# Contribution à l'évaluation des systèmes interactifs orientés agents

Abdelwaheb Trabelsi

LAMIH
LABORATOIRE D'AUTOMATIQUE, DE MÉCANIQUE ET D'INFORMATIQUE, INDUSTRIELLES ET HUMAINES
UMR CNRS 8530

UVHC

Numéro d'ordre : 06/26

# THÈSE

Présentée à

l'Université de Valenciennes et du Hainaut-Cambrésis

en vue de l'obtention du grade de

# DOCTEUR

*Spécialité Automatique et Informatique des Systèmes Industriels et Humains*
*Mention Informatique*

Par


## Abdelwaheb Trabelsi

**Maître ès Sciences**


# *Contribution à l'évaluation des systèmes interactifs orientés agents*

Application à un poste de supervision de transport urbain

Soutenu publiquement le 25 Septembre 2006 devant le jury composé de :

| | | |
|---|---|---|
| **P. Trigano** | Professeur à l'UTC de Compiègne | Examinateur |
| **J. Vanderdonckt** | Professeur à l'université Catholique de Louvain | Rapporteur |
| **A.M. Jolly-Desodt** | Professeur à l'ENSAIT de Roubaix | Rapporteur |
| **S. Hayat** | Chargé de recherche HDR à l'INRETS | Examinateur |
| **C. Kolski** | Professeur à l'université de Valenciennes | Co-Directeur |
| **H. Ezzedine** | Maître de conférences HDR à l'université de valenciennes | Co-Directeur |

LAMIH
LABORATOIRE D'AUTOMATIQUE, DE MÉCANIQUE ET D'INFORMATIQUE, INDUSTRIELLES ET HUMAINES
UMR CNRS 8530

UVHC

Numéro d'ordre : 06/26

# THÈSE

Présentée à

l'Université de Valenciennes et du Hainaut-Cambrésis

en vue de l'obtention du grade de

# DOCTEUR

*Spécialité Automatique et Informatique des Systèmes Industriels et Humains*
*Mention Informatique*

Par


## Abdelwaheb Trabelsi

**Maître ès Sciences**


# *Contribution à l'évaluation des systèmes interactifs orientés agents*

Application à un poste de supervision de transport urbain

Soutenance prévue le 25 Septembre 2006 devant le jury composé de :

| | | |
|---|---|---|
| **P. Trigano** | Professeur à l'UTC de Compiègne | Examinateur |
| **J. Vanderdonckt** | Professeur à l'université Catholique de Louvain | Rapporteur |
| **A.M. Jolly-Desodt** | Professeur à l'ENSAIT de Roubaix | Rapporteur |
| **S. Hayat** | Chargé de recherche HDR à l'INRETS | Examinateur |
| **C. Kolski** | Professeur à l'université de Valenciennes | Co-Directeur |
| **H. Ezzedine** | Maître de conférences HDR à l'université de valenciennes | Co-Directeur |

*A mes Parents*

*A tous ceux qui me sont chers*

# *Remerciements*

Le travail ayant fait l'objet de ce mémoire a été réalisé au Laboratoire d'Automatique, de Mécanique et d'Informatique industrielles et Humaines (LAMIH), U.M.R. C.N.R.S. 8530, à l'université de Valenciennes et du Hainaut-Cambrésis (UVHC) au sein du groupe de recherche « Raisonnement Automatique et Interaction Homme-Machine » (RAIHM) dirigé par le professeur Christophe Kolski, en collaboration avec la société SEMURVAL, dans le cadre du projet SART (Système d'Aide à la Régulation de Trafic) cofinancé par la région Nord-Pas de Calais et le FEDER.

Je tiens, tout d'abord, à remercier chaleureusement le Professeur Christophe Kolski, qui m'a accueilli dans son équipe de recherche, guidé, conseillé et encouragé avec enthousiasme et rigueur durant ma thèse. Sans lui cette thèse n'aurait pas pu voir le jour.

Mes remerciements vont ensuite à Mr Houcine Ezzedine, Maître de conférences HDR pour m'avoir suivi et encadré tout au long de mes travaux de recherche.

Pour m'avoir fait l'honneur d'accepter de participer au jury de cette thèse, je remercie mes deux rapporteurs, Monsieur Jean Vanderdonckt, Professeur à l'université Catholique de Louvain et Mme Anne Marie Jolly-Desodt, Professeur à l'ENSAIT de Roubaix. Je remercie également Mr Philippe Trigano, Professeur à l'UTC de Compiègne et Mr Saïd Hayat, Chargé de recherche HDR à l'INRETS d'avoir accepté d'examiner mon travail.

Je remercie le conseil régional Nord-Pas de Calais et le Fonds Européen de Développement Régional (FEDER) pour avoir financé ma thèse.

Je remercie également tous les membres du LAMIH et particulièrement les membres de l'équipe RAIHM pour leur accueil et les moments agréables passés durant ma thèse ; je pense particulièrement aux personnes avec qui j'ai partagé le bureau 10 : Mohammed-Amine et David. Je remercie également, Anas, Christelle, Mourad et Philippe pour leurs sympathies.

Je tiens à remercier tous les membres et partenaires du projet SART. Nos réunions étaient enrichissantes et productives.

Je ne saurais oublier tous ceux qui sont restés présents quelles que soient les épreuves et qui m'ont toujours soutenu :

Tout d'abord Merci :

à mes parents qui m'ont donné la chance de poursuivre mes études et qui m'ont appris à surpasser les moments difficiles. Merci à mes frères Khaled, Slim et ma sœur Salwa à qui j'ai beaucoup manqué,

à Karim, Mohammed-Amine, Salah, Anli et Hassen pour leurs amitiés, conseils et les bons moments qu'on a passés ensemble. Merci aussi à Mohammed-Ksontini, Mehdi et Skander pour leur soutien pendant leur court séjour à Valenciennes.

à Isabelle qui m'a soutenu, écouté et encouragé pendant les moments difficiles.

Ensuite je voudrais remercier :

mes ami(e)s en Tunisie qui m'ont toujours remonté le moral par leurs appels téléphoniques, je pense particulièrement à Amir, Naceur, Samir, Ahmad, Mansour, Mouna, Chanez, Meriam et à tous ceux qui j'ai oublié de citer.

mes ami(e)s à la résidence Jules Mousseron avec lesquels j'ai passé des bons moments inoubliables.

les enseignants Tunisiens invités avec qui j'ai passé des moments sympathiques et qui m'ont aidé de près ou de loin dans mon travail : je pense particulièrement à Adel, Slah-Eddine, Mohammad et Mounir.

les familles Pennequin, Sobczak et Miemiec pour leurs encouragements et leur soutien.

Merci à tous ceux dont je n'ai pas parlé et qui mériteraient d'être dans ces pages.

# *Table des matières*

# *Liste des figures*

# *Liste des Tableaux*

# *Introduction Générale*

Cette thèse se situe dans le domaine de l'Interaction Homme-Machine, domaine particulièrement riche en concepts, méthodes, modèles et outils [Helander, 1988 ; Helander *et al.*, 1997 ; Jacko et Sears, 2003], tout en s'intéressant plus particulièrement aux systèmes industriels complexes et leur supervision [Rasmussen, 1986 ; Kolski, 1997, Ezzedine, 2002].

Que ce soit en rapport ou non avec les systèmes industriels complexes, l'évaluation des systèmes interactifs sous l'angle de l'utilité et de l'utilisabilité est, depuis plus d'une trentaine d'années, un domaine à part entière largement traité dans la communauté en Interaction Homme-Machine, aussi bien nationale qu'internationale.

Les méthodes et outils d'évaluation actuellement disponibles sont nombreux et variés (observations, oculométrie, interviews, mouchards électroniques, questionnaires, tests utilisateurs, méthodes d'inspection, systèmes automatiques à base de connaissances, etc.). Tous présentent des avantages et des inconvénients et aucun d'eux ne peut prétendre à une évaluation exhaustive d'un système. Avec l'apparition de nouvelles architectures et de nouveaux systèmes à base d'agents, de nouvelles problématiques d'évaluation des systèmes interactifs orientés agents apparaissent. Parmi les problèmes les plus traités actuellement nous pouvons citer : le manque d'architectures dédiées aux systèmes interactifs orientés agents, l'absence d'outils exploitant les spécificités de telles architectures qui permettent l'extraction des actions et des réactions de l'utilisateur et du système dans un but d'évaluation, le manque d'outils (automatiques ou non) d'aide à l'évaluation dédiés aux systèmes interactifs orientés agents.

Dans ce courant de recherche, le travail présenté dans ce mémoire est une contribution à l'évaluation des systèmes interactifs orientés agent, l'évaluation étant basée sur un outil de recueil d'information. En effet, nous nous intéressons particulièrement à la semi-automatisation de l'évaluation des systèmes interactifs. Nous proposons, dans ce cadre, un mouchard électronique permettant l'acquisition et l'analyse des interactions homme-machine dans des systèmes orientés agents ; celui-ci peut être vu comme un outil d'aide aux évaluateurs.

Ce mémoire, organisé en cinq chapitres, débute par un chapitre intitulé « architecture des systèmes interactifs ». Ce chapitre présente sans souci d'exhaustivité, mais plutôt de représentativité, les principes des modèles d'architecture les plus cités et étudiés actuellement. En effet, aujourd'hui, les modèles d'architecture proposés assurent la structuration du système interactif en un ensemble de composants, tout en définissant leurs fonctions et leurs relations. Cette structuration doit faciliter le prototypage et le test des interfaces d'une application sans avoir à développer complètement les fonctionnalités de l'application. Nous présentons dans ce premier chapitre les trois principales catégories d'architecture dédiées à la conception des interfaces homme-machine : les architectures centralisées, les architectures réparties et les architecture hybrides. Cette présentation nous permettra de construire un cadre d'étude pour l'introduction de notre architecture dédiée au système interactif de supervision présentée plus loin, dans le chapitre 3.

Le deuxième chapitre, appelé « Des méthodes et techniques usuelles d'évaluation des systèmes interactifs aux outils d'aide à l'évaluation automatique », s'intéresse aux méthodes d'évaluation des systèmes interactifs et plus particulièrement des interfaces homme-machine. Nous y présentons un panorama des principales méthodes d'évaluation actuellement disponibles. Ces méthodes sont regroupées en trois grandes catégories : (1) les méthodes basées sur des techniques d'observation de l'utilisateur réel et de recueil des données de l'interaction, (2) les méthodes basées sur l'intervention d'experts (en interaction homme-machine, en psychologie cognitive ou en ergonomie) et (3) les méthodes analytiques basées sur des modèles formels prédictifs intégrant des connaissances sur la tâche et sur des grammaires ou des modèles formels de qualité de l'interface. Dans chaque catégorie sont décrites la ou les méthodes d'évaluation les plus représentatives ou celles qui ont servi de base à d'autres méthodes. Dans la mesure où nous visons la proposition d'un outil d'évaluation partielle comprenant une partie automatique (Cf. chapitre 3), nous terminons ce chapitre avec la présentation de quelques méthodes d'évaluation automatique.

En effet, l'automatisation de l'évaluation peut porter sur la capture, l'analyse ou la critique des interfaces homme-machine.

Jusqu'à présent, dans ces deux premiers chapitres, notre travail consiste en une présentation de l'état de l'art sur, d'une part les architectures des systèmes interactifs et particulièrement les interfaces homme-machine et d'autre part, sur les méthodes d'évaluation des systèmes interactifs. Cet état de l'art nous a offert un cadre d'étude pour la proposition d'un principe de couplage entre architecture à base d'agents de système interactif et son évaluation, présenté au troisième chapitre.

En effet, intitulé « Principe de couplage entre architecture à base d'agents de système interactif et son évaluation », le troisième chapitre présente notre principale contribution : (1) on y propose les principes d'une architecture à base d'agents dédiée aux systèmes interactifs. Nous y décrivons l'origine de cette architecture ainsi que ses différents composants : « *agents d'application* », « *agents contrôleur de dialogue* » et « agents *de présentation* ». Les réseaux de Petri (RdP) de haut niveau et particulièrement les RdP agents seront utilisés pour la modélisation des communications entre les agents ainsi que la dynamique de l'interface. (2) Dans ce chapitre, nous proposons aussi un outil d'aide à l'évaluation basé sur un mouchard électronique baptisé MESIA (Mouchard électronique dédié à l'Evaluation des Systèmes Interactifs orientés Agents). L'idée forte derrière ce mouchard réside dans son couplage avec l'architecture du système à évaluer. En effet, il est constitué de plusieurs *agents mouchard* déduits à partir de l'architecture du système à évaluer et plus particulièrement à partir des agents de présentation. Dans cette thèse en IHM, nous nous focalisons en effet sur le module de présentation.

Le projet SART (Système d'Aide à la Régulation de Trafic), impliquant plusieurs laboratoires et instituts de recherche (INRETS, LAGIS, LAMIH) et un exploitant (la SEMURVAL), nous servira de cadre d'application au quatrième chapitre intitulé « Mise en application du couplage architecture/module d'évaluation dans un contexte lié aux transports ». Notre contribution à ce projet concerne la réalisation d'un système d'aide à l'information (SAI) permettant d'assister les régulateurs humains d'un réseau de bus et de tramways (celui de Valenciennes) et de faciliter leurs tâches. Après l'analyse, la modélisation, la spécification et le maquettage du SAI, nous détaillerons ensuite l'implémentation du mouchard électronique ainsi que son couplage avec le SAI. Ce mouchard nous servira d'outil de base pour l'évaluation du SAI.

Enfin, dans le dernier chapitre appelé « Première évaluation et perspectives de recherche », nous présentons une première évaluation (menée en laboratoire) du Système d'Aide à l'Information (SAI) réalisé. Cette évaluation se basera essentiellement sur le système d'aide à l'évaluation qui assure le couplage entre le mouchard électronique et le SAI. Nos perspectives de recherche à court terme visent à améliorer le SAI en tenant compte des résultats obtenus lors de la première évaluation effectuée. A moyen et long terme, nous visons l'élaboration d'une méthode de mise en place d'un couplage entre architecture à base d'agents du système interactif et son évaluation. Nous proposons également des améliorations du mouchard électronique MESIA ; ces améliorations concernent en particulier le module de génération, confrontation et simulation de RdP, de même que la formalisation et l'intégration de règles ergonomique dans l'outil d'aide fourni aux évaluateurs.

*Chapitre 1*

# *Architecture des systèmes interactifs*

1. *Introduction*
2. *Architectures Centralisées*
3. *Architectures réparties*
4. *Architectures hybrides*
5. *Conclusion*

# I. Introduction

C'est au début des années 80 environ que les concepteurs et les développeurs des systèmes interactifs ont mis progressivement en évidence les problèmes liés à la conception des grosses applications interactives. En effet, le développement se faisait depuis longtemps d'une façon souvent artisanale et intuitive. Comme le montre la figure I.1, les aspects liés à l'interface homme-machine étaient dissous dans l'application. Cet état de fait conduisait souvent à des systèmes dont la structure logicielle manquant de modularité, maintenabilité, et aussi portabilité. Ces utilisateurs prenaient souvent connaissance du système interactif seulement en fin de réalisation, celui-ci pouvant ne correspondre que peu à ses besoins réels.

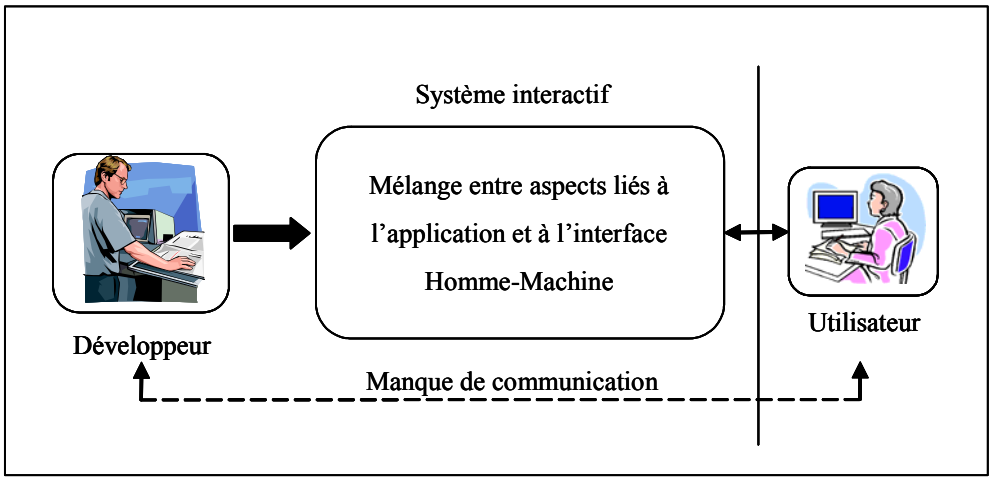



**Figure I.1.** Lacunes dans le développement des premiers systèmes interactifs

Devant la complexité croissante des applications, plusieurs modèles d'architecture pour la conception des systèmes interactifs ont vu le jour. En effet, un modèle d'architecture est avant tout un guide pour aider les concepteurs d'un logiciel à en maîtriser sa complexité. Á ce sujet, [Coutaz et Nigay, 2001] définissent l'architecture d'un système informatique comme étant un ensemble de structures comprenant chacune : des composants, les propriétés extérieurement visibles de ces composants et les relations que ces composants entretiennent.

Aujourd'hui, les modèles d'architecture proposés assurent la structuration du système interactif en un ensemble de composants, tout en définissant leurs fonctions et leurs relations. Cette structuration doit faciliter le prototypage et le test des interfaces d'une application sans avoir à développer complètement les fonctionnalités de l'application. Désormais, l'utilisateur n'est en principe plus ignoré durant la phase de conception du logiciel.

En règle générale, tous les modèles d'architecture partent du principe qu'un système interactif comporte une partie « interface » et une partie « application ». De ce fait, la plupart des modèles comprennent (presque tous) des éléments en contact direct avec l'utilisateur (présentations), des éléments en contact direct avec le noyau fonctionnel ou qui en font partie (interfaces du noyau fonctionnel, abstractions, modèles), et des éléments articulatoires (contrôleurs, adaptateurs) [Texier, 2000 ; Coutaz et Nigay, 2001 ; Dragicevic, 2004].

Malgré ces points communs, l'approche empruntée par chaque modèle est différente. En effet, on distingue deux grands groupes de modèles de référence : (1) les modèles centralisés (appelés aussi modèles fonctionnels) tels que le modèle de Seeheim [Pffaf, 1985] et le modèle

ARCH [Bass *et al.*, 1991] ; (2) les modèles repartis (appelés aussi modèles multi-agents) tels que le modèle MVC [Goldberg, 1983], le modèle PAC [Coutaz, 1987, 1990] et le modèle AMF [Tarpin-Bernard et David, 1999]. Par ailleurs, des modèles mixtes appelés aussi modèles hybrides tels que PAC-Amodeus [Nigay, 1994] et le modèle $H^4$ [Guittet, 1995] tentent de tirer partie des avantages respectifs de ces deux groupes de modèle.

On se propose dans ce chapitre de présenter, sans souci d'exhaustivité, mais plutôt de représentativité, les principes des modèles d'architecture les plus cités et étudiés actuellement. Cette présentation nous permettra de construire un cadre d'étude pour l'introduction de notre architecture dédiée au système interactif de supervision présentée plus loin, dans le chapitre 3. Pour ce faire nous avons structuré ce chapitre en trois parties.

La première partie est consacrée aux architectures centralisées. Nous y présentons le premier modèle d'architecture qui a mis en évidence la séparation entre les aspects liés à l'IHM et l'application. Ce modèle fût le modèle *Langage* [Foley et Van Dam, 1982]. Plusieurs modèles inspirés de celui-ci ont permis de rapprocher les développeurs et les utilisateurs des systèmes interactifs. Nous présentons à titre d'exemple le modèle de *Seeheim* et ses variantes ainsi que le modèle *ARCH*. La deuxième partie est dédiée aux architectures réparties. Nous y présentons les modèles d'architecture multi-agents (répartis) les plus connus et cités dans la littérature tels que le modèle *PAC*, *MVC* et *AMF*. Afin de mieux se positionner par rapport à ce type d'architecture, une introduction aux notions d'agent et de systèmes multi-agents sera aussi présentée. Les modèles hybrides (basés sur un mariage entre les modèles centralisés et les modèles repartis) tels *PAC-Amodeus*, le modèle *Multicouche* et *$H^4$* feront l'objet de la troisième partie.

## II. Architectures centralisées

Les architectures centralisées sont historiquement les premiers modèles d'architecture à avoir vu le jour. Ils décomposent un système interactif en plusieurs couches d'abstraction, de celui responsable des interactions directes avec l'utilisateur (*Présentation*) jusqu'au noyau fonctionnel (*application*). En revanche, ils ne définissent pas (ou très peu) ni la structure interne des modules, ni les interfaces d'échanges. Ces architectures sont aussi appelées architectures fonctionnelles [Ezzedine, 2002] ou encore architectures globales [Fekete, 2005].

Le premier modèle d'architecture qui a mis en évidence la séparation entre les aspects liés à l'IHM et l'application fût le modèle *Langage* [Foley et Van Dam, 1982]. Plusieurs modèles inspirés de celui-ci ont permis de rapprocher les développeurs et les utilisateurs des systèmes interactifs. Ce modèle est maintenant présenté.

### II.1. Le modèle Langage

Le modèle *Langage* (Cf. figure I.2) a été proposé par [Foley et Van Dam, 1982]. Ce modèle est basé sur l'analogie entre l'interaction homme-machine et le dialogue humain. Afin de pouvoir interagir avec le système, l'être humain doit utiliser un langage compréhensible par ce dernier. En effet, Foley et Van Dam distinguent deux types de langages ; un premier langage avec lequel l'utilisateur communique avec l'application ; et un deuxième langage qui assure la communication de l'application avec l'utilisateur. Par ailleurs, le langage est considéré comme l'interface entre l'application et l'utilisateur.

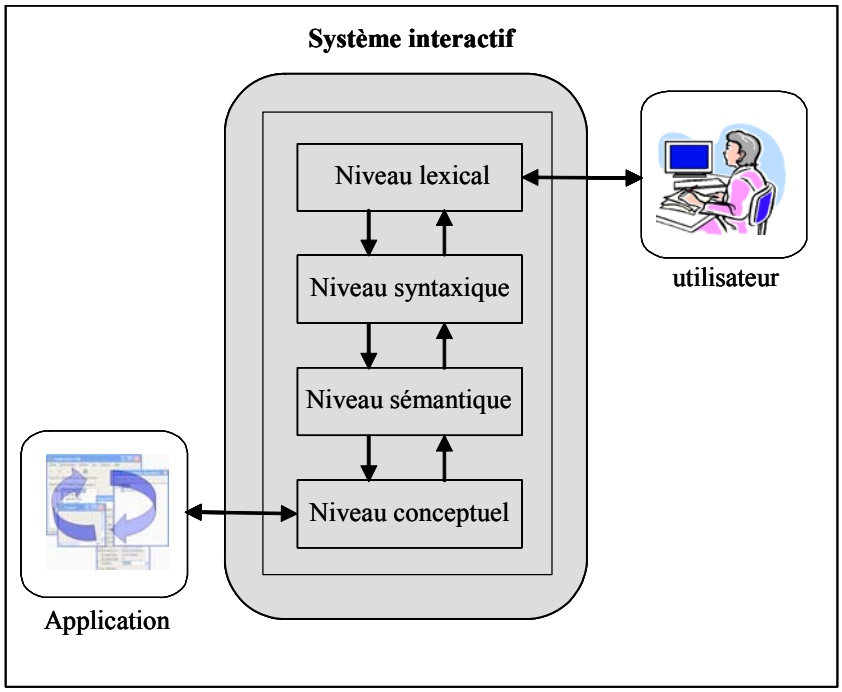



**Figure I.2.** Modèle Langage [Foley et Van Dam, 1982]

Le modèle *Langage* vise à assurer la communication entre l'utilisateur et le système en décomposant l'interface en quatre niveaux d'abstraction :

- **Niveau conceptuel** : définit les objets de l'application ainsi que leurs attributs, leurs relations et les actions possibles sur ces objets. Ce niveau correspond à l'application. Par exemple, dans un simple éditeur de texte, les objets manipulés sont les lignes et les fichiers ; la relation entre ces deux objets est telle qu'un fichier est composé d'une séquence de lignes. Les opérations sur l'objet *ligne* sont : insérer, supprimer, déplacer. Les opérations sur l'objet *fichier* sont : créer, supprimer, déplacer, renommer et copier ;

- **Niveau sémantique** : définit, d'une part l'utilisation des objets manipulés ainsi que leurs comportements, et d'autre part les erreurs sémantiques qui peuvent se produire tout en proposant une procédure pour leur résolution. En effet, ce niveau décrit les fonctionnalités du système en représentant les concepts et le savoir-faire dans un domaine donné. L'ensemble des actions possibles sur les différents objets de l'application est défini sans se préoccuper des liens séquentiels entre actions. Par exemple, pour la création d'un fichier, il faut avoir le nom de fichier ainsi que le nom du volume où il sera créé. Si le nom de fichier à créer existe déjà, il faut gérer l'erreur (plusieurs procédures d'erreur peuvent être envisageables) ;

- **Niveau syntaxique** : détermine les séquences des actions de l'utilisateur en termes d'entrées/sorties selon des règles syntaxiques bien définies. Ces séquences seront transmises au niveau supérieur après leur transformation au format spécifique. Par exemple, une icône de sauvegarde d'un éditeur de texte n'est plus perçu par l'utilisateur comme tel si on la décompose en primitives graphiques présentées séparément (rectangle, cercle, ligne, etc.) ;

- **Niveau lexical** : correspond à l'ensemble des éléments perçus par l'utilisateur. Il assure, d'une part la gestion des événements en entrée, c'est-à-dire provenant de l'utilisateur, et d'autre part l'affichage des sorties (les caractères saisis au clavier, les sons élémentaires d'un synthétiseur, les primitives graphiques d'une bibliothèque d'objets, etc.). Par exemple, les primitives de tracé de droite et de remplissage de régions permettent de construire l'élément syntaxique « icône ».

Les principes de base du modèle *Langage* ont été exploités par de nombreux chercheurs, et ont influencé la définition de plusieurs modèles génériques. Comme précisé dans [Coutaz, 1990], Foley et Van Dam furent les premiers à « plaquer » une vue linguistique sur l'interaction homme-ordinateur, mais le premier qui a exprimé cette source d'inspiration sous une forme utilisable fut le modèle *Seeheim* présenté ci-dessous.

## II.2. Le modèle de Seeheim

Le modèle de *Seeheim* (Cf. figure I.3) tient son nom du lieu où il a été défini. En novembre 1983, eût lieu à SEEHEIM en Allemagne un workshop sur les systèmes de gestion des interfaces utilisateur (UIMS)[1]. Le modèle d'architecture issu de ce workshop a été le premier à établir clairement la séparation entre l'interface et l'application.

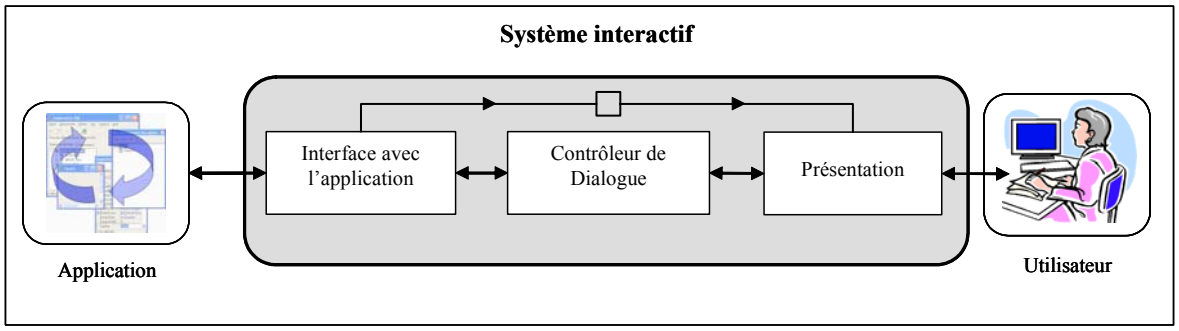



**Figure I.3.** Modèle de Seeheim

Le modèle de *Seeheim* raffine la structuration du système interactif en trois composantes :

- **La présentation** : est responsable de la représentation physique du système. Elle prend en charge les éléments d'interface qui apparaissent sur l'écran et les différentes interactions physiques avec l'utilisateur. Ce composant gère, d'une part la production effective des sorties perceptibles par l'utilisateur (affichage, production de sons, vibrations de joystick[2],…) en provenance du *contrôleur de dialogue*, et d'autre part les événements en provenance de l'utilisateur (Click souris, clavier, parole, joystick...). Ces événements sont convertis en unités de dialogue plus abstraites, adaptées au besoin du composant *contrôleur de dialogue*. Par rapport au modèle *Langage*, la présentation du modèle *Seeheim* correspond globalement au *niveau lexical* ;

- **Le contrôleur de dialogue** : prend en charge la structuration et l'analyse du dialogue entre l'utilisateur et l'application. Le *contrôleur de dialogue* a par ailleurs en charge le suivi du dialogue qui s'instaure entre l'usager et l'application. Selon

[1] On appelle UIMS (User Interface Management System) selon [Foley, 1984] un outil utilisé par un administrateur d'interface utilisateur pour construire les interfaces utilisateur des applications.
[2] Périphérique d'entrée/sortie utilisé dans les jeux et également utilisé dans certains types d'avions.

[Coutaz 1990], les requêtes émises par l'utilisateur sont converties en phrases syntaxiques exploitables par le module application. Dans le sens inverse, le contrôleur reçoit des phrases de sortie abstraites qu'il transmet vers les éléments spécialisés de la présentation. Ce composant peut être considéré comme le *niveau syntaxique* du modèle *Langage* ;

- **L'interface avec l'application** : définit la vue qu'a le *contrôleur de dialogue* de l'application. En effet, l'application réunit les concepts et les opérations propres à l'accomplissement de tâches dans un domaine précis ; elle est dépositaire du modèle interne du système. *L'interface avec l'application* désigne la sémantique de l'application indépendamment de l'interface. De ce fait, et en théorie, la modification de la présentation peut être effectuée sans remettre en cause les fonctions de l'application et réciproquement. Il est à noter que *l'interface avec l'application* correspond au *niveau sémantique* du modèle *Langage*.

Malgré le fait que le modèle de *Seeheim* propose une séparation en trois composants du système interactif, l'application de ce principe reste toujours difficile à réaliser ; et même si elle l'est au début du développement du logiciel, elle peut ne plus l'être après quelques modifications et évolutions que subit le logiciel.

Sur la figure I.3, la relation qui lie le composant de *présentation* à *l'interface avec l'application* montre en fait que la séparation totale entre l'interface et le noyau fonctionnel est difficile à atteindre. En théorie, toutes les informations en provenance du noyau fonctionnel et devant être présentées à l'utilisateur doivent être d'abord traitées par le *contrôleur de dialogue* avant d'être transmises au composant *présentation*. En pratique, il s'avère que certaines informations ne font que transiter par le *contrôleur de dialogue* sans subir aucun traitement de la part de celui-ci. On constate alors une communication directe entre le composant de *l'interface avec l'application et présentation* (voire même le noyau fonctionnel lui-même). Ceci est particulièrement vrai pour des opérations d'affichage complexe et/ou exigeant un temps de réponse minimal [Bellik, 1995].

L'un des reproches souvent fait à ce modèle est d'être monolithique, mais selon [Patry, 1999] ce reproche résulte d'une lecture trop stricte du modèle : rien n'interdit de découper le contrôle de l'application en plusieurs modules s'occupant chacun d'une partie indépendante du dialogue. A titre d'exemple, l'architecture proposée par [Keivunen et Mantyla, 1988], connue sous le nom « *Hut Windows* », définit d'une manière plus précise la communication entre les composants. En effet, la communication entre l'application et le *contrôleur de dialogue* est réalisée par messages, tandis que la communication entre la *présentation* et *le contrôleur de dialogue* se fait par une pile d'entrées dans un sens, et une série de commandes *HDL* (Hut Definition Language) dans l'autre sens.

Plusieurs autres variantes du modèle de *Seeheim* ont été proposées [3] :

- Le modèle de ***Seeheim étendu*** [Karsenty et Weikart, 1991] vise à inclure complètement l'application dans le modèle. En fait, il remplace les composants *interface avec l'application* et *application* du modèle de *Seeheim* par trois composants : *abstraction de l'interface*, *contrôleur de l'application* et *abstraction de l'application*.

[3] Le lecteur intéressé peut trouver dans [Duval, 1994] un panorama de modèles d'architecture apparentés au modèle de Seeheim où chaque modèle est analysé de façon générale et est appliqué à un exemple simple.

- Le modèle de ***Seeheim modifié*** [Dance *et al.*, 1987] divise un système interactif en quatre composants (Cf. figure I.4) : *agent de la station de travail*, *gestionnaire de dialogue*, *support sémantique* et *application*. Il propose de partager les informations concernant l'application entre deux composants, un composant *application* proprement dit, et un composant *support sémantique* qui serait chargé d'assurer la rétroaction sémantique. Il est à noter que l'idée forte derrière ce modèle est de fournir un haut niveau de rétroaction sémantique. Selon [Duval, 1994], en se référant au modèle de *Seeheim*, le composant *présentation* peut être assimilé à *l'agent de station de travail* ainsi qu'une partie du *gestionnaire de dialogue* ; et le composant *contrôleur de dialogue* au *gestionnaire de dialogue* (privé du *support sémantique*). Quand au composant *interface avec l'application*, il peut être assimilé au composant *support sémantique*.

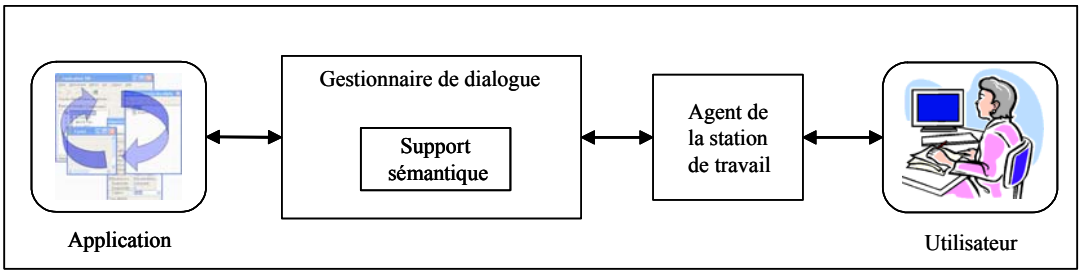



**Figure I.4.** Modèle de Seeheim modifié

- Le modèle de ***Hudson*** [Hudson, 1987] introduit la notion « d'objets actifs » qui jouent le rôle du *contrôleur de dialogue*. Ces objets sont définis comme des types abstraits comportant une structure de données et un ensemble de procédures définissant leur comportement.

Quoi qu'il en soit, le modèle de *Seeheim* reste la référence en ce qui concerne les modèles d'architectures logicielles pour les applications interactives. Nous verrons plus loin, dans le chapitre 3, comment nous exploiterons ce modèle afin d'aboutir à une architecture adaptée aux systèmes interactifs de supervision.

Le modèle le plus cité, dérivé du modèle de *Seeheim*, reste le modèle *ARCH* [Bass *et al.*, 1991] qui est présenté maintenant.

## II.3. Le modèle ARCH

Le modèle *ARCH* est issu du séminaire « User Interface Developer's Workshop » en 1991 [Bass *et al.*, 1991]. Il est une extension du modèle de *Seeheim*. On y retrouve les trois composants de *Seeheim* (*présentation*, *contrôleur de dialogue* et *application*). Il définit une décomposition fonctionnelle du système interactif en cinq éléments distincts (Cf. figure I.5).

Les deux composants imposés par la réalité sont *le noyau fonctionnel* et *l'interaction*. Ces deux composants constituent les pieds de l'arche. Les cinq éléments du modèle *ARCH* sont :

- **Le composant noyau fonctionnel** : permet la manipulation de concepts à travers l'exécution de fonctions spécifiques au domaine ;

- **Le composant adaptateur du domaine** : assure une totale indépendance entre le *contrôleur de dialogue* et le *noyau fonctionnel*. *L'adaptateur du domaine* permet de déclencher les procédures nécessaires à la tâche de l'utilisateur, de détecter et réparer les erreurs sémantiques et d'implémenter les protocoles de communication entre dialogue et application ;

- **Le composant contrôleur de dialogue** : assure l'enchaînement des tâches ainsi que la traduction de données entre la *présentation* et *l'adaptateur du domaine* ;

- **Le composant présentation** : prend en charge le lien entre le composant *contrôleur de dialogue* et le composant *d'interaction*. Il fournit au *contrôleur de dialogue* un ensemble d'objets de présentation conceptuels indépendants de toute boîte à outils. Cette indépendance assure une totale liberté pour le choix des composants d'interaction proposés à l'utilisateur ;

- **Le composant interaction** : gère les interactions physiques avec l'utilisateur au moyen des interacteurs (communément appelés aussi *widgets*) d'une boîte à outils.

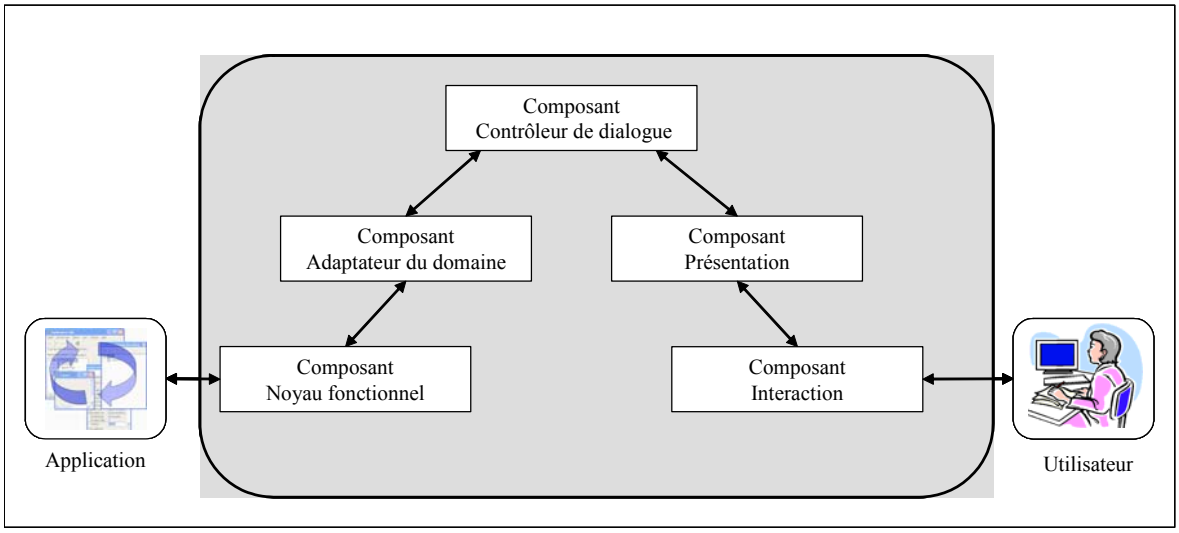



**Figure I.5.** Le modèle ARCH

Par rapport au modèle de *Seeheim*, le modèle *ARCH* se différencie par l'introduction de deux composants qui sont : *l'adaptateur de domaine* et *l'interaction*. Si cette introduction possède l'avantage de séparer clairement en cinq modules un système interactif tout en assurant une meilleure portabilité (garantie par l'introduction du composant *interaction*), elle impose une lourdeur de transition d'information entre les différents composants. En effet, le passage d'un niveau à un autre nécessite l'analyse et la traduction des informations à transmettre au niveau suivant. De ce fait, le méta-modèle *Slinky*[4] (Cf. figure I.6), inspiré du nom de ce jouet, a été introduit par [UIMS, 1992] pour mettre en évidence les notions de choix et de compromis inhérents au développement des applications interactives [Dragicevic, 2004].

[4] Le terme *Slinky* provient d'un jouet flexible qui une fois mis en mouvement voit son centre de gravité se déplacer suite à un changement dans la répartition de sa masse.

**Figure I.6.** Le jouet Slinky ayant inspiré le modèle du même nom

L'idée forte derrière ce *méta-modèle* est que les fonctionnalités du système interactif peuvent glisser d'un composant de l'ARCH à l'autre en fonction des objectifs des développeurs. Par exemple, si le temps d'exécution représente une contrainte, on peut « court-circuiter » le composant *adaptateur du domaine* ainsi que le composant *contrôleur de dialogue* et faire transiter des valeurs de données directement entre le composant *noyau fonctionnel* et le composant *présentation* [UIMS, 1992].

## II.4. Conclusion sur les architectures centralisées

Les architectures centralisées ont été les premières à proposer un découpage d'un système interactif en différents modules. Ainsi, elles assurent la séparation, d'une manière explicite, entre la partie application et la partie interface. Cette décomposition est fondamentale puisque elle assure une indépendance entre le module responsable de l'interaction avec l'utilisateur et l'application. De cette façon, on peut apporter des améliorations à l'interface sans avoir à se soucier de la partie application et vice versa. Ces architectures fournissent ainsi un cadre de pensée, permettant de séparer les difficultés de conception ou d'analyse d'un système interactif en le structurant en différents modules.

Certes, les architectures centralisées présentent l'avantage de décomposer un système interactif en plusieurs modules. Cependant, elles ne définissent ni la description de la structure interne des modules, ni les interfaces d'échange entre eux. En effet, comme nous l'avons vu avec le modèle de Seeheim ou le modèle *ARCH*, le dialogue entre les différents composants d'un système interactif n'est pas explicitement décrit et doit être géré à part entière par les concepteurs du système.

Or au sein des systèmes interactifs dédiés à la supervision, ce manque de structuration et de formalisme de communication entre les modules, constitue un problème pour la modélisation des différents modules (contrôle et commande). De ce fait, on s'aperçoit que les modèles d'architectures centralisées ne sont pas parfaitement adaptés pour les systèmes interactifs de supervision.

Le manque de structuration et de détail concernant les modules de ces architectures, ainsi que l'émergence de la programmation orientée objet au début des années 80, ont donné naissance à un nouveau type d'architectures appelées : architectures réparties ou encore architectures multi-agents. Elles visent une décomposition structurelle plus fine avec la possibilité d'introduction de la notion du parallélisme (notion absente dans les architectures centralisées).

# III. Architectures réparties

Les modèles d'architectures réparties permettent de représenter, à un niveau de granularité beaucoup plus fin que les modèles centralisés, la séparation entre les modules identifiés ainsi que la communication entre eux. En effet, si les architectures centralisées séparent un système interactif en trois différentes fonctions (application, contrôle, présentation), les architectures réparties proposent de les regrouper au sein d'une même entité.

Ces modèles sont aussi appelés modèles d'architecture multi-agents. Leur apparition résulte de l'avènement de la programmation orientée objet et de la notion de stimulus-réponse. En effet, ces architectures répartissent le système interactif en un ensemble d'agents réagissant à la réception d'événement ou de message, d'où l'expression « multi-agents ». Les différents modèles d'architectures réparties reposent donc sur le concept agent dont le rôle et la composition varient d'une architecture à une autre [Depaulis, 2002].

Afin d'éclaircir la notion d'agent tel que nous l'abordons dans ce mémoire et de mieux se positionner par rapport aux concepts introduits par les architectures orientées agents, nous rappelons dans la section suivante les notions d'agent et de système multi-agents.

## III.1. Notions d'agents et de système multi-agents

### III.1.1. Notion d'agent

Jusqu'à présent, il n'existe pas de définition universelle, adoptée par tous, du terme agent. Cependant, de nombreuses définitions pour ce terme existent [Ferber, 1995 ; Wooldridge et Jennings, 1995 ; Bradshaw, 1997 ; Franclin et Graesser, 1996[5] ; Luck et d'Inverno, 2001]. Les définitions données divergent principalement par rapport aux types d'applications envisagées et aux problématiques de recherche [Mandiau et Grislin-Le Strugeon, 2001].

Malgré ce flou relatif dans la définition d'un agent, les chercheurs sont généralement d'accord sur les fonctionnalités principales d'un agent : « Un agent agit en fonction d'un ensemble d'informations reçues et perçues de son environnement et en fonction des buts qu'il poursuit » [Ferber, 1995 ; Russell et Norvig, 1995]. On s'aperçoit alors qu'un agent est plus facile à définir par ce qu'il fait que par ce qu'il est. De ce fait, un agent est capable de : (1) percevoir, au moins partiellement son environnement, (2) agir sur son environnement, (3) se comporter de manière rationnelle.

En réalité, au sein de la communauté informatique, il existe deux approches assez différentes des univers multi-agents (notons aussi qu'il existe des agents hybrides, non traités ici [Chaib-Draa et Levesque, 1996 ; Chaib-Draa *et al.*, 2001]):

- Les chercheurs qui travaillent dans le domaine de l'intelligence artificielle construisent des univers composés **d'agents cognitifs** qui possèdent chacun des connaissances sur un domaine particulier. Ces agents sont généralement des entités de taille assez importante qui coopèrent pour résoudre des tâches complexes. La modélisation des relations entre les agents constitue alors un problème à part entière [Mandiau 2000 ; Chaib-Draa *et al.*, 2001 ; Hanon *et al.*, 2006].

[5] Dans cet ouvrage les auteurs passent en revue une quinzaine de définitions pour le terme agent selon plusieurs points de vue.

- En intelligence artificielle distribuée, les chercheurs qui travaillent dans la modélisation des interfaces homme-machine considèrent les systèmes interactifs comme étant des univers composés **d'agents réactifs**. Ainsi, le comportement de ces agents est essentiellement dicté par les interactions (directes ou indirectes par la communication avec d'autres agents) avec l'utilisateur [Chalon, 2004]. Nous verrons plus loin (Cf. III.2, 3, 4) comment ces agents représentent les composants de base des architectures réparties.

  Un agent réactif ou interactif est alors défini comme un agent dont le comportement est la conséquence de ses observations et de ses interactions avec l'utilisateur [Coutaz et Nigay, 2001]. A la différence des agents cognitifs, la granularité des agents réactifs peut être très petite. Par exemple, un agent réactif peut être réduit à un simple bouton et son intelligence est alors très restreinte.

Il est évident que les deux approches ne sont pas complètement antinomiques puisque les modèles cognitifs peuvent être pris en compte pour la constitution des agents réactifs. C'est d'ailleurs ce rapprochement qui permet d'envisager l'émergence de systèmes interactifs auto-adaptatifs (s'adaptant automatiquement à l'utilisateur) [Tarpin-Bernard et David, 1999].

Ainsi pour définir un agent nous citons la définition proposée par [Ferber, 1995]. L'auteur définit un agent d'une façon générique (sans préciser s'il s'agit d'agent réactif ou cognitif) comme étant une entité physique ou virtuelle qui :

- possède des ressources propres ;
- est capable de percevoir son environnement ;
- est capable d'agir dans un environnement ;
- peut communiquer directement avec d'autres agents ;
- est mue par un ensemble de tendances sous la forme d'objectifs individuels ou d'une fonction de satisfaction.

Cependant, cette définition nous semble assez large et ne reflète pas d'une façon adéquate la définition d'un agent au sein de la communauté de recherche en interaction homme-machine. La définition qui nous semble la plus appropriée est celle de [Coutaz et Nigay, 2001] : un agent est défini en tant que système de traitement de l'information : il se distingue par un jeu d'opérations (un ensemble de méthodes qui constituent les actions que peut effectuer un agent), des mécanismes d'entrée/sortie (qui permettent à l'agent de communiquer et d'agir) et une capacité à représenter un état que l'on appelle vecteur d'état (une mémoire qui permet à l'agent de mémoriser son état ainsi que l'état de son environnement). Nous retiendrons alors cette définition pour la présentation des architectures multi-agents (Cf. III.2, 3, 4) ainsi que l'architecture que nous proposons dans le chapitre 3.

Afin de réaliser une tâche donnée ou de parvenir à un objectif, l'agent doit être toujours à l'écoute de son environnement. Il perçoit alors, non seulement les différents messages transmis par les autres agents, mais aussi les événements qui se produisent dans son environnement. Suite à cette perception, l'agent passe dans une phase de raisonnement, souvent appelée phase de cognition. Dans cette phase, l'agent décide de(s) action(s) qu'il doit faire, comment et quand les faire. L'agent peut éventuellement décider de ne rien faire.

Décider d'effectuer une action n'est pas suffisant, l'agent doit alors agir et mettre en oeuvre les décisions prises dans la phase de cognition. Ces trois phases visibles à la figure I.7 forment un cycle par le bouclage de la phase d'action sur la phase de perception [Mandiau 2000 ; Chaib-Draa *et al.*, 2001].

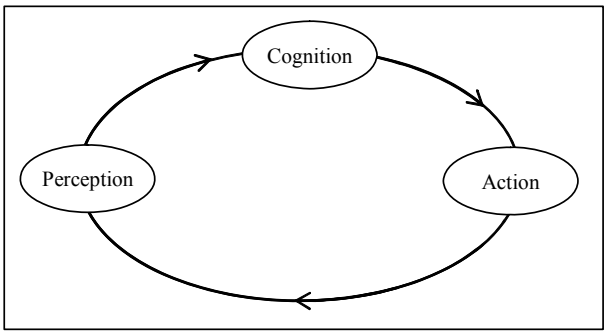



**Figure I.7.** Les trois phases générales de réalisation d'une tâche par un agent

## III.1.2. Systèmes multi-agents

Un système multi-agents est un ensemble d'entités qui coordonnent leurs connaissances, buts, expériences et plan pour agir ou résoudre des problèmes, incluant le problème de la coordination inter-agents lui-même [Ferber, 1995 ; Adam, 2000 ; Mandiau et Grislin-Le Strugeon, 2001]. Un système multi-agents peut donc être défini comme une entité logicielle composée de plusieurs agents intelligents. Il peut être perçu selon différents points de vue (Cf. figure I.8) :

- si on considère le système multi-agents en globalité, c'est-à-dire le système et son environnement, on peut assimiler le système multi-agents à son environnement. On parle alors de « vue globale ».

- si on s'intéresse au système multi-agents lui-même, on peut identifier les différentes entités, le composant ainsi que les relations sociales (« interactions ») entre les agents. On parle alors de « vue sociale ».

- si on étudie un agent de plus près, son architecture, les interactions entre ses différents modules internes éventuellement constitutifs. On parle alors de « vue locale ».

Dans un système multi-agents il doit exister au moins un élément commun entre les différents agents. Celui-ci peut être l'objectif visé, ou le langage de communication, mais le minimum doit être un environnement commun. Les interactions entre agents au sein d'un système multi-agents peuvent être de type coopératif (les agents coopèrent entre eux afin d'atteindre un objectif commun), conflictuel (les agents poursuivent des buts différents qui peuvent être conflictuels) ou d'un type hybride entre les deux [Mandiau et Grislin-Le Strugeon, 2001 ; Chaib-Draa *et al.*, 2001].

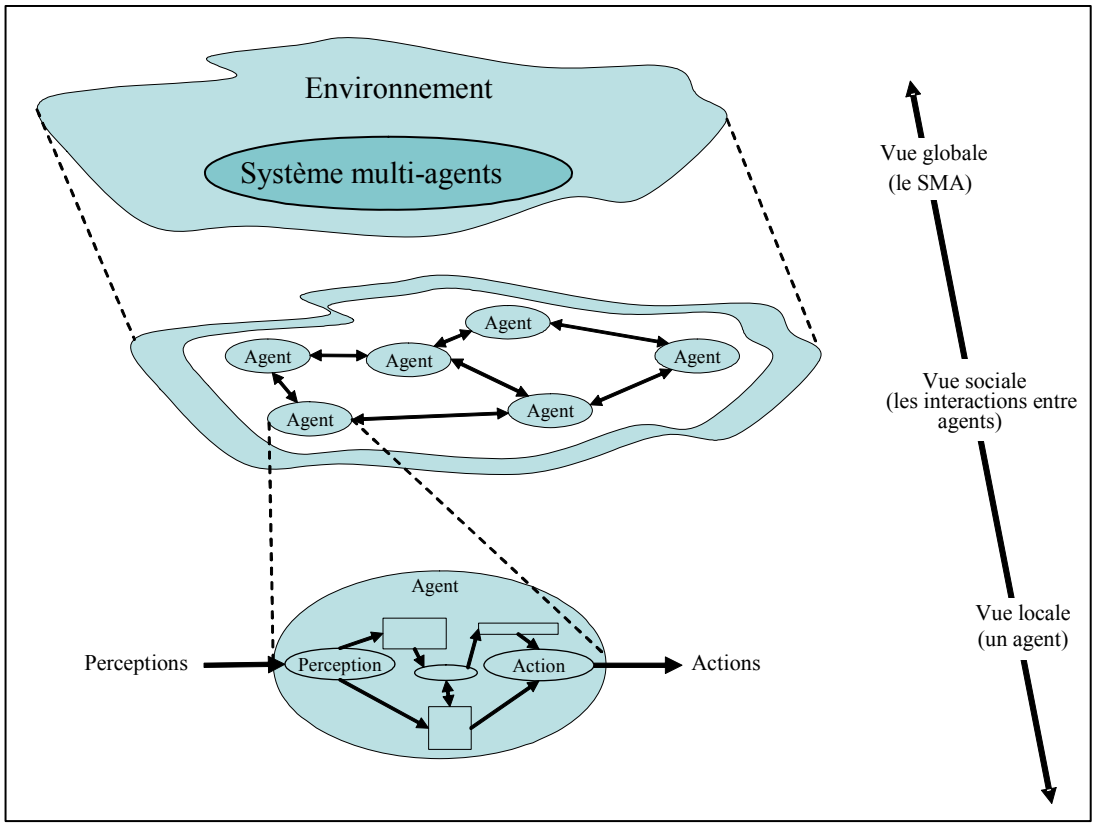



**Figure I.8.** Système multi-agents selon différents niveaux de détail [Mandiau et Grislin-Le Strugeon, 2001]

La question qui se pose souvent lorsque l'on parle d'un agent est : un agent n'est-il pas un objet au sens objet informatique ? La section suivante discute cette question et tente d'éclaircir la différence entre objet et agent.

## III.1.3. Agent et objet

La recherche sur les méthodes orientées objet a beaucoup contribué au développement de l'approche orientée agent. En effet, la programmation orientée agent (POA) peut être considérée comme une spécialisation du paradigme de la programmation orientée objet (POO) [Tarpin-Bernard, 1997 ; Coutaz et Nigay, 2001].

La différence principale entre agent et objet se situe principalement dans l'action [Mandiau et Grislin-Le Strugeon, 2001]. Un objet ne peut agir de lui même, c'est l'appel de l'une de ses méthodes qui le pousse à accomplir une tâche quelconque ; alors que l'agent est considéré comme une entité autonome capable d'agir sans être sollicité en fonction du changement de contexte de son environnement.

Avec l'évolution des théories concernant les agents, on peut également souligner d'autres différences entre objet et agent [Adina *et al.*, 2002] :

- les agents ont leurs propres buts et ils agissent d'une manière pro-active pour atteindre leurs buts (par exemple, ils saisissent des opportunités) alors que les objets ne le font pas ;

- les agents sont capables d'un comportement social : ils peuvent s'engager dans des interactions complexes, par exemple coopération, compétition, négociation, avec d'autres agents ; ce n'est pas le cas des objets ;

- un système multi-agents est, normalement, un système dans lequel les agents correspondent à des chemins d'exécution séparés ; chaque agent a son propre chemin d'exécution alors que les objets, à part les objets concurrents [Ferber, 1991], ne présentent pas cette caractéristique.

Comparé à un objet, un agent se positionne à un niveau d'abstraction plus élevé étant donné qu'il comporte un certain degré de réflexion au sens de l'intelligence artificielle. Comme l'explique [Franklin et Graesser, 1996] en passant en revue un ensemble de définitions, un agent n'est ni un programme ni un objet. [Iglesias *et al.*, 1999] considère un agent comme étant un objet actif ou un objet avec un « statut-mental ».

Il convient cependant de souligner que la différence fondamentale entre un objet et un agent est visible lors d'interactions entre deux entités. En effet, l'objet traite toujours les messages reçus alors qu'un agent filtre les messages qu'il reçoit en fonction de son propre intérêt. L'agent est considéré comme une entité informatique capable de « fonctionner » d'une façon autonome alors qu'un objet est une entité caractérisée par sa structure et un mécanisme d'exécution dont le fonctionnement est à la charge du concepteur [Shoham, 1993 ; Jennings *et al.*, 1998 ; Florez-Mendez, 1999].

Après avoir défini la notion d'agent, de système multi-agents et la différence entre agent et objet, nous présentons dans ce qui suit les principaux modèles d'architectures réparties. Ces architectures se basent essentiellement sur la notion d'agents qui communiquent entre eux.

## III.2. Le modèle MVC

Le modèle *MVC* (Modèle, Vue, Contrôleur) [Goldberg, 1983] a été créé pour structurer les applications écrites en langage *SmallTalk*. Ce modèle est donc intimement lié à ce langage. Il est constitué de trois composants relativement indépendants (Cf. figure I.9.a) :

- **Le Modèle** : définit les fonctionnalités propres au domaine et représente le noyau fonctionnel de l'agent. Le M*odèle* notifie les V*ues* qui lui sont associées à chaque fois que son état se trouve modifié par l'application ou par ses contrôleurs ;

- **La Vue** : assure une représentation du M*odèle* perceptible par l'utilisateur. Elle est en général constituée d'objets graphiques. Elle maintient la cohérence entre le *Modèle* et l'affichage présenté à l'utilisateur ; à chaque changement de l'état interne du *Modèle*, la V*ue* est notifiée et est mise à jour ;

- **Le Contrôleur** : reçoit et interprète les événements utilisateur en les répercutant sur le *Modèle*. Il joue le rôle d'interpréteur des actions de l'utilisateur tout en répercutant leurs effets sur le *Modèle*.

La communication entre les différents composants de ce modèle se fait par envoi de messages, conformément à la philosophie *Smalltalk*. Ceci assure une séparation entre la *Vue*, le *Contrôleur* et le *Modèle*. Un objet *Vue* peut être lui-même décomposé en plusieurs objets, selon une hiérarchie visible en figure I.9.b, où chaque sous-vue est elle même une *Vue*.

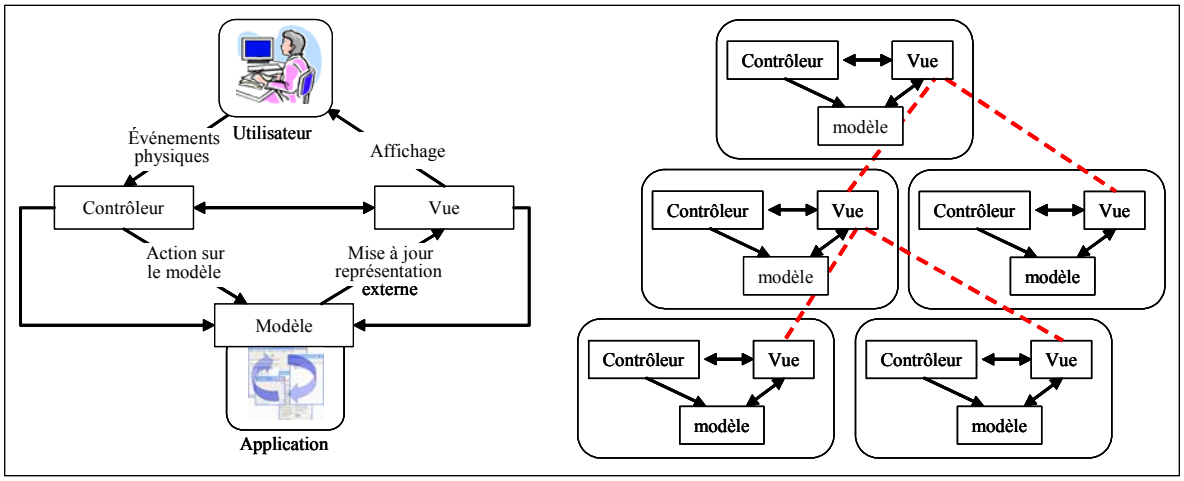



**Figure I.9.a.** Modèle MVC

**Figure I.9.b.** Exemple de hiérarchie de vues dans MVC

La communication entre les différents composants de ce modèle se fait par envoi de messages, conformément à la philosophie *Smalltalk*. Ceci assure une séparation entre la *Vue*, le *Contrôleur* et le *Modèle*. Un objet *Vue* peut être lui-même décomposé en plusieurs objets, selon une hiérarchie visible en figure I.9.b, où chaque sous-vue est elle même une *Vue*.

Afin de mieux comprendre le fonctionnement de ce modèle, nous reprenons l'exemple d'une simple application d'éditeur de dessin (Cf. figure I.10) cité dans [Van-Eylen et Hiraclides, 1996]. Cette application permet à l'utilisateur de tracer des formes graphiques dans un espace de dessin. L'éditeur est composé de deux éléments : la palette permet de sélectionner l'objet graphique à créer et l'espace de dessin sur lequel seront positionnés les objets créés. Nous reprenons ce même exemple plus loin (Cf. III.3 et III.4) pour comprendre le fonctionnement des modèles *PAC* et *AMF*.

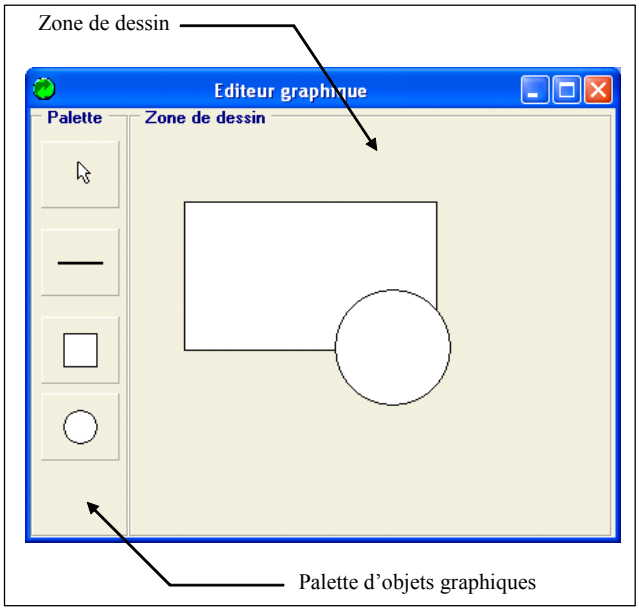



**Figure I.10.** Interface utilisateur d'un éditeur de dessin

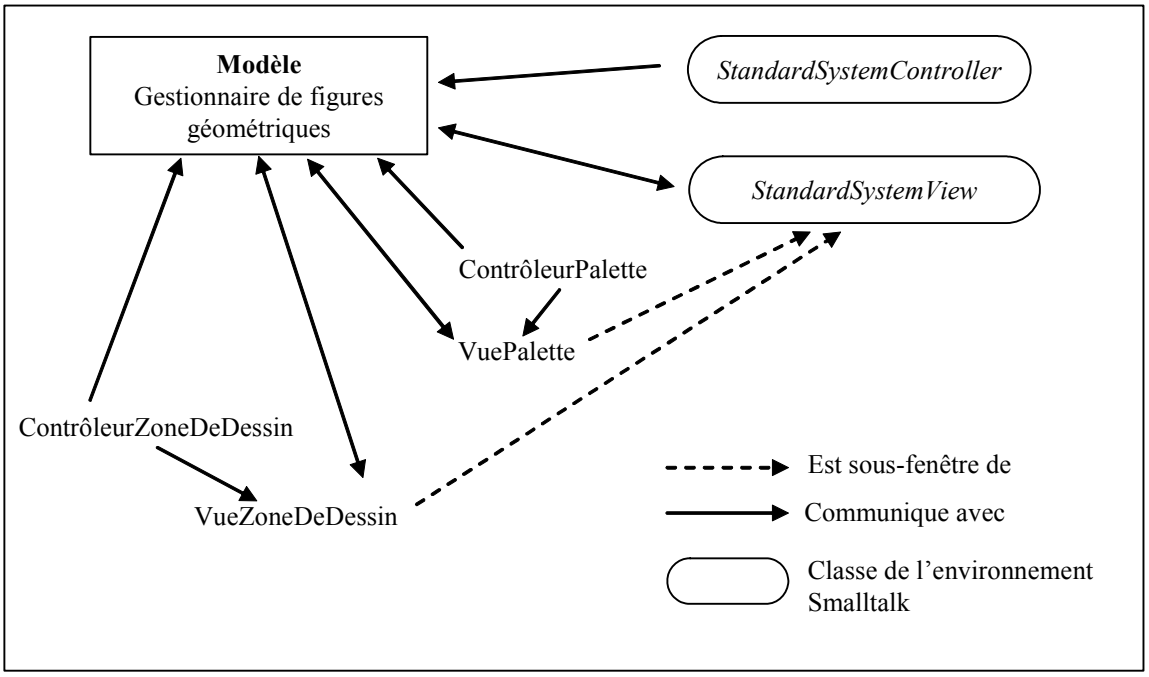



**Figure I.11.** Modélisation d'un éditeur de dessin selon l'architecture MVC [Van-Eylen et Hiraclides, 1996]

La figure I.11 propose une réalisation possible de l'éditeur de dessin selon l'architecture *MVC*. Les classes en *italique* sont celles réutilisées de l'environnement *Smaltalk*, les autres classes sont développées pour l'éditeur. Trois instances du triplet *MVC* sont présentes : une pour gérer la zone de dessin, une pour la palette et une standard qui correspond à la fenêtre principale de l'application de dessin. A titre d'exemple nous illustrons la dynamique de création d'un nouveau rectangle :

- l'utilisateur pointe le rectangle présent dans la palette. Cet événement est reconnu par le *ContrôleurPalette* qui indique à son *Modèle* l'intention de créer un rectangle ;
- le *Modèle* se met alors dans l'état création de rectangle, ce qui est communiqué à la *VueZoneDeDessin* et retransmis au *ContrôleurZoneDeDessin* ;
- puis, l'utilisateur positionne le pointeur sur la *ZoneDeDessinVue* dont le contrôleur associé recevant les événements de déplacement de la souris lui demande d'afficher le fantôme du rectangle à créer ; ceci jusqu'à un clic souris indiquant la position souhaitée ;
- alors le *Contrôleur* de la zone de dessin indique au *Modèle* de créer un objet rectangle ;
- le nouveau rectangle est créé dans le *Modèle* et la *Vue* de la zone de dessin doit se remettre à jour pour refléter ce changement d'état du modèle.

Selon [Baudel, 1995] le modèle *MVC* présente quelques inconvénients : il ne prend pas en compte les contraintes liant les objets entre eux et la structure globale de l'interaction. Il est donc relativement mal adapté aux interfaces à base de menus et d'écrans de saisie, ou lorsque le modèle de dialogue est complexe. Plus important, il ne présente pas de vue d'une

application sous forme de niveaux d'abstraction : fonctionnalités de haut niveau et aspects implantatoires sont présentés sur le même plan pour chaque objet considéré.

La mise en œuvre d'une application respectant le modèle *MVC* reste quand même une tâche difficile qui nécessite un long apprentissage de la programmation en *Smalltalk*. De ce fait, cette architecture s'est détachée du langage d'implémentation et l'on trouve à présent des boîtes à outils en *C*, *C++* ou même en *JAVA* conçues autour de ce modèle. A titre d'exemple, on peut citer les travaux de [Martin *et al.*, 2005] qui utilise le modèle MVC pour le développement d'un bureau collaboratif en 3D implémenté en Javascript et C++.

Le modèle *MVC* a été largement répandu pour le développement des applications pour le Web, et particulièrement les Servlets[6]. Une variante de ce modèle est le modèle *MVC2* issu du projet Struts[7] développé par la communauté Jakarta et soutenu par la fondation Apache.

*MVC* nécessite l'écriture d'une multitude de Servlets, qui sont autant de points d'entrée dans l'application. Pour pallier cet inconvénient, des frameworks ont été développés. Ces frameworks, qui sont composés d'une seule servlet (un seul contrôleur), sont regroupés sous l'étiquette « *Model 2* » encore appelé « *MVC2* » [Girard et Ribette, 2001].

## III.3. Le modèle PAC

Le modèle *PAC* (*Présentation*, *Abstraction*, *Contrôle*) a été introduit par [Coutaz, 1987] ; il structure l'application interactive de manière récursive sous forme d'une hiérarchie d'agents. Chaque entité est, comme dans le modèle de *Seeheim*, décomposée en trois composants, appelés facette. Les trois facettes (Cf. figure I.12) d'un agent *PAC* sont :

- **La facette Présentation** : visualise l'état interne de l'agent, c'est-à-dire son comportement en entrée comme en sortie par des objets interactifs de présentation. Elle détermine le comportement perceptible de l'agent pour l'utilisateur tout en interprétant ses actions. Selon le niveau d'abstraction dans lequel on se place, on s'aperçoit que l'utilisateur est en interaction avec le composant présentation d'un agent particulier ; la présentation d'une application est l'union des présentations de touts les agents qui la constituent. On peut noter que le composant P*résentation* du modèle *PAC* peut être vu comme l'union des deux composants V*ue* et C*ontrôle* du modèle *MVC* ;

- **La facette Abstraction** : encapsule les compétences sémantiques de l'agent et implémente les fonctions propres au domaine de l'application. Elle peut être considérée comme le noyau fonctionnel de l'agent. *L'Abstraction* du modèle *PAC* peut être comparée au composant *Modèle* de *MVC* ;

- **La facette Contrôle** : joue le rôle de connecteur entre les deux facettes *Abstraction* et *Présentation*. Elle maintient, d'une part la consistance entre les objets présentés à l'utilisateur et leurs correspondants dans l'application, et d'autre part la gestion de la communication avec les autres agents.

[6] Les servlets sont donc des applications Java fonctionnant du côté serveur.
[7] http://struts.apache.org/

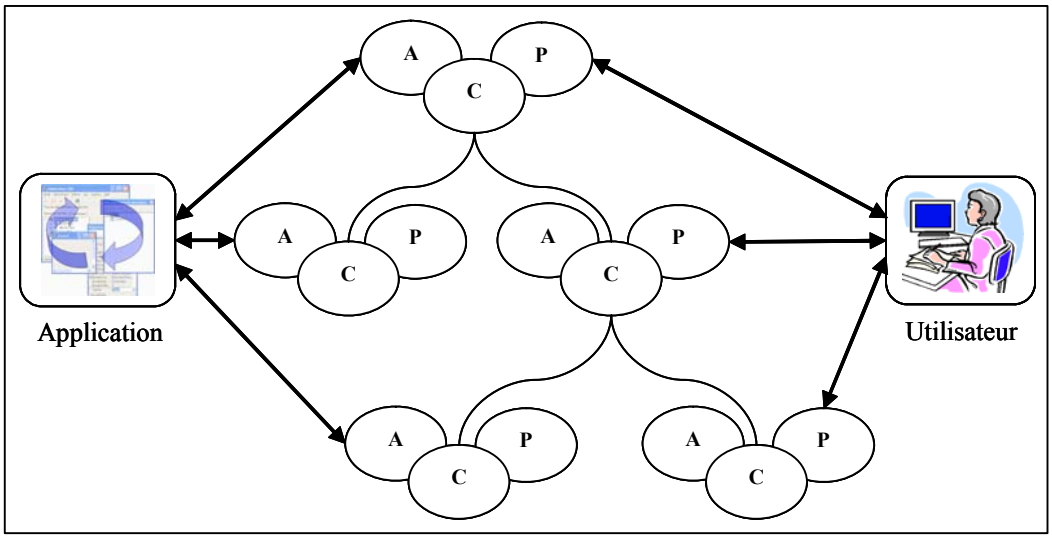



**Figure I.12.** Architecture PAC

Afin de mieux comprendre l'architecture *PAC*, nous reprenons l'exemple de l'éditeur de dessin utilisé précédemment pour l'illustration du modèle *MVC*. La modélisation de l'éditeur de dessin avec le modèle *PAC* est visible sur la figure I.13. Prenons l'hypothèse où nous avons décomposé le système en trois agents[8] : agent *EditeurGraphique*, agent *Palette* et agent *ZoneDeDessin*.

La dynamique de création d'un rectangle se déroule alors comme suit :

- l'utilisateur sélectionne sur la palette graphique l'objet rectangle ;
- l'agent Palette indique alors à l'agent *EditeurGraphique* l'intention qu'a l'utilisateur de créer un nouveau rectangle ;
- l'agent *EditeurGraphique* en informe l'agent *ZoneDeDessin* qui peut alors passer en mode « création de rectangle » ;
- quand il perçoit le pointeur entrer dans sa surface d'affichage, l'agent *ZoneDeDessin* réalise un feedback présentant un fantôme du rectangle en cours de création et ceci jusqu'à ce que l'utilisateur l'ait bien positionné ;
- à ce moment, la création de l'objet rectangle peut avoir lieu car toutes les informations nécessaires (sa position et la taille dans la zone de dessin) ont été fournies par l'utilisateur.

Ce court exemple tiré de [Van-Eylen et Hiraclides, 1996] montre la répartition des responsabilités des trois agents permettant la réalisation de la tâche de création d'un rectangle.

Comme le fait remarquer [Nigay, 1994], les liens entre les agents ne sont pas nécessairement des canaux de communication. Ils peuvent traduire aussi des relations de décomposition, c'est-à-dire des opérations de conception logicielle que sont l'encapsulation et l'affinement.

[8] Il est possible d'avoir une autre vision de l'application et par conséquence une décomposition différente. On peut par exemple remplacer l'agent Palette par un agent Menu proposant le même choix à l'utilisateur.

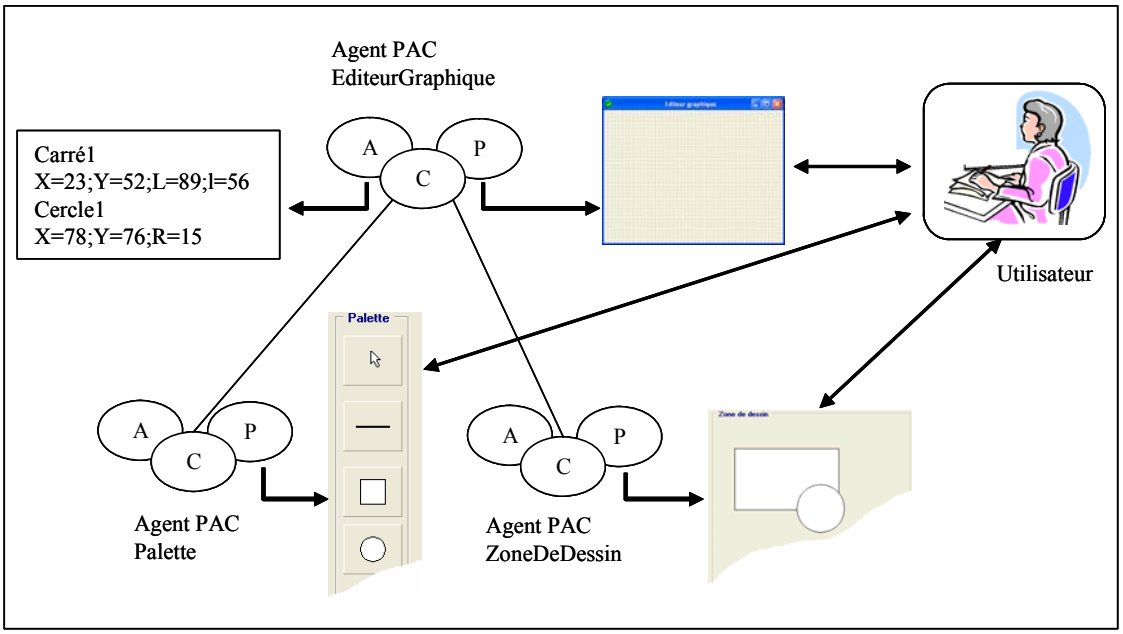



**Figure I.13.** Modélisation d'un éditeur de dessin selon l'architecture PAC

Le modèle *PAC* n'étant pas lié à un langage de programmation donné ni à une implémentation précise, ces agents peuvent être constitués par un objet unique ou par trois objets différents, un par facette. Cependant, [Coutaz, 1990] recommande la décomposition de chaque agent *PAC* en modules suivants : (1) autant de modules que de *Présentations* (plusieurs facettes peuvent être envisageables pour un seul composant présentation), (2) un module pour le *Contrôle*, (3) un module pour *l'Abstraction*.

Cette décomposition permet de respecter des critères de qualité telles que la portabilité, l'évolutivité et la réutilisabilité :

- **La portabilité** est assurée par la nette séparation entre la partie interface et le noyau fonctionnel ; en effet la *Présentation* isole clairement les parties du logiciel liées à la boîte à outils. De ce fait, en cas de changement de boîte à outils, la facette *Présentation* sera à récrire et le *Contrôle* risque d'être changé lui aussi, mais la partie noyau fonctionnel reste inchangée ;

- **L'évolutivité** de l'application est envisageable grâce à sa structuration en plusieurs niveaux d'abstraction ainsi qu'à la simplicité du graphe d'appel entre les agents. En effet, de nouveaux agents peuvent être facilement intégrés à l'application à différents niveaux d'abstraction. Une réorganisation des agents peut être nécessaire, dans ce cas, l'évolution est facile puisque le graphe d'appel entre les agents est simple ;

- **La réutilisabilité** est généralement liée aux deux composants *d'Abstraction* et de *présentation*. En effet, l'implémentation des différentes parties de l'agent dans des modules différents rend ces composants réutilisables. Le composant *Contrôle* assurera la transparence de cette portabilité. En cas de changement de boîte à outils de l'application, le contrôleur permet d'adapter les formats de la nouvelle application à ceux du composant réutilisable ;

Il est à noter que plusieurs variantes du modèle *PAC* ont été proposées. On peut en citer à titre d'exemple :

- **le modèle PAC+3** : [Calvary *et al.*, 1996] est un modèle d'architecture générique pour les systèmes multi-utilisateurs. Il concilie, d'une part la décomposition fonctionnelle du trèfle des collecticiels[8], et d'autre part l'indépendance entre les trois facettes du modèle *PAC*.

- **le modèle du *trèfle*** : [Sablier, 1995] est inspiré du modèle conceptuel d'un collecticiel proposé par [Ellis, 1994]. Il définit trois espaces qu'une application groupware[9] peut soutenir à savoir, la *production*, la *communication* et la *coordination* (Cf. figure I.14). L'*espace de production* désigne les objets qui résultent d'une activité de groupe, par exemple un livre, une oeuvre de cinéma, un logiciel, etc. L'*espace de coordination* définit les acteurs, les activités et les tâches, les acteurs responsables des tâches et des activités. L'espace de communication offre aux acteurs la possibilité d'échanger de l'information. Le contenu sémantique de cette information concerne les acteurs communicants.

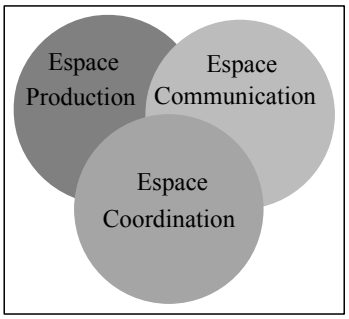



**Figure I.14.** Modèle du *trèfle* [Sablier, 1995]

- **le modèle *Clover*** : [Laurillau, 2002] est dérivé du méta-modèle d'architecture générique pour les collecticiels de [Dewan, 1999] en appliquant les cinq niveaux d'abstraction du modèle de référence Arch [Bass *et al.*, 1991]. Néanmoins le modèle *Clover* compte six niveaux, c'est-à-dire un niveau de plus que le modèle *Arch*. En effet, le *Noyau Fonctionnel* est divisé en deux niveaux : le *Trèfle Fonctionnel privé* (gère l'ensemble des objets du domaine propres à un seul utilisateur) et le *Trèfle Fonctionnel public* (gère l'ensemble des objets du domaine communs à l'ensemble des utilisateurs).

- **le modèle *REPAC*** : [Dassi *et al.*, 2003] qui reprend les mêmes composants qu'un agent *PAC* (*Abstraction*, *Contrôle*, *Présentation*). L'originalité de ce modèle est de permettre le raffinement de la facette présentation en un agent *PAC*. Ce raffinement permet une meilleure structuration de la facette présentation.

- **le modèle *Compact*** : (COntext Mouldable PAC for Plasticity) [Calvary *et al.*, 2005] est une déclinaison du modèle *PAC* pour la plasticité[10]. Ce modèle tranche chaque facette en deux, isolant ainsi en chacune d'elles une partie logique et une

[9] Un collecticiel ou encore Groupware est un logiciel qui permet aux membres d'un groupe de travailler ensemble sans forcément être présents en même temps au même endroit [Tarpin-Bernard, 1997].
[10] La plasticité d'une Interface Homme-Machine est sa capacité à s'adapter à son contexte d'usage (utilisateur, plate-forme, environnement) dans le respect de son utilisabilité [Calvary et al., 2006].

partie physique. La partie logique gère l'état de la facette indépendamment de son incarnation, et la partie physique représente l'incarnation courante de la facette.

La décomposition en facettes, introduite par le modèle *PAC*, n'a pas seulement que des avantages. En effet, quand la taille de l'application devient importante, la communication entre les agents risque de ralentir l'application. Il est à noter que ce problème est d'autant moins sensible quand la machine est puissante et les contraintes temps réel sont souples. Pour pallier ce problème, la majorité des développeurs n'hésitent pas à court-circuiter des niveaux lors de la transmission d'informations entre deux niveaux hiérarchiquement éloignés. Cette pratique risque de créer des problèmes quant à la maintenance de l'application ainsi que son évolution.

## III.4. Le modèle AMF

Le modèle *AMF* (modèle multi-agents multi-facettes) a été développé à l'Ecole Centrale de Lyon par Kamel Ouadou [Ouadou, 1994]. Ce modèle a été complété et étendu au travail collaboratif par [Tarpin-Bernard, 1997 ; Tarpin-Bernard et David, 1999]. Comme l'explique [Tarpin-Bernard et David, 1999], le modèle *AMF* est né d'une double constatation :

La décomposition *Abstraction* / *Présentation* est généralement insuffisante pour les applications complexes. Des fonctionnalités relevant de thématiques distinctes se retrouvent mélangées dans des composants trop macroscopiques.

Les formalismes de modélisation des composants *Contrôle* sont rares et généralement limités. Pourtant, ces composants constituent la clé de voûte des modèles d'architecture et il est indispensable de fournir un outil de modélisation performant.

Pour répondre à la première limite, le modèle *AMF* reprend les mêmes facettes du modèle *PAC*, c'est-à-dire la P*résentation*, le C*ontrôle* et *l'Abstraction*, tout en proposant une décomposition plus fine des agents. Rappelons qu'une facette est définie comme un composant contenant l'ensemble des données et traitements relevant d'une même thématique fonctionnelle. Comme dans le modèle *PAC*, le nombre des facettes n'est pas limité et aucune facette n'est obligatoire. On peut avoir de nouvelles facettes provenant d'une décomposition fine de la facette contrôle, de la duplication des facettes classiques (par exemple, plusieurs facettes présentation) ou pour modéliser de nouveaux aspects des agents (par exemple, facettes « Aide » ou « Erreur », modélisation de l'utilisateur, etc.). Chaque facette se voit donc attribuer un rôle précis, ce qui conduit à une meilleure modélisation [Chalon, 2004].

Comme dans *PAC*, les agents sont organisés hiérarchiquement selon des règles de composition. On retrouve au plus haut niveau un agent racine (appelé généralement « application »). Cette décomposition peut s'exprimer sous forme d'emboîtement (Cf. figure I.15.a) ou sous forme de graphe (Cf. figure I.15.b) lorsque les schémas deviennent trop complexes.

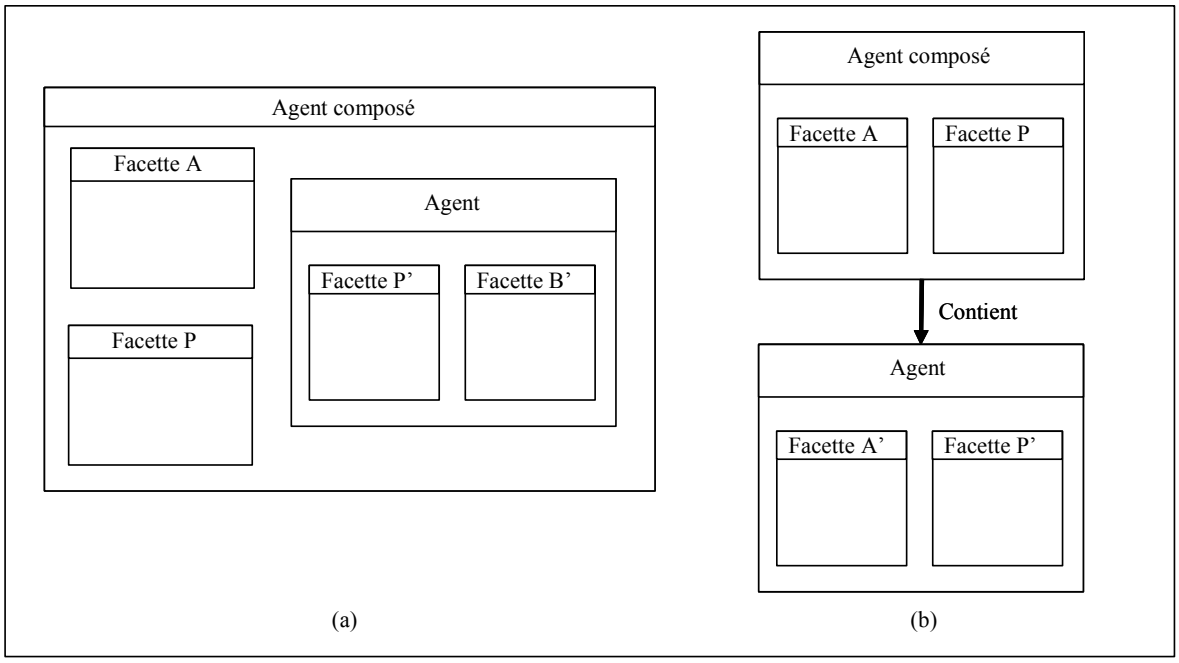



**Figure I.15.** Représentation emboîtée (a) et relationnelle (b) des agents AMF [Tarpin-Bernard, 1997]

Pour répondre à la deuxième limite, le modèle *AMF* intègre un formalisme complet pour la modélisation de la facette contrôle ainsi que les échanges entre les facettes de chaque agent et la communication entre les agents. Ce formalisme s'appuie sur des éléments situés à deux niveaux :

- au niveau des facettes, **les ports de communication** (Cf. figure I.16.b) définissent les services proposés (port de sortie) et les services utilisés (port d'entrée). Les ports d'entrée/sortie définissent des services dont l'activation se conclue par l'invocation d'un autre service distant. Par exemple, L'activation d'un port de sortie d'une facette A conduit à l'émission d'un message qui sera reçu par un port d'entrée d'une autre facette B.

- au niveau des agents, **les composants administrateur de contrôle** (Cf. figure I.16.a) gèrent les liens entre les ports de communication des facettes. Un administrateur de contrôle joue trois rôles : (1) un rôle de connexion qui consiste à gérer les relations logiques pouvant exister entre les ports de communication qui lui sont attachés, (2) un rôle comportemental qui exprime les règles d'activation de l'administrateur, (3) un rôle de traduction qui consiste à transformer les messages émis par les ports sources en messages compréhensibles par les ports cibles. Par exemple, les *administrateurs conjonctif* et *disjonctif* (visibles à la figure I.16.b) offrent le moyen de décrire respectivement une relation de conjonction sans contrainte d'ordre, et une disjonction équivalentes à : si A1 et A2 et ... An alors B, et si A1 ou A2 ou ... An alors B.

On peut noter que les facettes *Contrôle* sont différentes des autres facettes car elles n'ont pas véritablement d'existence propre et ne présentent pas de ports de communication. En effet, une facette contrôle est constituée des *administrateurs de contrôle* qui sont gérés par l'agent lui-même.

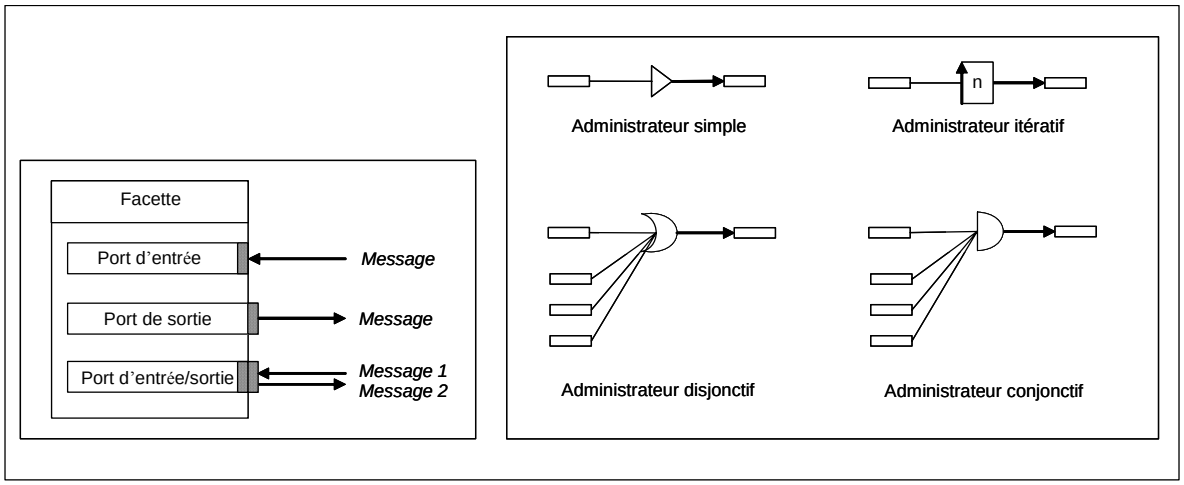



**Figure I.16.a.** Exemple de représentation d'administrateurs de contrôle [Tarpin-Bernard, 1997]

**Figure I.16.b.** Représentation des ports de communication d'une facette AMF [Tarpin- Bernard, 1997]

Afin de mieux comprendre l'architecture *AMF*, nous reprenons l'exemple de l'éditeur de dessin utilisé précédemment pour l'illustration du modèle *MVC* et *PAC*. La modélisation de l'éditeur de dessin avec le modèle *AMF* est visible sur la figure I.17. Nous avons décomposé le système en trois agents : Un agent EditeurGraphique, un agent Palette et un agent ZoneDe Dessin.

La dynamique de création d'un rectangle se déroule alors comme suit :

- l'utilisateur sélectionne via la *Présentation* de l'agent Palette l'objet rectangle ;
- l'agent Palette mémorise le type de l'objet sélectionné et indique alors à l'agent EditeurGraphique l'intention qu'a l'utilisateur de créer un nouveau rectangle ;
- dés que l'utilisateur clique sur la zone de dessin, *l'Abstraction* de l'agent ZoneDe Dessin calcule la position du clic et fait appel via un administrateur de conjonction au service de l'agent Palette afin de connaître l'objet sélectionné.
- l'agent ZoneDeDessin réalise alors un feedback présentant un fantôme du rectangle en cours de création et ceci jusqu'à ce que l'utilisateur relâche le bouton de la souris (clic-up) ;
- une fois que l'utilisateur relâche le bouton de la souris, l'objet est affiché. A ce moment, la création de l'objet rectangle peut avoir lieu. La facette *Présentation* de l'agent ZoneDeDessin envoie alors toutes les informations à l'abstraction de l'agent Editeur graphique afin de mémoriser l'objet rectangle.

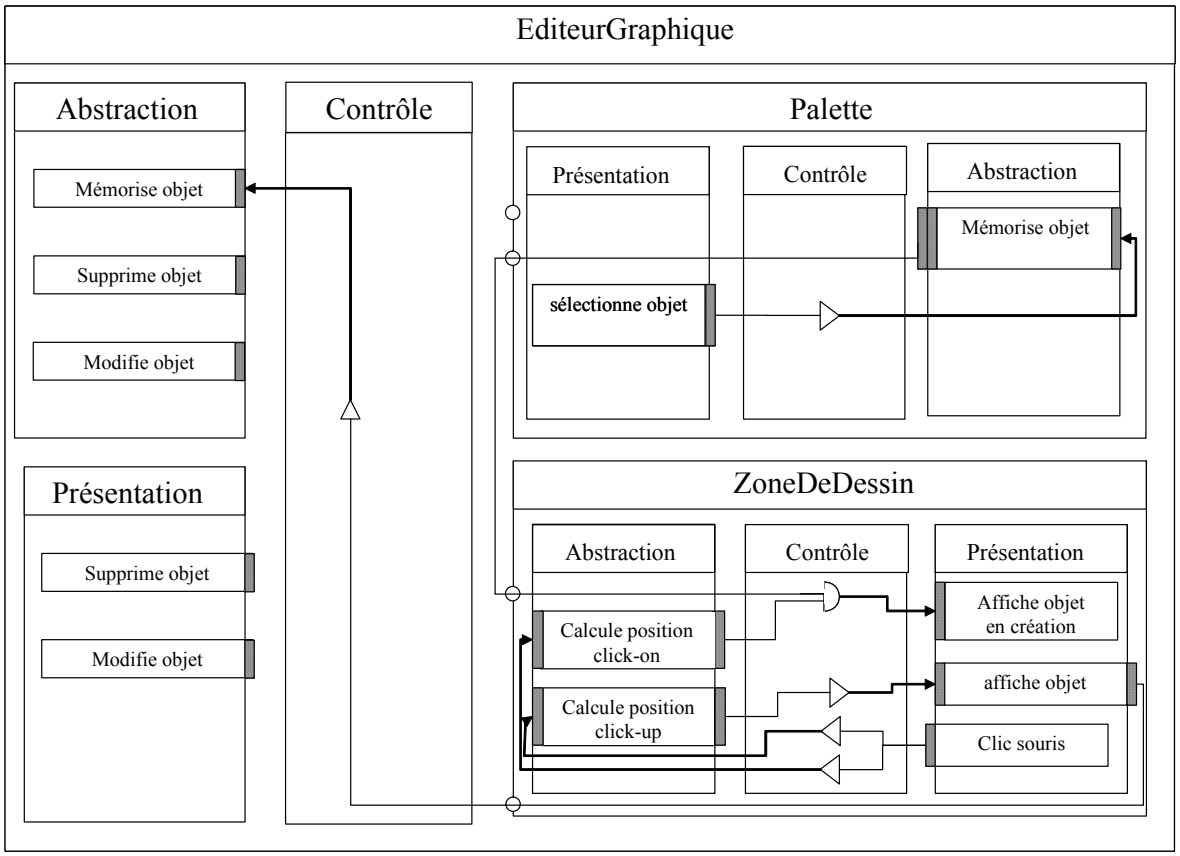



**Figure I.17.** Modélisation d'un éditeur de dessin selon l'architecture AMF

Afin de conserver le caractère pédagogique de cet exemple, nous avons simplement modélisé la création d'un objet. La suppression et la modification d'un objet sont des tâches un peu plus complexes. Le lecteur intéressé peut trouver dans [Tarpin-Bernard et David, 1999] un exemple détaillé de modélisation d'un logiciel interactif de gestion d'agenda.

## III.5. Conclusion sur les architectures réparties

Comme nous l'avons vu, les modèles d'architectures réparties permettent la représentation des différents modules d'un système interactif ainsi que la communication entre eux. L'avantage de cette représentation est de posséder un niveau de granularité beaucoup plus fin que les modèles d'architecture centralisés des systèmes interactifs.

En effet, si les architectures centralisées séparent un système interactif en trois différentes fonctions (application, contrôleur, présentation), les architectures réparties proposent de les regrouper au sein d'une même entité appelée « agent ».

Le reproche principal que l'on peut adresser à ce type d'architectures est de ne pas établir clairement les domaines d'application auxquels elles s'adressent et de ne pas identifier de manière précise le rôle et le nombre des agents qui doivent composer une application interactive [Coutaz 1990].

D'autre part, la décomposition en plusieurs entités autonomes et coopératives, permet d'accélérer en partie le retour de l'application à l'utilisateur. Dans la mesure où nous nous intéressons aux systèmes interactifs dédiés à la supervision, cet avantage peut être très intéressant pour la communication entre l'utilisateur et le système. En effet, les systèmes de

supervision peuvent être hautement interactifs, ainsi un dialogue intensif entre l'utilisateur et l'application risque de ralentir le système.

Malgré cet avantage apporté par les modèles répartis, on s'aperçoit qu'ils présentent un inconvénient relatif. En effet, la perception de l'interface globale est difficile pour le concepteur d'une application de supervision (avec une interface constituée de plusieurs éléments) car elle est répartie entre les différents composants *Présentation* de chaque agent. Or les architectures centralisées apportent une solution à ce problème puisqu'elles proposent un seul composant *Présentation* qui regroupe touts les éléments de l'interface.

L'idée est alors de combiner les deux types d'architectures afin de bénéficier des avantages de l'une et de l'autre. Nous parlerons alors d'architecture hybride. En effet, la combinaison de ces deux architectures n'est pas une idée récente. Plusieurs architectures hybrides on été proposées. Nous en présenterons les plus connues dans la section suivante.

# IV. Architectures hybrides

Comme nous l'avons vu dans les deux sections précédentes (Cf. II, III), les modèles d'architectures reparties définissent, à niveau de granularité très faible, le fonctionnement des différents composants du système interactif. Cette décomposition s'est faite aux dépens de la distinction des composants identifiés à un niveau macroscopique par les modèles centralisés [Depaulis, 2002]. Les architectures hybrides tentent de tirer profit de ces deux types d'architectures en reprenant l'avantage de chacun d'eux.

Les modèles hybrides ont vu le jour au début des années 90 dans le cadre des travaux de Laurence Nigay autour du modèle PAC-Amodeus, ceux de Kamel Ouadou concernant le modèle $H^4$ et les travaux de [Fekete, 1996] pour le modèle MultiCouche. Ces trois modèles d'architecture sont maintenant présentés.

## IV.1. Le modèle PAC-Amodeus

Le modèle *PAC-Amodeus*[11] [Nigay, 1994 ; Nigay et Coutaz, 1991] résulte d'une hybridation entre le modèle *PAC* et le modèle *ARCH* (Cf. figure I.5). En effet, cette architecture reprend les composants du modèle *ARCH* dont il affine le contrôleur de dialogue en terme d'agents *PAC*. Cette décomposition est justifiée par les deux aspects suivants [Coutaz et Nigay, 2001] :

- **Le découpage** met en valeur les fonctions essentielles d'un logiciel d'interaction et notamment le *contrôleur de dialogue*, point sensible de la conception logicielle. Il permet également, via les deux adaptateurs (*adaptateur du domaine*, et *adaptateur de présentation*), de définir des points de résolution en cas d'incompatibilité avec le noyau fonctionnel ainsi que les lieux de réutilisation de code (par exemple, il n'est pas nécessaire de récrire ou de changer l'implémentation d'une boîte à outil si elle assure la bonne interaction avec l'utilisateur).

[11] PAC-Amodeus : PAC pour Présentation, Abstraction, Contrôle (Cf. § III.3) et Amodeus pour le projet européen Esprit BRA Amodeus.

- **La structuration** du *contrôleur de dialogue* en agents *PAC* rend explicites les interactions à plusieurs fils (interaction multimodale), fournit le bon niveau de granularité en vue de la mise en œuvre, permet via les facettes *Abstraction* de jouer sur l'allocation et la migration des services du noyau fonctionnel dans l'IHM.

L'objectif de *PAC-Amodeus* est de combiner les avantages du modèle *ARCH*, qui intègre des aspects de génie logiciel comme la modifiabilité et la portabilité, et du modèle *PAC*, qui permet de structurer efficacement le contrôleur de dialogue jusque là monolithique. Pour ce faire, Le modèle Pac-Amodeus est composé, de la même manière que *ARCH*, de cinq composants visibles en figure I.18 :

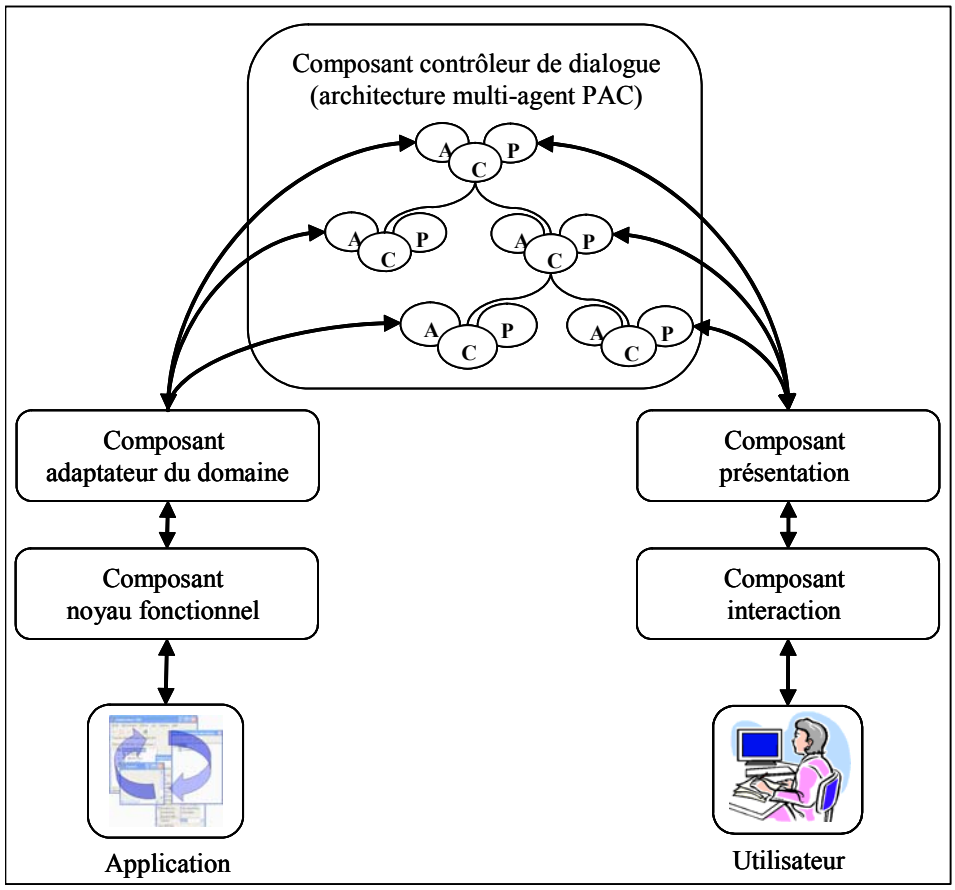



**Figure I.18.** Le modèle PAC-Amodeus [Nigay, 1994]

- **Le composant noyau fonctionnel** : réalise les concepts du domaine en ignorant la façon dont ces concepts sont rendus perceptibles à l'utilisateur. Il est la véritable sémantique de l'application. Il échange des concepts du domaine ou objet du domaine avec son voisin immédiat : l'*Adaptateur du domaine* ;

- **Le composant adaptateur de domaine :** assure la communication entre le noyau fonctionnel et l'*Abstraction* des agents *PAC* du C*ontrôleur de dialogue*. Ce composant garantit l'indépendance entre le noyau fonctionnel et le C*ontrôleur de dialogue* en gérant un ensemble d'objets conceptuels ;

- **Le composant contrôleur de dialogue** : est composé d'agents *PAC*. Il assure l'enchaînement des tâches à traiter ainsi que la transformation du formalisme pour

la communication entre le composant *Adaptateur du domaine* et le composant *Présentation*. D'autre part, il permet la mise en correspondance entre le monde des concepts et celui de l'interface. Par exemple, la modélisation d'un réel T (exprimant la température) du noyau fonctionnel au moyen d'une jauge J perceptible par l'utilisateur ;

- **la présentation** communique directement avec chaque agent *PAC* à travers son composant *Présentation*. Elle définit le comportement perceptible conceptuel du système en termes d'objets conceptuels de présentation. Un objet conceptuel de présentation est un interacteur abstrait, idéalisé, qui traduit un besoin communicationnel. Par exemple, un choix devient un menu et une alerte est traduite en une boîte de dialogue modale[12] ;

- **le Composant interaction** désigne la plate-forme d'accueil à la fois logicielle et matérielle. Il comprend le système de fenêtrage et ses boîtes à outils.

Le modèle *PAC-Amodeus* conserve les propriétés intrinsèques du modèle *ARCH* comme la portabilité et la décomposition fonctionnelle qu'il préconise. Grâce à la décomposition du *Contrôleur de dialogue* en agents *PAC*, il permet aussi de satisfaire les propriétés de qualité du logiciel que soit la modifiabilité et la réutilisabilité, ainsi que les propriétés d'utilisabilité comme le dialogue à fils multiples [Salber, 1995 ; Depaulis, 2002].

Le modèle *PAC-Amodeus* a été employé pour plusieurs applications telles que des plates-formes multimodales [Nigay et Coutaz 1995] ou encore des applications de simulation [Duval et Nigay, 1999]. Récemment, [Lachenal, et Coutaz, 2003] ont proposé un outil pour enseigner et comprendre PAC-Amodeus.

Il est à noter que plusieurs modèles d'architecture, pour les collecticiels, inspirés du modèle *PAC-Amodeus* ont vu le jour. A titre d'exemple :

- **le modèle CoPAC** : est une extension du modèle *PAC-Amodeus* aux systèmes multi-utilisateurs. Il augmente à la fois les agents *PAC* et le modèle *ARCH* [Salber, 1995]. Il introduit des nouveaux composants et connecteurs pour répondre aux demandes particulières des services de communication. En effet, ces services exigent de manipuler de nouveaux types de données, les flots de médias continus, pour lesquels les modèles multi-agents n'ont pas de solution explicite.

- **Le modèle PAC*** [Calvary *et al.*, 1997] : selon le même état d'esprit que CoPAC, le modèle PAC* est la version collaborative du modèle d'architecture hybride *PAC-Amodeus*. Ce modèle est dédié pour les systèmes multi-utilisateurs et plus particulièrement au CSCW (Computer Supported Collaborative Work). En effet, il effectue un découpage fonctionnel des agents PAC et l'affine selon les trois espaces fonctionnels du modèle du trèfle : production, communication et coordination (Cf. § III.3).

[12] Une boîte de dialogue modale est une boîte qui bloque l'accès au système tant que le message n'a pas été validé.

## IV.2. Le modèle MultiCouche

Le modèle *MultiCouche* a été proposé par Jean-Daniel Fekete en 1996 [Fekete, 1996]. Ce modèle est spécialisé pour la gestion des interactions dans les applications interactives de type MacDraw[13]. Il repose sur la collaboration de trois types d'objets : la pile, les couches et les outils. L'auteur part du principe que les objets graphiques sont de natures très différentes ; il existe en effet des objets internes (fond d'écran) et des objets sélectionnables et manipulables (déplacement, rotation). Ainsi, plutôt que de gérer les objets graphiques au sein d'un même niveau, il propose d'utiliser autant de niveaux nécessaires pour séparer les éléments graphiques qui suivent un comportement spécifique.

Le modèle *MultiCouche* est donc basé sur un ensemble de couches se projetant sur une surface de présentation commune (Cf. figure I.19). Ainsi, pour être visualisées, les couches sont placées dans une pile qui les compose pour qu'elles s'affichent sur une surface virtuelle. Chaque couche est responsable d'un aspect de l'interaction homme-machine. Par exemple, quand une couche gère les contraintes telles que « maintenir les coordonnées du curseur sur une grille », d'autres couches peuvent s'occuper de la gestion de sélection, de la visualisation des objets graphiques manipulés ou encore de la gestion du fond de l'image. La projection de l'ensemble de ces couches sur une même surface permet d'obtenir l'image finale destinée à l'utilisateur.

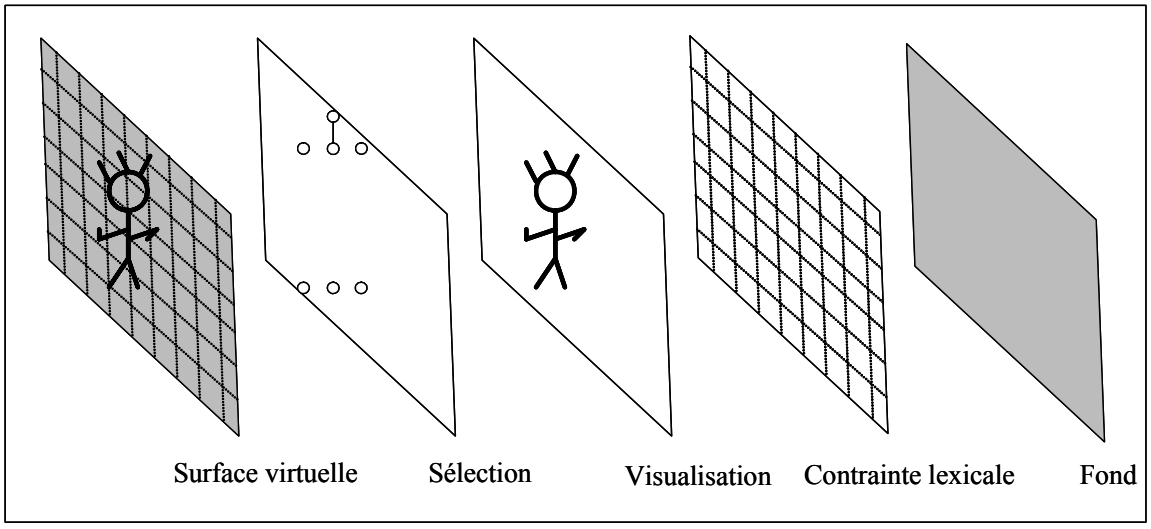



**Figure I.19.** Composition des couches graphiques dans l'architecture MultiCouche

La figure I.19 montre l'ordre dans lequel les couches sont classées dans une pile pour obtenir une image finale. Les couches s'affichent du fond vers le premier plan. La première couche surface virtuelle visualise l'image finale proposée à l'utilisateur ; la deuxième gère la sélection ; la troisième s'occupe de l'affichage des objets graphiques ; la quatrième affiche la grille ; et la dernière affiche le fond d'écran.

Ces différentes couches communiquent entre elles par des événements. Chaque événement produit par l'utilisateur parcourt toutes les couches, les unes après les autres (du premier plan vers le fond, c'est-à-dire de celui qui est proche de l'utilisateur jusqu'au dernier plan), jusqu'à

[13] MacDraw est un logiciel de dessin vectoriel de Macintosh (Cf. http://www.grenier-du-mac.net/fiches/applications/macdrawpro.htm).

ce qu'une couche prenne en charge l'événement. Une fois l'événement intercepté par une couche, sa transmission est interrompue. À chaque couche est associée un ensemble d'outils capables de gérer les événements reçus. Par exemple, si l'utilisateur effectue un « clic souris » sur le fond d'une image, la couche de fond intercepte l'événement et sait qu'il n'a été traité par aucune autre couche. L'outil de cette couche peut donc initier la gestion du rectangle de sélection en appelant l'outil de la couche responsable de cette opération. C'est ainsi que les outils d'une même couche ou de couches différentes peuvent collaborer entre eux pour gérer un événement complexe.

Notons que les couches ne se connaissent pas et communiquent entre elles via la capture des événements de l'utilisateur sans savoir ni l'origine de l'événement, ni sa destination. Cependant, les outils des différentes couches sont fortement liés les uns aux autres. En effet, pour activer un traitement dans une autre couche, un outil doit connaître l'existence de ses outils.

L'architecture *MultiCouche* a été implémentée dans la boîte à outils C++InterViews. En effet, cette boîte à outils offre plus qu'une simple encapsulation du système de fenêtrage et propose des mécanismes unificateurs pour la gestion des objets graphiques. Les propriétés de cette architecture permettent la construction effective d'éditeurs de qualité commerciale, c'est-à-dire optimisés, robustes et extensibles. Cependant, le niveau d'abstraction de l'architecture est élevé et requiert une bonne culture informatique qui est longue à acquérir [Fekete, 1996]. La structuration en couches permet de réaliser une application avec différents niveaux d'interaction et d'outils pour la gestion d'événements. Toutefois, lorsque le nombre de couches et d'outils devient important, leur gestion peut s'avérer difficile.

## IV.3. Le modèle H$^4$

*H$^4$* a été développé au laboratoire LISI à Poitiers, notamment par Laurent Guittet [Guittet et Pierra, 1993 ; Guittet, 1995]. Ce modèle tient son nom du fait qu'il structure l'architecture en quatre hiérarchies. Sa conception a été motivée par le besoin des AGICT[14] (Applications Graphiques Interactives de Conception Technique), et notamment des systèmes de CAO (Conception Assistée par Ordinateur). Ce modèle est basé sur l'architecture ARCH (décrite précédemment, Cf. § II.3), à laquelle il apporte quelques modifications. Les deux principales modifications sont : (1) l'ajout d'une communication directe entre les composants *Adaptateur de Domaine* et *Adaptateur de Présentation* ; (2) la structuration du dialogue permettant de décrire de manière précise le fonctionnement du contrôleur de dialogue.

Pour présenter le modèle *H$^4$*, nous nous baserons sur plusieurs travaux du LISI [Patry, 1999 ; Texier, 2000 ; Depaulis, 2002]. Le modèle H$^4$ est composé, de la même manière que *ARCH*, de cinq composants (Cf. figure I.20). Nous présentons ci-dessous le N*oyau fonctionnel*, *l'Adaptateur du noyau fonctionnel*, *l'Adaptateur de la présentation* et la P*résentation*. Nous détaillerons ensuite le composant C*ontrôleur de dialogue* qui, avec sa structuration du dialogue, représente l'un des principaux apports du modèle *H$^4$*.

[14] Les AGICT sont des systèmes interactifs permettant la création, la gestion et l'exploitation des représentations les plus proches possibles des objets du monde réel. Ils sont très couramment employés dans tous les domaines techniques comme la conception mécanique, la conception électronique, ou la conception architecturale [Patry, 1999].

- **le Noyau fonctionnel** représente les objets du domaine ainsi que toutes les fonctionnalités qui y sont associées. Il est la véritable sémantique de l'application ;

- **l'Adaptateur de noyau fonctionnel** est une surcouche logique du *Noyau fonctionnel*. Il s'agit d'une interface entre le *Contrôleur de dialogue* et le *Noyau fonctionnel* qui rend ces deux éléments indépendants entre eux. Il est donc chargé de traduire les informations du *Noyau fonctionnel* en données compréhensibles par le *Contrôleur de dialogue* et inversement. En outre, il est capable de communiquer avec *l'Adaptateur de présentation* pour afficher l'état du *Noyau fonctionnel*, quand le passage par le *Contrôleur de dialogue* n'est pas nécessaire ;

- **L'Adaptateur de présentation** est chargé de rendre l'application indépendante de l'implémentation de l'interface graphique en mettant à la disposition du *Contrôleur de dialogue* une surcouche des objets et des primitives de la Présentation (Boîte à Outils). Il fournit donc à l'application les outils permettant de réaliser l'affichage de l'état du *Noyau fonctionnel* ;

- **La Présentation** gère les entrées et les sorties avec l'utilisateur. Par l'intermédiaire de composants d'interaction (les Widgets d'une Boîte à Outils), cette couche logicielle récupère les informations fournies par l'utilisateur et affiche l'état du *Noyau fonctionnel*.

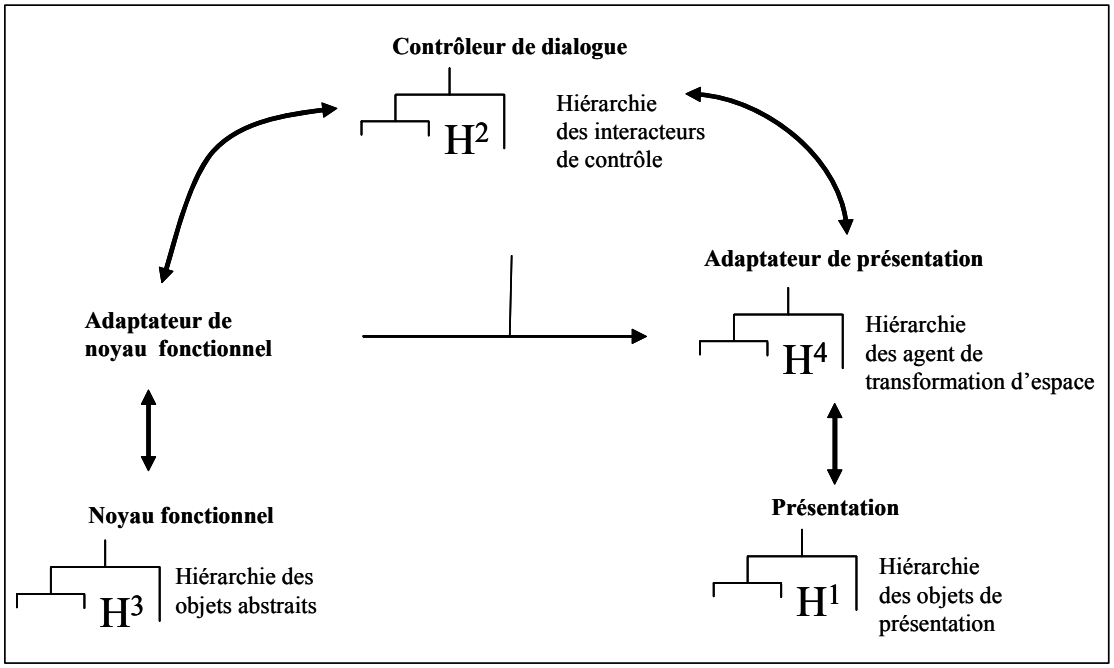



**Figure I.20.** Architecture $H^4$ [Guittet, 1995].

Contrairement à d'autres modèles d'architecture, la spécification de *H4* définit précisément le rôle, la composition et le fonctionnement du *Contrôleur de dialogue*. En effet, le *Contrôleur de dialogue* fait intervenir différents éléments à savoir : les jetons, les tâches, les interacteurs et les moniteurs.

- **Les jetons** : constituent l'unité d'information. On distingue les jetons paramètres des jetons commandes. Les premiers représentent une abstraction des données du *Noyau fonctionnel* et de la *Présentation*. Un jeton paramètre transporte ainsi des

valeurs. Par exemple, une position graphique donnée par l'utilisateur sera transportée par un jeton « position » stockant une abscisse et une ordonnée. Les jetons commandes, quant à eux, représentent l'intention de l'utilisateur. Ils transportent l'identifiant d'une tâche à activer ;

- **Les Tâches** : sont représentées par des signatures de fonction appelées Questionnaires. Un questionnaire est défini par son nom, ses paramètres d'entrée et, éventuellement, un paramètre de sortie. Les tâches sont implémentées dans *l'Adaptateur de noyau fonctionnel* et constituent le lien entre le *Contrôleur de dialogue* et les primitives de l'application. Les tâches remplissant des services similaires peuvent être regroupées à l'intérieur de niveaux d'abstraction communs.

- **Les interacteurs :** réunissent un ensemble de questionnaires (correspondant à des tâches) et représentent ainsi un niveau d'abstraction. Un interacteur peut agréger plusieurs entrées de bas niveau en une seule sortie de plus haut niveau. Il permet au système de faire abstraction de ces entrées de bas niveau. Notons que les interacteurs, dans $H^4$, sont réalisés à l'aide des automates à états finis particuliers qui sont les réseaux de transitions augmentés (ATN15 pour Augmented Transition Network [Woods, 1970]).

- **Le moniteur** : prend en charge l'organisation hiérarchique des interacteurs (donc des questionnaires et plus généralement des tâches). En effet, les interacteurs sont classés de bas en haut en suivant leur niveau d'abstraction (les interacteurs de plus bas niveau sont placés sous ceux de niveau plus élevé). Le moniteur récupère les jetons de la couche de présentation et les transmet successivement à tous les interacteurs, en commençant par celui de plus bas niveau.

Le fonctionnement du contrôleur de dialogue (Cf. figure I.21) se base sur les quatre entités présentées ci-dessus. En effet, lorsque l'utilisateur effectue une action (flèche numéro 1), *l'Adaptateur de présentation* transforme la donnée reçue de la *Présentation* en un jeton d'un type correspondant contenant une valeur (par exemple, un clic de souris peut être transformé en un jeton de type « position » contenant les coordonnées du point de l'écran sur lequel l'utilisateur a cliqué). Celui-ci est transmis au Moniteur qui le présente au premier interacteur de sa hiérarchie (flèche numéro 2). L'interacteur peut alors l'accepter, si l'état courant de l'automate qui le représente possède une transition sur ce type de jeton. Dans ce cas, l'automate change d'état et stocke la valeur du jeton. D'autre part, si la transition replace l'automate dans son état initial, l'interacteur appelle le questionnaire correspondant en lui transmettant toutes les valeurs enregistrées (flèche numéro 3). Celui-ci est ensuite chargé de faire le lien avec le *Noyau fonctionnel* en transformant les jetons en données du domaine et en appelant la primitive associée (flèche numéro 4). Dans le cas où cette primitive produit un résultat (cas d'une tâche productive), la valeur calculée est à son tour transformée en jeton puis rendue au moniteur (flèche numéro 5) qui se charge de la transmettre aux interacteurs (flèche numéro 6) suivants dans la hiérarchie [Depaulis, 2002].

Dans le cas où l'automate de l'interacteur ne se trouve pas dans un état correspondant à l'activation d'une transition par le jeton proposé, il le rend au moniteur qui le propage à l'interacteur suivant dans la hiérarchie, et ainsi de suite jusqu'à ce qu'il soit consommé ou qu'il n'y ait plus d'interacteur.

[15] Un ATN peut-être vu comme un automate à états fini comprenant une variable d'état, appelée registre.

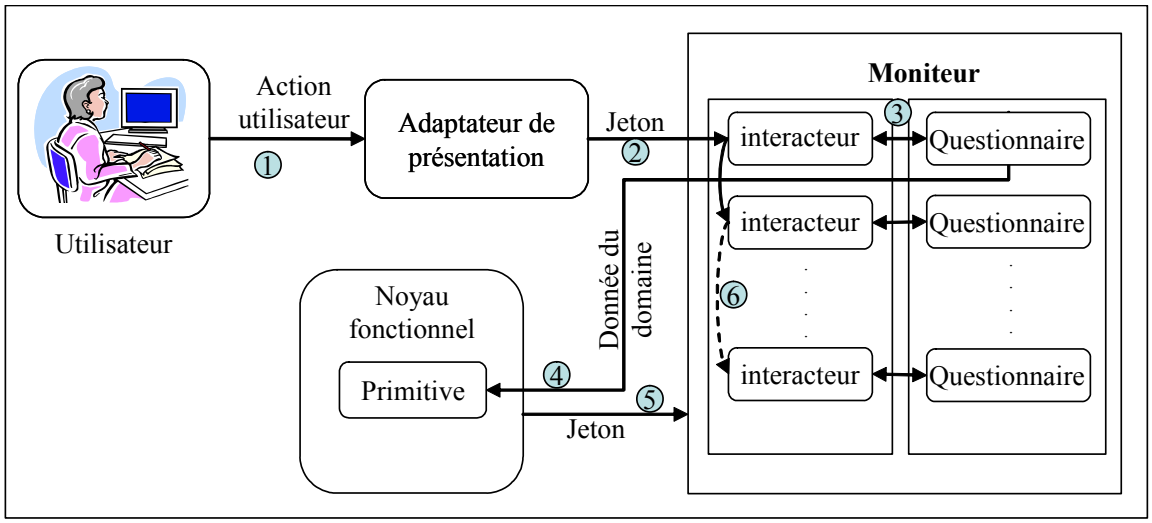



**Figure I.21.** Fonctionnement du contrôleur de dialogue de $H^4$

*$H^4$* présente de nombreux points communs avec les modèles hybrides présentés précédemment. Son architecture générale ressemble à celle de *PAC-Amodeus*, avec une organisation différente des agents de contrôle, ainsi que du mécanisme de passage de paramètres. Ces deux points ressemblent par ailleurs aux mécanismes présentés dans le modèle en couches avec des agents organisés hiérarchiquement s'échangeant des objets de plus en plus complexes [Patry, 1999].

Bien que le modèle *$H^4$* permette la structuration de dialogue avec son contrôleur de dialogue assez détaillé, il s'avère que l'implémentation de celui-ci est une tâche fastidieuse et difficile [Depaulis, 2002].

## IV.4. Conclusion sur les architectures hybrides

Les modèles hybrides que nous venons de voir tentent de répondre aux attentes laissées en suspens par les modèles centralisés et répartis. En utilisant une décomposition globale et en décrivant de manière plus précise le rôle et le fonctionnement des composants, les modèles hybride offrent un support considérable à la conception d'une application interactive [Fekete, 1996 ; Patry, 1999 ; Coutaz et Nigay, 2001 ; Depaulis, 2002].

Malgré cela, les modèles hybrides restent relativement inadaptés à la modélisation d'un dialogue structuré complexe. PAC-Amodeus reste trop imprécis sur l'identification des agents du Contrôleur de Dialogue et ne permet pas la décomposition des tâches sémantiques et articulatoires. Le modèle *Multi-couche*, quant à lui, présente l'inconvénient de lier de manière trop forte les différents outils de couches de son modèle. Dans le cas de tâches complexes, ces interactions fortes peuvent devenir difficilement gérables. Le modèle *$H^4$* propose la structuration de dialogue à travers un contrôleur de dialogue assez complexe en introduisant la notion de jeton, moniteur et interacteur ; mais son utilisation devient un handicap quand il s'agit d'interface hautement interactive [Depaulis, 2002].

# V. Conclusion

Ce chapitre a été consacré à un état de l'art sur les architectures des systèmes interactifs. Les principes de base de plusieurs modèles d'architecture ont été présentés. Leur objectif est de fournir des stratégies de décomposition fonctionnelle et/ou structurelle. Ils constituent une

avancée importante dans l'étude et la mise en œuvre des systèmes interactifs. Un aspect primordial de ces architectures est la description de la décomposition d'une application interactive en différents éléments directeurs. Une telle décomposition permet une conception plus simple dans la mesure où chaque module peut être réalisé de manière plus ou moins indépendante. Il existe deux grands groupes de modèles de référence : les modèles d'architectures centralisées et les modèles d'architectures reparties. Les modèles hybrides sont un mixage entre les deux.

La première partie de ce chapitre a été consacrée aux architectures centralisées. Ces architectures sont historiquement les premiers modèles d'architecture à avoir vu le jour. Elles décomposent un système interactif en plusieurs couches d'abstraction, de celles qui est responsable des interactions directes avec l'utilisateur (interface) jusqu'au noyau fonctionnel (application). Nous avons présenté dans cette partie et en premier lieu, le modèle *Langage*. Celui-ci fut le premier modèle qui a mis en évidence la séparation entre les aspects liés à l'IHM. Ensuite nous avons illustré le modèle de *Seeheim* et ses variantes ainsi que le modèle *ARCH*. Bien que tous les chercheurs dans le domaine d'IHM s'accordent sur le fait que ces modèles ont permis de séparer un système interactif en plusieurs modules, on peut noter que ces architectures ne définissent pas (ou très peu) ni la structure interne des modules, ni les interfaces d'échanges entre eux.

Les architectures réparties ont fait l'objet de la deuxième partie. Ces architectures sont aussi appelées multi-agents. Elles décrivent un autre style de décomposition pour les interfaces homme-machine. En effet, un système interactif est modélisé d'une façon homogène en un ensemble d'agents. Chaque modèle d'architecture répartie repose donc sur le concept agent dont le rôle et la composition varient d'une architecture à une autre. Afin d'éclaircir la notion d'agent, tel qu'elle est vue au sein de la communauté de recherche en IHM, nous avons définis brièvement la notion d'agent, d'un système multi-agents, ainsi que la différence entre objet et agent. Cette partie a passé en revue trois modèles d'architecture : *MVC*, *PAC* et *AMF*. Ces architectures sont particulièrement adaptées aux styles de programmation par objets, et permettent de décrire des aspects tels que la modularité, le parallélisme et la distribution.

La troisième partie a été consacrée aux architectures hybrides. Ceux-ci tentent de répondre aux attentes laissées en suspens par les modèles centralisés et répartis. Nous avons présenté dans cette partie les trois modèles les plus connus et cités dans la littérature : le modèle Pac-Amodeus, le modèle Multicouche et le modèle $H^4$. En utilisant une décomposition globale et en décrivant de manière plus précise le rôle et le fonctionnement des composants, ces architectures offrent un support considérable à la conception des systèmes interactifs.

C'est en s'inspirant de ces modèles que nous proposerons un modèle d'architecture dédié aux systèmes interactifs orientés agents (Cf. Chapitre 3), avec pour premier cadre d'application la supervision de processus industriel complexes. Cette architecture est une hybridation entre le modèle de *Seeheim* et le modèle *PAC*. Elle décompose un système interactif en trois composants fonctionnels, appelés respectivement : *Interface avec l'application* (connecté à l'application), *Contrôleur de dialogue* et Présentation (composant en lien direct avec l'utilisateur). Chaque composant est composé d'un ensemble d'agents. Ces agents sont modélisés via les réseaux de Petri de haut niveau (Cf. Annexe 1, § I) et communiquent entre eux pour répondre aux actions de l'utilisateur.

Comme nous l'avons vu, les modèles d'architecture sont avant tout un guide pour aider les concepteurs afin de concevoir et développer des systèmes interactifs. Mais la phase de conception et de développement ne suffisent pas pour aboutir à des applications de qualité. En effet, que ce soit en amont ou en aval de leur réalisation, l'évaluation est essentielle. Elle est nécessaire pour la production d'un système interactif de qualité en désignant et limitant les défauts de conception, à défaut de créer une l'interface correcte. C'est pour cela que nous proposons dans le deuxième chapitre un tour d'horizon des méthodes d'évaluation des systèmes interactifs, autour d'une classification.

*Chapitre 2*

# *Des méthodes et techniques usuelles d'évaluation des systèmes interactifs aux outils d'aide à l'évaluation automatique*

---

# I. Introduction

L'évaluation des systèmes interactifs et plus particulièrement de leur interface homme-machine est un problème récurrent depuis une trentaine d'années. L'évolution des moyens d'interaction entre l'utilisateur et la machine ainsi que les besoins et les spécificités inhérentes à l'évolution des sciences et technologies de l'information et de la communication nécessite systématiquement de nouveaux critères et méthodes pour l'évaluation des interactions Homme-Machine. Il s'avère en effet que les utilisateurs rencontrent aujourd'hui, des difficultés pour interagir avec les nouveaux systèmes interactifs, non bien étudiés, ou encore mal évalués, ne répondant pas toujours bien à leurs besoins.

Afin d'améliorer la qualité des interactions entre l'homme et la machine, des études récentes sont orientées vers l'évaluation des systèmes interactifs sous plusieurs angles et points de vue. On se propose dans ce chapitre de présenter, sans souci d'exhaustivité, mais plutôt de représentativité, les principales méthodes d'évaluations les plus citées et étudiées actuellement. Cette présentation nous permettra de construire un cadre d'étude pour l'introduction de notre outil d'aide à l'évaluation dédiée aux systèmes interactifs orientés agents présentée plus loin dans le chapitre trois. Pour ce faire nous avons structuré ce chapitre en trois parties.

Avant d'aborder la présentation des méthodes d'évaluation des systèmes interactifs et plus particulièrement des interfaces homme-machine, la première partie de ce chapitre illustre le principe de l'évaluation ainsi que les classifications possibles des méthodes d'évaluation en vue de les présenter dans la seconde partie.

Dans la seconde partie nous présentons un panorama des méthodes d'évaluation. Ces méthodes sont regroupées en trois grandes catégories : (1) les méthodes basées sur des techniques d'observation de l'utilisateur réel et de recueil des données de l'interaction, (2) les méthodes basées sur l'intervention d'experts (en interaction homme-machine, en psychologie cognitive ou en ergonomie) et (3) les méthodes analytiques basées sur des modèles formels prédictifs intégrant des connaissances sur la tâche et sur des grammaires ou des modèles formels de qualité de l'interface. Dans chaque catégorie sont décrites la ou les méthodes d'évaluation les plus représentatives ou celles qui ont servi de base à d'autres méthodes.

La troisième partie s'intéresse aux méthodes d'évaluation automatique. Après avoir introduit les méthodes d'aide aux choix de la (des) méthode(s) d'évaluation à utiliser pour l'évaluation d'un système interactif, nous abordons l'évaluation automatique proprement dite ; c'est-à-dire l'automatisation de la capture, l'analyse et la critique. Nous décrivons quelques outils logiciels d'évaluation automatique afin de construire un cadre de référence que nous exploiterons aux chapitres trois et quatre.

# II. Principes et classification des méthodes d'évaluation

## II.1. Principe de l'évaluation

Pour définir l'évaluation, nous reprenons la définition proposée par [Senach, 1990], qui énonce que « toute évaluation consiste à comparer un modèle de l'objet évalué à un modèle de référence permettant d'établir des conclusions ». Le principe de cette définition est visible sur la figure II.1. Lors de l'évaluation, le modèle que l'on qualifie « d'observé » (ou d'analysé) est comparé à un modèle de référence. Ce modèle doit être représentatif de l'adéquation de l'interface évaluée par rapport aux besoins spécifiques définis par le concepteur. Le résultat de cette comparaison, entre modèle observé et modèle de référence, définit le niveau d'adéquation de l'interface par rapport aux besoins déjà spécifiés et permet ainsi de déterminer les changements et les améliorations à apporter à l'IHM.

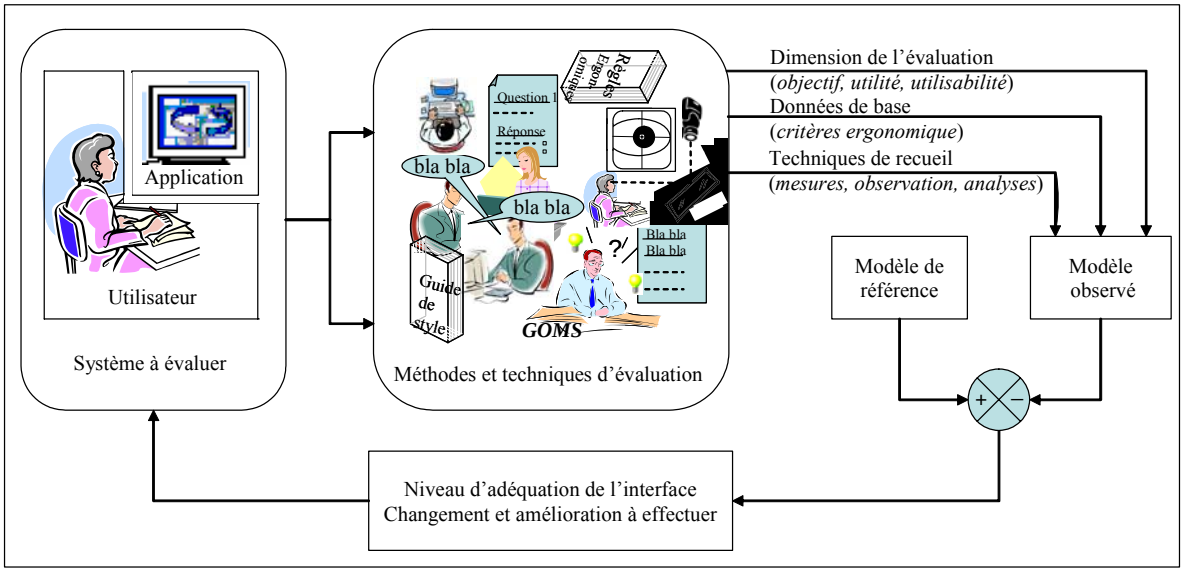



**Figure II.1.** Principe de base de l'évaluation inspiré de [Senach, 1990 ; Grislin, 1995]

Bien entendu, on ne peut pas aborder l'évaluation sans parler de l'utilisabilité et l'utilité. Ces deux propriétés sont habituellement explorées dans l'évaluation des interfaces homme-machine [Senach, 1990 ; Nielsen, 1993 ; Baccino *et al.*, 2005 ; Moha *et al.*, 2005]. Les chercheurs du domaine définissent l'utilité et l'utilisabilité comme suit :

- **l'utilité** : détermine si l'interface répond aux besoins de l'utilisateur ; en d'autres termes, si l'application permet à l'utilisateur d'atteindre ses objectifs de travail. L'utilité englobe la notion de performance du système, la capacité fonctionnelle et la qualité de l'assistance technique proposée [Senach, 1990 ; Nielsen, 1993].

- **l'utilisabilité** : en provenance de la traduction de « usability » en anglais, est un concept qui date du début des années 1980 [Eason, 1984]. En se posant la question «que faut-il savoir sur les utilisateurs pour construire une interface de qualité », [Robert, 2003] précise que les facteurs responsables de l'utilisabilité des l'interfaces homme-machine sont nombreux[16] : il s'agit de la compatibilité, la

[16] Une large définition de ces facteurs existe dans [Nielsen, 1993 ; Schneiderman, 1992 ; Grislin, 1995]. On retrouve également sur le site http://www.usabilityhome.com/ un panorama des méthodes d'évaluation de l'utilisabilité.

cohérence, la clarté visuelle, la flexibilité et le contrôle, les retours d'information, l'aide en ligne, la gestion des erreurs…(voir aussi les critères de [Ravden et Johnson, 1989] et ceux de [Nielsen, 1993]).

Comme nous le verrons plus loin (Cf. § III), les méthodes d'évaluation sont nombreuses ; chaque méthode choisit de privilégier un ou plusieurs attributs de l'utilisabilité et de l'utilité à travers la mesure de différentes variables : la durée d'exécution, le taux d'erreurs, etc., ou à travers l'opinion ou les attitudes des utilisateurs [Mick *et al.*, 2005].

Il n'existe pas, *a priori*, une méthode d'évaluation plus efficace que les autres. En effet, la qualité des résultats d'une technique d'évaluation est fortement liée à la fois au contexte[17] dans lequel elle est appliquée et à l'objectif de l'évaluation. Par exemple, il n'est pas efficace d'utiliser les questionnaires auprès d'utilisateurs pour déterminer entre deux options de conception celle qui permet de réduire la durée d'exécution de la tâche [Farenc, 1997].

Il convient cependant de souligner que l'évaluation des interfaces homme-machine dédiée aux systèmes interactifs traditionnels est différente de l'évaluation des interfaces Web [Ivory et Hearst, 2001]. En effet, les systèmes interactifs sont plus fonctionnels que les applications Web. Plus concrètement, l'utilisateur d'un système interactif accomplit des tâches telles que par exemple l'enregistrement d'un document suivi d'une séquence spécifique d'actions. Les applications Web (en tout cas la plupart d'entre eux), offrent des fonctionnalités limitées tel que la sélection d'un lien ou le remplissage d'un formulaire, leurs principaux rôles est de fournir de l'information. A ce sujet, dans une étude portant sur cent trente méthodes d'évaluation des systèmes interactifs et des sites Web, [Ivory et Hearst, 2001] précisent que seulement vingt-neuf méthodes peuvent être applicables aussi bien aux sites Web qu'aux systèmes interactifs. Nous nous intéressons dans ce mémoire à l'évaluation des systèmes interactifs et plus particulièrement à l'évaluation des interfaces homme-machine. Le lecteur intéressé par l'évaluation des sites Web[18] peut trouver dans [Mariage, 2005 ; Beirekdar, 2004] d'intéressantes études, très complètes à ce sujet.

Les méthodes d'évaluation usuelles sont nombreuses ; afin de pouvoir bien les présenter, il est nécessaire de les classifier selon des approches. Nous présentons dans la section suivante plusieurs classifications parmi les plus connues dans la littérature.

## II.2. Classifications des méthodes d'évaluation

Le problème de classification des méthodes d'évaluation est de taille. En effet selon [Bastien et Scapin, 2001], la classification devrait permette de décrire de manière exhaustive et adéquate les méthodes d'évaluation existantes afin d'en faciliter la sélection en fonction de critères tels que :

- **les objectifs d'évaluation** : s'agit-il d'un diagnostic, c'est-à-dire de détection des erreurs de conception en vue de fournir des alternatives de conception, ou d'évaluation permettant de déterminer jusqu'à quel point le système interactif est adapté pour les tâches pour lesquelles il a été conçu, ou encore d'une évaluation de conformité à des normes ?

[17] Le lecteur intéressé peut trouver dans [Rey, 2005] une étude complète de la notion de contexte en interaction homme-machine.

[18] Les sites Web suivants traitent de l'évaluation des sites Web : http://www.ergoweb.ca/ ; http://www.auditweb.net ; http://www.cyberbee.com/guides.html

- **la source des données de l'évaluation** : quelles sont les performances utilisateurs ? les caractéristiques de l'interface ? de l'interaction ? etc. ;

- **le moment de l'évaluation** : qui détermine l'état, la forme, la représentation du système interactif à évaluer. Par exemple on peut se trouver en présence d'une spécification, d'une maquette, d'un prototype, etc.

On peut facilement constater que plusieurs axes de classification des méthodes d'évaluation sont possibles. Notre objectif n'est pas de critiquer et d'analyser les classifications existantes en vue d'en proposer une nouvelle[19], mais d'illustrer les classifications les plus connues dans la littérature afin de décrire plus loin (Cf. § III) les méthodes d'évaluation usuelles tout en les classant selon des approches.

Plusieurs approches ont été proposées pour la classification des méthodes d'évaluation des systèmes interactifs :

- **la classification de [Senach, 1990]** distingue les méthodes d'évaluation suivant les données comportementales (méthodes empiriques) ou suivant les données sur l'interface (méthodes analytiques) :
  - Les évaluations empiriques consistent à recueillir des données relatives au comportement de l'utilisateur lors de l'utilisation du système. Ce type d'évaluation nécessite l'existence d'un système réel (maquette, prototype ou système final) et la présence d'utilisateurs.
  - Les évaluations analytiques évaluent la conception du système et non son utilisation. Les interfaces sont étudiées selon un ensemble de référents afin de contrôler qu'elles possèdent bien certaines qualités et de détecter les problèmes qu'elles peuvent poser. Ces référents peuvent être formels ou informels.

- **la classification de [Whitefield, 1991]** classe les méthodes d'évaluation selon la présence réelle ou représentée de l'utilisateur et du système. Whitefield obtient ainsi quatre approches d'évaluation possibles :
  - les méthodes analytiques (utilisateur et système non réels) : font recours à une représentation de l'utilisateur et du système pour prédire les performances que l'utilisateur peut réaliser. Comme par exemple les modèles de tâche (Cf. III.3.1.1),
  - les rapports de spécialistes (système réel, utilisateur représenté) : ne font pas appel aux utilisateurs réels, mais on dispose de la présence du système réel pour l'évaluation. Comme par exemple les guides ergonomiques (Cf. III.2.2),
  - les rapports d'utilisateurs (utilisateur réel, système représenté) : sont généralement utilisés quand on ne dispose pas de système réel. Dans ce cas on demande à l'utilisateur de juger certains aspects conceptuels généraux du système. Par exemple la technique de questionnaire (Cf. III.1.1.1),
  - les méthodes d'observation (utilisateur et système réels) : consistent, comme leur nom l'indique, à observer l'utilisateur en interaction avec le système afin de déterminer les erreurs et les défaillances de ce dernier. Par exemple les approches centrées sur l'utilisateur (Cf. III.1).

  Cette classification n'est pas pour autant différente de celle proposée par [Bastien et Scapin, 2001] que nous verrons plus loin.

[19] Le lecteur intéressé par une étude presque exhaustive de la classification des méthodes d'évaluation peut se référer au volume 13 (1998) de la revue Human-Computer Interaction ou encore à plusieurs thèses, telles celles de [Farenc, 1997] et de [Mariage, 2005].

- **la classification de [Holyer, 1993]** introduit une classification selon la discipline qui sert de fondement aux méthodes envisagées. L'idée forte derrière cette classification est de séparer les méthodes d'évaluation selon leur « philosophie d'origine ». Quatre types de méthodes sont ainsi distinguées : (1) les méthodes issues de la psychologie cognitive, qui tentent d'identifier le processus cognitif de l'utilisateur interagissant avec le système ; (2) les méthodes issues de la psychologie sociale, qui sont principalement basées sur les questionnaires, la verbalisation et l'interview ; (3) les méthodes issues des sciences sociales, qui consistent en l'observation d'un utilisateur par un ou plusieurs experts, par exemple les approches empiriques (Cf. III.1.1) ; (4) les approches issues de l'engineering, qui regroupent les méthodes d'évaluation basées sur l'expérience de un ou plusieurs spécialistes, par exemple les approches basées sur des experts (Cf. III.2).

- **la classification de [Balbo, 1994]** classe les méthodes d'évaluation selon plusieurs paramètres. Les dimensions intervenant dans cette classification sont :
  − les connaissances requises caractérisées par le coût cognitif d'accès à la méthode et les descriptions qui représentent les informations sur l'objet de l'évaluation nécessaires au processus d'évaluation (modèle de l'utilisateur, modèle de tâche, spécifications externes, etc.),
  − les ressources matérielles qui correspondent à tous les moyens physiques mis en oeuvre pour l'évaluation telles que : les supports pour la capture des données, l'objet de l'évaluation et le niveau d'abstraction des données capturées,
  − les facteurs situationnels qui décrivent le contexte de l'évaluation suivant cinq dimensions : l'étape dans le processus de développement durant laquelle peut se faire l'évaluation, le lieu de l'évaluation, le type d'application caractérisé par rapport à la tâche, la typologie des interfaces (interface graphique à manipulation directe, interface multimodale, etc.) ainsi que l'enveloppe budgétaire et les plannings d'évaluation,
  − les moyens humains qui désignent l'ensemble des personnes impliquées dans le processus d'évaluation : leur nombre et leurs origines,
  − les résultats fournis sont caractérisés par le niveau d'abstraction et le type des informations issues de l'évaluation.

- **la classification de [Farenc, 1997]** sépare les approches d'évaluation selon trois éléments centraux de l'interaction homme-machine, à savoir : l'utilisateur, la tâche et le système.
  − l'utilisateur : une méthode d'évaluation peut être réalisée soit en prenant en compte un ou plusieurs utilisateurs représentatifs des utilisateurs finaux, soit sans prendre en compte les utilisateurs. Dans ce dernier cas la méthode d'évaluation est utilisée quel que soit l'utilisateur et donc valable pour N utilisateurs,
  − la tâche : de la même manière, certaines méthodes d'évaluation s'appliquent pour une tâche qui doit être précisée pour l'évaluation, d'autres, quelle que soit la tâche, donc pour N tâches,
  − le système : une autre dimension relative au choix de la méthode d'évaluation est la nature des résultats de la méthode. Ces résultats peuvent porter sur le système, sur l'interface ou sur les deux.

- **la classification de [Bastien et Scapin, 2001]** sépare les approches d'évaluation en deux grandes catégories :

– les méthodes requérant la participation directe des utilisateurs qui se basent essentiellement sur la présence de l'utilisateur. Cette catégorie regroupe les deux approches : les rapports des utilisateurs et l'observation de l'utilisateur [Whitefield, 1991],
– les méthodes s'appliquant aux caractéristiques de l'interface qui se distinguent des précédentes par l'absence de l'interaction directe entre l'utilisateur et le système. Dans cette catégorie, les utilisateurs tout comme leurs tâches sont représentés.

- **La classification de [Grislin et Kolski, 1996]** classe les méthodes d'évaluation en trois grandes approches :
  – les approches centrées sur les avis et/ou les activités des utilisateurs,
  – les approches qualifiées d'expertes centrées sur le jugement d'experts en communication homme-machine ou sur l'utilisation de grilles d'évaluation ou de questionnaires listant les qualités d'une bonne IHM,
  – les approches qualifiées d'analytiques, centrées sur une modélisation de l'IHM et/ou de l'interaction homme-machine. Celles-ci consistent le plus souvent à effectuer l'évaluation à l'aide de métriques objectives à partir d'un modèle descriptif des tâches humaines, ou à partir d'une description des pages-écrans.

Nous retiendrons cette dernière classification (Cf. figure II.2) puisque l'on retrouve, explicitement ou implicitement dans la plupart des classifications existantes, les trois approches proposées par [Grislin et Kolski, 1996]. La classification complète est disponible dans [Grislin, 1995] et recense une quarantaine de méthodes.

Toutefois comme le soulignent [Grislin, 1995 ; Abed, 2001 ; Bastien et Scapin, 2001], il paraîtrait irréaliste de considérer ces regroupements de manière indépendante. En effet, des relations transversales entre méthodes sont souvent très utiles. Par exemple, lors de l'analyse de situations de références avec des utilisateurs (Cf. § III.1), il est souvent utile de procéder à une modélisation des tâches (Cf. § III.3.1.1) dans un but de spécification ; mais il est aussi possible de réutiliser cette modélisation lorsque le système interactif est réalisé, afin de la confronter à une modélisation des activités réelles des utilisateurs (Cf. § III.1.1).

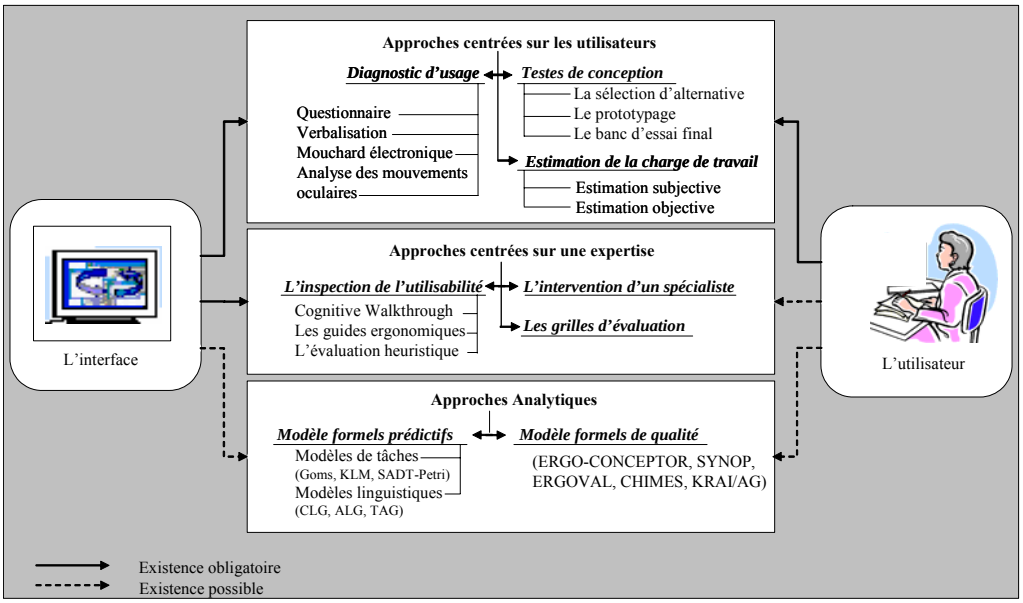



**Figure II.2.** Extrait de la classification des méthodes et techniques d'évaluation [Grislin et Kolski, 1996]

Rappelons que notre objectif dans ce chapitre n'étant pas de faire un recensement exhaustif de toutes les méthodes d'évaluation existantes dans la mesure où on retrouve plusieurs documents qui traitent ce sujet, mais plutôt de présenter les méthodes d'évaluation les plus représentatives. Nous accorderons plus de place aux approches empiriques de diagnostic d'usage. En effet ces approches nous intéressent plus particulièrement dans la mesure où nous proposons dans le chapitre trois un outil d'aide à l'évaluation basée sur ces derniers. Ces méthodes sont présentées dans la section suivante.

## III. Méthodes, techniques et outils d'évaluation des systèmes interactifs

Cette section présente un panorama des méthodes d'évaluation. Ces méthodes sont regroupées en trois grandes catégories suivant le principe énoncé au paragraphe II.2 et présenté sur la figure II.2 :

- les méthodes basées sur des techniques d'observation de l'utilisateur réel et de recueil des données de l'interaction,
- les méthodes basées sur l'intervention d'experts en interaction homme-machine, en psychologie cognitive ou en ergonomie,
- les méthodes analytiques basées sur des modèles formels prédictifs intégrant des connaissances sur la tâche et sur des grammaires ou des modèles formels de qualité.

Dans chaque catégorie sont décrites la ou les méthodes d'évaluation les plus représentatives ou celles qui ont servi de base à d'autres méthodes. Pour chaque méthode, sont définis les points suivants :

- objectif : l'objectif d'une méthode d'évaluation est en général de tester l'utilité et l'utilisabilité du système à évaluer ; cependant chaque méthode se focalise sur une dimension particulière ou une interprétation particulière de ces deux facteurs,
- principe et description : le principe ainsi que l'utilisation de la méthode sont décrits dans cette partie,
- avantages et inconvénients : sont ici décrits les principaux avantages et Inconvénients de la méthode qui peuvent être rencontrés lors de son utilisation.

### III.1. Approches centrées sur l'utilisateur

Ces approches sont basées sur des techniques d'observation de l'utilisateur réel (utilisateurs finaux) et de recueil des données de l'interaction (questionnaire, interview, verbalisation, oculométrie, estimation de la charge de travail, etc.) afin d'analyser les traces de l'activité des utilisateurs. Ces approches permettent de détecter les problèmes réels que rencontre l'utilisateur lorsqu'il réalise sa tâche avec le système. Les résultats portent sur l'interface et le système mais comme le précise [Farenc, 1997], ils n'offrent pas les moyens de corriger les erreurs.

Parmi ces approches centrées sur l'utilisateur on peut citer les approches empiriques de diagnostic d'usage (utilisables lorsque l'IHM est réalisée totalement ou partiellement), les approches centrées sur l'estimation de la charge de travail et les approches basées sur les tests de conception (interviennent tout au long du cycle de développement de l'IHM).

## III.1.1. Approche empirique de diagnostic d'usage

Cette approche n'est possible que si l'IHM est opérationnelle, et est prête pour être présentée aux utilisateurs finaux (des exceptions pour les interfaces qui sont dans une phase bien avancée de leurs développements peuvent être faites). L'évaluation se base essentiellement sur un recueil d'informations.

Dans la mesure où nous retiendrons les méthodes d'évaluation empirique de diagnostic d'usage (notre choix sera justifié plus loin) pour la proposition d'un outil d'aide à l'évaluation dédié aux systèmes interactifs orientée agent (Cf. Chapitre 3 et 4), nous leur consacrerons une partie plus importante par rapport aux autres approches.

Parmi les nombreuses méthodes empiriques qui existent, on peut citer les plus connues qui sont : les interviews, les questionnaires, le mouchard électronique, l'oculomètre. Ils sont maintenant présentés.

## III.1.1.1. Questionnaire d'utilisation

### Objectif

L'objectif de cette technique est de recueillir, à l'aide de documents en principe structurés, un ensemble d'appréciations, d'opinions et d'attitudes, de l'utilisateur après qu'il ait interagi avec le système. Les questionnaires permettent surtout de déterminer la satisfaction ou les problèmes de l'utilisateur qu'il est difficile ou impossible de mesurer autrement qu'avec le questionnaire [Nielsen, 1993].

### Principe et description

Il s'agit de questionner l'utilisateur sur les besoins d'information, les incohérences de l'interface, les points forts ou faibles qu'il a ressentis lors de l'utilisation d'un système et éventuellement ses suggestions sur la manière de les corriger [Baccino *et al.*, 2005]. Le questionnaire peut aborder les aspects liés au fonctionnement du système et/ou les aspects liés à l'ergonomie de l'interface. Un questionnaire est composé d'une série de questions. Il doit être bien établi avec des finalités bien définies afin d'éviter dispersion, redondance et données inutiles [Kovács *et al.*, 2004]. En effet, chaque question doit répondre en partie à la problématique traitée et ne doit pas consister à fournir à elle toute seule une réponse globale. Par exemple pour évaluer l'aspect ergonomique de l'interface, plusieurs questions peuvent être envisagées afin de recueillir l'avis de l'utilisateur sur cet aspect. Une question trop générale du type « pensez-vous que l'ergonomie de l'interface est bien faite ? » est à éviter. Il faut plutôt privilégier des questions plus ciblées du type « que pensez-vous des couleurs utilisées ? » ou encore « les éléments des menus sont-ils facilement identifiables » avec par exemple une réponse sous forme d'échelle.

Dans un questionnaire on peut trouver différents types de questions qui permettent soit d'évaluer un aspect particulier du système ou tout simplement d'avoir l'opinion de l'utilisateur :

- **les questions ouvertes** : permettent à l'utilisateur de s'exprimer librement en utilisant son propre langage. Elles permettent de connaître une opinion ou une information générale sur le système. Une question ouverte peut être par exemple,

« quel aspect de l'interface préférez-vous ? » ou encore « quels sont les problèmes majeurs que vous avez rencontrés lors de l'utilisation du système ? ». Ces questions ne sont pas recommandées quand il s'agit d'évaluer un aspect bien déterminé du système. Dans ce cas il faut privilégier les questions fermées.

- **les questions fermées** : à l'inverse des questions ouvertes, elles proposent un ensemble de réponses prédéterminées. Ces questions sont à la base des QCM (Questions à Choix Multiples). L'évaluateur peut imposer à l'utilisateur une réponse unique ou lui laisser la possibilité d'effectuer plusieurs choix. Cependant l'élaboration de la liste des choix doit être bien étudiée afin que l'utilisateur puisse trouver une réponse lui convenant. Dans le cas contraire l'utilisateur risque de répondre d'une façon hasardeuse ou de ne pas répondre du tout. Une question fermée peut par ailleurs donner à l'utilisateur la possibilité de s'exprimer. On parle alors de questions semi-fermées. La figure II.3.a montre un exemple d'une question semi-fermée à choix multiples ; dans cet exemple l'utilisateur peut proposer sa propre réponse s'il n'en trouve pas une qui lui convient.

- **les questions scalaires** : permettent à l'utilisateur d'exprimer son opinion sur une échelle prédéfinie. Cette échelle peut avoir seulement deux valeurs (échelle binaire : Vrai/Faux ou Oui/Non) ou plusieurs valeurs (échelle multiple : à 5, 7, 10 valeurs). Dans [Baccino *et al.*, 2005] les questions scalaires à échelle multiple sont classifiées en trois catégorie :

  – échelle de Likert : on interroge l'utilisateur sur son accord ou son désaccord avec une assertion (Cf. figure II.3.c),

  – échelle de rang : on invite l'utilisateur à ranger par ordre d'importance des réponses à une question (Cf. figure II.3.b),

  – échelle sémantique différentielle : on demande à l'utilisateur de juger le système selon deux listes d'antonymes. Par exemple, (facile, utile, plaisant, rapide) et (difficile, inutile, déplaisant, lent).

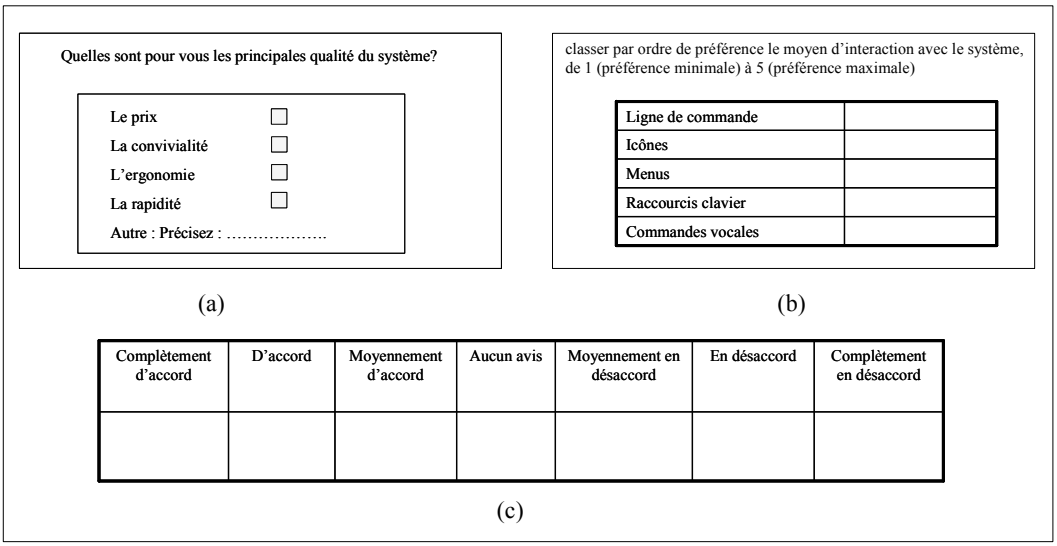



**Figure II.3.** Exemple de questionnaire : (a) ouvert à choix multiples (b) à échelle de rang (c) à échelle de Likert

### Avantages et inconvénients

Cette technique possède l'avantage d'être économique et utilisable dans la plupart des situations [Rushinek et Rushinek, 1986]. Elle permet à l'utilisateur de travailler « à tête reposée », même si ceci est susceptible, d'après [Root et Draper, 1983] d'introduire un biais dans ses réponses puisqu'il n'est pas toujours en mesure de se souvenir des difficultés rencontrées lors des différentes situations de travail.

Bien que cette technique semble être facile à mettre en place, l'élaboration d'un questionnaire fiable (permettant de récupérer des informations réellement utiles pour l'évaluation) n'est pas toujours facile. Selon [Baccino *et al.*, 2005] les problèmes récurrents que l'on rencontre lorsqu'on construit des questionnaires sont : (1) l'utilisation de termes chargés affectivement, trop imprécis ou trop ambigus. Par exemple, une question adressée à des étudiants américains en informatique sous la forme « what is your first langage ? » reçut comme réponse C++, JAVA… alors que les auteurs du questionnaire s'attendaient à des réponses du type anglais, français… (2) la possibilité de suggérer la réponse désirée ou d'entraîner une gêne du sujet lorsqu'il croit être évalué sur sa réponse. Par exemple, une question du type « avez-vous une bonne mémoire ? » peut intimider l'utilisateur et il aura tendance à donner une réponse erronée.

## III.1.1.2. l'interview ou l'entretien

### Objectif

L'interview a pour objectif de déterminer une première vue d'ensemble sur les problèmes d'utilité et d'utilisabilité du système à évaluer. Elle renseigne l'évaluateur sur les difficultés du système, les améliorations à envisager et le degré de satisfaction de l'utilisateur.

### Principe et description

L'interview consiste à s'entretenir avec des sujets représentatifs afin d'avoir un aperçu rapide des problèmes, concernant l'utilité et l'utilisabilité rencontrées par l'utilisateur lors de l'interaction avec le système proposé (Cf. figure II.4). Elle se base globalement sur des techniques utilisées en psychologie sociale [Holyer, 1993]. L'efficacité de l'interview dépend de sa préparation et des compétences de l'évaluateur qui assure son guidage. En effet, l'évaluateur doit préparer et planifier à l'avance les questions centrales à poser à l'utilisateur.

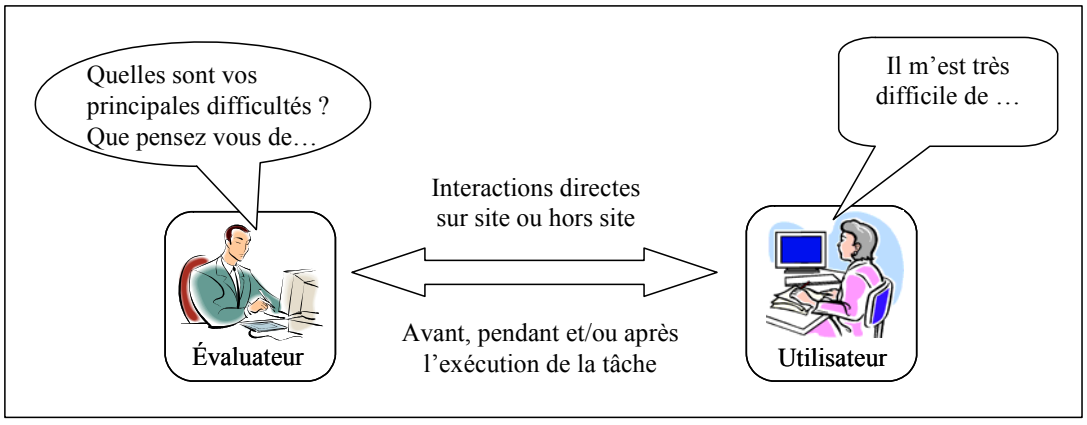



**Figure II.4.** Interaction directe par interview entre utilisateur et évaluateur

**Avantages et inconvénients**

L'interview a l'avantage d'être plus flexible que le questionnaire et permet d'orienter les questions directement vers l'information recherchée par l'évaluateur. D'ailleurs [Moha *et al.*, 2005] précise, d'après une enquête sur les pratiques de tests d'utilisabilité, que l'interview ou l'entretien est une technique très recommandée par les professionnels de l'évaluation.

Si l'interview est particulièrement efficace, elle ne peut être utilisée seule pour l'évaluation des IHM. En effet les interviews peuvent être effectuées sur le site de travail de l'utilisateur et/ou hors site, donc les risques de manquer de structuration et de sortir du contexte de travail lors des interviews sont importants.

## III.1.1.3. Le mouchard électronique

**Objectif**

Le mouchard électronique ou monitoring a pour objectif de recueillir automatiquement des données objectives (les actions des utilisateurs et leurs répercutions sur le système) en situation réelle de travail. Les données recueillies seront analysées ultérieurement (manuellement ou automatiquement) [Bastien et Scapin, 2002 ; Mariage, 2005 ; Ezzedine *et al.*, 2005].

**Principe et description**

Il convient tout d'abord de faire la différence entre le monitoring et le mouchard électronique. Le monitoring consiste simplement à faire de la surveillance ou la capture de données sans pour autant proposer une analyse quelconque. Le mouchard électronique quand à lui permet à la fois la capture et l'enregistrement des données et peut proposer une analyse plus au moins approfondie des données recueillies [Mariage, 2005]. Nous nous intéressons ici au mouchard électronique.

Un « mouchard électronique » est un outil logiciel qui assure le recueil automatique, en situation réelle, des actions des utilisateurs et de leurs répercussions sur le système (appui sur une touche, mouvement de la souris, sélection d'un objet dans un menu, apparition d'un message d'avertissement, fermeture d'une fenêtre, etc.). Le recueil d'information se fait de façon discrète, transparente pour l'évaluateur. L'utilisateur ne doit se sentir à aucun moment gêné par la présence du mouchard. Les données objectives recueillies au travers des interactions homme-machine peuvent ensuite être analysées et affichées sous une forme synthétique à l'évaluateur, lui facilitant l'analyse des problèmes d'utilisation.

La figure II.5 illustre l'utilisation de l'outil mouchard électronique comme support pour l'évaluation. La première étape consiste à recueillir les données provenant de l'interaction de l'utilisateur avec le système à évaluer. Stocker dans une base de données, les données recueillies seront ensuite analysées par l'évaluateur durant la deuxième phase. Il est à noter qu'avant d'être présentées à l'évaluateur, les données peuvent être au préalable analysées par le mouchard électronique afin de fournir des diagrammes, statistiques et des recommandations à l'évaluateur. La troisième étape consiste quant à elle à confronter l'utilisateur avec les données recueillies. La technique de questionnaire (décrite ci-dessus) peut être alors utilisée afin de mieux comprendre les réactions de l'utilisateur face à des situations de travail bien précises. En effet grâce aux enregistrements obtenus, l'évaluateur peut reconstituer les

modèles de l'activité réelle de l'utilisateur et les réactions de l'IHM et les comparer aux modèles des tâches à effectuer [Ezzedine et Abed, 1997]. Le résultat de cette comparaison peut être utile au concepteur pour revoir les spécifications de l'IHM.

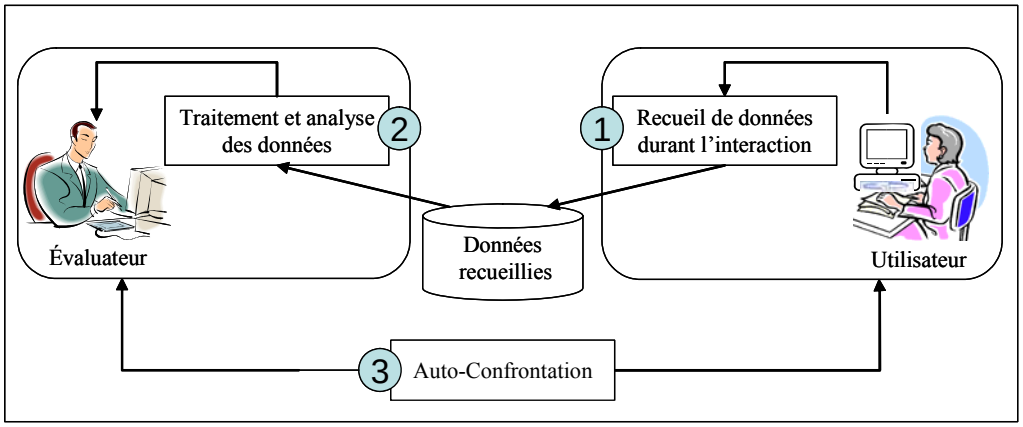



**Figure II.5.** Principe du mouchard électronique

**Avantages et inconvénients**

Le principal avantage du mouchard électronique est sa transparence vis-à-vis de l'utilisateur. En effet, lors de l'évaluation l'utilisateur ne se sent en principe à aucun moment gêné par la présence du mouchard, et se comporte ainsi d'une façon normale.

L'inconvénient des mouchards électroniques réside dans le traitement de la quantité de données qu'il enregistre et dans le niveau de granularité de l'analyse [Bastien et Scapin, 2002]. En effet, l'évaluateur peut se trouver devant une énorme base de données recueillies qui nécessite des heures (voir des journées) de travail pour le dépouillement. Par ailleurs, si le mouchard électronique ne propose pas une aide à l'évaluateur pour l'interprétation des données (sous forme de graphes, recommandations, etc.), les données recueillies peuvent être mal exploitées ou nécessiteraient plusieurs heures pour en tirer des résultats utiles pour l'évaluation.

## III.1.1.4 L'analyse des mouvements oculaires

**Objectif**

L'analyse des mouvements oculaires ou le « eye-tracking » permet de suivre les mouvements de l'œil et de connaître la localisation du regard. En effet, les mouvements oculaires de l'utilisateur peuvent être recueillis grâce à un appareil d'acquisition appelé oculomètre afin de déterminer des indices comme la durée de fixation d'une zone de l'interface homme-machine, l'identification des zones d'intérêt de l'IHM et ou les séquences successives d'observation, pour les corréler avec les actions et réactions de l'IHM [Abed, 1990 ; Simon 1993 ; Ezzedine et Abed, 1997 ; Pivec *et al.*, 2006 ; Chalon *et al.*, 2001].

**Principe et description**

La technique des mouvements oculaires consiste à enregistrer les trajectoires et les pauses oculaires générées lors de la lecture, de la recherche d'information ou de l'inspection d'une interface homme-machine. Il s'avère que dans la conduite de procédé industriel à l'aide

d'outil graphique (et particulièrement la supervision), il est fait énormément appel à la perception visuelle des informations. Dans ce cas, la direction absolue du regard devient un élément utilisable en parallèle avec d'autres moyens pour analyser l'activité d'un opérateur humain devant un écran ou un ensemble de supports d'information [Abed, 1990]. Ainsi les enregistrements des mouvements oculaires peuvent être rejoués en différé et commentés avec (et par) les utilisateurs.

La figure II.6 donne l'exemple d'un dispositif de capture de mouvement oculaire. Ce dispositif se compose principalement d'un émetteur qui génère un champ magnétique, d'un récepteur solidaire du casque, et d'une unité de traitement. Le centre de l'émetteur constitue l'origine du référentiel dans le quel évolue l'opérateur. La position et l'orientation du récepteur par rapport à l'émetteur sont calculées à partir des caractéristiques du champ magnétique et des tensions mesurées aux bornes des trois bobines constituant le récepteur [Berger, 1992 ; Simon, 1993].

Le lecteur intéressé peut avoir plus de détail concernant cette technique en se référant au travaux menés au sein du laboratoire LAMIH [Abed, 1990 ; Simon, 1993 ; Ezzedine et Abed, 1997 ; Loslever *et al.*, 2003] ou encore aux travaux de [Chalon *et al.*, 2001 ; Mullin *et al.*, 2001 ; Baccino *et al.*, 2005].

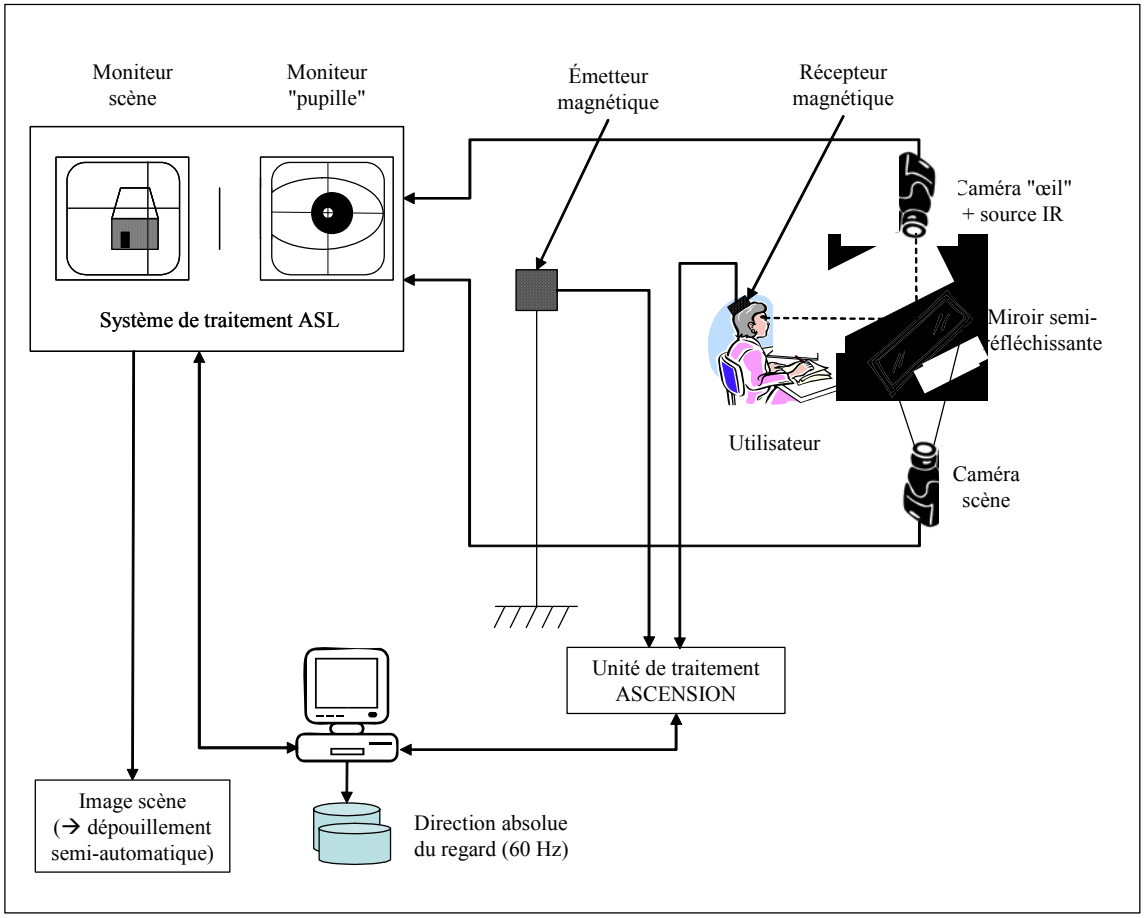



**Figure II.6.** Principe de l'évaluation par acquisition et analyse des mouvements oculaires système ASL-ASCENSION [Simon, 1993]

### Avantages et inconvénients

L'avantage de cette technique réside dans le fait qu'elle permet facilement (après dépouillement) d'étudier la manière dont l'utilisateur recherche et localise les informations utiles en fonction des différentes situations de fonctionnement du système (mode normal, mode dégradé) et d'identifier des lacunes liées aux interfaces homme-machine lors de certaines situations de crise [Kolski, 1997].

Il est à noter qu'il existe actuellement des équipements de capture du mouvement oculaire assez sophistiqués et qui permettent une totale liberté à l'utilisateur. En effet, l'utilisateur n'est plus obligé de mettre un casque qui peut le perturber pendant le déroulement de l'évaluation. A titre d'exemple le dispositif qu'utilise [Chalon, 2004] comprend deux émetteurs infra rouges articulés et une caméra infrarouge à orienter sur l'œil de l'utilisateur.

L'inconvénient majeur de cette technique réside dans la limitation du champ de visée. En effet avec les nouveaux dispositifs, la tête de l'utilisateur doit rester quasiment immobile. Le dépouillement des résultats est généralement une tâche fastidieuse et nécessite beaucoup de temps. Le prix du dispositif est relativement élevé.

## III.1.2. L'estimation de la charge de travail

### Objectif

Basée sur l'estimation du travail cognitif de l'utilisateur, l'estimation de la charge de travail permet de mesurer le niveau de difficulté liée à l'utilisation de l'IHM. Cette technique permet une estimation qualitative et/ou quantitative du niveau d'activité de l'utilisateur effectuant une tâche [Wierwille et Casali, 1983 ; Millot, 1988 ; Bayssié et Chaudron, 2002].

### Principe et description

L'estimation de la charge de travail résulte des répercussions directes des contenus des vues graphiques ainsi que de la manière de présenter l'information à l'utilisateur. Plusieurs spécialités contribuent à l'évaluation de la charge de travail : physiologie, psychologie, ergonomie, sciences pour l'ingénieur, etc. Il est à noter que les méthodes d'estimation de la charge de travail peuvent être :

- **subjectives** [Cooper et Harper, 1969 ; Wierwille et Casali, 1983], consistant à demander à l'utilisateur d'estimer sa charge de travail avec des échelles subjectives.

- **objective**s [Millot, 1988 ; Sperandio, 1996], exploitant les paramètres physiques ou physiologiques mesurés pour estimer la charge de travail. Parmi les méthodes d'estimation de la charge de travail objectives on peut citer les approches temporelles qui définissent la charge de travail comme étant le rapport entre le temps nécessaire à l'opérateur humain pour effectuer une tâche et le temps dont il dispose effectivement pour l'exécuter.

Le lecteur intéressé peut trouver dans [Berger, 1999] un panorama de méthodes et les techniques de mesure de la charge de travail.

### Avantages et inconvénients

L'avantage de cette méthode est qu'elle peut être utilisée *a priori*, lorsque l'interface est au stade du prototypage et *a posteriori* quand l'interface est finie. Par ailleurs, les résultats obtenus par cette méthode peuvent mettre en évidence des points sensibles pour les outils graphiques utilisés par les opérateurs humains en salle de contrôle et de procéder par la suite à des aménagements ergonomiques de ceux-ci [Kolski, 1997]. Les inconvénients majeurs de ces méthodes est de ne fournir qu'une estimation de la charge de travail ; pour des mesures plus précises, il s'agit de combiner plusieurs méthodes, nécessitant éventuellement la participation des spécialistes et d'experts dans le domaine de psychologie ou de la physiologie.

## III.1.3 Tests de conception

### Objectif

Les méthodes de tests de conception ont pour objectif d'évaluer et de valider un système interactif ou une interface homme-machine selon un cycle itératif tout au long du processus de développement avec des utilisateurs si possible représentatifs.

### Principe et description

Inspiré du modèle en Spirale[20] [Boehm, 1988], les méthodes de tests de conception consistent à recueillir des données au moyen de tests qui engendrent des modifications à apporter au système à évaluer. Ces modifications donnent naissance à une nouvelle version qui sera elle aussi mise à l'épreuve, et ainsi de suite jusqu'à obtenir le système final (Cf. figure II.7).

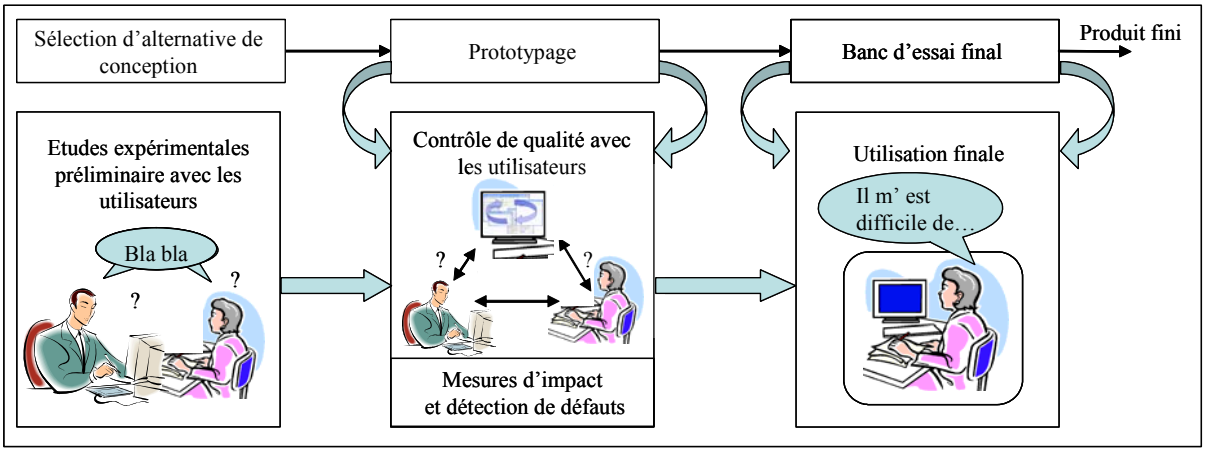



**Figure II.7.** Trois approches complémentaires pour les tests de conception (inspirées de [Senach 1990 ; Grislin 1995])

Selon [Senach, 1990], cette organisation séquentielle reste théorique étant donné que dans la réalité, du fait des contraintes liées au temps, budget, développement, etc., les approches sont souvent moins structurées et les tests ne sont pas systématiques.

[20] Le modèle en Spirale a été introduit par [Boehm, 1988]. Il constitue un processus itératif où chaque cycle de spirale développe quatre activités : l'identification des objectifs de la phase, les alternatives pour atteindre ces objectifs et leurs contraintes, l'analyse et la résolution des risques, le développement et la vérification de l'objet de la phase, et la planification de la phase suivante.

Les tests de conception regroupent trois démarches complémentaires :

- **la sélection d'alternative de conception** : elle est réalisée lorsqu'il n'existe aucun critère de choix évident entre plusieurs possibilités ; le recueil de données empiriques doit permettre de hiérarchiser les solutions envisagées. Cette méthode est généralement conduite en amont du prototypage [Senach, 1990].

- **le prototypage** : il permet l'évaluation précoce, avec les utilisateurs, de certains aspects particuliers de l'interface. Cette technique a pour objectif de minimiser les coûts de développement de l'interface tout en visant l'optimisation progressive de sa qualité selon l'état d'esprit introduite par le modèle en Spirale. Deux principaux critères sont considérés : les critères ergonomiques généraux comme ceux cités dans [Bastien et Scapin, 1993 ; Vanderdonckt, 1994] et les critères spécifiques à l'application (qualité, sécurité, environnement…).

- **le banc d'essai final** : il est utilisable une fois que l'interface est implémentée. Il permet ainsi de contrôler la qualité du produit final par une mesure d'utilisabilité globale du système. On recueille des données objectives et subjectives sur des sujets ayant à prendre en main le logiciel et/ou à l'utiliser, grâce à des stations d'évaluation spécialement aménagées (oculomètre, caméra, microphone, etc.).

**Avantages et inconvénients**

Les méthodes de tests de conception sont bien connues en génie logiciel et présentent plusieurs avantages. La méthode de sélection d'alternative assure l'hiérarchisation des solutions envisagées. Le prototypage permet de confronter les choix de conception à la réalité du terrain et de détecter ainsi précocement les problèmes. Le banc d'essai final permet à l'utilisateur de tester le système final et ainsi de détecter des nouveaux problèmes (ou les problèmes qui subsistent depuis la sélection d'alternative de conception).

Les méthodes de tests de conception possèdent aussi des limites. Par exemple : on reproche à la *sélection d'alternative de conception* qu'elle est souvent effectuée en dehors du contexte réel de la tâche, de ses contraintes, et de son environnement ; le *prototypage* ne permet pas de tester toute les fonctions en détail lorsqu'il est de « basse fidélité » par contre il limite la marge de manœuvre en terme de correction quand il est de « haute fidélité », le tout est de trouver un juste milieu selon le système à développer [Baccino *et al.*, 2005] ; le *banc d'essai final* est une approche exploratoire dans de nombreux cas et peut ne pas s'appuyer sur une véritable méthode [Grislin, 1995].

## III.1.4. Conclusion sur les approches centrées sur l'utilisateur et leur positionnement dans le cycle de développement

Les approches centrées sur l'utilisateur ne sont généralement applicables que lorsque le développement de l'interface est bien avancé. Elles se basent sur le principe que la meilleure façon de détecter les problèmes liés à un système est d'observer l'utilisateur réel qui interagit avec l'interface pour accomplir une tâche réelle. A ce propos [Karat, 1988] affirme que les tests auprès d'utilisateurs permettent dans la plupart des cas de détecter des erreurs de conception que le plus perspicace des développeurs n'a pas vues. Comme le précise [Farenc, 1997], les résultats et les données nécessaires à l'application de ces méthodes ne sont pas

réutilisables. Pour chaque système, chaque utilisateur (ou type d'utilisateur) et chaque tâche, la méthode doit être réalisée entièrement à nouveau.

Le développement d'un produit démarre rarement de rien et vise généralement l'amélioration d'une situation existante, déjà informatisée ou non. Dans ce cas, toutes ces techniques d'évaluation peuvent être envisagées *a priori* dès les étapes d'analyse et de spécification et introduire un état d'esprit d'évaluation, en visant à tirer les leçons des points forts et points sensibles du ou des sites de référence. Bien entendu, on les utilise aussi, le plus souvent d'ailleurs, *a posteriori*, c'est-à-dire après les phases d'implémentation, de tests et d'exploitation, lorsque l'IHM existe concrètement, et ceci dans le but de l'évaluer et de l'améliorer [Grislin et Kolski, 1996].

L'avantage majeur de ces méthodes est qu'elles prennent en compte la réalité de l'utilisation du système avec la tâche réelle et l'utilisateur réel. Cet aspect assure aux résultats une fiabilité non égalable par les méthodes dites théoriques (méthodes basées sur des experts § III.2, méthodes analytiques § III.3) [Senach, 1990 ; Balbo, 1994 ; Farenc, 1997 ; Bastien et Scapin, 2002].

Dans le cadre de notre objectif d'évaluation des systèmes interactifs orientés agents, et plus particulièrement dans un cadre de supervision de système complexe (Cf. Chapitre 5) les méthodes d'évaluation centrées sur l'utilisateur nous semblent être bien adaptées. Nous retiendrons en particulier les approches empiriques de diagnostic d'usage suivant :

- **le mouchard électronique** : qui est un outil d'inspection objectif permettant l'enregistrement des actions de l'opérateur ainsi que les réactions du système (Cf. § III.1.1.3). Notre idée est d'associer un agent mouchard à chaque agent de l'interface du système à évaluer. Nous détaillerons au chapitre trois le mouchard électronique que nous proposons afin d'enregistrer les actions et réactions de l'utilisateur et des agents d'interface.

- **la technique d'enregistrement et d'analyse des mouvements oculaires** : qui permettra l'analyse de l'activité d'un opérateur humain devant un écran ou un ensemble de supports d'information. Ces informations seront présentées à l'opérateur par des agents d'interface. Chaque agent d'interface occupera une zone de l'écran. L'oculomètre nous permettra ainsi de déterminer : la durée de visualisation de chaque zone, le nombre de basculement entre les différentes zones, l'enchaînement séquentiel de visualisation de chaque zone, la durée de fixation et la zone de fixation [Ezzedine et Abed, 1997].

- **le questionnaire** et la **verbalisation** permettront à l'opérateur de s'exprimer clairement et de donner un ensemble d'appréciations, d'opinions sur l'interface. Cette technique nous permettra surtout de déterminer le niveau de satisfaction de l'utilisateur qu'il est impossible de mesurer autrement qu'avec le questionnaire.

Lorsque l'interface n'existe pas ou lorsque les données et les mesures relatives à l'interaction homme-machine ne peuvent être enregistrées, l'évaluation peut s'effectuer par les approches basées sur des experts.

## III.2. Approches basées sur des experts

L'évaluation effectuée par un expert en ergonomie ou un spécialiste en communication homme-machine, permet de fournir un jugement sur la qualité ergonomique d'une interface, comparer les attributs et caractéristiques de l'IHM, par rapport aux recommandations ergonomiques ou normes dans le but de détecter les erreurs de conception et proposer des améliorations.

Les approches basées sur des experts sont utilisées lorsque les données relatives à l'interaction avec l'interface ne peuvent être enregistrées, ne sont pas disponibles ou lorsque l'interface n'existe pas. Elles peuvent être combinées avec d'autres approches centrées sur l'utilisateur. Parmi les nombreuses méthodes d'évaluation, basées sur des experts, qui existent, on peut citer l'intervention d'un spécialiste, les méthodes d'inspection (cognitive walkthrough, les directives de recommandations (guidelines), l'évaluation heuristique) et les grilles d'évaluation. Ces méthodes sont maintenant présentées.

### III.2.1. L'intervention d'un spécialiste

#### Objectif

L'objectif de cette méthode est de juger la qualité ergonomique de l'interface par un spécialiste en communication homme-machine et de proposer des améliorations.

#### Principe et description

Cette méthode est celle qui, sans doute, donne les meilleurs résultats, pourtant elle est encore sous-employée. L'évaluation par intervention d'un spécialiste consiste à effectuer une expertise de l'interface et de déterminer l'ensemble des problèmes susceptible d'être rencontrés lors de l'utilisation de l'interface. L'intervention de différents spécialistes pour l'expertise d'une interface montre que chacun se focalise sur des aspects particuliers de l'interface et fonde son évaluation sur une démarche qui lui est propre [Hammond *et al.*, 1984 ; Nielsen et Molich, 1990]. Par conséquent, l'idéal serait de faire intervenir plusieurs spécialistes (psychologues, cogniticiens, graphistes, etc.) l'ensemble des expertises étant ensuite confronté et synthétisé afin de déterminer les améliorations à apporter à l'interface (Cf. figure II.8).

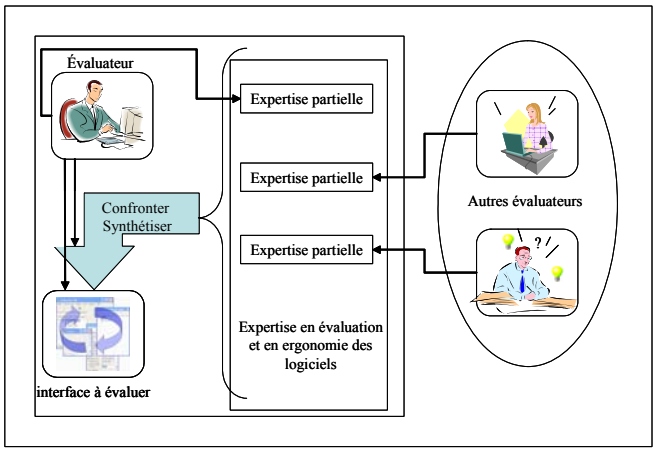



**Figure II.8.** Principe d'intervention des experts

### Avantage et Inconvénient

L'avantage de cette méthode est qu'elle est peu coûteuse [Bastien et Scapin, 2002] et facile à mettre en place pour un diagnostic rapide des erreurs de conception [Nendjo Ella, 1999]. L'inconvénient est que l'intervention d'un seul spécialiste des facteurs humain même expérimenté permet d'identifier en moyenne moins de la moitié des problèmes potentiels d'une interface [Pollier, 1991]. Des raisons budgétaires ou organisationnelles peuvent rendre impossible l'intervention de plusieurs évaluateurs.

## III.2.2. Les méthodes d'inspection de l'utilisabilité

### Objectif

Les méthodes d'inspection de l'utilisabilité regroupent un ensemble d'approches faisant appel au jugement d'évaluateurs, que ces derniers soient experts ou non en utilisabilité [Nielsen et Mack, 1994 ; Virzi, 1997]. Toutes ces méthodes visent généralement la détection des aspects des interfaces pouvant entraîner des difficultés d'utilisation ou alourdir le travail des utilisateurs.

### Principe et description

Les méthodes d'inspection[21] sont nombreuses et n'ont pas toutes le même objectif. Elles se distinguent les unes des autres par la manière dont les jugements des évaluateurs sont dérivés et par les critères d'évaluation à la base de leurs jugements. Selon [Bastien et Scapin, 2001 ; Abed, 2001] celles qui sont les plus documentées et qui ont fait l'objet de tests et de comparaisons sont :

- **la méthode Cognitive Walkthrough :** cette méthode permet d'évaluer la facilité d'apprentissage du système et de prédire les problèmes rencontrés par l'utilisateur [Lewis *et al.*, 1990 ; Jacobsen et John, 2000]. En se mettant à la place d'un utilisateur devant explorer l'interface en réalisant des tâches prédéfinies, l'évaluateur détermine à chaque étape de l'accomplissement d'une tâche les problèmes susceptibles d'être rencontrés par l'utilisateur ; l'interface doit donc être modifiée en conséquence en effectuant les améliorations nécessaires (décrite dans des fiches rédigées par l'évaluateur).

- **la conformité à des recommandations et à des dimensions ergonomiques (guides ergonomiques/guidelines) :** l'évaluation de la conformité à des recommandations et à des dimensions ergonomiques consiste à juger la conformité des éléments de l'interface par rapport à des règles disponibles dans différents documents prenant généralement la forme de recueils. On trouve ainsi des guides de style [Smith et Mosier, 1986], des guides ergonomiques [Bastien et Scapin, 1993 ; Vanderdonckt, 1994] ou encore des normes [ISO, 1996 ; Afnor, 2003]. Notons que des recueils ont été proposés plus spécifiquement pour des

[21] Des références sur les méthodes d'inspection de l'utilisabilité sont disponibles à l'adresse : http://mentalmodels.mitre.org/cog_eng/ce_references_III.htm.

domaines particuliers, par exemple le multimédia et le Web[22] [Bastien *et al.*, 1998 ; Vanderdonckt, 1998].

- **l'évaluation heuristique :** l'évaluation heuristique [Nielsen et Mack, 1994] consiste principalement à effectuer une revue de l'interface par un ou plusieurs évaluateurs. Elle a pour objectif de recenser les problèmes d'utilisabilité liés à l'interface. En s'appuyant sur une liste de critères heuristiques fondés sur l'expérience dans le domaine de l'évaluation axés sur l'utilisabilité et tout en tenant compte du contexte d'usage, les experts (évaluateurs) regroupent les problèmes rencontrés relativement à ces heuristiques. Il s'agit ensuite de donner des indications sur la criticité des problèmes rencontrés.

**Avantages et inconvénients**

La méthode Cognitive Walkthrough a été utilisée à des nombreuses reprises, d'abord avec des tâches relativement simples pour les systèmes interactifs [Polson *et al.*, 1992] et récemment pour l'évaluation des sites Web ; voir la méthode (Cognitive Walkthrough for the Web) [Blackmon et al, 2002]. Des tentatives d'automatisation de cette méthode ont été effectuées [Rieman *et al.*, 1991]. Notons que pour des tâches complexes, la méthode cognitive Walkthrough peut s'avérer lourde à utiliser.

L'inconvénient majeur des guides de recommandation réside généralement dans leur taille importante (ils peuvent contenir des milliers de règles), ainsi leur utilisation devient fastidieuse et complexe. Selon [Bastien et Scapin, 2001], lorsque on étudie ces guides, on peut constater une absence d'uniformité dans la présentation des recommandations ce qui rend la recherche d'une règle bien précise extrêmement difficile.

L'évaluation heuristique a déjà été testée et a fait ses preuves. A titre d'exemple, [Nielsen et Mack, 1994 ; Garzotto et Matesa, 1997] s'appuient sur l'évaluation heuristique pour proposer des évaluations de l'utilisabilité à moindre coût. L'inconvénient de cette méthode réside dans la difficulté de reproduction du contexte d'usage. En effet, les experts ne peuvent pas prévoir les actions de l'utilisateur qui peuvent parfois être inattendues. Ils peuvent se contenter de remarques visant à couvrir la majorité des cas, mais pas les cas particulier.

## III.2.3. Les grilles d'évaluation

**Objectif**

Les grilles d'évaluation permettent d'évaluer l'interface selon plusieurs critères ergonomiques. En utilisant des grilles d'évaluation, l'évaluateur note systématiquement l'interface selon une échelle comportant généralement de 3 à 5 points, parfois 10.

**Principe et description**

Une grille d'évaluation regroupe un ensemble de questions auxquelles l'évaluateur, passant en revue l'interface, doit répondre. Généralement la réponse à ces questions se fait via des cases à cocher ou des notes à attribuer à chaque aspect ergonomique à étudier. Une

[22] Cf. par exemple : le guide de conception de site Web d'IBM, disponible à l'adresse : http://www-3.ibm.com/ibm/ easy/eou_ext.nsf/Publish/572.

méthode d'évaluation par « check-list » a par exemple été mise au point par [Ravden et Johnson, 1989] (Cf. figure II.9) ; elle est utilisable par des concepteurs, des spécialistes des facteurs humains et des utilisateurs finaux du système.

| Répondre aux questions ci-après en mettant une croix dans la colonne correspondante | | | | | | |
|---|---|---|---|---|---|---|
| **N°** | **Questions** | **Toujours** | **Souvent** | **Quelquefois** | **Jamais** | **Ne sait pas** |
| **Prise en compte de la <u>cohérence</u>** | | | | | | |
| 1 | Les couleurs sont-elles utilisées de façon cohérente ? | X | | | | |
| 2 | Les abréviations, codes et autres informations sont-ils utilisés de façon cohérente ? | | X | | | |
| 3 | La représentation graphique, symboles, icônes et autres informations picturales sont-ils utilisés de façon cohérente ? | X | | | | |
| 4 | Le curseur apparaît-il dans la même position initiale pour des affichage similaires | | | | | X |
| . | … | | | | | |
| **Prise en compte de la <u>clarté</u> et de la <u>netteté</u>** | | | | | | |
| 12 | Les étapes que peut atteindre l'interface lors de l'exécution d'une tâche sont-elles bien comprises ? | X | | | | |
| 13 | L'utilisateur sait-il toujours où il est ? | | X | | | |
| 14 | Les opérations dans chaque partie de l'interface sont-elles bien comprises ? | | | X | | |
| . | … | | | | | |
| | Y a-t-il d'autres commentaires que vous désirez ajouter ? | Bla bla bla… | | | | |

**Figure II.9.** Extrait d'une grille d'évaluation (inspirée de [Ravden et Johnson, 1989])

### Avantages et inconvénients

C'est une méthode considérée comme efficace puisque si l'évaluateur dispose d'un ensemble de grilles types [Scapin, 1986 ; Ravden et Johnson, 1989] ainsi que de guides de critères ergonomiques [Nielsen, 1993 ; Vanderdonckt, 1994], il risque moins d'oublier des critères importants lors de l'évaluation. En revanche, bien que cette méthode soit un moyen satisfaisant d'obtenir des réponses cohérentes, [Senach, 1990] précise que ses limites sont évidentes étant donné qu'elle ne permet pas d'identifier l'organisation de l'activité de l'utilisateur ; aucune information n'est fournie sur la structure des procédures de travail.

## III.2.5. Conclusion sur les approches basées sur des experts et leur positionnement dans le cycle de développement

Nous avons présenté ci-dessus plusieurs approches d'évaluation basées sur des experts. Ces approches peuvent être utilisées dans des situations d'existence ou non du système à évaluer. Comme le précise [Grislin, 1995], l'intervention d'un spécialiste n'est envisageable qu'une fois que l'interface est réalisée, alors que les grilles d'évaluation et les méthodes d'inspection de l'utilisabilité sont utilisables en parallèle à la conception.

L'avantage de ces évaluations est qu'elles peuvent s'effectuer assez rapidement, elles sont relativement peu coûteuses et peuvent intervenir assez tôt dans le processus de conception [Nielsen et Mack, 1994]. Cependant, les études effectuées sur l'activité des experts en situation d'évaluation montrent que les performances individuelles sont très variables quant au nombre et aux types d'erreurs détectées et aux stratégies d'évaluation adoptées [Pollier, 1991 ; Bastien et Scapin, 2001].

## III.3. Approches Analytiques

Ces approches qualifiées aussi d'approches centrées sur une modélisation [Grislin et Kolski, 1996], se basent sur des modèles formels de l'interface et/ou de l'interaction homme-machine ainsi que sur la mise en œuvre progressive de métriques objectives. Les modèles sont utilisés pour prédire, au moyen des spécifications, certains aspects liés à l'interaction homme-machine tels que la hiérarchie des tâches, le cheminement des actions de l'utilisateur, le temps requis pour réaliser une tâche,... Les métriques sont mises en place pour mesurer de façon objective certains aspects associés à la qualité de l'interaction (cohérence, compatibilité avec l'image mentale de l'utilisateur, qualité des écrans,...). Le recours à ces représentations abstraites autorise des prédictions relatives aux performances qui ne peuvent être établies avec une approche empirique, lorsqu'il n'y a pas encore d'expérience d'utilisation du système interactif [Senach, 1990 ; Grislin, 1995].

Pour illustrer ces approches nous reprenons la classification de [Senach, 1990] qui consiste à faire la distinction entre deux classes de modèles : modèles formels prédictifs et modèles de qualité de l'interface. Ces deux modèles sont présentés ci-dessous.

## III.3.1. Les modèles formels prédictifs

Les modèles formels prédictifs prennent comme hypothèse que certaines performances de l'utilisateur peuvent être prédites, et donc considérées, lors de la phase de conception et d'évaluation de l'interface. On distingue deux types de modèles qui peuvent être utilisés comme support à l'évaluation : les modèles de tâches et les modèles linguistiques.

## III.3.1.1. Utilisation des modèles de tâche comme support à l'évaluation

### Objectif

Ces modèles ont pour objectifs d'analyser et de représenter l'activité d'un utilisateur sous la forme d'une structure de tâches. Ils fournissent ainsi un support formel pour l'évaluation prédictive de la performance en proposant une description mesurable du comportement de l'utilisateur et de ses performances.

### Principe et description

Les modèles de tâche consistent en une décomposition la plus souvent hiérarchique des tâches en unités élémentaires pour analyser et représenter l'activité cognitive de l'utilisateur.

Parmi les plus connues, on peut citer sans souci d'exhaustivité[23] les modèles de tâche suivants :

- **le modèle GOMS** : GOMS (Goals, Operators, Methods, and Selector) [Card *et al.*, 1983] décrit l'activité cognitive d'un utilisateur (expert) engagé dans la réalisation d'une tâche de routine sans erreur. Il se décompose en *Buts* (« Goals ») organisés en une hiérarchie, en Opérateur correspondant aux actions élémentaires (« Operators »), en *Méthodes* ou procédures de réalisation d'un but (« Methods ») et en *Règles de sélection* (« Selection rules ») des méthodes lorsque plusieurs solutions permettent d'atteindre un même but. Notons que le modèle CPM-GOMS [John et Kieras, 1996] propose une extension de GOMS sous la forme d'un calcul de chemin critique pour réaliser la tâche ; il décrit les tâches de l'utilisateur par une représentation sous forme de réseau.
- **le modèle Keystroke-Level-Model** : KLM [Card *et al.*, 1983] permet essentiellement de prédire le temps d'exécution des tâches utilisateurs. Une tâche est analysée au niveau de ses tâches-unités. Par exemple : saisir, déplacer, supprimer… Le temps d'exécution de chaque tâche-unité est estimé par addition des temps d'exécution des opérateurs primitifs qui le composent.

  Ce modèle contient six types d'opérateurs primitifs, et pour chacun de ces opérateurs, un temps d'exécution est estimé ; K : appuyer sur une touche, P : indiquer avec la souris une zone de l'écran, H : diriger la main vers un dispositif, D : dessiner un segment de ligne, M : se préparer mentalement à faire une action, R : temps de réponse du système pendant lequel l'utilisateur est en attente.
- **la méthode SADT-Petri** : développée par [Abed, 1990] au sein du LAMIH, cette méthode exploite les outils d'analyse et de modélisation SADT [IGL, 1989], pour la décomposition fonctionnelle du système et la modélisation statique des tâches, et les Réseaux de Petri [David et Alla, 1992], pour décrire la composante dynamique des tâches. Cette méthode a évolué vers TOOD (Task Oriented Object Design) en introduisant une description orientée objet des tâches [Mahfoudhi, 1997]. Récemment, [Tabary, 2001] et [Abed, 2001], ont introduit l'approche MBD[24] (Model Based user interface Design) dans la méthodologie TOOD, afin de faciliter la modélisation d'applications hautement interactives.

**Avantage et Inconvénient**

Chaque méthode ou modèle a ses avantages et ses inconvénients. Le modèle GOMS permet de prédire et simuler le comportement d'un opérateur expert mais il suppose que cet opérateur ne commet pas d'erreur, ce qui représente une grande limite à ce modèle.

Le modèle KLM a l'avantage d'être facilement compréhensible par les concepteurs ; par contre sa mise en œuvre pose rapidement de nombreux problèmes [Coutaz, 1990]. Par ailleurs, il ne permet pas la prédiction de l'efficacité de l'opérateur quant celui-ci effectue des tâches complexes.

[23] Il aurait par exemple été possible de rajouter DIANE [Barthet, 1988] et son extension DIANE+ [Tarby, 1993], HTA [Shepherd, 1993] et MAD [Scapin et Pierret-Golbreich, 1989].

[24] Les approches MBD (Model-Based user interface Design) se réfèrent dans l'absolu, à une description explicite, largement déclarative, capturent la sémantique de l'application et toutes les connaissances nécessaires à la spécification tant de l'apparence que du comportement du système interactif [Tabary et al, 2003].

La méthode SADT-Petri à l'avantage de pouvoir exploité les RdP pour la confrontation du modèle de la tâche à réalisée (modèle de tâche théorique) et de la tâche observé (tâche réellement effectuée) afin de reboucler sur la spécification de l'interface.

## III.3.1.2. Utilisation des modèles linguistiques comme support à l'évaluation

### Objectif

Ces modèles tentent de produire, d'une part une représentation explicite de la structure de l'interface, et d'autre part les actions susceptibles d'être effectuées par les utilisateurs finaux de l'IHM, au moyen de grammaires.

### Principe et description

Les modèles linguistiques explicitent la structure de l'interaction Homme-Machine au moyen de grammaire. Ces modèles sont utilisables pour décrire les tâches d'interaction entre l'utilisateur et l'interface, dans le but principalement d'évaluation du langage de commande et de sa cohérence [Grislin et Kolski, 1996]. Parmi ces modèles, nous citons à titre présentatif les modèles suivants :

- **le modèle CLG** (Command Language Grammar) [Moran, 1981] : c'est une structure grammaticale qui permet de représenter un système par quatre niveaux d'abstraction : les niveaux tâches, sémantique, syntaxique et interaction. D'après Moran, le modèle CLG peut contribuer à assurer une évaluation prédictive de la facilité d'utilisation de l'interface et de localiser, selon [Coutaz, 1990], les décisions d'évaluation qui seront critiques pour l'utilisateur.

- **le modèle ALG** (Action Language Grammar) [Reisner, 1984] : il consiste à construire un modèle des actions mises en jeu par l'utilisateur, sous la forme de règles de production. Ces règles sont du type : *POUR effectuer-telle-action FAIRE telles-opérations*. Lors de l'évaluation le modèle ALG permet :
  - d'identifier des choix de conception susceptibles d'engendrer des erreurs d'utilisation et permettre ainsi de reboucler sur la spécification,
  - de comparer les alternatives de comparaison selon un critère de facilité d'utilisation.

- **le modèle TAG** (Task Action Grammar) [Payne et Green, 1989] : TAG est une grammaire formelle qui permet de décrire l'ensemble des tâches et actions sous forme de règles et de mettre en correspondance la sémantique des tâches avec des séquences d'actions. L'utilisation de TAG se heurte à plusieurs limitations, en particulier TAG ne considère que des tâches élémentaires et ne modélise pas les systèmes interactifs. Pour remédier à ce type de problème [Tauber, 1990] a introduit une extension de TAG : ETAG (Extended Task-Action Grammar) et fournit un modèle conceptuel du système interactif [De haan, 2001].

### Avantages et inconvénients

L'inconvénient majeur du modèle CLG est qu'il ne permet pas l'évaluation d'un produit fini développé avec un autre modèle ; il faudrait traduire l'interface en CLG pour en faire l'analyse.

Même si le modèle ALG présente un intérêt pour la construction d'un modèle des actions mises en jeu par l'utilisateur, il ne prend pas en compte le fait que des tâches faciles à décrire peuvent être difficiles à réaliser [Senach, 1990].

TAG aide à juger la structure de tâches décrites à l'aide d'un langage, mais celui-ci se limite aux tâches simples, c'est-à-dire à la description des fonctions spécifiques du dispositif. Il est à remarquer qu'il n'y a pas de mise en relation des différentes tâches simples à des tâches aux niveaux supérieurs. Pourtant pour évaluer les qualités ergonomiques d'une interface, il est nécessaire d'observer les informations visualisées en se basant sur la description de modèles conceptuels de l'interface [Ezzedine, 2002].

## III.3.2. Modèles formels dits de qualité de l'IHM

### Objectif

Les modèles formels de qualité de l'interface sont complémentaires des modèles formels dits prédictifs décrits en § II.3.1. Ils permettent d'identifier les propriétés mesurables de l'IHM, selon des critères d'utilisabilité formalisés, qui auront un effet sur la performance de l'utilisateur, notamment en limitant les difficultés d'utilisation.

### Principe et description

De telles approches partent du principe que certains problèmes ergonomiques liés à l'utilisation de l'interface peuvent être évités *a priori*. Mais elles doivent s'intégrer dans une démarche globale d'évaluation selon un ensemble de critères interconnectés dont il s'agit de gérer la cohérence par rapport aux différents contextes opérationnels d'utilisation [Grislin, 1995].

La classification de [Senach, 1990] distingue deux approches :

- **une approche cognitive de la qualité de l'interface** : cette approche prend en compte le traitement cognitif effectué sur l'information. En d'autres termes la qualité de l'interface dépend de sa comptabilité avec les représentations mentales élaborées par ses utilisateurs.
- **une approche optimale de la qualité du système** : cette approche ne considère pas les aspects sémantiques et évalue l'interface selon des critères le plus souvent quantitatifs liés à la présentation de l'information.

[Grislin, 1995] classifie les méthodes formelles dites de qualité en deux catégories :

- **Les systèmes d'évaluation automatique de l'affichage** : ils consistent en la mise en œuvre d'outils informatiques capables d'effectuer automatiquement des mesures ergonomiques sur des pages-écrans, indépendamment du contexte d'utilisation. Pour l'évaluation des systèmes interactifs ces approches essayent d'éviter les problèmes ergonomiques liés à l'utilisation de l'interface avant leur apparition en s'intégrant dans la démarche de conception.

  Pour des raisons de clarté et de structuration de ce chapitre, nous détaillerons plus loin ces approches ainsi que leurs avantages et inconvénients, dédiées à l'évaluation automatique des interfaces homme-machine (Cf. § IV).

- **Les systèmes de génération automatique de l'affichage** : ils visent, comme leurs noms l'indiquent, la génération automatique de l'IHM ou la génération de spécifications de l'IHM. Ils ont pour objectifs de conduire à des affichages respectant *a priori* des concepts ergonomiques de base.

### III.3.3 Conclusion sur les approches analytiques et leur positionnement dans le cycle de développement

Comme nous l'avons vu, les méthodes d'évaluation analytiques sont généralement utilisées pour prédire au moyen de spécifications, certains aspects liés à l'interaction homme-machine (hiérarchie des tâches, cheminement des actions de l'utilisateur, temps requis pour réaliser une tâche,...).

Les modèles prédictifs énumérés ci-dessus, basés sur des modèles de tâches ou des modèles linguistiques, décrivent l'interaction homme-machine avec une visée d'évaluation prédictive en particulier de performance, puisqu'ils proposent une description mesurable du comportement de l'utilisateur. Le modèle obtenu est ainsi utilisé pour l'évaluation ergonomique en analysant la représentation des activités probables des utilisateurs lors de l'exécution de leurs tâches. Comme le souligne [Kolski, 1997], on oublie très souvent que de tels modèles peuvent en plus être utilisés une fois l'interface réalisée pour modéliser les activités des utilisateurs et comparer cette modélisation à celle effectuée vis-à-vis des tâches prévues par les concepteur, afin d'en tirer les conséquences en termes d'aménagement des IHMs et /ou de formations des utilisateurs.

Quant aux modèles formels dit de qualité de l'IHM, nous verrons plus loin (Cf. § IV) qu'il faux considérer que la plupart de ces systèmes sont à l'état de prototypes de recherche, même s'ils ont été validés sur des cas concrets. De tels systèmes devraient être à terme intégrés dans des environnements graphiques de développement. Notons tout de même que pour l'évaluation des sites Web, certains outils d'évaluation automatique ou d'aide à l'évaluation existent déjà [Beirekdar, 2004 ; Mariage, 2005].

## III.4 Synthèse sur les méthodes, techniques et outils d'évaluation

Il résulte de la présentation que nous avons effectuée ci-dessus qu'une grande variété de méthodes contribuant à l'évaluation des interfaces homme-machine existe. Celles-ci sont issues plus ou moins directement de domaines variés, tel que le génie logiciel, l'ergonomie, la psychologie, l'acquisition de connaissance ou l'automatique humaine ; elles contribuent de manière générale au domaine vaste et pluridisciplinaire que constitue l'interaction homme-machine.

Nous avons présenté quelques classifications existantes des méthodes et techniques d'évaluation dédiées aux systèmes interactifs et particulièrement aux interfaces homme-machine. Nous avons constaté que plusieurs axes de classification des méthodes d'évaluation sont possibles. Notre objectif n'étant pas de critiquer et d'analyser les classifications existantes en vu d'en proposer une nouvelle, dans la mesure où on retrouve explicitement ou implicitement dans la plupart des classifications existantes, les trois approches proposées par [Grislin et Kolski, 1996], nous avons sélectionné cette classification en vu de présenter un panorama des méthodes et techniques d'évaluation existantes :

- les approches centrées sur l'utilisateur consistent à recueillir des données représentatives de l'interaction à l'aide des mesures ou d'observations provenant de l'utilisation de l'interface par des utilisateurs représentatifs de la population finale. L'évaluation doit se faire dans un contexte le plus proche possible de la réalité.

- les approches basées sur l'interaction d'experts sont utilisées lorsque les données relatives aux interactions homme-machine ne peuvent être enregistrées, ne sont pas disponibles ou lorsque l'interface n'existe pas.

- les approches analytiques centrées sur la modélisation de l'interface et/ou de l'interaction homme-machine sont envisagées lorsque l'interface est inexistante ; que l'utilisateur n'est pas disponible et/ou lorsqu'il est possible de procéder à une automatisation de l'évaluation ou d'une partie de celle-ci. Dans ce cas, le recours à des représentations abstraites autorise des prédictions relatives aux performances qui ne peuvent être établies avec une approche empirique, lorsqu'il n'y a pas encore d'expérience d'utilisation du système interactif [Senach, 1990].

Le point commun de toutes les méthodes et techniques d'évaluation demeure la problématique de choix de la méthode la plus appropriée pour l'évaluation d'un système donnée. Il est vrai que le choix d'une méthode dépend, d'une part de la capacité de vérification de critères d'évaluation que peut fournir la méthode, et d'autre part des facteurs contextuels qui sont en fait les contraintes qui se présentent aux évaluateurs, tel le budget disponible, le type de l'application, le temps disponible, etc. Par ailleurs, lors de l'évaluation, les méthodes et techniques utilisées ainsi que les données recueillies sont généralement nombreuses et nécessitent parfois un temps non négligeable de traitement pour tirer des conclusions sur la qualité de l'interface, surtout lors de l'évaluation de système complexe. Dans ce cas, l'évaluateur peut se trouver « perdu » si aucune aide ne lui est proposée faute des mauvais choix en termes d'évaluation.

Ces problèmes trouvent aujourd'hui un certain nombre d'éléments de réponse dans des systèmes d'aide à l'évaluation. Nous entendons par système d'aide à l'évaluation : (1) les systèmes d'aide au choix de la/les méthode(s) d'évaluation, (2) les systèmes d'aide à l'évaluation qui proposent une certaine assistance et accompagnement à l'évaluateur, c'est-à-dire en aidant l'évaluateur à structurer et organiser son évaluation du système en question. La section suivante est consacrée à ces outils et systèmes d'aide à d'évaluation.

## IV. Vers des outils d'aide à l'évaluation automatique

Comme nous l'avons vu dans la section précédente, la qualité de l'évaluation des systèmes interactifs dépend de plusieurs paramètres. En effet, en plus des problèmes de procédure, de traitement des données recueillies et d'organisation (sur lesquels nous reviendrons plus loin, Cf. IV.2), que peuvent rencontrer les évaluateurs, le premier problème auquel ils sont confrontés est celui du choix des méthodes et techniques appropriées pour l'évaluation d'un système interactif donné [Nendjo Ella *et al.*, 1999 ; Baccino *et al.*, 2005]. Ainsi, pour les aider dans le processus d'évaluation, il existe plusieurs types d'outils dont les niveaux d'aide sont très variables.

## IV.1. Les outils d'aide aux choix de la méthode d'évaluation

Le choix de méthodes ou techniques appropriées est un axe important dans le domaine de l'évaluation des systèmes interactifs. En effet ce choix est crucial car le résultat final en dépend ; à cet effet l'évaluation qui laisserait le choix de la méthode ou technique d'évaluation au sort de l'évaluateur passerait à côté de la tâche la plus complexe en dehors de l'analyse et du traitement des résultats. Plusieurs travaux ont tenté d'apporter des solutions à ce problème [Stanton et Baber 1997 ; Denley et Long, 1997 ; Dix *et al.*, 1998 ; Nendjo Ella, 1999].

A ce sujet, [Nendjo Ella, 1999] propose un système d'aide à la décision (ADHESION : Aide à la Décision Humaine pour l'Evaluation des Systèmes Interactifs et leur validatiON) pour le choix des méthodes et techniques d'évaluation dont le principe est visible en figure II.10.

Cette approche se base sur les sous-ensembles flous. Les utilisateurs d'ADHESION, qui peuvent être des chefs de projets ou des évaluateurs expérimentés ou non, introduisent les données relatives à la situation d'évaluation via un questionnaire proposé par le système. Le système propose alors une sélection de méthodes d'évaluation en principe adaptées à la requête de l'utilisateur. En cas de non satisfaction, des alternatives sont proposées (augmenter les moyens financiers, acquérir du matériel spécialisé,…) afin d'obtenir d'autres propositions de méthodes.

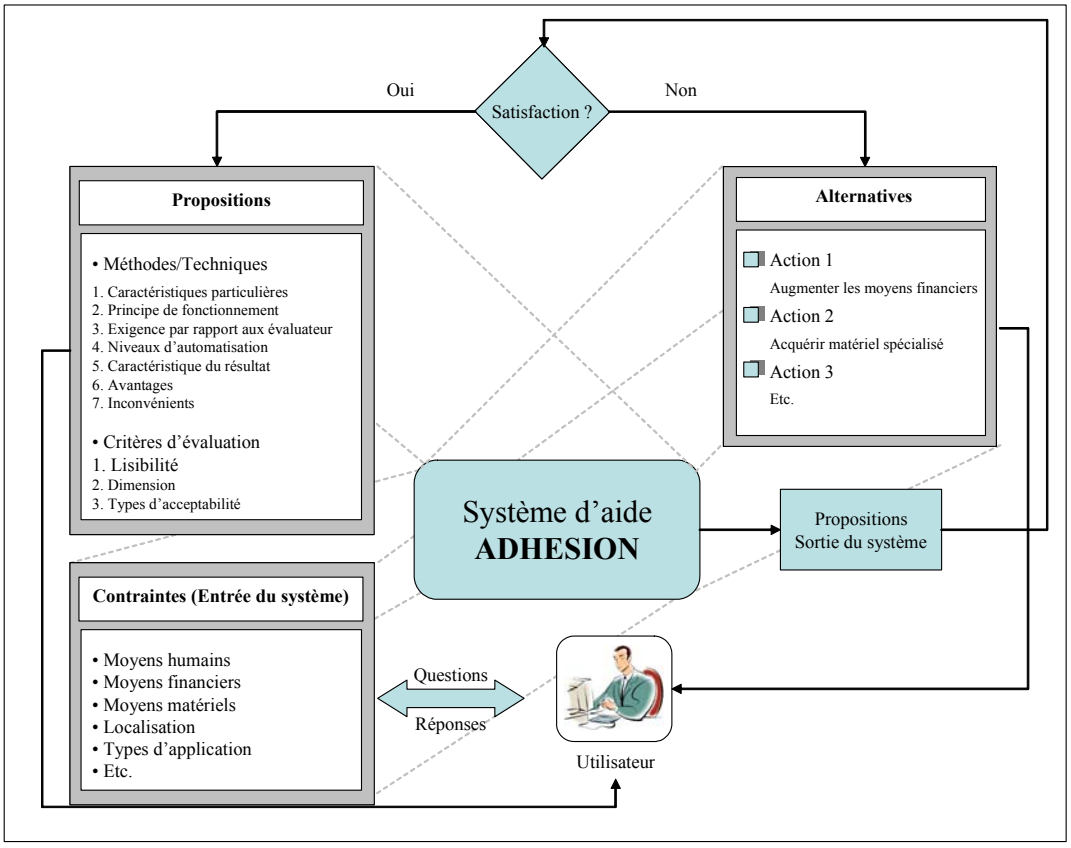



**Figure II.10.** Principe général d'utilisation du système d'aide à la décision ADHESION [Nendjo Ella, 1999]

[Stanton et Baber, 1996] tentent de répondre à la question de choix de méthodes que se pose l'évaluateur avant de commencer son évaluation en proposant quatre facteurs pour la prise de décision : l'étape dans le cycle de développement, la forme actuelle du système à évaluer (prototype, conception papier), la présence des utilisateurs finaux, le temps à accorder à l'évaluation. La détermination de ces facteurs permet à l'évaluateur de déterminer la ou les méthodes d'évaluation à utiliser. [Dix *et al.*, 1998] quand à lui, propose huit critères pour aider l'évaluateur dans son choix tout en classifiant les méthodes d'évaluation en deux catégories : les méthodes analytiques et les méthodes empiriques. L'outil d'aide à l'évaluation de [Denley et Long, 1997] est surtout destiné aux évaluateurs novices. Il fournit une structure et une démarche pour le choix des techniques en évaluation des systèmes interactifs en termes de facteurs humains. Il offre aussi, des informations relatives à la planification de l'évaluation sous forme d'heuristiques.

Parallèlement aux recherches menées sur les choix de la technique et méthode d'évaluation, d'autres axes de recherches s'intéressent à l'automatisation de l'évaluation afin de réduire le coût, le temps et l'effort de dépouillement des données de l'évaluation. Plusieurs outils représentatifs sont maintenant présentés.

## IV.2. Types d'outils automatiques et principes de base

L'automatisation de l'évaluation des systèmes interactifs a débuté au début des années 80 avec le système Playback développé dans le service facteur « humain » des laboratoires IBM [Neal, 1983]. Ce système était simple et permettait la capture des actions clavier au moyen d'un dispositif physique situé entre le clavier et la machine. En mode enregistrement, le dispositif détecte, transmet, date et sauvegarde sur disque toutes les actions de l'utilisateur sur le clavier. Le fichier contenant les sauvegardes alimente un analyseur statistique pour, par exemple, détecter le nombre de sollicitations de la touche « aide » ou « détruire ».

Depuis, divers outils logiciels d'aide à l'évaluation ont été proposés. Selon [Bastien et Scapin, 2002 ; Ivory et Hearst, 2001], il existe trois types d'outils d'aide à l'évaluation :

- les outils constituant une version logicielle des documents papiers : par exemple la version logicielle de la méthode cognitive Walkthrough [Rieman *et al.*, 1991] (Cf. II.2.2) permet d'automatiser des nombreuses saisies d'informations. Dans la méthode cognitive Walkthrough la tâche de l'évaluateur consiste à remplir des formulaires sur les actions des utilisateurs. Cette tâche devient souvent lourde avec des saisies répétitives d'informations souvent identiques. C'est essentiellement pour réduire cette charge de travail que l'outil a été développé sous forme de pile HyperCard pour plateforme Macintosh.

- Les outils d'accompagnement de l'évaluation : ceux-ci aident l'évaluateur à structurer et organiser l'évaluation. On retrouve dans cette catégorie les outils d'aide au choix de la méthode d'évaluation présentés ci-dessus (Cf. IV.1) ainsi que des outils d'organisation de l'évaluation comme par exemple l'outil FACE (Fast Audit base on Cognitive Ergonomics) [Hulzebosch et Jameson, 1996]. FACE permet à l'évaluateur de cerner la situation de l'évaluation, de choisir la technique d'évaluation qui sera appliquée et de présenter les résultats.

- Les outils d'évaluation automatique proprement dits : ces outils permettent l'automatisation de l'évaluation selon trois approches :

- la capture automatique : certains outils enregistrent différents types d'informations audiovisuelles, tels que la frappe au clavier, la manipulation de la souris, la parole, etc.

- l'analyse automatique des données capturées dans un but de détection de problèmes d'utilisabilité,

- la critique automatique permet de localiser certains problèmes rencontrés et de fournir des propositions d'amélioration et des recommandations.

L'automatisation de l'évaluation présente plusieurs avantages [Ivory et Hearst, 2001]. En effet, l'automatisation de l'évaluation permet de :

- réduire le coût de l'évaluation : les méthodes qui automatisent la capture, l'analyse, ou la critique (proposition de commentaires et d'alternatives) peuvent réduire le temps de l'évaluation et par conséquent le coût. Par exemple, les outils logiciels qui enregistrent automatiquement les événements (utilisateur/système) pendant l'évaluation éliminent le besoin de notation manuelle, qui nécessite un temps considérable ;

- faciliter la détection des erreurs : dans certains cas dans la mesure où l'interface peut être spécifiée selon des modèles de tâche (par exemple avec CTT [Paterno *et al.*, 1997] ; cf. aussi III.3.1.1), il devient possible de détecter automatiquement des incohérences et des erreurs, au niveau de l'interface à évaluer, par rapport à cette spécification ;

- réduire le besoin de recours à des experts en évaluation : automatiser quelques aspects d'évaluation, tels que les activités d'analyse ou de critique, apporte en principe une aide première aux concepteurs qui n'ont pas d'expérience approfondie en évaluation ;

- augmenter le recouvrement des aspects évalués : en raison du manque de temps, du coût élevé, et des contraintes en termes de ressources disponibles, il n'est pas toujours possible d'évaluer indépendamment chaque aspect de l'interface. Ainsi, avec le recours à des outils logiciels d'évaluation automatique, il peut devenir possible de couvrir la totalité ou quasi-totalité des aspects de l'interface à évaluer.

Parmi les outils automatiques d'aide à l'évaluation les plus connus, on peut citer sans souci d'exhaustivité[25] les outils suivants : SYNOP, KRI/AG, ERGOVAL, CHIMES, ÉMA. Ils sont décrits brièvement ci-dessous.

[25] Il aurait été possible de rajouter d'autre outils très intéressants, tels Hawk [Guzdial, 1993] ; DRUM [Macleod et Rengger, 1993] ; Sherlock [Mahajan et Shneiderman, 1995] ; Sherlock [Grammenos *et al.*, 2000] ; EDEM [Hilbert et Redmiles, 1998] ; KALDI [Al-Qaimari et McRostie, 1999] ; GUIDE [Henninger, 2000] ainsi que les outils dédiés à l'évaluation automatique des sites Web, tels DESTINE [Mariage, 2005] ; les travaux de [Beirekdar, 2004] ; KWARESMI (Cf. http://www.isys.ucl.ac.be/bchi/research/Kwaresmi.htm) ; WebRemUSINE [Paganelli et Paternò, 2002] ; WebTango (Cf. http://webtango.berkeley.edu). Dans un article très complet, [Ivory et Hearst, 2001] effectuent un état de l'art sur les méthodes d'évaluation automatique. Le lecteur intéressé peut également consulter la thèse de Balbo [Balbo, 1994].

## IV.2.2 SYNOP

### Objectif

Le système expert SYNOP développé au Laboratoire d'Automatique Industrielle et Humaine de Valenciennes [Kolski, 1989 ; Kolski et Millot, 1991], a pour objectif d'évaluer et de corriger des ébauches de vues graphiques de supervision selon des règles ergonomiques automatisées dans des bases de connaissance. Il contribue ainsi à l'évaluation et l'amélioration automatique de synoptiques industriels.

### Principe et description

SYNOP est écrit en langage LISP et intègre, dans sa base, des connaissances ergonomiques statiques de présentation d'information. Ces connaissances sont formalisées en règles de production exploitables par un moteur d'inférence. Elles permettent d'évaluer et de modifier automatiquement les vues graphiques. Les vues sont créées avec le progiciel de création de synoptiques industriels IMAGIN qui génère automatiquement des fichiers binaires au format imposé par ce progiciel. Ces fichiers sont ensuite traités par l'outil SYNOP, via une interface spécifique, pour récupérer la description de l'interface. Ensuite commence la phase de l'évaluation proprement dite. La chaîne logicielle dans laquelle s'intègre SYNOP est représentée schématiquement à la figure II.11.

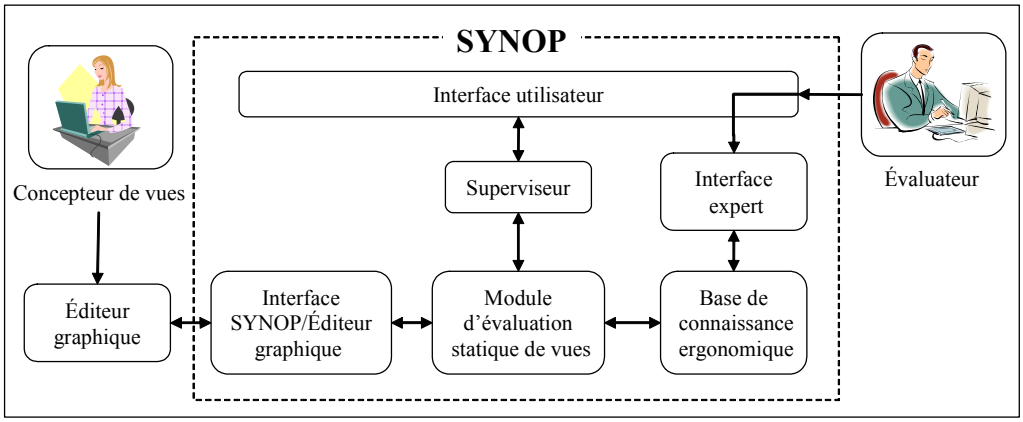



**Figure II.11.** Chaîne logiciel dans SYNOP

Le principe d'évaluation et amélioration ergonomique d'une vue graphique par SYNOP peut être décrit schématiquement par trois phase : (1) interprétation des fichiers graphiques dans le but de traiter et créer un réseau sémantique d'objets structurés LISP; (2) évaluation ergonomique des objets du réseau à l'aide de règles ; (3) reconstruction des fichiers graphiques correspondant à la vue améliorée.

### Avantage et Inconvénient

Le système expert SYNOP est le premier système d'évaluation automatique qui a proposé une évaluation dédiée aux applications de type systèmes de contrôle. Cependant il n'évalue que les vues statiques et exclue totalement l'utilisateur et les remarques constructives qui découlent de l'observation de ce dernier.

## IV.2.3. KRI/AG

### Objectif

KRI/AG (Knowledge-based Review of user Interfaces) [Lowgren et Nordqvist, 1992] est un système à base de connaissance capable d'évaluer automatiquement une interface à base de formulaires ou menus en fournissant des commentaires et suggestions de modification (il n'y a pas de modification automatique au contraire de SYNOP).

### Principe et description

KRI/AG est un système à base de connaissance connecté directement à l'UIMS (User Interface Management System) TeleUse[26] dans l'environnement Xwindows. Il a pour objectif d'évaluer les fichiers générés par cet UIMS. Il est également capable d'évaluer toute interface réalisée avec un UIMS qui génère des fichiers au format UIL[27]. KRI/AG s'appuie sur une base d'environ cent règles ergonomiques et règles de style qui, appliquées à la description formelle d'une interface, produisent des commentaires et suggestions de modification (Cf. figure II.12).

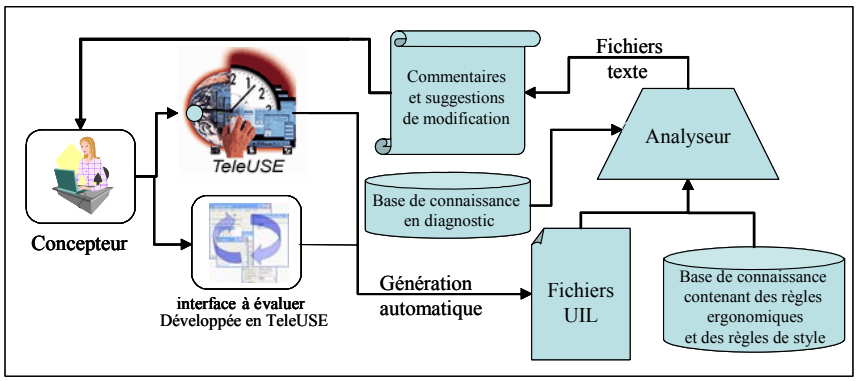



**Figure II.12.** Principe de fonctionnement de KRI/AG

KRI/AG est organisé en trois modules :

- une base de connaissance modélise le savoir du système en évaluation ergonomique sous forme de règles. Ce savoir est issu de guides ergonomiques et guides de styles (Cf. § III.2.2). Notons que, comme dans SYNOP, seuls les niveaux lexical et syntaxique sont considérés.

- une base de connaissance en diagnostic des IHM contient des règles sous forme textuelle. Ces règles sont dépouillées de toute interprétation et servent de base pour la génération des commentaires et recommandations.

- une classification des techniques d'interaction permet au système de structurer sa base de connaissance en fonction du type d'interface évaluée. Le concepteur utilise cette base pour orienter l'évaluation selon la nature de l'interface à évaluer. Notons

[26] TeleUse est un UIMS de AONIX fournisseur global des solutions pour des applications industrielles : http://www.aonix.com/teleuse.html

[27] UIL (User Interface Langage) est un langage qui assure la séparation entre l'interface et l'application. Ce langage est une description textuelle de l'interface utilisateur qui est compilée sous une forme binaire appelée UID (User Interface Definition). Notons que UIL ne prend en compte que la description des interfaces statiques.

que l'évaluateur peut consulter la (les) règle(s) ergonomique à l'origine du (des) commentaire(s) proposé.

**Avantages et inconvénients**

Dans la même lignée que SYNOP, KRI/AG adopte une structuration de la base de connaissance en différents thèmes. Mais, à sa différence, KRI/AG ne prolonge pas la critique, non plus en conseils mais en mise en oeuvre de corrections [Calvary, 1998]. Cependant, comme SYNOP l'évaluation est effectuée hors du contexte de travail.

## IV.2.4. ERGOVAL

**Objectif**

ERGOVAL [Liberati *et al.*, 1995 ; Farenc, 1997] est un outil d'évaluation qui a pour objectif comme les deux précédents de mesurer la qualité ergonomique des interfaces en vue de définir quelles doivent être les modifications à réaliser pour améliorer celles-ci. Cette méthode correspond à la vérification sur la présentation d'une interface du respect d'un ensemble de règles ergonomiques provenant de guides de recommandations (Cf. § III.2.2).

**Principe et description**

Comme les outils précédents, ERGOVAL traite de la présentation statique, aux niveaux lexical et syntaxique. Pour ce faire, il implique trois sous-systèmes :

- une base de connaissance contenant les règles ergonomiques issues de la littérature ;
- la décomposition structurelle des objets de l'interface ;
- une typologie des objets graphiques permettant de les regrouper afin de leur attribuer les règles correspondantes.

Dans la décomposition structurelle d'une fenêtre secondaire établie à partir de la norme CUA (Common User Access)[28] visible à la figure II.13, les objets graphiques sont reliés entre eux par une relation de composition. Cependant, pour mieux structurer la connaissance ergonomique, cette relation est affinée en quatre types :

- la relation d'activation. Exemple : il peut exister un lien entre une icône et un menu système ; en cliquant sur celle-ci, le menu système apparaît ;
- la relation d'agrégation (au sens objet). Exemple : un lien peut exister entre un menu système et une option de menu système. Un menu système est constitué d'options relatives aux menus. Un menu système n'existe pas sans ses options ;
- la relation de position géographique (aussi appelée « se trouve dans »). Exemple : lien entre bureau et fenêtre principale. La fenêtre principale s'affiche dans la fenêtre du bureau. Cette relation ressemble fortement à la relation d'association (au sens objet) ;

[28] LA norme CUA (Common User Access) a été développée par IBM en 1987.

- la relation d'enchaînement. A titre d'exemple on peut citer le lien entre fenêtre principale et fenêtre secondaire. Une fenêtre secondaire peut s'ouvrir seulement à partir d'une fenêtre principale.

Notons que si ERGOVAL est un outil d'évaluation, c'est aussi, et avant tout, une méthode de structuration des règles ergonomiques, ce qui différencie ERGOVAL de SYNOP et KRI/AG.

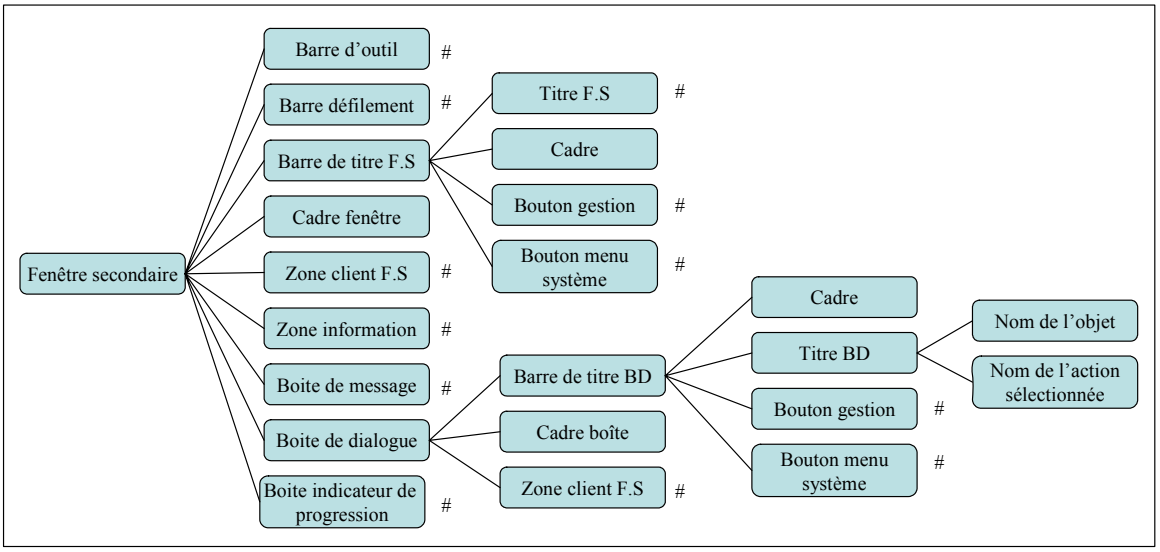



**Figure II.13.** Extrait de la décomposition structurelle [Liberati *et al.*, 1995]

(Le caractère # signifie que l'objet est encore décomposable)

### Avantages et inconvénients

Grâce à ses principes de structuration d'objets, cette méthode a pour avantage d'être automatisable et d'être efficace pour évaluer la présentation statique d'une interface concernant des aspects indépendants de la tâche. L'inconvénient majeur de cette méthode est qu'elle ne tient pas compte de la dynamique de l'interface et des aspects liés à la tâche de l'utilisateur.

## IV.2.5. CHIMES

### Objectif

CHIMES (Computer Human Interaction Models) [Jiang *et al.*, 1992] est un système expert développé au centre spatial de la NASA. L'objectif de cet outil est d'évaluer automatiquement une interface développée sous OSF/MOTIF[29] sous l'angle de la conformité à des recommandations provenant du guide de style MOTIF. Il cible, plus particulièrement, les recommandations relatives à l'usage de la couleur.

[29] MOTIF a été développé par OSF (Open Software Foundation). Le nom complet de MOTIF est en fait OSF/MOTIF.

**Principe et description**

CHIMES est composé de cinq modules principaux : un module d'acquisition des données, un vérificateur de conformité, une base de connaissance, un générateur de conseils, et une interface homme-machine :

- le module d'acquisition des données récupère la description de l'interface et la transmet, sous une forme exploitable, au vérificateur de conformité ;
- le vérificateur de conformité récupère les données en provenance du module d'acquisition des données et les convertit sous un format exploitable par la base de connaissance ;
- la base de connaissance contient des règles ergonomiques. Ces dernières portent sur les niveaux lexical et syntaxique ;
- le générateur de conseils est responsable de la présentation à l'évaluateur des résultats de l'évaluation, sous forme d'une liste de conseils ;
- L'interface permet à l'évaluateur d'effectuer l'évaluation et de récupérer les résultats sous forme d'une liste de conseils.

**Avantages et inconvénients**

CHIMES est un outil d'évaluation dédié spécialement aux interfaces critiques et à hauts risques en aviation. Son inconvénient majeur est qu'il est rattaché à la station de travail SUN et s'exécute seulement sur le système d'exploitation SunOS avec MOTIF.

## IV.2.6. ÉMA

**Objectif**

ÉMA (Évaluation par Mécanisme Automatique) est un outil d'évaluation qui a pour objectif de produire automatiquement une évaluation ergonomique à partir d'une capture de données comportementales numérisées [Balbo, 1994].

**Principe et description**

ÉMA utilise des graphes de séquences, c'est-à-dire des graphes orientés, dont les noeuds sont des tâches élémentaires et les arcs des relations d'enchaînement entre tâches. Ces graphes traduisent la logique de fonctionnement du système et les tâches dont il est ici question sont en fait les tâches telles qu'elles sont implantées dans l'application. En d'autres termes, les graphes de séquences modélisent les tâches prévues pour une application donnée. C'est l'accomplissement des tâches utilisateurs par le biais des tâches prévues dans l'application, plus particulièrement les événements utilisateurs, que capture ÉMA. A partir des traces d'utilisation et des graphes de séquences, une détection automatique d'anomalies est effectuée.

La figure II.14 montre l'enchaînement des différentes étapes d'ÉMA.

Dans ÉMA l'analyse des tâches utilise trois sources d'information :

- une base de profils de comportements modélisés sous forme de règles de détection d'anomalie. Les profils sont identifiés empiriquement en étudiant des fichiers de capture et en analysant des expériences faites au laboratoire. Les règles de détection d'anomalie sont au nombre de quatre : rupture, annulation immédiate, répétition, et invalidité ;

- une représentation formelle de l'espace des tâches réalisables avec le logiciel testé. Cette représentation formelle est sous forme de graphes orientés dont les nœuds sont des tâches et les arcs des relations d'enchaînement entre tâches

- les données comportementales enregistrées de manière automatique au cours des sessions d'utilisation du logiciel testé. Le recueil s'effectue sur support audiovisuel, et/ou par capture automatique d'actions ou encore par interviews et questionnaires.

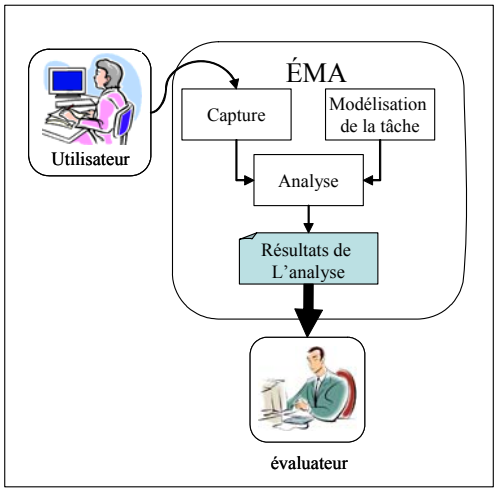



**Figure II.14.** Fonctionnement d'ÉMA [Balbo, 1994]

Alors que la base de profils est réutilisable pour l'évaluation de différents systèmes, l'espace des tâches est propre à chaque logiciel et les données comportementales dépendent avant tout des sujets observés et des scénarios expérimentaux. L'algorithme d'analyse recherche des profils de comportement dans les données capturées, utilise le modèle de tâche prévu et met en correspondance profil et anomalie. Les anomalies détectées sont issues des principes ergonomiques.

**Avantages et inconvénients**

L'avantage qu'offre cette méthode réside dans le fait que contrairement aux autres méthodes automatiques, ÉMA nécessite la présence des utilisateurs, ce qui permet de mieux apprécier certains problèmes réels d'utilisabilité que n'offrent pas d'autres outils automatiques.

Comme l'explique [Balbo, 1994], l'inconvénient de cette méthode est que diverses anomalies peuvent apparaître lors de sa mise en oeuvre. Les erreurs d'incitation peuvent en effet entraîner une rupture dans l'espace de tâche, et l'utilisateur peut alors s'engager à tort

dans une tâche qu'il ne désire pas effectuer parce que des informations le guident incorrectement.

## IV.3. Conclusion sur les méthodes d'évaluation automatiques

Comme nous l'avons vu ci-dessus, les techniques et les outils d'aide aux choix des méthodes d'évaluation, à adopter pour l'évaluation d'un système interactif, ont plusieurs avantages, donc celui de permettre à l'évaluateur de structurer et organiser son évaluation.

Les outils automatiques d'aide à l'évaluation que nous avons présentés sont tous, d'une façon ou d'une autre, liés à un environnement de développement quelconque et par la suite difficilement exploitables. Cependant notre ambition n'est pas de proposer un outil d'évaluation indépendant de son environnement de développement (et ceci sans introduire du code dans l'application à évaluer), mais de proposer un outil d'évaluation dédié aux systèmes interactifs orientés agent. En effet, les méthodes d'évaluation automatiques que nous avons présentées ne prennent pas en compte la spécificité d'un système interactif orienté agent et ne préconisent aucun traitement particulier pour ce type de système. Par ailleurs, les techniques de récupération et de capture des comportements des utilisateurs utilisées par ces outils d'évaluation automatiques semblent être prometteuses. Parmi les méthodes que nous avons présentées ci-dessus seul EMA préconise la participation de l'utilisateur pour la récupération des données liées à son comportement.

Selon [Bastien et Scapin, 2001 ; Ivory et Hearst, 2001], les méthodes qui se basent sur la présence des utilisateurs (comme EMA) pour la capture de leurs comportements et de leurs interactions avec le système, permettent de mieux apprécier certains problèmes réels d'utilisabilité que n'offrent pas d'autres outils automatiques. La méthode d'évaluation que nous visons se base sur ce principe. En effet, comme nous le verrons au chapitre quatre, nous proposons un d'outil de recueil d'information pour l'évaluation de systèmes interactifs à base d'agents, qu'il s'agit de combiner avec des approches plus traditionnelles (oculométrie, interviews, et questionnaires).

# V. Conclusion

Nous avons présenté dans ce chapitre un panorama des méthodes d'évaluation. Nous avons en premier lieu présenté les méthodes d'évaluation usuelles selon trois grandes catégories : les méthodes basées sur des techniques d'observation de l'utilisateur réel et de recueil des données de l'interaction, les méthodes basées sur l'intervention d'experts en interaction homme-machine, en psychologie cognitive ou en ergonomie et les méthodes analytiques basées sur des modèles formels prédictifs ou sur des modèles dit de qualité de l'IHM. En second lieu, nous avons présenté des méthodes d'aide à l'évaluation ainsi que des méthodes d'évaluation automatiques.

Nous avons constaté, après la présentation d'une grande variété de méthodes contribuant à l'évaluation des interfaces homme-machine, qu'à notre connaissance, il n'existe pas, au stade actuel de la recherche, d'outil ou de méthode spécifique à l'évaluation des systèmes interactifs possédant une architecture à base d'agents. A ce sujet, pour une évaluation aussi bien technique qu'ergonomique, il s'agit de recueillir puis d'étudier l'ensemble des interactions entre, d'une part les utilisateurs et les agents, et d'autre part entre les agents eux-mêmes. Les interactions entre agents ne concernent pas cette thèse en interaction homme-machine, mais nous y reviendrons dans les perspectives de recherche.

De manière générale, les chercheurs du domaine sont d'accord sur le fait que toutes les méthodes d'évaluation existantes peuvent et doivent être combinées pour arriver à une évaluation pertinente de l'utilité et l'utilisabilité de l'interface, notamment dans le cas des systèmes interactifs complexes. C'est dans cet état d'esprit que nous avons orienté nos travaux de recherche en proposant dans le quatrième chapitre une méthode d'évaluation basée essentiellement sur un outil spécifique de recueil d'information (mouchard électronique) pour l'évaluation de systèmes interactifs orientés agents. L'utilisation de ce mouchard est combinée avec des approches empiriques plus traditionnelles telles que : l'oculométrie, le questionnaire et la verbalisation.

Nous ne pouvons pas aborder l'évaluation des systèmes interactifs orientés agents sans se préoccuper de leur conception et de leur architecture. En effet, la méthode d'évaluation que nous proposons au chapitre quatre est étroitement liée à l'architecture du système à évaluer. C'est dans cet état d'esprit, et en se basant sur les architectures dédiées au système interactif présenté au chapitre précédent que nous présentons dans le chapitre suivant une architecture orientée agent et son évaluation.

*Chapitre 3*

# *Principe de couplage entre architecture à base d'agents de système interactif et son évaluation*

---

1. *Introduction*
2. *Principe et cadre global*
3. *Principe d'une architecture à base d'agents dédiée aux systèmes interactifs*
4. *Proposition d'un outil d'aide à l'évaluation des systèmes interactifs orientés agents : le mouchard électronique MESIA*
5. *Conclusion*

# I. Introduction

Nous avons présenté au chapitre précédent un état de l'art donnant un aperçu de la grande variété des méthodes contribuant à l'évaluation des interfaces homme-machine. Cet état de l'art nous a permis de déduire deux constats : (1) à notre connaissance, il n'existe pas, au stade actuel de la recherche, d'outil ou de méthode spécifique à l'évaluation des systèmes interactifs orientés agents, (2) on sait aujourd'hui qu'un évaluateur même expérimenté identifie en moyenne moins de la moitié des problèmes potentiels d'une interface, d'où la nécessité d'avoir des outils d'aide à l'évaluation permettant d'assister les évaluateurs et de structurer leurs démarches.

Dans la mesure où nous nous intéressons particulièrement à l'évaluation des interfaces homme-machine des systèmes interactifs orientés agents dédiés à la supervision de procédés industriels, on peut ajouter aux deux constats, énoncés ci-dessus, notre constat du premier chapitre qui est l'absence d'architecture bien adéquate aux systèmes interactifs orientés agents dédiés à la supervision.

Afin de résoudre les problèmes liés à ces constats, nous proposons dans ce chapitre un principe de couplage entre architecture à base d'agent du système interactif et un outil d'évaluation. Notre proposition vise d'une part, à fournir une architecture à base d'agent pour la modélisation des systèmes interactifs orientés agents, et d'autre part, à offrir aux évaluateurs un outil d'aide à l'évaluation basé sur un mouchard électronique. Ces deux principaux apports sont étroitement liés dans la mesure où on pense implicitement à la phase d'évaluation tout en développant notre modèle d'architecture.

Pour bien illustrer ce chapitre, nous l'avons structuré en trois parties.

La première partie décrit le principe et le cadre global de notre proposition. Nous y expliquons notre idée de base qui consiste à effectuer un couplage entre l'architecture à base d'agents du système interactif et son évaluation.

La deuxième partie est consacrée à la description des principes d'une architecture à base d'agent dédiée aux systèmes interactifs. Nous y décrivons l'origine de cette architecture ainsi que ses différents composants : les agents d'application (manipulent les concepts du domaine), les agents contrôleur de dialogue (offrent à la fois des services à l'application et à la présentation) et les agents de présentation (ou *agents d'interface* qui assurent l'interaction avec l'utilisateur). Les réseaux de Petri (RdP) de haut niveau et particulièrement les RdP agents seront utilisés pour la modélisation des communications entre les agents ainsi que la dynamique de l'interface.

Nous proposons dans la troisième partie un outil d'aide à l'évaluation basé sur un mouchard électronique baptisé MESIA (Mouchard électronique dédié à l'Evaluation des Systèmes Interactifs orientés Agents). L'idée forte derrière ce mouchard réside dans son couplage avec l'architecture du système à évaluer. En effet, il est constitué de plusieurs *agents mouchard* déduits à partir de l'architecture du système à évaluer et plus particulièrement à partir des agents de présentation. Dans cette thèse en IHM, nous nous focalisons en effet sur le module de présentation. Une connexion aux autres modules fera l'objet de nos perspectives de recherche (Cf. Chapitre V). Outre le module mouchard électronique, le système d'aide à l'évaluation est composé d'un module générateur de réseaux de Petri agent qui exploite les données automatiquement recueillies par le mouchard. Le module Simulation-Confrontation-

Spécification de RdP agents offre aux concepteurs/évaluateurs les trois fonctionnalités suivantes : la Simulation de RdP agents, la confrontation de RdP agents, la spécification de RdP agents.

## II. Principe et cadre global

Nous présentons dans cette section le principe et le cadre global de notre proposition. Notre idée consiste à effectuer un couplage entre architecture à base d'agent du système interactif et son évaluation. Pour mieux structurer cette proposition, nous utilisons le formalisme SADT (Structured Analysis and Design Technique) [Ross, 1997 ; IGL, 1989] bien connue en IHM (Cf. Annexe 1, § II).

Selon le formalisme SADT, on retrouve dans la boîte $A_0$ (la boîte au sommet du digramme) l'objectif global. La figure III.1 montre l'objectif global de notre approche qui consiste à coupler l'architecture à base d'agent du système interactif et son évaluation. Pour atteindre ce but, nous l'avons décomposé en trois sous-buts visibles aux boîtes $A_1$, $A_2$ et $A_3$ :

- **la boîte $A_1$ :** Modéliser l'architecture de l'IHM. Cette tâche consiste à élaborer une architecture dédiée aux systèmes interactifs orientés agents. L'architecture proposée est une combinaison entre les modèles centralisés (Cf. Chapitre I. § II) et les modèles répartis (Cf. Chapitre I. § III). Ces modèles d'architecture représentent (selon le formalisme SADT) les contraintes à respecter.

  Cette boîte sera détaillée à la section § III. Les sorties de cette boîte sont la modélisation de l'IHM et des *agents d'interface*. Ces deux sorties représentent respectivement les entrées des boîtes $A_2$ et $A_3$.

- **la boîte A2** : Configurer l'outil d'évaluation : mouchard électronique. Cette tâche consiste à configurer le mouchard électronique en vue de l'évaluation d'un système interactif orienté agent.

  Ce mouchard est constitué de plusieurs *agents mouchard* déduits à partir de l'architecture du système à évaluer et plus particulièrement à partir du système multi-agent de présentation. Cette boîte sera détaillée à la section § IV.

- **la boîte $A_3$ :** Évaluer l'IHM. Comme on peut le voir sur la figure III.1, les entrées de cette boîte sont : l'IHM modélisée selon l'architecture orientée agent proposée et la configuration du mouchard.

  Nous associons au mouchard électronique d'autres outils d'évaluation traditionnels tels que l'oculomètre, la verbalisation et le questionnaire (Cf. figure III.1). Cette boîte sera détaillée dans le chapitre suivant (Cf. Chapitre IV § IV).

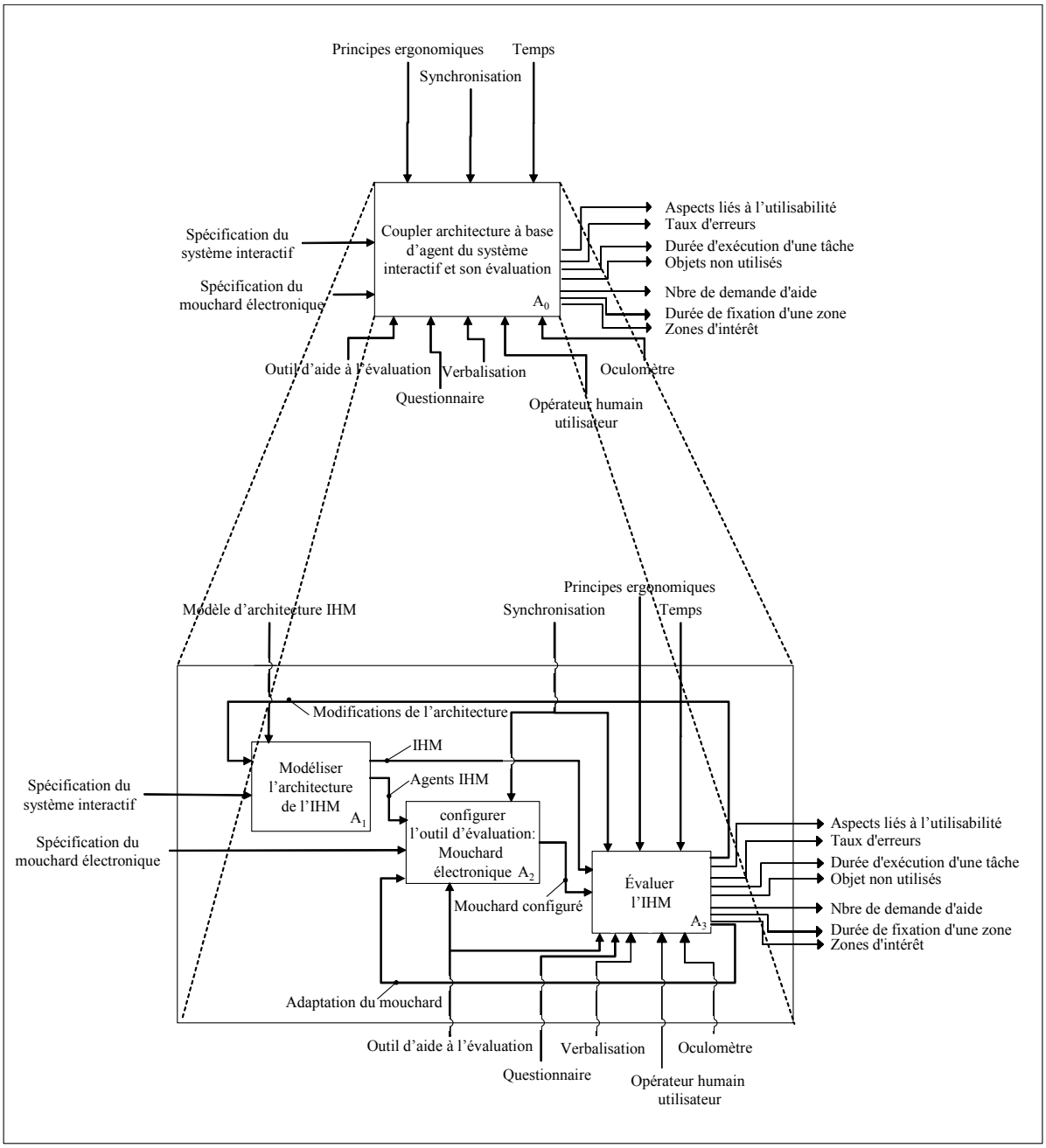



**Figure III.1.** Principe de couplage entre architecture à base d'agents du système interactif et son évaluation

Notons que notre ambition dans ce chapitre n'est pas d'élaborer une architecture explicitement détaillée, mais de proposer les principes d'une architecture dédiée aux systèmes interactifs orientés agents, puisant ses origines entre architectures reparties et architectures centralisées (Cf. § III). L'architecture ainsi proposée nous permettra d'une part, la modélisation des systèmes interactifs orientés agents et particulièrement leurs interfaces, et d'autre part, d'avoir un cadre de pensée pour l'évaluation pour ces systèmes. Ainsi, dans un but d'évaluation aussi bien technique qu'ergonomique, il s'agit : (1) de modéliser le système interactif orienté agents avec l'architecture proposée (2) de recueillir puis d'étudier l'ensemble des interactions entre, d'une part les utilisateurs et les agents, et d'autre part entre les agents eux-mêmes.

Par ailleurs, nous avons constaté au deuxième chapitre, après la présentation d'une grande variété de méthodes contribuant à l'évaluation des interfaces homme-machine, qu'à notre connaissance, il n'existe pas, au stade actuel de la recherche, d'outil ou de méthode spécifique à l'évaluation des systèmes interactifs orientés agents.

Ainsi nous proposons dans se chapitre un système d'aide à l'évaluation basé sur un mouchard électronique baptisé MESIA (Mouchard électronique dédié à l'Evaluation des Systèmes Interactifs orientés Agents) (Cf. § IV).

Nous détaillerons dans la section suivante la boîte A1 de la figure III.1. Nous y présentons les principes d'une architecture à base d'agents dédiée aux systèmes interactifs.

## III. Principes d'une architecture à base d'agents dédiée aux systèmes interactifs

Comme nous l'avons vu au premier chapitre, la définition de l'architecture du système interactif constitue un domaine de recherche à part entière et plusieurs modèles d'architecture sont proposés dans la littérature [Coutaz et Nigay, 2001]. On distingue deux grandes catégories d'architecture :

- une première catégorie est constituée des modèles dits centralisés (Cf. Chapitre 1 II), les plus connus étant le modèle initiateur (le modèle Langage), le modèle dit de Seeheim et ARCH, ces deux modèles ayant fait l'objet de différentes variantes. Ces architectures présentent l'avantage de répartir un système interactif en plusieurs modules. Cependant, elles ne définissent ni la description de la structure interne des modules, ni les interfaces d'échange entre eux.
- une deuxième catégorie est celle des modèles qualifiés de répartis (Cf. Chapitre 1 III), tels MVC, PAC (et ses variantes) et AMF ; ces modèles sont aussi appelés multi-agents. L'avantage de ces architectures est qu'ils ont fait émerger l'idée de décomposer le système interactif selon un ensemble d'agents. Cependant elles rendent la perception de l'interface globale difficile pour le concepteur d'une application de supervision (avec une interface constituée de plusieurs éléments) car l'interface est répartie entre les différents composants « Présentation » de chaque agent.

Afin d'exploiter les avantages de chaque catégorie, nous proposons une architecture mixte entre architecture centralisée et architecture répartie. Cette idée de combinaison n'est pas nouvelle dans la mesure où on trouve dans la littérature des architectures hybrides (Cf. Chapitre 1, § IV) tels que le modèle PAC-Amodeus et $H^4$.

Cependant notre contribution réside dans le fait que l'architecture que nous proposons sera principalement dédiée aux systèmes interactifs de supervision, de plus nous exploiterons plus loin (Cf. § IV) les caractéristiques de cette architecture pour l'évaluation de ces types de systèmes interactifs. Le principe de cette architecture est détaillé à la section suivante.

## III.1. Principe d'une architecture à base d'agents

Nous partons sur l'hypothèse, en nous inspirant d'une part du modèle du cadre général offert par le modèle de Seeheim, et d'autre part de la possibilité de répartir des agents dans chacun des modules, que l'architecture visée suit le principe de celle visible en figure III.2.

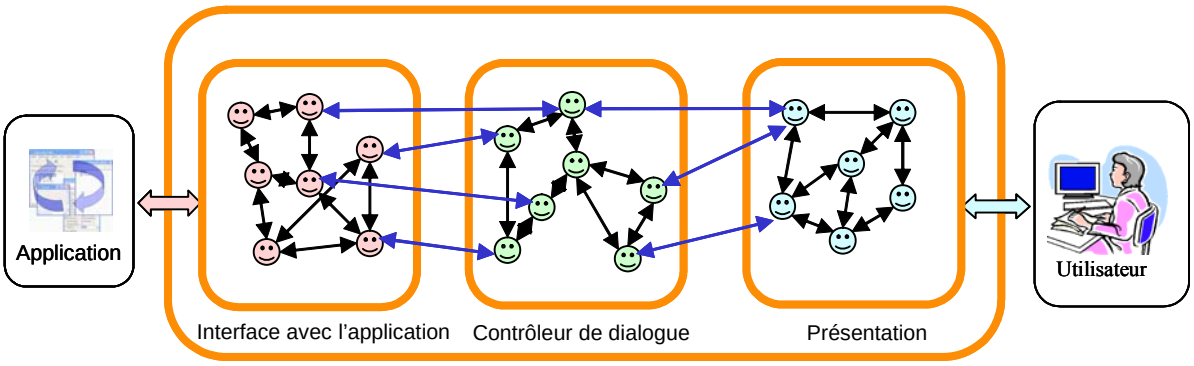



**Figure III.2.** Principe d'une architecture à base d'agent du système interactif [Ezzedine, 2002 ; Trabelsi *et al.*, 2004 a]

Nous avons combiné ces deux approches en proposant d'utiliser dans notre architecture pour les systèmes interactifs, la séparation en trois composants fonctionnels (Cf. figure III.2), que nous avons appelés respectivement : interface avec l'application (connectée à l'application), contrôleur de dialogue, présentation (ce composant étant en lien direct avec l'utilisateur), en nous inspirant de la terminologie citée dans ce domaine (cf. modèle de Seeheim chapitre 1 II.2) [Trabelsi *et al.*, 2004 a]. Ces trois composants regroupent des agents et peuvent être considérés comme des systèmes multi-agent. Ces agents sont des agents très simples et leur intelligence est limitée. Il s'agit ici d'*agents d'interface* comme nous l'avons défini au premier chapitre (Cf. III.1.1).

### III.1.1. Agents Application

Les agents d'application manipulent les concepts du domaine et ne sont pas accessibles directement par l'utilisateur. Ils assurent entre autre le bon fonctionnement de l'application et l'émission en temps réel d'informations nécessaires au déroulement du travail des autres agents.

### III.1.2. Agents contrôleurs de dialogue

Les agents contrôleurs de dialogue appelés aussi agents mixtes, qui offrent à la fois des services à l'application et à l'utilisateur. Ils visent à gérer la cohérence dans les échanges entre l'application et l'utilisateur. Les agents contrôleur de dialogue ont par ailleurs en charge le suivi du dialogue qui s'instaure entre les agents d'application et les *agents d'interface*.

### III.1.3. Agents d'interface

Les *agents d'interface* (ou agents interactifs), sont en contact direct avec l'utilisateur (visibles pour l'utilisateur). Ils prennent en charge les éléments d'interface qui apparaissent sur l'écran et les différentes interactions physiques avec l'utilisateur. Ces agents se coordonnent les uns les autres pour intercepter les commandes de l'utilisateur et composer

pour lui une présentation lui permettant une compréhension globale de la situation courante de l'application.

L'architecture de ces *agents d'interface* peut être assez simple comme celle présentée au premier chapitre (Cf. chapitre 1 § III.1.1) ou assez complexe [Enembreck et Barthès, 2005 ; Mandiau et Grislin-Le Strugeon, 2001 ; Doniec *et al.*, 2006 ; Simonin, 2001].

Les *agents d'interface* communiquent entre eux pour répondre aux actions de l'utilisateur. Cette communication peut être considérée comme des services entre agents et modélisée par les réseaux de Petri de haut niveau [Sibertin-Blanc, 1985] et particulièrement les RdP Agent [Ezzedine *et al.*, 2006].

Comme nous l'avons énoncé au début de ce chapitre, notre ambition n'est pas de proposer une architecture parfaitement détaillée en vue de la modélisation de tout le système interactif mais plutôt de proposer les principes d'une architecture à base d'agents pour la modélisation de l'interface du système. Ainsi nous proposons de définir la notion de services pour la communication entre les agents du composant présentation (Cf. figure III.2). Cette notion de service est maintenant présentée.

## III.2. Notion de service

Chaque agent du composant *présentation* joue un rôle au sein de son groupe. Ce rôle peut s'exprimer sous forme de services[30] qu'il propose dans le système interactif. Ainsi la communication entre agent est assurée par l'invocation de service entre eux comme visible à la figure III.3.

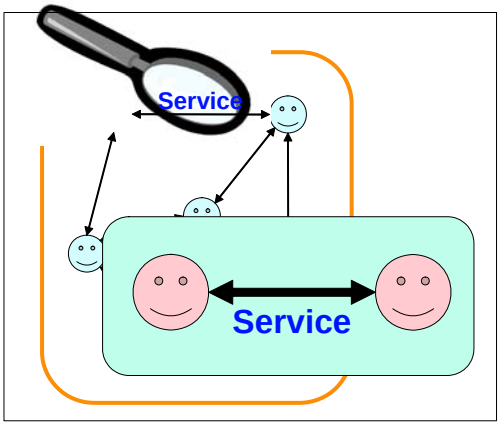



**Figure III.3.** Notion de service entre agents

Afin de définir formellement la notion de services nous utiliserons les réseaux de Petri de hauts niveaux et particulièrement les RdP agent. Nous présentons dans la section suivante la définition d'un service.

[30] Un service est défini dans le dictionnaire « Petit Larousse » de la manière suivante : « Un service est une action effectuée par une entité (personne physique ou morale, entreprise, machine, programme) pour le bien d'une autre, avec ou sans contrepartie ». Notons que l'on retrouve cette notion de service par exemple dans la méthode GAIA (Methodology for Agent-Oriented Analysis and Design) [Wooldridge et al., 2000].

## III.2.1. Définition d'un service

Nous définissons un service [Ezzedine *et al.*, 2003, 2006] par un quadruplet :{E, C, R, P} (Cf. figure III.4) :

- E : évènement qui déclenche le service,
- C : condition à vérifier pour exécuter le service,
- R : ressources éventuelles nécessaires à l'accomplissement du service,
- P : propriété du service qui peut être interne (l'action résultante porte sur l'agent lui-même et nécessite l'utilisation d'une ressource) ou externe (l'action résultante porte sur d'autres agents et n'utilise pas de ressource) ; cette dernière sera interprétée comme un évènement.

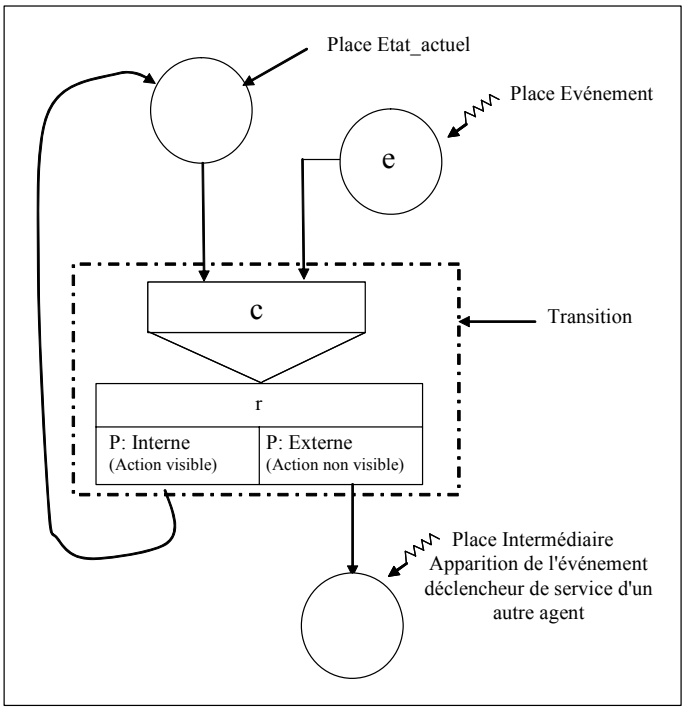



**Figure III.4.** Modélisation d'un service avec les RdP

Pour qu'un service soit déclenché, trois conditions doivent être vérifiées ; ces conditions portent sur :

- l'apparition de l'évènement déclencheur,
- la validation de la condition du déclenchement du service,
- la disponibilité de certaines ressources éventuellement requises pour le déclenchement du service.

On distingue à la figure III.4 trois places qui sont :

- une place *Etat_actuel* : qui contient toujours la vue actuelle de l'agent (résultat d'un service),

- une place *Evènement* : qui contient l'évènement déclencheur de service,
- une place Intermédiaire : qui contient les évènements à destination d'un autre agent.

La place *Etat_actuel* définit l'état actuel de l'agent. Quand un nouvel événement apparaît à la place *événement*, si la condition « c » est vérifiée et la ressource « r » est disponible alors la transition est validée et le service est déclenché. L'agent exécute alors soit, une propriété interne, soit une propriété externe. Si la propriété du service est interne l'action résultante porte sur l'agent lui-même ; on parle alors d'action visible, puisque l'action changera forcement la vue de l'agent. Si la propriété est externe, l'action résultante porte sur un autre agent ; on parle alors d'action non visible. La place *Intermédiaire* stocke l'action non visible en destination d'un autre agent.

Comme nous l'avons énoncé préalablement, la communication entre les agents s'effectue par le déclenchement de service. Ainsi un agent peut être défini par l'ensemble de ses services.

## III.2.2. Modélisation d'un agent sous forme d'un ensemble de services

Nous partons du principe qu'un agent est un ensemble de services. La modélisation de l'agent suit le principe visible sur la figure III.5. Un agent est constitué de n services. Chaque service est défini par le quadruplet :{E, C, R, P}.

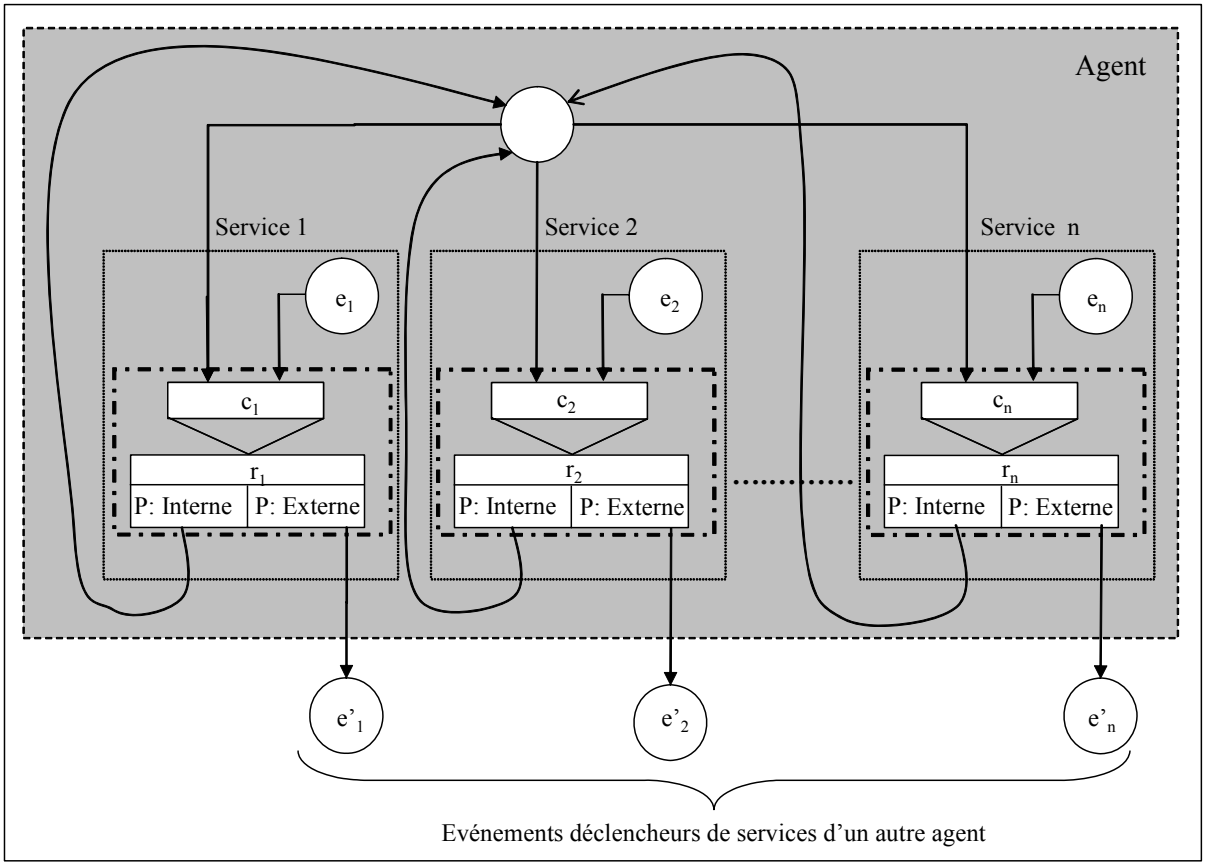



**Figure III.5.** Modélisation d'un agent sous forme d'un ensemble de services

La modélisation d'un agent par l'ensemble de ses services sous la forme visible à la figure III.5 peut être difficilement lisible dès que le nombre de service devient important. Pour pallier ce problème de complexité du RdP agent, on propose d'augmenter le modèle par la capacité d'abstraire l'ensemble des services d'un agent dans une seule forme (Cf. figure III.6).

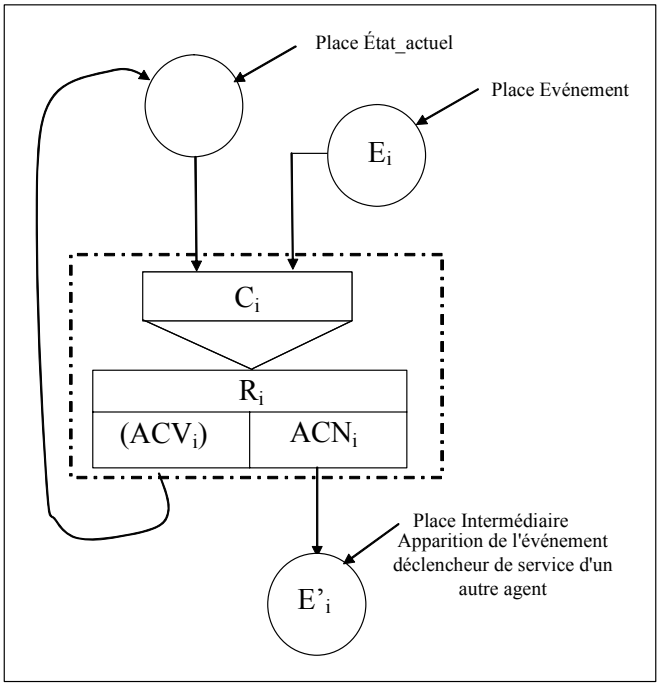



**Figure III.6.** Modélisation d'un agent avec les RdP agent

Ainsi nous définissons un agent (ensemble de n services) par un quadruplet {E, C, R, P} où :

- E = {$e_1$, $e_2$..., $e_i$,…, $e_n$}, ensemble des évènements déclencheurs de services de l'agent (n évènements pour n services). Un évènement est caractérisé par : son identifiant et l'instant de l'apparition.

- C = {c1, c2,..., ci, ..., cn}, ensemble des conditions nécessaires à l'établissement des services. Chaque service doit vérifier une condition pour qu'il puisse se déclencher. Une condition peut être formée de plusieurs conditions élémentaires, elle comporte au minimum la condition élémentaire qui est la présence de l'évènement déclencheur de service.

- R = {r1, r2, ..., ri, ..., rp}, ensemble des ressources éventuellement nécessaires à l'établissement des services (si les services comportent des actions visibles). Une ressource peut être formée de plusieurs ressources élémentaires.

- P = Propriétés des services. Chaque service résulte par l'exécution :

- Soit d'une action visible acv (action qui porte sur l'agent lui-même) ; on parle alors de propriété interne du service.

- Soit d'une action non visible acn (action qui porte sur un autre agent) ; on parle alors de propriété externe du service.

Nous proposons dans la section suivante un modèle formel pour la spécification des agents. Ce modèle nous permettra de modéliser l'ensemble des *agents d'interface* ainsi que la communication entre eux.

## III.3. Un modèle formel pour la spécification des agents

La modélisation formelle d'un agent par un ensemble de services a un objectif double. Elle permet d'une part, la spécification formelle des agents et d'autre part, l'évaluation de l'interface du système interactif. Cette spécification est enregistrée dans une base appelée Base de Spécification des Agent BSA. Cette Base sera utilisée plus loin (Cf. § IV) pour la génération d'un RdP agent dans un but d'évaluation de l'interface du système interactif.

Nous définissons un agent $a_j$ par un ensemble $S_j$ de n services, Sj= $\{s_1, s_2, ..., s_i, ..., s_n\}$.

A chaque service est associé un évènement $e_i$ appartenant à l'ensemble E des évènements, E = $\{e_1, e_2, ..., e_i, \ldots, e_n\}$.

A chaque service est associé une condition $c_i$ appartenant à l'ensemble C des conditions, C= $\{c_1, c_2, ..., c_i, ..., c_n\}$ ; une condition peut être formée de plusieurs conditions élémentaires.

Si le service comporte une action visible, une ressource $r_i$ appartenant à l'ensemble R des ressources est indispensable pour l'exécution de l'action, R= $\{r_1, r_2, ..., r_i, ..., r_p\}$.

L'ensemble des actions de l'agent est formé de deux sous-ensembles ; Acv ensemble des actions visibles et Acn ensemble des actions non visibles.

Acv = $\{acv_1, acv_2,..., acv_p\}$, Acn= $\{acn_1, acn_2, ..., acn_q\}$/ $p \leq n$ et $q \leq n$.

Les cardinalités des ensembles définis ci-dessus sont données dans le tableau III.1.

| Ensemble | Cardinalité |
|---|---|
| E | N |
| C | N |
| R | P |
| Acv | P |
| Acn | Q |

**Tableau III.1.** Cardinalité des ensembles définissant un service

Un service résulte au minimum par une action (soit visible soit non visible) et au maximum par deux actions (visible et non visible). Le nombre de tous les services de l'agent est défini par : Nb_Services = Card (E) = n.

Le nombre de toutes les actions de l'agent est défini par : Nb_Actions = Card (Acv) + Card (Acn) = p + q. d'où : $n \leq$ Nb_Actions $\leq 2 \times n$.

On définit le résultat du service par un couple d'actions ($acv_k$, $acn_t$), où les indices k et t prennent la valeur zéro si le service ne contient pas une action visible ou une action non visible.

Le nombre des couples qui contiennent les deux actions est défini par : Nb_Couples = Nb_Actions – Card (E)

Afin d'avoir un modèle formel qui peut être décrit sous forme d'équations mathématiques, nous ordonnons les couples d'actions de sorte que les couples qui contiennent des actions visibles et non visibles soient les premiers puis les couples qui ne contiennent que des actions visibles en deuxième et en dernier les couples qui ne contiennent que des actions non visibles (Cf. Tableau III.2).

| Couples | Couples avec actions visibles et non visibles | Couples avec actions visibles | Couples avec actions non visibles |
|---|---|---|---|
| | $(acv_{\mathbf{k},j} ; acn_{\mathbf{t},j})$ | $(acv_{k,j} ; acn_{\mathbf{0},j})$ | $(acv_{\mathbf{0},j} ; acn_{t,j})$ |

**Tableau III.2.** Ordonnancement des couples d'actions

Les services sont définis par la fonction paramétrée suivante (le paramètre j étant l'identificateur de l'agent) [Rehim, 2005]:

$S_{i,j} : E_j \times C_j \times R_j \longrightarrow Acv \times Acn.$

$(e_{i,j} ; c_{i,j} ; r_{k,j}) \longmapsto (acv_{k,j} ; acn_{t,j})$. j = 1..nombre des agents

i = 1..n.

$0 \leq k \leq p$ / si $i \leq p$ alors k=i.

sinon k=0.

$0 \leq t \leq q$ / si $i \leq$ Nb_Couples alors t=i

sinon

si i > p alors t = i – p + Nb_Couples.

sinon t=0.

j : identificateur de l'agent
i : numéro de l'évènement
n : numéro du service
p : nombre d'action visible
q : nombre d'action non visible
Acv: action visible.
Acn : action non visible.
k,t : numéro de l'action (si le service ne résulte pas par : une action visible k=0, ou action non visible t=0).

Ainsi, la spécification d'un agent $a_j$ consiste en la définition des ensembles $E_i$, $C_i$, $R_k$, et la spécification mathématique d'un service consiste en la définition d'une séquence de triplets $(e_i, c_i, r_k)$ associés à leurs couples d'actions $(acv_{k,j} ; acn_{t,j})$.

## III.4. Conclusion

Nous avons présenté dans cette partie les principes d'une architecture à base d'agent. Cette architecture est une combinaison entre les architectures centralisées et les architectures réparties. Elle s'inspire du cadre général du modèle de Seeheim pour la séparation du système interactif en trois composants fonctionnels tout en répartissant des agents dans chacun de ces composants. Nous avons également proposé un modèle formel pour la spécification des *agents d'interface*. Ce modèle se base sur les réseaux de Petri agent. La spécification des agents est stockée dans une base appelée BSA (Base de Spécification de Agent). La BSA sera exploitée dans un outil d'aide à l'évaluation pour la génération de RdP agent.

Nous avons exposé au début de ce chapitre le cadre général de notre approche. Nous avons ainsi défini à la figure III.1, selon le formalisme SADT, l'objectif de ce chapitre qui est le couplage entre architecture à base d'agent du système interactif et son évaluation. Nous avons jusqu'à présent, présenté les principes d'une architecture à base d'agent. Nous présentons maintenant un système d'aide à l'évaluation basé sur un mouchard électronique. Ce mouchard nous permettra d'acquérir les différentes interactions de l'utilisateur avec l'interface. Le mouchard électronique est connecté au composant présentation du système interactif à évaluer comme visible à la figure III.7.

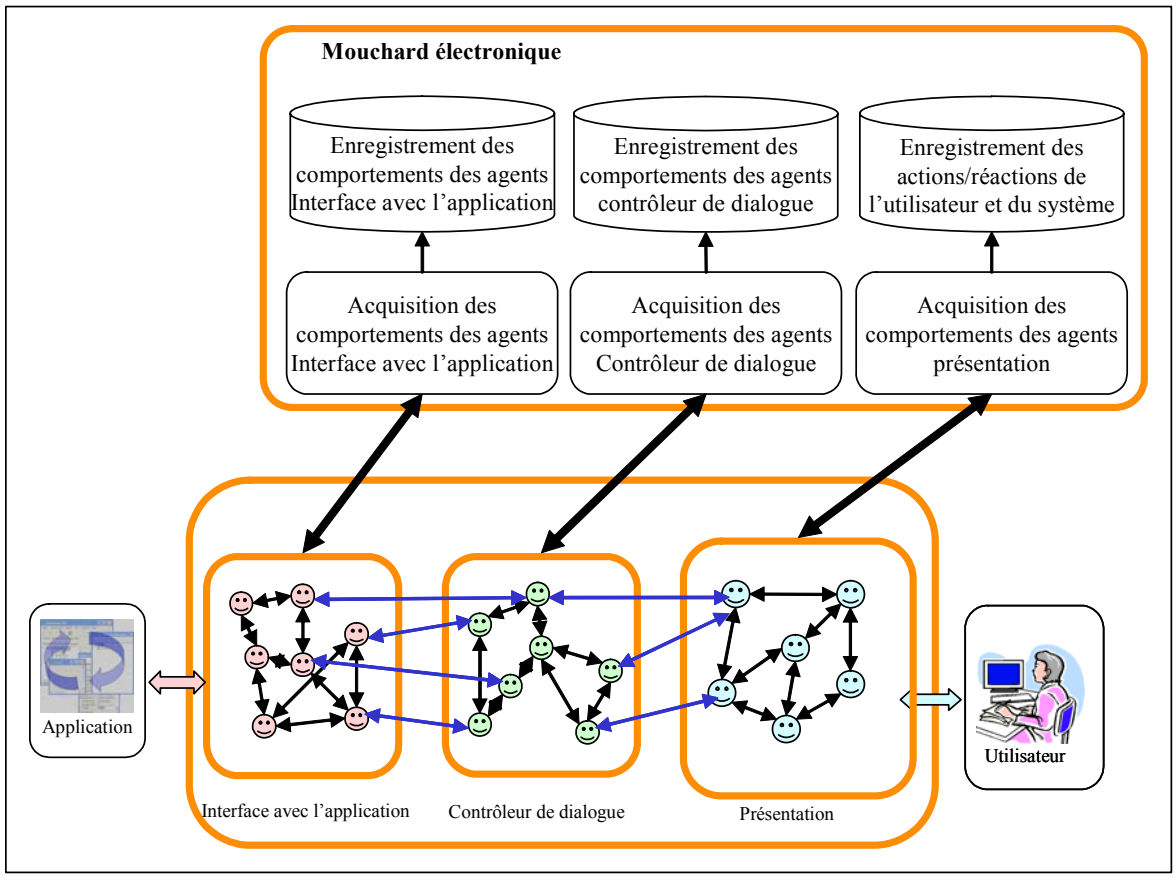



**Figure III.7.** Principe d'utilisation du mouchard électronique pour l'évaluation des systèmes interactif orientés agents

Nous proposons dans la section suivante un outil d'aide à l'évaluation basé sur le mouchard électronique.

# IV. Proposition d'un outil d'évaluation des systèmes interactifs orientés agents : le mouchard électronique MESIA

Nous avons constaté au deuxième chapitre, après la présentation d'une grande variété de méthodes contribuant à l'évaluation des interfaces homme-machine, qu'à notre connaissance, il n'existe pas, au stade actuel de la recherche, d'outil ou de méthode spécifique à l'évaluation des systèmes interactifs orientés agents.

Ainsi nous proposons dans cette section, un outil d'aide à l'évaluation dédié aux systèmes interactifs orientés agents basé sur un mouchard électronique. Rappelons que le mouchard électronique (Cf. Chapitre 2 § III.1.1.3) est un outil qui repose sur le recueil automatique, en situation réelle, des actions des utilisateurs et de leurs répercussions sur le système interactif ainsi que les réactions de ce dernier [Scapin et Bastien, 2002]. Le recueil d'information se fait de façon discrète, l'utilisateur ne se sentant en principe en aucun moment gêné par la présence du mouchard. Ceci constitue un point fort de l'outil et nous permet de recueillir des données objectives.

## IV.1. Le mouchard électronique

### IV.1.1. Principe

L'utilisation d'un mouchard électronique pour l'évaluation des systèmes interactif n'est pas une idée nouvelle puisqu'elle à déjà été proposée au début des années 80 [Hammontree, 1992] ; cependant l'originalité du mouchard que nous proposons réside dans sa liaison directe avec le système à évaluer. En effet, l'existence d'un agent au niveau du système interactif et particulièrement le composant présentation (Cf. figure III.8) conduit à la création d'un *agent mouchard* (dans un but d'étude de faisabilité, nous supposons que le nombre d'agents est raisonnable, de l'ordre d'une vingtaine au maximum). Cette association entre *agent d'interface* et *agent mouchard*, permettra non seulement d'enregistrer les actions de l'utilisateur mais aussi les réactions de l'interface.

La figure III.8 illustre le couplage entre l'architecture à base d'agent du système interactif et son évaluation. Nous y retrouvons l'architecture à base d'agent que nous avons proposé ci-dessus (Cf. § III). Rappelons que cette architecture se décompose en trois composants : le composant interface avec l'application, le composant contrôleur de dialogue et le composant présentation. Comme le suggère la figure III.8, l'architecture du mouchard électronique est liée à celle du composant présentation du système à évaluer. En effet, le composant présentation est constitué d'*agent d'interface* (Cf. § III). L'interaction de l'utilisateur avec le système s'effectue via ces agents. De ce fait, nous proposons l'association d'un *agent mouchard* à chaque *agent d'interface*.

Comme indiqué sur la figure III.8, le mouchard électronique est constitué de trois modules [Ezzedine et Trabelsi, 2005] :

- **le module *Recueil de données*** : ce module assure l'enregistrement des interactions (actions de l'utilisateur et réactions de l'interface). Les données recueillies seront stockées dans une base de données puis exploitées par le module d'analyse de données ;

- **le module *traitement des données*** : ce module récupère les données enregistrées par le module recueil des données et les traite par la suite. Dans ce module le traitement des données se limite pour l'instant à leur classification et leur analyse en vue d'obtention de statistiques. En effet, ce module fournit à l'évaluateur un compte rendu des données recueillies sous forme de statistiques de graphes afin de tirer des conclusions en ce qui concerne la performance, les temps de réponse, etc. ;

- **le module *Recommandation*** : ce module, faisant l'objet essentiellement de perspectives dans le cadre de cette recherche, a pour rôle de fournir à l'évaluateur des recommandations et des changements à effectuer sur l'interface. Il se base essentiellement sur des règles ergonomiques et des règles issues de guides de styles (Cf. Chapitre 2, III.2.2).

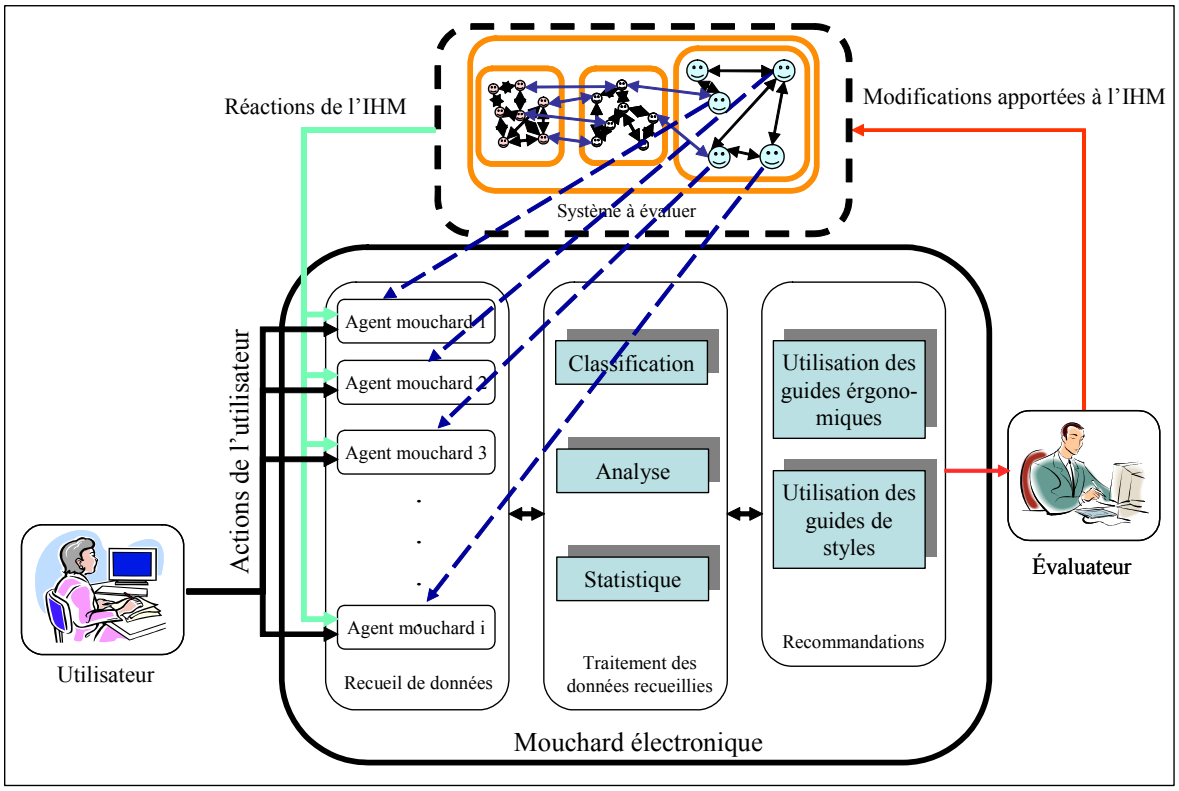



**Figure III.8.** Principe de couplage entre architecture à base d'agent du système interactif et son évaluation

## IV.1.2. Architecture du mouchard électronique

Comme nous l'avons cité ci-dessus, nous avons proposé l'association d'un *agent mouchard* à chaque *agent d'interface*. En parallèle, nous proposons l'intégration d'un agent *MSouris* (M pour mouchard) et d'un agent *MClavier* (Cf. figure III.9), qui en collaborant avec les autres *agents mouchard* permettront l'enregistrement des différentes actions/réactions de l'interface (*agents d'interface*) et de l'utilisateur [Trabelsi et al, 2003]. L'architecture du mouchard est visible à la figure III.9.

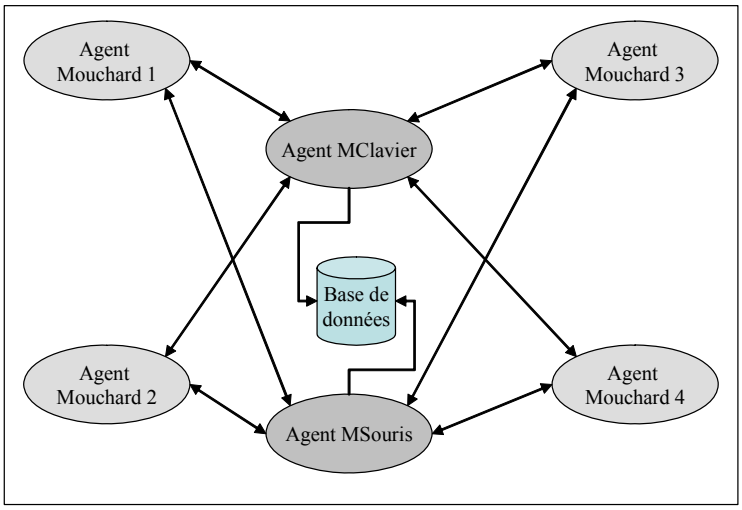



**Figure III.9.** Architecture du mouchard électronique. Le nom des *agents mouchard* est celui des *agents d'interface* précédé d'un « M » [Trabelsi *et al.*, 2004 b].

La tâche des *agents mouchard* consiste à percevoir les actions/réactions de l'*agent d'interface* correspondant ainsi que celles de l'utilisateur, et de les enregistrer. En effet, une architecture possible de ces *agents d'interface* est visible à la figure III.10. Cette architecture est constituée d'un module de perception, assurant l'acquisition des données fournies par l'utilisateur et l'agent présentation, d'un module interprétation et classification assurant un tri des données recueillis et d'un module enregistrement assurant l'enregistrement des données dans une base de données.

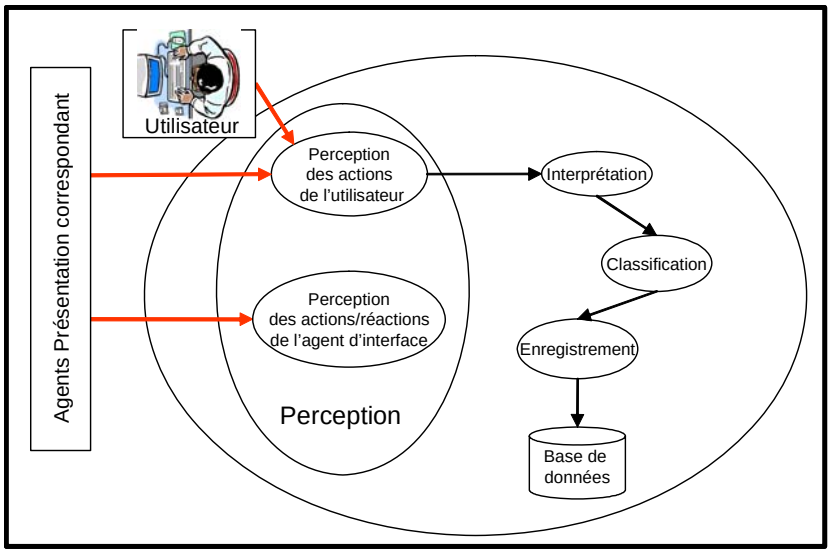



**Figure III.10.** Architecture interne d'un *agent mouchard* [Trabelsi *et al.*, 2004 b].

La phase de classification dans le module d'acquisition des données permet d'attribuer chaque donnée enregistrée à une catégorie d'informations spécifique. On peut distinguer cinq catégories possibles [Trabelsi *et al.*, 2004 b]:

- **mouvement de la souris** : tous les événements de la souris sont interceptés par le mouchard. on distingue trois types d'événements possibles : le déplacement de la souris, le click sur le bouton droit et le click sur le bouton gauche. L'évaluateur

peut alors choisir les événements à visualiser. Une visualisation de tous les événements simultanément est possible,

- **données saisies par le clavier** : durant son interaction avec le système, l'utilisateur peut utiliser le clavier pour fournir des données au système. La saisie d'une mauvaise valeur peut entraîner de graves conséquences dans certains systèmes industriels. De ce fait il est intéressant de dédier au mouchard la tâche de recueil de toutes les informations saisies au clavier,

- **données saisies par la parole** : l'utilisation de la parole comme moyen d'interaction est de plus en plus importante dans les systèmes interactifs. Le mouchard intercepte les paroles échangées entre le système et l'utilisateur d'une part, entre l'utilisateur et d'autres membres de l'équipe d'autre part, et les enregistre. Vu la complexité d'analyse automatique sémantique et syntaxique de ces données, seul l'enregistrement sous format audio est envisageable. Ces données sont ensuite exploitées directement par les évaluateurs pour être mises en relation avec les autres données acquises,

- **durée de visualisation de chaque vue** : à chaque apparition d'une vue, un chronomètre est enclenché ; et s'arrête au moment de sa disparition ou sa mise en arrière plan. La valeur en sortie représente la durée de visualisation d'une vue. La durée de l'apparition des vues est très importante. En effet, il peut exister une relation étroite entre la durée de consultation d'une vue et les stratégies mises en oeuvre par les utilisateurs (cf. par exemple les travaux de [Keyser *et al.*, 1987]),

- **utilisation de l'aide** : la fréquence d'utilisation de l'aide peut révéler le degré de compréhension de l'interface, ayant des conséquences sur son utilisabilité. Le mouchard électronique détecte la fréquence d'utilisation de l'aide (si elle est disponible) relative à chaque vue.

Pour récupérer les données citées ci-dessus chaque *agent mouchard* est connecté à un agent du système à évaluer. L'établissement de cette connexion ainsi que la coordination entre les différents *agents mouchard* est assuré par l'*agent mouchard* superviseur qui est maintenant présenté.

## IV.1.3. Agent mouchard superviseur

L'*agent mouchard* superviseur est responsable de la coordination entre les différents *agents mouchard*. Il assure la visualisation en temps réel d'un histogramme permettant l'affichage des durées de visualisation de chaque *agent d'interface* du système à évaluer. Cet agent permet de rejouer toute la manipulation. A la fin de la phase d'enregistrement, un fichier journal contenant les positions absolues de la souris peut être exécuté. Nous verrons ainsi la souris se déplacer tout en reproduisant les mêmes actions effectuées par le régulateur.

La connexion entre le système à évaluer et le mouchard électronique s'effectue via un réseau local. En cas de problème de transmission de données c'est l'*agent mouchard* superviseur qui assure, soit la bonne reprise de connexion ou bien la déconnexion (en toute sécurité) des deux systèmes.

Le système à évaluer peut être exécuté sur une machine autre que celle sur laquelle s'exécute le mouchard électronique. L'adresse IP de la machine où s'exécute le système à évaluer peut être spécifiée grâce à l'onglet « IP Connexion ».

Afin de mieux assister les évaluateurs des systèmes interactifs orientés agents, nous proposons dans la section suivante un outil d'aide à l'évaluation basé sur le mouchard électronique.

## IV.2. Le système d'aide à l'évaluation

### IV.2.1. Architecture structurelle

L'outil d'aide à l'évaluation est composé de trois modules (Cf. figure III.12)

- le module mouchard électronique,
- le module de génération de RdP,
- le module Simulation/Confrontation/Spécification de RdP agent.

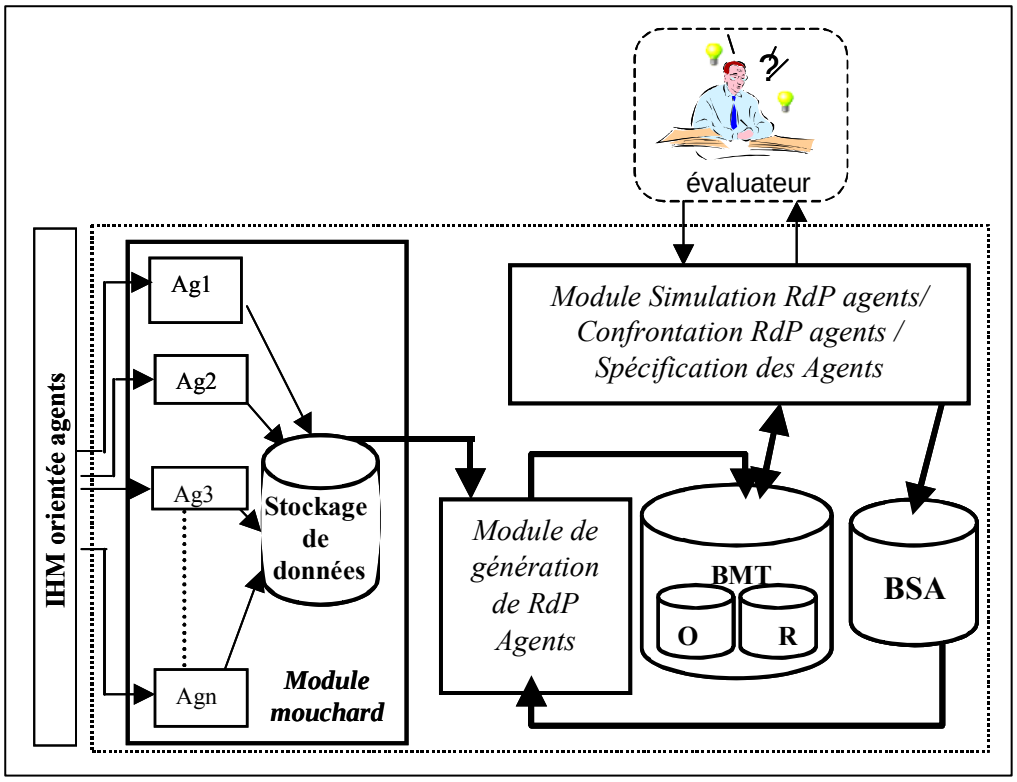



**Figure III.11.** Architecture du système d'aide à l'évaluation [Rehim, 2005 ; Trabelsi et Ezzedine, 2006]

### IV2.2. Module mouchard électronique

Ce module utilise le principe du mouchard électronique (détaillé dans la section III) tout en exploitant à la fois le modèle orienté agent de celui-ci qui propose l'association d'un *agent mouchard* à chaque agent de l'interface.

## IV.2.3. Module de génération de RdP Agent

Ce module exploite les données recueillies par le mouchard électronique pour les comparer aux spécifications existantes dans la base de spécifications des agents (Cf. figure III..12). Il fournit un RdP agents qui représente le modèle de la tâche effectuée selon les principes décrits dans [Abed *et al.*, 2001]. Ce module est lui-même composé de deux modules générateurs de RdP agent : de la tâche à Réaliser (modèle de tâche) et de la tâche Observée (modèle de l'activité) (respectivement R et O sur la figure III.11) et deux bases BSA et BMT.

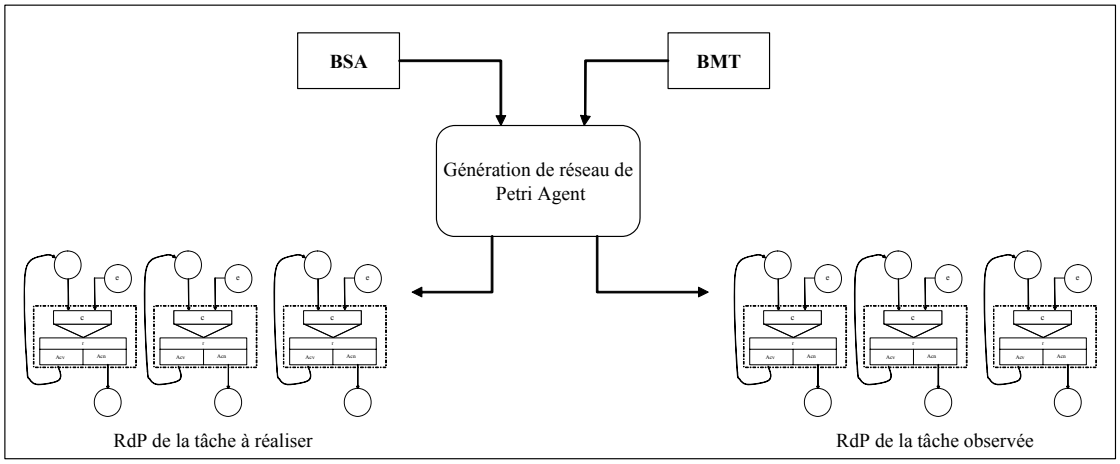



**Figure III.12.** Génération de RdP de la tâche à réaliser et celle observée

## IV.2.3.1. La Base de Spécification des Agents (BSA)

La Base de Spécification des Agents contient la définition (pour chaque agent) des ensembles E (ensemble des évènements), C (ensemble des conditions), R (ensemble des ressources), Acv (ensemble des Actions visibles telles que l'action de l'opérateur humain utilisant la souris ou le clavier et la réaction de l'interface par affichage des nouvelles fenêtres et/ou changement de leurs contenus), Acn (ensemble des Actions non visibles concernant les interactions entre agents internes de l'interface) ce qui permet et facilite leur exploitation par le module de génération de RdP agents

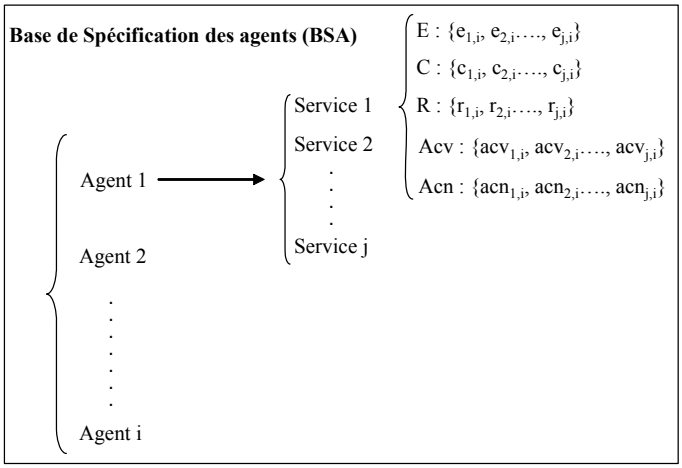



**Figure III.13.** Base de spécification des agents

### IV.2.3.2 La Base des Modèles des Tâches (BMT)

La Base des Modèles des tâches (BMT) se décompose en (1) BMT (O) qui contient les modèles des tâches humaines Observées (qui sont générées par le module de génération de RdP agents), (2) BMT (R) qui contient les modèles de référence des tâches à réaliser par les opérateurs humains, introduits par les concepteurs/évaluateurs via le module Simulation-Confrontation-Spécification de RdP agents

## IV.2.4. Module de Simulation/Confrontation/Spécification de RdP agent

Ce module offre aux évaluateurs/concepteurs les trois fonctionnalités suivantes : la simulation de RdP agents, la confrontation de RdP agents, la spécification de RdP agents.

- Simulation de RdP agents : cette fonction permet la visualisation de la dynamique du RdP agents et par conséquent la dynamique de l'IHM, ceci par l'exploitation des modèles de tâches ainsi que la formulation mathématique qui assure l'évolution dans le RdP agents.

- Confrontation de RdP agents : cette fonction exploite les modèles de tâches (Observée, Référence) pour la confrontation. Cette dernière a pour objectif de faciliter aux évaluateurs l'identification de problèmes ergonomiques éventuels liés à l'utilisabilité du système interactif ; par exemple s'apercevoir que le RdP agents du modèle de la tâche observée contient des états en plus, c'est-à-dire que l'utilisateur passe par des étapes inutiles, ou alors que le temps pris pour réaliser une tâche est bien supérieur à celui prévu au départ par les évaluateurs/concepteurs.

- Spécification de RdP agents : cette fonction consiste en la mise à la disposition des évaluateurs/concepteurs de moyens (fenêtres) permettant la gestion (saisie, modification,…) de spécification des agents, autrement dit la définition des ensembles E, C, R, Acv, Acn, et leur stockage dans la base de spécifications des agents.

## IV.3. Conclusion sur l'outil d'évaluation proposé

Nous avons proposé dans cette section les principes d'un mouchard électronique dédié à l'évaluation des systèmes interactifs et particulièrement leur composant présentation. L'architecture de cet outil est liée à l'architecture du système à évaluer. A chaque *agent d'interface* est associé un *agent mouchard*. Deux *agents mouchard MSousirs* et *MClavier* se coordonnent avec les autres *agents mouchard* afin d'enregistrer les actions/réactions de l'utilisateur et de l'interface.

Nous avons également proposé un système d'aide à l'évaluation. C'est un système qui exploite les RdP agents pour permettre l'évaluation de l'IHM à base d'agents. Il est composé de trois modules principaux : mouchard électronique, générateur de RdP agent et Simulation/Confrontation/Spécification de RdP agent. Ce dernier fournit aux évaluateurs/ concepteurs la possibilité de simuler la dynamique de l'IHM, spécifier mathématiquement le RdP agents, confronter des modèles des tâches (observées, référence).

Notons que le système d'aide à l'évaluation dont nous avons présenté les principes est basé sur des techniques d'observation de l'utilisateur réel et de recueil des données de l'interaction, présentés au chapitre 2 (Cf Chapitre 2, § III.1). Par ailleurs, dans la mesure où ce système a pour objet d'assurer la capture et l'analyse automatique des actions/réactions de l'utilisateur et du système à évaluer, il peut être également classé parmi les outils d'aide à l'évaluation automatique tel que SYNOP, KRI/AG, ERGOVAL, CHIMES, ÉMA (cf. chapitre II, § IV.2).

A la différence des outils automatiques d'aide à l'évaluation existant, le système d'aide visé est le seul, à notre connaissance, qui est particulièrement dédié à l'évaluation des systèmes interactifs orientés agents. Cette particularité est assurée par le couplage entre le système interactif orienté agent à évaluer et le mouchard électronique. En effet, les outils existants ne préconisent aucun traitement particulier pour ce type de système. D'autre part, à l'exception de EMA, le système d'aide visé est le seul qui préconise la participation de l'utilisateur pour la récupération des données liées à son comportement. Comme nous l'avons vu au chapitre II, les méthodes qui se basent sur la présence des utilisateurs pour la capture de leurs comportements et de leurs interactions avec le système, permettent de mieux apprécier certains problèmes réels d'utilisabilité que n'offrent pas d'autres outils automatiques

Pareillement à ERGOVAL et EMA, nous préconisons que le système d'aide visé s'intègre dans une combinaison entre plusieurs approches traditionnelles d'évaluation (mouchard électronique, oculométrie, interviews et questionnaires). L'utilisation de plusieurs outils et techniques est justifié par la diversité des informations à récupérer lors de l'évaluation. En effet les informations fournies par ces différents outils et techniques seront complémentaires et permettront à l'évaluateur de détecter un ensemble représentatif de problèmes d'utilité et/ou d'utilisabilité inhérents à l'exploitation du système à évaluer.

Basé sur les mêmes principes que KALDI [Al-Qaimari et McRostie, 1999] et WebRemUSINE [Paganelli et Paternò, 2002] qui exploitent les fichiers log pour comparer les séquences d'actions effectuées par l'utilisateur aux séquences de tâches définies préalablement à l'aide de l'environnement CTTE [Paterno *et al.*, 1997], il est prévu que l'outil d'aide visé exploite les données capturées par le module mouchard électronique afin d'établir les modèles de tâches observés et les confronter aux modèles de taches réels.

Notre outil d'aide à l'évaluation rejoint aussi l'idée proposée par [Tarby, 2006] qui préconise le recueil des actions/réactions du système à évaluer et de l'utilisateur en produisant des traces « orientées usage » au niveau du noyau fonctionnel et de l'IHM. Plutôt que de mettre en place un système de trace après la conception, [Tarby, 2006] propose de concevoir les systèmes à évaluer en greffant dès le début un tel système. Ces traces fournissent des informations plus complètes et plus facilement exploitables lors des phases d'évaluation. On parle alors de conception orientée évaluation qui repose, dans la proposition de cet auteur, sur les tâches et la programmation par aspects.

## V. Conclusion

Dans la mesure où, dans différents domaines d'application, on se dirige maintenant vers de nouveaux types de systèmes interactifs possédant des architectures à base d'agents, il est maintenant nécessaire d'outiller les démarches d'évaluation de tels systèmes. Nous avons proposé dans ce chapitre un principe de couplage entre architecture à base d'agent du système interactif et son évaluation.

La première partie a décrit le principe et le cadre global de notre proposition. Nous y avons illustré notre idée de base qui consiste à effectuer un couplage entre l'architecture à base d'agents du système interactif et son évaluation. La deuxième partie a été consacrée à la description des principes d'une architecture à base d'agent dédiée aux systèmes interactifs. L'origine de cette architecture est un couplage (éviter ce mot ! puisque ce mot est au centre du chapitre) entre les architectures centralisées et les architectures réparties. Elle est structurée en trois composants : l'interface avec l'application (constitué d'agents d'application qui manipulent les concepts du domaine), le contrôleur de dialogue (constitué d'agents contrôleur de dialogue qui offrent à la fois des services à l'application et à la présentation) et la présentation (constituée d'*agents d'interface* qui assurent l'interaction avec l'utilisateur). Les réseaux de Petri (RdP) de haut niveau et particulièrement les RdP agents ont été utilisés pour la modélisation des communications entres les agents ainsi que la dynamique de l'interface. Dans la troisième partie nous avons proposé un outil d'aide à l'évaluation basé sur un mouchard électronique baptisé MESIA (Mouchard électronique dédié à l'Evaluation des Systèmes Interactifs orientés Agents). L'idée forte derrière ce mouchard réside dans son couplage avec l'architecture du système à évaluer. Ce mouchard est constitué de plusieurs *agents mouchard* déduits à partir de l'architecture du système à évaluer et plus particulièrement à partir des agents de présentation. Le système d'aide à l'évaluation est constitué de trois modules : le module mouchard électronique, le module générateur de réseaux de Petri agent qui exploite les données automatiquement recueillies par le mouchard et le module Simulation-Confrontation-Spécification de RdP agents qui offre aux évaluateurs/concepteurs les trois fonctionnalités suivantes : la Simulation de RdP agents, la confrontation de RdP agents et la spécification de RdP agents.

Le système d'aide à l'évaluation que nous avons proposé a été conçu pour faciliter l'évaluation des systèmes interactifs orientés agents. Afin de valider celui-ci, notre cadre d'application est le SAI (Système d'Aide à l'Information voyageurs) qui devrait être utilisé à terme dans la future salle de régulation du réseau de bus et tramway de Valenciennes. Nous détaillerons dans le chapitre suivant notre contribution, consistant en la spécification et au maquettage du SAI, ainsi qu'à son couplage à un module d'évaluation.

*Chapitre 4*

# *Mise en application du couplage architecture/module d'évaluation dans un contexte lié aux transports*

---

1. *Introduction*
2. *Cadre d'application : le projet SART*
3. *Analyse, Modélisation, Spécification et Maquettage du SAI*
4. *Implémentation du mouchard électronique et couplage SAI*
5. *Conclusion*

# I. Introduction

Dans le chapitre précédent, nous avons proposé un principe de couplage entre architecture à base d'agent du système interactif et son évaluation. Nous avons présenté une architecture à base d'agent dédiée aux systèmes interactifs. Les réseaux de Petri (RdP) de haut niveau et particulièrement les RdP agents ont été utilisés pour la modélisation des communications entre les agents ainsi que la dynamique de l'interface. En se basant sur la spécificité de l'architecture proposée (séparation entre les agents d'application, agents contrôleurs de dialogue et les *agents d'interface*), nous avons proposé un outil d'aide à l'évaluation basé sur un mouchard électronique baptisé MESIA (Mouchard électronique dédié à l'Evaluation des Systèmes Interactifs orientés Agents). L'originalité de ce mouchard réside non seulement dans sa capacité à enregistrer les actions/réactions de l'utilisateur et du système, mais aussi dans son couplage avec l'architecture du système à évaluer. En effet, le mouchard MESIA est constitué de plusieurs agents mouchard déduits à partir de l'architecture du système à évaluer et plus particulièrement à partir des *agents d'interface*.

Nous visons dans ce chapitre la validation de notre proposition établie au chapitre précédent. Le projet SART (Système d'Aide à la Régulation de Trafic) nous servira de cadre d'application. Ce projet vise la réalisation d'un système d'aide à l'information (SAI) permettant d'assister les régulateurs humains et de faciliter leurs tâches. Ce système devrait être utilisé à terme dans la future salle de régulation du réseau de bus et tramway de Valenciennes

Nous avons structuré ce chapitre en trois parties.

La première partie concerne une présentation globale du projet SART. Nous y présentons les objectifs du projet ainsi que les différentes parties qui le constituent. Afin de mieux comprendre les besoins du projet SART en termes de système d'aide à l'information, une analyse de l'existant est faite.

La deuxième partie est consacrée à l'analyse, la modélisation, la spécification et au maquettage du SAI. La phase d'analyse consiste à déterminer les caractéristiques de l'information à délivrer aux voyageurs et à déterminer les événements qui agissent sur SAI. Nous y présentons aussi les besoins informationnels du SAI pour communiquer avec les autres modules du projet SART. Dans toute application interactive, l'utilisation d'un formalisme décrivant la dynamique de l'interface est nécessaire ; nous utilisons les réseaux de Petri interprétés pour la modélisation des interactions entre le régulateur et le SAI ainsi que le fonctionnement du SAI en mode normal et en mode perturbé. La spécification sera quand à elle basée sur l'architecture orientée agents que nous avons proposée au chapitre trois (Cf. chapitre trois, § III). Ainsi, nous effectuerons la spécification des différents composants de l'architecture proposée à savoir : les *agents d'application*, les *agents contrôleurs de dialogue* et les *agents d'interface*. Nous utiliserons les réseaux de Petri agent pour la spécification des *agents d'interface*. Une première maquette sera proposée pour le SAI.

Notre objectif étant l'évaluation du SAI, nous détaillerons dans la troisième partie l'implémentation du mouchard électronique qui nous servira d'outil de base pour l'évaluation du SAI. Les agents mouchard seront décrits. Afin de pouvoir enregistrer les actions/réactions du régulateur et du SAI, il est nécessaire de connecter ce dernier au mouchard électronique ; on parle alors de « couplage ». Or, comme nous l'avons déjà vu au chapitre précédent, nous associons à chaque *agent d'interface* du système à évaluer un *agent Mouchard*. Ainsi, nous

proposons une connexion directe entre chaque *agent d'interface* du SAI et son *agent Mouchard* correspondant.

# II. Cadre d'application : le Projet SART

## II.1. Présentation globale

Le projet SART (Système d'Aide à la Régulation de Trafic) du réseau de transport valenciennois et de ses pôles d'échanges est un projet coopératif du GRRT (Groupement Régionale Nord-Pas-de-Calais pour la Recherche dans les Transport). Il consiste en une amélioration de la qualité des correspondances dans les réseaux de transports urbains, et une mise en œuvre pratique du projet MOST (Méthodologies pour l'Optimisation des Systèmes de Transport).

Le projet SART fait intervenir un partenaire industriel, la SEMURVAL qui exploitera le futur réseau de transport urbain (Tramway/Bus) de la ville de Valenciennes ainsi que plusieurs laboratoires de recherche (LAMIH, LAGIS, INRETS-ESTAS). Ce projet est supporté par la région Nord-Pas-de-Calais et le FEDER (Fonds Européen de Développement Régional).

L'objectif du projet SART est de réaliser un système permettant d'assister les régulateurs et de faciliter leur tâche en mode normal et en mode dégradé de fonctionnement du réseau. Ce système doit être bien intégré dans le poste de travail des régulateurs et d'utilisation facile. L'impact de ce système sur les voyageurs est de minimiser leurs temps d'attente, en mode dégradé, dans les pôles d'échange et de leur assurer, dans la mesure du possible, la continuité des déplacements dans les réseaux multimodaux. Il s'agit donc d'améliorer la qualité du service rendu aux voyageurs et de les maintenir informés [Hayat *et al.*, 2001].

La régulation[31] du trafic est une régulation globale du réseau de transport collectif. Il s'agit d'arriver à un système d'aide à la régulation du trafic terrestre du réseau urbain de la ville de Valenciennes. Ce système de régulation est constitué de trois sous-systèmes (Cf. Figure IV.1) :

- **Système d'Aide à l'Exploitation (SAE) :** il permet la régulation de trafic dans les inter-stations et les nœuds de correspondances ainsi qu'en station selon le flux de passagers,
- **Système d'Aide à la Décision (SAD) :** il offre une aide aux régulateurs humains pour la prise de décision en nœuds de correspondance et en inter-stations,
- **Système d'Aide à l'Information (SAI) :** il assure d'une part l'information des régulateurs en salles de contrôle, et d'autre part l'information des voyageurs en stations et dans les véhicules.

[31] La régulation est le processus d'adéquation en temps réel des tableaux de marche des différents modes de transport aux conditions réelles d'exploitation [Fayech, 2003].

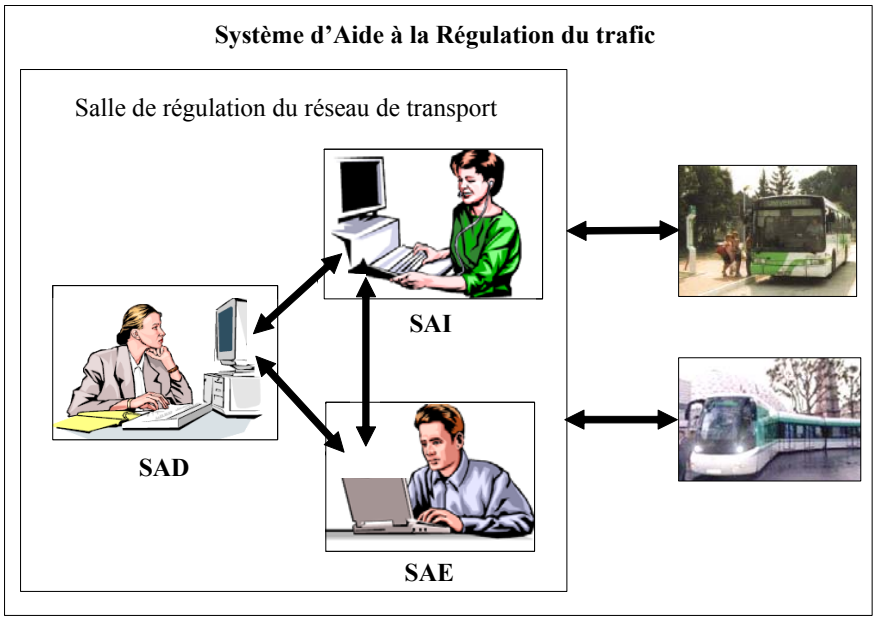



**Figure IV.1.** Décomposition du Système d'Aide à la Régulation du Trafic (SART) en trois modules : SAE, SAD et SAI

Ces différents modules de la régulation sont liés les uns aux autres, la régulation en inter-station ayant en particulier une influence sur la régulation dans les nœuds de correspondances. De même les flux de passagers en station ont un impact important sur la régulation du trafic en ligne. L'information voyageurs a une influence sur le comportement des usagers et par conséquent sur la régulation du trafic.

Notre contribution au sein du LAMIH dans le projet SART concerne l'étude du Système d'Aide à l'Information (SAI). Cette étude consiste à effectuer la spécification et la conception du SAI qui débouchera sur le développement d'une maquette permettant d'une part l'information des voyageurs, se trouvant dans les stations et/ou dans les véhicules ; et d'autre part l'aide des régulateurs relativement à la visualisation, l'édition, la création et transmission de messages aux utilisateurs du réseau.

Par ailleurs, le projet SART nous servira de cadre de validation pour l'architecture ainsi que l'outil d'aide à l'évaluation basé sur le mouchard électronique que nous avons proposé au chapitre précédent.

Les différents sous-systèmes du système d'aide à la régulation sont maintenant brièvement décrits.

## II.1.1. Système d'Aide à l'Exploitation (SAE)

Le SAE a pour rôle de centraliser les informations relatives à l'exploitation (horaires, retards, avances, messages alertes, etc.) et de permettre leur gestion [Fayech, 2003]. Dans notre cas, le SAE consiste en un simulateur assurant le suivi du trafic au sein du réseau urbain. Il détermine automatiquement et en temps réel, l'emplacement des véhicules et les différentes anomalies présentes sur le réseau.

## II.1.2. Système d'Aide à la Décision (SAD)

Le SAD est une solution informatique utilisée pour aider les régulateurs dans leurs prises de décision et la résolution de problèmes complexes [Shim *et al.*, 2002]. L'aide à la décision est l'étude de la manière dont les décisions sont actuellement prises, et de celle de les améliorer pour qu'elles puissent être prises avec plus de rigueur et d'efficacité [Fayech, 2003]. Le SAD analyse les perturbations survenues dans le réseau (données par le SAE) et fournit des décisions possibles au régulateur ; celui-ci choisit la solution la plus adéquate (selon son expérience) ou bien demande d'autres solutions tout en élaguant des contraintes d'exploitation. Le SAD permet d'alléger la tâche du régulateur mais ne prétend en aucun cas remplacer le régulateur [Mesghouni *et al.*, 2001 ; Laichour *et al.*, 2001 ; Chihaib *et al.*, 2001 ; Bouamrane *et al.*, 2005]

## II.1.3. Système d'Aide à l'information (SAI)

Le SAI a pour objectif l'information des voyageurs à l'intérieur des véhicules et en attente aux arrêts. Une partie de ces informations peut être transmise aux voyageurs de façon automatique par le SAE ; d'autres informations sont transmises aux voyageurs à l'initiative du régulateur via le SAI. Par ailleurs, le SAI a pour objet d'aider les régulateurs à visualiser, éditer, créer et transmettre les messages aux utilisateurs du réseau [Trabelsi *et al.*, 2005].

Afin de mieux comprendre les besoins de la SEMURVAL en termes de système d'aide à l'information, une analyse de l'existant à la SEMURVAL s'avère nécessaire.

## II.2. Analyse de l'existant

Notre analyse de l'existant se base principalement sur le rapport final de la première phase du projet SART [Ezzedine *et al.*, 2001]. L'étude du poste d'information Valenciennois pour l'exploitation a montré qu'il est composé de cinq blocs principaux :

- **Le bloc système temps réel** : il a pour fonction la réception, le traitement et la transmission en temps réel des informations issues de l'exploitation. Cette fonction est assurée par un logiciel SAE conçu par la société SEMA en 1986 ;
- **Le bloc système de communication** : il a pour finalité l'échange des données et la phonie entre le système temps réel et les autres périphériques. Ce système est mis en œuvre par une liaison bidirectionnelle par :
  - faisceau hertzien : canal de données relatif à l'information des voyageurs à bord des bus et canal phonie pour la communication avec les chauffeurs,
  - modem et réseau interne : canal de données relatif à l'information aux garages et des voyageurs en station ;
- **Le bloc PC d'exploitation** : il a pour rôle la mise en œuvre opérationnelle du SAE ;
- **Le bloc système embarqué bus** : il remplit les fonctions suivantes :
  - la localisation des voitures,
  - la transmission des données et la liaison phonie (communication radio chauffeur et régulateur),

- l'affichage du prochain arrêt,
- l'affichage des avances/retards destinés aux chauffeurs ;

- **Le bloc système information voyageurs** : qui nous concerne, se limite :
  - dans la majorité des arrêts, à l'affichage des heures prévues de passage (information par rapport à des horaires théoriques) ; en conséquence il y a absence quasi-totale d'informations sur les retards,
  - à l'affichage du prochain arrêt à l'intérieur des bus. Cette façon de faire manque de lisibilité ; il aurait été préférable de la remplacer par le plan de la ligne et un défilement du point d'arrêt lié à ce plan. De plus, on constate qu'il n'y a aucune attention particulière envers les personnes illettrées ou handicapées (non-voyants ou malvoyants) en raison d'une absence totale de messages vocaux.

La transmission de ces informations aux voyageurs peut être effectuée d'une façon automatique (affichage de l'heure du prochain bus dans les stations, affichage des retards/avances), ou encore à l'initiative du régulateur, qui a la possibilité d'éditer, de créer ou d'envoyer des messages aux stations ou à l'intérieur des véhicules, suite à une perturbation quelconque survenue sur le réseau (manifestation, accident…).

## II.3. Conclusion

Nous avons présenté notre cadre d'application qui est le projet SART. Ce projet fait intervenir plusieurs laboratoires de recherche et se décompose en trois parties principales : le SAE (Système d'Aide à l'Exploitation), le SAD (Système d'Aide à la Décision) et le SAI (Système d'Aide à l'Information). Notre participation au projet SART concerne le SAI. Après une analyse de l'existant, effectuée dans la section précédente, nous proposons dans la section suivante l'analyse, la modélisation, la spécification, et le maquettage du SAI.

# III. Analyse, Modélisation, Spécification et maquettage du SAI à architecture à base d'agents

## III.1. Analyse du SAI

La phase d'analyse du SAI est une phase importante dans le cycle de développement d'une application interactive en génie logiciel [Kolski, 1997]. Nous abordons l'analyse du SAI en trois points différents : (1) l'information des voyageurs en stations et à l'intérieur des véhicules, (2) les évènements liés aux SAI, (3) le besoin informationnel pour la communication entre le SAI, le SAD et le SAE.

### III.1.1. Information des voyageurs aux stations et à l'intérieur des véhicules

Afin d'assurer l'information des voyageurs aux stations et à l'intérieur des véhicules, le système d'aide à l'information doit répondre à cinq questions principales [Ezzedine *et al.*, 2001] :

- **Quelle information délivrer ?**

  Pour répondre à cette question, on distingue essentiellement deux classes d'information :
  - l'information temporelle car les usagers sont très sensibles aux temps d'attente, aux horaires de départ et surtout aux horaires de correspondance,
  - l'information circonstancielle qui consiste en l'envoi de messages spécifiques afin d'avertir les passagers des difficultés sur le réseau, en cas de perturbations (transfert d'arrêts à cause de travaux, panne de bus, arrêts non desservis en cas de déviation, etc.).

  A ces deux types d'informations s'ajoutent celles en lien avec la destination des voitures, les arrêts desservis par chaque voiture, les points de correspondance, etc.

- **Où la délivrer ?**

  L'analyse de l'existant a montré que l'information jugée pertinente doit être acheminée aux points d'arrêts et à bord des véhicules (bus ou tramway) concernés. Toutefois, l'information voyageurs à l'intérieur des véhicules (bus et tramway) est différente de celle aux points d'arrêts. En effet, l'information à bord du véhicule se concentre beaucoup plus sur le lieu (permettre aux voyageurs de se repérer, exemple : affichage du prochain arrêt) tandis que l'information aux points d'arrêts est fortement liée au temps (ex : horaire réel ou estimé d'attente).

- **Comment la délivrer ?**

  La manière de délivrer l'information désigne l'ensemble du dispositif mis en œuvre pour la transmission de l'information vers les voyageurs. On distingue trois possibilités :
  - affichage sur écrans en stations et dans les véhicules (bus ou tramway),
  - messages vocaux par haut-parleurs,
  - affichage directement par les agents aux points d'arrêts et par les chauffeurs à l'intérieur des véhicules.

- **Quand la délivrer ?**

  Répondre à cette question revient à définir des pré-conditions pour chaque information. Ainsi, une information sera destinée à l'affichage si et seulement si toutes ses pré-conditions sont validées. Exemple : l'affichage du message «arrêt Wilson est transféré au 3 rue des canaux » ne sera actif que si par exemple l'événement travaux est validé.

- **Pendant combien de temps ?**

  L'information est arrêtée lorsqu'elle devient obsolète. Exemple : normalement, le message cité précédemment ne doit plus exister à partir du moment où les travaux sont terminés.

## III.1.2. Événements liés au SAI

Le SAI est sensible à deux types d'événements[32] visibles (figure VI.2) ; ceux-ci peuvent considérablement changer son état [Trabelsi *et al.*, 2005] :

- **les événements *régulateur*** : correspondent aux commandes assignées par les régulateurs depuis le poste de contrôle/commande. Ils peuvent correspondre à :
  - une modification de la topologie du réseau (supervision d'un point d'arrêt, etc.),
  - des messages envoyés par le régulateur,
  - une modification du tableau de marche[33].
- **Les événements *réseau*** : ils proviennent directement du terrain. Leurs origines peuvent être :
  - une perturbation survenu dans le réseau du à une panne d'un véhicule, des travaux, une manifestation, etc.
  - un message envoyé par un conducteur d'un véhicule pour informer le régulateur d'un incident (véhicule en surcharge, circulation difficile).

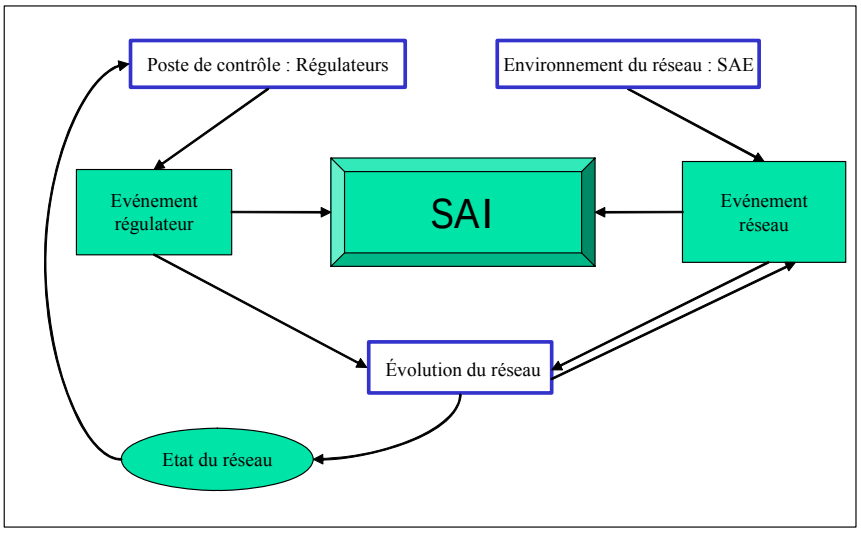



**Figure IV.2.** Événements liés au SAI

A la réception de ces événements, le SAI effectue une analyse des nouvelles informations reçues. Les informations pertinentes serviront à la mise à jour du système et par conséquence à l'évolution du réseau. Lors du maquettage du SAI (Cf. § III.4) et pour des raisons de faisabilité, les événements réseau seront simulés via un « mini » simulateur du réseau de transport urbain intégré au SAI.

[32] Ces événement seront exploités plus loin (Cf. § III.3.3, Tableau IV.2) pour la modélisation des services des *agents d'interface*.

[33] Le Tableau de Marche définit la fréquence de la ligne à chaque moment de la journée, le nombre de bus total dont dispose la ligne, le nombre de bus qui rouleront sur ligne le matin, le soir ainsi que le nombre de bus en relais (servant en cas de panne, accident...). Le tableau de marche est relié à la charge de la ligne.

## III.1.3. Besoin informationnel du SAI et connexion avec le SAD et le SAE

Comme nous l'avons déjà vu précédemment, le système d'aide à la régulation du trafic terrestre du réseau urbain de la ville de Valenciennes est constitué de trois sous-systèmes. Afin d'assurer le fonctionnement global du système SART et de répondre aux attentes des régulateurs, ces trois sous-systèmes doivent communiquer entre eux. La figure IV.3 spécifie les liaisons entre les trois modules SAI, SAE et SAD. Chacun de ces Modules est développé par une équipe de recherche (Cf. § I). Pour communiquer avec le module SAD et SAE, nous avons introduit au SAI un agent *intermédiaire* (Cf. figure IV.3). Cet agent *intermédiaire* évite de surcharger les agents du SAI[34] et les libère de la tâche de communication avec les modules SAD et SAE.

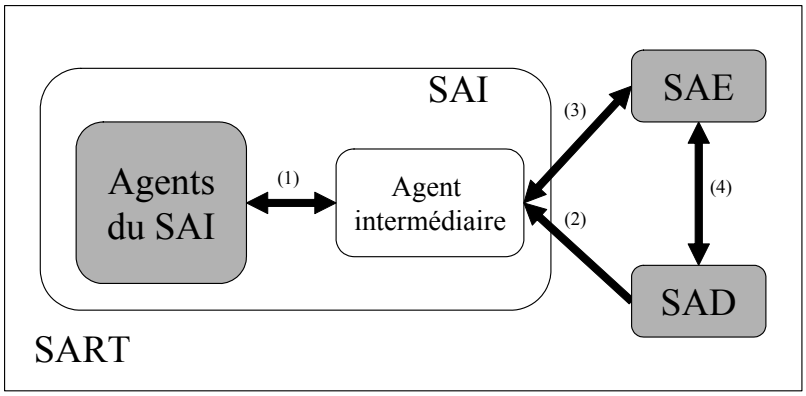



**Figure IV.3.** Connexion SAI/SAE/SAD

Le besoin informationnel (données nécessaires au système pour qu'il exécute correctement ses traitements) du SAI est représenté sur la figure IV.3 par les flots (2) et (3) qui désignent respectivement les échanges SAI/SAD et SAI/SAE. Ainsi les informations attendues du SAI sont des :

- **Informations attendues du SAD (2) :** on distingue trois types d'information :
  - les décisions prises par les régulateurs en salle de régulation. Ces décisions sont transmises au SAI afin de mettre à jours les vues concernées,
  - l'impact des décisions prises par le régulateur sur le réseau ; par exemple, si le régulateur décide de mettre hors service un véhicule, le SAD doit en informer le SAI afin de prendre ce changement en compte,
  - l'horaire prévu pour le retour à la normale.
- **Informations attendues du SAE (3) :** on distingue deux types d'information :
  - information concernant les caractéristiques de chaque véhicule mis en circulation sur le réseau. Ces caractéristiques concernent le nom du prochain arrêt desservi par le véhicule, l'horaire théorique du passage devant le prochain arrêt et le retard du véhicule.
  - information concernant les perturbations sur le réseau. pour chaque perturbation survenue, le SAD envoie au SAI la nature de la perturbation (panne de voiture, travaux, manifestation,…), son impact sur le réseau (lignes concernées, durée, canton(s) concerné(s)) et son origine (perturbation planifiée ou perturbation accidentelle).

[34] Nous verrons plus loin que le SAI est constitué d'agents d'interface (Cf. § III.3).

Chaque module est développé sous un langage particulier : le module SAE est développé avec Quest[35], le module SAD est développé avec Visual Basic et C, le module SAI est développé avec Borland C++ builder. Cette diversification de langages de programmation nous a poussés à choisir le mécanisme de socket[36] pour la communication entre les différents modules.

## III.2. Modélisation du SAI

Dans toute application interactive, l'utilisation d'un formalisme décrivant la dynamique du système est nécessaire. En effet, la modélisation a pour but de donner une description de l'ensemble des états possibles d'un système ainsi que les séquences de commandes conduisant à chacune d'entre elles [Abed, 2001 ; Palanque et Bastide, 1997].

La modélisation est d'autant plus nécessaire si elle offre des techniques d'analyse formelles permettant de prouver des propriétés sur la conception avant son implémentation. Ainsi, nous avons fait appel à l'outil réseaux de Petri (RdP) et plus particulièrement les RdP interprétés pour la modélisation, d'une part du comportement simplifié de l'interaction SAI / régulateur, et d'autre part la modélisation du fonctionnement interne du SAI.

### III.2.1. Modélisation des interactions SAI/Régulateur

Dans la salle de contrôle, le régulateur interagit avec le SAI pour mener à bien sa tâche de régulation. Il peut créer, éditer et envoyer des messages aux voyageurs en stations et/ou à bord des véhicules. La partie 1 de la figure IV.4 montre les interactions possibles entre SAE, SAD et SAI ainsi que le traitement des événements reçus par le SAI. La partie 2 quand à elle, illustre les différentes interactions possibles entre le régulateur et le SAI. Notons que le RdP ne s'intéresse uniquement qu'à la partie interactive et donc uniquement aux informations pertinentes qui transitent obligatoirement par le régulateur.

Comme le montre la figure IV.4, les événements venant du SAE, liés à l'information temps et/ou à l'information circonstancielle, sont pris en compte par le SAI par le franchissement respectif des transitions T1 et T'1. Après obtention de complément d'information du SAD et/ou du SAE (franchissement des transitions T3 et/ou T4), plusieurs tâches sont activées (mise à jour du tableau de marche, affichage automatique de certains messages) puis traitées avant d'être présentées au régulateur (franchissement des transitions T6 et T7). Ces informations qui désignent l'information voyageur à l'intérieur des voitures (bus ou tramway) et l'information voyageurs en stations sont sélectionnées une par une par le régulateur (franchissement des transitions T8 et/ou T9). Après l'évaluation de l'information sélectionnée, le régulateur a le choix de l'élaguer (transition T10), de l'envoyer (transition T11) ou bien de la modifier avant l'envoi (séquence T12-T14) de cette dernière. Par ailleurs, le régulateur se voit la possibilité d'envoyer d'autres messages, venant de son intention, par l'intermédiaire de la séquence de franchissement des transitions T13 et T15.

[35] QuEST (Queuing Event Simulation Tool » est un logiciel de création, gestion et contrôle des systèmes de production. Il permet de simuler et d'analyser les flux de production dans un environnement 3D virtuel (Cf. le site : http://www.quest.com/)

[36] Socket : Communication logique entre deux systèmes reliés au réseau Internet.

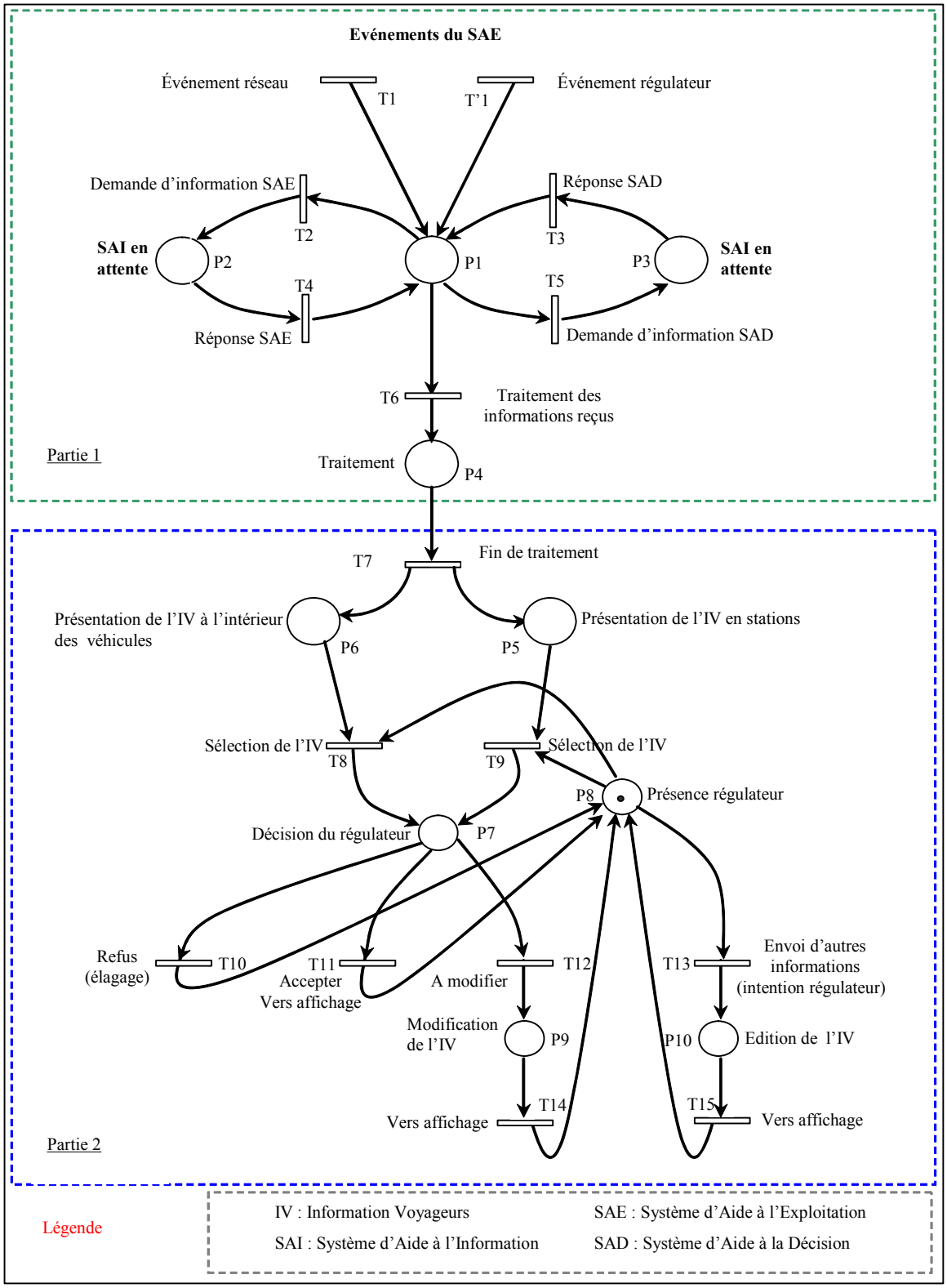



**Figure IV.4.** Modélisation de l'interaction SAI / Régulateur

Cette modélisation ne permet pas la distinction des deux modes de fonctionnements du SAI, à savoir le fonctionnement en mode normal et le fonctionnement en mode perturbé. Nous présenterons dans la section suivante une modélisation plus détaillée du SAI (toujours en utilisant les réseaux de Pétri interprétés).

## III.2.2. Modélisation de fonctionnement du SAI

Le SAI suit deux modes de fonctionnement différents :

- le fonctionnement en mode normal,
- le fonctionnement en mode perturbé.

### III.2.2.1. Fonctionnement en mode normal

Le fonctionnement en mode normal est représenté à droite de la figure VI.5. La place P1 représente le système d'aide à l'information en attente. En mode de fonctionnement normal, le SAI peut réagir à deux actions possibles (franchissement de T1 ou T2) :

- le franchissement de T1 représente une demande d'information par le régulateur. Cette demande est traitée directement par le SAI sans intervention particulière du SAD ou du SAE. Un exemple de cette demande d'information peut être la visualisation d'une ligne.
- le franchissement de T2 représente une demande d'information par le régulateur via le SAE. Dans ce cas, la place P2 modélise l'attente du SAI de la réponse du SAE. Le franchissement de T3, symbolisant la réponse du SAE, assure l'affichage de l'information modélisée par la place P3.

En mode normal de fonctionnement, le régulateur peut effectuer un envoi de messages aux véhicules et aux stations suite à sa propre initiative (franchissement de T12).

### III.2.2.2. Fonctionnement en mode perturbé

Le fonctionnement en mode perturbé est représenté par la figure IV.5. Suite à la détection d'une perturbation (franchissement de la transition T18), le SAE prépare les informations nécessaires pour le SAI (P14) et le SAD (P15). Les informations reçues par le SAI sont traitées (P4) pour distinguer les informations pertinentes.

Les informations en provenance d'une perturbation non grave, serviront à la mise à jour du SAI (P5). En cas d'une perturbation grave, le SAI peut réagir de deux façons :

- présenter directement les informations nécessaires au régulateur ;
- attendre la décision du SAD pour afficher les informations au régulateur.

La place P11 est une place virtuelle (une place non visible). Elle représente la décision du régulateur vis-à-vis des messages proposés :

- le franchissement de T14 représente le refus du message proposé.
- le franchissement de T15 est réalisé suite à l'acceptation du message proposé.
- le franchissement de T16 est effectué quand le régulateur décide de modifier le message proposé.

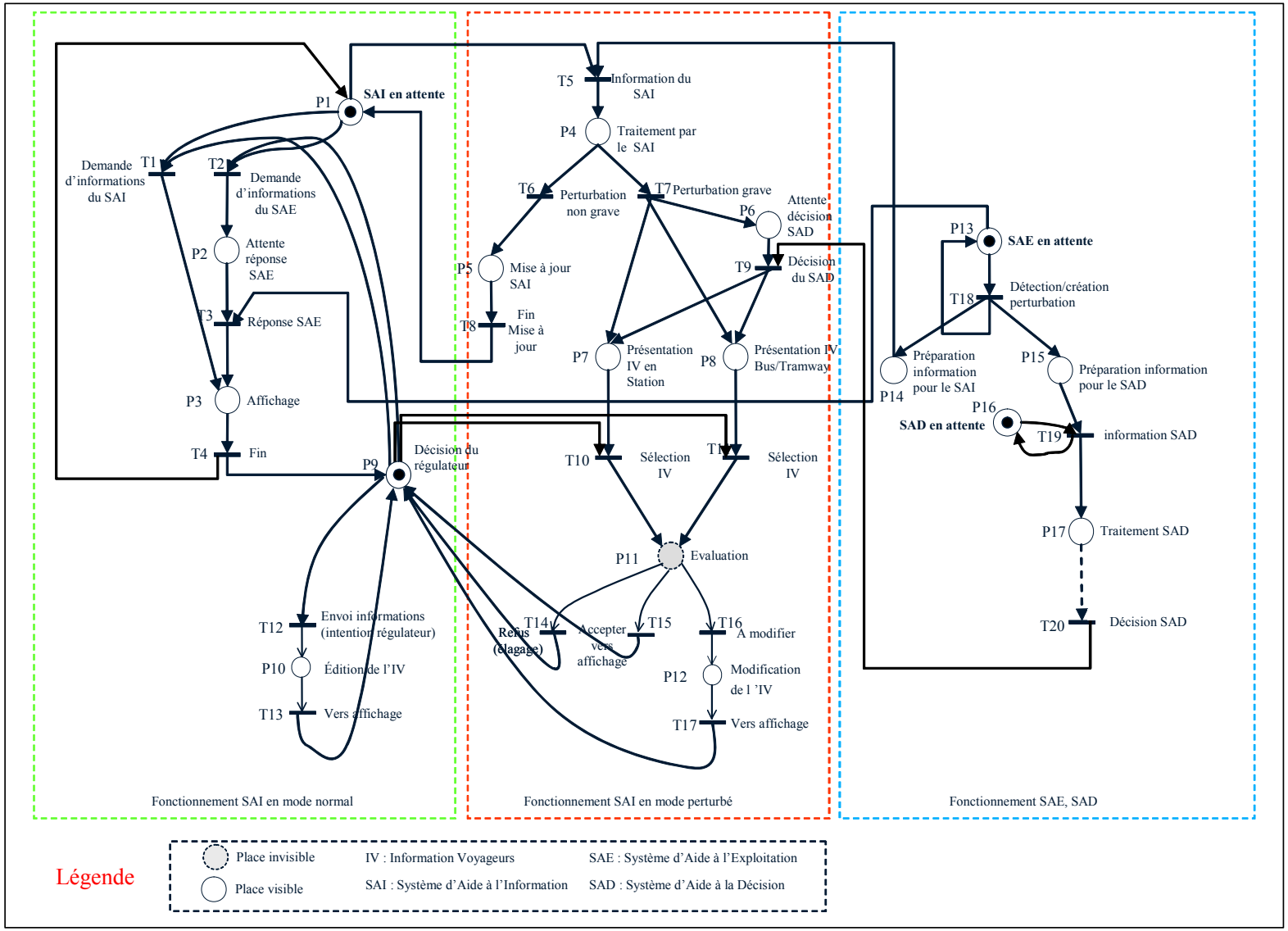



**Figure IV.5.** Modélisation du fonctionnement du SAI avec les réseaux de Petri interprétés

La modélisation du SAI par les réseaux de Petri interprétés a permis la compréhension du fonctionnement de celui-ci. Nous présentons dans la section suivante la spécification du SAI en se basant sur l'architecture à base d'agent que nous avons proposée au troisième chapitre.

## III.3. Spécification du SAI

Pour la spécification du SAI nous proposons d'utiliser l'architecture orientée agents que nous avons proposée au chapitre trois (Cf. chapitre trois, § III). Rappelons que cette architecture assure la séparation des agents en trois classes appelées : *interface avec l'application* (composé d'agents appelés *agents d'application*), *contrôleurs de dialogue* (composé d'agents appelés *agents contrôleurs de dialogue*), *présentation* (composé d'agents appelés *agents d'interface*). Dans la mesure où nous nous intéressons particulièrement aux *agents d'interface*, nous ferons d'abord une brève description des *agents d'application* et des *agents contrôleurs de dialogue* avant de passer à la spécification des *agents d'interface*.

### III.3.1. Spécification des agents d'application

Comme nous l'avons déjà vu au chapitre précédent (Cf. Chapitre 3, § III.1.3), les *agents d'application* manipulent les concepts du domaine et ne sont pas accessibles directement par l'utilisateur. Ils assurent entre autre le bon fonctionnement de l'application et l'émission en temps réel d'informations nécessaires au déroulement des activités des autres agents. Ainsi les *agents d'application* ont pour rôle principal de gérer l'information voyageur en véhicule (bus/tramway) et de prévoir l'affichage automatiquement de l'information comme par exemple : l'horaire de passage des véhicules aux stations, le retard de passage d'un véhicule par rapport à son horaire prévu, etc.

D'après l'analyse du SAI et des besoins des régulateurs (Cf. § III.1), nous avons identifié six classes d'*agents d'application* :

- *Agent d'application Véhicule* : regroupe toutes les informations relatives à un véhicule (Nom du conducteur, numéro de service, message à destination du conducteur, message à destination des voyageurs, etc.),
- *Agent d'application Station* : regroupe toutes les informations relatives à une station (Nom de la station, station de correspondance ou station simple, à quelle(s) ligne(s) appartient la station, etc.),
- *Agent d'application Message* : regroupe toutes les informations relatives aux messages à transmettre aux voyageurs en stations et aux véhicules (texte du message, type de message, taille du message, etc.),
- *Agent d'application Etat_Ligne* : regroupe toutes les informations relatives à une ligne (les stations qui composent la ligne, les véhicules qui appartiennent à la ligne, etc.),
- *Agent d'application Etat_Trafic* : regroupe toutes les informations relatives à l'état du trafic (les lignes à afficher, le nombre maximal de ligne à superviser, etc.),
- *Agent d'application Vue_Globale* : regroupe toutes les informations relatives au réseau (nombre de lignes, nombre de véhicules en réserve, etc.).

Le réseau de transport valenciennois est composé de plusieurs lignes ; nous avons donc associé à chaque ligne un *agent d'application Etat_Ligne*. Parallèlement, nous avons associé à

chaque véhicule et à chaque station du réseau respectivement un *agent d'application Véhicule* et un *agent d'application Station*. Un seul agent d'application Message est nécessaire, il gère la création et l'enregistrement des message prés-définis. L'*agent d'application Etat_Trafic* enregistre le nombre maximal de lignes à superviser ainsi que les lignes en cours de supervision. L'*agent d'application Vue_Globa*le contient les informations nécessaires pour rendre compte de l'état de tout le réseau. Notons que tout ces agents sont des agents réactifs au sens où on l'entend en IHM depuis les années 80 (Cf. chapitre 1, § III.1.1).

Par ailleurs les *agents d'application* assurent la communication avec le SAE et le SAD. Ils sont connectés à l'agent intermédiaire lui-même connecté au SAE et au SAD (Cf. figure IV.3).

## III.3.2. Spécification des agents contrôleurs de dialogue

Les *agents contrôleurs de dialogue* servent d'intermédiaire pour ce qui est de l'interaction entre les *agents d'interface* et les agents d'application. Ils ont un rôle double ; ils permettent de :

- propager les modifications observées au niveau des *agents d'application* sur les *agents d'interface*,
- propager les actions de commande du régulateur (interaction avec les *agents d'interface*) vers *les agents d'application*.

Afin d'assurer la communication entre les *agents d'application* et les *agents d'interface*, on associe à chaque *agent d'application* un *agent contrôleur de dialogue*. Ainsi le rôle des *agents contrôleurs de dialogue* consiste à transmettre les requêtes entre les *agents d'application* et les *agents d'interface*. Par exemple si un *agent d'application* reçoit une information de retard d'un véhicule par le SAE, il transmet directement l'information à l'*agent contrôleur de dialogue*. Ce dernier transmet l'information à *l'agent d'interface* correspondant.

## III.3.3. Spécification des agents d'interface

Les *agents d'interface* assurent la gestion de l'affichage et le suivi du processus et des commandes envoyées par le régulateur. Ces agents regroupent différents éléments interactifs afin de former des interfaces aux informations pertinentes pour l'utilisateur (le régulateur) : le synoptique d'une ligne, les informations voyageurs en station, les informations voyageurs en bus/ tramway, etc.

Nous avons identifié six types d'*agents d'interface* chargés des interactions directes avec le régulateur humain. Ces agents se présentent sous forme de fenêtres interactives. Le régulateur peut interagir avec ces agents via les différentes fonctionnalités possibles dans les fenêtres comme par exemple : les boutons, les zones d'édition, les images, etc. Les six types d'*agents d'interface* sont les suivant:

- *Agent d'interface Etat_Trafic* : cet agent surveille en temps réel les retards des véhicules. En effet, selon l'analyse du SAI existant (CF § II.2), les informations qui font l'objet d'une surveillance prioritaire sont les retards des véhicules. L'*agent d'interface Etat_Trafic* gère aussi les incidents rencontrés par un véhicule

sur la ligne ainsi que les messages d'alerte émis automatiquement par le système à l'intention du régulateur (pannes sur véhicules ou en station).

- *Agent d'interface Etat_Ligne* : cet agent assure la supervision d'une ligne du réseau. Une ligne est constituée de stations et de véhicules. Les informations nécessaires à ce niveau de supervision sont les positions des véhicules relatives aux stations ainsi que les retards des véhicules.
- *Agent d'interface Station* : il affiche les informations relatives à une station. C'est le niveau le plus fin de la supervision ; le régulateur doit y retrouver toutes les informations nécessaires concernant une station. On trouvera donc les informations nominatives de la station ainsi que ses caractéristiques, les horaires de passage en fonction des différentes lignes et directions, les différentes alertes et messages présentés de manière détaillée.
- *Agent d'interface Véhicule* : de manière analogue à la vue détaillée de station, on y retrouve les informations relatives à un véhicule, à savoir : nom, type, nom du conducteur, numéro de service, retard actuel, messages à destination du conducteur et des passagers à bord du véhicule.
- *Agent d'interface Message* : en interagissant avec cet agent, le régulateur gère l'envoi et la modification des messages à destination des stations et véhicules. Cet agent assure aussi l'envoi de messages pré-enregistrés.
- *Agent d'interface Vue_Globale* : il assure une vue globale du réseau supervisé. Le régulateur possède ainsi une vision globale sur l'état du réseau et peut facilement localiser un véhicule ou visualiser une zone de perturbation.

Les *agents d'interface* sont en contact direct avec le régulateur (visibles pour le régulateur). Ils prennent en charge les éléments d'interface qui apparaissent sur l'écran et les différentes interactions physiques avec le régulateur. Ces agents se coordonnent les uns les autres pour intercepter les commandes de l'utilisateur et composer pour lui une présentation lui permettant une compréhension globale de la situation courante du réseau supervisé.

Comme nous l'avons vu au chapitre 3 (Cf. Chapitre 3, § III.2.1), tous ces agents communiquent entre eux pour répondre aux actions de l'utilisateur. Cette communication est considérée comme des services entre agents. Nous avons ainsi défini (Cf. Chapitre 3, § III.2.2) un *agent d'interface* par l'ensemble de ses services. Rappelons qu'un agent (ensemble de n services) est défini comme étant un quadruplet {E, C, R, P} où :

- $E = \{e_1, e_2..., e_i,\ldots, e_n\}$, ensemble des évènements déclencheurs de services de l'agent,
- $C = \{c_1, c_2,..., c_i, ..., c_n\}$, ensemble des conditions nécessaires à l'établissement des services,
- $R = \{r_1, r_2, ..., r_i, ..., r_p\}$, ensemble des ressources éventuellement nécessaires à l'établissement des services,
- P = Propriétés des services. Chaque service résulte par l'exécution :
  - Soit d'une action visible acv (action qui porte sur l'agent lui-même) ; on parle alors de propriété interne du service.

- Soit d'une action non visible acn (action qui porte sur un autre agent) ; on parle alors de propriété externe du service.

Nous avons modélisé les six *agents d'interface* déjà définis avec les RdP agent. Cette modélisation nécessite la définition, pour chaque agent, des services, évènements, conditions, ressources, actions visibles et non visibles.

Nous prenons ici comme exemple l'agent Etat_Trafic. La modélisation des autres agents est mise en Annexe (Cf. Annexe 2).

Les tableaux IV.1 à IV.6 synthétisent toutes les informations concernant les services, les évènements, les conditions, les ressources, les actions visibles et les actions invisibles pour l'*agent d'interface Etat_Trafic*.

Le tableau IV.1 illustre les différents services qui constituent l'*agent d'interface Etat_Trafic*. Le premier indice désigne le numéro du service et le deuxième indice désigne l'identificateur de l'agent.

| Identifiant du service | Services | Description |
|---|---|---|
| $s_{1,1}$ | Sélectionner_Ligne | Sélection d'une ligne pour sa supervision |
| $s_{2,1}$ | Désélectionner_Ligne | Désélection d'une ligne |
| $s_{3,1}$ | Pointer_Véhicule | Pointage avec la souris d'un véhicule |
| $s_{4,1}$ | Mise_En_Service_Véhicule | Ajout d'un véhicule au réseau |
| $s_{5,1}$ | Mise_Hors_Service_Véhicule | Suppression d'un véhicule du réseau |
| $s_{6,1}$ | Changer_Retard_Véhicule | Changement du temps de retard d'un véhicule |
| $s_{7,1}$ | Changer_Retard_Seuil_Véhicule | Changement du temps de retard supérieur à un seuil d'un véhicule |
| $s_{8,1}$ | Changer_Avance_Véhicule | Changement du temps d'avance d'un véhicule |
| $s_{9,1}$ | Changer_Avance_Seuil_Véhicule | Changement du temps d'avance supérieur à un seuil d'un véhicule |
| $S_{10,1}$ | Visualiser_Ligne | Clic sur une ligne |
| $S_{11,1}$ | Agrandir_Ligne | Faire un zoom avant sur la vue de l'agent Etat_Ligne |
| $S_{12,1}$ | Réduire_Ligne | Faire un zoom arrière sur la vue de l'agent Etat_Ligne |

**Tableau IV.1.** Services de l'*agent d'interface Etat_Trafic*

Le tableau IV.2 regroupe les événements qui peuvent agir sur l'agent *Etat_Trafic*. A titre d'exemple l'événement «$e_{4,1}$» (A*jout-Véhicule*) désigne l'ajout d'un véhicule au réseaux.

| Identifiant de l'évènement | Evènement |
|---|---|
| $e_{1,1}$ | Sélection_Ligne |
| $e_{2,1}$ | Désélection_Ligne |
| $e_{3,1}$ | Pointage_Véhicule |
| $e_{4,1}$ | Ajout_Véhicule |
| $e_{5,1}$ | Suppression_Véhicule |
| $e_{6,1}$ | Retard_Véhicule |
| $e_{7,1}$ | Retard_Seuil_Véhicule |
| $e_{8,1}$ | Avance_Véhicule |
| $e_{9,1}$ | Avance_Seuil_Véhicule |
| $e_{10,1}$ | Clic_Ligne |
| $e_{11,1}$ | Zoom_Avant |
| $e_{12,1}$ | Zoom_Arrière |

**Tableau IV.2.** Evénements déclencheurs des services de l'*agent d'interface Etat_Trafic*

Le tableau IV.3 regroupe les conditions que doit vérifier l'agent avant le déclenchement d'un service. Une condition peut être une condition simple ou une condition composée. Par exemple la condition $c_{6,1}$ est composée de deux sous conditions ; l'apparition de l'événement $e_{6,1}$ et le retard du véhicule compris entre 1 et 5 minutes.

| Identifiant de la condition | Conditions |
|---|---|
| $c_{1,1}$ | Apparition de l'évènement |
| $c_{2,1}$ | Apparition de l'événement |
| $c_{3,1}$ | Apparition de l'évènement |
| $c_{4,1}$ | Apparition de l'évènement |
| $c_{5,1}$ | Apparition de l'évènement |
| $c_{6,1}$ | Apparition de l'évènement et $1 \leq$ retard $< 5$ |
| $c_{7,1}$ | Apparition de l'évènement et retard $\geq 5$ |
| $c_{8,1}$ | Apparition de l'évènement et $1 \leq$ avance $< 5$ |
| $c_{9,1}$ | Apparition de l'évènement et avance $\geq 5$ |
| $c_{10,1}$ | Apparition de l'évènement |
| $c_{11,1}$ | Apparition de l'évènement |
| $c_{12,1}$ | Apparition de l'évènement |

**Tableau IV.3.** Conditions à vérifier par l'agent Etat_Trafic

Dans le tableau IV.4 sont représentées les différentes ressources nécessaires à l'agent pour l'exécution d'un service. Ces ressources sont des composants logiciels. Elles peuvent être une fenêtre d'information, un barre-graphe[37], etc.

| Identifiant de la ressource | Ressource |
|---|---|
| $r_{1,1}$ | Liste Barre_Graphe |
| $r_{2,1}$ | Liste Barre_Graphe |
| $r_{3,1}$ | Liste Barre_Graphe |
| $r_{4,1}$ | Barre_Graphe |
| $r_{5,1}$ | Barre_Graphe |
| $r_{6,1}$ | Barre_Graphe |
| $r_{7,1}$ | Boite_De_Dialogue |
| $r_{8,1}$ | Barre_Graphe |
| $r_{9,1}$ | Boite_De_Dialogue |

**Tableau IV.4.** Ressources requises par l'agent Etat_Trafic

Le tableau IV.5 regroupe les différentes actions visibles que peut exécuter l'agent Etat_Trafic suite au déclenchement de l'un de ses services.

| Identifiant de l'action visible | Action Visible | Description |
|---|---|---|
| $acv_{1,1}$ | Afficher_Ligne | Afficher la ligne sélectionnée |
| $acv_{2,1}$ | Cacher_Ligne | Cacher la ligne sélectionnée |
| $acv_{3,1}$ | Colorer_Panel_Véhicule | Colorer le panel correspondant au véhicule pointé |
| $acv_{4,1}$ | Ajouter_Barre_Graphe_Véhicule | Ajouter le barre-graphe correspondant au véhicule ajouté |
| $acv_{5,1}$ | Supprimer_Barre_Graphe_Véhicule | Supprimer le barre-graphe correspondant au véhicule supprimé |
| $acv_{6,1}$ | Colorer_Barre_Graphe_Reatrd | Colorer le barre-graphe correspondante au véhicule avec le retard correspondant |
| $acv_{7,1}$ | Afficher_Message_Avance_Seuil | Afficher un message signalant une avance d'un véhicule supérieur au seuil |
| $acv_{8,1}$ | Colorer_Barre_Graphe_Avance | Colorer le barre-graphe correspondante au véhicule avec l'avance correspondante |
| $acv_{9,1}$ | Afficher_Message_Retard_Seuil | Afficher un message signalant un retard d'un véhicule supérieur au seuil |

**Tableau IV.5.** Actions visibles exécutées par l'agent Etat_Trafic

[37] Le barre-graphe est un composant logiciel qui représente le retard ou l'avance d'un véhicule. Ce composant est présenté plus loin lors du maquettage du SAI (Cf. Figure IV.9).

Le tableau IV.6 regroupe les différentes actions non visibles que peut exécuter l'agent Etat_Trafic suite au déclenchement de l'un de ses services.

| Identifiant de l'action non visible | Action Non Visible |
|---|---|
| $acn_{1,1}$ | Informer l'agent d'interface Etat_Ligne : $e_{1,2}$ |
| $acn_{2,1}$ | Informer l'agent d'interface Etat_Ligne : $e_{2,2}$ |
| $acn_{3,1}$ | Informer l'agent d'interface Etat_Ligne : $e_{3,2}$ |
| $Acn_{4,1}$ | Informer l'agent d'interface Etat_Ligne : $e_{7,2}$ |
| $acn_{5,1}$ | Informer l'agent d'interface Etat_Ligne : $e_{8,2}$ |
| $acn_{6,1}$ | Informer l'agent d'interface Etat_Ligne : $e_{1,2}$ |
| $acn_{7,1}$ | Informer l'agent d'interface Etat_Ligne : $e_{4,2}$ |
| $acn_{8,1}$ | Informer l'agent d'interface Etat_Ligne : $e_{5,2}$ |

**Tableau IV.6.** Actions non visibles exécutés par l'*agent d'interface Etat_Trafic*

Suite à l'apparition d'un événement, l'*agent d'interface Etat_Trafic* vérifie la condition liée à l'événement apparu pour déclencher une action visible et/ou une action invisible. Une condition peut être simple ou composée. L'apparition de l'événement peut être une condition ; dans ce cas l'apparition de l'événement suffit pour l'exécution d'une action par l'agent.

Afin de faciliter la spécification des *agents d'interface* avec les réseaux de Petri agents, nous avons développé un outil dont l'interface est visible à la figure IV.6. Cet outil permet la spécification des services, événements, conditions, ressources, actions visibles et actions non visibles pour chaque agent. A chaque action non visible est associé un événement d'un *agent d'interface*. La spécification des *agents d'interface* est stockée dans une base appelée Base de Spécification des Agents (BSA) (Cf. chapitre III, § IV.2).

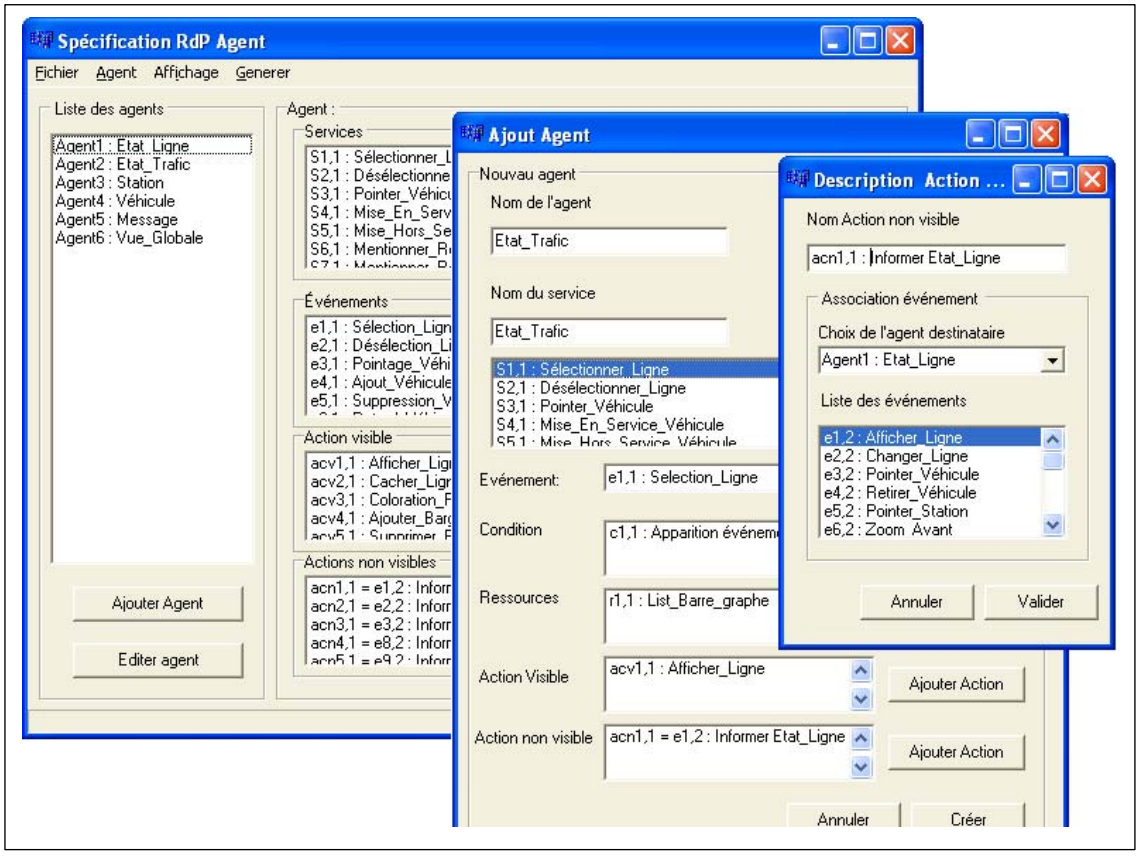



**Figure IV.6.** Outil interactif destiné à la spécification des *agents d'interface*

## III.4. Maquettage de l'interface de supervision

Nous rappelons que le but du SAI est de permettre au régulateur de visualiser, éditer, créer et transmettre des informations destinées aux voyageurs se trouvant dans les stations et/ou dans des véhicules.

Pour mener à bien sa tâche de régulation, le régulateur interagit avec les six *agents d'interface* : l'agent *Etat_Trafic*, l'agent *Etat_Ligne*, l'agent *Station*, l'agent *Véhicule*, l'agent *Message* et l'agent *Vue_Globale* [Trabelsi *et al.*, 2005]. Ceux-ci sont successivement présentés.

### III.4.1. Agent d'interface Etat_Trafic

La vue de l'*agent d'interface Etat_Trafic* représente de manière synthétique l'ensemble des retards des mobiles circulant sur le réseau (figure VI.7). Ainsi, avec la collaboration du SAE, il assure une surveillance, en temps réel, des retards/avances des véhicules dans le réseau supervisé.

Les valeurs du retard ou de l'avance sont affichées à l'aide de barre-graphe. Les barre-graphes permettent d'effectuer une lecture et une détection rapide des valeurs. Pour une meilleure visibilité les barres_graphes d'une même ligne du réseau sont regroupés en liste_barre-graphe (Cf. figure VI.7).

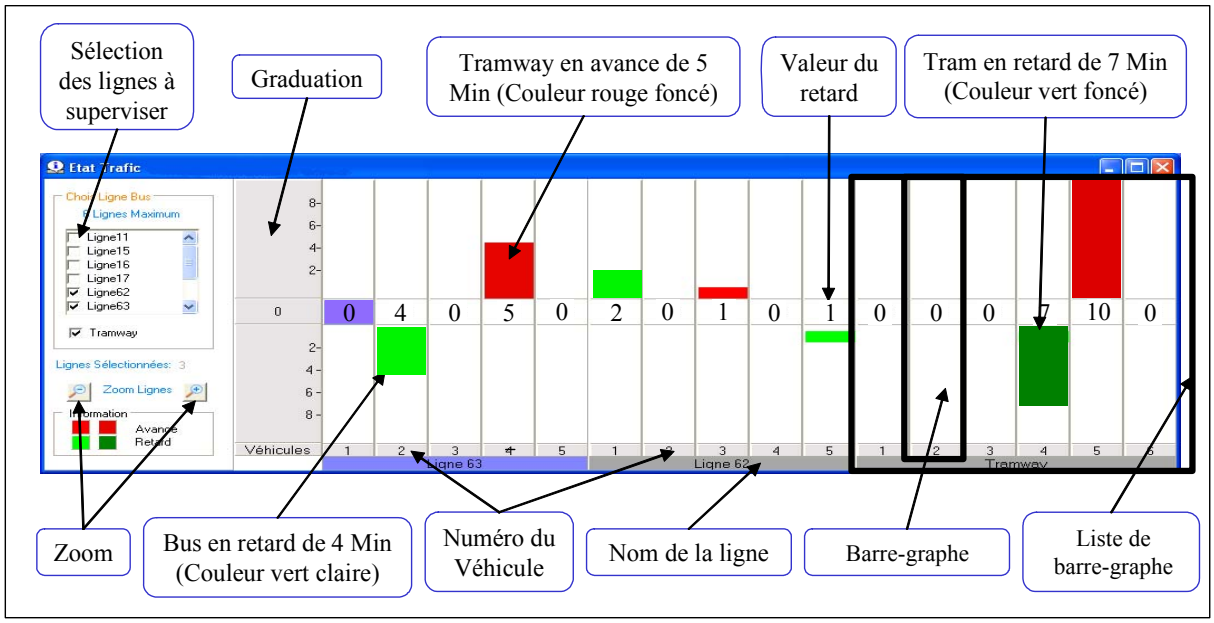



**Figure IV.7.** Agent d'*interface Etat_Trafic*

Comme nous l'avons vu dans la section précédente (Cf. Tableau IV.1), l'*agent d'interface Etat_Trafic* est constitué d'un ensemble de services $S_{1,1}$…$S_{1,12}$. Le déclenchement d'un service donne lieu à une action visible et/ou une action invisible. Afin de mieux expliquer la notion de service, que nous avons utilisée ci-dessus pour la spécification des *agents d'interface*, nous illustrons deux exemples de déclenchement de services : un service dont le déclanchement engendre seulement une action visible et un service dont l'activation engendre une action visible et une action invisible.

**Exemple 1 :**

La figure IV.8 montre un exemple de déclenchement du service $s_{7,1}$ « *Changer_Retard_Seuil_Véhicule* » de l'*agent d'interface Etat_Trafic*. Suite à l'apparition de l'événement $e_{7,1}$ « *Retard_Seuil_Véhicule* » l'agent vérifie la condition correspondante $c_{7,1}$ « *Apparition de l'événement et retard* ≥ *5* ». Si la condition est vérifiée, le service est déclenché ; l'action visible $acv_{7,1}$ est activé. L'activation de cette action exploite la ressource $r_{7,1}$ «*Boite_De_Dialogue*» et fait apparaître un message d'alerte à destination du régulateur.

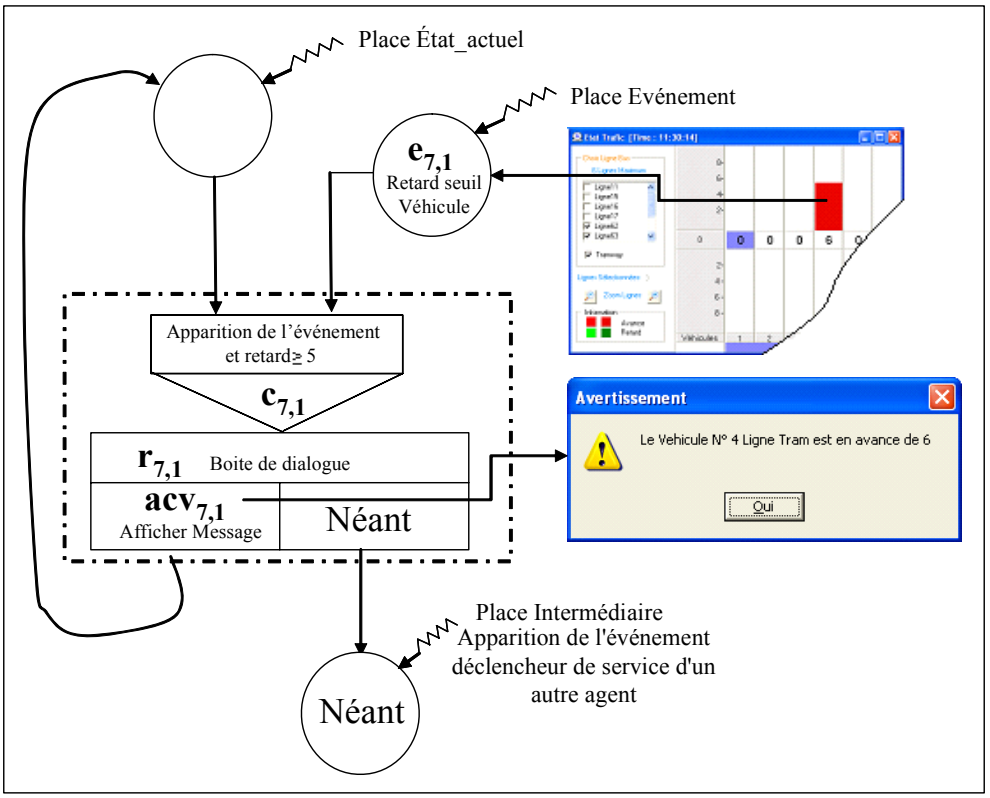



**Figure IV.8.** Exemple de déclenchement d'un service de l'agent *Etat_Trafic*

**Exemple 2 :**

La figure IV.9 montre un exemple de déclenchement du service $s_{5,1}$ « Supprimer_Véhicule » de l'*agent d'interface Etat_Trafic*. Suite à la suppression d'un véhicule (pour une raison de panne) du réseau, le SAE informe le SAI (via l'agent intérimaire Cf. III.1.3) de ce changement au sein du réseau. L'apparition de l'événement $e_{5,1}$ « Suppression_Véhicule » déclenche le service $s_{5,1}$ après avoir vérifié la condition $c_{5,1}$.

L'activation du service entraîne l'exécution de l'action visible $acn_{5,1}$ et l'action invisible $acn_{5,1}$. L'action visible $acv_{5,1}$ exploite la ressource $r_{5,1}$ « Barre_Graphe » pour la suppression du Barre-graphe correspondant au véhicule supprimé. L'action non visible $acn_{5,1}$ informe l'agent Etat_Ligne via l'événement $e_{8,2}$ de la suppression du véhicule. Quand l'*agent d'interface Etat_Ligne* reçoit cet événement, il vérifie la condition qui lui est associée et active le service $s_{8,2}$ « *Suppression_Véhicule* ».

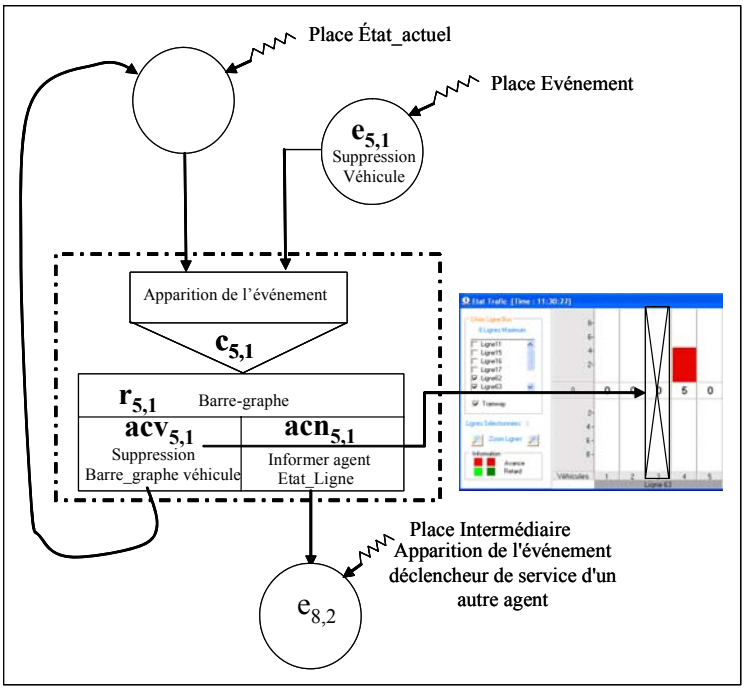



**Figure IV.9.** Exemple de déclenchement d'un service de l'agent *Etat_Trafic*

## III.4.2. Agent d'interface Etat_Ligne

La vue de l'*agent d'interface Etat_Ligne* se compose d'éléments graphiques, tels que stations, tronçons, véhicules, etc. Elle assure la visualisation de l'ensemble d'une ligne (Cf. figure VI.10).

Le clic sur un véhicule (enclenchement du service $e_{5,2}$ de l'agent Etat_Ligne) affiche directement la vue (fenêtre) de l'*agent d'interface Véhicule* qui prendra en charge les nouvelles interactions du régulateur.

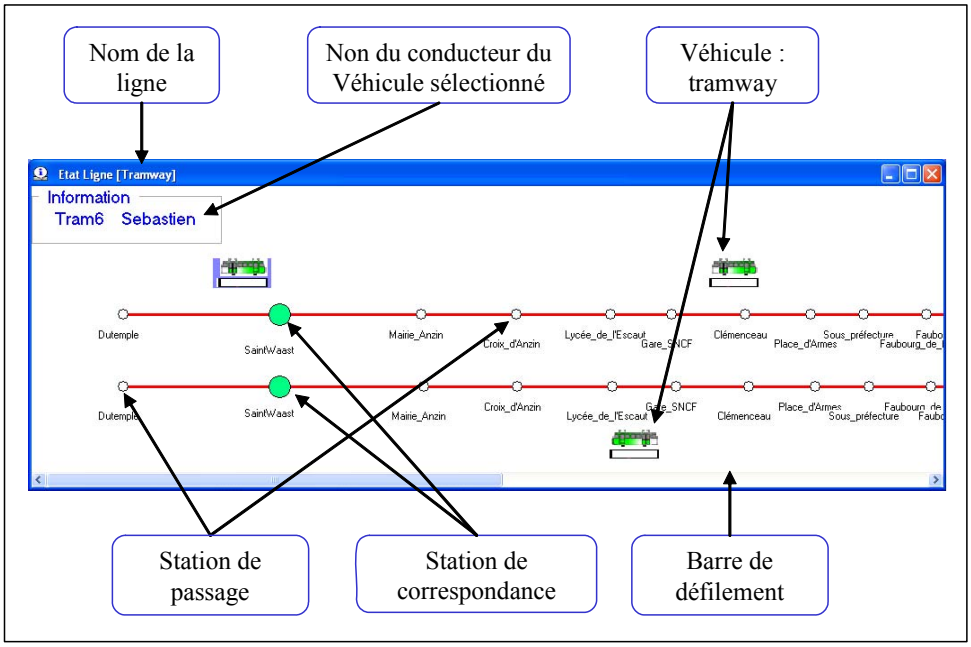



**Figure IV.10.** *Agent d'interface Etat ligne*

Les éléments graphiques visibles en figure IV.10 ont une représentation variable, en fonction des paramètres de l'entité qu'ils représentent : un type de véhicule correspond à une forme, un type de station correspond à une couleur (station de correspondance en couleur verte, station de passage en couleur blanche).

## III.4.3. Agent d'interface Station

La vue détaillée d'une station est accessible par action sur sa représentation associée dans la vue de l'*agent d'interface Etat_Ligne* (Cf. figure IV.11).

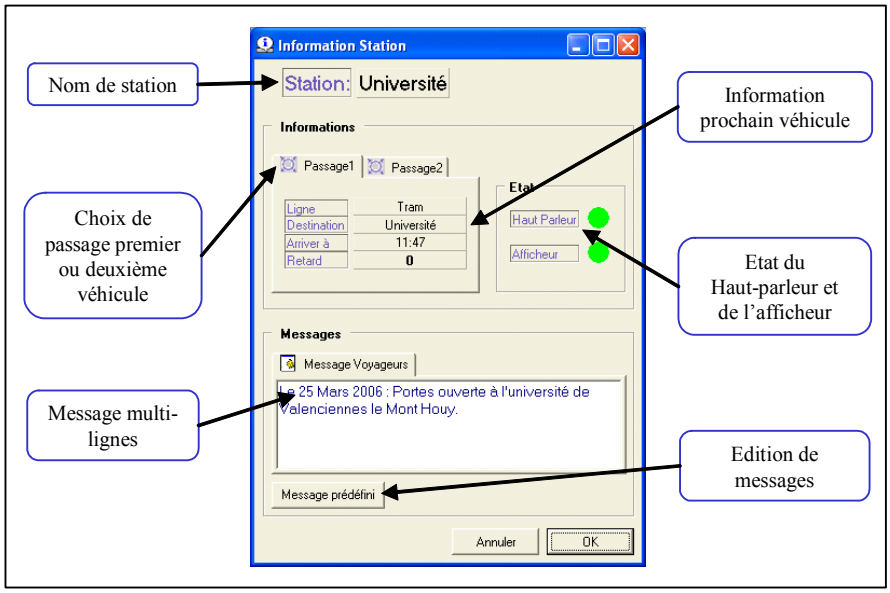



**Figure IV.11.** *Agent d'interface Station*

La vue se décompose en deux onglets indépendants : informations et message. L'onglet informations visualise l'état du haut-parleur et de l'afficheur d'une station. Il informe le régulateur sur le premier et le deuxième passage d'un véhicule. Les informations disponibles sont : le nom de la ligne, la destination du véhicule, l'heure d'arrivée et le retard/avance du véhicule. L'onglet message permet au régulateur l'envoi direct d'un message en destination de la station. Si le régulateur désire envoyer un message prédéfini à une station, il doit cliquer sur le bouton « *Message prédéfini* » ; il fait ainsi appel au service de l'*agent d'interface Message*.

## III.4.4. Agent d'interface Véhicule

La vue de l'*agent d'interface Véhicule* présente au régulateur les informations relatives à un véhicule (Cf. figure IV.12). Elle comporte globalement les mêmes éléments que la vue de l'*agent d'interface Station* à l'exception de la représentation des informations associées à son tableau de marche.

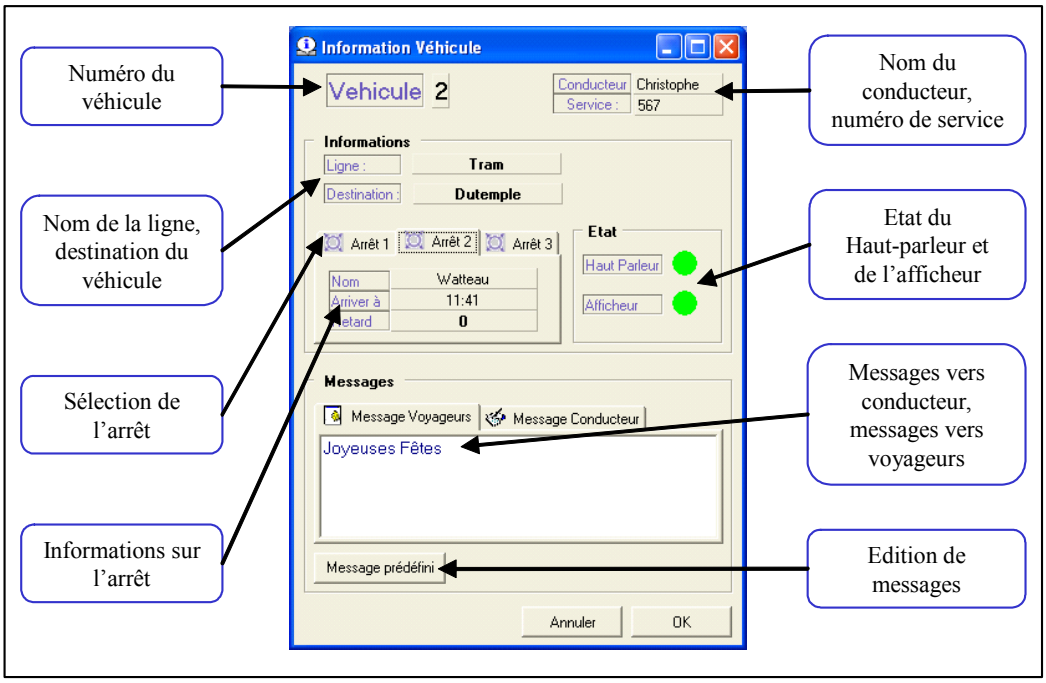



**Figure IV.12.** *Agent d'interface Véhicule*

Les deux onglets de la vue de l'*agent d'interface Véhicule* permettent aux régulateurs de visualiser les trois prochains arrêts du véhicule. Les informations disponibles pour chaque station sont : le nom, l'heure d'arrivée et le retard/avance par rapport au tableau de marche.

La vue de l'*agent Véhicule* permet au régulateur de communiquer unidirectionnellement avec le conducteur du véhicule ainsi que les passagers à bord en utilisant l'onglet *Messages*.

## III.4.5. Agent d'interface Message

La vue de l'*agent d'interface Message* permet au régulateur d'obtenir une vue synthétique de tous les messages à destination des véhicules et des stations.

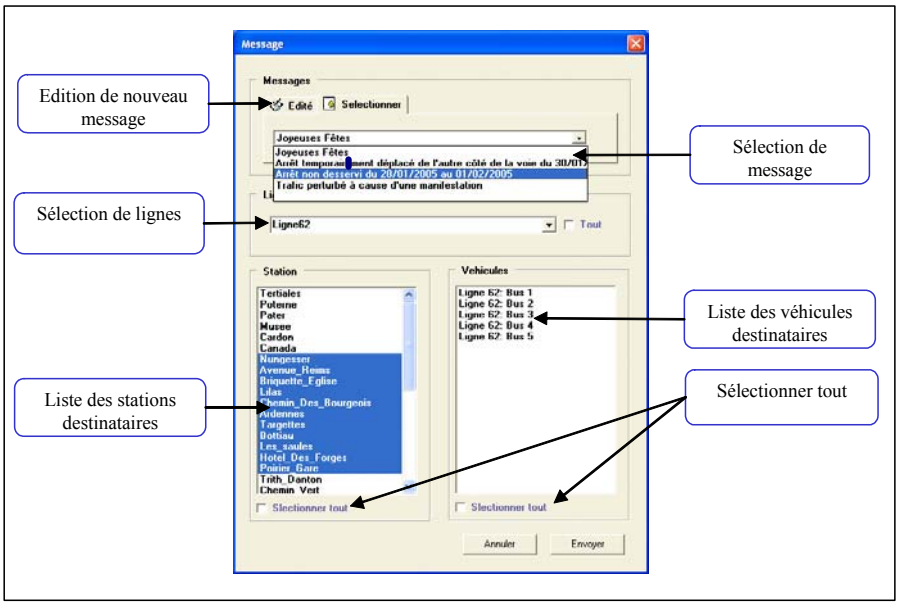



**Figure IV.13.** *Agent d'interface Message*

Elle est composée de quatre types de panneaux (figure IV.13) : un panneau gérant une liste de messages éditables, un panneau permettant le choix de la ligne destination, offrant la possibilité d'une sélection multiple, les deux autres panneaux permettent au régulateur la sélection des stations et des véhicules destinataires pour la diffusion du message sélectionné.

## III.4.6. Agent d'interface Vue_Globale

*L'agent d'interface Vue_Globale* facilite au régulateur la tâche de supervision du trafic. Comme son nom l'indique, la vue de cet agent assure au régulateur une vision globale du trafic dans le réseau. Cette vue englobe toutes les lignes à superviser et facilite l'accès aux lignes, stations et véhicules (Cf. figure IV.14).

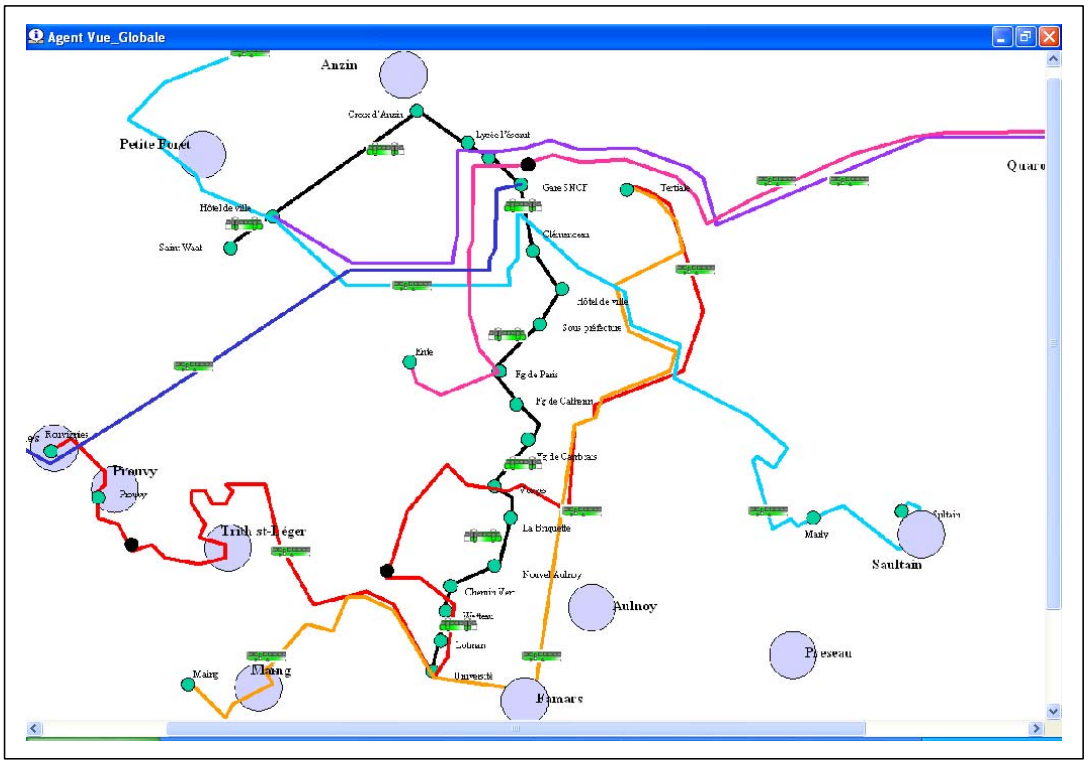



**Figure IV.14.** *Agent d'interface Vue_Globale*

Au stade actuel du développement, la vue de l'agent *Vue_Globale* permet seulement la visualisation de l'ensemble du réseau. Pour accéder à une ligne, un véhicule ou une station, le régulateur doit basculer vers la vue de l'agent *Etat_Ligne*, Véhicule ou station.

## III.5. Conclusion

Cette partie a eu pour objet l'analyse, la modélisation, la spécification et le maquettage du Système d'Aide à l'Information (SAI). Nous avons dans un premier temps effectué une analyse du SAI afin de déterminer les informations pertinentes qui doivent être présentées au régulateur, les éventuels événements liés au SAI et le besoin informationnel du SAI pour la communication avec le SAD et le SAE. Dans un deuxième temps, nous avons modélisé, avec les réseaux de Petri interprétés, les interactions du régulateur avec le SAI ainsi que le fonctionnement de ce dernier. Après l'analyse et la modélisation du SAI nous avons utilisé l'architecture orientée agents que nous avons proposée au chapitre trois pour la spécification du SAI. Cette spécification nous a permis d'identifier six types d'*agents d'interface* avec

lesquels interagit le régulateur. Chaque type d'*agent d'interface* a été défini à l'aide de l'ensemble de ses services tout en utilisant les réseaux de Petri agent introduits au chapitre trois. A l'issue de la phase de spécification nous avons proposé une première maquette du SAI développé sous Borland C++ builder.

# IV. Implémentation du mouchard électronique et son couplage avec le SAI

## IV.1. Implémentation du mouchard électronique

Le mouchard électronique MESIA (Mouchard électronique dédié à l'Evaluation des Systèmes Interactifs orientés Agents) a été conçu dans le but de contribuer à l'évaluation des systèmes interactifs orientés agents et a fait l'objet d'une configuration pour l'évaluation du SAI (Système d'Aide à l'Information voyageurs). De manière plus globale, l'évaluation a pour but d'évaluer l'utilisabilité ainsi que l'utilité du SAI.

Rappelons que le mouchard électronique que nous proposons est un outil d'inspection objectif permettant l'enregistrement des actions du régulateur ainsi que les réactions du système. Notre idée (Cf. chapitre 3 § IV.1) est d'associer un *agent Mouchard* à chaque *agent d'interface* du système à évaluer notamment le SAI [Trabelsi *et al.*, 2006].

La figure IV.15 montre un aperçu de la correspondance entre agents constitutifs du SAI et la création d'*agents Mouchards* [Trabelsi et Ezzedine, 2006]. Par exemple AgV (Agent Véhicule) étant un type d'*agent d'interface* chargé de fournir à l'utilisateur une vue sur un véhicule, un agent de type AgMV (*Agent Mouchard Véhicule*) sera chargé d'enregistrer l'ensemble des interactions en rapport avec un agent concret du type AgV

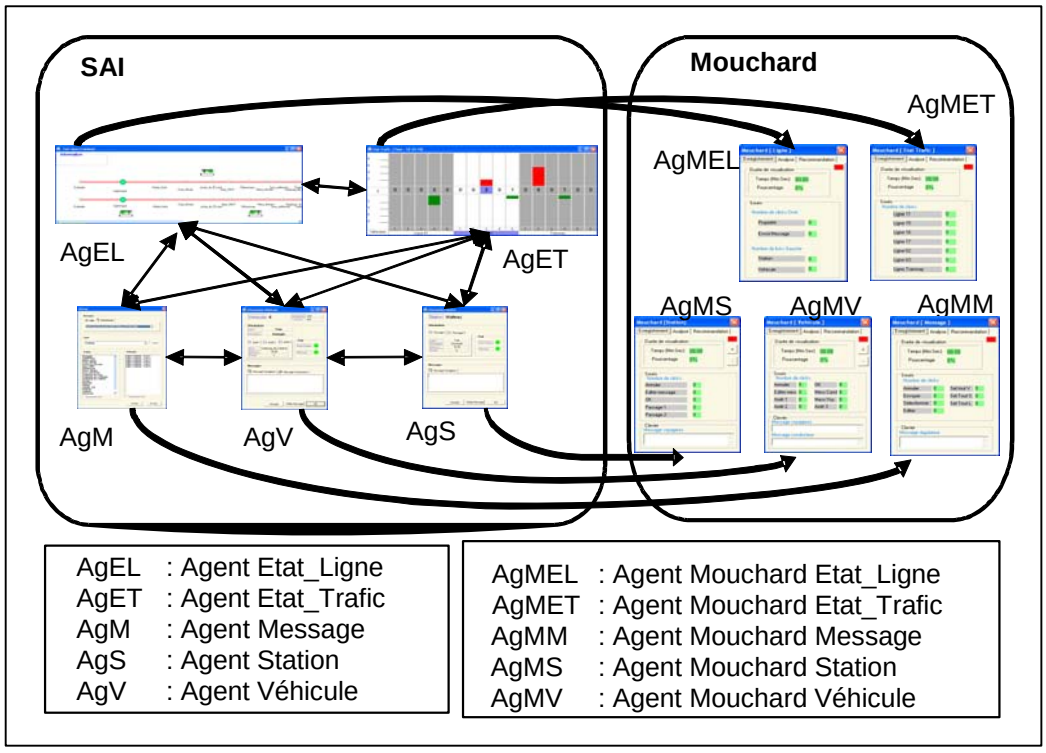



**Figure IV.15.** Association d'un *agent Mouchard* à chaque *agent d'interface*

Deux mouchards particuliers sont ajoutés (non visibles sur la figure IV.15) : un mouchard *Souris* assure l'enregistrement des mouvements et des clics sur la souris (par le régulateur

humain) ; un mouchard *Clavier* assure l'enregistrement des messages saisis par le clavier ou simplement des touches utilisées (Entrée, Echappement). Nous donnons un aperçu dans ce qui suit de l'IHM du mouchard électronique MESIA.

Le *SAI* est constitué de six *agents d'interface*. A chacun de ces agents on associe un *agent Mouchard* qui permettra l'enregistrement de toutes les actions du régulateur (clic souris, saisie de message…) ainsi que les réactions de l'interface (affichage d'un message d'alerte, changement de vue…). L'*agent d'interface Vue_Globale* n'est pas pris en compte par le mouchard. En effet le régulateur ne peut pas interagir avec la vue de l'agent *Vue_Globale* dans la mesure où elle propose seulement la visualisation du réseau.

Ainsi le Mouchard électronique est constitué de :

- Cinq agents mouchard connectés aux *agents d'interface* du SAI (Cf. figure IV.15).
  - *Agent Mouchard Etat_Trafic*
  - *Agent Mouchard Etat_Ligne*
  - *Agent Mouchard Message*
  - *Agent Mouchard Station*
  - *Agent Mouchard Véhicule*
- Un mouchard superviseur assurant la communication avec le *SAI* (établissement de la connexion, traitement des erreurs de transmission, fin de connexion…).
- Deux mouchards particuliers :
  - Mouchard Souris : assure l'enregistrement des mouvements et des clics sur la souris.
  - Mouchard clavier : assure l'enregistrement des messages saisis par le clavier (en particulier l'utilisation des touches : Entrée, Echappement).

Nous présentons dans ce qui suit plusieurs agents Mouchard, en visant la représentativité plutôt que l'exhaustivité.

## IV.2. Description de plusieurs agents mouchard représentatifs

La figure IV.16 présente une vue globale de l'IHM du *Mouchard* (MESIA). Chaque *agent Mouchard* dispose de trois « onglets » [Trabelsi *et al.*, 2006 ; Trabelsi et Ezzedine, 2006]:

**Premier onglet : (Enregistrement)**

Il permet l'affichage de la durée de visualisation de l'*agent d'interface* correspondant. Cette durée est représentée en minutes et en pourcentage. La somme totale des pourcentages est égale à 100. Il permet aussi la visualisation du nombre d'utilisations des différents composants de l'*agent d'interface* correspondant dans le *SAI*.

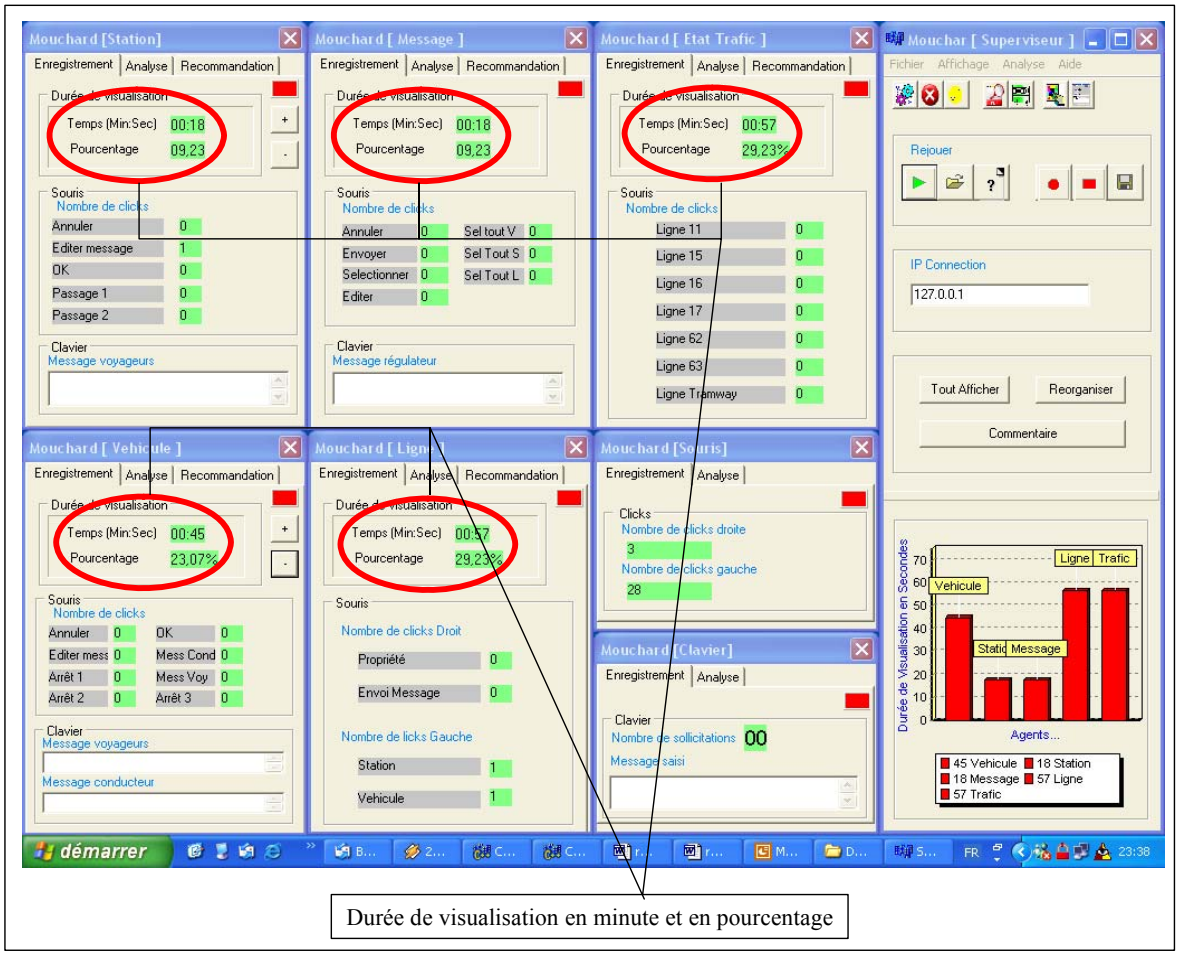



**Figure IV.16.** Vue globale du SMA Mouchard
(outil d'aide aux évaluateurs)

## Deuxième onglet : (Analyse)

Cet onglet permet l'affichage d'un histogramme en 3D représentant le nombre d'utilisations des composants des *agents d'interface* du *SAI*. Cet histogramme permet d'avoir une idée sur la proportion d'utilisation des composants. La figure IV.17 montre un exemple d'histogramme.

Dans cet exemple on distingue les fonctionnalités de l'*agent d'interface Station* à savoir : le clic sur les boutons ok, annuler et éditer ; le clic sur les onglets *Passage 1* et *Passage 2* qui représentent l'horaire du passage des deux prochains véhicules par la station.

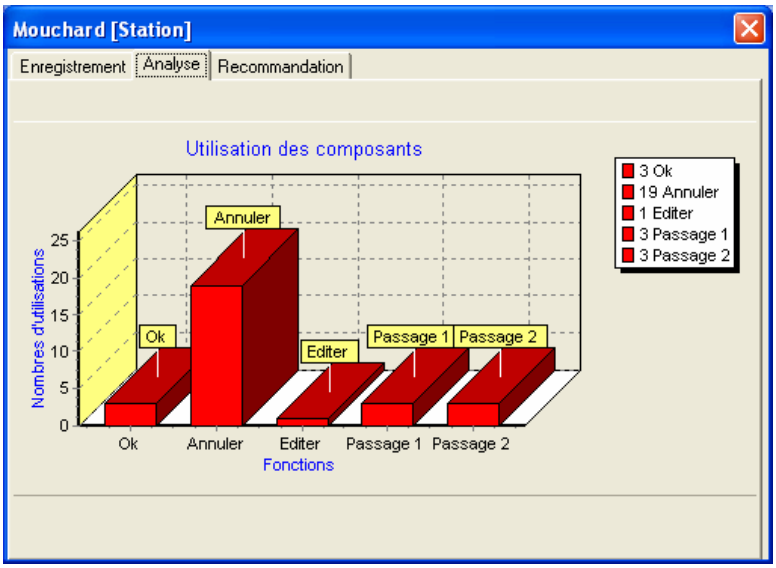



**Figure IV.17.** Utilisation des composants de l'interface

**Troisième onglet : (recommandations)**

Cet onglet, non encore réalisé, mais seulement envisagé pour l'instant, permettra à terme de proposer des recommandations à l'évaluateur afin de reboucler sur la spécification et la réalisation de l'interface. Les recommandations fournies se baseront sur des règles ergonomiques (cf. à ce sujet [Vanderdonckt, 1998]).

Les règles ergonomiques visent à évaluer l'interface Homme-Machine par rapport à des connaissances formalisées provenant de la psychologie cognitive [Farenc, 1997]. L'étude de cet onglet *recommandations* fait partie de nos perspectives de recherche (Cf. Chapitre 5). En effet, un ensemble de règles ergonomiques seront intégrées au système d'aide à l'évaluation proposé (Cf. chapitre 3, § IV.2). Les résultats fournis par le système d'aide à l'évaluation seront sous forme de recommandations correspondant à la comparaison entre les paramètres observés sur l'interface et les valeurs de référence données par les règles ergonomiques.

## IV.2.1. Agent Mouchard Véhicule

L'agent *Mouchard Véhicule* assure l'enregistrement des différentes informations liées à l'*agent d'interface Véhicule* (Cf. figure IV.18). Ces informations sont les suivantes :

- Numéro du véhicule,
- Ligne à laquelle appartient le Véhicule,
- Actions effectuées : ces actions correspondent à l'interaction du régulateur avec l'agent d'interface *Agent Véhicule*. Les differentes actions possibles sont (Cf. en colonne « Action » sur la partie droite de la figure IV.18) :
  - Afficher : ce qui signifie qu'il y a eu un affichage de la fenêtre véhicule demandé par le régulateur (Cf. ①, figure IV.18).
  - Ok : clic sur le bouton *ok* (Cf. ②, figure IV.18).
  - Annuler : clic sur le bouton *annuler* (Cf. ③, figure IV.18).

- Editer : clic sur le bouton *éditer* (Cf.④, figure IV.18).
- Mess_cond : ce qui signifie que le régulateur a cliqué sur l'onglet *message conducteur* afin de saisir un message à destination du conducteur (Cf⑤, figure IV.18).
- Mess_Voy : ce qui signifie que le régulateur a cliqué sur l'onglet *message voyageur* afin de saisir un message à destination des voyageurs (Cf.⑥, figure IV.18).
- Arret1 : clic sur l'onglet *arret1* correspondant à l'horaire d'arrêt pour la prochaine station (Cf⑦, figure IV.18).
- Arret2 : clic sur l'onglet *arret2* correspondant à l'horaire d'arrêt pour la deuxième prochaine station (Cf⑧, figure IV.18).
- Arret3 : clic sur l'onglet *arret3* correspondant à l'horaire d'arrêt pour la troisième prochaine station (Cf⑨, figure IV.18).
- Message_Voyageur et Message_Conducteur : ce qui signifie que le régulateur a tappé un message à destination des voyageurs à bord du véhicule (Cf. ⑩, figure IV.18) ou à destination du conducteur (Cf⑪, figure IV.18).

- Heure d'action : l'instant auquel se produit une action parmi les actions citées ci-dessus (Cf. en colonne « Heure » sur la partie droite de la figure IV.18).
- Retard-Avance : le retard ou l'avance (au moment de l'action effectuée par le régulateur) est enregistré à la colonne « Retard-Avance ».

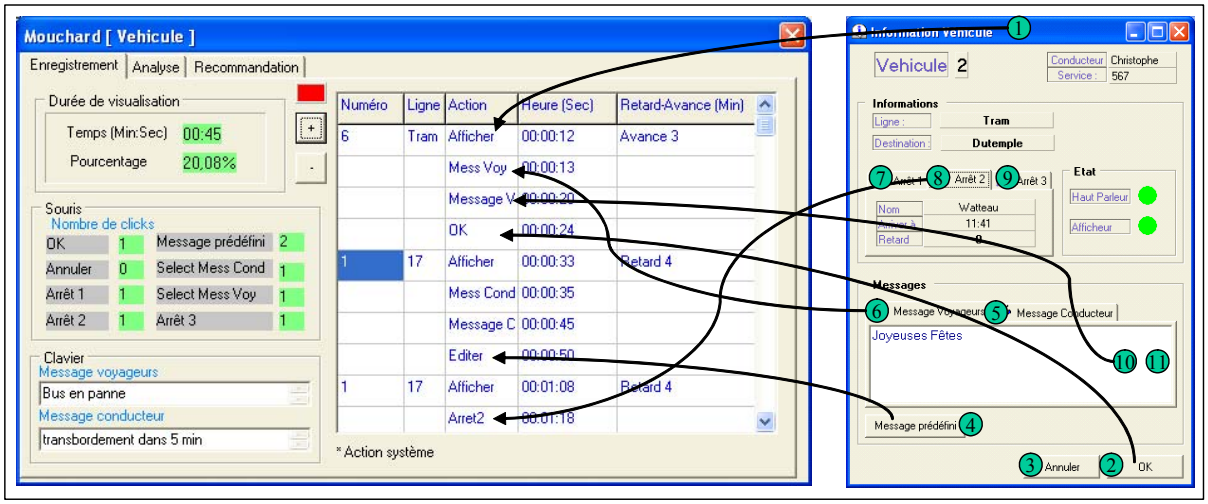



**Figure IV.18.** *Agent Mouchard Véhicule*

Les numéros de véhicules précédés d'un astérisque « * » symbolisent les actions système (affichage d'un message d'alerte).

Les données recueillies durant l'évaluation sont enregistrées dans des fichiers Excel.

## IV.2.2. Agent Mouchard Station

L'agent *Mouchard Station* assure l'enregistrement des différentes informations liées à l'interaction entre le régulateur et l'*agent d'interface Station*. La figure IV.19 montre la

correspondance entre l'*agent d'interface Station* et l'*agent mouchard Station*). Les informations enregistrées par l'*agent mouchard Station* sont les suivantes :

- Le nom de la station.
- La ligne à laquelle appartient la station.
- Actions effectuées : ces actions correspondent à l'interaction du régulateur avec l'*agent d'interface Station*. Les differentes actions possibles sont (Cf. en colonne « Action » sur la partie droite de la figure IV.19) :
  - Afficher : ce qui signifie qu'il y a eu un affichage de la fenêtre station demandé par le régulateur (Cf.①, figure IV.19).
  - Ok : clic sur le bouton *ok* (Cf.②, figure IV.19).
  - Annuler : clic sur le bouton *annuler* (Cf.③, figure IV.19).
  - Editer : clic sur le bouton *éditer* (Cf. ④, figure IV.19).
  - Message_Voyageur : message à l'intention des voyageurs aux stations. Ce message est édité par le régulateur via l'*Agent Station* (Cf⑤, figure IV.19).
  - Passage1 : clic sur l'onglet *passage 1*, correspondant à l'horaire du passage du prochain véhicule par la station (Cf.⑥, figure IV.19).
  - Passage2 : clic sur l'onglet *passage 2*, correspondant à l'horaire du passage du deuxième prochain véhicule par la station (Cf.⑦, figure IV.19).
- Heure d'action : l'instant auquel se produit l'action du régulateur. L'heure 00:00:00 correspond au début de la simulation.
- « C » : Station de Correspondance : suite au clic du régulateur sur une station, l'*agent Mouchard Station* enregistre dans la case « C » la valeur 1 (Station de correspondance) ou 0 (station simple).

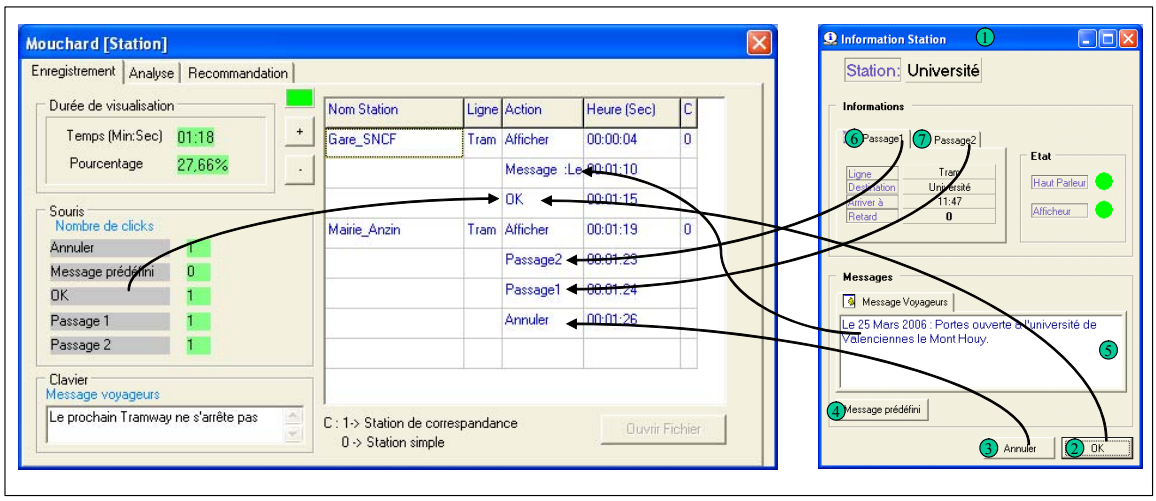



**Figure IV.19.** Correspondance entre *Agent d'interface Station* et *Agent Mouchard Station*

La représentation d'informations enregistrées est effectuée sous forme de tableau. Le nom de la station sollicitée est representé une seule fois pour toutes les actions effectuées. Le

nombre de clics sur chaque composant ainsi que les messages saisis par le régulateur sont également enregistrés.

### IV.2.3. Agent Mouchard Superviseur

L'agent *Mouchard superviseur* (Cf. figure IV.120) assure l'établissement de la connexion entre le *SAI* et le *SMA Mouchard*.

La connexion entre le *SAI* et le *SMA Mouchard* s'effectue via un réseau local. En cas de problème de transmission de données c'est l'agent *mouchard superviseur* qui assure, soit la bonne reprise de connexion ou bien la déconnexion (en toute sécurité) des deux systèmes.

L'agent *mouchard superviseur* assure la visualisation en temps réel d'un histogramme permettant l'affichage des durées de visualisation de chaque *agent d'interface* du *SAI*.

Cet agent permet de rejouer toute la manipulation. A la fin de la phase d'enregistrement, un fichier journal contenant les positions absolues de la souris peut être exécuté. L'évaluateur peut voir ainsi la souris se déplacer tout en reproduisant les mêmes actions effectuées par le régulateur.

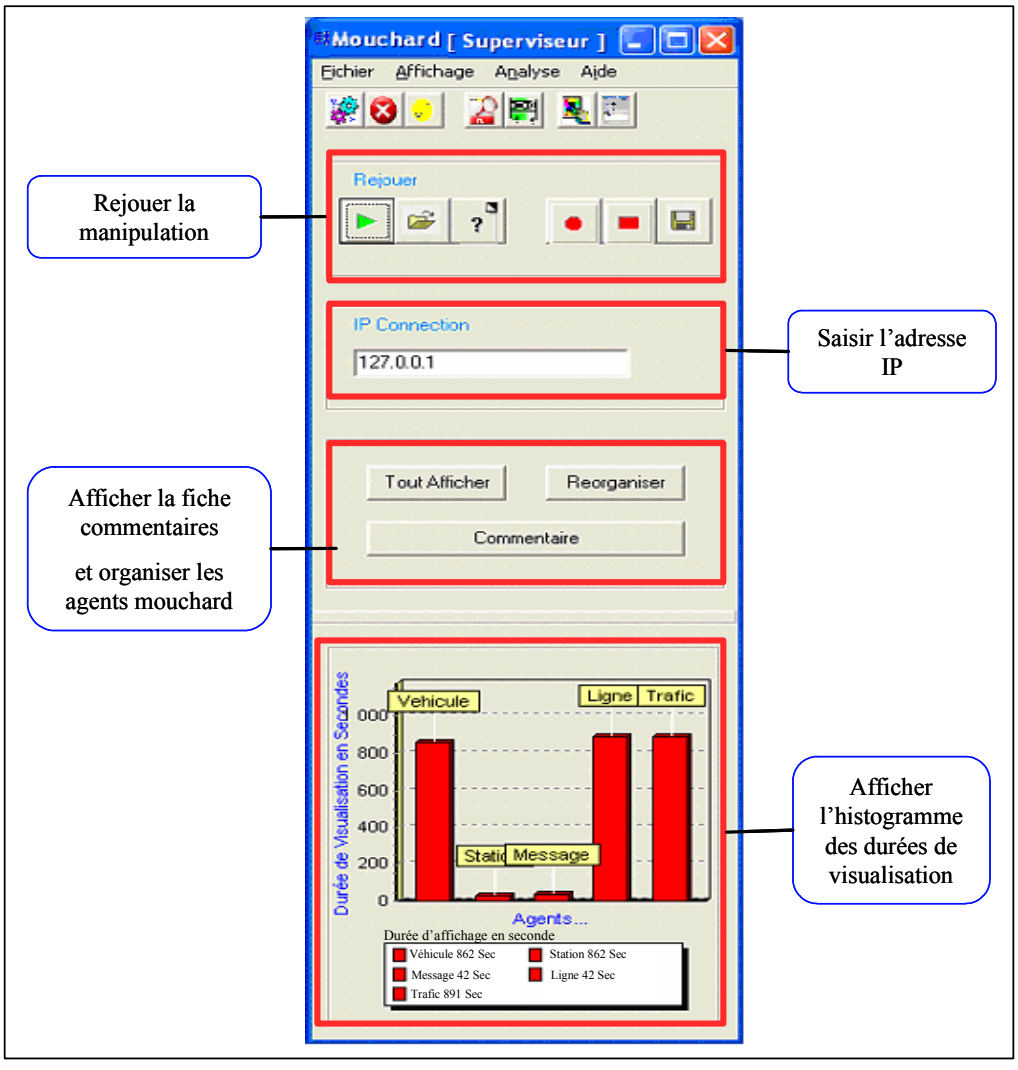



**Figure IV.20.** *Agent Mouchard Superviseur*

Le *SAI* peut être exécuté sur une machine autre que celle sur laquelle s'exécute le mouchard électronique. L'adresse IP de la machine où s'exécute le *SAI* peut être spécifiée grâce à l'onglet « IP Connexion ».

## IV.2.4. Agent Mouchard Souris et Clavier

Les agents Mouchard *Clavier* et *Souris* (Visibles sur la figure IV.16) permettent l'enregistrement des actions du régulateur. Par exemple l'*agent mouchard Souris* enregistre la position de tous les clics du régulateur par rapport au coin supérieur gauche de l'écran. Ces enregistrements sont exploités par l'*agent mouchard Superviseur* pour rejouer toutes les actions effectuées par le régulateur. Par ailleurs, l'*agent Mouchard clavier* permet d'enregistrer tous les messages saisis par le clavier ou simplement les touches utilisées (Entrée, Echappement).

## IV.3. Couplage SAI et SMA Mouchard

Le *SMA Mouchard* électronique est un logiciel qui assure la capture des actions du régulateur ainsi que les réactions du système. Le système à évaluer et le *SMA mouchard* peuvent être exécutés sur deux machines différentes. Nous illustrons dans ce qui suit l'architecture utilisée pour le couplage entre le SAI et le mouchard électronique MESIA.

L'architecture utilisée pour le couplage entre le SAI et le Mouchard électronique MESIA est visible figure IV.21. Elle se base sur le principe d'association d'un port de communication entre chaque agent du SAI et son *agent Mouchard* correspondant.

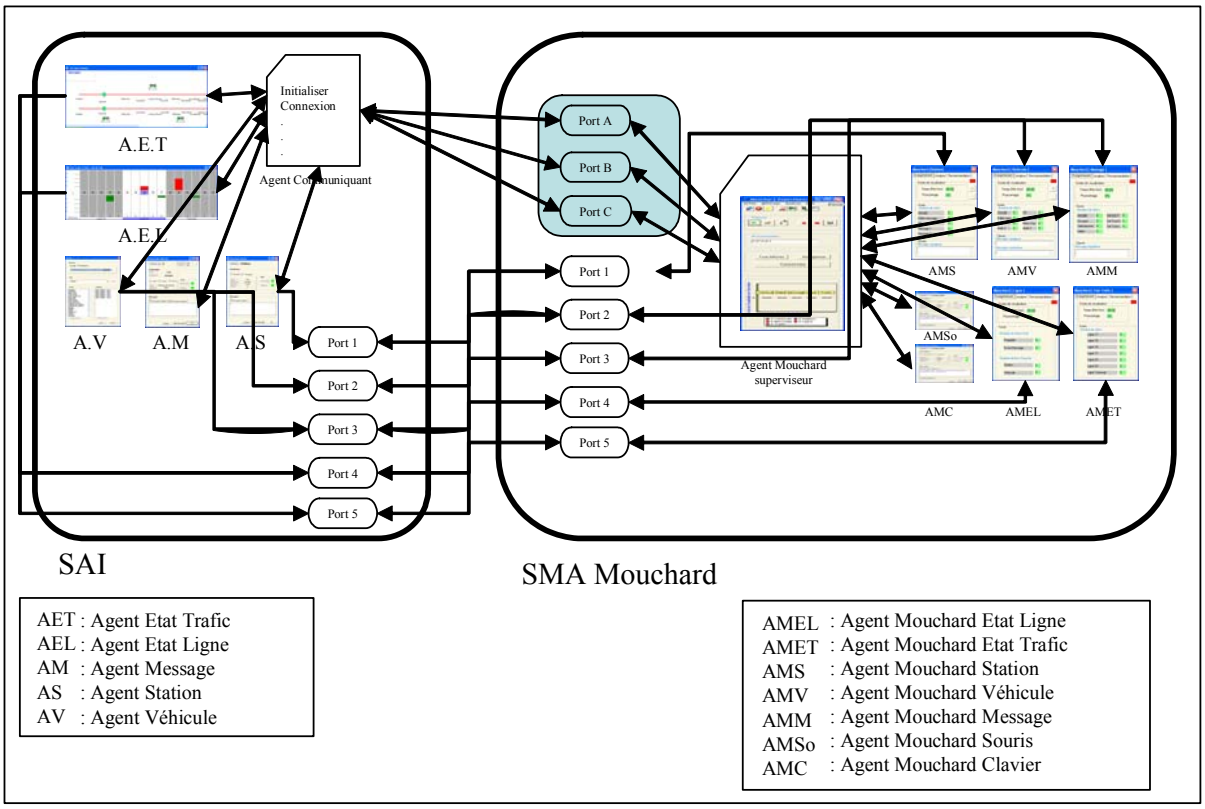



**Figure IV.21.** Couplage SAI et SMA Mouchard

L'agent communiquant (qui fait partie du SAI) assure via les ports A, B et C :

- l'établissement de la connexion entre le SAI et le Mouchard électronique,
- le transfert de données relatives aux mouvements et clics de la souris,
- le transfert des messages saisis à l'aide du clavier.

Les ports de 1 à 5 de la figure IV.21 établissent la connexion entre les *agents d'interface* du SAI et les *agents Mouchards* correspondants. A chaque *agent d'interface* du SAI est associé un port donné. L'*agent mouchard superviseur* assure l'établissement de la connexion.

## IV.4. Conclusion sur l'implémentation du mouchard électronique et son couplage avec le SAI

Cette partie a eu pour objet l'implémentation du mouchard électronique et son couplage avec le SAI. Nous avons illustré quelques exemples d'*agent Mouchard* ainsi que leur fonctionnement. Le coulage du mouchard avec le SAI est basé sur le principe d'association d'un *agent Mouchard* à chaque *agent d'interface* du SAI.

# V. Conclusion

Ce chapitre a fait l'objet d'une mise en application de notre proposition du chapitre précédent. Le projet SART nous a servi de cadre d'application.

Dans la première partie nous avons présenté le projet SART. Plusieurs équipes de recherche ont travaillé ou travaillent actuellement sur ce projet. Notre tâche consistait à effectuer l'analyse, la spécification, et le maquettage du SAI (Système d'Aide à l'Information des voyageurs) tout en prévoyant son évaluation. Après la présentation du projet SART, nous avons effectué une analyse de l'existant du système d'aide à la régulation de transport urbain valenciennois.

Dans la deuxième partie nous avons effectué une analyse du SAI qui consistait à déterminer les caractéristiques de l'information à délivrer aux voyageurs et à déterminer les événements qui agissent sur le SAI. L'architecture proposée au chapitre trois a été utilisée pour la spécification des *agents d'interface* du SAI. Nous avons détaillé à titre d'exemple l'*agent d'interface Etat_Trafic*. Les événements, conditions, ressources, actions visibles et actions non visibles qui constituent les services de cet agent ont été présentés.

Dans la troisième partie l'implémentation du mouchard électronique qui nous servira d'outil de base pour l'évaluation du SAI (Cf. Chapitre 5) a été présentée. Plusieurs agents mouchard ont été décrits. Afin de pouvoir enregistrer les actions/réactions du régulateur et du SAI, nous avons utilisé le principe (énoncé au chapitre trois) d'association d'un *agent Mouchard* à chaque *agent d'interface* du système à évaluer. Ainsi, nous avons proposé un couplage entre l'architecture à base d'agent du SAI et le mouchard électronique.

Après avoir effectué la spécification et le maquettage du SAI ainsi que l'implémentation du mouchard électronique et son couplage avec le SAI, nous présentons dans le chapitre suivant une première évaluation en laboratoire du SAI. Plusieurs perspectives de recherche seront également fournies dans ce chapitre.

*Chapitre 5*

# *Première évaluation et perspectives de recherche*

1. *Introduction*
2. *Évaluation expérimentale*
3. *Perspectives de recherche et développement*
4. *Conclusion*

# I. Introduction

Dans ce dernier chapitre, nous présentons d'abord une première évaluation pratique du Système d'Aide à l'Information des voyageurs (SAI) que nous avons spécifié puis conçu et réalisé (Cf. chapitre 4). Cette évaluation se basera essentiellement sur le système d'aide à l'évaluation proposé (Cf. Chapitre 3) qui assure le couplage entre le mouchard électronique et le SAI (Cf. Chapitre 4). Parallèlement à l'utilisation du système d'aide à l'évaluation, nous utiliserons la technique du questionnaire ainsi que la verbalisation avec les 5 sujets sélectionnés pour cette première évaluation.

Afin de s'assurer que le SAI répond globalement aux attentes des utilisateurs, il est nécessaire d'effectuer une première évaluation de ce dernier. Cette première évaluation se déroulera en laboratoire. En effet, le but d'une évaluation au laboratoire est d'étudier de manière plus contrôlée que la situation réelle, les causes et les effets que certains aspects de l'interface peuvent provoquer sur l'utilisateur [Baccino et al., 2005]. Notons que les utilisateurs qui participent à cette évaluation en laboratoire sont des doctorants et par conséquence des utilisateurs non expérimentés en régulation. Une deuxième évaluation « sur le terrain » est prévue en perspective à court terme. Les sujets qui y participeront seront des utilisateurs professionnels provenant de l'exploitant du réseau de transport collectif.

L'évaluation en laboratoire permettra à l'évaluateur de détecter un premier ensemble représentatif de problèmes d'utilité et/ou d'utilisabilité inhérents à l'exploitation du SAI. Par ailleurs l'évaluation du SAI nous permettra de tester techniquement le système d'aide à l'évaluation proposé ainsi que le couplage entre le mouchard électronique MESIA et le SAI.

Ce cinquième chapitre se compose de deux parties.

La première partie présente une première évaluation en laboratoire du SAI. Elle fera intervenir cinq sujets non spécialistes en régulation du trafic terrestre et se basera sur trois outils et techniques d'évaluation qui sont : l'outil d'aide à l'évaluation (proposé au troisième chapitre), le questionnaire et la verbalisation. Les deux scénarii sur lesquels se base l'évaluation ainsi que les résultats obtenus seront détaillés.

La deuxième partie est consacrée à la présentation de nos perspectives de recherche et de développement à court, moyen et long terme. En effet, ces perspectives se divisent en deux grandes catégories : la première s'intéresse d'une part à l'amélioration du SAI suite aux résultats obtenus par l'évaluation, et d'autre part, à la description du dispositif et du protocole expérimentaux liés à l'évaluation. La deuxième catégorie quant à elle, concerne l'amélioration du système d'aide à l'évaluation qui portera sur plusieurs modules liés à ce dernier.

# II. Évaluation expérimentale

Cette section illustre l'évaluation du SAI en débutant par la description de la méthode d'évaluation utilisée et le dispositif expérimental mis en place et en finissant par la présentation du protocole expérimental et les résultats obtenus. La méthode d'évaluation exploitée en vue d'une première évaluation (en laboratoire) du SAI est maintenant présentée.

## II.1. Principe et méthodes d'évaluation

Comme nous l'avons vu dans le deuxième chapitre, les méthodes d'évaluation des systèmes interactifs et particulièrement des interfaces homme-machine sont nombreuses ; chaque méthode choisit de privilégier un ou plusieurs attributs de l'utilisabilité et de l'utilité à travers la mesure de différentes variables : la durée d'exécution, le taux d'erreurs, etc., ou à travers l'opinion ou les attitudes des utilisateurs. Nous avons vu aussi que les méthodes d'évaluation pouvaient être regroupées en trois grandes catégories :

- les méthodes basées sur des techniques d'observation de l'utilisateur et de recueil des données de l'interaction,
- les méthodes basées sur l'intervention d'experts en interaction homme-machine, en psychologie cognitive ou en ergonomie,
- les méthodes analytiques basées sur des modèles formels prédictifs intégrant des connaissances sur la tâche et sur des grammaires ou des modèles formels de qualité.

Chacune des méthodes de ces trois catégories possède des avantages et des inconvénients. L'état de l'art que nous avons effectué au deuxième chapitre sur ces méthodes d'évaluation, nous suggère que les méthodes basées sur les techniques d'observation de l'utilisateur semblent être bien adaptées pour l'évaluation du SAI. En effet, cette catégorie de méthodes possède l'avantage de la prise en compte de la réalité de l'utilisation du système dans le cadre de tâches (réelles ou représentatives), tout en impliquant l'utilisateur (réel ou représentatif). Cet aspect assure aux résultats une « fiabilité » non égalable par les méthodes dites théoriques (méthodes basées sur des experts, méthodes analytiques) [Balbo, 1994 ; Farenc, 1997 ; Bastien et Scapin, 2002], et ceci même si cette première évaluation sera menée en laboratoire (et non en salle de régulation du réseau) avec des sujets qui ne sont pas pour l'instant des exploitants professionnels.

Ainsi, pour l'évaluation du SAI, nous nous baserons en premier lieu sur l'outil d'aide à l'évaluation proposé au troisième chapitre (et particulièrement sur le mouchard électronique MESIA), et en deuxième lieu sur la technique du questionnaire et de la verbalisation[38]. L'utilisation de plusieurs outils et techniques est justifiée par la diversité des informations à récupérer lors de l'évaluation. En effet les informations fournies par ces différents outils et techniques seront complémentaires et permettront à l'évaluateur de détecter un premier ensemble représentatif de problèmes d'utilité et/ou d'utilisabilité inhérents à l'exploitation du SAI. Rappelons ici que l'utilisabilité rend compte principalement de la qualité de l'interaction homme-machine en termes de facilité d'apprentissage et d'utilisation. L'utilité détermine si

[38] A ces trois techniques on peut ajouter la technique de l'utilisation de l'oculomètre pour la détermination de la position absolue du regard. L'utilisation de cette technique sera présentée plus loin dans les perspectives de recherche à court terme relatives aux évaluations avec des utilisateurs professionnels provenant de l'exploitant du réseau de transport collectif.

l'interface répond aux besoins de l'utilisateur ; en d'autres termes, si l'application permet à l'utilisateur d'atteindre ses objectifs de travail [Nielsen, 1993].

Le choix de ces trois outils et techniques pour l'évaluation du SAI est justifié par les trois raisons suivantes :

- Les données à recueillir sont nombreuses et variées (Cf. les sorties de la boîte A3, figure V.1). Ainsi, des données objectives[39] seront recueillies avec le mouchard électronique MESIA et des données subjectives[40] seront recueillies via l'utilisation du questionnaire et de la verbalisation.

- La mise en place du dispositif expérimental qui en résulte ne pose pas de difficulté particulière. En effet, la manipulation de l'outil d'aide à l'évaluation ainsi que du mouchard électronique est relativement intuitive[41]. Par ailleurs, l'élaboration du questionnaire reste tout de même une tâche partiellement délicate, dans le sens où ce dernier doit couvrir, dans la mesure du possible, un maximum de points relatifs à l'utilisabilité et l'utilité de l'interface ; mais une fois mis en place, le questionnaire est facile à utiliser. En fournissant des résultats pertinents, puisqu'elle représente l'opinion réelle de l'utilisateur, la verbalisation ne nécessite pas de préparation particulière (mis à part le fait de préparer à l'avance les questions qui seront posées, et le déroulement global du dialogue avec les sujets).

- Le coût de la mise en place du protocole expérimental est faible. Dans un but de première évaluation en laboratoire, les trois outils et techniques utilisés ne nécessitent pas d'investissements particuliers.

L'évaluation du SAI a pour objectif de fournir après expérimentation les données indiquées en sortie de la boîte A3 à la figure V.1. Rappelons que cette boîte est reprise de la figure III.1 (Cf. Chapitre 3). Les entrées de la boîte A3 nommée « Evaluer le SAI » sont : l'IHM qui représente l'interface spécifiée et développée selon les principes d'architecture proposées au chapitre trois et le mouchard électronique préalablement configuré (Cf. Chapitre, 4 § IV).

Selon la figure V.1, les contraintes (jargon utilisé par la méthode SADT ; Cf. Annexe 1) relatives à l'évaluation sont : la synchronisation (effectuée entre le SAI et le mouchard électronique lors du couplage), les principes ergonomiques (qui seront exploités à long terme[42] par l'outil d'aide à l'évaluation afin de proposer des améliorations possibles concernant l'aspect ergonomique du SAI) et le temps (qui représente l'instant d'exécution d'une action effectuée par l'utilisateur, la durée de visualisation de chaque vue du SAI, etc.).

[39] Une donnée peut se définir de manière objective ; c'est une donnée qui résulte d'une mesure observée.
[40] Une donnée peut être définie de manière subjective ; c'est une donnée qui provient de l'opinion de l'utilisateur.
[41] Par utilisation intuitive, nous entendons une facilité d'utilisation et de manipulation ; il est cependant intéressant de valider cet outil avec des évaluateurs professionnels afin de déterminer son utilité et ses limites. Ceci fait partie de nos perspectives de recherche à long terme.
[42] L'intégration des règles ergonomiques à l'outil d'aide à l'évaluation proposé fait partie de nos perspectives de recherche (Cf. § III.2.3).

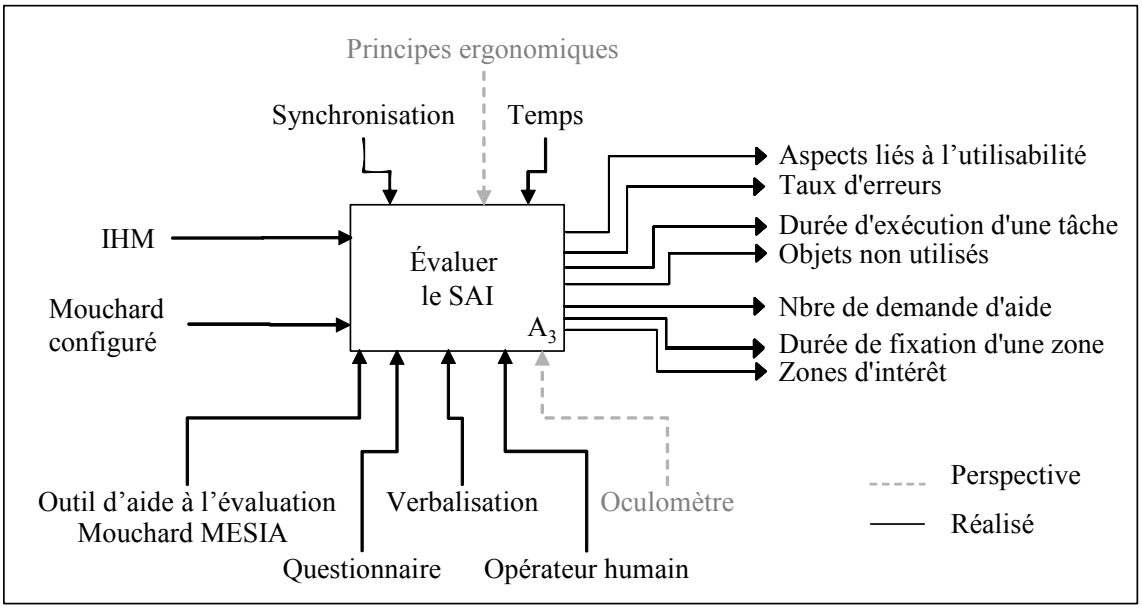



**Figure V.1.** Evaluation du SAI

## II.2. Dispositif expérimental

Le dispositif expérimental de l'évaluation du SAI est constitué de plusieurs techniques et outils (Cf. figure V.2) : l'outil d'aide à l'évaluation intégrant le mouchard électronique MESIA, le questionnaire et la verbalisation (on y a aussi positionné l'oculomètre, même si celui-ci n'est pas exploité pour cette première évaluation en laboratoire). L'utilisation de chaque outil et technique est maintenant présentée.

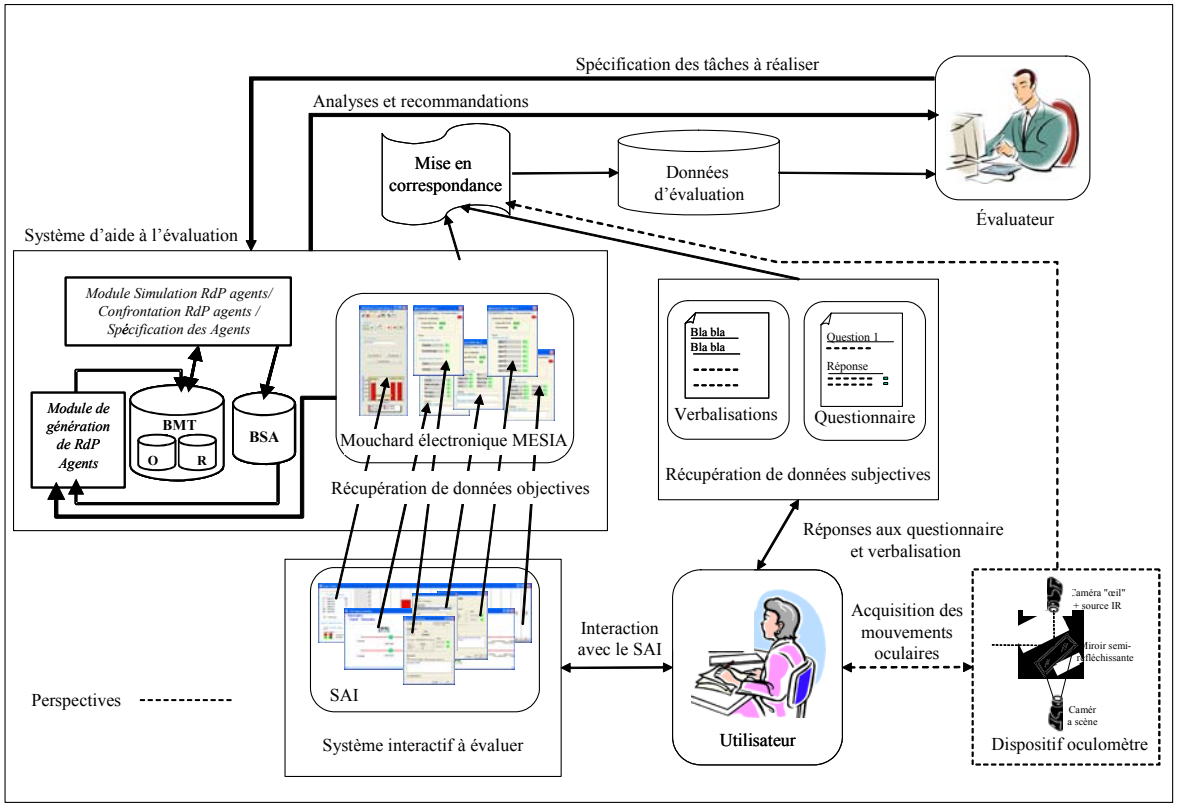



**Figure V.2.** Dispositif expérimental utilisé pour l'évaluation du SAI

## II.2.1. Le Système d'aide à l'évaluation

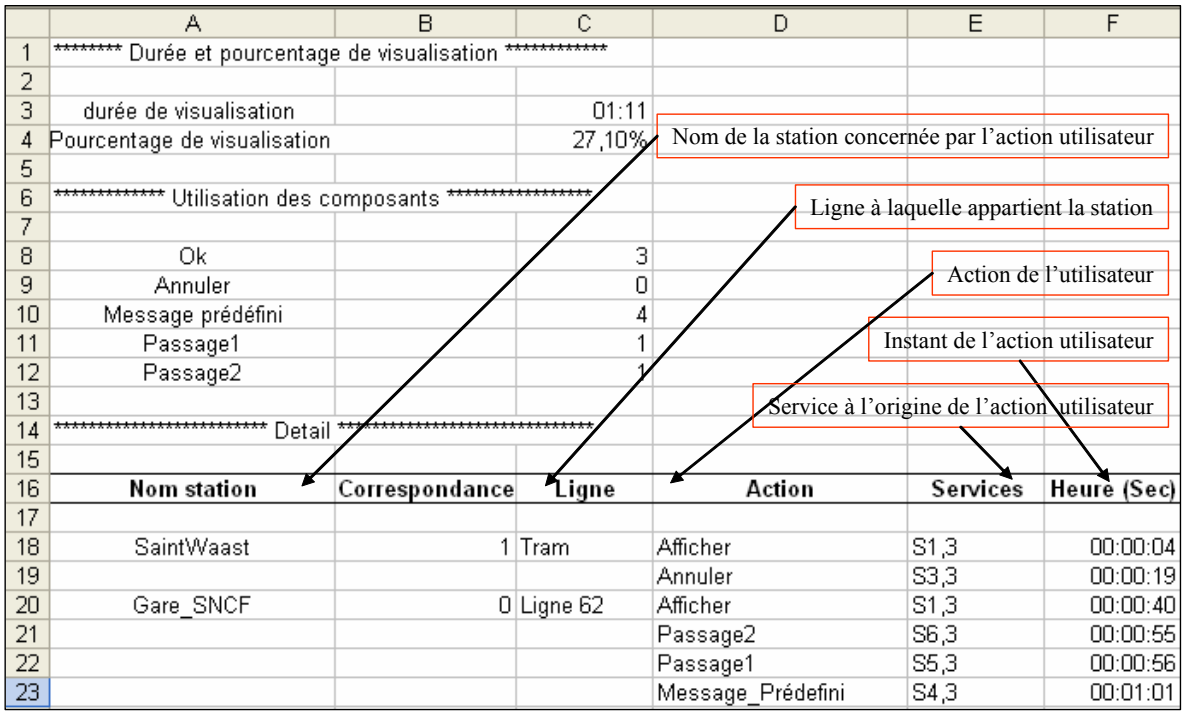
Comme nous l'avons vu au troisième chapitre, le système d'aide à l'évaluation a été conçu afin d'aider et d'assister les évaluateurs des systèmes interactifs orientés agents. Rappelons qu'il est composé de trois modules (Cf. chapitre III, figure III.11) :

- **Le module mouchard électronique** : couplé avec le SAI, le mouchard électronique assure le recueil des actions/réactions de l'utilisateur et du système. Ces données sont enregistrées dans des fichiers sous format « Excel ». Dans l'exemple de fichier Excel visible en figure V.3, on distingue un extrait du fichier enregistrant les données relatives à l'interaction de l'utilisateur avec l'*agent d'interface Station.* Ces données recueillies sont très importantes dans la mesure où on peut reconstituer les modèles des tâches réellement effectuées par l'opérateur et les comparer aux modèles des tâches à effectuer. La reconstruction des modèles de tâches réellement effectuées est assurée par le module de génération de RdP.

- **Le module de génération de RdP** : ce module exploite les données fournies par le mouchard électronique sous forme de fichier Excel (Cf. ci-dessus), afin de générer un RdP agents représentant le modèle de la tâche effectuée dans la BMT (O) (Base des Modèles de Tâches Observées) et le modèle de la tâche à réaliser dans la BMT (R) (Base des Modèle de Tâches à Réaliser) (Ces deux bases avaient été décrites dans le troisième chapitre ; Cf. figure III.11). Au stade actuel du développement de l'outil d'aide à l'évaluation, ce module fournit des résultats sous forme textuelle. Nous proposons dans la section relative aux perspectives de recherche une amélioration de ce dernier.

- **Le module de Simulation/Confrontation/Spécification de RdP agent :** ce module offre aux évaluateurs/concepteurs les trois fonctionnalités suivantes : la simulation de RdP agents, la confrontation de RdP agents et la spécification de RdP agents (effectuée avec l'outil interactif visible en figure IV.6 du chapitre 4). Nous proposons dans la section relative aux perspectives de recherche quelques pistes d'améliorations relatives à ces trois fonctionnalités.

Rappelons ici que l'avantage de l'utilisation du mouchard électronique réside, d'une part dans sa capacité de couplage avec le SAI en vue de recueillir les actions/réactions entre l'utilisateur et le SAI, et d'autre part dans sa discrétion par rapport à l'utilisateur (il n'est pas intrusif). Outre les verbalisations, l'utilisation du mouchard électronique comme outil pour l'évaluation du SAI est accompagnée de réponses subjectives de l'utilisateur au questionnaire. Ce dernier est maintenant présenté.



| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | ******** Durée et pourcentage de visualisation ********** | | | | | |
| 2 | | | | | | |
| 3 | durée de visualisation | | 01:11 | | | |
| 4 | Pourcentage de visualisation | | 27,10% | | | |
| 5 | | | | | | |
| 6 | ********** Utilisation des composants ********** | | | | | |
| 7 | | | | | | |
| 8 | Ok | | 3 | | | |
| 9 | Annuler | | 0 | | | |
| 10 | Message prédéfini | | 4 | | | |
| 11 | Passage1 | | 1 | | | |
| 12 | Passage2 | | 1 | | | |
| 13 | | | | | | |
| 14 | ********** Detail ********** | | | | | |
| 15 | | | | | | |
| 16 | **Nom station** | **Correspondance** | **Ligne** | **Action** | **Services** | **Heure (Sec)** |
| 17 | | | | | | |
| 18 | SaintWaast | 1 | Tram | Afficher | S1,3 | 00:00:04 |
| 19 | | | | Annuler | S3,3 | 00:00:19 |
| 20 | Gare_SNCF | 0 | Ligne 62 | Afficher | S1,3 | 00:00:40 |
| 21 | | | | Passage2 | S6,3 | 00:00:55 |
| 22 | | | | Passage1 | S5,3 | 00:00:56 |
| 23 | | | | Message_Prédefini | S4,3 | 00:01:01 |



**Figure V.3.** Extrait d'enregistrements effectués par le mouchard électronique MESIA

## II.2.2. Questionnaire

L'intérêt du questionnaire est de complémenter sous une forme structurée les verbalisations ; il s'agit de compléter les résultats expérimentaux, obtenus par le mouchard électronique (données objectives) par le recueil d'appréciations subjectives exprimant globalement l'opinion de l'utilisateur.

Le questionnaire préparé spécifiquement pour cette évaluation s'inspire de [Ravden et Johnson, 1989], ainsi que d'autres questionnaires exploités dans l'équipe dans le cadre de projets précédents depuis les années 90 [Benaissa, 1993 ; Kolski et al., 2000]. Il se compose de trois parties : la première présente des questions concernant des aspects généraux de l'interface tel que le temps de réponse, la deuxième regroupe des questions spécifiques à chaque vue du SAI et la troisième partie présente une évaluation ergonomique globale de l'interface. Les critères utilisés dans cette dernière partie du questionnaire sont inspirés de [Ravden et Johnson, 1989]. On distingue huit critères : la clarté visuelle, la cohérence, la compatibilité, le retour informationnel, la flexibilité, le contrôle, la prévention et la correction d'erreurs.

Le questionnaire complet est disponible en annexe 3. La figure V.4 en illustre un extrait utilisé pour l'évaluation de la vue de l'agent d'interface Station (Cf. chapitre 4, § III.4.3). Comme visible à la figure V.4, nous avons utilisé trois types de questions : questions scalaires à choix binaire (a), questions scalaires à choix multiples (b) et questions à échelle de jugement (c). Ce questionnaire est orienté vers les problèmes ergonomiques liés à l'utilisation de l'IHM du SAI. Il vise particulièrement à savoir si l'IHM, vis-à-vis des utilisateurs, est claire, fonctionnelle, etc.

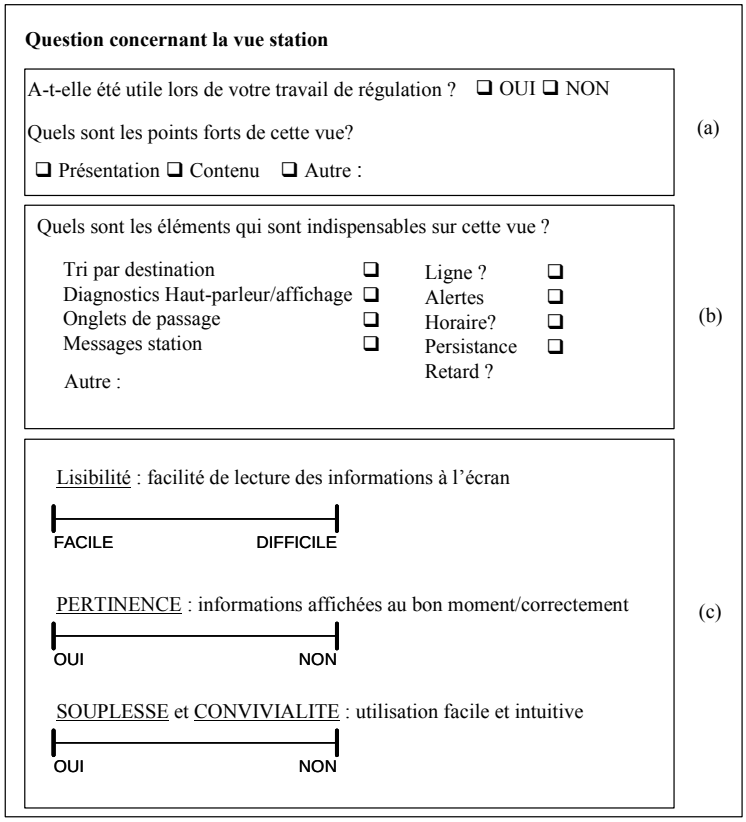



**Figure V.4.** Un extrait du questionnaire utilisé lors de l'évaluation de la vue de l'*agent d'interface Station* du SAI

La technique complémentaire par nature à celle des questionnaires est celle de la verbalisation. L'utilisation de celle-ci pour l'évaluation du SAI est maintenant présentée.

## II.2.3. Verbalisation

La verbalisation ou l'entretien (Cf. Chapitre 2, § III.1.1.2) est un moyen facile et direct pour recueillir de l'information sur la qualité du système et particulièrement l'interface. Contrairement au questionnaire (décrit ci-dessus), la verbalisation possède l'avantage d'être plus flexible dans la mesure où elle permet l'orientation des questions vers l'information recherchée par l'évaluateur.

## II.2.4. Données à recueillir

Lors de l'évaluation du SAI, les données à recueillir peuvent provenir de trois sources différentes :

- le mouchard électronique MESIA : il fournit des données objectives à partir de la capture des actions/réactions de l'utilisateur et du SAI. Comme nous l'avons vu au quatrième chapitre, le couplage entre le mouchard électronique et le SAI permet à chaque agent mouchard de capturer les différentes interactions entre l'utilisateur et l'agent d'interface correspondant. Les données recueillies sont classifiées en cinq catégories selon le principe énoncé au troisième chapitre (Cf. Chapitre trois, § IV.1.2) ;
- le questionnaire : il fournit des données subjectives portant sur les trois parties qui le constituent (Cf. § II.2.2). Des pourcentages relatifs aux réponses des utilisateurs

peuvent nous permettre de déterminer des problèmes d'utilité et/ou d'utilisabilité liés au SAI ;

- la verbalisation : elle permet de décrire d'une façon informelle les problèmes rencontrés par les utilisateurs ainsi que leurs suggestions. Les données recueillies par cette technique seront prises en compte pour des éventuelles améliorations du SAI.

## II.3. Population impliquée dans l'évaluation

La question relative au nombre de sujets (utilisateurs) qu'il faut impliquer pour identifier un pourcentage suffisant de problèmes d'utilisation était et demeure toujours pertinente. Ainsi pour l'évaluation du SAI nous nous baserons sur les études réalisées par [Nielsen et Landauer, 1993 ; Virzi, 1990] qui indiquent qu'avec 4 ou 5 participants il était possible de détecter près de 80 à 85 % des problèmes d'utilisabilité que pouvait présenter une interface.

Pour cette première évaluation en laboratoire du SAI, la population est composée de cinq sujets. Ils ont une moyenne d'âge de 29 ans et sont tous de sexe masculin.

Les sujets sont tous doctorants en informatique. Le fait d'avoir des sujets qui sont experts en informatique peut en quelque sorte compenser leur manque d'expérience au domaine de la régulation[43].

## II.4. Protocole expérimental

Chaque expérience dure environ une heure et demie et comporte quatre phases reparties de la manière indiquée en tableau V.1.

| Phase | Durée | Tâche à effectuer |
|---|---|---|
| 1 | 15 min | Accueil, description des objectifs de l'évaluation, et présentation globale du Système d'Aide à l'Information (SAI) |
| 2 | 10 min | Apprentissage et essai libre du SAI |
| 3 | 20 min | Réalisation (par l'utilisateur) de deux scénarios préalablement préparés |
| 4 | 45 min | Réponses au questionnaire et verbalisation de l'utilisateur |

**Tableau V.1.** Durée de déroulement de l'expérience

Durant la première phase de l'expérience, le fonctionnement du SAI est expliqué au sujet. Des réponses à des éventuelles questions complémentaires (posées par le sujet) sont fournies.

Le participant à l'expérimentation est ensuite familiarisé avec les différentes vues du SAI pendant environ dix minutes avant de subir véritablement les tests expérimentaux.

[43] Mais il est clair que les résultats obtenus dans le cadre de cette première évaluation ne seront jamais aussi riches que ceux qui auraient pu être obtenus avec des utilisateurs professionnels de la régulation.

La manipulation proprement dite se déroule en vingt minutes et porte sur deux scénarios : le premier concerne l'évaluation des aspects liés au dialogue utilisateur-SAI et le deuxième s'intéresse à l'évaluation des aspects liés à la supervision et à la régulation. Dix minutes sont consacrées à chaque scénario.

La dernière phase de l'évaluation porte sur le remplissage du questionnaire et la verbalisation ; elle dure environ quarante-cinq minutes.

A ce stade du chapitre, nous avons effectué une brève description du principe de la méthode d'évaluation adoptée. Nous avons aussi défini le dispositif expérimental à utiliser pour l'évaluation du SAI ainsi que la population impliquée dans l'évaluation. Nous présentons dans ce qui suit les deux scénarios proposés pour l'évaluation du SAI : le premier concerne l'évaluation en mode de fonctionnement normal, alors que dans le second il s'agit d'un mode de fonctionnement perturbé[44]. Les tâches à réaliser relatives à chaque scénario ainsi que les résultats obtenus seront décrits.

## II.4.1. Premier scénario : évaluation du SAI en mode de fonctionnement normal

Dans la réalité, en mode de fonctionnement normal, les analyses ont montré que la tâche du régulateur se résume à la supervision du trafic, mais qu'il peut aussi envoyer des messages aux véhicules et aux stations de sa propre initiative. Dans le cadre de l'évaluation, nous nous rapprochons donc de cet état de fait.

Pour s'assurer que l'utilisateur (le sujet) réussit facilement à interagir avec le SAI en envoyant des messages au (x) station(s) et au(x) véhicule (s), nous proposons un scénario d'évaluation composé de quatre tâches décrites dans la section suivante.

### II.4.1.1. Tâches à réaliser

Il s'agit d'exécuter séquentiellement les quatre tâches prédéfinies décrites au tableau V.2. Ce tableau est fourni à l'utilisateur en version papier en lui expliquant qu'il doit réaliser les quatre tâches qui y sont décrites et en respectant si possible la durée théorique réservée pour chaque tâche.

Les quatre tâches à réaliser sont modélisées avec le module *spécification des RdP* (Cf. chapitre trois IV.2.4) et sont enregistrées dans la base BMT(R) (Base des Modèles de Tâches à Réaliser). Ces spécifications sont exploitées par le module de *génération de RdP* afin de générer le réseau de Petri correspondant à la tâche à réaliser. Dans la mesure où l'implémentation du module génération des RdP n'est pas finalisée (et fera d'ailleurs l'objet de perspectives de recherche), nous avons modélisé les tâches à réaliser manuellement. Un exemple de la troisième tâche à effectuer est présenté[45] à la partie gauche de la figure V.5.

Le modèle de la tâche à réaliser sera confronté au modèle de la tâche observée qui est déterminé à partir des enregistrements effectués avec le mouchard électronique.

[44] Les deux modes de fonctionnement du SAI (normal et perturbé) ont été décrits au quatrième chapitre (Cf. chapitre 4, § III.2.2, figure IV.5)

[45] La figure du modèle de la tâche à effectuer est illustrée plus loin lors de la présentation du modèle de la tâche observée.

| Tâches à réaliser | Durée théorique du temps nécessaire pour la réalisation de la tâche | Description de la tâche |
|---|---|---|
| T1 | 45 secondes | Envoyer un message à la Station « Gare SNCF » de la ligne Tramway (Arrêt du prochain tramway 2 minutes en station). |
| T2 | 45 secondes | Envoyer un message au conducteur du Tramway 6 (Arrêtez-vous 2 minutes à la prochaine station) |
| T3 | 60 secondes | Envoyer un message à toutes les stations de la ligne 16 (Trafic perturbé : à cause d'une manifestation) |
| T4 | 60 secondes | Envoyer un message à tous les véhicules (Joyeuses fêtes) |

**Tableau V.2.** Scénario 1 : tâches à réaliser

Notons que les durées théoriques nécessaires pour la réalisation des tâches sont des durées approximatives. Elles sont déterminées par un expert en supervision initié au SAI.

## II.4.1.2. Résultats

Les cinq sujets ont réalisé les quatre tâches prévues au premier scénario. Le dispositif expérimental utilisé nous a permis de recueillir des données objectives au moyen du mouchard électronique MESIA et des données subjectives grâce à la verbalisation et aux remplissages du questionnaire. Nous présentons dans ce qui suit quelques résultats pertinents.

Le tableau V.3 obtenu grâce au mouchard électronique, présente un récapitulatif de la moyenne de durée réelle du temps de réalisation de chaque tâche ainsi que le taux de succès de sa réalisation.

Les résultats obtenus dans ce tableau indiquent que tous les sujets ont bien réalisé la première et la deuxième tâche avec un temps moyen de réalisation acceptable. Pour la troisième et quatrième tâche, seulement trois sujets sur cinq ont pu les mener à terme. Nous remarquons aussi que la durée réelle de réalisation de ces deux tâches dépasse la moyenne (moyenne calculée par rapport aux trois sujets qui ont réussi leur tâche). En effet, ceci peut s'expliquer par le fait que les sujets ne sont pas des experts en régulation.

| Tâches réalisées | Durée théorique nécessaire pour la réalisation de la tâche | Durée réelle de la réalisation de la tâche (Moyenne) | Taux de succès de réalisation de la tâche |
|---|---|---|---|
| T1 | 45 secondes | 41 secondes | 100 % |
| T2 | 45 secondes | 39 secondes | 100 % |
| T3 | 60 secondes | 67 secondes | (3 sujets sur 5) |
| T4 | 60 secondes | 75 secondes | (3 sujets sur 5) |

**Tableau V.3.** Tableau récapitulatif de la durée d'exécution des tâches et les taux de succès

Pour mieux comprendre les résultats obtenus au tableau précédant, nous comparons le modèle de la tâche observée et le modèle de la tâche à réaliser. Nous prenons comme exemple la tâche T3 visible en figure V.5.

La figure V.5 concerne la tâche T3. Elle montre le modèle de la tâche à réaliser (partie gauche), le modèle de la tâche réalisée, avec succès, par les sujets 1,3 et 5 (partie du milieu), la tâche réalisée, avec échec, par le sujet 2 (partie en haut à droite) et la tâche réalisée, avec échec, par le sujet 4 (partie en bas à droite). Les services présentés sur la figure V.5, sous la forme ($S_{i,j}$), représentent les services enclenchés par l'agent pour passer d'un état à l'autre.

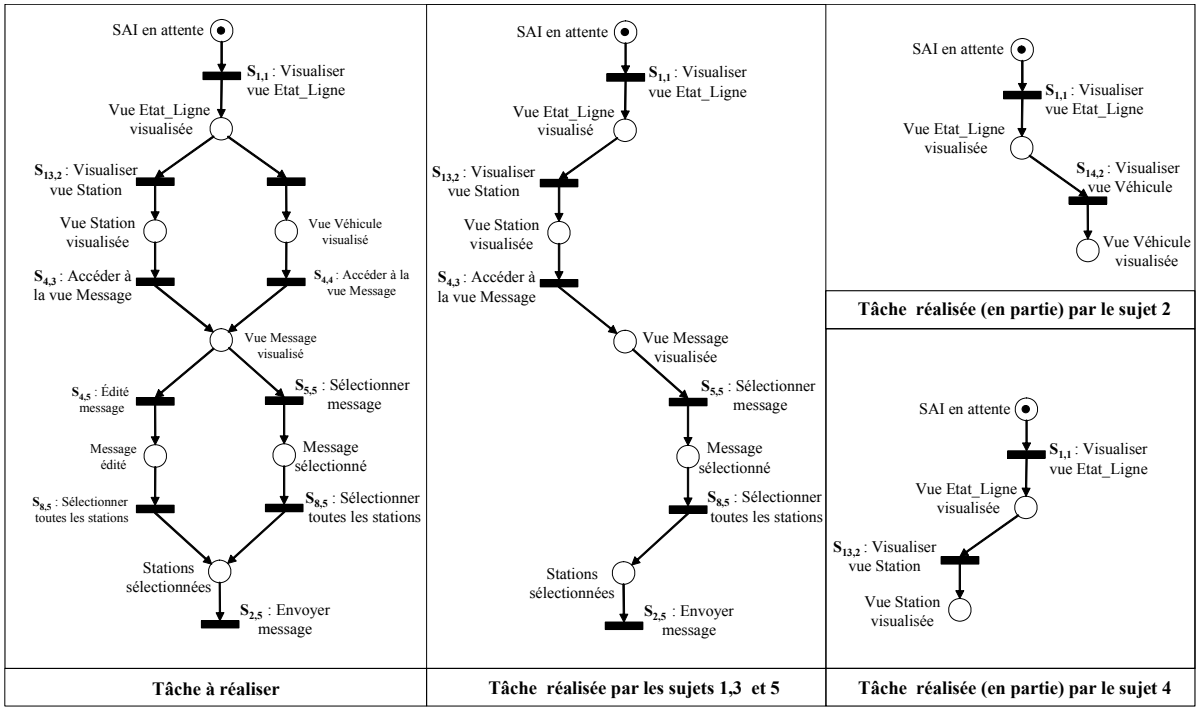



**Figure V.5.** Modélisation avec les RdP de la tâche à effectuer et de la tâche observée (Tâche 3, scénario 1)

Les RdP présents sur la figure V.5 montrent que les sujets numéro 2 et 4 n'ont pas réussi à accomplir leur tâche. En effet, les deux sujets n'arrivent pas à accéder à (afficher) la vue Message. Le sujet numéro 2 se bloque à la vue *Station* et le sujet numéro 4 se bloque à la vue Véhicule. Ce constat confirme les résultats du tableau V.3.

Ce problème de blocage peut être vu comme un problème d'utilisabilité. En effet, les résultats obtenus après l'évaluation du SAI avec le premier scénario révèle que le SAI ne permet pas un accès intuitif et facile à la Vue *Message*. Nous verrons plus loin (Cf. § III.1.1) quelques améliorations relatives à la vue *Etat_Ligne* qui visent à y introduire une zone spécifique pour l'envoi de message (Cf. figure V.9). Cette zone permettra à l'utilisateur d'accéder directement à la vue Message sans avoir à passer par la vue *Station* ou la vue *Véhicule*.

Par ailleurs, nous avons remarqué que les sujets numéro 2 et 4 ont effectué des consultations d'aide relative à la vue *Station* et la vue *Véhicule*. La figure V.6 qui est extraite de l'onglet *analyse* (Cf. Chapitre 3, § IV.2) du mouchard électronique montre le nombre d'utilisation des différents composants de la vue *Véhicule*. A la partie inférieure (Cf. figure V.6) on remarque que, pour le sujet numéro 1 (sujet qui a effectué l'ensemble de ces tâches

avec succès), l'aide n'a pas été sollicitée alors que pour le sujet numéro 2, l'aide a été sollicitée deux fois. Notons que, malgré la consultation de l'aide, le sujet numéro 2 n'est pas arrivé à réaliser ni la troisième ni la quatrième tâche avec succès. Il est donc nécessaire de revoir cette aide et de l'améliorer.

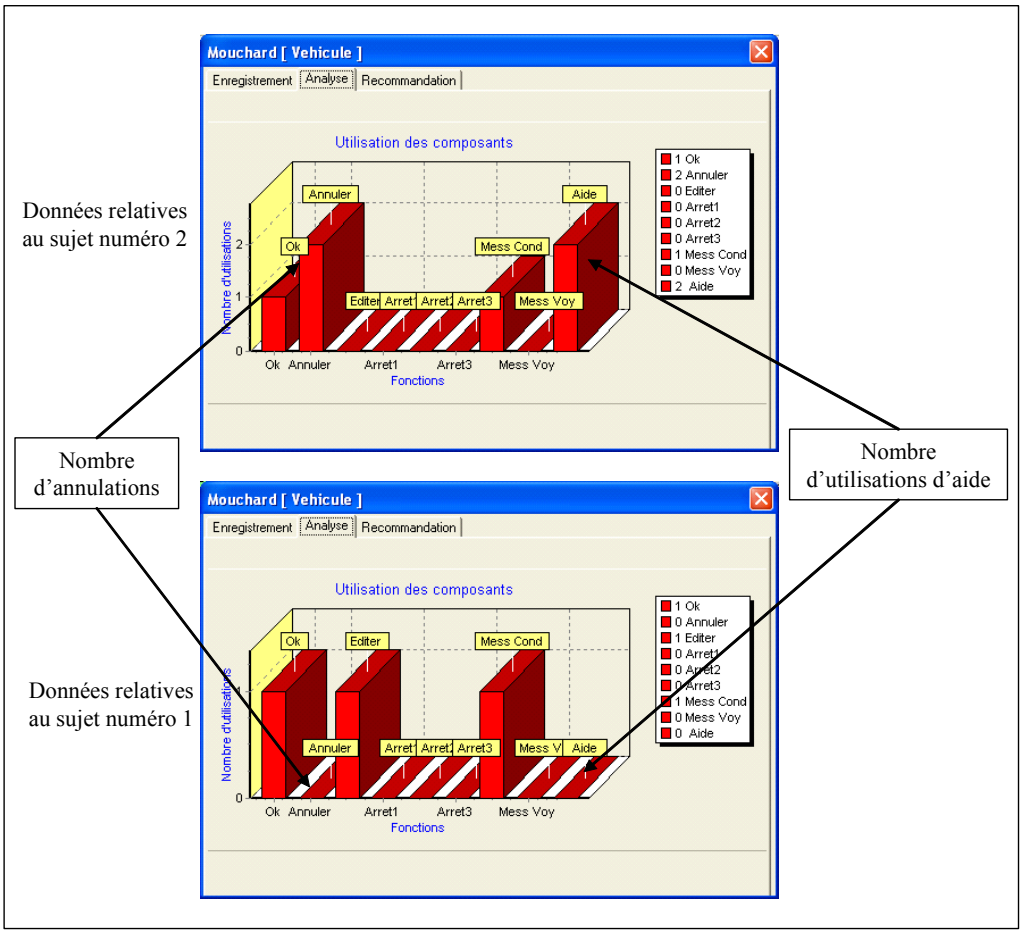



**Figure V.6.** Nombre d'utilisations des composants de la vue Véhicule par les sujets 2 et 4

On peut aussi noter des problèmes ponctuels révélés par les données recueillies grâce aux réponses au questionnaire et à la verbalisation. Le problème le plus important est celui de la difficulté d'accès aux stations et véhicules à partir de la vue *Etat_Trafic*. En effet, tous les sujets pensent qu'il est difficile de trouver une station ou un véhicule particulier sur la vue *Etat_Ligne*.

## II.4.2. Deuxième scénario : évaluation du SAI en mode de fonctionnement perturbé

Dans la réalité, en mode de fonctionnement anormal, les analyses ont montré que la tâche du régulateur se résume à réagir par des actions de régulation[46], aux messages d'avertissement, d'anomalie ou de panne en provenance du système. Le régulateur peut également envoyer des messages aux véhicules et aux stations de sa propre initiative.

[46] Par action de régulation, on sous-entend l'information qu'envoie le régulateur et qui est destinée aux conducteurs de véhicules et aux voyageurs en stations et dans les véhicules.

Pour s'assurer que l'utilisateur (le sujet) arrive facilement à interagir avec le SAI en effectuant des actions de régulation, nous proposons un deuxième scénario, complémentaire au premier, composé de quatre tâches qui sont maintenant décrites.

## II.4.2.1. Tâche à réaliser

Dans ce deuxième scénario, il s'agit d'exécuter quatre tâches prédéfinies. A l'inverse du premier scénario l'exécution des tâches n'est pas séquentielle. En effet, le début de chaque tâche est annoncé par un message en provenance du SAE[47] (Système d'Aide à l'Exploitation). Ainsi l'utilisateur (sujet) peut être en train d'exécuter une tâche relative à l'apparition d'un message et en recevoir un autre en même temps. Le tableau V.4 décrit la nature, l'instant d'apparition et le continu du message.

| Instant | Type de message | Message en provenance du SAE |
|---|---|---|
| t = 15 secondes | Avertissement | Message 1 : « Le bus N° 4 Ligne 16 est en avance de 5 min » |
| t = 35 secondes | Avertissement | Message 2 : « Le bus N° 2 Ligne 17 est en retard de 5 min » |
| t = 1 minute | Alarme | M3 : « Le véhicule N° 4 Ligne 16 est en panne » |
| t = 2 minutes | Anomalie | M4 : « Incident sur la ligne Tramway à proximité de la station Gare-SNCF » |

**Tableau V.4.** Scénario 2 : Messages affichés à l'utilisateur

Il est à noter que la durée entre l'apparition de deux messages est relativement courte ; Ceci n'est pas un hasard. En effet, nous souhaitons mettre l'utilisateur dans une situation proche de la réalité où plusieurs incidents peuvent survenir simultanément au sein du réseau.

Deux exemples de messages affichés à l'utilisateur sont illustrés par la figure V.7.

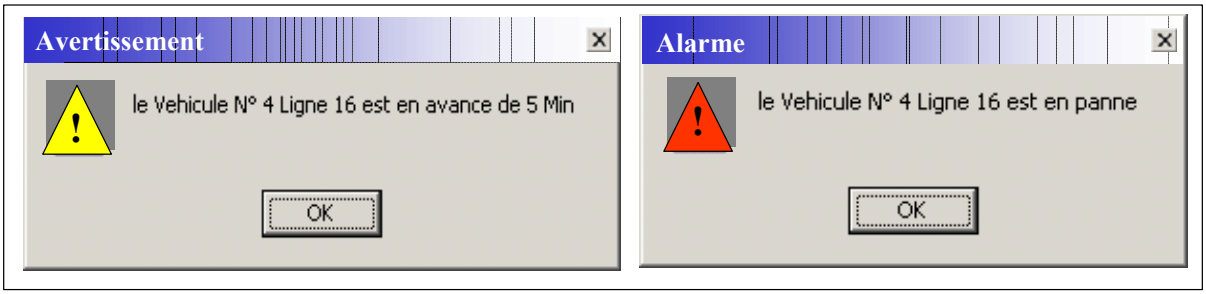



**Figure V.7.** Exemples de messages affichés à l'utilisateur

Le message à gauche de la figure V.7 représente un message d'avertissement. C'est un message à « basse priorité de régulation ». Le message à droite de la figure représente un message d'alarme. C'est un message à « haute priorité de régulation ».

[47] Vu que l'évaluation se déroule en laboratoire, la connexion entre le SAI et le SAE de l'exploitant industriel est impossible. Nous avons alors développé un mini-simulateur du SAE pour le maquettage du SAI et le déroulement de l'évaluation. Le SAI et le mini-simulateur ont été développés avec le langage C++ et particulièrement avec Borland C++ Builder (BCB). Nous avons choisi BCB pour sa facilité de prise en main ainsi que pour son aspect intuitif pour la conception d'interfaces complexes.

A chaque apparition d'un message, l'utilisateur doit effectuer la tâche de régulation correspondante. Le tableau V.5 montre la correspondance entre les messages reçus et à traiter, et les tâches à réaliser. Les durées théoriques nécessaires pour la réalisation des tâches sont déterminées par un expert en supervision initié au SAI.

| Tâches à réaliser | Message à traiter | Durée théorique du temps nécessaire pour la réalisation de la tâche | Description de la tâche |
|---|---|---|---|
| T1 | M1 | 55 secondes | Envoyer un message au conducteur du véhicule numéro 4 de la ligne 16 (Arrêtez-vous 2 minutes à la prochaine station) |
| T2 | M2 | 55 secondes | Envoyer un message au conducteur du véhicule numéro 2 de la ligne 17 (Vous êtes en retard, veuillez accélérer si possible) |
| T3 | M3 | 2 minutes | • Envoyer un message aux passagers du véhicule numéro 4 de la ligne 16 (Bus en panne, arrivée du prochain bus dans 15 minutes)<br>• Envoyer un message au conducteur du véhicule numéro 4 de la ligne 16 (Le service de dépannage arrive dans 10 minutes)<br>• Retirer le véhicule du réseau |
| T4 | M4 | 1 minute | Envoyer un message aux stations concernées (Trafic perturbé : accident sur la ligne Tramway) |

**Tableau V.5.** Scénario 2 : tâches à réaliser

Comme nous l'avons expliqué au premier scénario, le modèle de la tâche à réaliser sera confronté au le modèle de la tâche observée qui est déterminé à partir des enregistrements effectués avec le mouchard électronique.

Notons qu'à la fin de la réalisation du premier scénario, une brève explication aux sujets numéro 2 et 4 à propos de l'accès à la vue *Message* a été fournie. En effet, l'échec de ces deux sujets à l'accomplissement des tâches trois et quatre du premier scénario, nous a permis de détecter les problèmes liés à l'accès à la vue *Message*. Il était donc inutile que ces deux utilisateurs se bloquent à nouveau au même niveau.

## II.4.2.2. Résultats

Les cinq sujets ont réalisé les quatre tâches prévues au deuxième scénario. Le tableau V.6 obtenu grâce au mouchard électronique, présente un récapitulatif de la durée moyenne du temps de réalisation de chaque tâche ainsi que le taux de succès de sa réalisation.

Les résultats visibles au tableau V.6 montrent que les cinq sujets ont réussi à effectuer les quatre tâches qui leur ont été proposées. Cependant, on note un dépassement du temps théorique prévu pour effectuer les quatre tâches à réaliser. Cette constatation trouve une explication dans les données recueillies avec la verbalisation. En effet, tous les sujets, sans exception, font remarquer qu'il leur est impossible de mémoriser les messages en provenance du SAE. Par ailleurs, le SAI se bloque tant que l'utilisateur n'a pas validé la lecture du message ; l'utilisateur est donc obligé de mémoriser le message ou de le noter.

Une amélioration de la vue Etat_Trafic est proposée en perspective (Cf. III.1.1). Cette amélioration consiste à introduire une zone de sauvegarde des messages reçus en vue de leur traitement un à un.

| Tâches réalisées | Durée théorique du temps nécessaire pour la réalisation de la tâche | Durée réelle du temps de réalisation de la tâche (Moyenne) | Taux de succès de réalisation de la tâche |
|---|---|---|---|
| T1 | 55 secondes | 1 mn 10 s | 100 % |
| T2 | 55 secondes | 1 mn 30 s | 100 % |
| T3 | 2 minutes | 3 mn 20 s | 100 % |
| T4 | 1 minute | 1 mn 25 s | 100 % |

**Tableau V.6.** Tableau récapitulatif de la durée d'exécution des tâches et les taux de succès

Par ailleurs, la reconstruction des réseaux de Petri des tâches réalisées n'a pas révélé de problème particulier. En effet, mise à part la tâche trois, les modèles de la tâche à réaliser et de la tâche réalisée ont une parfaite similitude.

Le modèle de la tâche à réaliser de la troisième tâche ainsi que les modèles de la tâche réalisée par les cinq sujets sont visibles à la figure V.8. Nous remarquons que les sujets numéro 1, 2, 3 et 5 ont parfaitement réussi leur tâche ; alors que le sujet numéro quatre a exécuté des actions inutiles avant d'arriver à la fin de la réalisation de sa tâche.

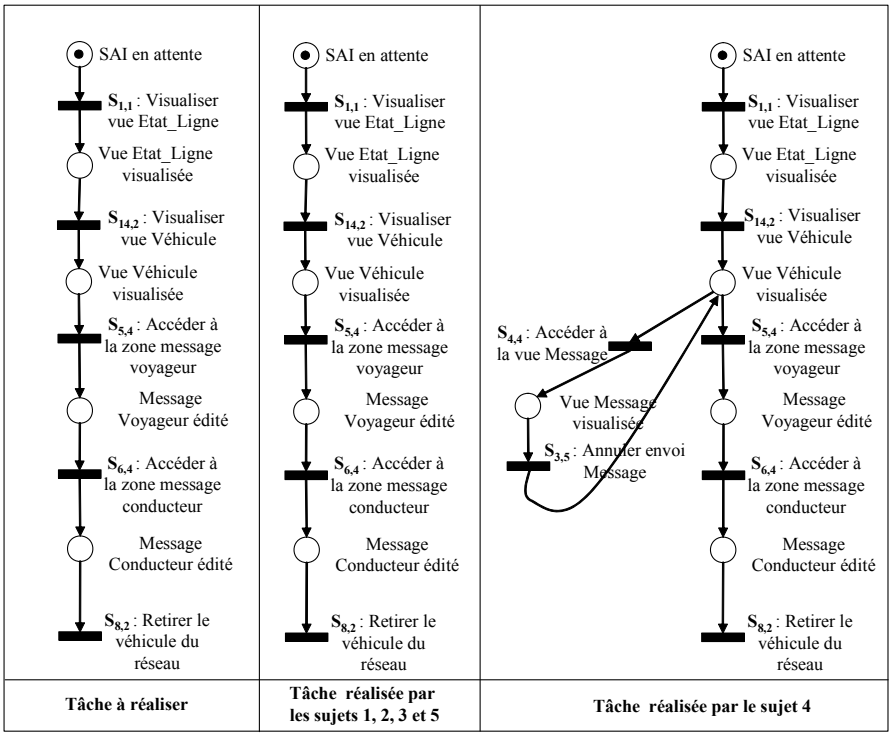



**Figure V.8.** Modélisation avec les RdP de la tâche à effectuer et de la tâche observée (Tâche 3, scénario 2)

En consultant les données enregistrées par le mouchard électronique, on s'aperçoit que chacun des cinq sujets a effectué un accès à l'aide relative aux vues Etat_Trafic et Etat_Ligne. En effet, la verbalisation a révélé, dès que le deuxième message est apparu, les sujets espéraient disposer d'un moyen pour retrouver une trace du premier message à la vue Etat_Trafic ou à la vue Etat_Ligne.

## II.5. Conclusion sur l'évaluation expérimentale du SAI

La première évaluation en laboratoire, proposée dans cette section, nous a permis de détecter un premier ensemble représentatif de problèmes d'utilité et/ou d'utilisabilité inhérents à l'exploitation du Système d'Aide à l'Information. Cinq sujets, doctorants en informatique, ont participé à l'évaluation.

Cette évaluation a porté sur deux scénarios tirés de la réalité. Le premier a assuré l'évaluation du SAI en mode de fonctionnement normal en proposant aux sujets quatre tâches à réaliser ne nécessitant aucune régulation. Le deuxième quant à lui, a permis l'évaluation du SAI en mode de fonctionnement perturbé tout en proposant aux sujets quatre tâches à réaliser nécessitant leur réaction à des messages d'avertissement, d'anomalie et d'alarme.

Basée sur l'outil d'aide à l'évaluation, le questionnaire et la verbalisation, les résultats obtenus de l'évaluation du SAI, nous permettront de proposer des améliorations possibles à apporter au SAI. Toutefois, comme nous l'avons déjà signalé, il est clair que les résultats obtenus dans le cadre de cette première évaluation ne seront jamais aussi riches que ceux qui auraient pu être obtenus avec des utilisateurs professionnels de la régulation.

Par ailleurs l'évaluation du SAI nous a permis de tester techniquement le système d'aide à l'évaluation proposé ainsi que le couplage entre le mouchard électronique MESIA et le SAI. En effet, le couplage entre le SAI et le mouchard électronique a été effectué selon le principe énoncé précédemment (Cf. chapitre quatre, § IV.3).

Nos perspectives de recherche à court et à long terme sont maintenant présentées.

# III. Perspectives de recherche et développement

Nos perspectives se divisent en deux grandes parties.

La première partie concerne nos perspectives à court terme et porte sur l'amélioration du Système d'Aide à l'Information, particulièrement concernant son interface homme-machine. Nous abordons également dans cette partie la préparation de l'évaluation du SAI « sur le terrain » avec des utilisateurs professionnels provenant de l'exploitant du réseau de transport collectif.

Nous abordons dans la deuxième partie nos perspectives à moyen et à long termes. Nous y proposons des améliorations possibles concernant le système d'aide à l'évaluation. Ces améliorations portent sur les deux modules suivants : *simulation/confrontation/spécification de RdP agent* et *génération de RdP*. Des principes sur la « généralisation » de l'acquisition des données recueillies par le mouchard MESIA aux agents contrôleurs de dialogue et application sont également donnés. Afin de faciliter l'application de notre outil d'aide à l'évaluation dans le cadre d'un nouveau projet lié à la supervision et basé sur un système interactif orienté agent, nous visons l'élaboration d'une méthode de mise en place d'un

couplage entre architecture à base d'agents du système interactif et son évaluation. La formalisation et l'intégration de règles ergonomiques dans l'outil d'aide à l'évaluation font aussi partie de nos perspectives de recherche.

## III.1. Perspectives concernant le Système d'Aide à l'information

Les améliorations du système d'aide à l'information présentées dans cette section résultent de la première évaluation effectuée en laboratoire ; elles concernent plusieurs vues du SAI. Nous proposons également dans cette section les principes liés à une deuxième évaluation « sur le terrain ».

### III.1.1. Premières améliorations du SAI

En tenant compte des résultats d'évaluation du SAI obtenus avec les deux scénarios proposés, les réponses au questionnaire et la verbalisation, nous proposons des améliorations du SAI concernant principalement les vues *Etat_Trafic*, *Etat_Ligne* et *Message*.

Améliorations relatives à la vue Etat_Trafic :

La figure V.9 présente une amélioration possible (sous forme de maquette) de la Vue *Etat_Trafic* pour une meilleure utilisabilité. Nous proposons d'introduire (en haut de la fenêtre) une zone spécifique aux messages d'avertissement, d'anomalie et d'alarme. En effet, quand un message apparaît, l'utilisateur en prend connaissance et le valide (en appuyant sur le bouton Ok). Ainsi, au lieu d'être perdu, le message pourrait être placé automatiquement dans la zone prévue pour les messages. De cette façon, l'utilisateur n'aurait plus besoin de mémoriser des messages ou de les noter sur papier. Cette façon de présenter les messages permettrait à l'utilisateur de les traiter un à un et de les supprimer une fois traités.

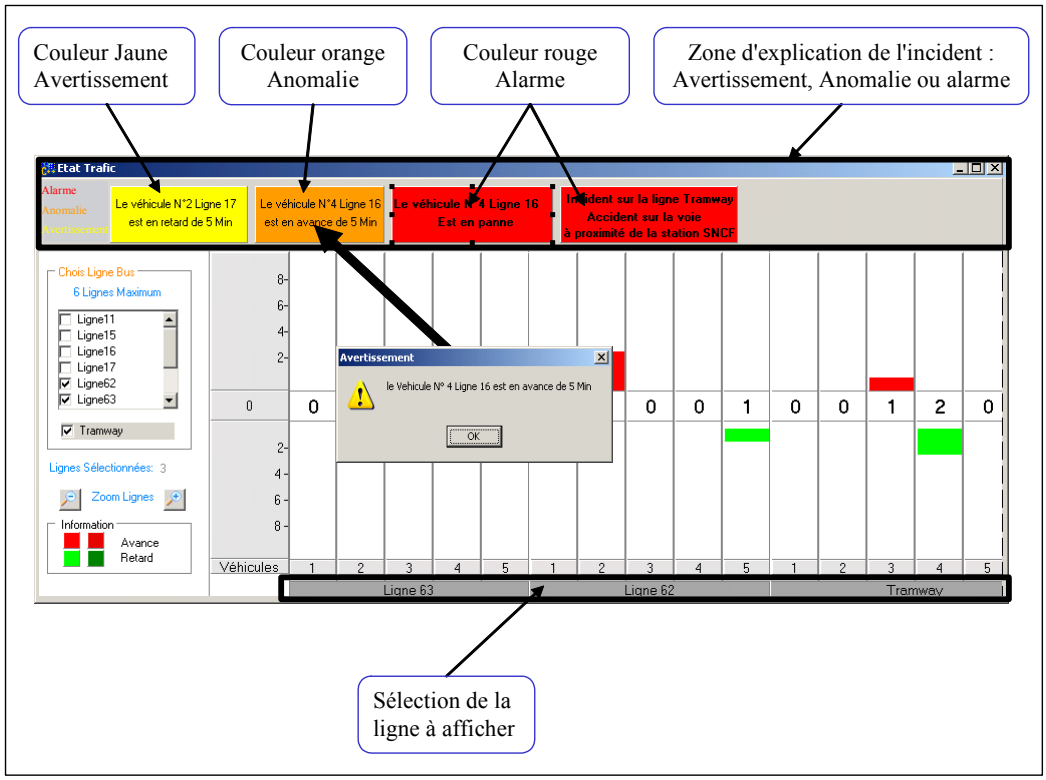



**Figure V.9.** Exemple d'amélioration de la vue *Etat_trafic*

Améliorations relatives à la vue Etat_Ligne :

Les résultats obtenus lors de l'évaluation du SAI ont montré que parmi les cinq utilisateurs qui ont participé à l'évaluation, deux n'ont pas réussi à accomplir la troisième et la quatrième tâche du premier scénario. Rappelons que ces deux tâches visaient l'envoi de message à plusieurs stations ou plusieurs véhicules. Le réseau de Petri de la tâche réalisée (Cf. figure V.8) montre que les deux utilisateurs n'arrivent pas à accéder à la vue *Message*. Ils se bloquent soit à la vue *Station* soit à la vue *Véhicule*.

Pour résoudre ce problème lié à l'ergonomie de l'interface, nous proposons d'introduire à la vue *Etat_Ligne* une zone spécifique pour l'envoi de message (Cf. figure V.10). Cette zone permettrait à l'utilisateur d'accéder directement à la vue Message sans avoir à passer par la vue *Station* ou la vue *Véhicule*.

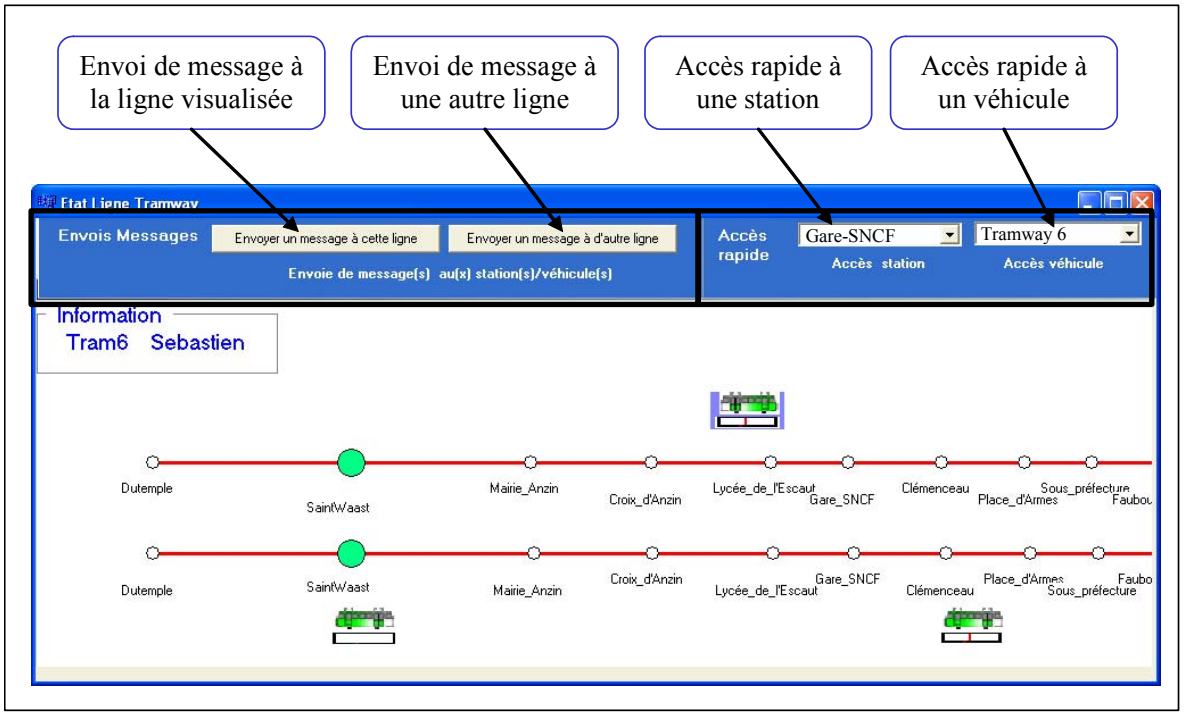



**Figure V.10.** Exemple d'amélioration de la vue Etat_Ligne

Améliorations relatives à la vue Message :

Suite aux changements effectués sur la vue *Etat_Trafic*, la vue *Message* devrait être aussi changée. En effet, si l'utilisateur désire envoyer un message à l'ensemble des stations d'une ligne spécifique, il serait intéressant de masquer les informations relatives aux lignes non concernées. Cette amélioration concerne l'ergonomie de la vue ; en effet il nous paraît recommandé de masquer les informations inutiles qui peuvent nuire au déroulement de la tâche de l'utilisateur.

Autres améliorations possibles :

Nous envisageons aussi de finaliser l'implémentation de la vue *vue_globale*. Cette vue est potentiellement intéressante pour la supervision dans la mesure où elle fournit une vision globale de l'ensemble du réseau.

Par ailleurs, nous avons constaté lors de l'évaluation que les messages édités par l'utilisateur ne sont pas enregistrés par le SAI. En effet, si l'utilisateur désire envoyer le même message deux fois, il est obligé de le rééditer.

Les améliorations proposées ci-dessus sont sans doute insuffisantes pour aboutir à une version finale du SAI. En effet, la deuxième évaluation (évaluation sur le terrain avec des régulateurs) permettra elle aussi de détecter d'autres problèmes d'utilité et/ou utilisabilité liés au SAI. Des principes relatifs à cette évaluation sont maintenant présentés.

## III.1.2. Deuxième évaluation du SAI : évaluation « sur le terrain »

La première partie de ce chapitre a fait l'objet d'une évaluation en laboratoire avec des utilisateurs non experts au domaine de la supervision du trafic terrestre. Cette évaluation a fourni des résultats intéressants et a donné lieu à des premières propositions d'amélioration du SAI (Cf. § III.1.1). Néanmoins, une évaluation en laboratoire ne permet pas la prise en compte ni du contexte ni de l'environnement habituel du travail.

Ainsi, une deuxième évaluation sur le terrain s'avère nécessaire pour la détection des problèmes d'utilité et/ou d'utilisabilité non décelés lors de la première évaluation. Nous décrivons ci-dessous les principales différences entre la première et la deuxième évaluation en termes de dispositif et de protocole expérimentaux.

- **Dispositif expérimental** : le dispositif expérimental utilisé pour cette deuxième évaluation serait globalement le même que celui utilisé lors de la première évaluation, tout en ajoutant l'exploitation de la technique de mesure des mouvements oculaires (Cf. chapitre 2, § III.1.1.4). L'introduction de cette technique nous permettra d'enregistrer en continu les mouvements du regard de l'utilisateur au fur et à mesure de son interaction avec le SAI. Ces enregistrements seront ensuite déployés afin de fournir les zones de fixation du regard et les sauts d'une zone à une autre. La décomposition du SAI en six zones de fixation nous semble être une conséquence logique du fait que le SAI est structuré en six vues. Cependant, lors de l'utilisation de l'oculomètre, pour obtenir des données faciles à déployer et des résultats corrects, les vues du SAI ne doivent pas se superposer. Or, tel qu'il est développé actuellement, le SAI ne tient pas compte de cette contrainte. Pour pallier ce problème, une restructuration spatiale des vues du SAI tel que visible sur la figure V.11 peut être adoptée. Nous prévoyons aussi d'effectuer cette évaluation avec des écrans de grande taille pour une meilleure lisibilité des informations délivrées par le SAI.

  Notons que le dispositif expérimental lié à l'oculomètre est relativement complexe à mettre en place et à manipuler (installation, calibrage, etc.). Afin de bien l'exploiter, un spécialiste en oculométrie de notre laboratoire[48] sera présent tout au long de l'évaluation.

  Le questionnaire utilisé lors de la première évaluation sera légèrement modifié et augmenté. En effet face à des utilisateurs experts en régulation, des questions du

[48] Le LAMIH utilise depuis la fin des années 80 l'oculomètre dans le cadre de nombreux projets, le plus souvent en lien avec des entreprises (Cf. par exemple les travaux de [Abed, 1990, 2001], [Ezzedine, 2002] ou bien la thèse de [Simon 1993]).

type : « est-ce qu'il manque des fonctionnalités à cette vue ? Si oui lesquelles ? » peuvent révéler des résultats pertinents.

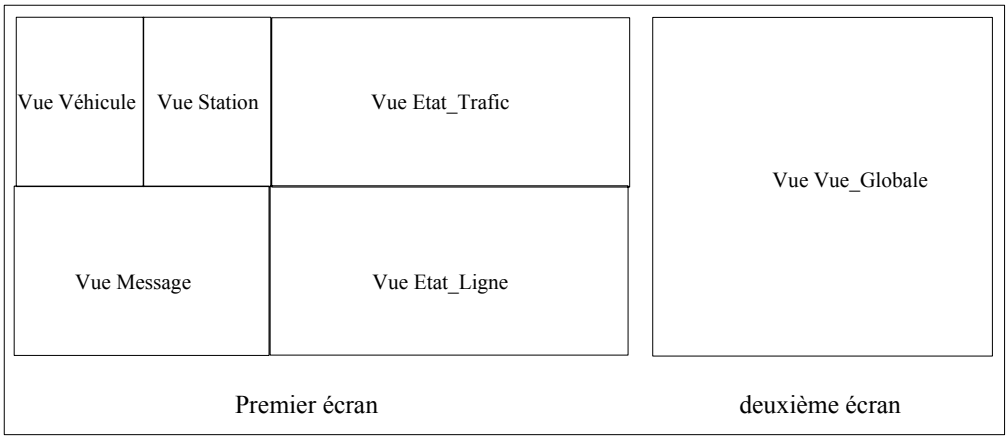



**Figure V.11.** Une distribution spatiale possible des vues du SAI

Parallèlement à l'utilisation de l'oculomètre, nous prévoyons l'utilisation d'une caméra pour enregistrer la scène de l'expérience. En effet, certains gestes et mimiques de l'utilisateur peuvent révéler à l'évaluateur des problèmes d'utilisabilité.

- **Protocole expérimental** : Des scénarii plus complexes que ceux utilisés pour l'évaluation en laboratoire seront définis ; par exemple : régulation d'un incident qui touche deux lignes en même temps, gestion de plusieurs alarmes en parallèle, etc.

## III.2. Perspectives concernant le système d'aide à l'évaluation

Les perspectives concernant le système d'aide à l'évaluation sont avant tout des perspectives à court et moyen terme. Elles constituent des objectifs prioritaires pour la suite de nos travaux. Elles concernent l'amélioration du module Simulation/Configuration/Spécification, la « généralisation » de l'acquisition des données recueillies par le mouchard MESIA aux *agents contrôleurs de dialogue* et *application*, la proposition d'une méthode de mise en place d'un couplage entre architecture à base d'agents du système interactif et son évaluation et la formalisation et l'intégration des règles ergonomiques au système d'aide à l'évaluation.

## III.2.1. Module Simulation/Confrontation/Spécification

Comme son nom l'indique, ce module offre aux évaluateurs/concepteurs les trois fonctionnalités suivantes : la simulation de RdP agents, la confrontation de RdP agents, la spécification de RdP agents. Nous proposons ci-dessous quelques pistes de recherches relatives à chaque fonctionnalité.

Fonction Simulation :

Cette fonction permet la visualisation de la dynamique du RdP agents et par conséquent la dynamique de l'IHM, ceci par l'exploitation des modèles de tâches ainsi que la formulation mathématique qui assure l'évolution dans le RdP agents. Sans l'introduction d'une variable

« temporelle », cette fonctionnalité se limite à simuler le déclenchement du service correspondant à l'événement apparu. Autrement dit, en simulant l'apparition d'un événement on observe directement les conséquences sur la dynamique de l'IHM.

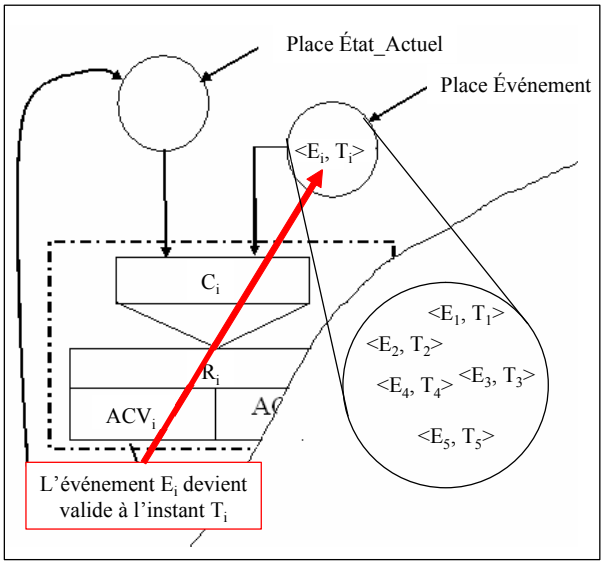



**Figure V.12.** Introduction d'une variable temporelle aux RdP agent

En associant une variable Ti à chaque événement, il serait possible de définir un ensemble de couples <Ei, Ti>. En introduisant ces couples à la place Evénement (Cf. figure V.12), une simulation des enclenchements des services correspondant à chaque événement est possible. Nous nous inspirerons des travaux de [David et Alla, 1992] pour peaufiner cette idée.

Fonction Confrontation :

Cette fonction exploite les modèles de tâche (Observée, Référence) pour la confrontation. Cette dernière a pour objectif de faciliter aux évaluateurs l'identification d'éventuels problèmes ergonomiques liés à l'utilisabilité du système interactif ; par exemple s'apercevoir que le RdP agents du modèle de la tâche observée contient des états en plus, c'est-à-dire que l'utilisateur passe par des étapes inutiles, ou alors que le temps pris pour réaliser une tâche est bien supérieur à celui prévu au départ par les évaluateurs/concepteurs.

Cependant, effectuer la confrontation entre les réseaux de Petri s'avère être une tâche assez complexe. Ainsi, nous proposons la confrontation des enchaînements de services. Autrement dit, la confrontation s'effectue entre l'enchaînement des services de la tâche à réaliser et des tâches réalisées (observées).

Fonction Spécification:

Cette fonction consiste en la mise à disposition des évaluateurs/concepteurs de moyens (fenêtres) permettant la gestion (saisie, modification,…) de la spécification des agents, autrement dit la définition des ensembles E, C, R, Acv, Acn, et leur stockage dans la base de spécifications des agents.

Nous avons présenté précédemment (Cf. chapitre 4, figure IV.6) un outil interactif destiné à la spécification des *agents d'interface*. Nous prévoyons d'ajouter à cet outil la fonctionnalité de création de scénario. En effet, un scénario sera composé de tâches, elles-mêmes composées

d'un ensemble de services. Ainsi, nous mettons à la disposition de l'évaluateur un outil interactif pour la création du modèle de la tâche à réaliser. La spécification fournie par cet outil sera exploitée par le module de génération de RdP pour la modélisation de la tâche à réaliser avec le RdP.

## III.2.2. Vers un couplage entre MESIA et les agents contrôleurs de dialogue et application

Nous avons vu au troisième chapitre le principe de couplage entre le mouchard électronique et le système à évaluer. En effet, l'existence d'un agent au niveau du système interactif et particulièrement le composant présentation conduit à la création d'un *agent mouchard*.

Nous visons la généralisation de ce principe aux *agents contrôleurs de dialogue* et aux *agents application*. Ainsi nous suggérons d'associer un *agent mouchard* à chaque a*gent contrôleur de dialogue* et à chaque *agent d'application*. Une fois ce principe mis en œuvre, le mouchard électronique permettrait l'acquisition des comportements de ces agents. Les comportements pourraient être utilisés pour : l'évaluation des temps de réponse de l'application, la vérification de la correspondance entre les données affichées et les données réellement existantes dans le processus (application), etc. Par exemple si un *agent d'application* reçoit une information de retard d'un véhicule par le Système d'Aide à l'Exploitation, il transmet directement l'information à l'*agent contrôleur de dialogue* qui la transmet à *l'agent d'interface* correspondant. Ainsi, l'acquisition des comportements des *agents contrôleurs de dialogue* et *application* nous permettrait par exemple de déterminer et d'évaluer le temps de « propagation » du message.

## III.2.3. Formalisation et intégration de règles ergonomiques au système d'aide à l'évaluation

L'un des intérêts du système d'aide à l'évaluation est de proposer des recommandations à l'évaluateur (Cf. chapitre 4, IV.2). Ces recommandations varient des plus basiques, avec des messages d'avertissement basés sur les données recueillies par le mouchard électronique, comme par exemple des recommandations de type : « Attention utilisation excessive de l'aide pour cette vue », aux plus complexes avec une critique de l'aspect ergonomique de l'interface comme par exemple des recommandations de type : « attention la couleur utilisée pour les messages d'alerte n'est pas adéquate ».

Nous décrivons ci-dessous une piste de recherche pour l'intégration de règles ergonomiques à l'outil d'aide à l'évaluation.

Comme précisé dans [Vanderdonckt et Bodart, 1994 ; Vanderdonckt, 1998], il est impossible d'utiliser efficacement un ensemble de règles ergonomiques, aussi grand et précis soit-il, au sein d'un système automatique pour l'évaluation de n'importe quel système. En effet, l'utilisation des règles ergonomiques dépend considérablement du domaine d'application. Ainsi, dans notre cas de première tentative d'intégration de règles ergonomiques au système d'aide à l'évaluation, nous nous limiterons à un ensemble restreint de règles en rapport avec le domaine de la supervision.

La figure V.13 présente une première proposition d'intégration de règles ergonomiques à l'outil d'aide à l'évaluation. Nous nous basons principalement sur le mouchard électronique MESIA. En effet, nous suggérons d'essayer de doter le mouchard électronique d'un module d'acquisition des propriétés des ressources utilisées par le service d'un agent. On sous-entend ici les propriétés basiques d'un composant graphique tels que : la largeur, la longueur, la couleur, etc. Ces propriétés seront exploitées par un système expert (inspiré des travaux effectués sur les systèmes experts pour l'évaluation des IHM (Cf. chapitre 2, § IV.2). Le système expert intégrerait une base de règles ergonomiques implémentées selon les principes énoncés par [Balbo, 1994].

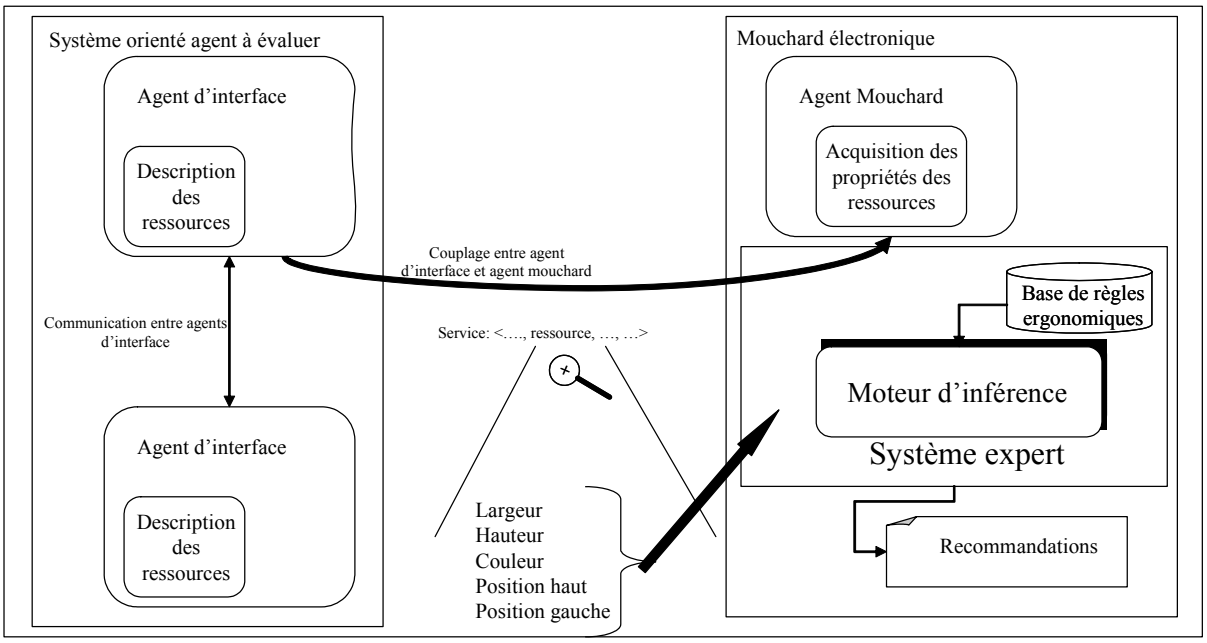



**Figure V.13.** Intégration des principes ergonomiques à l'outil d'aide à l'évaluation

## III.3. Vers une méthode de mise en place d'un couplage entre architecture à base d'agents du système interactif et son évaluation

L'évaluation du SAI présentée au début de ce chapitre nous a permis de tester techniquement le système d'aide à l'évaluation proposé ainsi que le couplage entre le mouchard électronique MESIA et le SAI. Cependant, l'application de notre outil d'aide à l'évaluation dans le cadre d'un nouveau projet lié à la supervision et basé sur un système interactif orienté agent, peut être facilitée par l'élaboration d'une méthode qui précise les différentes étapes à suivre.

C'est dans cet état d'esprit que nous visons l'élaboration d'une méthode pour la mise en place du couplage entre architecture à base d'agent du système interactif et son évaluation. Cette méthode pourrait se baser sur les cinq étapes suivantes :

- la spécification des agents d'interface du système à évaluer selon l'architecture proposée au troisième chapitre. Cette spécification serait assistée par l'outil interactif destiné à la spécification des agents d'interface (Cf. figure IV.6). À l'issue de cette étape, la spécification des services, événements, conditions, ressources, actions visibles et actions non visibles pour chaque agent sera disponible dans la BSA (Base de Spécification des Agents).

- la spécification des modèles de référence des tâches à réaliser dans la BMT (R) (Cf. chapitre III, section IV.2.3.2). Cette spécification serait introduite par les concepteurs/évaluateurs via le module Simulation-Confrontation-Spécification de RdP agents.

- le couplage entre le système à évaluer et le mouchard électronique MESIA. Ce couplage serait basé sur l'association d'un *agent mouchard* à chaque *agent d'interface* du système à évaluer (Cf. chapitre III, section IV).

- l'élaboration du protocole expérimental et le recueil des données. Cette étape consisterait à déterminer en premier lieu, le protocole expérimental de l'évaluation (scénario, tâches à réaliser, sujets…) (Cf. chapitre V, section II.4). En deuxième lieu, il s'agirait de recueillir les actions/réactions de l'utilisateur et du système à évaluer en se basant sur le système d'aide à l'évaluation et particulièrement sur le mouchard électronique MESIA (Cf. chapitre V, section II.2.4).

- le traitement des données recueillies par le mouchard électronique et d'autres techniques et méthodes complémentaires (questionnaire, verbalisation…). Cette étape nous permettrait d'apprécier le degré d'utilité et d'utilisabilité de IHM du système à évaluer afin de l'améliorer et de la valider si possible par les utilisateurs finaux.

## III.4. Vers d'autres validations de l'outil d'aide à l'évaluation

Enfin nous souhaitons appliquer notre outil d'aide à l'évaluation dans le cadre d'un autre projet lié à la supervision et basé sur un système interactif orienté agent, ce qui nous permettra de continuer à le tester et l'améliorer. Par ailleurs, l'application de l'outil d'aide à l'évaluation à un autre projet nous permettra de trouver des pistes pour la proposition d'un outil d'aide à l'évaluation qui, nous le souhaitons à long terme, sera générique. Nous souhaitons également valider notre outil d'aide à l'évaluation, après amélioration, avec des évaluateurs professionnels afin d'avoir un retour sur son utilité et ses limites. Ces perspectives à long terme restent le moyen le plus efficace pour une validation « correcte » du système d'aide à l'évaluation et particulièrement du mouchard électronique MESIA.

# IV. Conclusion

L'évaluation en laboratoire du Système d'aide à l'Information a mis en évidence un premier ensemble représentatif de problèmes d'utilité et/ou d'utilisabilité inhérents à l'exploitation du SAI. Malgré cela, cette évaluation en laboratoire reste, sans aucun doute, insuffisante pour aboutir à une version finale du SAI. En effet, la deuxième évaluation « sur le terrain » avec des utilisateurs professionnels provenant de l'exploitant du réseau de transport collectif permettra elle aussi de détecter d'autres problèmes d'utilité et/ou utilisabilité liés au SAI. Néanmoins, suite aux résultats obtenus par cette première évaluation, nous avons proposé des améliorations possibles du SAI.

Par ailleurs, grâce à l'évaluation du SAI, nous avons pu tester notre outil d'aide à l'évaluation ainsi que le couplage entre le SAI et le mouchard électronique MESIA. Nous pensons que différents points relatifs à l'outil d'aide à l'évaluation peuvent être améliorés et

augmentés afin de faciliter davantage la tâche de l'évaluateur. L'extension du couplage du Mouchard électronique aux agents contrôleurs de dialogue et application peut consolider l'évaluation des systèmes interactifs orientés agents.

Différentes perspectives, à court et à long terme, ont été proposées à la fin de ce chapitre. Elles concernent essentiellement la deuxième évaluation du SAI ainsi que l'amélioration de différents modules du système d'aide à l'évaluation.

# *Conclusion Générale*

Le travail présenté dans ce mémoire contribue à l'évaluation des systèmes interactifs, particulièrement ceux qui sont orientés agent, avec pour toile de fond les systèmes industriels complexes. Cette contribution comporte deux principaux aspects : la proposition d'un principe global d'architecture dédiée aux systèmes interactifs à base d'agents, cette architecture étant modélisée en utilisant les réseaux de Pétri de haut niveau (en l'occurrence les réseaux de Petri agents) ; la proposition d'un outil d'aide à l'évaluation dédié aux systèmes interactifs orientés agents basé sur un mouchard électronique baptisé MESIA (Mouchard électronique dédié à l'Evaluation des Systèmes Interactifs orientés Agents).

Ainsi, à travers le premier chapitre, nous avons présenté sans souci d'exhaustivité mais plutôt de représentativité, des architectures de systèmes interactifs. Trois tendances se dégagent pour les architectures recensées dans la littérature. Certaines architectures décomposent l'interface en différents niveaux logiciels visant une séparation nette entre l'interface et l'application tels les modèles : Langage, Seeheim et Arch qui sont des modèles qualifiés de centralisés. D'autres approches qualifiées de réparties, appelées aussi orientées agent (cette dénomination étant utilisée en IHM même si les agents intégrés dans ces approches ne se situent pas au même niveau que les agents intelligents étudiés dans le domaine de l'intelligence artificielle distribuée), ont été abordées : les plus connus sont MVC, PAC, AMF et leurs variantes. Leur apparition résulte de l'avènement de la programmation orientée objet et d'une certaine manière met aussi en avant la notion de stimulus-réponse (on est donc plutôt au niveau des agents dits réactifs considérés en intelligence artificielle distribuée). Le troisième type de modèle d'architecture, qualifié d'hybride, tels que : PAC-AMODEUS, MultiCouche et le modèle $H^4$ tente de tirer profit des deux types d'architectures précédentes en reprenant l'avantage de chacun d'eux. Pour notre part, nous nous sommes situés dans les architectures dites hybrides.

Après avoir étudié les trois types d'architectures citées ci-dessus, on s'aperçoit qu'elles ne sont pas vraiment adéquates pour la modélisation des systèmes interactifs de supervision. En effet, les architectures centralisées manquent de structuration et de formalisme de communication entre les modules, ce qui constitue un problème pour la modélisation des différents modules (contribuant au contrôle et à la commande de système industriel complexe). Les architectures réparties, quant à elles, peuvent déboucher sur une perception difficile de l'interface globale pour le concepteur d'une application de supervision (avec une interface constituée de nombreux éléments graphiques faisant l'objet parfois d'une multitude de vues) car l'interface est répartie entre les différents composants *Présentation* de chaque agent. Par ailleurs, l'idée d'architecture hybride (jamais encore préconisée à notre connaissance dans les systèmes complexes), nous semble prometteuse, en essayant d'exploiter les avantages des deux catégories précédentes. Ainsi, nous avons proposé au troisième chapitre les principes de base d'une architecture hybride nous semblant adaptée aux systèmes de supervision, celle-ci s'appuyant sur une combinaison entre le modèle de Seeheim (modèle centralisé) et la notion d'agent (modèle réparti), sans être nécessairement basé sur une arborescence d'agents comme dans PAC.

Le deuxième chapitre a présenté un état de l'art sur les méthodes et techniques usuelles d'évaluation de systèmes interactifs, tout en mettant en avant quelques outils d'aide à l'évaluation automatique ou semi-automatique. Quelle que soit la méthode d'évaluation choisie, l'objectif consiste à apprécier le degré d'utilité et d'utilisabilité d'une IHM afin de l'améliorer et de la valider si possible par les utilisateurs finaux (mais pour différentes raisons il s'avère que ceux-ci ne sont pas toujours disponibles). Nous avons présenté trois types d'approches d'évaluation : les méthodes centrées sur l'utilisateur, les approches basées sur des experts et enfin les approches analytiques. Enfin et sans souci d'exhaustivité nous avons présenté des outils d'aide à l'évaluation automatique : SYNOP, KRI/AG, ERGOVAL, CHIMES, EMA. Après la présentation d'une variété de méthodes contribuant à l'évaluation des interfaces homme-machine, nous avons constaté, qu'à notre connaissance, il n'existe pas, au stade actuel de la recherche, d'outil ou de méthode spécifique à l'évaluation des systèmes interactifs possédant une architecture à base d'agents.

Ainsi, nous avons proposé au troisième chapitre un outil d'aide à l'évaluation dédié aux systèmes interactifs à base d'agents, l'outil étant fondé sur le principe de mouchard électronique.

Le principe de couplage entre architecture à base d'agents de système interactif et son évaluation a été présenté au troisième chapitre. Nous avons proposé un nouvel outil d'aide à l'évaluation des systèmes interactifs orientés agents, baptisé MESIA (Mouchard électronique dédié à l'Evaluation des Systèmes Interactifs orientés Agents). Ce mouchard est constitué de plusieurs agents mouchard déduits à partir de l'architecture du système à évaluer et plus particulièrement à partir des agents de présentation. Le système d'aide à l'évaluation est constitué de trois modules : le module mouchard électronique, le module générateur de réseaux de Petri agent qui exploite les données automatiquement recueillies par le mouchard et le module Simulation-Confrontation-Spécification de RdP agents qui offre aux évaluateurs/concepteurs les trois fonctionnalités suivantes : la simulation de RdP agents, la confrontation de RdP agents et la spécification de RdP agents.

Le quatrième chapitre a permis de présenter une mise en application dans un contexte lié aux transports, en l'occurrence le projet SART (Système d'Aide à la Régulation de Trafic) dans lequel notre apport concerne le SAI (Système d'aide à l'information) intégré à terme dans la salle de régulation d'un réseau de bus et de tramway (et connecté à d'autres outils d'aide). Cette mise en application concerne en premier lieu l'architecture ; nous avons pour cela exploité les réseaux de Petri agent pour la modélisation du SAI, et en deuxième lieu le couplage entre l'architecture du SAI et le mouchard électronique (en vue de l'évaluation du SAI). Une première version de l'outil d'aide MESIA a donc été réalisée.

Enfin, une première évaluation du Système d'Aide à l'Information (SAI) a été présentée dans le cinquième chapitre. Cette évaluation nous a permis, dans un premier temps de tester notre outil d'aide à l'évaluation ainsi que le couplage entre le mouchard électronique MESIA et le SAI, et dans un deuxièmes temps de déterminer plusieurs problèmes d'utilité et d'utilisabilité liés au SAI. Ainsi, nos perspectives de recherche à court terme concernent l'amélioration du SAI en tenant compte des résultats obtenus lors de cette première évaluation. Une évaluation du SAI avec des experts en régulation est envisagée.

Nos perspectives de recherche à moyen et à long terme présentent une piste pour l'intégration de règles ergonomique dans l'outil d'aide à l'évaluation fournie aux évaluateurs. Nous nous intéressons aussi à la « généralisation » de l'acquisition des données recueillies par le mouchard MESIA aux agents contrôleurs de dialogue et application. Des possibilités d'amélioration du module de spécification, génération et confrontation de RdP ont été proposées. Les principes d'une méthode de mise en place d'un couplage entre architecture à base d'agents du système interactif et son évaluation ont été présentés.

Les résultats obtenus lors des évaluations sont encourageants et permettent d'envisager de tester le mouchard électronique MESIA dans d'autres domaines.

# *Bibliographie*


**[Abed, 1990]** M. Abed. Contribution à la modélisation de la tâche par des outils de spécification exploitant les mouvements oculaires : application à la conception et à l'évaluation des interfaces homme-machine. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis, septembre.

**[Abed, 2001]** M. Abed. Méthodes et modèles formels et semi-formels pour la conception et l'évaluation des systèmes homme-machine. Habilitation à diriger des recherches, Université de Valenciennes et du Hainaut-Cambrésis, 02 mai 2001.

**[Adam, 2000]** E. Adam. Modèle d'organisation multi-agent pour l'aide au travail coopératif dans les processus d'entreprise application aux systèmes administratifs complexes. Thèse de l'Université de Valenciennes et du Hainaut-Cambrésis, Janvier 2000.

**[Adina *et al.*, 2002]** F. Adina, D. Kayser, S. Pentiuc, A. El-Fallah Segrounichi. Cours francophone sur les agents intelligents. Accessible à http://turing.cs.pub.ro/auf2/.

**[AFNOR, 2003]** Ergonomie de l'informatique. Aspects logiciels, matériels et environnementaux. N° 9388. AFNOR, Paris.

**[Al-Qaimari, DMcRostie, 1999]** G. Al-Qaimari, D. McRostie. KALDI: A Computer-Aided Usability Engineering Tool for Supporting Testing and Analysis of Human-Computer Interaction. Proceedings CADUI 1999, pp. 337-355.

**[Baccino *et al.*, 2005]** T. Baccino, C. Bellino, T. Colombi. Mesure de l'utilisabilité des interfaces. TIC et Sciences Congnitive. Hermes. 2005. ISBN : 2-7462-1026-6.

**[Balbo, 1994]** S. Balbo. Evaluation ergonomique des interfaces utilisateur : un pas vers l'automatisation. Thèse de doctorat, université Joseph Fourier, Grenoble I, septembre 1994.

**[Barthet, 1988]** M.F. Barthet. Logiciels interactifs et ergonomie, modèles et méthodes de conception. Dunod informatique, Paris, 1988.

**[Bass *et al.*, 1991]** L. Bass, R. Little, R. Pellegrino, S. Reed. The Arch Model : Seeheim revisited. Proceedings of User Interface Developpers'Workshop, Seeheim, 1991.

**[Bastien et Scapin, 1993]** J.M.C. Bastien, et Scapin, D.L. (1993). Critères ergonomiques pour l'évaluation d'interfaces utilisateurs. Rapport technique INRIA n° 156, Juin 1993, INRIA : Le Chesnay.

**[Bastien *et al.*, 1998]** J.M.C Bastien, Leulier, M. et Scapin, D.L. (1998). L'ergonomie des sites Web. In Créer et maintenir un service Web, cours INRIA (28 Sept-2 Oct 1998), Pau (France), ADBS Editions : Paris.

**[Bastien et Scapin, 2001** J.M.C Bastien, L. Scapin. Evaluation des systèmes d'information et critères ergonomiques. In Kolski C. (Ed.), Environnement évolués et évaluation de l'IHM. Interaction Homme Machine pour les SI, Volume 2, pp. 53-80. Paris : Hermès, 2001.

**[Bastien et Scapin, 2002]** J.M.C. Bastien, D.L. Scapin. Les méthodes ergonomiques : de l'analyse à la conception et à l'évaluation. In : Ergonomie et Informatique Avancée, Ergo-IA'2002, A. Drouin, J. Robert (Eds), I.D.L.S., Biarritz, 8-10 octobre 2002.

**[Baudel, 1995]** T. Baudel. Aspects morphologiques de l'interaction humain-ordinateur: Etude de modèles d'interaction gestuels. Thèse de l'université de Paris Sud. Décembre 1995.

**[Bayssié et Chaudon, 2002]** L. Bayssié, L. Chaudron. Evaluation de la performance d'un opérateur en fonction de sa tâche. Application aux IHM. In Actes du congrès IHM 2002, Poitiers, 26-29 Novembre 2002.

**[Beirekdar, 2004]** A. Beirekdar. Methodology for automating web usability and accessibility evaluation by guideline. Thèse de doctorat, UCL, Louvain-la-Neuve.

**[Bellik, 1995]** Y. Bellik. Interfaces Multimodales : Concepts, Modèles et Architectures. Thèse de Doctorat, Université Paris XI, Orsay, 1995.

**[Benaïssa, 1993]** M. L. Benaïssa. Une démarche de conception, réalisation et évaluation d'IHM : application au projet ferroviaire ASTREE. Mémoire de Doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes, janvier.

**[Berger, 1992]** T. Berger. Contribution à l'étude de l'activité cognitive de l'operateur et à l'évaluation de sa charge de travail. Thèse de doctorat soutenu à l'université de valenciennes, mars, 1992.

**[Blackmon *et al.*, 2002]** M. H. Blackmon, P.G. Polson, K. Muneo, C. Lewis, 2002. Cognitive Walkthrough for the Web, CHI 2002, vol.4 No.1, pp. 463-470.

**[Boehm, 1988]** B.W. Boehm. A spiral model of software development and enhancement. IEEE Computer, 21(5), pp. 61-72.

**[Bouamrane, *et al.*, 2005]** K. Bouamrane, T. Bonte, M. Sevaux, C. Tahon. SART : un système d'aide à la décision pour la régulation d'un réseau de transport bimodal. Proceedings of the workshop Méthodologies et Heuristiques pour l'Optimisation des Systèmes Industriels, MHOSI 2005, Hammamet, Tunisie, avril.

**[Bradshaw, 1997]** J.M. Bradshaw. Software agent. MIT Press, Cambridge, USA, ISBN 0-262-52234-9.

**[Calvary *et al.*, 1996]** G. Calvary, J. Coutaz, L. Nigay, D. Salber. PAC+3, un modèle d'architecture générique pour systèmes multi-utilisateurs. In Proc. IHM'96, 8èmes Journées de l'Ingénierie de l'Interaction Homme-Machine, Grenoble, Septembre 1996, pp. 125-130.

**[Calvary *et al.*, 1997]** G. Calvary, J. Coutaz, L. Nigay. From Single-User Architectural Design to PAC*: a Generic Software Architecture Model for CSCW. Proceedings of CHI 97, ACM publ. pp. 242-249.

**[Calvary, 1998]** G. Calvary. Proactivité et réactivité : de l'assignation à la complémentarité en conception et évaluation d'interfaces homme-machine. Thèse de doctorat informatique préparée au laboratoire de communication langagière et interaction personne-système (IMAG), Université Joseph Fourier, Octobre, 1998.

**[Calvary *et al.*, 2005]** G. Calvary, O. Daassi, J. Coutaz, A. Demeure. Des widgets aux comets pour la plasticité des systèmes interactifs. Revue d'interaction Homme-Machine (RIHM), Vol 6, N° 1, pp. 33-53.

**[Card *et al.*, 1983]** S.K. Card, T.P. Moran, A. Newell. The psychology of Human Computer Interaction. Lawrence Erlbaum Associates, 1983.

**[Chaib-Draa et Levesque, 1996]** B. Chaib-Draa, P. Levesque. Hierarchical Model and Communication by Signs, Signals and symbols in Multiagent Environments. Modeling Autonomous Agents in a Multi-Agent World, MAAMAW'94, 1994. Appeared also in Distributed Software Agents and Applications, Perram J. W. and Müller (eds.), Lecture Notes in AI 1069, Springer Verlag, Berlin, 1996.

**[Chaib-draa *et al.*, 2001]** B. Chaib-Draa, B. Moulin, I. Jarras. Agent et Systèmes Multiagents. In Principes et architecture des systèmes multi-agents. JP. Briot et Y Demazeau (eds). Hermes, Lavoisier, 2001.

**[Chalon, *et al.*, 2001]** R. Chalon, B.T. David, M. Beldame, N. Cherief, J. Lasalle, J. Moinard. L'oculomètre comme support d'évaluation et d'interaction. 13ème Conférence Francophone sur l'Interaction Homme-Machine. Editions Cépaduès-Éditions, Toulouse, France, éditeur(s): Vanderdonckt, J., Blandford, A., & Derycke, A. (Eds.), vol. 2, Lille, France, 10-14 sept. 2001, pp. 117-122.

**[Chalon, 2004]** R. Chalon. Réalité Mixte et Travail Collaboratif : IRVO, un modèle de l'Interaction Homme–Machine. Thèse de l'école centrale de Lyon, Décembre 2004.

**[Chihaib *et al.*, 2001]** F. Chihaib, S. Hammadi, P. Borne. Régulation du trafic inter-stations. Final report, GRRT Cooperative Project, Lille, France, April 2001.

**[Cooper et Harper, 1969]** G. Cooper, B. Harper. The use of pilot rating in the evaluation of aircraft handling qualities. Moffett Field, California : NASA Ames, Report TN-D5153, Avril, 1969.

**[Coutaz, 1987]** J. Coutaz. PAC, an Object-Oriented Model for Dialog Design. In Bullinger, Hans-Jorg, Shackel, Brian (eds.): INTERACT 87 - 2nd IFIP International Conference on Human-Computer Interaction. September 1-4, 1987, Stuttgart, Germany, pp.431-436.

**[Coutaz, 1990]** J. Coutaz. Interface homme-ordinateur, conception et réalisation. Dunod, Paris.

**[Coutaz et Nigay, 2001]** J. Coutaz, L. Nigay. Architecture logicielle conceptuelle des systèmes interactifs. Dans [Kolski, 2001], pp. 208-246.

**[Djenidi *et al.*, 2002]** H. Djenidi, A. Ramdane-Cherif, C. Tadj, N. Levy. *Architectures multi-agents génériques à base de réseaux de Pétri temporisés colorés pour la fusion Multimodale en entrée*. IHM2002, 14ème Conférence Francophone sur l'Interaction Homme-Machine. Poitiers, France, novembre 26-29, 2002.

**[Dance *et al.*, 1987]** J.R. Dance, T.E. Granor, R.D. Hill, S.E. Hudson, J. Meadas, B.A. Myers, A. Sdhulert. The run-time structure of UIMS-Supported application. Computer graphics, Volume 21, N° 2, avril 1987.

**[Dassi *et al.*, 2003]** O. Daassi, G. Calvary, J. Coutaz, A. Demeure. Comet : Une nouvelle génération de widget pour la plasticité des interfaces. Actes du congrès IHM03, ACM Press, pp. 64-71.

**[David et Alla, 1992]** R. David, H. Alla. Du Grafcet aux réseaux de Petri. Editions Hermès, Paris, 1992.

**[De Haan, 2001]** G. de Haan. Accomodating Diverse Users in ETAG and ETAG-Based Design: task knowledge and presentation. In Proceedings of HCI International 2001 / 1st International Conference on Universal Access in HCI, 5-10 August 2001, New Orleans, USA.

**[Denley et Long, 1997]** I. Denley, J. Long. A planning aid for human factors evaluation practice. Behaviour & Information Technology, vol.16, n°415, pp. 203-219.

**[Depaulis, 2002]** F. Depaulis. Vers un environnement générique d'aide au développement d'applications interactives de simulations de métamorphoses. Thèse de l'université de Potiers, novembre 2002.

**[Dewan, 1999]** P. Dewan. Architectures for Collaborative Applications. In Computer-Supported Cooperative Work, M. Beaudouin-Lafon (ed.), Trends in Software, pages 169-194, John Wiley et Sons.

**[Diaz, 2001]** M. Diaz. Les Réseaux de Petri - Modèles Fondamentaux. Éditions Hermes. ISBN 2-7462-0250-6, 2001.

**[Dix *et al.*, 1998]** A. Dix, J. Finlay, G. Abowd, R. Beale. Human-Computer Interaction. London, Prentice Hall International, 1998.

**[Doniec et al, 2006]** A. Doniec, R. Mandiau, S. Piechowiak, S. Espié. L'anticipation comme modèle d'interaction : application à la coordination multi-agent en simulation. In Actes de /RFIA 2006 15ème congrès francophone Reconnaissance des formes et Intelligence Artificielle (25-27 janvier 2006, Tours), Presses Universitaires François-Rabelais, Tours, janvier, ISBN 2-86906-216-8**.**

**[Dragicevic, 2004]** P. Dragicevic. Un modèle d'interaction en entrée pour des systèmes interactifs multi-dispositifs hautement configurables. Thèse de l'université de Nantes, Mars 2004.

**[Duval, 1994]** T. Duval. Modèles d'architecture pour les applications graphiques interactives : la famille Seeheim. Revue Internationale de CFAO et d'Infographie, vol.9, n.4, pp. 529-555, 1994.

**[Duval et Nigay, 1999]** T. Duval, L. Nigay. Implémentation d'une application de simulation selon le modèle PAC-Amodeus. Actes du congrès IHM'99, Montpellier, Cepadues Publ. 22-26 nov. 1999. pp. 86-93.

**[Eason, 1984]** K. D. Eason. Towards the Experimental Study of Usability. Behaviour and Information Technology, 3, pp. 133-143, 1984.

**[Ellis, 1994]** C. Ellis. Keepers, Synchronizers, Communicators and Agents. Workshop on Software architectures for cooperative systems at CSCW'94, ACM Conference on Computer-Supported Cooperative Work, Chapel Hill, North Carolina, USA, 1994.

**[Enembreck et Barthès, 2005]** F. Enembreck, J-P. A. Barthès. ELA - A new Approach for Learning Agents. Journal of Autonomous Agents and Multi-Agent Systems, 3(10), pp. 215-248, 2005.

**[Ezzedine et Abed, 1997]** H. Ezzedine, M. Abed. Une méthode d'évaluation d'Interface Homme-Machine de supervision d'un procédé industriel. Journal Européen des Systèmes Automatisés (RAIRO-APII-JESA), 1997, 7, pp. 1078-1110.

**[Ezzedine, 2002]** H. Ezzedine. Méthodes et modèles de spécification et d'évaluation des interfaces homme-machine dans les systèmes industriels complexes. Habilitation à Diriger des Recherches, Université de Valenciennes et du Hainaut-Cambrésis, décembre 2002.

**[Ezzedine, *et al.*, 2003]** H. Ezzedine, A. Trabelsi, C. Kolski. Modelling of Agent oriented Interaction using Petri Nets, application to HMI design for transport system supervision. In P. Borne, E. Craye, N. Dangourmeau (Eds.), CESA2003 IMACS Multiconference Computational Engineering in Systems Applications (Lille, France, July 9-11, 2003), Ecole Centrale Lille, Villeneuve D'Ascq, pp. 1-8.

**[Ezzedine et Trabelsi, 2005]** H. Ezzedine, A. Trabelsi. From the design to the evaluation of an agent-based human-machine interface. Application to supervision for urban transport system. In P. Borne, M. Benrejeb, N. Dangoumeau, L. Lorimier (Ed.), IMACS World Congress "Scientific Computation, Applied Mathematics and Simulation" (July 11-15, Paris), ECL, pp. 717-725, juillet.

**[Ezzedine *et al.*, 2006]** H. Ezzedine, A. Trabelsi, C. Kolski. Modelling of an interactive system with an agent-based architecture using Petri nets, application of the method to the supervision of a transport system. Mathematics and Computers in Simulation, 70, pp. 358-376, 2006.

**[Farenc, 1997]** C. Farenc. ERGOVAL: une méthode de structuration des règles ergonomiques permettant l'évaluation automatique d'interfaces graphiques. Thèse de Doctorat de l'Université Toulouse 1, janvier 1997.

**[Fayech, 2003]** B. Fayech. Régulation des réseaux de transport multimodal : Systèmes multi agents et algorithmes évolutionnistes. Thèse de doctorat. Université des Sciences et Technologies de Lille. Octobre 2003.

**[Fekete, 1996]** J.D. Fekete. Un modèle multicouche pour la construction d'applications graphiques interactives. Thèse de l'université Paris XI Orsay, Janvier 1996.

**[Fekete, 2005]** J.D. Fekete. Nouvelle génération d'Interfaces Homme-Machine pour mieux agir et mieux comprendre. Habilitation à diriger les Recherches, Université Paris-Sud, mai 2005.

**[Ferber, 1991]** J. Ferber. Conception et programmation par objets. Hermes, 1991.

**[Ferber, 1995]** J. Ferber. Les systèmes multi-agents. InterEditions, 1995.

**[Florez-Mendez, 1999]** R. Florez-Mendez. Towards the Standardization of Multi-Agent System Architectures: An Overview. In ACM Crossroads, Special Issue on Intelligent Agents, Association for Computer Machinery, Issue 5.4, pp. 18-24.

**[Foley et Van Dam 1982]** J.D Foley, A. van Dam. Fundamentals of Interactive Computer Graphics, Addison-Wesley (IBM Systems Programming Series), Reading, MA, 664 pp., 1982.

**[Foley, 1984]** J.D. Foley. Managing the Design of User- Computer Interfaces, Proceedings of the Fifth Annual NCGA Conference and Exposition. Anaheim, CA. Vol. II. May 13-17, 1984. pp. 436-451.

**[Franklin et Graesser, 1996]** S. Franklin, A. Graesser. Is it an agent or just a program ? : a taxonomy for autonomous agents. In Intelligent Agents III: Agents Theories, Architectures and Languages, ECAI'96 Workshop (ATAL), Springer-Verlag, 1996.

**[Garzotto et Matesa, 1997]** F. Garzotto, M. Matesa. A systematic method for hypermedia usability inspection. The new review of hypermedia, Vol. 3, pp. 39-65, 1997.

**[Girard et Ribette, 2001]** D. Girard, J.N. Ribette. OpenSourceJava-Troisième partie : MVC2 avec Struts. Avril 2001. accessible à l'URL : http://www.application-servers.com/articles/pdf/opensourcejava-struts.pdf

**[Goldberg, 1983]** A. Goldberg. Smalltalk-80, the interactive programming environnement. Addision-Wesley, 1983.

**[Grammenos, *et al.*, 2000]** D. Grammenos, D. Akoumianakis, C. Stephanidis. Integrated Support for Working with Guidelines : the Sherlock Management System. Interacting with Computers, 12 (3), pp. 281-311, 2000.

**[Grislin, 1995]** M. Grislin. Définition d'un cadre pour l'évaluation a priori des interfaces homme-machine dans les systèmes industriels de supervision. Mémoire de Doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes, janvier 1995.

**[Grislin et Kolski, 1996]** M. Grislin, C. Kolski. Evaluation des interfaces homme-machine lors du développement de système interactif. Technique et Science Informatiques (TSI), 3, pp. 265-296.

**[Guittet et Pierra, 1993]** L. Guittet, G. Pierra. Conception modulaire d'une application graphique interactive de conception technique : la notion d'interacteur. In Proceedings of Interfaces Homme-Machine, Lyon, 1993, pp. 151-156.

**[Guittet, 1995]** L. Guittet. Contribution à l'Ingénierie des IHM - Théorie des Interacteurs et Architecture H4 dans le système NODAOO. Thèse de l'Université de Poitiers, 1995.

**[Guzdial, 1993]** M. Guzdial. Deriving software usage patterns from log files. Tech Report GIT-GVU-93-41. 1993.

**[Hanon *et al.*, 2006]** D. Hanon, E. Grislin-Le Strugeon, R. Mandiau. Un modèle décisionnel orienté comportement utilisant le vote, application à la navigation autonome en environnement simulé. In RFIA 2006 15ème congrès francophone Reconnaissance des formes et Intelligence Artificielle (25-27 janvier 2006, Tours), Presses Universitaires François-Rabelais, Tours, janvier, ISBN 2-86906-216-8.

**[Hammond *et al.*, 1984]** N. Hammond, G.E. Hinton, P. Barnard, J. Long, A. Whitefield. Evaluating the interface of a document processor: A comparison of expert judgement and user observation. In Human-Computer Interaction, INTERACT'84, B.Schackel (Ed.), Elsevier Science Publishers B.V.: North-Holland, IFIP, pp. 725-729, 1984.

**[Hammontree 1992]** M.L. Hammontree, J.J. Hendrickson, B.W. Hensley. Integrated data capture and analysis tools for research and testing on graphical user interfaces. In Proceedings of the CHI'92 Conference, pp. 431-432, ACM Press, Monterey, 3-7 May 1992.

**[Hayat, 2001]** S. Hayat. Amélioration de la qualité des correspondances da les réseaux de transports urbains. Rapport final du projet SART, 2001.

**[Helander, 1988]** M.Helander (ed). Handbook of Human-Computer Interaction. Elsevier Science Publisher B.V. (North-Holland), 1988.

**[Helander *et al.*, 1997]** M. G. Helander, T. K. Landauer, P. V. Prabhu. Handbook of human-computer interaction. 2nd ed., Elsevier Science, Amsterdam, The Netherlands, 1997.

**[Henninger, 2000]** S. Henninger. A Methodology and Tools for Applying Context-Specific Usability Guidelines. Interacting with Computers, 12 (3), pp. 225–243, 2000.

**[Hilbert et Redmiles, 2000]** D. M. Hilbert, D.F. Redmiles. Extracting usability information from user interface events. ACM Computing Surveys (CSUR), 32 (4), pp. 384-421, Dec. 2000

**[Holyer, 1993]** A. Holyer. Methods for evaluating User Interfaces. School of Cognitive and Computing Sciences. University of Sussex, Brighton, 1993.

**[Hudson, 1987]** S.E. Hudson. UIMS Support for Direct Manipulation Interfaces. Computer Graphics, Avril 1987, Vol. 21, N 2, pp. 120-124.

**[Hulzebosch et Jameson, 1996]** R. Hulzebosch, A. Jameson. FACE: A Rapid Method for Evaluation of User Interfaces. In P. W. Jordan, B. Thomas, B. A. Weerdmeester, & I. L. McClelland (Eds.), Usability evaluation in industry, pp. 195-204, Taylor & Francis, London.

**[IGL, 1989]** IGL Technology. SADT, un langage pour communiquer. Editions Eyrolles, Paris, 1989.

**[Iglesias *et al.*, 1999]** C. A. Iglesias, M. Garijo, J. C. Gonzalez. A survey of agent-oriented methodologies. In J. P. Müller, M. P. Singh, A. S. Rao (Eds.), Intelligent Agents Proceedings of the Fifth International Workshop on Agent Theories, Architectures, and Languages (ATAL-98). Springer-Verlag, Heidelberg, 1999.

**[ISO, 1996]** International Standards Organisation. ISO/DIS 9241-10. Ergonomic requirements for office work with visual display terminals. Dialogue principles, 1996.

**[Ivory et Hearst, 2001]** M. Ivory, M. Hearst. The State of the Art in Automated Usability Evaluation of User Interfaces. ACM Computing Surveys, 33 (4), 2001, pp. 173-197.

**[Jacko et Sears, 2003]** J. Jacko, A. Sears. The Human-Computer Interaction Handbook. Lawrence Erlbaum, 2003.

**[Jacobsen et John, 2000]** N. E. Jacobsen, B.E. John. Two Case Studies in Using Cognitive Walkthrough for Interface Evaluation. Carnegie Mellon University, Report No. CMU-CHII-00-100m May 2000.

**[Jaulent, 1994]** P. Jaulent. Génie Logiciel, les méthodes. A. Colin, Paris, 1994.

**[Jennings *et al.*, 1998]** N.R. Jennings, K. Sycara, M. Wooldridge. A Roadmap of Agent Research and Development. Autonomous Agents and Multi-Agent Systems, vol 1, N° 1, pp. 7-38, 1998.

**[Jiang *et al.*, 1992]** J. Jiang, E.D Murphy., S.C. Bailin, W.F. Truszkowski, M. Szczur. Prototyping a knowledge based compliance checker for User-Interface. Evaluation on Motif development environments: Second Annual International Motif Users Meeting, Bethesda, MD: Open Systems, pp. 258-268, 1992.

**[John et Kieras, 1996]** B.E. John, D.E. Kieras. Using GOMS for user interface design and evaluation: which technique? ACM Transactions on Computer-Human Interaction, vol. 3, pp. 287-319, 1996.

**[Karat, 1988]** J. Karat . Software Evaluation Methodologies. In Handbook of Human-Interaction M.Helander (ed), pp. 891-903. Elsevier Science Publisher B.V. (North-Holland), 1988.

**[Karsenty et Weikart, 1991]** S. Karsenty, C. Weikart. Une extension de l'interface d'application dans le modèle de Seeheim. Actes du congrès IHM'1991, Dourdan, Décembre 1991.

**[Keivunen et Mantyla, 1988]** M.R. Keivunen, M. Mantyla. An improved architecture for user interface management system. IEEE Computer and Application. Janvier 1988, pp. 43-52.

**[Keyser *et al.*, 1987]** V. Keyser, F. Decortis, A. Housiaux, A. Van Daele. Les communications homme-machine dans les systèmes complexes. Rapport politique scientifique FAST n°8, Université de Liège, 1987.

**[Klusch, 2001]** M. Klusch. Information agent technology for the internet: a survey. Data and Knowledge Engineering, 36 (3), 2001, pp. 337-372.

**[Kolski, 1989]** C. Kolski. Contribution à l'ergonomie de conception des interfaces graphiques homme-machine dans les procédés industriels : application au système expert SYNOP. Thèse de Doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Janvier 1989.

**[Kolski, 1997]** C. Kolski. Interfaces Homme-Machine : Application aux systèmes industriels complexes. Hermès, Paris, ISBN 2-86601-578-9.

**[Kolski, et *al.*, 2000]** C. Kolski, B. Riera, T. Berger. A questionnaire-based discount evaluation mathod using guidelines for process control interactive applications. In J. Vanderdonckt, C. Farenc (Ed.), Tools for Working With Guidelines TFWWG'2000, Springer, London, pp. 127-138, janvier, 2000.

**[Kolski, 2001]** C. Kolski. Analyse et conception de l'IHM, Interaction Homme-Machine pour les Systèmes d'Information, vol. 1. Hermès, Paris, ISBN 2-7462-0239-5.

**[Kovács *et al.*, 2004]** B. Kovács, F. Gaunet, X. Briffault. Les techniques d'analyse de l'activité pour l'IHM. Hermes, 2004, ISBN 2-7462-0944-6.

**[Lachenal et Coutaz, 2003]** C. Lachenal, J. Coutaz. IntrosPAC : un outil pour enseigner et comprendre PAC-Amodeus. Actes du congrès IHM 2003, ACM Press, Novembre 2003, pp. 212-215.

**[Laichour et Maouche. 2001]** K. Laichour, S. Maouche. Régulation du trafic dans les nœuds de correspondances. Rapport final du projet coopératif GRRT, Avril 2001.

**[Laurillau, 2002]** Y. Laurillau. Conception et réalisation logicielles pour les collecticiels centrées sur l'activité de groupe : le modèle et la plate-forme Clover. Thèse de l'université Joseph Fourier. Gronoble septembre 2002.

**[Lewis *et al.*, 1990]** C.H. Lewis, P.G. Polson, C. Wharton, J. Rieman. Testing a Walkthrough Methodology for Theory-Based Design of Walk-Up-and-Use Interfaces. In: J. Carrasco, J. Whiteside (eds.), Proceedings of the ACM CHI 90 Human Factors in Computing Systems Conference, 1990, Seattle, Washington, USA. pp.235-242.

**[Liberati *et al.*, 1995]** V. Liberati, C. Farenc, M-F. Barthet. Modélisation des connaissances d'Ergoval, un outil d'évaluation ergonomique des interfaces. IHM'95, septièmes journées sur l'Ingénierie des Interfaces Homme-Machine, Toulouse, 11-13 Octobre 1995.

**[Loslever *et al.*, 2003]** P. Loslever, J-C. Popieul, P. Simon. Analyse de données provenant d'unités temporelles relatives au comportement d'un système complexe. Exemple des mouvements oculaires dans le système conducteur-véhicule. APII-JESA, 37, 6, pp. 799-824, 2003.

**[Lowgren et Nordqvist, 1992]** J. Lowgren, T. Nordqvist. Knowledge-based evaluation as design support for graphical user interfaces. Proceeding of ACM conference on Human Factors in Computing Systems, CHI'92, Bauersfeld, (Eds.), Monterey, California, pp.181-188. May 3-7, 1992.

**[Luck et d'Inverno, 2001]** M. Luck, M. d'Inverno. A Conceptual Framework for Agent Definition and Development. The Computer Journal, vol 44, n° 1, pp. 1-20.

**[Macleod et Rengger, 1993]** M. Macleod, R. Rengger. The Development of DRUM: A Software Tool for Video-assisted Usability Evaluation. In: People and Computers VIII Proc. of HCI'93 Conf., Loughborough UK, September 1993, Cambridge, pp. 293-309.

**[Mahajan et Shneiderman, 1995]** R. Mahajan, B. Shneiderman. A family of user interface consistency checking tools. Report CS-TR-3472, College Park: University of Maryland.

**[Mahfoudhi, 1997]** A. Mahfoudhi. TOOD : Une méthodologie de description orientée objet des tâches utilisateur pour la spécification et la conception des interfaces homme-machine. Thèse de Doctorat,de l'Université de Valenciennes et du Hainaut-Cambrésis, Juin 1997.

**[Mandiau, 2000]** R. Mandiau. Modélisation et évaluation d'organisations multi-agents. Habilitation à Diriger des Recherches à l'Université de Valenciennes, Décembre 2000.

**[Mandiau et Grislin-Le Strugeon, 2001]** R. Mandiau, E. Grislin-Le Strugeon. Systèmes multi-agents. In Techniques De L'ingénieur (Ed.), Techniques de l'Ingénieur, Paris, pp. 1-17.

**[Mariage, 2005]** C. Mariage. MetroWeb : logiciel de support à l'évaluation de la qualité ergonomique des sites web. Thèse de Doctorat, UCL, Louvain-la-Neuve 2005.

**[Martin *et al.*, 2005]** S. Martin, S. Degrande, C, Chaillou. Une utilisation du modèle MVC pour une plate-forme de travail virtuel. Actes du congrès IHM 2005, ACM Press, pp. 131-138.

**[Mesghouni *et al.* 2001]** K. Mesghouni, E. Castelain, J.C. Gentina, S. Hammadi. Introduction du critère flux de passagers pour la régulation du trafic en station. Rapport final du projet coopératif GRRT, Avril 2001.

**[Mick *et al.*, 2005]** M. Philippon, A. Mille, G. Caplat. Aide à l'utilisateur : savoir quand intervenir. Dans IHM'05 : 17ème conférence francophone sur l'interaction homme-machine, ACM Press, Toulouse, France, 2005.

**[Millot, 1988]** P. Millot. Supervision des procédés automatisés et ergonomie. Hermes. Paris, 1988.

**[Moha *et al.*, 2005]** N. Moha, Q. Li, A. Gaffar, A. Seffah. Enquête sur les pratiques de tests d'utilisabilité. IHM'05, 17ème Conférence Francophone sur l'Interaction Homme-Machine, pp. 115-122, September 27-30, 2005, Toulouse, France.

**[Moldt et Wienberg, 1997]** M. Moldt, F. Wienberg. Multi-Agent-Systems based on Coloured Petri Nets. In Proceedings of the 18th International Conference on Application and Theory of Petri Nets, Toulouse, 1997.

**[Moran, 1981]** T.P. Moran. The Command Language Grammar: a representation for the user interface of interactive computer systems. International Journal of Man-Machine Studies, n°15, pp. 3-50, 1981.

**[Mullin *et al.*, 2001]** J. Mullin, A.H. Anderson, L. Smallwood, M. Jackson, E. Katsavras. Eye-tracking explorations in multimedia communications. In A. Blanford, J. Vanderdonckt, P. Gray. (Eds.), People and Computer XV- Interaction without Frontiers – Joint Proceedings of HCI 2001 and IHM 2001, pp. 367-382, London : Springer 2001.

**[Neal, 1983]** A.S. Neal, R.M. Simons. Playback : a method for evaluating the usability of software and its documentation. Proceedings of the CHI'83 Conference on Computer Human Interaction, ACM New York, pp. 78-82, December 1983

**[Nendjo Ella *et al.*, 1999]** A. Nendjo Ella, C. Kolski, C. Jacques, P. Yim. Vers un système d'aide à la décision pour l'évaluation des systèmes interactifs. In J. Nanard, P. Girard (Eds.), Onzièmes journées sur l'ingénierie de l'Interaction Homme-Machine IHM'99, Cepadues-edition, Toulouse, pp. 126-133, janvier.

**[Nendjo Ella, 1999]** A. Nendjo Ella. Vers un outil d'aide à la décision en évaluation des systèmes interactifs et la prise en compte conjointe de critères techniques et socio-culturels. Mémoire de Doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes, janvier, 1999.

**[Nielsen et Molich, 1990]** J. Nielsen, R. Molich. Heuristic evaluation of user interfaces. CHI'90 Conf. Proc., Seatle, ACM Press, New York, pp. 349-356, 1990.

**[Nielsen, 1993]** J. Nielsen. Usability Engineering. Academic Press, Boston, MA, 1993.

**[Nielsen et Landauer, 1993]** J. Nielsen, T. K. Landauer. A mathematical model of the finding of usability problems. In S. Ashlund, K. Mullet, A. Henderson, T. White (Eds.), Proceedings of ACM INTERCHI'93 Conference on Human Factor inComputing Systems 24-29 April, pp. 206-213, New York, ACM Press.

**[Nielsen et Mack, 1994]** J. Nielsen. Heuristic evaluation. In J. Nielsen et R.L. Mack (Eds.), Usability ispection methods, New York, John Wiley & Sons, pp. 25-62, 1994.

**[Nigay, 1994]** L. Nigay. Conception et modélisation logicielles des systèmes interactifs : application aux interfaces multimodales. Thèse de l'université Joseph Fourier, Grenoble, janvier 1994.

**[Nigay et Coutaz, 1995]** L. Nigay, J. Coutaz. A Generic Platform for Addressing the Multimodal Challenge. Proc. CHI'95, Denver, ACM Press.

**[Ouadou, 1994]** K. Ouadou. AMF : Un modèle d'architecture multi-agents multi-facettes pour Interfaces Homme-Machine et les outils associés. Thèse de l'Ecole Centrale de Lyon, 1994.

**[Paganelli et Paternò, 2002]** L.Paganelli, F. Paternò. Intelligent Analysis of User Interactions with Web Applications. Proceedings ACM Intelligent User Interfaces, pp. 111-118, ACM Press, San Francisco, January 2002.

**[Palanque et Bastide, 1997]** P. Palanque, R. Bastide. Synergistic modelling of tasks, users and systems using formal specification techniques. Interacting With Computers, 9 (2), 1997, pp. 129-153.

**[Paterno *et al.*, 1997]** F. Paterno, C.Mancini, S.Meniconi. ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. Proceedings Interact'97, pp. 362-369, Sydney, Chapman&Hall.

**[Patry, 1999]** G. Patry. Contribution à la conception du dialogue Homme-Machine dans les applications graphiques interactives de conception technique: le système GIPSE. Doctorat d'Université, Poitiers, 1999.

**[Payne et Green, 1989]** S.J. Payne, T.R.G. Green. Task-Action Grammar: the model and developments. In Task analysis for Human-Computer Interaction, D. Diaper (Ed.), John Wiley and sons, pp. 75-107, 1994.

**[Petri, 1962]** C. A. Petri. Kommunikation mit Automaten. Schriften des IIM 3, 1-128, Universität Bonn, Institut für Instrumentelle Mathematik, Bonn, 1962.

**[Pfaff, 1985]** G.E. Pfaff. User interface management system. Springer-Verlag, 1985.

**[Pivec *et al.*, 2006]** M. Pivec, C. Trummer, J. Pripfl. Eye-Tracking Adaptable e-Learning and Content Authoring Support. Informatica, Vol. 30, N 1, pp. 83-86, 2006.

**[Pollier, 1991]** A. Pollier. Evaluation d'une interface par des ergonomes : diagnostics et stratégies. Rapport de recherche INRIA, n° 1391, février 1991.

**[Polson *et al.*, 1992]** P.G. Polson, C. Lewis, J. Rieman, C. Wharton. Cognitive walkthroughs: A method for theory- based evaluation of user interfaces. International Journal of Man-Machine Studies, 36, pp. 741-773, 1992.

**[Rasmussen, 1983]** J. Rasmussen. Skill, rules and knowledge ; signals, signs, and symbols, and other distinctions in human performance models. IEEE Transactions on Systems, Man and Cybernetics, SMC-13(3), pp. 257–267, 1983.

**[Ravden et Johnson, 1989**] S.J. Radven, G.I. Johnson. Evaluating usability of human-computer interfaces: a practical method. Ellis Horwood, Chicester, 1989.

**[Rehim, 2005]** A. Rehim. Etude des outils exploitant des réseaux de Petri agent pour l'évaluation des systèmes interactifs. Mémoire de Master recherche, LAMIH, Valenciennes, juillet 2005.

**[Reiman *et al.*, 1991]** J. Reiman, S. Davies, C. Hair, M. Esemplare, E. Polson, C. Lewis. An automated cognitive walkthrough. In Robertson, S.E, Olsen, G.M., and Olsen, J.S. (Eds.), Human Factors in Computing Systems, CHI 91 Conference Proceedings, 1991, ACM Press, New York.

**[Reisner, 1984]** P. Reisner. Formal grammars as a tool for analysing ease of use: some fundamental con-cepts. In Thomas and Schneider (Eds.), Human Factors in Computer Systems, Norwood, N.J.: Ablex, 1984.

**[Rey, 2005]** G. Rey. Contexte en Interaction Homme-Machine : le contexteur. Communication Langagière et Interaction Personne-Système. Thèse de l'Université Joseph Fourier de Grenoble, 1er août, 2005.

**[Robert, 2003]** J.M. Robert. Que faut- il savoir sur l'utilisateur pour concevoir des interfaces de qualité ? In G.A. Boy (Ed.), L'ingénierie cognitive : IHM et Cognition, Hermès, Paris, 2003.

**[Root et Draper, 1983]** R.W. Root, S. Draper. Questionnaire as a software evaluation tool. In Human Factors in Computing Systems-I, A. Janda (Ed.), ACM Press, North-Holland, Amsterdam, pp. 83-87, 1983.

**[Rossm 1977]** D.T. Ross, Structured analysis (SA): a language for communicating ideas. IEEE Transactions on Software engineering, 3 (1), Janvier 1977.

**[Rushinek et Rushinek, 1986]** A. Rushinek, S. Rushinek. What makes users happy ? Communication of ACM, 29, pp. 594-598, 1986.

**[Russell et Norvig 1995]** S. Russell, P. Norvig, Artificial Intelligence: a Modern Approach. Prentice Hall Series in Artificial Intelligence, Englewood Cliffs, New Jersey, 1995.

**[Salber, 1995]** D. Salber. De l'interaction homme-machine individuelle aux systèmes multi-utilisateurs. Thèse de l'université Joseph Fourier - Grenoble 1, Septembre 1995.

**[Scapin, 1986]** D.L. Scapin. Guide ergonomique de conception d'interface homme-machine. Rapport technique INRIA, N° 77, le Chesnay, 1986.

**[Scapin et Pierret-Golbreich, 1989]** D.L. Scapin, C. Pierret-Golbreich. MAD : Une méthode analytique des descriptions des tâches. Colloque sur l'ingénierie des IHM, pp. 131-148, Sophia Antipolis, 24-26 Mai, 1989.

**[Shim *et al.*, 2002]** J.P. Shim, J.F. Courtney, D.J. Power, R. Sharda, C. Carlsson. Past, present, and future of decision support technologie. Journal of Decision Support Systems, Vol 33, Issue 2, pp. 111-126, June 2002.

**[Senach, 1990]** B. Senach. Evaluation ergonomique des IHM : Une revue de la littérature. Rapport INRIA, n°1180, mars, 1990.

**[Shepherd, 1993]** A. Shepherd. An approach to information requirements specification for process control tasks. Ergonomics, 36 (11), pp. 1425-1427, 1993.

**[Shoham, 1993]** Y. Shoham. Agent-oriented programming. Artificial Intelligence, 60(1), pp. 51–92, 1993.

**[Sibertin-Blanc, 1985]** C. Sibertin-Blanc. High-level Petri nets with Data Structure. Proceedings 6th EWPNA, June, Espoo, Finland, 1985.

**[Simon, 1993]** P. Simon. Contribution de l'analyse des mouvements oculaires à l'évaluation de la charge de travail mental. Mémoire de Doctorat, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes, janvier 1993.

**[Simonin, 2001]** O. Simonin. Le modèle satisfaction-altruisme : coopération et résolution de conflits entre agents situés réactifs, application à la robotique. Thèse de l'Université de Montpellier 2, Décembre 2001.

**[Shoham, 1993].** Y. Shoham. Agent-Oriented Programming. AI, 60, pp. 51-92, 1993.

**[Smith et Mosier, 1986]** S. L. Smith, J. N. Mosier. Guidelines for Designing User Interface Software. The MITRE Coporation, Bedford ESD-TR-86-278 MTR-10090, 1986.

**[Sperandio, 1996]** J.-C. Sperandio. L'apport de la psychologie du travail. In: Traité d'ergonomie, P. Cazamian, F. Hubault (eds.), Octares, Toulouse, France, pp. 165-207, 1996.

**[Stanton, Baber, 1996]** N.A. Stanton, C. Baber. Factors affecting the selection of methods and techniques prior to conducting a usability evaluation. In P.W. Jordan, B. Thomas, B.A. Weerdmeester, I. McClelland (eds.), Usability Evaluation in Industry, London: Taylor and Francis, pp. 39-48, 1996.

**[Tabary, 2001** D. Tabary. Contribution à TOOD, une méthode à base de modèles pour la spécification et la conception des systèmes interactifs. Thèse de l'Université de Valenciennes et du Hainaut-Cambrésis, décembre 2001.

**[Tabary *et al.*, 2003]** D. Tabary, M. Abed, C. Kolski. Contribution to task representation in model-based user interface design: application to new people-organization interactions. In J. Jacko, C. Stephanidis (Ed.), Human-Computer Interaction: Theory and Practice (Part I), Lawrence Erlbaum Associates, Mahwah, New Jersey, pp. 1258-1262, 2003.

**[Tarby, 1993]** J.C. Tarby. Gestion automatique du dialogue homme-machine à partir de spécifications conceptuelles. Thèse de l'Université Paul Sabatier, Toulouse I, 1993.

**[Tarby Jean-Claude, 2006]** J.C. Tarby. Evaluation précoce et conception orientée évaluation. Actes de la 10ème conférence internationale Ergo'IA, Bidart/Biarritz, France, 11-13 Octobre 2006, ISBN: 2-9514772-6-0.

**[Tarpin-Bernard, 1997]** F. Tarpin-Bernard. Travail coopératif synchrone assisté par ordinateur : Approche AMF-C. Thèse de l'Ecole Centrale de Lyon, juillet 1997.

**[Tarpin-Bernard et David, 1999]** F. Tarpin-Bernard, B. David. AMF : un modèle d'architecture multi-agents multi-facettes. Techniques et Sciences Informatiques, Vol. 18, No. 5, pp. 555-586, 1999.

**[Tauber, 1990]** M.J. Tauber. ETAG Extended Task Action Grammar – A language for the description of the user's task language. In D. Diaper ***et al.*** (Eds.), Human-Computer Interaction, INTERACT'90, pp. 163-168, 1990.

**[Texier, 2000]** G. Texier. Contribution à l'ingéniérie des systèmes interactifs : Un environnement de conception graphique d'applications spécialisées de conception. Thèse de l'Université de Poitiers, novembre 2000.

**[Trabelsi *et al.*, 2004 a]** A. Trabelsi, H. Ezzedine, C. Kolski. Architecture modeling and evaluation of agent-based interactive systems. IEEE-SMC2004, International Conference on Systems, Man and Cybernetics, The Hague, the Netherlands, October 10-13, 2004.

**[Trabelsi *et al.*, 2004 b]** A. Trabelsi, H. Ezzedine, C. Kolski. Principes et outil d'évaluation de systèmes interactifs orientés agents. CIFA2004, Conférence internationale francophone d'automatique, Douz, Tunisie, 22-24 Novembre 2004.

**[Trabelsi *et al.*, 2005]** A. Trabelsi, H. Ezzedine, C. Kolski. Système d'aide à l'information des voyageurs en station et dans les véhicules. Rapport 1 (janvier 2003) et rapport final (juillet 2005), projet TACT SART, LAMIH, Valenciennes, juillet.

**[Trabelsi *et al.*, 2006]** A. Trabelsi, H. Ezzedine, C. Kolski. Un mouchard électronique orienté agent pour l'évaluation de systèmes interactifs de supervision. CIFA2006, Conférence Internationale Francophone d'Automatique, Bordeaux, France, 30-31 Mai et 1 juin 2006.

**[Trabelsi, et Ezzedine, 2006]** A. Trabelsi, H. Ezzedine. Un pas vers un outil d'aide aux évaluateurs de systèmes interactifs à base d'agents. ERGOIA2006, Colloque scientifique sur l'ergonomie et l'informatique avancée, Biarritz, France, 11-13 Octobre 2006.

**[UIMS 1992]** UIMS. The UIMS Workshop Tool Developers. A metamodel for the runtime architecture of an interactive system. SIGCHI Bulletin, 1992, vol 24, n°1, pp. 32–37.

**[Vanderdonckt et Bodart, 1994]** J. Vanderdonckt, F. Bodart, Jusqu'au bout avec nos règles ergonomiques. In Actes des Sixièmes Journées sur l'Ingénierie des Interfaces Homme-machine IHM'94, Lille, 8-9 December 1994, pp. 231-236.

**[Vanderdonckt, 1994]** J. Vanderdonckt. Guide ergonomique des interfaces homme-machine. Presse universitaire de Namur, Belgique.

**[Vanderdonckt, 1998]** J. Vanderdonckt. Conception ergonomique de pages WEB, Vesale, 1998.

**[Van-Eylen et Hiraclides, 1996]** H. Van Eylen, G. Hiraclides. GRAAL en quête d'une démarche de développement d'interface utilisateur. Édition Angkor, 1996.

**[Virzi, 1997]** R.A. Virzi. Usability inspection methods. In M. Helander, T.K. Landauer, P. Prabhu (Eds.) Handbook of human-computer interaction, 2nd ed. Elsevier Science. pp. 705-715, 1997.

**[Virzi, 1990]** R. A. Virzi. Streamling the design process : Running fewer subjects. Proceedings of the Human Factors Society 34th Annual Meeting, Santa Monica, CA, October 8-12, pp. 291-294, Orlando, FL, 1990.

**[Whitefield, 1991]** A. Whitefield, F. Wilson, J. Dowell. A framework for human factors evaluation. Behaviour & Information Technology, Vol 10, n°1, pp. 65-79, 1991.

**[Wierwille et Casali, 1983]** W.W. Wierwille, J.G. Casali. Evaluation of 20 workload measures using a psychomotor task in a moving-base aircraft simulator. Human factors, 25(1), pp. 1-16, 1983.

**[Wooldridge et Jennings, 1995]** M. Wooldridge, N.R. Jennings. Intelligent agents: Theory and practice. Knowledge Engineering Review, Vol. 10, N° 2, 1995, pp.115-152.

**[Wooldridge et Jennings 2000]** V. Wooldridge, N.R. Jennings, D. Kinny. The Gaia Methodology for Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems, 3 (3), pp. 285-312, 2000.

**[Woods, 1970]** W. Woods. Transition Network Grammars for Natural Language Analysis. Communications of the ACM, vol. 13, n° 10, 1970, pp. 591-606.

**[Xu et Shatz, 2001]** H. Xu, S.M. Shatz. An Agent-Based Petri Net Model with Application to Seller/Buyer Design in Electronic Commerce. ISADS 2001: 11-18dp.

*Annexes 1*

# *Outils de modélisation exploités dans la thèse : les réseaux de Petri et les actigrammes de SADT*

---

1. *Rappels sur les réseaux de Petri*
2. *Rappels sur les actigrammes de la méthode SADT*

# I. Rappel sur les réseaux de Petri

Les réseaux de Petri (RdP) ont été développés au début des années 1960 par le mathématicien allemand C. A. Petri [Petri, 1962]. Ce sont des outils graphiques et mathématiques utilisés pour formaliser, décrire et analyser le comportement des systèmes dynamiques où les notions d'événement et de concurrence sont prépondérantes. En effet ils sont adaptés plus particulièrement à la modélisation des systèmes discrets à évolution parallèle. Les réseaux de Petri ordinaires sont constitués de deux types de noeud (Cf. Figure A1.1) :

- **les places** : une place est représentée par un cercle et représente les états possibles du système ;

- **les transitions** : une transition est représentée par un trait ; il s'agit ici de modéliser les opérateurs de changement d'état.

Ces deux types de nœuds sont reliés entre eux avec des arcs orientés. Ainsi un RdP est défini par : un ensemble fini de places, P = {P1, P2, P3, ..., Pm}, représentant les états du système ; un ensemble fini de transitions, T = {T1, T2, T3, ..., Tn}, représentant l'ensemble des événements qui agissent sur le système et un ensemble fini d'arcs orientés qui assurent la liaison d'une place vers une transition ou d'une transition vers une place.

Les réseaux de Petri présentent deux principaux avantages :

- ils reposent sur un formalisme mathématique : ils se prêtent au calcul, à l'expression et à l'analyse de la synchronisation et du parallélisme ;

- ce sont des modèles graphiques par excellence : ils permettent de présenter graphiquement et de visualiser certaines notions telles que le parallélisme, la synchronisation, le partage de ressource, la mémorisation, la lecture, la capacité limitée…[David et Alla, 1992].

Malgré le fait que les réseaux de Petri soient définis mathématiquement et offrent une représentation graphique claire et visible, ils présentent un manque de structuration ainsi qu'un faible pouvoir d'expression conduisant très vite à des modèles volumineux, inexploitable pour modéliser des systèmes complexes [Diaz, 2001].

Pour pallier ce problème, des travaux de recherche se sont intéressés à la définition même du RdP en introduisant de nouvelles notions telles que les couleurs pour différencier les jetons et compacter le réseau, la temporisation pour la modélisation des systèmes à événements discrets. Ces travaux ont donné naissance à plusieurs variantes des réseaux de Petri, qui présentent chacun leurs spécificités, liées aux domaines auxquels ils s'appliquent. On parle alors de RdP de haut niveau. On peut citer à titre d'exemple :

- **les réseaux à inhibition** : dans un RdP ordinaire une transition est validée lorsque chacune de ses places d'entrée contient au moins un jeton. Cette règle ne permet pas de réaliser un test à zéro ou autrement dit de conditionner le franchissement d'une transition à l'état vide de l'une de ses places d'entrée. Les RdP à Arcs Inhibiteurs, permettent de réaliser ce test à zéro en introduisant des arcs

inhibiteurs. La transformation d'un RdP à arc inhibiteur en RdP ordinaire n'est pas toujours possible.

- **les réseaux généralisés** : dans un RdP généralisé, les arcs orientés comportent une indication de poids qui est un entier supérieur à 1 ; le tir des transitions dans ce type de réseau, modifie le marquage des places en fonction des poids indiqués. Tout RdP généralisé peut être transformé en RdP ordinaire.

- **les réseaux temporisé**s [David et Alla, 1992] : ils apportent la notion de temps ; Un RDPT (réseau de Petri temporel) est un outil puissant pour la modélisation et la simulation des systèmes temps réel ; il associe pour chaque transition « t » un intervalle de tir [tmin, tmax] modélisant les délais pour lesquels «t» doit être franchie une fois sensibilisée. Lorsque plusieurs transitions sont sensibilisées à la fois, le choix de la transition à tirer se fait de façon non déterministe.

- **les réseaux colorés** [Jensen, 1996] : pour un réseau de Petri de base, on ne distingue pas les différents jetons. Dans un réseau de Petri colorié, on associe une valeur à chaque jeton. Pour plusieurs outils associés aux réseaux de Petri colorés, les valeurs des jetons sont typées, et peuvent être testées et/ou manipulées avec un langage fonctionnel.

D'autres variantes des réseaux de Petri existent ; notre ambition ici n'est pas de les énumérer. Il s'agit de s'attarder plus particulièrement sur les réseaux de pétri adaptés à la modélisation des systèmes multi-agents.

L'idée de la modélisation des systèmes multi-agent avec les réseaux de Petri est apparue en 1997 avec un article proposé par Moldt et Wienberg [Moldt et Wienberg, 1997]. Les auteurs se sont basés sur les travaux de [Shoham, 1993].

Shoham a proposé une méthode de « programmation » d'agents qui se base sur la programmation logique pour représenter l'état mental d'un agent sous forme de faits (actions), de croyances et d'engagements envers d'autres agents. Cette méthode se base sur les trois principes suivants :

- les actions d'un agent peuvent être de deux sortes : communicatives (envoyer des messages aux autres agents), privées ou internes (exécuter des sous-routines).

- le comportement de l'agent est commandé par des règles d'engagement (commitment rules) qui déterminent les actions à effectuer par l'agent suivant les messages reçus et l'état mental de l'agent.

- chaque règle d'engagement contient une condition sur les messages, une condition sur l'état mental et une action. Pour savoir si une règle d'engagement est déclenchée ou non, l'agent compare la condition sur les messages avec les messages reçus, il compare la condition sur l'état mental et ses croyances du moment. Si la règle est activée, alors l'agent est engagé à réaliser l'action.

Cette modélisation est semblable à la modélisation de service entre agent que nous avons proposée (Cf. chapitre trois, § III.2). En effet, les actions communicatives représentent les actions visibles et les communications privées représentent les actions non visibles. Cependant, nous avons adopté une modélisation plus simple (sans l'introduction de règles

d'engagement mais plutôt des conditions et des événements) en vue d'effectuer un couplage entre architecture à base d'agent et son évaluation (Cf. chapitre trois).

Parmi les travaux qui exploitent les réseaux de Petri pour la modélisation des systèmes multi-agent, on peut citer les travaux de [Xu et Shatz, 2001] qui proposent un modèle de réseau de Petri agent pour des application en commerce électronique, ou encore ceux de [Djenidi *et al.*, 2002] qui proposent une architectures multi-agent générique à base de réseau de Petri coloré temporisé pour la fusion multimodale en entrée.

## II. Rappel sur les actigrammes de la méthode SADT

La méthode SADT (Structured Analysis and Design Technique) est une méthode créée en 1976 par D.T. Ross et est la propriété de la société Softech [Jaulent, 1994]. Elle suit une approche cartésienne qui décrit un système de manière descendante, selon différents niveaux d'abstraction, modulaire, hiérarchique et structurée, en couvrant essentiellement les phases d'expression des besoins, de spécification et de conception.

La méthode SADT s'appuie sur un modèle spécifique composé de datagrammes décrivant la transformation des données et d'actigrammes décrivant l'enchaînement des activités. Dans la mesure où nous utilisons simplement les actigrammes (Cf. Chapitre III, § II), nous présentons ci-dessous quelques descriptions de ces derniers.

L'actigramme est modélisé par des "boîtes" identifiées chacune par une activité, un événement ou un verbe. Ce modèle est représenté par des boîtes avec quatre types de liens : entrées, sorties, contrôles et mécanismes (Cf. figure A1.1). L'ensemble des diagrammes est ordonné hiérarchiquement, avec au niveau le plus général, le diagramme de contexte. Les actigrammes sont basés sur une analyse orientée sur les flots de données qui permettent de mettre en évidence les activités.

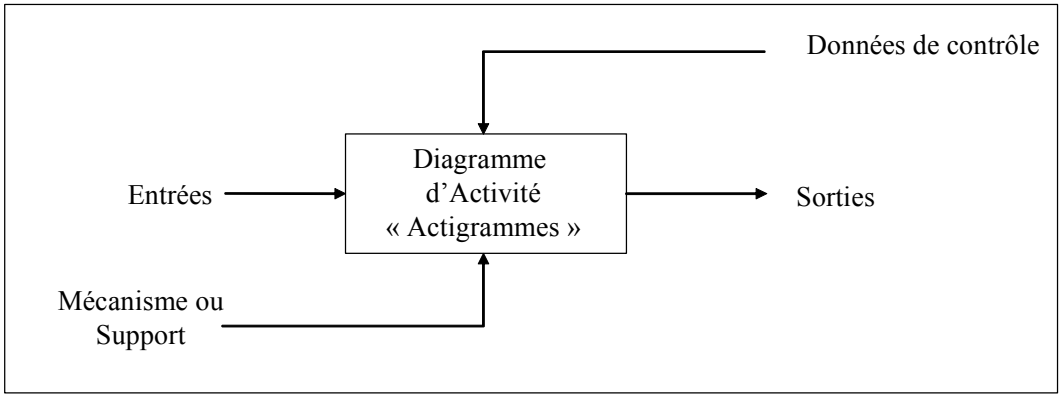



**Figure A1.1.** Les types de liens d'un actigramme dans SADT

Notons que les données de contrôle ne sont pas modifiées par l'activité mais influent sur son déroulement ; les mécanismes, ou supports, de l'activité désignent les outils et les supports nécessaires à la réalisation de l'activité.

Dans un actigramme, les nœuds d'un modèle SADT sont numérotés d'une façon précise. Le premier nœud représente le système global. Il porte le numéro particulier A-0. Il est décomposé sur la feuille A0 en plusieurs noeuds portant les numéros A1, A2, ... An, décomposés à leur tours en A11, A12, etc. (Cf. figure A1.2).

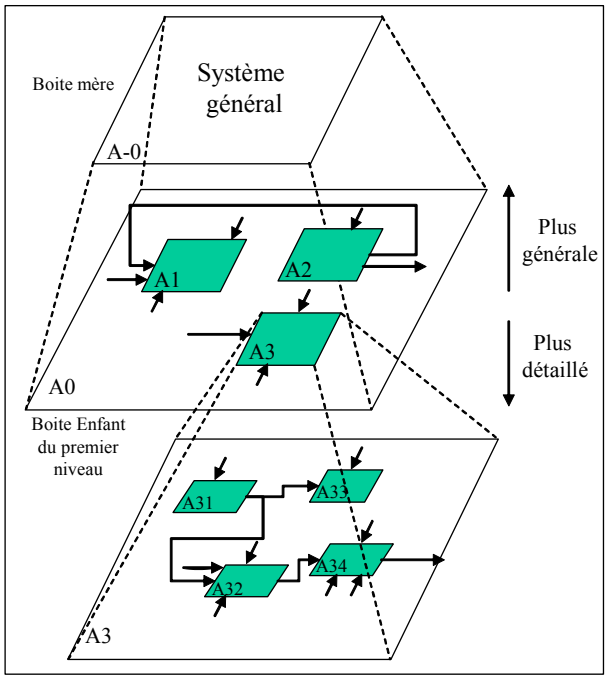



**Figure A1.2.** Principes de décomposition et de numérotation d'actigramme dans SADT

*Annexes 2*

# *Spécification des agents d'interface du SAI*

1. *Spécification de l'agent d'interface Etat_Ligne*
2. *Spécification de l'agent d'interface Station*
3. *Spécification de l'agent d'interface Véhicule*
4. *Spécification de l'agent d'interface Message*
5. *Spécification de l'agent d'interface Vue_Globale*

Cette annexe présente les spécifications des services des *agents d'interface* du SAI. Comme nous l'avons vu au quatrième chapitre, lors de la spécification du SAI nous avons identifié six *agents d'interface* : *Etat_Trafic*, *Etat_Ligne*, *Véhicule*, *Station*, *Message*, *Vue_Globale*. La spécification des services de l'*agent d'interface Etat_Trafic* a été présentée au quatrième chapitre. Nous présentons dans cette annexe la spécification des services des autres *agents d'interface*.

Pour chaque *agent d'interface*, les tableaux fournis synthétisent toutes les informations concernant les services, les évènements, les conditions, les ressources, les actions visibles et les actions invisibles. Pour des raisons de clarté nous donnons quelques explications relatives aux six premiers tableaux de l'*agent d'interface Etat_Ligne*. Les mêmes commentaires seront valables pour les tableaux concernant les autres *agents d'interface*

# I. Spécification de l'agent d'interface Etat_Ligne

## I.1. Les Services

Un *agent d'interface* est défini par l'ensemble de ses services (Cf. chapitre trois, § III.3). Le tableau A2.1 représente les services de l'*agent d'interface Etat_Ligne*. Le service $s_{i,j}$ désigne le service d'indice i de l'*agent d'interface* j. Dans le cas de cet agent, j est égal à 2 et i va de 1 à 14. Ainsi, conformément au modèle proposé (Cf. chapitre trois, § III.3), l'*agent d'interface* Etat_Ligne est défini par l'ensemble de ses services $\{s_{1,2}, s_{2,2}, \ldots, s_{14,2}\}$. Une explication est fournie pour chaque service.

| Identifiant du service | Services | Description |
|---|---|---|
| $s_{1,2}$ | Afficher_Ligne | Affichage de la ligne à superviser |
| $s_{2,2}$ | Masquer_Ligne | Masquage de la ligne supervisée |
| $s_{3,2}$ | Pointer_Véhicule | Pointage avec la souris de la représentation d'un véhicule |
| $s_{4,2}$ | Agrandir_Ligne | Faire un zoom avant sur la Ligne |
| $s_{5,2}$ | Réduire_Ligne | Faire un zoom arrière sur la Ligne |
| $s_{6,2}$ | Pointer_Station | Pointage avec la souris de la représentation d'une station |
| $s_{7,2}$ | Mise_En_Service_Véhicule | Ajout d'un véhicule à la ligne supervisée |
| $s_{8,2}$ | Mise_Hors_Service_Véhicule | Suppression d'un véhicule de la ligne supervisée |
| $s_{9,2}$ | Changer_Retard_Véhicule | Changement du temps de retard d'un véhicule |
| $s_{10,2}$ | Changer_Retard_Seuil_Véhicule | Changement du temps de retard supérieur à un seuil d'un véhicule |
| $s_{11,2}$ | Changer_Avance_Véhicule | Changement du temps d'avance d'un véhicule |
| $s_{12,2}$ | Changer_Avance_Seuil_Véhicule | Changement du temps d'avance supérieur à un seuil d'un véhicule |
| $s_{13,2}$ | Contacter_Station | Clic avec la souris sur la représentation d'une station pour afficher la vue de l'agent Station |
| $s_{14,2}$ | Contacter_Véhicule | Clic avec la souris sur la représentation d'une station pour afficher la vue de l'agent Station |

**Tableau A2.1.** Services de l'*agent d'interface Etat_Ligne*

## I.2. Les Evénements

A chaque service $s_{i,j}$ est associé un évènement $e_{i,j}$. Un événement est responsable du déclenchement du service auquel il est associé. Pour l'*agent d'interface* Etat_Ligne, nous avons identifié l'ensemble des événements $\{e_{1,2}, e_{2,2}, \ldots, e_{14,2}\}$ illustrés dans le tableau A2.2.

| Identifiant de l'évènement | Evènement |
|---|---|
| $e_{1,2}$ | Affichage_Ligne |
| $e_{2,2}$ | Masque_Ligne |
| $e_{3,2}$ | Pointage_Véhicule |
| $e_{4,2}$ | Zoom_Avant |
| $e_{5,2}$ | Zoom_Arrière |
| $e_{6,2}$ | Pointage_Station |
| $e_{7,2}$ | Ajout_Véhicule |
| $e_{8,2}$ | Suppression_Véhicule |
| $e_{9,2}$ | Retard_Véhicule |
| $e_{10,2}$ | Retard_Seuil_Véhicule |
| $e_{11,2}$ | Avance_Véhicule |
| $e_{12,2}$ | Avance_Seuil_Véhicule |
| $e_{13,2}$ | Contact_Station |
| $e_{14,2}$ | Contact_Véhicule |

**Tableau A2.2.** Evénements déclencheurs des services de l'*agent d'interface Etat_Ligne*

## I.3. Les Conditions

A chaque service $s_{i,j}$ est associée une condition $c_{i,j}$. Une condition peut être formée de plusieurs conditions élémentaires. Cependant, dans la plupart des cas, la condition d'un service est assimilée à l'apparition de l'événement. Autrement dit le service est déclenché dès l'apparition de l'événement (pas de condition particulière à vérifier). Le tableau A2.3 représente plusieurs fois la condition « apparition de l'événement » ce qui signifie l'apparition de l'événement correspondant au service. Par exemple pour $c_{1,2}$, la condition à vérifier est simplement l'apparition de l'événement $e_{1,2}$.

| Identifiant de la condition | Conditions à vérifier |
|---|---|
| $c_{1,2}$ | Apparition de l'évènement |
| $c_{2,2}$ | Apparition de l'évènement |
| $c_{3,2}$ | Apparition de l'évènement |
| $c_{4,2}$ | Apparition de l'évènement |
| $c_{5,2}$ | Apparition de l'évènement |
| $c6_{,2}$ | Apparition de l'événement |
| $c_{7,2}$ | Apparition de l'événement |
| $c_{8,2}$ | Apparition de l'événement |
| $c_{9,2}$ | Apparition de l'évènement **et** $1 \leq$ retard $< 5$ |
| $c_{10,2}$ | Apparition de l'évènement **et** retard $\geq 5$ |
| $c_{11,2}$ | Apparition de l'évènement $1 \leq$ avance $< 5$ |
| $c_{12,2}$ | Apparition de l'évènement avance $\geq 5$ |
| $c_{13,2}$ | Apparition de l'évènement |
| $C_{14,2}$ | Apparition de l'évènement |

**Tableau A2.3.** Conditions à vérifier par l'*agent d'interface Etat_Ligne*

## I.4. Les Ressources

Les ressources sont nécessaires à l'établissement des services qui comportent des actions visibles (Cf. chapitre trois, § III.2). Une ressource peut être formée de plusieurs ressources élémentaires. Elle peut correspondre à une fenêtre, un cadre, une boîte de dialogue, etc. Nous n'imposons pas d'identification particulière pour les ressources, c'est-à-dire que nous précisons simplement la nature de la ressource et nous laissons libre choix au concepteur de les identifier. Ainsi, l'apparition multiple d'une ressource « cadre » dans le tableau A2.4 ne signifie pas que l'on utilise un seul « cadre » mais signifie simplement que le service a besoin d'une ressource de type « cadre ».

| Identifiant de la ressource | Nom de la ressource |
|---|---|
| $r_{1,2}$ | Fenêtre (*Form*) |
| $r_{2,2}$ | Fenêtre (*Form*) |
| $r_{3,2}$ | Cadre (*Panel*) |
| $r_{4,2}$ | Fenêtre (*Form*) |
| $r_{5,2}$ | Fenêtre (*Form*) |
| $r_{6,2}$ | Cadre (*Panel*) |
| $r_{7,2}$ | Cadre (*Panel*) |
| $r_{8,2}$ | Cadre (*Panel*) |
| $r_{9,2}$ | Cadre (*Panel*) |
| $r_{10,2}$ | Cadre (*Panel*) |
| $r_{11,2}$ | Cadre (*Panel*) |
| $r_{12,2}$ | Cadre (*Panel*) |

**Tableau A2.4.** Ressources requises par l'*agent d'interface Etat_Ligne*

## I.5. Les Actions visibles

Les actions visibles représentent les propriétés internes du service (Cf. Chapitre III, § III.2) et portent sur l'agent lui-même. Le tableau A.2.5 présente les actions visibles que peut effectuer l'*agent d'interface* Etat_Ligne.

| Identifiant de l'action visible | Nom de l'action | Description |
|---|---|---|
| $acv_{1,2}$ | Afficher_Ligne | Afficher la ligne sélectionnée |
| $acv_{2,2}$ | Cacher_Ligne | Cacher la ligne sélectionnée |
| $acv_{3,2}$ | Colorer_Panel_Véhicule | Colorer le panel correspondant au véhicule pointé |
| $acv_{4,2}$ | Agrandir_Ligne | Faire un zoom avant sur la Ligne sélectionnée |
| $acv_{5,2}$ | Réduire_Ligne | Faire un arrière avant sur la Ligne sélectionnée |
| $acv_{6,2}$ | Colorer_Panel_Station | Colorer le panel correspondant à la station pointée |
| $acv_{7,2}$ | Ajouer_Panel_Véhicule | Ajouter à la ligne sélectionnée la représentation d'un véhicule |
| $acv_{8,2}$ | Supprimer_Panel_Véhicule | Supprimer de la ligne sélectionnée la représentation d'un véhicule |
| $acv_{9,2}$ | Colorer_Retard_Panel_Véhicule | Colorer le panel correspondant au véhicule avec le retard correspondant |
| $acv_{10,2}$ | Colorer_Retard_Seuil_Panel_Véhicule | Colorer le panel correspondant au véhicule avec le retard seuil correspondant |
| $acv_{11,2}$ | Colorer_Avance_Panel_Véhicule | Colorer le panel correspondant au véhicule avec l'avance correspondante |
| $acv_{12,2}$ | Colorer_Avance_Seuil_Panel_Véhicule | Colorer le panel correspondant au véhicule avec l'avance seuil correspondante |

**Tableau A2.5.** Actions visibles exécutées par l'*agent d'interface Etat_Ligne*

## I.6. Les Actions non visibles

Les actions non visibles représentent les propriétés externes du service (Cf. Chapitre trois, § III.2) et portent sur un autre agent. Le tableau A.2.6 présente les actions non visibles que peut effectuer l'*agent d'interface* Etat_Ligne.

| Identifiant de l'action non visible | Action Non Visible | Description |
|---|---|---|
| $acn_{1,2}$ | Informer_Agent_Station_e13 | Informer l'agent d'interface Station : $e_{1,3}$ |
| $acn_{2,2}$ | Informer_Agent_Véhicule_e14 | Informer l'agent d'interface Véhicule : $e_{1,4}$ |

**Tableau A2.6.** Actions non visibles exécutées par l'*agent d'interface Etat_Ligne*

## II. Spécification de l'agent d'interface Station

### II.1. Les Services

| Identifiant du servuce | Services | Description |
|---|---|---|
| $s_{1,3}$ | Afficher_Vue | Affichage de la vue station |
| $s_{2,3}$ | Envoyer_Message | Validation de l'envoi d'un message aux voyageurs en station |
| $s_{3,3}$ | Annuler_Envoi_Message | Annulation de l'envoi d'un message |
| $s_{4,3}$ | Accéder_Messages_Prédéfinis | Accès aux messages prédéfinis |
| $s_{5,3}$ | Visualiser_Passage1 | Visualisation des informations relatives au prochain véhicule qui s'arrête à la station |
| $s_{6,3}$ | Visualiser_Passage2 | Visualisation des informations relatives au deuxième prochain véhicule qui s'arrête à la station |
| $s_{7,3}$ | Changer_Retard/Avance1 | Changement de la valeur du retard/avance du prochain véhicule |
| $s_{8,3}$ | Changer_Destination1 | Changement de la destination du prochain véhicule |
| $s_{9,3}$ | Changer_Horaire_Arrivée1 | Changement de l'horaire d'arrivée du prochain véhicule |
| $s_{10,3}$ | Changer_Retard/Avance2 | Changement de la valeur du retard/avance du deuxième prochain véhicule |
| $s_{11,3}$ | Changer_Destination2 | Changement de la destination du deuxième prochain véhicule |
| $s_{1,3}$ | Changer_Horaire_Arrivée2 | Changement de l'horaire d'arrivée du deuxième prochain véhicule |

**Tableau A2.7.** Services de l'*agent d'interface Station*

### II.2. Les événements

| Identifiant de l'évènement | Evènement |
|---|---|
| $e_{1,3}$ | Affichage_Vue_Station |
| $e_{2,3}$ | Clic_Ok |
| $e_{3,3}$ | Clic_annuler |
| $e_{4,3}$ | Clic_Messages_Prédéfini |
| $e_{5,3}$ | Clic_Passage1 |
| $e_{6,3}$ | Clic_Passage2 |
| $e_{7,3}$ | Changement_Retard/Avance1 |
| $e_{8,3}$ | Changement_Destination1 |
| $e_{9,3}$ | Changement_Horaire_Arrivée1 |
| $e_{10,3}$ | Changement_Retard/Avance2 |
| $e_{11,3}$ | Changement_Destination2 |
| $e_{12,3}$ | Changement_Horaire_Arrivée2 |

**Tableau A2.8.** Evénements de l*'agent d'interface Station*

### II.3. Les Conditions

| Identifiant de la condition | Conditions à vérifier |
|---|---|
| $c_{1,3}$ | Apparition de l'événement |
| $c_{2,3}$ | Apparition de l'événement |
| $c_{3,3}$ | Apparition de l'événement |
| $c_{4,3}$ | Apparition de l'événement |
| $c_{5,3}$ | Apparition de l'événement |
| $c_{6,3}$ | Apparition de l'événement |
| $c_{7,3}$ | Apparition de l'événement |
| $c_{8,3}$ | Apparition de l'évènement |
| $c_{9,3}$ | Apparition de l'événement |
| $c_{10,3}$ | Apparition de l'événement |
| $c_{11,3}$ | Apparition de l'événement |
| $c_{12,3}$ | Apparition de l'événement |

**Tableau A2.9.** Conditions à vérifier de l*'agent d'interface Station*

## II.4. Les Ressources

| Identifiant de la ressource | Nom de la ressource |
|---|---|
| $r_{1, 3}$ | Fenêtre (*Form*) |
| $r_{2, 3}$ | Bouton |
| $r_{3, 3}$ | Bouton |
| $r_{4, 3}$ | Bouton |
| $r_{5, 3}$ | boîte de dialogue multi-page |
| $r_{6, 3}$ | boîte de dialogue multi-page |
| $r_{7, 3}$ | Cadre (*Panel*) |
| $r_{8,3}$ | Cadre (*Panel*) |
| $r_{9,3}$ | Cadre (*Panel*) |
| $r_{10,3}$ | Cadre (*Panel*) |
| $r_{11,3}$ | Cadre (*Panel*) |
| $r_{12,3}$ | Cadre (*Panel*) |

**Tableau A2.10.** Ressources requises par l'*agent d'interface Station*

## II.5. Les Actions visibles

| Identifiant de l'action visible | Nom de l'action | Description |
|---|---|---|
| $acv_{1,3}$ | Apparaître_Vue | Afficher la vue station |
| $acv_{2,3}$ | Envoyer_Message | Envoyer le message |
| $acv_{3,3}$ | Disparaître_Vue | Cacher la vue station |
| $acv_{4,3}$ | Mettre_En_Avant_Boîte_De_Dialogue1 | Mettre en avant la boîte de dialogue qui contient les informations sur le prochain véhicule s'arrêtant à la station |
| $acv_{5,3}$ | Mettre_En_Avant_Boîte_De_Dialogue2 | Mettre en avant la boîte de dialogue contenant les informations sur le deuxième prochain véhicule s'arrêtant à la station |
| $acv_{6,3}$ | Changer_Retard/Avance_Prochain_Véhicule | Changer la valeur du retard/avance du prochain véhicule qui s'arrête à la station |
| $acv_{7,3}$ | Changer_Horaire_Arrivée_Prochain_Véhicule | Changer l'horaire d'arrivée en station du prochain véhicule |
| $acv_{8,3}$ | Changer_Déstination_Prochain_Véhicule | Changer la destination du prochain véhicule qui s'arrête en station |
| $acv_{9,3}$ | Changer_Retard/Avance_Deuxième_Prochain_ Véhicule | Changer la valeur du retard/avance du deuxième prochain véhicule qui s'arrête en station |
| $acv_{10,3}$ | Changer_Horaire_Arrivée_ Deuxième _Prochain_Véhicule | Changer l'horaire d'arrivée en station du deuxième prochain véhicule |
| $acv_{11,3}$ | Changer_Déstination_ Deuxième_ Prochain_Véhicule | Changer la destination du prochain véhicule qui s'arrête en station |

**Tableau A2.11.** Actions visibles exécutées par l'*agent d'interface Station*

## II.6. Les Actions non visibles

| Identifiant de l'action non visible | Actions non visibles | Description |
|---|---|---|
| $Acn_{1,3}$ | Informer_Agent_Message_e15 | Informer l'agent Message e1,5 |

**Tableau A2.12.** Actions non visibles exécutées par l'*agent d'interface Station*

# III. Spécification de l'agent d'interface Véhicule

## III.1. Les Services

| Identifiant du service | Services | Description |
|---|---|---|
| $s_{1,4}$ | Afficher_Vue | Affichage de la vue véhicule |
| $s_{2,4}$ | Envoyer_Message | Validation de l'envoi d'un message au voyageurs/conducteur en véhicule |
| $s_{3,4}$ | Annuler_Envoi_Message | Annulation de l'envoi d'un message |
| $s_{4,4}$ | Accéder_Messages_Prédéfinis | Accès aux messages prédéfinis |
| $s_{5,4}$ | Accéder_Message_Voyageur | Accès du message destiné aux voyageurs à bord du véhicule |
| $s_{6,4}$ | Visualiser_Message_Conducteur | Visualisation du message à bord du véhicule destiné au conducteur |
| $s_{7,4}$ | Changer_Nom_Conducteur | Changement du nom du conducteur du véhicule |
| $s_{8,4}$ | Changer_Service_Véhicule | Changement du service du véhicule |
| $s_{9,4}$ | Changer_Déstination_Véhicule | Changement de la destination du véhicule |
| $s_{10,4}$ | Changer_Nom_Ligne_Véhicule | Changement du nom de la ligne à laquelle appartient le véhicule |
| $s_{11,4}$ | Visualiser_Arrêt1 | Visualisation des informations relatives à la prochaine station |
| $s_{12,4}$ | Visualiser_Arrêt2 | Visualisation des informations relatives à la deuxième prochaine station |
| $s_{13,4}$ | Visualiser_Arrêt3 | Visualisation des informations relatives à la troisième prochaine station |
| $s_{14,4}$ | Changer_Retard/Avance_Arrêt1 | Changement de la valeur du retard/avance du véhicule pour atteindre la prochaine station |
| $s_{15,4}$ | Changer_Horaire_Arrêt1 | Changement de l'horaire d'arrivée du véhicule à la prochaine station |
| $s_{16,4}$ | Changer_Nom_Arrêt1 | Changement du nom de la prochaine station desservie par le véhicule |
| $s_{17,4}$ | Changer_Retard/Avance_Arrêt2 | Changement de la valeur du retard/avance du véhicule pour atteindre la deuxième prochaine station |
| $s_{18,4}$ | Changer_Horaire_Arrêt2 | Changement de l'horaire d'arrivée du véhicule à la deuxième prochaine station |
| $s_{19,4}$ | Changer_Nom_Arrêt2 | Changement du nom de la deuxième prochaine station desservie par le véhicule |
| $s_{20,4}$ | Changer_Retard/Avance_Arrêt3 | Changement de la valeur du retard/avance du véhicule pour atteindre la troisième prochaine station |
| $s_{21,4}$ | Changer_Horaire_Arrêt3 | Changement de l'horaire d'arrivée du véhicule à la troisième prochaine station |
| $s_{21,4}$ | Changer_Nom_Arrêt3 | Changement du nom de la troisième prochaine station desservie par le véhicule |

**Tableau A2.13.** Services de l'*agent d'interface Véhicule*

## III.2. Les événements

| Identifiant de l'évènement | Evènement |
|---|---|
| $e_{1,4}$ | Afficher_Vue_Véhicule |
| $e_{2,4}$ | Cilck_Ok |
| $e_{3,4}$ | Clic_Annuler |
| $e_{4,4}$ | Clic_Messages_Prédéfini |
| $e_{5,4}$ | Clic_Message_Voyageur |
| $e_{6,4}$ | Clic_Message_Condicteur |
| $e_{7,4}$ | Changement_Nom_Conducteur |
| $e_{8,4}$ | Changement _Service_Véhicule |
| $e_{9,4}$ | Changment_Déstination_Véhicule |
| $e_{10,4}$ | Changement_Nom_Ligne_Véhicule |
| $e_{11,4}$ | Clic_Arrêt1 |
| $e_{12,4}$ | Clic_Arrêt2 |
| $e_{13,4}$ | Clic_Arrêt3 |
| $e_{14,4}$ | Changement_Retard/Avance_Arrêt1 |
| $e_{15,4}$ | Changement_Horaire_Arrêt1 |
| $e_{16,4}$ | Changement_Nom_Arrêt1 |
| $e_{17,4}$ | Changement_Retard/Avance_Arrêt2 |
| $e_{18,4}$ | Changement_Horaire_Arrêt2 |

| | |
|---|---|
| $e_{19,4}$ | Changement_Nom_Arrêt2 |
| $e_{20,4}$ | Changement_Retard/Avance_Arrêt3 |
| $e_{21,4}$ | Changement_Horaire_Arrêt3 |
| $e_{22,4}$ | Changement_Nom_Arrêt3 |

**Tableau A2.14.** Evénements de l'*agent d'interface Véhicule*

## III.3. Les Conditions

| Identifiant de la condition | Conditions à vérifier |
|---|---|
| $c_{1,3}$ | Apparition de l'événement |
| $c_{2,3}$ | Apparition de l'événement |
| $c_{3,3}$ | Apparition de l'événement |
| $c_{4,3}$ | Apparition de l'événement |
| $c_{5,3}$ | Apparition de l'événement |
| $c_{6,3}$ | Apparition de l'événement |
| $c_{7,3}$ | Apparition de l'événement |
| $c_{8,3}$ | Apparition de l'évènement |
| $c_{9,3}$ | Apparition de l'événement |
| $c_{10,3}$ | Apparition de l'événement |
| $c_{11,3}$ | Apparition de l'événement |
| $c_{12,3}$ | Apparition de l'événement |
| $c_{13,3}$ | Apparition de l'évènement |
| $c_{14,3}$ | Apparition de l'évènement |
| $c_{15,3}$ | Apparition de l'évènement |
| $c_{16,3}$ | Apparition de l'évènement |
| $c_{17,3}$ | Apparition de l'évènement |
| $c_{18,3}$ | Apparition de l'évènement |
| $c_{19,3}$ | Apparition de l'évènement |
| $c_{20,3}$ | Apparition de l'évènement |
| $c_{21,3}$ | Apparition de l'évènement |
| $c_{22,3}$ | Apparition de l'évènement |

**Tableau A2.15.** Conditions à vérifier de l'*agent d'interface Véhicule*

## III.4. Les Ressources

| Identifiant de la ressource | Nom de la ressource |
|---|---|
| $r_{1, 4}$ | Fenêtre (*Form*) |
| $r_{2, 4}$ | Bouton |
| $r_{3, 4}$ | Bouton |
| $r_{4, 4}$ | Bouton |
| $r_{5, 4}$ | boîte de dialogue multi-page |
| $r_{6, 4}$ | boîte de dialogue multi-page |
| $r_{7, 4}$ | Cadre (*Panel*) |
| $r_{8,4}$ | Cadre (*Panel*) |
| $r_{9,4}$ | Cadre (*Panel*) |
| $r_{10,4}$ | Cadre (*Panel*) |
| $r_{11,4}$ | boîte de dialogue multi-page |
| $r_{12,4}$ | boîte de dialogue multi-page |
| $r_{13,4}$ | boîte de dialogue multi-page |
| $r_{14,4}$ | Cadre (*Panel*) |
| $r_{15,4}$ | Cadre (*Panel*) |
| $r_{16,4}$ | Cadre (*Panel*) |
| $r_{17,4}$ | Cadre (*Panel*) |
| $r_{18,4}$ | Cadre (*Panel*) |
| $r_{19,4}$ | Cadre (*Panel*) |
| $r_{20,4}$ | Cadre (*Panel*) |
| $r_{21,4}$ | Cadre (*Panel*) |
| $r_{22,4}$ | Cadre (*Panel*) |

**Tableau A2.16.** Ressources requises par l'*agent d'interface Véhicule*

## III.5. Les Actions visibles

| Identifiant de l'action visible | Nom de l'action | Description |
|---|---|---|
| $acv_{1,4}$ | Apparaître_Vue | Afficher la vue véhicule |
| $acv_{2,4}$ | Envoyer_Message | Envoyer le message |
| $acv_{3,4}$ | Disparaître_Vue | Cacher la vue véhicule |
| $acv_{4,4}$ | Mettre_En_Avant_Boîte_De_Dialogue2 | Mettre en avant la boîte de dialogue qui affiche le message dessiné aux voyageurs |
| $acv_{5,4}$ | Mettre_En_Avant_Boîte_De_Dialogue2 | Mettre en avant la boîte de dialogue qui affiche le message dessiné au conducteur |
| $acv_{6,4}$ | Changer_Nom_Conducteur_Véhicule | Changer le nom du conducteur du véhicule |
| $acv_{7,4}$ | Changer_Service_Véhicule | Changer le service du véhicule |
| $acv_{8,4}$ | Changer_Destination_Véhicule | Changer la destination du véhicule |
| $acv_{9,4}$ | Changer_Ligne_Véhicule | Changer la ligne à laquelle appartient le véhicule |
| $acv_{10,4}$ | Mettre_En_Avant_Boîte_De_Dialogue1 | Mettre en avant la boîte de dialogue qui contient les informations sur la prochaine station desservie par le véhicule |
| $acv_{11,4}$ | Mettre_En_Avant_Boîte_De_Dialogue2 | Mettre en avant la boîte de dialogue qui contient les informations sur la deuxième prochaine station desservie par le véhicule |
| $acv_{12,4}$ | Mettre_En_Avant_Boîte_De_Dialogue3 | Mettre en avant la boîte de dialogue qui contient les informations sur la troisième prochaine station desservie par le véhicule |
| $acv_{13,4}$ | Changer _Retard/Avance_Arrêt1 | Changer la valeur du retard/avance du véhicule pour atteindre la prochaine station |
| $acv_{14,4}$ | Changer _Horaire_Arrêt1 | Changer de l'horaire d'arrivée du véhicule à la prochaine station |
| $acv_{15,4}$ | Changer _Nom_Arrêt1 | Changer le nom du prochain arrêt desservie par le véhicule |
| $acv_{16,4}$ | Changer _Retard/Avance_Arrêt2 | Changer la valeur du retard/avance du véhicule pour atteindre la deuxième prochaine station |
| $acv_{17,4}$ | Changer _Horaire_Arrêt2 | Changer de l'horaire d'arrivée du véhicule à la prochaine station |
| $acv_{18,4}$ | Changer _Nom_Arrêt2 | Changer le nom du deuxième prochain arrêt desservie par le véhicule |
| $acv_{19,4}$ | Changer _Retard/Avance_Arrêt3 | Changer la valeur du retard/avance du véhicule pour atteindre la troisième prochaine station |
| $acv_{20,4}$ | Changer _Horaire_Arrêt3 | Changer de l'horaire d'arrivée du véhicule à la prochaine station |
| $acv_{21,4}$ | Changer _Nom_Arrêt3 | Changer le nom du troisième prochain arrêt desservie par le véhicule |

**Tableau A2.17.** Actions visibles exécutées par l'*agent d'interface Véhicule*

## III.6. Les Actions non visibles

| **Identifiant de l'action non visible** | **Actions non visibles** | **Description** |
|---|---|---|
| $acn_{1,4}$ | Informer_Agent_Message_e15 | Informer l'agent Message e1,5 |

**Tableau A2.18.** Actions non visibles exécutées par l'*agent d'interface Véhicule*

# IV. Spécification de l'agent d'interface Message

## IV.1. Les Services

| Identifiant du service | Services | description |
|---|---|---|
| $s_{1,5}$ | Afficher_Vue | Affichage de la vue message |
| $s_{2,5}$ | Envoyer_Message | Validation de l'envoi d'un message au(x) station(s)/véhicule(s) |
| $s_{3,5}$ | Annuler_Envoi_Message | Annulation de l'envoi d'un message |
| $s_{4,5}$ | Accéder_Edition_Messages | Accès à l'édition d'un nouveau message |
| $s_{5,5}$ | Accéder_Sélection_Message | Accès à la sélection d'un message existant |
| $s_{6,5}$ | Sélectionner_Ligne | Sélection de la ligne à laquelle le message à envoyer est destiné |
| $s_{7,5}$ | Sélectionner_Toutes_Lignes | Sélection de toutes les lignes pour un envoi de message multiple |
| $s_{8,5}$ | Sélectionner_Toutes_Stations | Sélection de toutes les stations pour un envoi de message multiple |
| $s_{9,5}$ | Sélectionner_Tous_Véhicules | Sélection de tous les véhicules pour un envoi de message multiple |

**Tableau A2.19.** Services de l'*agent d'interface Véhicule*

## VI.2. Les événements

| Identifiant de l'évènement | Evènement |
|---|---|
| $e_{1,5}$ | Afficher_Vue_Message |
| $e_{2,5}$ | Cilc_Ok |
| $e_{3,5}$ | Clic_Annuler |
| $e_{4,5}$ | Clic_Editer_Messages |
| $e_{5,5}$ | Clic_Sélectionner_Message |
| $e_{6,5}$ | Clic_Sélectionner_Ligne |
| $e_{7,5}$ | Cocher_Sélectionner_Touts_Lignes |
| $e_{8,5}$ | Cocher_Sélectionner_Touts_Stations |
| $e_{9,5}$ | Cocher_Sélectionner_Touts_Véhicules |

**Tableau A2.20.** Evénements de l*'agent d'interface Message*

## IV.3. Les Conditions

| Identifiant de la condition | Conditions à vérifier |
|---|---|
| $c_{1,5}$ | Apparition de l'événement |
| $c_{2,5}$ | Apparition de l'événement |
| $c_{3,5}$ | Apparition de l'événement |
| $c_{4,5}$ | Apparition de l'événement |
| $c_{5,5}$ | Apparition de l'événement |
| $c_{6,5}$ | Apparition de l'événement |
| $c_{7,5}$ | Apparition de l'événement |
| $c_{8,5}$ | Apparition de l'évènement |
| $c_{9,5}$ | Apparition de l'événement |

**Tableau A2.21.** Conditions à vérifier de l*'agent d'interface Message*

### IV.4. Les Ressources

| Identifiant de la ressource | Nom de la ressource |
|---|---|
| $r_{1,5}$ | Fenêtre (*Form*) |
| $r_{2,5}$ | Bouton |
| $r_{3,5}$ | Bouton |
| $r_{4,5}$ | boîte de dialogue multi-page |
| $r_{5,5}$ | boîte de dialogue multi-page |
| $r_{6,5}$ | Liste déroulante à choix unique (*ComboBox*) |
| $r_{7,5}$ | Case à cocher (*CheckBox*) |
| $r_{8,5}$ | Case à cocher (*CheckBox*) |
| $r_{9,5}$ | Case à cocher (*CheckBox*) |

**Tableau A2.22.** Ressources requises par l'*agent d'interface Message*

### IV.5. Les Actions visibles

| Identifiant de l'action visible | Nom de l'action | Description |
|---|---|---|
| $acv_{1,5}$ | Apparaître_Vue | Afficher la vue Message |
| $acv_{2,5}$ | Envoyer_Message | Envoyer le message |
| $acv_{3,5}$ | Disparaître_Vue | Cacher la vue Message |
| $acv_{4,5}$ | Mettre_En_Avant_Boîte_De_Dialogue1 | Mettre en avant la boîte de dialogue qui affiche le message dessiné aux voyageurs |
| $acv_{5,5}$ | Mettre_En_Avant_Boîte_De_Dialogue2 | Mettre en avant la boîte de dialogue qui affiche le message dessiné au conducteur |
| $acv_{6,5}$ | Sélectionner_Ligne | Sélection de la ligne à laquelle le message à envoyer est destiné |
| $acv_{7,5}$ | Sélectionner_Toutes_Lignes | Sélection de toutes les lignes pour un envoi de message multiple |
| $acv_{8,5}$ | Sélectionner_Toutes_Stations | Sélection de toutes les stations pour un envoi de message multiple |
| $acv_{9,5}$ | Sélectionner_Tous_Véhicules | Sélection de tous les véhicules pour un envoi de message multiple |

**Tableau A2.23.** Actions visibles exécutées par l'*agent d'interface Message*

### IV.6. Les Actions non visibles

| Identifiant de l'action non visible | Actions non visibles | Description |
|---|---|---|
| $Acn_{1,5}$ | Informer_Agent_Contrôleur_Station | Informer l'agent Contrôleur Station |
| $Acn_{2,5}$ | Informer_Agent_Contrôleur_Véhicule | Informer l'agent Contrôleur Véhicule |

**Tableau A2.24.** Actions non visibles exécutées par l'*agent d'interface Message*

## V. Spécification de l'agent d'interface Vue_Globale

La spécification de l'*agent d'interface Vue_Globale* ne nécessite pas la définition des services. En effet, au stade actuel du développement du SAI, la vue de l'*agent d'interface Vue_Globale* se limite à la supervision du trafic sur le réseau. Le régulateur ne peut donc pas interagir avec cette vue pour la sélection, ou l'édition d'un composant graphique.

# *Questionnaire utilisé lors de l'évaluation du Système d'Aide à l'Information*

1. *Questions générales concernant l'interface*
2. *Questions spécifiques à chaque type de vue*
3. *Evaluation ergonomique globale de l'interface*

# Questionnaire

## Entretien

Date : / /

Par :

## Interlocuteur

Nom et prénom :

Age :

Verres correcteurs Non□ Oui□

Gaucher □ Droitier □

Fonction :

## Contenu du questionnaire

**I. Questions générales concernant l'interface**

**II. Questions spécifiques à chaque type de vue**

- II.1 La synthèse des retards (Etat Trafic)
- II.2. La synoptique de ligne (Etat Ligne)
- II.3. La vue de station
- II.4. La vue de véhicule
- II.5. La Messagerie

**III. Evaluation ergonomique globale de l'interface**

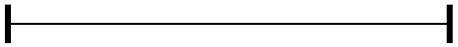
- III.1. Clarté
- III.2. Cohérence
- III.3. Compatibilité
- III.4. Retour informatif
- III.5. Flexibilité et contrôle
- III.6. Prévention et correction d'erreurs

# I. Questions générales concernant l'interface

I.1. Quels sont les paramètres que vous surveillez en priorité ?

❑ Retards ❑ Défaillances ❑ Pannes ❑ Position ❑ Correspondances ❑ Passages
❑ Arrêts ❑ Autre :

I.2. Ces paramètres sont-ils facilement accessibles grâce à l'interface ?



FACILE DIFFICILE

I.3. Quelles vues affichez-vous sur les écrans pendant le fonctionnement **normal** du procédé ? Précisez, si possible, le fait qui vous amène à utiliser cette vue.

| Vue Nécessaire | Evénement déclenchant / Cas |
|---|---|
| Synthèse des retards (Etat Trafic) | |
| Synoptique de Ligne (Etat Ligne) | |
| Vue de Station | |
| Vue de Véhicule | |
| Messagerie | |

I.4. Quelles vues affichez-vous sur les écrans pendant le fonctionnement **anormal** du procédé ? Précisez, si possible, le fait qui vous amène à utiliser cette vue.

| Vue Nécessaire | Evénement déclenchant / Cas |
|---|---|
| Synthèse des retards (Etat Trafic) | |
| Synoptique de Ligne (Etat Ligne) | |
| Vue de Station | |
| Vue de Véhicule | |
| Messagerie | |

I.5. Vous servez-vous souvent de votre mémoire pour mémoriser des paramètres ?

❑ OUI ❑ NON

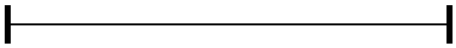
Si oui, dans quel but ?

I.6. Etes-vous parfois gêné par certaines caractéristiques ou fonctionnalités de l'interface?

❑ OUI ❑ NON

Si oui, lesquelles ?

I.7. Le temps de réponse de l'interface est-il trop lent ou trop rapide ?



LENT RAPIDE

I.8. A première vue, quels sont les points forts de l'interface ?

Synthèse des retards (Etat Trafic) □
Synoptique de Ligne (Etat Ligne) □
Vue de Station □
Vue de Véhicule □
Messagerie □

Autre :

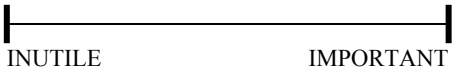
I.9. A première vue, quels sont les points faibles de l'interface ?

Synthèse des retards (Etat Trafic) □
Synoptique de Ligne (Etat Ligne) □
Vue de Station □
Vue de Véhicule □
Messagerie □

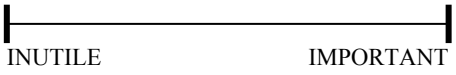
Autre :

I.10. Evaluez l'importance des vues suivantes :

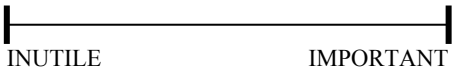
Synthèse des retards (Etat Trafic)



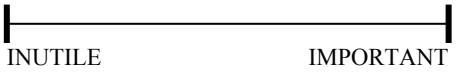

Synoptique de Ligne (Etat Ligne)



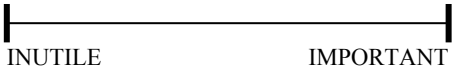

Vue de Station




Vue de Véhicule




Messagerie




## II. Questions spécifiques à chaque type de vue

II.1. La vue synthèse des retards (Etat Trafic)

- A-t-elle été utile lors de votre travail de régulation ? ❑ OUI ❑ NON

- Quels sont ses points forts ?
  ❑ Présentation ❑ Contenu ❑ Autre :

- Quels sont ses points faibles ?
  ❑ Présentation ❑ Contenu ❑ Autre :

- Quels sont les éléments qui sont indispensables sur cette vue?

| | |
|---|---|
| □Retard Graphe Bar | □Graduations |
| □Valeur Retard | □Regroupage par ligne |
| □Numéro de Bus | □Accès vue véhicule |
| □Ligne de seuil | □Accès vue ligne |
| □Numéro de Ligne | |

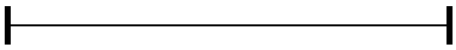
Autre :

- Quels sont les oublis ou les erreurs que vous avez remarqués ?

- Est-il possible de commettre des erreurs en utilisant cette vue ? ❑ OUI ❑ NON
  Si oui, lesquelles ?

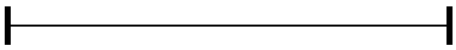
- Serait-il utile d'améliorer cette vue et quelles seraient vos suggestions ?

LISIBILITE : facilité de lecture des informations à l'écran



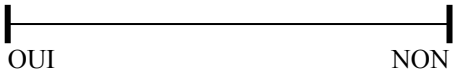
FACILE DIFFICILE

INTERPRETATION : facilité d'interprétation des informations affichées à l'écran



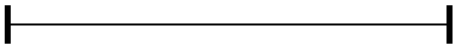
FACILE DIFFICILE

SATISFACTION : les informations affichées répondent-elles aux besoins



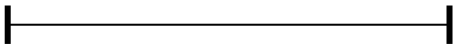
OUI NON

PERTINENCE : informations affichées au bon moment/correctement



OUI NON

SOUPLESSE et CONVIVIALITE : utilisation facile et intuitive



OUI NON

II.2. La vue synoptique de ligne (Etat Ligne)

- A-t-elle été utile lors de votre travail de régulation ? ❑ OUI ❑ NON

- Quels sont ses points forts ?
  ❑ Présentation ❑ Contenu ❑ Autre :

- Quels sont ses points faibles ?
  ❑ Présentation ❑ Contenu ❑ Autre :

- Quels sont les éléments qui sont indispensables sur cette vue?

| | |
|---|---|
| □Zoom | □Nom de la station |
| □Vue linéaire | □Aspect du véhicule |
| □Retards | □Accès vue vehicule |
| □Accès vue station | □Accès edition message |

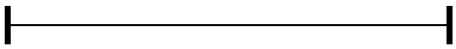
Autre :

- Est-il possible de commettre des erreurs en utilisant cette vue ? ❑ OUI ❑ NON
  Si oui, lesquelles ?

- Quels sont les oublis que vous avez remarqués ?

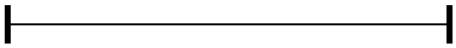
- Serait-il utile d'améliorer cette vue et quelles seraient vos suggestions ?

LISIBILITE : facilité de lecture des informations à l'écran



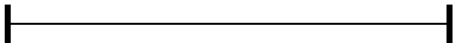
FACILE DIFFICILE

INTERPRETATION : facilité d'interprétation des informations affichées à l'écran



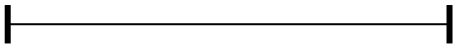
FACILE DIFFICILE

SATISFACTION : les informations affichées répondent-elles aux besoins ?



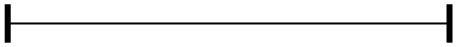
OUI NON

PERTINENCE : informations affichées au bon moment/correctement



OUI NON

SOUPLESSE et CONVIVIALITE : utilisation facile et intuitive



OUI NON

II.3. La vue Station

- A-t-elle été utile lors de votre travail de régulation ? ❑ OUI ❑ NON

- Quels sont ses points forts ?
  ❑ Présentation ❑ Contenu ❑ Autre :

- Quels sont ses points faibles ?
  ❑ Présentation ❑ Contenu ❑ Autre :

- Quels sont les éléments qui sont indispensables sur cette vue ?

| | |
|---|---|
| □Tri par destination | □Diagnostics Haut-parleur/affichage |
| □Onglets de passage | □Messages station |
| □Ligne | □Alertes |
| □ Horaire | □Persistance |
| □Retard | |

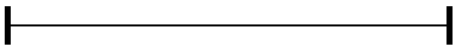
Autre :

- Est-il possible de commettre des erreurs en utilisant cette vue ? ❑ OUI ❑ NON
  Si oui, lesquelles ?

- Quels sont les oublis que vous avez remarqués ?

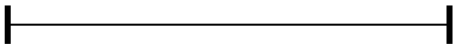
- Serait-il utile d'améliorer cette vue et quelles seraient vos suggestions ?

LISIBILITE : facilité de lecture des informations à l'écran



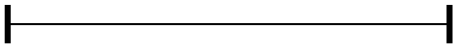
FACILE DIFFICILE

INTERPRETATION : facilité d'interprétation des informations affichées à l'écran



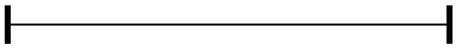
FACILE DIFFICILE

SATISFACTION : les informations affichées répondent-elles aux besoins ?



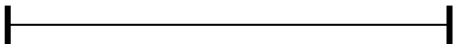
OUI NON

PERTINENCE : informations affichées au bon moment/correctement



OUI NON

SOUPLESSE et CONVIVIALITE : utilisation facile et intuitive



OUI NON

II.4. La vue Véhicule

- A-t-elle été utile lors de votre travail de régulation ? ❑ OUI ❑ NON

- Quels sont ses points forts ?
  ❑ Présentation ❑ Contenu ❑ Autre :

- Quels sont ses points faibles ?
  ❑ Présentation ❑ Contenu ❑ Autre :

- Quels sont les éléments qui sont indispensables sur cette vue ?

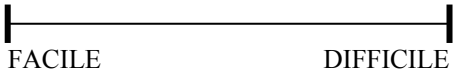
□Ligne □Horaire
□Destination □Diagnostics Haut-parleur/affichage
□Onglets Arrêts □Conducteur et service
□ Retard □Message Conducteur
Autre : □Message voyageurs

- Est-il possible de commettre des erreurs en utilisant cette vue ? ❑ OUI ❑ NON
  Si oui, lesquelles ?

- Quels sont les oublis que vous avez remarqués ?

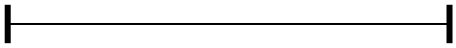
- Serait-il utile d'améliorer cette vue et quelles seraient vos suggestions ?

LISIBILITE : facilité de lecture des informations à l'écran



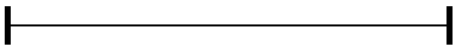
FACILE DIFFICILE

INTERPRETATION : facilité d'interprétation des informations affichées à l'écran



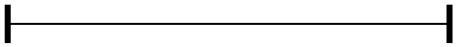
FACILE DIFFICILE

SATISFACTION : les informations affichées répondent-elles aux besoins ?



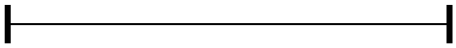
OUI NON

PERTINENCE : informations affichées au bon moment/correctement



OUI NON

SOUPLESSE et CONVIVIALITE : utilisation facile et intuitive



OUI NON

II.5. La vue "messagerie"

- A-t-elle été utile lors de votre travail de régulation ? ❑ OUI ❑ NON

- Quels sont ses points forts ?
  ❑ Présentation ❑ Contenu ❑ Autre :

- Quels sont ses points faibles ?
  ❑ Présentation ❑ Contenu ❑ Autre :

- Quels sont les éléments qui sont indispensables sur cette vue ?

  □ Sélection des stations □ Edition des messages
  □ Sélection des véhicules □ Sélection des messages
  □ Sélection de la ligne

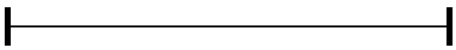
Autre :

- Est-il possible de commettre des erreurs en utilisant cette vue ? ❑ OUI ❑ NON
  Si oui, lesquelles ?

- Quels sont les oublis que vous avez remarqués ?

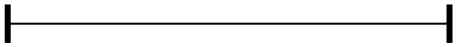
- Serait-il utile d'améliorer cette vue et quelles seraient vos suggestions ?

LISIBILITE : facilité de lecture des informations à l'écran



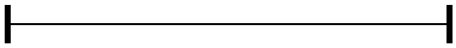
FACILE DIFFICILE

INTERPRETATION : facilité d'interprétation des informations affichées à l'écran



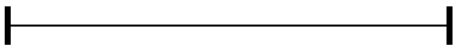
FACILE DIFFICILE

SATISFACTION : les informations affichées répondent-elles aux besoins



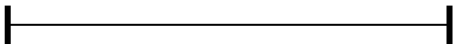
OUI NON

PERTINENCE : informations affichées au bon moment/correctement



OUI NON

SOUPLESSE et CONVIVIALITE : utilisation facile et intuitive



OUI NON

## III. Evaluation ergonomique globale de l'interface

Passons en revue des caractéristiques ergonomiques, avec, comme pré-requis : le système doit répondre aux besoins et aux conditions de l'opérateur quand il effectue des tâches.

III.1. CLARTE : les informations affichées sur un écran doivent être claires, bien organisées, sans ambiguïtés et faciles à lire.

- Quelles sont les vues "fatiguantes" ou trop chargées ?

Synthèse des retards (Etat Trafic) □
Synoptique de Ligne (Etat Ligne) □
Vue de Station □
Vue de Véhicule □
Messagerie □

Autre :

- Les informations importantes sont-elles toujours bien mises en évidence ? ❑ OUI ❑ NON

- Les informations sont-elles toujours organisées logiquement ? ❑ OUI ❑ NON

- La manière dont chaque vue est structurée est-elle correcte ? ❑ OUI ❑ NON

- Les couleurs rendent-elles les informations plus compréhensibles ? ❑ OUI ❑ NON

- Peut-on bien lire chaque valeur ? ❑ OUI ❑ NON

- Est-il facile de trouver l'information voulue ? ❑ OUI ❑ NON

- L'organisation de l'interface est-elle évidente ? ❑ OUI ❑ NON

III.2. COHERENCE : tout doit être cohérent d'une vue à l'autre.

- Les couleurs sont-elles toujours cohérentes d'une vue à l'autre ? ❑ OUI ❑ NON

- Les noms, les libellés et les abréviations sont-ils toujours cohérents d'une vue à l'autre ?
  ❑ OUI ❑ NON

  Si non, lesquels qui ne sont pas cohérents ?

- Certaines représentations sont-elles différentes d'une vue à l'autre ? ❑ OUI ❑ NON
  Si oui lesquelles ?

III.3. COMPATIBILITE : l'interface doit être compatible avec les conventions et les attentes de l'utilisateur.

- Les couleurs respectent-elles les associations conventionnelles (ex : couleurs des alarmes, etc.) ?
  ❑ OUI ❑ NON
- Le jargon utilisé pour l'interface vous est-il familier ? ❑ OUI ❑ NON

- Est-ce que chaque nom, libellé ou abréviation est facile à comprendre ? ❑ OUI ❑ NON

- y-a-t-il des éléments ou des détails que vous ne comprenez pas sur certaines vues ?
  ❑ OUI ❑ NON

Si oui Lesquels ?

- Est-ce que chaque vue respecte les tâches à réaliser avec celles-ci et les besoins qui en découlent ?
  ❑ OUI ❑ NON

III.4. RETOUR INFORMATIF : les utilisateurs doivent avoir des informations claires, des retours informatifs sur l'endroit où ils se trouvent dans le système, sur les actions qu'ils doivent effectuer, si ces actions ont réussi et savoir quelles sont les actions suivantes.

- Lorsqu'on doit saisir une donnée ou indiquer une valeur, chaque action est-elle claire ?
  ❑ OUI ❑ NON

- Dispose-t-on toujours d'un retour d'informations suffisant, suite à une action ou une demande ?
  ❑ OUI ❑ NON

- Quand un paramètre important change de valeur, est-ce évident à constater ?
  ❑ OUI ❑ NON

- Si des messages d'erreurs existent pour ces vues, sont-ils explicites ?
  ❑ OUI ❑ NON

III.5. FLEXIBILITE ET CONTROLE : l'interface doit être suffisamment flexible dans la structure, dans la manière dont l'information est présentée et dans ce que peut faire l'utilisateur, pour adapter les besoins et les conditions de tous les utilisateurs, et pour leur permettre de percevoir le contrôle du système.

- Existe-t-il toujours des raccourcis pour aller vers l'information désirée ? ❑ OUI ❑ NON

- Prévoir d'autres raccourcis serait-il utile ? ❑ OUI ❑ NON
- Lesquels ?

- Eprouvez-vous parfois des difficultés pour parcourir une séquence de vues, cette séquence étant nécessaire pour bien effectuer la tâche ? ❑ OUI ❑ NON

- Souhaiteriez-vous pouvoir configurer en fonction de vos besoins certaines caractéristiques de l'interface (voir un ensemble spécifique de variables en même temps, changer certaines couleurs, des seuils, etc.) ? ❑ OUI ❑ NON

III.6. PREVENTION ET CORRECTION D'ERREURS : Le système doit être conçu pour minimiser les possibilités d'erreurs de l'utilisateur, avec des facilités incorporées pour détecter et traiter celles qui apparaîtront ; les opérateurs pourront vérifier leurs entrées et corriger leurs erreurs, ou les situations d'erreurs potentielles avant le traitement des entrées.

- Lors d'une saisie d'information, quand vous vous apercevez d'une erreur de manipulation, pouvez-vous revenir en arrière ? ❑ OUI ❑ NON

- Pouvez vous corriger cette erreur d'une autre manière ? ❑ OUI ❑ NON

## Résumé

Ce mémoire présente une contribution à l'évaluation des systèmes interactifs basés sur des architectures orientées agents. Avec l'apparition de nouvelles architectures et de nouveaux systèmes à base d'agents, de nouvelles problématiques d'évaluation des systèmes interactifs orientés agents apparaissent. Notre contribution consiste à proposer un principe de couplage entre une architecture à base d'agents du système interactif et son évaluation. En se basant sur ce principe, un système d'aide à l'évaluation basé sur un mouchard électronique baptisé MESIA (Mouchard électronique pour l'Evaluation des Systèmes interactifs Orientés Agent) a été proposé.

Nous avons utilisé ce mouchard, dans le cadre du projet SART (Système d'Aide à la Régulation de Trafic), pour évaluer un Système d'Aide à l'Information des voyageurs (SAI). Une première évaluation a permis de valider globalement la maquette proposée. Les perspectives de recherche, visant en particulier à améliorer le système d'aide à l'évaluation proposé et à effectuer une évaluation dans un cadre réel de fonctionnement du SAI, terminent ce mémoire de cette thèse.



**Mots clés** : Évaluation, Interaction Homme-Machine, systèmes interactifs, architecture à base d'agents, mouchard électronique.



## Abstract

This thesis presents a contribution to the evaluation of interactive systems based on oriented agent architecture. The appearance of new agent based architectures and systems provide new problems concerning the evaluation of agent based interactive systems. So, our work aims at the proposal of a principle that couples agent-based architecture of interactive systems and their evaluation. Based on this principle, we propose an evaluation aid system based on an electronic informer which allows us to acquire and analyse human-machine interactions in agent-based interactive environments. This electronic informer is used in the framework of the SART (Système d'Aide à la Régulation de Trafic) project in order to evaluate an Information System. This system was validated by a first evaluation in laboratory. The perspectives of this work aim mainly the improvement of the evaluation aid system and the evaluation of the Information System in a real work case.



**Keywords**: Evaluation, Human-Machine Interaction, Interactive systems, agent-based architecture, electronic informer.