tel conflit.*

ABSTRACT. *In the context of missions accomplished jointly by an artificial agent and a human agent, we focus on a controller of the authority dynamics based on a dependence graph of resources that can be controlled by both agents. The controller is designed to adapt the behaviours of the artificial agent or of the human agent in case of an authority conflict occurring on these resources. The relative authority of two agents regarding the control of a resource is defined so as the authority conflict, which appears relevant to trigger authority reallocation between agents as shown by a first experiment. Finally a second experiment shows that beyond the modification of the artificial agent's behaviour, it is indeed possible to adapt the human operator's behaviour in order to solve such a conflict.*

MOTS-CLÉS : *initiative mixte, partage d'autorité, interaction homme-robot, réseaux de Petri.*

KEYWORDS: *mixed-initiative, authority sharing, human-robot interactions, Petri nets.*

1. Introduction

Nous considérons un système d'agents hétérogènes dans lequel un agent robotique (robot terrestre d'exploration ou de surveillance, drone) ou logiciel (pilote automatique d'un avion) réalise une mission en interaction avec un agent humain (opérateur, pilote). Au cours de la mission, les agents se voient attribuer ou prennent l'autorité (Hardin *et al.*, 2009) pour utiliser une ressource, réaliser une tâche, satisfaire un but : par exemple, un pilote automatique d'avion en mode *Vertical Speed* a l'autorité sur la chaîne de commande des gouvernes pour atteindre une altitude de consigne tout en garantissant certaines performances ; si l'équipage déconnecte le pilote automatique, c'est l'équipage qui a l'autorité sur la chaîne de commande pour atteindre ce but. Un changement dans l'affectation de l'autorité peut être prévu dans les procédures, le plan de mission ; ou bien il peut se produire de manière imprévue :

– l'opérateur humain reprend en main une tâche contrôlée par l'agent artificiel (logiciel ou robotique) parce qu'il constate une défaillance, un obstacle, un danger imminent, ou pour toute raison qui lui est propre et n'est pas forcément exprimée ;

– l'agent artificiel reprend en main une tâche contrôlée par l'opérateur parce que l'action de l'opérateur viole certaines contraintes (sortie de domaine de vol, collision probable avec un obstacle, etc.) ou parce que les communications avec l'opérateur sont rompues ;

– plus aucun agent n'exerce l'autorité : par exemple, le pilote automatique s'est déconnecté et l'équipage, qui n'en est pas conscient, ne pilote pas (Mumaw *et al.*, 2001).

Le problème posé par ces changements imprévus dans l'affectation de l'autorité est le risque de conflit dans l'état du système, dû au fait que le plan prévu pour l'agent artificiel n'est plus suivi, que l'opérateur a une conscience de situation (Wickens, 2008) erronée, voire les deux. Ceci est illustré par l'expérience suivante, réalisée à l'ISAE Supaéro, sur une mission de recherche et d'identification de cibles par un robot terrestre et un opérateur humain qui n'est pas en vue directe du robot. Au cours de la mission, l'opérateur doit, *via* son écran d'interface, piloter le robot manuellement et finement afin d'identifier des cibles. Alors que l'opérateur prend en main le robot pour cette inspection, celui-ci entame une procédure autonome de retour à la base, déclenchée par le capteur de décharge de la batterie (aléa simulé par une interface de magicien d'Oz [1]). Cet événement est perceptible pour l'opérateur sur son interface *via* trois alertes : l'icône représentant la batterie passe de vert à orange, le mode de pilotage clignote deux fois de « manuel » à « supervisé » et le synoptique affiche en vert « retour base ». Toutefois, comme cet aléa se produit à un moment crucial de la mission où l'opérateur est particulièrement focalisé sur sa tâche d'identification en mode manuel, il est attendu qu'il ne perçoive pas ces changements d'état et que chacun des agents (opérateur et robot) « s'entête » à poursuivre son but (identifier la cible et retourner à la base, respectivement).

1. Interface permettant de déclencher des événements à l'insu de l'opérateur.

Sujet	Durée conflit (en s.)	Résolution	Alertes regardées
Dasje	6	oui	supervisé, retour base, batterie
Deffra	50	non	supervisé
Dupni	50	non	supervisé, retour base
Gatthi	50	non	supervisé, retour base
Guiju	50	non	-
Guiny	50	non	batterie
Hosal	18	oui	retour base, batterie
Jacchi	50	non	supervisé
Nival	10	oui	supervisé, batterie
Peich	29	non	retour base
Penju	50	non	supervisé
Pense	35	oui	supervisé, batterie
Rojan	50	non	-
Schpa	50	non	-

Tableau 1. *Comportement des participants pendant le conflit d'autorité*

Le tableau 1 résume les résultats d'oculométrie de quatorze participants à l'expérience [2] analysés à l'aide du logiciel Eye tech Lab. La première colonne est l'identifiant du participant, la deuxième indique la durée du conflit, la troisième si la raison du conflit a été comprise et la quatrième si les trois alertes nécessaires à la compréhension du conflit ont été regardées au moins une fois. Par exemple, le participant Dasje a mis 6 secondes pour détecter et comprendre le conflit, il a porté au moins une fois son regard sur le mode supervisé, sur le synoptique indiquant le retour base et sur l'état de la batterie. Le participant Dupni n'a pas compris le conflit après 50 secondes, bien que son regard se soit posé sur l'état du mode « supervisé » et sur « retour base », en revanche il n'a jamais posé son regard sur l'état de la batterie. Les résultats de cette expérimentation sont conformes à nos hypothèses puisque le conflit d'autorité a conduit 10 sujets sur 14 à persévérer à tort dans l'identification de la cible. L'analyse du comportement oculaire des 4 participants qui ont réussi à interpréter la situation révèle que ceux-ci ont au moins regardé les couples d'information (« état de la batterie », mode « supervisé ») ou (« état de la batterie », « retour base ») si ce n'est le triplet d'information (« état de la batterie », mode « supervisé », « retour base »). Enfin, il convient de noter que le robot a également continué son plan de retour à la base sans prendre en compte le conflit avec l'opérateur.

Le travail présenté ici a pour objectif de concevoir et de mettre en œuvre au sein de l'architecture de l'agent artificiel un *contrôleur de la dynamique de l'autorité* pour :

2. Les sujets sont équipés d'un oculomètre de type Pertech (25Hz) afin d'analyser leur comportement oculaire (où se pose leur regard).

– détecter les conflits liés à un changement d'autorité imprévu ;

– identifier les conséquences de ces conflits sur la mission ;

– adapter les plans des agents et si nécessaire, l'affectation de l'autorité, pour « aller dans le sens » du changement d'autorité ou au contraire « contrer » ce changement d'autorité si certaines contraintes risquent d'être violées ; l'adaptation comprend l'envoi d'informations ou de recommandations à l'opérateur humain.

Nous supposons que l'agent artificiel est équipé d'une fonction de planification (ou d'une bibliothèque de procédures) et d'une fonction de suivi de situation (ou d'estimation d'état). Le contrôleur de la dynamique de l'autorité sera fondé sur des éléments objectifs de la mission (ressources physiques et logicielles, tâches, buts, contraintes), que nous appelons *ressources*, sur lesquelles nous définissons l'autorité d'un agent relativement à un autre agent. L'état de ces ressources est une abstraction de l'état du système estimé par la fonction de suivi de situation. Le plan de mission (ou la procédure) est abstrait sous la forme d'un *graphe de ressources* au sein duquel les conflits liés à un changement d'autorité imprévu vont apparaître. La résolution d'un conflit consistera à modifier le graphe de ressources – *via* une modification du plan ou de la procédure – voire à modifier l'autorité relative des agents sur un sous-ensemble de ressources.

La section 2 propose une structuration de la littérature relative à l'autonomie et à son contrôle et explique pourquoi nous préférons parler d'*autorité*. La section 3 introduit les concepts de ressource, de graphe de ressources et d'autorité, qui constituent la base du modèle formel présenté dans la section 4. Cette modélisation permet de définir formellement ce qu'est un conflit dans la section 5, ce qui nous conduit à présenter dans la section 6 les méthodes de résolution de conflits proposées. Enfin, la section 7 présente les expérimentations réalisées afin de valider l'utilisation du conflit comme déclencheur de l'adaptation du comportement des agents ainsi qu'une première approche pour réaliser cette adaptation.

2. Autonomie et contrôle de l'autonomie

Dans le cadre du contrôle d'un agent robotique (robot, drone) évoluant dans le monde physique et supervisé par un opérateur humain, l'autonomie se caractérise par la capacité de l'agent robotique à minimiser le besoin de supervision et à évoluer seul dans son environnement (Schreckenghost *et al.*, 1998). Considérant l'autonomie comme une relation (Castelfranchi *et al.*, 2003; Brainov *et al.*, 2003) entre un agent artificiel et un agent humain, l'autonomie totale est le cas extrême consistant précisément à réduire cette relation au minimum. Cependant, dans la pratique, les objectifs d'une mission ne peuvent pas toujours être atteints de façon purement autonome par un agent artificiel et l'intervention de l'opérateur est nécessaire. (Goodrich

et al., 2001; Goodrich *et al.*, 2007) illustrent ce point en rapprochant le temps pendant lequel un robot peut agir sans intervention de l'opérateur et ses performances sur le terrain : celles-ci déclinent rapidement lorsque le robot est négligé par l'opérateur. Un équilibre doit donc être trouvé entre le contrôle purement manuel, qui permet en général d'avoir une grande confiance dans le système mais qui soumet l'opérateur humain à une charge de travail importante, et l'autonomie totale, qui décharge l'opérateur mais offre moins de garanties en environnement incertain (Brookshire *et al.*, 2004). Ainsi, de façon à tirer parti des compétences complémentaires des agents humain et robotique, la variation de l'autonomie de l'agent robotique est largement considérée.

Les *approches évaluatives* proposent une mesure *a posteriori* de l'autonomie d'un agent robotique qui réalise une mission particulière : on peut citer par exemple MAP (Hasslacher *et al.*, 1996), ACL (Clough, 2002) ou Alfus (Huang *et al.*, 2005). Cette dernière propose d'évaluer l'autonomie suivant trois dimensions : complexité de mission, difficulté environnementale et interface avec l'opérateur. Cette méthode présentée comme objective nécessite cependant d'agréger sur chaque axe les résultats d'un grand nombre de métriques hétérogènes, ce qui pose le problème de la sémantique du résultat.

Les *approches prescriptives* s'intéressent à la conception de systèmes intégrant le contrôle de l'opérateur, en posant les questions de la définition des niveaux d'autonomie et du passage d'un niveau à un autre. On peut ranger ces approches selon plusieurs axes : l'*objet* sur lequel porte l'autonomie, le *mode d'initiative* adopté pour faire évoluer l'autonomie, les *critères* pour décider de cette évolution, la *perception* qu'a l'agent artificiel de l'agent humain.

2.1. *L'objet de l'autonomie*

Deux objets principaux sont considérés : le *rôle* et la *tâche*.

Un *rôle* est défini comme un ensemble de tâches à assurer par un agent donné (Hardin *et al.*, 2009). Les spécifications selon des niveaux d'autonomie correspondent alors à une répartition pré-établie des rôles entre l'agent humain et l'agent artificiel : (Sheridan *et al.*, 1978) avaient proposé dès 1978 une classification de l'autonomie opérationnelle (« automation ») d'un système robotique sur dix niveaux. Cependant ce modèle est abstrait et ne tient pas compte de la complexité de l'environnement dans lequel évolue le robot, ni du contexte de mission. D'autres classifications reposent sur le même principe, notamment (Dorais *et al.*, 1999), pour lesquels un niveau d'autonomie est caractérisé par la complexité des commandes traitées, (Goodrich *et al.*, 2001) pour qui un niveau représente la capacité d'un robot à fonctionner un certain temps indépendamment de l'opérateur ou encore (Bradshaw *et al.*, 2003), qui considèrent que les rôles des agents varient en fonction des actions qui leur sont obligatoires, permises, possibles, et de l'initiative qu'ils ont pour entreprendre ces actions.

Les principales limitations de ces approches sont le caractère figé du rôle des agents pour chaque niveau ainsi que le nombre restreint de niveaux qui impose un

cadre rigide. En effet, l'utilisation de niveaux prédéfinis et en nombre limité ne peut rendre compte de la variété des situations rencontrées par un robot au cours de sa mission, sans parler des problèmes de basculement d'un niveau à un autre. De plus, il est difficile de trouver un juste milieu entre des définitions universelles et abstraites de niveaux, et des définitions très concrètes mais spécifiques à un système particulier dans le cadre d'une mission donnée.

L'autonomie peut également être considérée au niveau de la *tâche* (ou de l'activité) (Sellner *et al.*, 2006; Finzi *et al.*, 2005), un agent étant autonome pour réaliser une tâche si cette tâche lui est affectée. L'autonomie considérée au niveau de chaque tâche est susceptible d'apporter la solution la plus adaptée à chaque situation de la mission, par contraste avec l'utilisation de niveaux d'autonomie rigides. En revanche, elle nécessite de pouvoir distinguer, pour chaque tâche, qui de l'agent artificiel ou humain est le plus à même de l'effectuer avec les meilleures performances.

2.2. *Le mode d'initiative*

Le *mode d'initiative* concerne la dynamique de l'autonomie : qui de l'agent artificiel ou humain a le pouvoir de modifier le niveau d'autonomie de l'agent artificiel ? Dans la littérature, on distingue essentiellement l'*autonomie adaptative*, l'*autonomie ajustable* et l'*initiative mixte* (Hardin *et al.*, 2009).

L'*autonomie adaptative* confère au seul agent artificiel le pouvoir de modifier son autonomie ; par exemple (Scerri *et al.*, 2003) utilisent des techniques d'optimisation pour que l'agent artificiel gère lui-même son autonomie en conservant ou transférant à un autre agent la prise de décision. Afin d'éviter les comportements dangereux, (Chopinaud *et al.*, 2006) génèrent des agents capables de se surveiller eux-mêmes et de s'autoréguler, en adoptant dynamiquement des comportements de régulation prédéfinis.

Un des avantages de l'autonomie adaptative est de pouvoir systématiser et fiabiliser certains comportements, notamment avec des tâches précises et répétitives. De plus, elle permet d'obtenir des réactions plus rapides que celles que l'on pourrait attendre d'un humain. En revanche, le système est privé des capacités d'analyse de l'opérateur humain, en particulier il ne peut intervenir si le système n'agit pas correctement : son interaction avec l'agent robotique est restreinte à ce qui est attendu de lui (Hardin *et al.*, 2009).

À l'inverse, l'*autonomie ajustable* confère au seul opérateur l'initiative du changement d'autonomie : l'opérateur « conseille » le robot en mission (Myers *et al.*, 2001), ou choisit son niveau d'interaction (Dorais *et al.*, 1999). Dans ce cadre, le système permet à l'opérateur de faire usage de ses capacités d'analyse et de compréhension globale de la mission et d'anticiper les aléas. L'inconvénient est que si l'opérateur commet une erreur, s'il réagit trop lentement, ou si son expertise est insuffisante, les performances du système global s'en ressentent ; les interactions de l'opérateur sur le système ne sont d'ailleurs pas toujours bénéfiques (Schurr *et al.*, 2006).

Enfin, l'*initiative mixte* donne aussi bien au robot qu'à l'opérateur humain le pouvoir de décider de l'autonomie du robot (Hardin *et al.*, 2009; Sellner *et al.*, 2006). Le principe sous-jacent est de combiner les avantages des deux approches précédentes, tout en en réduisant les inconvénients. Bien qu'idéale sur le papier, cette approche reste à parfaire pour montrer ses bénéfices dans la pratique (Hardin *et al.*, 2009).

2.3. *Les critères de décision de l'évolution de l'autonomie*

Dans le cas où le choix du basculement d'un niveau d'autonomie à un autre est laissé à l'agent robotique (autonomie adaptative ou initiative mixte), plusieurs techniques permettent de définir les conditions du basculement et de rendre *opérationnelle* la variation de l'autonomie : (Scerri *et al.*, 2003) dotent les agents robotiques de capacités d'apprentissage fondées sur des processus décisionnels de Markov (MDP), qui leur permettent de transférer la prise de décision si nécessaire à l'agent humain. (Schurr *et al.*, 2009) utilisent une approche similaire pour transférer la prise de décision, mais en tenant compte des possibilités d'incohérences entre agents ; cependant, ces approches nécessitent de pouvoir estimer l'utilité de chaque stratégie possible, ce qui est de fait une approche subjective. Les critères de (Hardin *et al.*, 2009) sont explicites et constitués par des événements présélectionnés (certaines actions de l'opérateur humain, certains événements de mission) ; cependant, ces critères, bien qu'objectifs, sont spécifiques d'une mission donnée et d'un ensemble restreint de tâches. Dans les travaux de (Myers *et al.*, 2001), le comportement de l'agent robotique change en fonction de la vérification ou non d'un ensemble de règles définies par l'opérateur, qui constituent les permissions et comportements que l'opérateur humain veut faire respecter par le robot. Ce système pose toutefois le problème des conflits potentiels entre règles. Citons également les travaux de (Sellner *et al.*, 2006), où la décision de l'affectation à l'agent robotique ou à l'agent humain d'une nouvelle tâche à exécuter dépend de statistiques préétablies quant au succès de la tâche.
Une grande variété d'approches et d'outils est utilisée, qui sont souvent fondés sur des critères et métriques spécifiques du système considéré et donc difficilement applicables à d'autres systèmes.

2.4. *Quelle perception de l'agent humain l'agent artificiel a-t-il ?*

L'opérateur humain a en général une connaissance des capacités de l'agent artificiel qu'il utilise, ainsi qu'une connaissance de son état courant, de ses possibles états futurs (conscience de situation (Endsley, 1995)). Dans le cadre du contrôle de l'autonomie de l'agent artificiel à l'initiative de l'un ou de l'autre agent, l'agent artificiel doit lui aussi avoir un modèle des capacités de l'agent humain, ainsi que de son « état ».

En ce qui concerne les capacités, l'opérateur humain peut intervenir à différents niveaux d'ordres, depuis des ordres « bas niveau » (commande directe des actuateurs)

jusqu'à des ordres « haut niveau » (buts) (Kortenkamp *et al.*, 1997). Bien qu'une approche fréquemment utilisée consiste à automatiser les tâches dites de « bas niveau » et à confier à l'opérateur les tâches de « haut niveau » (Hexmoor *et al.*, 2009; Goodrich *et al.*, 2007), il peut être important pour l'opérateur de pouvoir intervenir à tout niveau, du but jusqu'à la commande : par exemple, en cas de défaillance dans le système, l'opérateur a une meilleure conscience de situation s'il intervient à un niveau d'automatisation faible, c'est-à-dire à un niveau « bas » (Kaber *et al.*, 2000).

Dans le cadre du contrôle collaboratif de (Fong *et al.*, 2002), le robot contacte l'opérateur à chaque fois qu'il rencontre une difficulté : le robot sait ce qu'il peut attendre de l'opérateur ; cependant, les interactions spécifiées ne se font *a priori* que sur demande de l'agent robotique. Dans les travaux de (Kortenkamp *et al.*, 1997), le robot dispose de modèles de tâches réalisables par l'opérateur, ce qui lui permet de planifier de manière globale (c'est-à-dire pour lui et pour l'opérateur, en requérant *explicitement* l'opérateur pour réaliser certaines tâches), et de suivre l'exécution des tâches réalisées par l'opérateur. Dans les travaux de (Finzi *et al.*, 2005), le modèle de tâches (incluant celles de l'opérateur) qu'a le robot lui permet de diagnostiquer un problème à l'exécution et de lancer une procédure de réparation : en ce sens, l'opérateur agit sous vérification du robot.

Enfin, l'intégration de données relatives à l'« état » de l'opérateur (physiologie, expertise, charge de travail) permet à l'agent artificiel d'adapter son autonomie s'il estime que cet état est dégradé (Barber *et al.*, 2008; Coppin *et al.*, 2009).

2.5. *De l'autonomie à l'autorité*

La revue de la littérature relative au contrôle de l'autonomie nous montre plusieurs façons d'envisager ce contrôle, selon la granularité, le mode d'initiative, les critères d'évolution, la perception qu'a l'agent artificiel de l'agent humain. Cependant, à notre connaissance, les changements de contrôle ne sont pas étudiés sous l'angle des conflits qu'ils peuvent provoquer entre les agents. Comme le soulignent l'expérience décrite en introduction ainsi que l'analyse de rapports d'accidents aéronautiques (Dehais *et al.*, 2005), les changements d'autorité imprévus ou incompris peuvent en effet mener à des situations inefficaces, dangereuses, voire catastrophiques. De façon à pouvoir considérer de la même manière l'agent humain et l'agent artificiel, nous préférons manipuler les notions d'autorité et de contrôle d'autorité plutôt que d'autonomie, qui concerne exclusivement l'agent artificiel. Le contrôleur de la dynamique de l'autorité que nous proposons a donc pour objectif la détection et la résolution de conflits d'autorité, en prenant en compte les différentes facettes du contrôle :

– une mise en œuvre des différents modes d'initative (initiative donnée à l'agent artificiel, à l'agent humain ou aux deux, en fonction du contexte) ;

– des objets de l'autorité de granularité pertinente pour la détection et la résolution de conflits (objets sur lesquels les agents peuvent effectivement agir) ;

– un critère d'évolution de l'autorité objectif et indépendant de l'application ;

– l'intégration de modèles de tâches et de l'« état » de l'opérateur dans les connaissances de l'agent artificiel et la prise en compte de l'intervention de l'opérateur aux niveaux d'ordres pertinents, qu'ils soient de « haut » ou de « bas » niveau.

3. Ressources et autorité

3.1. *Ressources*

Un conflit d'autorité se manifeste par une incohérence dans le plan d'un agent (par exemple, une contrainte n'est pas respectée) ou une interférence entre les plans des agents (par exemple, utilisation d'une même ressource non partageable au même moment). Afin d'éventuellement revoir la distribution de l'autorité des agents dans le système, le conflit doit être évalué : identifier quels agents sont concernés, quelles sont les contraintes violées, identifier ce que le conflit remet en question au niveau du plan (connu) des agents. Le contrôleur de la dynamique de l'autorité sera donc fondé sur une représentation des plans courants des agents, c'est-à-dire des tâches qu'ils réalisent, des moyens qu'ils utilisent, des contraintes qu'ils doivent respecter, pour satisfaire leurs buts. Nous représentons le plan d'un agent sous la forme d'un graphe de dépendances.

Nous supposons que l'agent artificiel, dont nous réalisons le contrôleur de la dynamique de l'autorité, dispose de capacités de planification embarquées. Un graphe de dépendances s'obtient par abstraction d'un plan calculé par le planificateur. Par exemple, considérons un planificateur hiérarchique de type HTN - Hierarchical Task Network - tel que JSHOP2 (Nau *et al.*, 2001). L'opérateur de planification *Navigation-Robot*, qui décrit la tâche de navigation automatique pour atteindre un point d'arrivée donné, est le suivant :

```
   1/       ( :operator ( !NavigationRobot ?z ?a)
2/           ((is_zone ?z)(is_robot ?a)(infoPosition_available ?a)(enough_energy ?z ?a)
/                  (SteeringWheel_available ?a)(SafetyDistance_respected ?a) ))
3/           ((forall ( ?zone) ((is_zone ?zone)) ((position ?a ?zone))))
4/           ((position ?a ?z))
5/       )
```

L'opérateur *NavigationRobot* prend en paramètre la zone de destination z et l'agent devant effectuer la tâche a (ligne 1). En ligne 2 sont donnés les prédicats/préconditions devant être validés avant l'exécution de la tâche, à vérifier en fonction des paramètres fournis (z et a). Ensuite viennent les effets de l'opérateur : les effets négatifs en ligne 3, c'est-à-dire ce qui est « retiré » de l'état du système et les effets positifs en ligne 4 (ce qui est « ajouté »).

L'opérateur n'est pas instancié. Lorsque le planificateur a calculé un plan, celui-ci se présente sous la forme d'une liste ordonnée de tâches instanciées à effectuer du type $\{\tau_1, \tau_2, ..., \tau_n\}$. Dans cette liste, les liens entre tâches n'apparaissent pas. Pour obtenir un graphe de dépendances à partir du plan, il faut associer le plan aux opérateurs de

planification, en substituant les variables par leurs valeurs instanciées données dans le plan. Si une précondition pc d'une tâche τ_j correspond à un des effets d'une tâche précédente τ_i, alors τ_j est dépendante de τ_i *via* pc. Cette relation construit le graphe de dépendances.

Définition : Le graphe de dépendances généré à partir d'un plan est un graphe tel que les nœuds sont des *ressources* et les arcs représentent la relation de précondition « a besoin de » entre les ressources.

Une ressource représente donc un élément physique du système (capteur, énergie, etc.) ou un élément symbolique (information, tâche, but, contrainte, entrée opérateur, etc.) Un nœud puits du graphe est un but. Un tel graphe de dépendances sera appelé *graphe de ressources*. Il s'agit d'un graphe dynamique qui sera mis à jour en fonction de la réactualisation du plan au cours de la mission. Seuls les buts poursuivis en cours de mission ainsi que les dépendances associées sont présents dans le graphe de ressources : cela signifie que seul le plan nominal, qui est créé sur un horizon fini, est représenté sous forme de graphe ; cela limite par construction le nombre de noeuds à inclure dans le graphe, qui augmente linéairement avec le nombre de tâches à inclure.

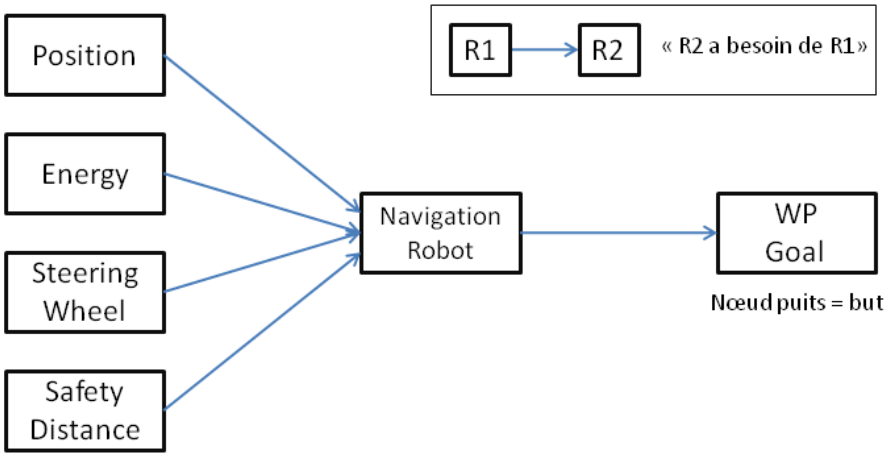

Figure 1. *Graphe de ressources pour le plan de but* WP-Goal

Exemple

Afin de produire le but « Atteindre point d'arrivée WP-Goal », l'opérateur de planification *NavigationRobot* est instancié avec comme paramètres la destination *WP-Goal* et comme agent exécutant le robot. On obtient ainsi le graphe de dépendances présenté figure 1, avec *WP-Goal* comme but, la tâche instanciée *Navigation Robot* et les préconditions sur l'énergie, l'information de position, la commande de cap et le respect de la distance de sécurité.

On remarque que si l'une des ressources du graphe disparaît en cours d'exécution (l'objet symbolisé par la ressource subit un aléa ou est détruit par l'action d'un agent), les ressources qui en dépendent vont elles-mêmes disparaître.

3.2. *Autorité*

Définition : l'autorité d'un agent X sur une ressource R par rapport à un autre agent Y est caractérisée par les trois propriétés suivantes :

– le droit d'*accès* : la capacité de l'agent X à inclure la ressource R dans un graphe de ressources, c'est-à-dire à contrôler R en vue de l'accomplissement d'un but ;

– le droit de *préemption* : si R est une ressource non partageable, bien que X ait le droit d'accès à R, elle peut être contrôlée par Y. Le droit de préemption confère à X le droit de contrôler R dès qu'il le souhaite, aux dépens de Y le cas échéant ;

– la *garantie de contrôle* : une fois que X contrôle R (c'est-à-dire y a effectivement accédé), X peut être menacé de perdre ce contrôle au profit de Y par préemption. La garantie de contrôle assure à X que Y ne pourra pas lui prendre R (Y n'a pas le droit de préemption sur R vis-à-vis de X).

Conséquences :

– l'autorité est *graduelle* : le contrôle de l'agent X sur la ressource R est de plus en plus fort au fur et à mesure qu'il possède, dans cet ordre, les droits : accès, préemption, garantie de contrôle.

– l'autorité, comme l'autonomie (Castelfranchi *et al.*, 2003) est une notion *relative* entre deux agents : par exemple, pour une ressource R donnée, un agent X peut avoir le droit de préemption sur un agent Y, mais pas sur un agent Z. Ainsi il y a autant de relations d'autorité qu'il y a de couples d'agents susceptibles d'utiliser R.

– l'autorité est *partagée* entre les agents : pour un couple d'agents $<X, Y>$ pouvant contrôler une ressource R, le gain d'autorité de X sur R se fait au détriment de Y. Par exemple, si X obtient la garantie de contrôle sur R, Y n'a pas la préemption. À l'extrême, X peut avoir l'exclusivité de la ressource au détriment de Y si celui-ci n'y a pas accès du tout : X *interdit* l'accès de Y à R, même si X ne contrôle pas R en permanence.

4. Modèle formel

Le modèle que nous proposons a pour objectif de définir formellement les liens entre ressources dans le graphe de ressources, ainsi que l'autorité d'un agent X sur une ressource par rapport à un agent Y. Nous définissons successivement la ressource, le lien entre ressources (interface) et l'autorité.

4.1. *Ressources*

Une ressource est modélisée par un ensemble fini de propriétés et d'états, les changements d'états étant déclenchés par des événements. Cette description permet l'utilisation d'un formalisme de type réseaux de Petri, dont une des extensions classiques convient particulièrement, les réseaux de Petri synchronisés (permettant de synchroniser des réseaux de Petri entre eux et avec des programmes ou des actions extérieurs *via* des événements, cf Annexe).

Quel que soit l'élément physique ou symbolique représenté par la ressource R, R est modélisée par le réseau de Petri de la figure 2. Les propriétés particulières de R, ainsi que ses états, seront représentés par le marquage de ce réseau de Petri.

4.1.1. *Notations*

Une transition étiquetée « transition1 » est notée $\mathrm{t}_{\mathrm{transition1}}$; une place étiquetée « place1 » est notée $\mathrm{p}_{\mathrm{place1}}$. Le marquage $\mathrm{M_R} = \{Present, Allocated, \#Users^3\}$ d'un réseau de Petri R correspond au fait que les places $\mathrm{p_{Present}}$ et $\mathrm{p_{Allocated}}$ contiennent un jeton, la place $\mathrm{p_{\#Users}}$ trois jetons et que toutes les autres places de R sont vides. La relation $\{Present\} \subseteq \mathrm{M_R}$ signifie que la place $\mathrm{p_{Present}}$ du réseau R est marquée avec *au moins* un jeton. Elle ne précise rien sur l'état des autres places du réseau. La relation $\{Present\} \nsubseteq \mathrm{M_R}$ signifie que la place $\mathrm{p_{Present}}$ n'est pas marquée (aucun jeton dans la place).

La figure 2 est le réseau de Petri générique représentant une ressource R. Ce réseau comprend deux sous-ensembles de places, permettant de représenter les propriétés et les états de R.

4.1.2. *Les propriétés de ressource*

Les propriétés intrinsèques, invariables de la ressource, conditionnent son comportement :

– R est *partageable* si et seulement si $\{Shareable\} \subseteq \mathrm{M_R}$; plusieurs autres ressources peuvent dépendre simultanément de R.

– R est *non partageable* si et seulement si $\{NonShareable\} \subseteq \mathrm{M_R}$; une seule autre ressource peut dépendre de R.

– R est *préemptable* si et seulement si $\{NonShareable, Preemptable\} \subseteq \mathrm{M_R}$; bien que reliée à une ressource R' dans le graphe de dépendances, ce lien peut être désactivé au profit d'une autre ressource R''. Une ressource ne peut être préemptable que si elle est non partageable.

– R est *non préemptable* si et seulement si $\{Preemptable\} \nsubseteq \mathrm{M_R}$.

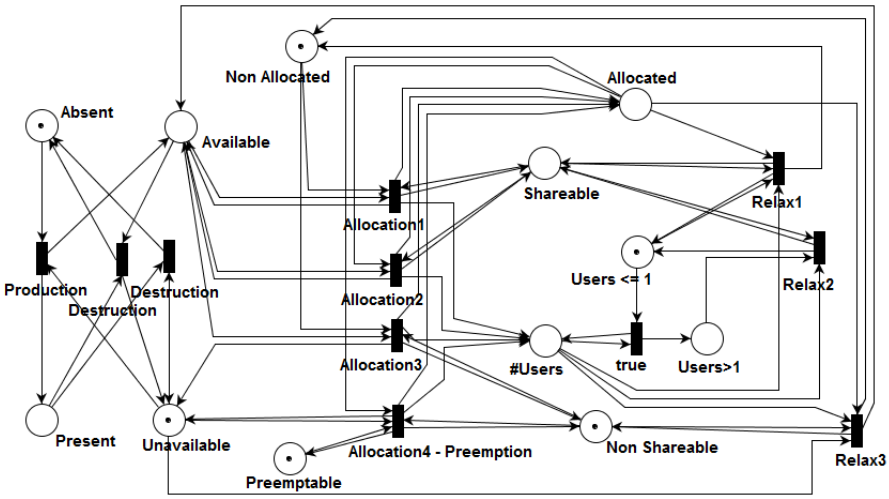

Figure 2. *Réseau générique de ressource (ici non partageable, préemptable)*

4.1.3. *Les états de ressource*

Le graphe de ressources est construit par les relations de dépendances entre celles-ci, ce qui se traduit au niveau de chaque ressource par des états internes d'affectation : une ressource est affectée à une autre dans le cadre de la relation de dépendance qui les unit. Les états de la ressource changent en fonction des événements reçus par la ressource et de ses propriétés.

R peut être dans les états suivants : R existe si et seulement si $\{Present\} \subseteq \mathrm{M_R}$; R n'existe pas si et seulement si $\{Absent\} \subseteq \mathrm{M_R}$; R est disponible pour être affectée à une autre ressource si et seulement si $\{Available\} \subseteq \mathrm{M_R}$; R est indisponible si et seulement si $\{Unavailable\} \subseteq \mathrm{M_R}$, cependant R peut toujours être affectée *via* $\mathrm{t_{Allocation4}}$ par préemption ; R est affectée à une ressource utilisatrice si et seulement si $\{Allocated\} \subseteq \mathrm{M_R}$; R n'est pas affectée si et seulement si $\{NonAllocated\} \subseteq \mathrm{M_R}$. La place $\mathrm{p_{\#Users}}$ contient autant de jetons qu'il y a de ressources utilisant R. L'une ou l'autre des places $\mathrm{p_{Users<=1}}, \mathrm{p_{Users>1}}$ est marquée en fonction du nombre de ressources qui utilisent R simultanément.

Exemple

La ressource *Steering Wheel* est définie comme non partageable et préemptable. Lors de la mission, seul un agent à la fois peut utiliser ce volant. En revanche, à tout moment, le volant peut être préempté par un autre agent venant concurrencer le premier.

4.2. *Arcs du graphe de ressources : interfaces*

4.2.1. *Définition*

On appelle *interface* un arc du graphe de ressources. Une interface représente donc le lien de dépendance qui existe entre deux ressources. Ce lien peut avoir différentes propriétés.

La figure 3 est le réseau de Petri générique représentant une interface $I(R_i, R_j)$ entre deux ressources R_i et R_j, c'est-à-dire la relation de dépendance entre les ressources R_i (ressource affectée) et R_j (ressource utilisatrice).

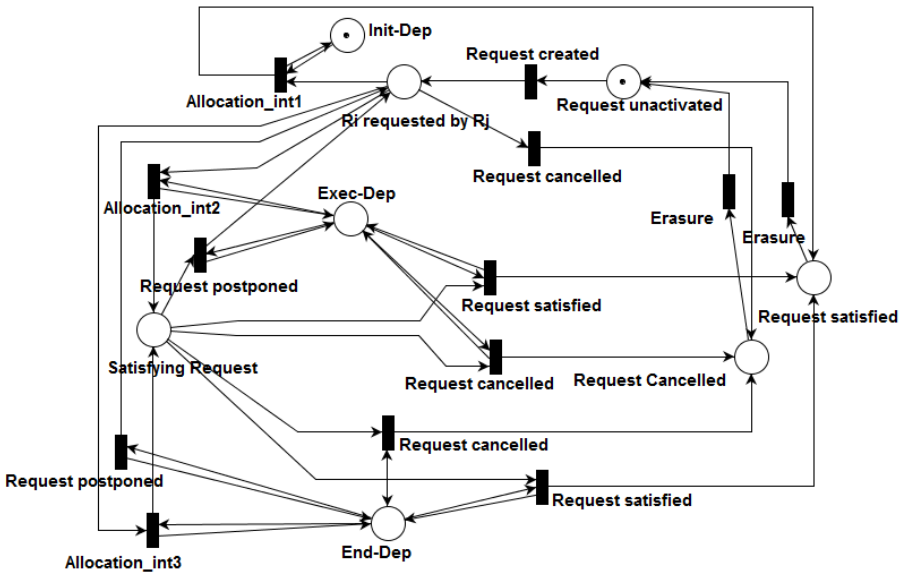

Figure 3. *Réseau générique d'interface (ici interface dépendante en initialisation)*

Une interface entre ressources est toujours créée par un agent, qu'il soit humain ou artificiel : il est le résultat d'une décision, l'agent choisissant à un instant donné de connecter entre elles des ressources afin d'atteindre un but. Une interface est ainsi marquée par le « sceau » de l'agent qui l'a créée. En ce sens, le réseau d'interface utilise un jeton coloré (voir Annexe), la couleur du jeton étant propre à chaque agent (humain ou artificiel).

Tout comme pour le réseau de ressource, on distingue deux catégories de places dans le réseau d'interface, qui représentent :

4.2.2. *Les états de l'interface*

Les places d'état représentent l'état de la requête d'affectation de R_i à R_j, qui change en cours de mission, en fonction des événements reçus par l'interface $I(R_i, R_j)$ et des propriétés de celle-ci :

– la requête est *inactive* si et seulement si $\{RequestUnactivated\} \subseteq \mathrm{M_{I(R_i,R_j)}}$: la requête n'existe pas encore ;

– la requête est *active* si et seulement si $\{RirequestedbyRj\} \subseteq \mathrm{M_{I(R_i,R_j)}}$: la connexion entre R_i et R_j est effective, R_j exprime son besoin envers la ressource R_i ;

– la requête est *en cours de satisfaction* si et seulement si $\{SatisfyingRequest\} \subseteq \mathrm{M_{I(R_i,R_j)}}$: R_i est affectée à R_j, la partie de plan impliquant R_i et R_j est en cours d'exécution ;

– la requête est *satisfaite* si et seulement si $\{RequestSatisfied\} \subseteq \mathrm{M_{I(R_i,R_j)}}$: le besoin de R_j envers R_i a été satisfait. Cet état est conservé jusqu'à réinitialisation du réseau d'interface par tir de la transition $\mathrm{t_{Erasure}}$;

– la requête est *annulée* si et seulement si $\{RequestCancelled\} \subseteq \mathrm{M_{I(R_i,R_j)}}$: le besoin de R_j envers R_i a été annulé par la ressource utilisatrice R_j. Cet état est conservé jusqu'à réinitialisation du réseau d'interface par tir de la transition $\mathrm{t_{Erasure}}$.

4.2.3. *Les propriétés de l'interface*

Les propriétés intrinsèques et invariables de l'interface $I(R_i, R_j)$ sont les suivantes (voir figure 4) :

– R_j dépend de R_i à l'*initialisation* si et seulement si $\{Init{-}Dep\} \subseteq \mathrm{M_{I(R_i,R_j)}}$: R_j a besoin de R_i uniquement pour devenir disponible ; lorsque la transition $\mathrm{t_{Allocation\_int1}}$ sur $I(R_i, R_j)$ est franchie, alors $\mathrm{t_{Production}}$ de R_j est tirée. Il n'y a alors plus de lien entre R_i et R_j : $\{Requestsatisfied\} \subseteq I(R_i, R_j)$, la requête est close. Si R_i disparaît par la suite, cela n'affecte pas R_j ;

– R_j dépend de R_i à l'*exécution* si et seulement si $\{Exec{-}Dep\} \subseteq \mathrm{M_{I(R_i,R_j)}}$: R_j a besoin de R_i pour devenir disponible et pour le rester. Le tir de $\mathrm{t_{Allocation\_int2}}$ sur $I(R_i, R_j)$ entraîne le tir de $\mathrm{t_{Production}}$ de R_j, mais R_i reste affectée à R_j jusqu'à ce que celle-ci la restitue : $\{SatisfyingRequest\} \subseteq I(R_i, R_j)$;

– R_j dépend de R_i à l'*exécution complète* si et seulement si $\{End{-}Dep\} \subseteq \mathrm{M_{I(R_i,R_j)}}$: R_i ne peut être restituée que lorsque R_j est produite. L'affectation se fait *via* $\mathrm{t_{Allocation\_int3}}$ sur $I(R_i, R_j)$. C'est uniquement lorsque $\mathrm{t_{Production}}$ de R_j est tirée que $\{Requestsatisfied\} \subseteq I(R_i, R_j)$. Ceci provoque alors le tir d'une transition $\mathrm{t_{Relax(1,2ou3)}}$ de R_i.

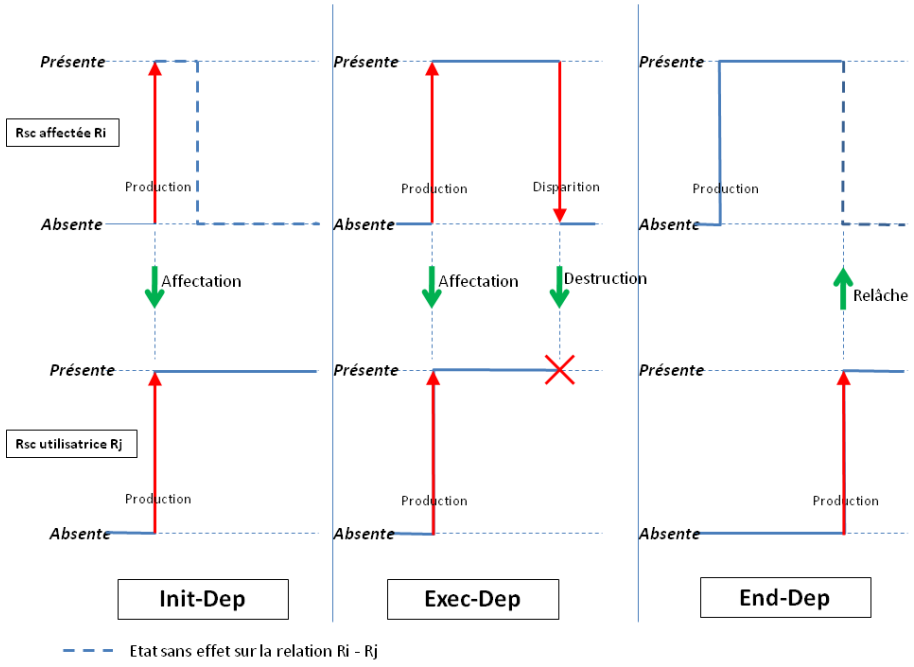

Figure 4. *Dépendances entre R_i et R_j*

4.2.4. *Exemple*

Reprenons l'exemple de l'expérience décrite en introduction : le plan initial consiste pour le robot à aller de manière autonome sur le point d'arrivée *WP-Goal*. La figure 5 présente le graphe de ressources correspondant. Le nœud puits est le but *WP-arrival*, qui dépend de la ressource *Navigation Robot* ; celle-ci requiert les ressources *Position*, *Energy* et *Steering Wheel* (commande de cap), ainsi que la ressource *SafetyDistance*.

Les ressources sont en gris alors que les interfaces sont en blanc. Les arcs représentent ici les liens entre réseaux de Petri : connexion par fusion de transitions ou envoi d'événements, cf Annexe. La fusion de transitions et l'envoi d'événements permettent de coordonner les changements d'états entre les différents réseaux de Petri du modèle. C'est la création d'arcs de dépendance entre ressources qui détermine quelles transitions doivent être fusionnées (de manière préétablie), et quels événements les transitions doivent envoyer ou attendre.

La ressource *Position* est affectée à la ressource *Navigation Robot via* l'interface *Int1*. La relation de dépendance est de type $Exec-Dep$ car *Navigation Ro-*

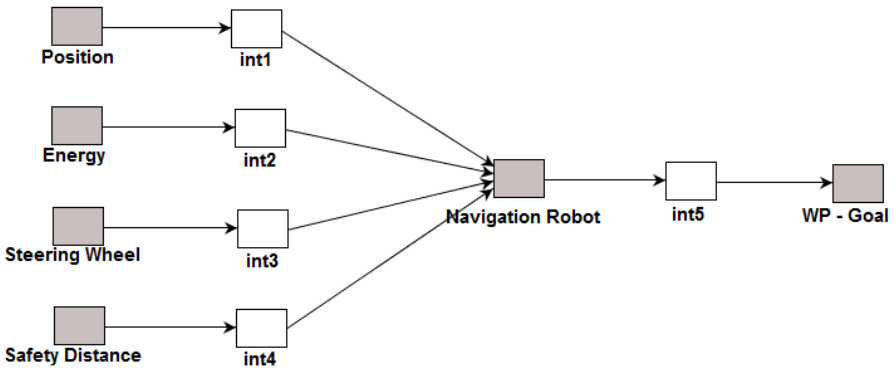

Figure 5. *Graphe de ressources initial*

bot a besoin de *Position* jusqu'à achèvement de la tâche. La connexion de *Position* à *Int1* a un impact au niveaux des réseaux de Petri sous-jacents : chacune des transitions $\mathrm{t_{Allocation(1,2,3ou4)}}$ de *Position* est fusionnée avec chacune des transitions $\mathrm{t_{Allocation\_int(1,2ou3)}}$ de *Int1*, ce qui donne douze combinaisons de paires de transitions fusionnées. Si à un instant donné, une de ces paires de transitions fusionnées est validée par le marquage respectif des réseaux (une transition validée sur *Position* et simultanément une sur *Int1*), alors *Position* sera prête à être affectée. Il en va de même pour les ressources *Energy*, *Steering Wheel* et *Safety-Distance* avec leur interface respective. Pour que l'affectation ait lieu, il faut que toutes les ressources dont dépend la ressource utilisatrice soient prêtes à être affectées ; ainsi en pratique *Position*, *Energy*, *Steering Wheel* et *SafetyDistance* ne sont affectées simultanément à *NavigationRobot* que si toutes individuellement peuvent l'être. Prenons le cas de la ressource *Steering Wheel* : il s'agit d'une ressource *non partageable*, et *préemptable*. Elle est affectée à *Navigation Robot viaint3*, qui représente une dépendance en exécution. Pour que *Steering Wheel* soit affectable, il faut que la transition $t_{Allocation3}$ sur *Steering Wheel* soit validée par son marquage (c'est-à-dire que $\{Available, NonAllocated, NonShareable\} \subseteq \mathrm{M_{SteeringWheel}}$) et que $t_{Allocation\_int2}$ sur *int3* le soit également ($\{RirequestedbyRj, Exec{-}Dep\} \subseteq \mathrm{M_{int3}}$), simultanément.

4.3. *Autorité sur une ressource*

L'autorité se définit comme une relation $\mathrm{A_R(X, Y)}$ entre deux agents X et Y et se caractérise par trois propriétés : le droit d'accès, le droit de préemption et la garantie de contrôle. Pour chaque agent, l'autorité prend une des valeurs suivantes : *No Access*, *Access* ou *Preemptability*. En fonction de la valeur de l'autorité de l'autre agent dans la relation $\mathrm{A_R(X, Y)}$, le tableau 2 présente les propriétés résultantes pour chaque agent vis-à-vis de la ressource R :

Agent	Autorité	Accès	Préemption	Garantie
X	No Access	Non	-	-
Y	Preemptability	Oui	Oui	Oui
X	Access	Oui	Non	Non
Y	Preemptability	Oui	Oui	Oui
X	Access	Oui	Non	Oui
Y	Access	Oui	Non	Oui
X	Preemptability	Oui	Oui	Non
Y	Preemptability	Oui	Oui	Non

Tableau 2. *Propriétés de l'autorité relative de X et Y pour la ressource R*

Chaque double ligne donne, pour chaque agent X et Y, la valeur de son autorité sur la ressource R et le statut des trois propriétés associées. On observe deux états intermédiaires pour lesquels l'autorité des agents est équivalente ; il s'agit des relations (Access / Access) et (Preemptability / Preemptability). Dans la première, les agents ne peuvent se prendre mutuellement le contrôle de la ressource R, chacun doit attendre si nécessaire que l'autre agent ait terminé. Il s'agit d'une configuration de *coopération*. La deuxième relation peut conduire à une oscillation du contrôle : les agents peuvent se prendre mutuellement et indéfiniment le contrôle de la ressource R, ce qui provoque un comportement du système inefficace voire dangereux. Il s'agit d'une configuration de *concurrence*.

4.3.1. *Représentation de la relation d'autorité*

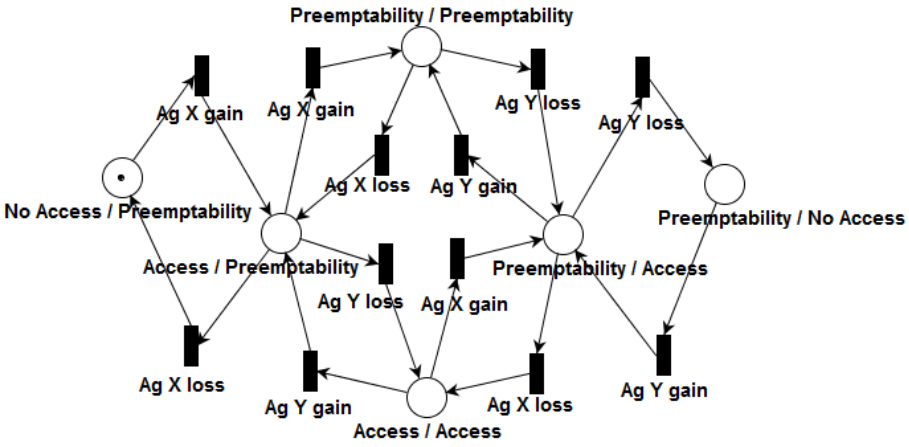

Figure 6. *Relation d'autorité entre les agents X et Y sur la ressource R :* $\mathrm{A_R(X, Y)}$.

Le réseau de Petri de la figure 6 représente la relation d'autorité $\mathrm{A_R(X, Y)}$ entre deux agents X et Y pour une ressource R. Ce réseau est sauf : une et une seule place peut être marquée à la fois. Chacune des places de ce réseau représente une des six combinaisons de valeurs d'autorité pour les deux agents : chaque place du réseau est étiquetée par (Autorité de l'agent X / Autorité de l'agent Y), par exemple : (No Access / Preemptability). Une transition entre deux états indique un gain ou une perte d'autorité pour un seul des agents à la fois. Pour passer d'un extrême à l'autre, un agent doit gagner de l'autorité (de « No Access » à « Preemptability ») et l'autre agent en perdre (de « Preemptability » à « No Access »).

Une interface (non représentée ici) fait le lien entre ce réseau d'autorité et le réseau de la ressource R associée. Par fusion de transition avec les transitions $\mathrm{t_{Allocation(1,2,3ou4)}}$ de la ressource R, c'est bien la relation d'autorité $\mathrm{A_R(X, Y)}$ qui définit si la ressource R peut être affectée à une ressource utilisatrice *via* une interface créée par l'agent X ou l'agent Y. Les couleurs de jeton, propres à chaque agent, permettent de distinguer les droits en fonction des agents.

4.3.2. *Exemple*

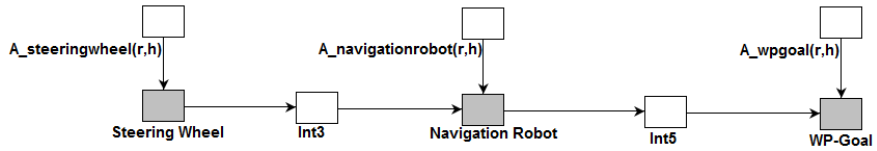

Figure 7. *Exemple de graphe de ressources incluant les relations d'autorité*

La figure 7 représente une partie de l'exemple précédent, partant de la ressource *Steering Wheel*. Les relations d'autorité $\mathrm{A_{Ri}(X, Y)}$ ont ici été ajoutées : il y a une relation d'autorité par couple d'agents (ici le robot et l'opérateur) pour chaque ressource.

5. Conflits sur les ressources

Lors du déroulement de la mission, des différences peuvent apparaître entre l'exécution prévue du plan et son exécution observée. Ces différences proviennent d'aléas liés à l'environnement, de défaillances, ou d'actions imprévues des agents, et vont se manifester sur une ou plusieurs des ressources du graphe de ressources qui vont se retrouver dans un état incohérent. L'incohérence pour une ressource se définit par un marquage atteignable indésiré.

Un marquage indésiré peut apparaître au niveau de toute ressource du modèle. Les ressources étant toutes représentées par le même réseau de Petri, il suffit d'étudier

les marquages atteignables de ce réseau (figure 2) en fonction de son marquage initial, c'est-à-dire en fonction des propriétés intrinsèques de la ressource. Nous avons classé ces marquages atteignables selon qu'ils résultent d'une séquence de tirs de transitions correspondant à l'exécution d'un plan nominal ou au contraire à une séquence correspondant à un aléa, ces derniers étant des marquages indésirés.

Définition : un conflit sur une ressource R est un marquage atteignable de R indésiré. Il existe deux types de conflits : le conflit de destruction $\mathrm{Conflict}_{\mathrm{Destruction}}$ et le conflit de préemption $\mathrm{Conflict}_{\mathrm{Preemption}}$:

– $\mathrm{Conflict}_{\mathrm{Destruction}} = \{\mathrm{Absent}, \mathrm{Allocated}\}$ se produit lorsqu'une transition $\mathrm{t}_{\mathrm{Destruction(1ou2)}}$ de la ressource R est tirée alors que $Allocated \subseteq \mathrm{M_R}$ (voir figure 8). Ce conflit résulte d'une panne, de l'échec d'une tâche, ou d'une violation de contrainte ; R est détruite alors qu'elle était affectée.

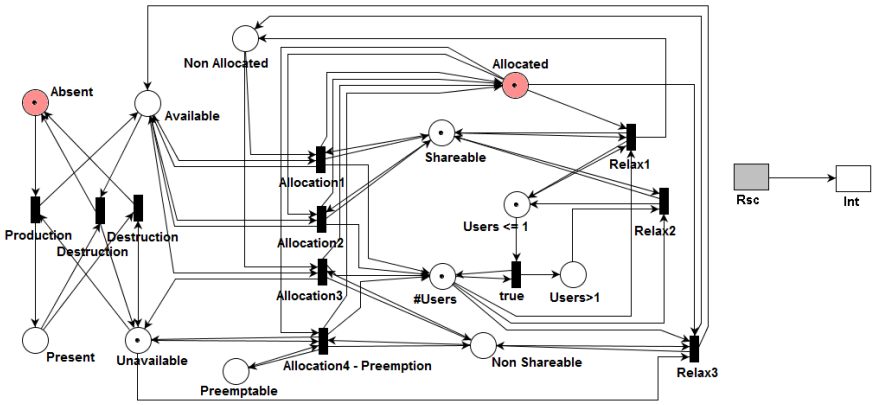

Figure 8. *Conflit de destruction*

– $\mathrm{Conflict}_{\mathrm{Preemption}} = \{\mathrm{NonShareable}, \#\mathrm{Users}^2\}$ se produit lorsque la transition d'affectation $\mathrm{t}_{\mathrm{Allocation4-Preemption}}$ est tirée alors que $\{Allocated, NonShareable\} \subseteq \mathrm{M_R}$. C'est le cas d'un agent qui requiert une ressource non partageable déjà utilisée par un autre agent (voir figure 9), deux réseaux d'interface sont alors connectés à R.

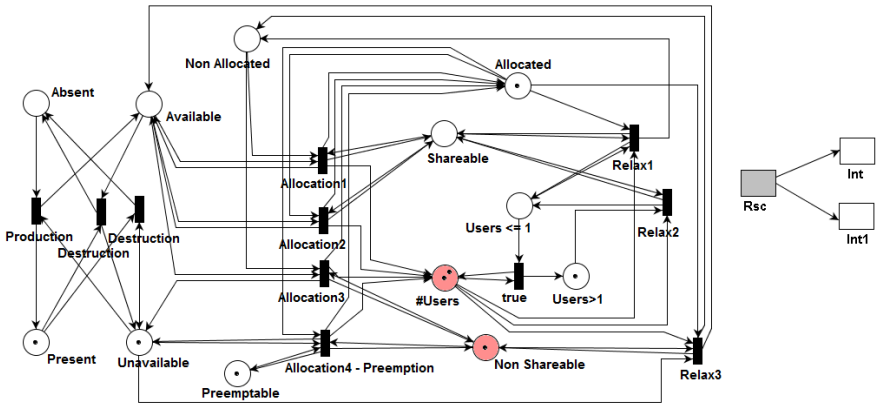

Figure 9. *Conflit de préemption*

Exemple

En reprenant l'exemple précédent, alors que le robot réalise sa navigation autonome vers *WP-arrival*, l'opérateur humain reprend le contrôle de la trajectoire via son joystick.

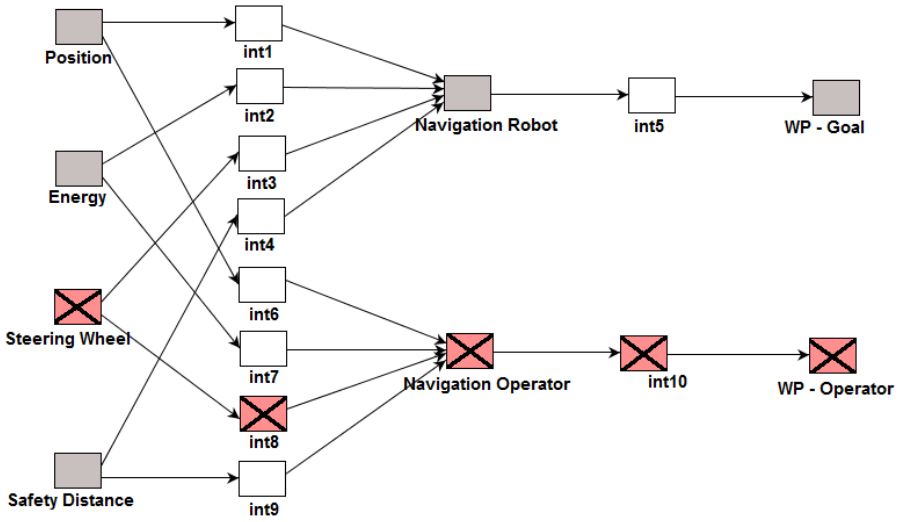

Figure 10. *Graphe de ressources après insertion des actions connues de l'opérateur humain et conflit (les relations d'autorité ne sont pas représentées pour alléger la figure)*

Il peut le faire car $\{\mathrm{Access/Preemptability}\} \subseteq \mathrm{M}_{\mathrm{A}_{\mathrm{SteeringWheel}}(\mathrm{r,h})}$. Cette action de l'agent humain modifie le graphe de ressources (voir figure 10) : la ressource *Steering Wheel*, déjà affectée à *Navigation Robot via int4*, est préemptable et requise par *Navigation Operator viaint8* ; les transitions $\mathrm{t}_{\mathrm{Allocation4}}$ de *Steering Wheel* et $\mathrm{t}_{\mathrm{Allocation\_int2}}$ sur *int8* sont tirées. *Steering Wheel* est alors affectée simultanément à *Navigation Robot* et à *Navigation Operator*, ce qui est incohérent car il s'agit d'une ressource non partageable. On vérifie alors $\mathrm{Conflict}_{\mathrm{Preemption}} \subseteq \mathrm{M}_{\mathrm{SteeringWheel}}$. On notera que le but de l'opérateur *WP-Operator* est inclu dans le graphe de ressources, tout en étant inconnu, suite aux actions de l'opérateur sur son interface utilisateur. Le graphe de ressources de la figure 10 constitue ainsi l'état explicite de connaissance de l'agent robotique consécutif aux actions de l'agent humain.

6. Résolution de conflits d'autorité

6.1. *Résolution sans changement d'autorité*

Nous avons vu à la section 5 qu'il existe deux types de conflits : $\mathrm{Conflict}_{\mathrm{Destruction}}$ et $\mathrm{Conflict}_{\mathrm{Preemption}}$, qui correspondent à des marquages atteignables indésirés au sein d'une ou plusieurs ressources du graphe de ressources.

Le rôle du contrôleur de la dynamique de l'autorité, fonction de l'agent robotique, est de détecter les conflits puis de restaurer la cohérence au sein du graphe de ressources :

– Surveillance : observation du marquage de chacune des ressources du graphe de ressources, pour détecter l'apparition d'un des deux marquages indésirés au sein de l'une d'entre elles.

– Détection du conflit : $\mathrm{Conflict} \subseteq \mathrm{M}_{\mathrm{R}_{\mathrm{Conflict}}}$ pour une ressource $R_{Conflict}$ du graphe de ressources.

– Analyse du conflit : identification des ressources affectées par le conflit, ce sont les ressources en aval de $R_{Conflict}$ dans le graphe. Les relations d'autorité sur ces ressources sont collectées, en vue de l'étude des solutions disponibles.

– Requête à la planification : sur demande du contrôleur de la dynamique de l'autorité, un plan est recherché pour restaurer la cohérence en réaffectant les ressources entre agents, tout en conservant les buts poursuivis jusqu'alors : il s'agit des buts terminaux, c'est-à-dire les ressources nœuds-puits du graphe, et des sous-buts définis par l'opérateur humain que le robot doit respecter, n'ayant pas l'autorité suffisante pour les modifier, les relations d'autorité entre agents restant ici inchangées. L'espace de recherche de plan est contraint par les actions réalisables par les agents ainsi que par les relations d'autorité entre eux sur les ressources. Si un plan est trouvé, un certain nombre d'opérations sont nécessaires pour transformer le graphe de ressources avec conflit en graphe sans conflit : les interfaces ne faisant pas partie du graphe sans conflit sont détruites. À l'inverse, à partir du plan solution généré, de nouvelles interfaces sont créées, connectées éventuellement à de nouvelles ressources. Les opérations

pour passer du graphe avec conflit à un graphe sans conflit sont conditionnées par l'autorité de l'agent artificiel sur chaque ressource : en particulier, lors de la réorganisation du graphe, l'agent artificiel ne peut détruire une ressource utilisée par l'opérateur humain s'il ne dispose pas de la préemptabilité sur celle-ci. Il est à noter que lors de la recherche de plan solution, l'interaction avec l'opérateur humain est une des actions disponibles pour l'agent artificiel afin d'initier la restauration de la cohérence de la ou des ressources en conflit.

Exemple

Sur l'exemple de la figure 10, un conflit de préemption a lieu sur la ressource *Steering Wheel*. Les ressources *Navigation Robot* et *Navigation Operator* sont en concurrence pour le contrôle de la ressource *Steering Wheel* qui est non partageable. Soit le réseau d'autorité $\mathrm{A}_{\mathrm{SteeringWheel}}(\mathrm{r},\mathrm{h})$, qui représente la relation d'autorité entre l'agent robotique r et l'opérateur humain h sur la ressource *SteeringWheel*. Cette relation d'autorité vérifie $\{\mathrm{Access/Preemptability}\} \subseteq \mathrm{M}_{\mathrm{A}_{\mathrm{SteeringWheel}}(\mathrm{r},\mathrm{h})}$, donc l'interface créée par l'opérateur a priorité sur celle de l'agent robotique. Cependant, il doit y avoir replanification, le plan de l'agent robotique étant désormais irréalisable : on cherche un plan satisfaisant les buts *WP-Operator* et *WP-Goal* des agents, et dont la représentation sous forme de graphe de ressources contient la ressource *Navigation Operator*. Un plan solution possible est de laisser la main à l'opérateur et de relancer la navigation du robot pour atteindre *WP-Goal* dès qu'il relâche le joystick de commande (libérant la ressource *SteeringWheel*).

6.2. *Dynamique du partage de l'autorité*

S'il n'y a pas de plan solution sans changement d'autorité, la planification requiert la modification des relations d'autorité afin d'agrandir son espace de recherche : les transitions *Ag r gain*, *Ag r loss*, *Ag h gain* et *Ag h loss* sur les réseaux $\mathrm{A}_{\mathrm{R}}(\mathrm{r},\mathrm{h})$ peuvent être tirées, R étant une des ressources en conflit ou toute autre ressource disponible permettant de créer un plan solution et sur laquelle l'agent robotique n'a initialement pas l'autorité suffisante pour la contrôler.

Si même avec cet espace de recherche élargi aucun plan n'est trouvé, la planification doit calculer un plan dégradé qui induit la perte d'un ou plusieurs buts initialement poursuivis.

Exemple

La main étant laissée à l'opérateur pour la tâche de navigation manuelle (ressource *Navigation Operator*), l'opérateur viole la contrainte de distance de sécurité, détruisant la ressource *SafetyDistance*. On a alors $\mathrm{Conflict}_{\mathrm{Destruction}} \subseteq \mathrm{M}_{\mathrm{SafetyDistance}}$. Un plan solution à ce conflit consiste à redonner la navigation en urgence au robot : les transitions *Ag r gain* puis *Ag h loss* sont tirées successivement sur

$\mathrm{A}_{\mathrm{SteeringWheel}}(\mathrm{r},\mathrm{h})$ afin d'obtenir $\{\mathrm{Preemptability}/\mathrm{Access}\} \subseteq \mathrm{M}_{\mathrm{A}_{\mathrm{SteeringWheel}}(\mathrm{r},\mathrm{h})}$: l'agent robotique a gagné de l'autorité sur l'opérateur humain pour la ressource *SteeringWheel*. Cela rend possible l'affectation de la ressource *Steering Wheel* à la ressource *EmergencyNavRobot* pour produire la ressource *SafetyDistance* devenue but. Le graphe de ressources correspondant à ce plan solution est présenté figure 11. On note que la ressource *EmergencyNavRobot* ne requiert pas *SafetyDistance* puisqu'elle est destinée à la produire. Une fois la distance de sécurité rétablie, l'agent robotique relâche la ressource *Steering Wheel*, la main est rendue à l'opérateur (ressource *Navigation Operator*) qui dispose toujours de l'accès à cette ressource, dans le cadre de la relation d'autorité modifiée précédemment (elle reste inchangée, $\{\mathrm{Preemptability}/\mathrm{Access}\} \subseteq \mathrm{M}_{\mathrm{A}_{\mathrm{SteeringWheel}}(\mathrm{r},\mathrm{h})}$).

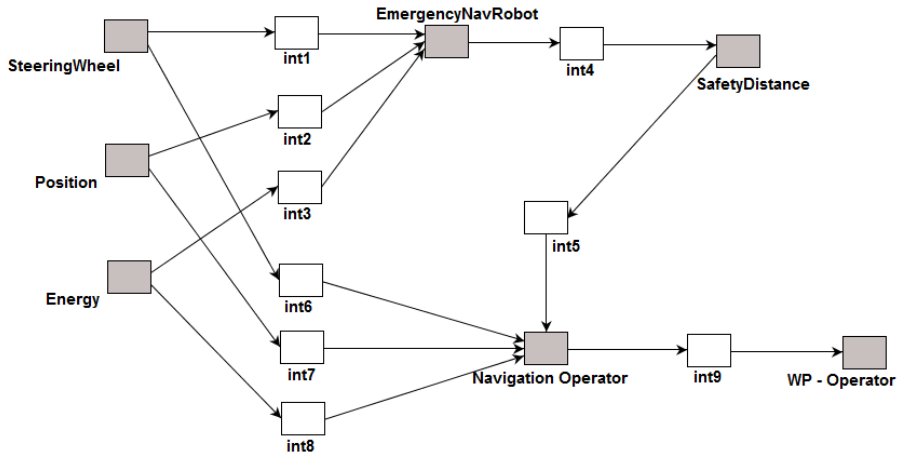

Figure 11. *Graphe de ressources après prise d'autorité par l'agent robotique (les relations d'autorité ne sont pas représentées pour alléger la figure)*

7. Mise en œuvre

7.1. *Architecture*

L'architecture embarquée de l'agent robotique, presentée sur la figure 12, est composée de :

– un *Superviseur* : c'est le cœur exécutif de l'agent robotique. À partir du plan généré par le *Planificateur* et des événements en provenance du *Contrôleur de la dynamique de l'Autorité*, le *Superviseur* lance ou stoppe les algorithmes, envoie les ordres aux actuateurs ;

– un *Planificateur* : le planificateur génère le plan du robot (en intégrant des tâches à réaliser par l'opérateur) afin d'atteindre les buts de mission tout en respectant les

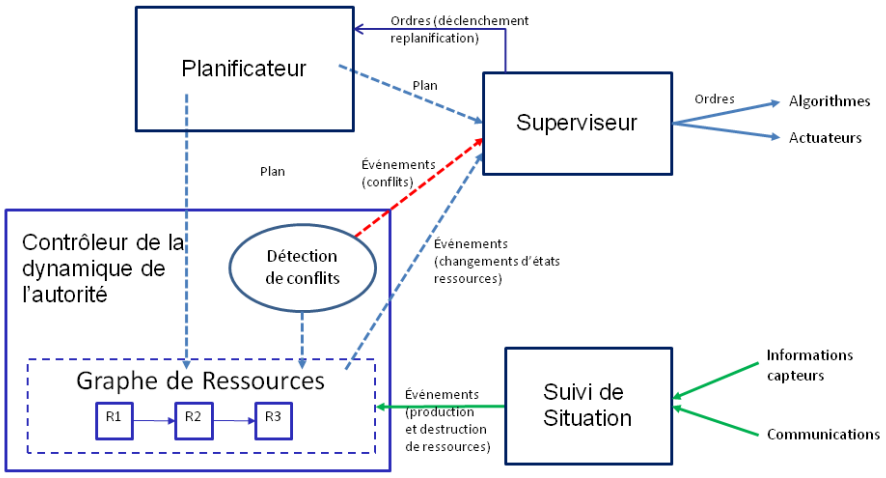

Figure 12. *Architecture décisionnelle de l'agent robotique*

contraintes. Une abstraction du plan en est extraite sous forme d'un graphe de ressources et est transmis au *Contrôleur de la dynamique de l'Autorité* ;

– le *Suivi de Situation* : il réalise en permanence une estimation de l'état courant du système dans son ensemble (y compris l'opérateur humain[3]) et prédit ses états futurs, sur la base des informations en provenance des capteurs et des communications. Les événements (changements d'état, franchissements de seuils, etc.) détectés par le suivi de situation viennent tirer des transitions dans les réseaux de ressources du graphe de ressources, pour marquer les ressources comme produites (*Present*) ou détruites (*Absent*) ;

– le *Contrôleur de la dynamique de l'autorité* : il contrôle la cohérence du graphe de ressources, détecte les conflits consécutifs aux modifications du marquage des ressources effectuées par le *Suivi de Situation*, restaure la cohérence avec le *Planificateur*.

7.2. *Outil de modélisation*

Chaque ressource du système, tout comme chaque interface entre ressources ou chaque relation d'autorité, est modélisée par un réseau de Petri générique. Afin de représenter les plans de mission et leurs évolutions, il faut pouvoir créer des graphes de ressources en s'affranchissant de la création et la connexion des réseaux de Petri sous-jacents. De plus, ces concepts devant être mis en œuvre sur une plate-forme expérimentale, il est nécessaire de disposer d'un outil à même d'être connecté à cette

3. Par exemple, *via* un oculomètre.

plate-forme et de jouer ces réseaux de Petri. Il n'existe pas de logiciel correspondant à ces besoins particuliers, aussi avons-nous développé notre propre outil, MNMS[4], basé sur un éditeur de réseaux de Petri diffusé en OpenSource, Pipe[5] - (Bonet *et al.*, 2006).

Notre logiciel permet en particulier :

– le jeu des réseaux de Petri du graphe de ressources, synchronisés par fusion de transitions et envoi d'événements ;

– l'insertion en ligne (pendant le jeu des réseaux de Petri) de nouveaux réseaux et la suppression de réseaux, afin de mettre à jour le graphe de ressources ;

– la connexion par socket avec tout autre programme externe afin d'inclure le graphe de ressources dans une plate-forme expérimentale, incluant notamment les fonctions de suivi de situation, planification, interface utilisateur, etc.

– en conception, la manipulation directe des ressources et interfaces (création, gestion des propriétés, connexions entre réseaux) sans devoir manipuler les réseaux de Petri sous-jacents ;

– en conception, l'édition directe manuelle des réseaux de Petri pour créer des réseaux génériques.

7.3. *Expérimentations*

L'expérience décrite en introduction est reprise (même plate-forme expérimentale, même scénario), avec la mise en place du contrôleur de la dynamique du partage de l'autorité. Rappelons que la première expérience nous avait permis d'établir la validité du concept de conflit comme déclencheur d'une redistribution de l'autorité entre l'agent humain et l'agent artificiel.

Cette deuxième expérience a été réalisée avec 12 sujets. Il s'agit de tester empiriquement un des moyens d'action du contrôleur de la dynamique de l'autorité pour restaurer la cohérence suite à détection de conflit, à savoir une interaction planifiée avec l'opérateur. Sur les bases d'un modèle simplifié de la conscience de situation de l'opérateur (son comportement oculaire), l'agent artificiel va résoudre le conflit d'autorité (l'opérateur persévère dans l'examen des cibles tandis que le robot tente d'exécuter son plan de retour à la base) en modifiant à la fois son propre comportement et celui de l'opérateur : l'agent artificiel utilise une procédure d'interaction visant à modifier l'information présentée à l'opérateur. La figure 13 présente le graphe de ressources visant à la fois à faire rentrer le robot à la base tout en permettant à l'opérateur de sortir de sa persévération et de comprendre le comportement du robot.

L'agent artificiel poursuit simultanément deux buts distincts *WP-Base* et *SA_Operator*, ce qui explique que le graphe de ressources n'est pas connexe. Le sous-graphe de ressources pour le but *WP-Base* suppose que *SteeringWheel* soit affectée

4. Multiple (Petri) Nets Management System.
5. Platform Independent Petri net Editor.

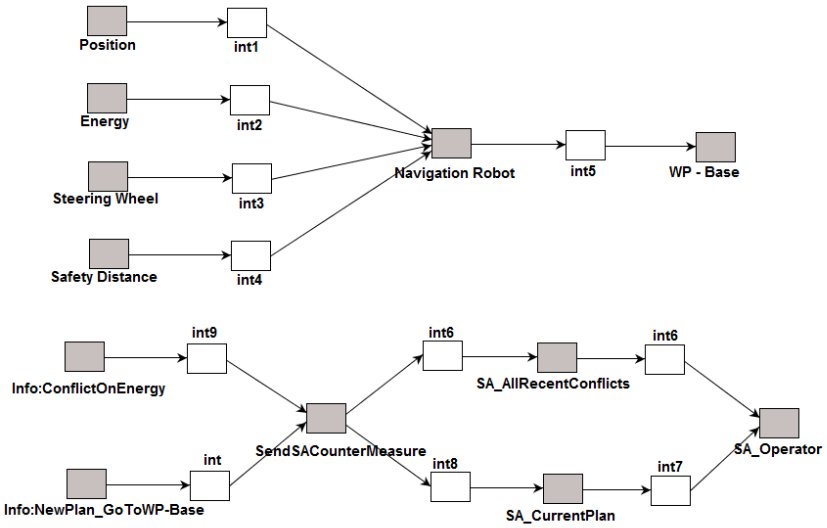

Figure 13. *Graphe de ressources correspondant à la résolution du conflit*

à la tâche *Navigation Robot*, ce qui implique que le robot a préempté *SteeringWheel* à l'opérateur (il faut que $\{\mathrm{Preemptability}/\mathrm{Access}\} \subseteq \mathrm{M}_{\mathrm{A}_{\mathrm{SteeringWheel}}(\mathrm{r,h})}$). Le deuxième but, *SA_Operator*, signifie que l'agent artificiel cherche à sortir l'opérateur de sa persévération en l'informant du conflit (conflit de destruction sur *Energy*) et du plan créé depuis pour résoudre ce conflit, afin de lui restaurer sa conscience de situation. Ce graphe de ressources contient ainsi une tâche d'interaction *SendSACounterMeasure*, il s'agit d'une contre-mesure cognitive (Dehais *et al.*, 2003) portant sur les informations de conflit *Info :ConflictOnEnergy* et de plan *Info :NewPlanGoToWP-Base*, les données oculométriques ayant permis d'estimer qu'elles n'ont pas été perçues (voir figure 14). La contre-mesure cognitive consiste à retirer sur l'interface de l'opérateur humain la vision panoramique sur laquelle il est focalisé (pour localiser la cible) et à la remplacer pendant 4 secondes par un message « Panne batterie, le robot rentre à la base ».

Le tableau 3 résume les résultats de cette seconde expérience. La première colonne indique l'identifiant du participant, la deuxième la durée du conflit après l'envoi de la contre-mesure et la troisième détaille si les trois alertes nécessaires à la compréhension du conflit ont été regardées au moins une fois après l'envoi de la contre-mesure. Par exemple, le participant Pouch a immédiatement perçu et compris le conflit après l'envoi de la contre-mesure cognitive et celle-ci l'a conduit à porter au moins une fois son regard sur le mode « supervisé », sur le synoptique indiquant

Figure 14. *Interface de contrôle du robot : l'opérateur humain est équipé d'un oculomètre qui permet de déterminer les directions du regard à partir de la position de la croix rouge. Dans cet exemple, l'opérateur humain est focalisé sur le mode de pilotage « manuel ».*

Sujet	Durée conflit (en s.)	Alertes regardées
Pouch	0	supervisé, retour base, batterie
Gargu	26	supervisé, retour base, batterie
Lesc	0	supervisé, retour base, batterie
Berdo	0	supervisé, retour base, batterie
Thola	0	supervisé, retour base, batterie
Lasni	0	supervisé, retour base, batterie
Gabje	0	supervisé, retour base, batterie
Jacra	0	supervisé, retour base, batterie
Magma	0	supervisé, retour base, batterie
Rogma	50	supervisé, retour base, batterie
Treau	0	supervisé, retour base, batterie
Herma	0	supervisé, retour base, batterie

Tableau 3. *Résultats de l'expérience avec contre-mesure décidée par le contrôleur dynamique de l'autorité*

le retour base et sur l'état de la batterie. Le participant Gargu a perçu l'ensemble des informations pertinentes mais il a mis 26 secondes avant de rendre la main complètement au robot.

L'analyse des résultats montrent que ce principe d'interaction avec l'opérateur initiée par le contrôleur de la dynamique de l'autorité est efficace : 10 participants sur 12 ont immédiatement rendu la main au robot pour le laisser rentrer à la base. Les deux autres participants, qui n'ont pas rendu immédiatement la main, ont déclaré avoir perçu le conflit à l'aide de la contre-mesure mais ils ont décidé de rester proches des cibles car ils estimaient disposer d'une marge de manœuvre suffisante avant que la batterie ne se décharge totalement.

8. Conclusion

Dans le cadre de missions réalisées conjointement par un agent artificiel et un agent humain, nous avons présenté dans cet article un contrôleur de la dynamique de l'autorité, fondé sur un graphe de dépendances entre ressources contrôlables par les deux agents, dont l'objectif est d'adapter le comportement de l'agent artificiel ou de l'agent humain en cas de conflit d'autorité sur ces ressources.

Nous avons défini l'autorité relative de deux agents par rapport au contrôle d'une ressource à travers les propriétés d'accès, de préemption, de garantie de contrôle. L'autorité d'un agent sur une ressource peut ainsi évoluer au cours de la mission, afin d'ajuster son comportement à celui de l'autre agent.

De futures expérimentations devraient permettre d'étudier en particulier les critères justifiant le gain ou la perte d'autorité pour un agent sur une ou plusieurs ressources, ce que nous nommons *méta-autorité*. L'aspect expérimental sera primordial car permettre à un agent artificiel de reprendre de l'autorité sur les ressources manipulées par un opérateur humain soulève de nombreux problèmes : cohérence de l'agent artificiel, maintien de la conscience de situation de l'opérateur, communication entre agents, acceptation par l'opérateur. En ce sens, des solutions permettant d'influencer les actions de l'opérateur sans le perturber (guidage « subliminal », actions sur sa conscience de situation par l'envoi de contre-mesures, etc.) doivent être développées.

Les premières expérimentations montrent que cette approche a le potentiel de résoudre des conflits d'autorité. D'autres expérimentations doivent être conduites afin d'observer si l'agent artificiel est capable d'adapter son propre plan en cas de prise d'autorité imprévue par l'opérateur, ou s'il est capable de prendre de l'autorité lorsque l'opérateur n'agit pas ou disparaît (conduisant à une absence momentanée d'agent contrôlant le système).

L'utilisation d'un graphe de ressources et la localisation du conflit au sein de ce graphe fournit des informations pertinentes pour l'interaction, ce qui implique également un travail à venir sur les interfaces opérateur. Enfin, des travaux pour amé-

liorer la représentation que l'agent artificiel a de l' « état » de l'opérateur (physiologie, conscience de situation, charge de travail) sont en cours, afin de permettre une meilleure interaction.

9. Bibliographie

Barber D., Davis L., Nicholson D., « The Mixed Initiative Experimental (MIX) Testbed for Human Robot Interactions with Varied Levels of Automation », *26th Army Science Conference*, 2008.

Bonet P., Llado C., Puijaner R., Knottenbelt W., « PIPE v2.5 : A Petri Net Tool for Performance Modelling », *AAAI 2006 Spring Symposium "Distributed Plan and Schedule Management"*, Stanford University, CA, 2006.

Bradshaw J., Sierhuis M., Acquisti A., Feltovich R., Hoffman R., Jeffers R., Prescott D., Suri N., Uszok A., Van Hoof R., « Adjustable autonomy and human-agent teamwork in practice : An interim report on space application », in *Agent Autonomy* (Hexmoor *et al.*, 2003), chapter 11, 2003.

Brainov S., Hexmoor H., « Quantifying Relative Autonomy in Multiagent Interaction », in *Agent Autonomy* (Hexmoor *et al.*, 2003), chapter 4, 2003.

Brookshire J., Singh S., Simmons R., « Preliminary Results in Sliding Autonomy for Coordinated Teams », *AAAI04 Spring Symposium Series*, Stanford, CA, 2004.

Castelfranchi C., Falcone R., « From Automaticity to Autonomy : the Frontier of Artificial Agents », in *Agent Autonomy* (Hexmoor *et al.*, 2003), chapter 6, 2003.

Chopinaud C., Seghrouchni A., Taillibert P., « Prevention of harmful behaviors within cognitive and autonomous agents », *ECAI 2006 : 17th European Conference on Artificial Intelligence*, Riva del Garda, Italy, p. 205, 2006.

Christensen S., Damgaard Hansen N., « Coloured Petri nets extended with channels for synchronous communication », *Application and Theory of Petri Nets 1994* p. 159-178, 1994.

Clough B. T., « Metrics, Schmetrics ! How The Heck Do You Determine A UAV's Autonomy Anyway ? », *Performance Metrics for Intelligent Systems Workshop*, Gaithersburg, Marylan, 2002.

Coppin G., Legras F., Saget S., « Supervision of Autonomous Vehicles : Mutual Modelling and Interaction Management », *HCI International 2009*, San Diego, CA, USA, 2009.

David R., Alla H., *Discrete, continuous and hybrid Petri nets*, Springer, 2005.

Dehais F., Goudou A., Lesire C., Tessier C., « Towards an anticipatory agent to help pilots », *AAAI 2005 Fall Symposium "From Reactive to Anticipatory Cognitive Embodied Systems"*, Arlington, Virginia, 2005.

Dehais F., Tessier C., Chaudron L., « Ghost : experimenting countermeasures for conflicts in the pilot's activity », *IJCAI03 Conference*, Acapulco, Mexico, 2003.

Dorais G., Bonasso P., Kortenkamp D., Pell B., Schreckenghost D., « Adjustable autonomy for human-centered autonomous systems », *IJCAI'99 Workshop on Adjustable Autonomy Systems*, Stockholm, Sweden, 1999.

Endsley M., « Toward a theory of situation awareness in dynamic systems », *Human Factors*, vol. 37, p. 32-64, 1995.

Finzi A., Orlandini A., « Human-Robot Interaction Through Mixed-Initiative Planning for Rescue and Search Rovers », *AIIA*, Milan, Italy, p. 483-494, 2005.

Fong T., Thorpe C., Baur C., « Collaboration, dialogue and human-robot interaction », *10th International Symposium on Robotics Research*, Lorne, Victoria, Australia, 2002.

Goodrich M., McLain T., Crandall J., Anderson J., Sun J., « Managing Autonomy in Robot Teams : Observations from four Experiments », *ACM/IEEE International Conference on Human-robot interaction*, Washington, DC, 2007.

Goodrich M., Olsen D., Crandall J., Palmer T., « Experiments in adjustable autonomy », *Ijcai'01 Workshop on Autonomy, Delegation and Control : Interacting with autonomous agents*, Seattle, WA, 2001.

Hardin B., Goodrich M., « On using mixed-initiative control : a perspective for managing large-scale robotic teams », *HRI'09, 4th ACM/IEEE International Conference on Human-Robot Interaction*, San Diego, CA, 2009.

Hasslacher B., Tilden M. W., « IEEE Workshop on Bio-Mechatronics », *Living Machines*, Minneapolis, MN, 1996.

Hexmoor H., Castelfranchi C., Falcone R., *Agent Autonomy*, Kluwer Academic Publishers, 2003.

Hexmoor H., McLaughlan B., Tuli G., « Natural human role in supervising complex control systems », *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 21, n° 1, p. 59-77, 2009.

Huang H., Pavek K., Novak B., Albus J., Messin E., « A Framework For Autonomy Levels For Unmanned Systems ALFUS », *AUVSI's Unmanned Systems North America 2005*, Baltimore, Maryland, June, 2005.

Jensen K., Kristensen L., *Coloured Petri Nets : Modeling and Validation of Concurrent Systems*, Springer-Verlag New York Inc, 2009.

Kaber D. B., Endsley M. R., Onal E., « Design of automation for telerobots and the effect on performance, operator situation awareness and subjective workload », *Human Factors and Ergonomics in Manufacturing*, vol. 10, n° 4, p. 409-430, 2000.

Kortenkamp D., Bonasso P., Ryan D., Schreckenghost D., « Traded control with autonomous robots as mixed initiative interaction », *AAAI-97 Spring Symposium on Mixed Initiative Interaction*, Stanford University, CA, March, 1997.

Mumaw R., Sarter N., Wickens C., « Analysis of pilots' monitoring and performance on an automated flight deck », *11th International Symposium on Aviation Psychology*, Colombus, OH, 2001.

Myers K., Morley D., « Human directability of agents », *K-CAP 2001, 1st International Conference on Knowledge Capture*, Victoria, Canada, 2001.

Nau D., Muñoz-Avila H., Cao Y., Lotem A., Mitchell S., « Total-Order Planning with Partially Ordered Subtasks. », *IJCAI-2001*, Seattle, WA, 2001.

Scerri P., Pynadath D., Tambe M., « Adjustable Autonomy for the Real World », in *Agent Autonomy* (Hexmoor *et al.*, 2003), chapter 10, 2003.

Schreckenghost D., Ryan D., Thronesbery C., Bonasso P., Poirot D., « Intelligent control of life support systems for space habitat », *AAAI-IAAI Conference*, Madison, WI, 1998.

Schurr N., Marecki J., Tambe M., « Improving Adjustable Autonomy Strategies for Time-Critical Domains », *AAMAS'09*, Budapest, Hungary, 2009.

Schurr N., Patil P., Pighin F., Tambe M., « Using Multiagent Teams to Improve the Training of Incident Commanders », *AAMAS'06 Industry Track*, Hakodate, Japan, 2006.

Sellner B., Heger F., Hiatt L., Simmons R., Singh S., « Coordinated Multi-Agent Teams and Sliding Autonomy for Large-Scale Assembly », *Proceedings of the IEEE, 94(7)*, 2006.

Sheridan T., Verplank W., Human and Computer Control of Undersea Teleoperators, Technical report, MIT Man-Machine Systems Laboratory, Cambridge, MA, 1978.

Wickens C. D., « Situation awareness : review of Mica Endsley's 1995 articles on situation awareness theory and measurement », *Human Factors*, vol. 50, n° 3, p. 397-403, 2008.

Annexe. Réseaux de Petri

Un *réseau de Petri* $< P, T, F, B >$ (David *et al.*, 2005) est un graphe biparti avec P, ensemble fini de places, T, ensemble fini de transitions, les arcs orientés représentant respectivement la fonction d'incidence avant $F : P \times T \rightarrow \mathbb{N}$ et la fonction d'incidence arrière $B : P \times T \rightarrow \mathbb{N}$. Un *réseau de Petri synchronisé* est tel que le marquage $M : P \rightarrow \mathbb{N}$ représente les conditions vérifiées et des événements sont associés aux transitions ; l'évolution des jetons dans le réseau suit les règles de franchissement des transitions : une transition est franchissable si elle est validée (ses places amont sont suffisamment marquées) et l'événement associé se produit.

Fusion de transitions (Christensen *et al.*, 1994) : deux transitions peuvent être fusionnées même si elles n'appartiennent pas au même réseau. Pour pouvoir être tirées, il faut que simultanément les deux transitions soient validées par leur marquage respectif ; elles sont alors tirées simultanément. Dans le cas où le marquage des réseaux est coloré, les deux transitions doivent être validées par un marquage de la même couleur.

Envoi d'événements entre transitions : lorsqu'une transition t_1 est tirée, elle peut émettre à cet instant un événement e. Si une autre transition t_2 est réceptrice de e, et qu'elle est validée lorsque e est envoyé, alors t_2 est tirée à son tour. En tant que réceptrice, t_2 doit obligatoirement attendre l'envoi de l'événement e pour être tirée. Si aucune transition réceptrice n'est validée par son marquage lorsque e est envoyé, e disparaît et ne persiste pas. Un événement e peut également être porteur de la couleur du marquage validant la transition émettrice.

Les mécanismes de fusion de transition et d'envoi d'événements permettent de synchroniser des réseaux *a priori* non connectés. Cependant, il est toujours possible de mettre à plat ces réseaux et de se ramener à un seul et unique réseau correspondant à ces différents réseaux connectés par fusions ou événements.

Réseaux colorés (Jensen *et al.*, 2009) : il est possible d'associer une couleur aux jetons d'un réseau de Petri, qui correspond à un type. Si un réseau est marqué par des jetons de couleurs différentes, les couleurs n'interfèrent pas : le réseau coloré se comporte comme autant de réseaux non colorés indépendants qu'il y a de couleurs sur son marquage.