N° d'ordre : 02/2011-M/EL

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE HOUARI BOUMEDIENE

FACULTE D'ELECTRONIQUE ET D'INFORMATIQUE

MEMOIRE

Présenté pour l'obtention du diplôme de **MAGISTER**

EN : ELECTRONIQUE

Spécialité: Contrôle de Processus et Robotique

Par: M^me^ DJAID Nadia née TOUILEB

Thème

Architectures et modèles dynamiques dédiées aux applications multimodales pour renforcer l'interaction du robot

Soutenu publiquement le 22-09-2011 , devant le jury composé de :

M^me^ L.BOUMGHAR	**Professeur,**	**à l'USTHB.**	**Président**
M^r^ A. RAMDANE CHERIF	**Professeur,**	**à l'U.V.S.Q/ FRANCE.**	**Examinateur**
M^me^ N. GHARBI	**Maître de Conférences/A,**	**à l'USTHB.**	**Examinateur**
M^r^ M.GUIATNI	**Maître de Conférences/A,**	**à l'E.M.P.**	**Examinateur**
M^me^ N. SAADIA	**Maître de Conférences/A,**	**à l'USTHB.**	**Directrice de Mémoire**

REMERCIEMENTS

Je tiens à remercier tous ceux qui m'ont aidées à élaborer ce travail, en particulier ma promotrice Madame N. SAADIA qui a cru en moi et qui par ses conseils et ses commentaires précieux m'a aidé à surmonter les difficultés et avancer ainsi dans mon travail.

Je voudrai aussi exprimer mes remerciements à Madame F. BOUMGHAR, Professeur à l'U.S.T.H.B, d'avoir accepté de présider ce jury.

Je tiens également à remercier Monsieur A. RAMDANE CHERIF, Professeur à l'U.V.S.Q., pour son aide précieuse et ses remarques constructives, Madame N. GHARBI, Maître de conférences A à l'U.S.T.H.B. et Monsieur M. GUIATNI, Maître de conférences A à L'E.M.P., pour avoir accepté d'évaluer ce travail.

Un grand remerciement à tous les membres de l'équipe Robotique 2 du laboratoire LRPE de l'USTHB pour leur soutien.

Je remercie aussi les membres de ma famille, qui m'ont encouragée tout le long de ma vie et m'ont donné les moyens de réaliser mes projets. Qu'ils trouvent ici ma profonde reconnaissance pour leur confiance, leur support ainsi que les conseils qu'ils n'ont cessés de me prodiguer.

Enfin je salue tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail.

DÉDICACES

Je dédie ce travail à mon mari qui m'a toujours aidé et soutenu, à mon adorable fille Imene, à mes chers parents, à mes deux sœurs et à mes beaux- parents.

Sommaire

Liste des Tableaux

Table des figures

Liste des abréviations

- RPCTS: Réseau de Pétri Coloré Temporisé Stochastique
- Tm: Température du patient
- Sm: Tension artériel du patient
- Gest: Détection de la position désignée par le geste
- T1: Temps1 (Temps d'arrivé de la parole)
- T2: Temps2 (Temps d'arrivé d'un événement)
- B: Température basse
- N: Température Normale
- F: Température de fièvre
- MT1: Mauvaise tension 1
- MT2: Mauvaise tension 2
- MT3: Mauvaise tension 3
- MT4: Mauvaise tension 4
- BT: Bonne Tension
- TB: Très Bonne valeur de désignation (Erreur de désignation faible)
- B: Bonne valeur de désignation
- M: Mauvaise valeur de désignation
- Tmf: Température du patient traitée par la logique floue
- Smfp: Tension minimale du patient traitée par la logique floue
- SmfG: Tension maximale du patient traitée par la logique floue
- GC: Geste-Clic
- Prob.loc: Problème lors de la désignation ou absence de désignation
- ProbTm : Problème de température corporelle chez l'utilisateur
- ProbSm : Problème de tension artérielle chez l'utilisateur
- ProbTmSm: Problème de température et de tension chez l'utilisateur

- ArriveParole : Détection de l'arrivé d'un mot de commande
- ArriveEvent : Détection de l'arrivé d'un événement
- Event : Evénement
- TempsReconEvent : Temps de reconnaissance d'un événement
- ReconEvent : Reconnaissance d'un événement
- TEntreEvent : Temps entre l'arrivé des événements
- EventReconnu : Evénement reconnu
- FileEventReconu : File d'attente des événements reconnus
- ReconEnCours : Reconnaissance en cours
- ValeurContexte : Valeur correspondante au contexte (Température, tension et localisation par GPS)
- RecCMDSortie : Reconnaissance de la commande de sortie
- ErreurLocalis : Mauvaise désignation d'un événement
- PoseGest : Commande « Poser l'objet saisi à l'emplacement désigné par le geste »
- PoseClic : Commande « Poser l'objet saisi à l'emplacement désigné par le clic »
- PrendGest : Commande « Prendre l'objet désigné par le geste »
- PrendClic : Commande « Prendre l'objet désigné par le clic »
- Avance : commande « Avancer »
- AvanceGeste : Commande « Avancer jusqu'à la position désigné par le geste »
- AvanceClic : Commande « Avancer jusqu'à la position désigné par le clic »
- Droite : Commande « Tourner à droite »
- DroiteGeste : Commande « Tourner à droite jusqu'à la position désigné par le geste »
- DroiteClic : Commande « Tourner à droite jusqu'à la position désigné par le clic »
- Gauche : Commande « Tourner à gauche »

- GaucheGeste : Commande « Tourner à gauche jusqu'à la position désigné par le geste »
- GaucheClic : Commande « Tourner à gauche jusqu'à la position désigné par le clic »
- Arrière : Commande « Marche arrière »
- ArrièreGeste : Commande « Marche arrière jusqu'à la position désigné par le geste »
- ArrièreClic : Commande « Marche arrière jusqu'à la position désigné par le clic »
- Stop : Commande « arrêter » le fauteuil
- VerErreur : Vérification de l'erreur
- EventErreur : Evénement détecté et son erreur
- CmdSortie : Commande de sortie

Introduction Générale

INTRODUCTION GENERALE

Ce mémoire présente nos travaux de recherches qui portent sur l'élaboration d'une architecture logicielle multimodale qui prend en compte le contexte pour renforcer l'interaction homme-robot. Dans notre étude nous nous orienterons vers le domaine d'aide à la personne dépendante. Ceci dit, l'architecture proposée sera conçue pour permettre la commande d'un fauteuil roulant muni d'un bras manipulateur.

Les personnes dépendantes ne pouvant pas utiliser leurs corps pour pouvoir se déplacer et vivre correctement, ils rencontrent de très grandes difficultés dans leurs vies quotidiennes. Le simple fait de se déplacer, d'atteindre un objet, de le prendre ou même d'ouvrir une porte devient une chose difficile voir impossible à réaliser. C'est pour cette raison qu'il est indispensable pour eux d'avoir quotidiennement et continuellement de l'aide.

Les travaux de recherches de ces dernières années se sont orientés vers l'introduction de la multimodalité dans la commande des systèmes. Ceci a permis une facilité de communication entre les personnes et les machines. De plus, étant donné que les personnes évoluent dans un environnement complexe et varié, il est nécessaire de prendre en considération l'évolution et les changements possibles de ce dernier. Cet apport se traduit par la prise en compte du contexte qu'il soit relatif à la personne, à son environnement ou encore au système. Cette prise en charge du contexte permet d'apporter une facilité et un confort de vie indispensable pour les personnes dépendantes. En effet, la combinaison des deux parties dans la commande, la multimodalité et la prise en compte du contexte, va permettre d'offrir un confort de vie et une sécurité pour la personne utilisant ces systèmes.

De nos jours, des questions se posent encore sur cette combinaison et peu de travaux offrent des propositions d'architecture logicielle qui permettent de réaliser de telles applications. De ce fait, notre projet va porter sur la conception d'un système qui va permettre le guidage d'un fauteuil roulant muni d'un bras manipulateur grâce à une commande qui combine en entrée la multimodalité et la prise en compte du contexte. De ce fait, il est indispensable d'avoir une idée sur les travaux déjà existants dans ce domaine. Pour ce faire, nous présenterons dans le chapitre I un état de l'art sur les travaux effectués dans ce domaine.

Tout d'abord, nous donnerons quelques exemples sur les fauteuils roulant munis de bras manipulateur existant et nous présenterons différentes architectures de commande citées dans la littérature. Nous exposerons aussi quelques travaux sur les architectures multimodales et les

systèmes qui prennent en compte le contexte. En s'inspirant de l'ensemble de ces travaux, nous terminerons ce chapitre en établissant le cahier des charges du travail demandé.

Dans le chapitre II, nous présenterons des notions de base sur la multimodalité et le contexte en donnant quelques définitions et concepts.

Nous exposerons dans le chapitre III une architecture multimodale qui intègre le contexte en donnant des définitions et des formalismes de spécification du dialogue multimodal. Dans cette architecture nous ne considérons que l'état de santé de l'utilisateur. Cet état est modélisé par des variables binaires.

Nous intégrerons dans le chapitre IV la logique floue dans la prise en compte du contexte. En effet dans cette seconde architecture nous intégrons lors de la prise de décision l'état réel de santé de l'utilisateur et l'erreur qu'elle pourrait engendrer lors de la désignation d'un objet ou d'un lieu par le geste ou le clic. Ces informations seront modélisées par des variables réelles.

Nous terminerons ce mémoire par une conclusion et une présentation de nos perspectives.

Chapitre I: Etat de l'art et cahier des charges

Nous présentons dans ce chapitre un état de l'art non exhaustif sur les dispositifs de robot manipulateur mobile ainsi que leurs commandes. Nous présenterons aussi un ensemble de travaux relatif aux systèmes multimodaux et la prise en charge du contexte. En s'inspirant de l'ensemble de ces travaux nous établirons le cahier des charges auquel doit répondre le système mis en œuvre pour atteindre l'objectif de ce travail de magister.

I.1 Etat de l'art

I.1.1 Travaux relatifs aux fauteuils roulants munis de bras manipulateur

Les premiers travaux de recherche effectués dans le domaine des fauteuils roulants munis de bras manipulateurs ont débuté à la fin des années 80 lors de l'intégration d'un bras manipulateur MANUS [1] sur un fauteuil roulant. Ce bras à six degrés de liberté est doté d'une pince à son extrémité. La commande des mouvements s'effectue par l'intermédiaire d'interfaces et de procédures modulaires, individuellement adaptables à l'opérateur. De plus, pour des opérations dans un environnement humain non-structuré la commande interactive est privilégiée. Dans cette dernière, l'utilisateur commande directement les mouvements de la pince en s'appuyant sur ses propres capacités de perception et de planification des mouvements. Une autre approche de commande a été introduite en 2004. Elle consiste en l'intégration de deux webcaméras installées sur la pince ce qui a permis d'obtenir une interface graphique et l'intégration d'une commande référencée vision. Ainsi en utilisant les informations visuelles, le robot peut aller saisir de manière autonome un objet sélectionné à l'écran [2].

D'autres dispositifs similaires existent ; nous pouvons citer à titre d'exemple [2]:

- Le robot RAPTOR développé par Mahoney et commercialisé outre Atlantique depuis 2000. C'est le premier robot approuvé aux USA. Tout comme le robot Manus, RAPTOR utilise une technologie de pointe issue de la NASA et de la station spatiale internationale.

- Développé en 2005 par Alqasemi, McCaffrey, Edwards, et Dubey, WRMA est un robot manipulateur qui comporte à son extrémité une pince tri digitale. Il peut être soit embarqué sur le fauteuil roulant ou utilisé en base fixe.

- De conception coréenne, développé par Chang, Park, Jung, et Jeon, en 2001, le robot WAM (Whole Arm Manipulator) est actionné par une transmission à câble.

L'utilisation d'un fauteuil roulant peut se révéler être une tâche très difficile pour certains handicapés. En effet, les tremblements et les capacités physiques très réduites de certains patients ne font qu'augmenter leur invalidité. Afin de prendre en compte l'état réel du patient et les services demandés, l'architecture de commande proposée doit être modulaire et reconfigurable pour permettre une adaptation spécifique à chaque personne et à l'environnement où il évolue. Les recherches effectuées dans ce domaine sont récentes et constituent un sujet très convoité par les chercheurs. On peut citer à titre d'exemple les travaux suivants:

- **Ubibo:** Développé en 2004 par « Robot Intelligence Laboratory KAIST » est basé sur la technologie robotique et le concept de l'informatique omniprésente (ubiquitaire). Ce robot de 3ème génération peut fournir plusieurs services par un dispositif à travers n'importe quel réseau, à tout endroit et à tout moment dans un espace ubiquitaire. Il est présenté comme étant un robot intégrant trois formes de robots [3]:
    - Robot logiciel (Sobot). C'est un robot virtuel, qui a la capacité de se déplacer à n'importe quel endroit grâce à un réseau.
    - Robot embarqué (Embot). Il est incorporé au sein de l'environnement ou dans le Mobot.
    - Robot mobile (Mobot). Il fournit des services mobiles et une sensibilité au contexte.
- Le projet VAHM (**V**éhicule **A**utonome pour **H**andicapés **M**oteurs): Ce projet rentre dans le cadre de la robotique mobile appliqué à l'aide du déplacement de personnes handicapées. Ce dernier consiste en l'élaboration d'une architecture logicielle en utilisant des systèmes multi agents (SMA) qui coopèrent pour mener à bien une tâche et obtenir un système intelligent qui peut évoluer dans un environnement et faire face au changement non planifié [4].

I.1.2 Travaux relatif à l'application des réseaux de Pétri dans le domaine de la robotique

Les applications dans le domaine de la robotique mobile qui utilisent les réseaux de pétri sont jusqu'à présent limitées. Néanmoins, on peut citer comme exemple les travaux effectués au Laboratoire SIS/AI, Université de Toulon-Var, qui a pour titre: La modélisation par réseaux de Pétri à flux de données pour une programmation VHDL conçu pour une application en robotique mobile d'assistance aux handicapés [5].

D'autres études ont été réalisées sur l'interaction homme robot comme dans [6] et [7]. Les auteurs montrent dans ces deux articles la nécessité d'avoir une bonne interaction entre les robots et les humains pour pouvoir espérer une bonne coexistence.

I.1.3 Etat de l'art autour des systèmes multimodaux

Nous savons tous que le robot reste une machine et que la communication avec une machine est une communication rigide. Elle nécessite la connexion d'outils pour pouvoir interagir avec le robot, comme un clavier ou une souris. Cette façon de communiquer peut parfois être difficile pour certaines personnes surtout lorsqu'elles sont invalides. De ce fait, les chercheurs se sont orientés ces dernières années vers l'introduction de moyens plus naturels dans cette interaction comme la parole, le geste, la capture du regard… etc. Cette introduction se présente sous la forme de conception de systèmes multimodaux pour la commande de robot. L'intérêt majeur de ces applications est de faciliter la communication entre l'homme et le robot en la rendant presque similaire à une conversation entre êtres humains.

L'un des premiers travaux effectués dans ce domaine est le « pose ça ici » introduit par Bolt en 1980 [8], [9]. Il permet à l'utilisateur de manipuler des objets sur un écran 2D par le biais de la voix et du geste. Aussi, le QuickSet est l'un des premiers systèmes qui intègre la parole et le geste comme entrées. Le premier prototype de ce système a été développé en 1994 et permet la création et le contrôle de simulations militaires. Une comparaison entre ce système et l'utilisation d'une interface graphique simple est présentée dans [10]. Cette étude montre la rapidité obtenue par une interaction multimodale et la satisfaction des utilisateurs par rapport à l'utilisation de ce type d'interaction.

D'autre part, nous pouvons citer comme travaux la conception d'une interface multimodale par le *Naval Research Laboratory (Washington)* en 2001 [11], ou encore la conception d'une architecture logicielle dynamique dédiée aux applications multimodales dans le cadre de l'obtention d'un doctorat en informatique de l'université de Versailles-Saint-Quentin (France) [8].

Plus récemment, nous pouvons citer le travail de Hina Manolo [12] qui porte sur le paradigme d'un système multimodal multimédia ubiquitaire sensible au contexte d'interaction. De plus, l'intégration de la parole dans le cadre de l'apprentissage du robot dans les travaux de Lallee et Co [13] montre l'intérêt à utiliser ce genre de modalités pour une meilleure communication entre le robot et l'être humain. En effet, ces travaux consistent en une collaboration entre l'humain et le robot pour la construction d'une table en utilisant la parole comme outil de communication.

Nous pouvons aussi citer comme exemple d'application dans ce domaine les travaux effectués dans [14], qui consistent en l'utilisation de la parole dans une application multimodale. En effet, EMMA est un langage XML qui fournit un mécanisme de capture et d'annotation de différentes étapes de traitement des entrées utilisateurs. Aussi, dans [15], l'auteur nous présente différents moyens de fusion et fission multimodal dans l'intégration d'information relative à une relation interactive entre l'homme et la machine. En effet, les travaux présentés dans cet article donnent une nouvelle méthode de fusion. Cette nouvelle approche n'utilise plus la temporalité et la sémantique, mais préfère distinguer plusieurs sous processus pour la fusion multimodale: « multimodal coordination », « multimodal content fusion », et « multimodal event fusion ». Aussi, d'autres langages ont été conçus pour assurer une bonne interaction multimodale entre l'homme et la machine, comme c'est le cas dans [16] où l'on trouve une présentation du langage MultiML. Ce langage est un langage pour l'annotation multimodale des déclarations de l'homme. MultiML est capable de représenter les entrées de plusieurs modalités ainsi que leurs relations. L'article nous montre comment la parole et le geste sont pris en charge et décrit par MultiML.

I.1.4 Travaux relatifs à la prise en compte du contexte

L'interaction multimodale permet une meilleure communication entre l'homme et la machine. Néanmoins, il est nécessaire de prendre en compte le fait que l'utilisateur interagit avec son environnement (les objets, les humains, les conditions climatiques,

…etc.). De ce fait, les applications dans ce domaine doivent être capables de prendre en charge le contexte et de l'intégrer dans la conception de l'architecture de commande.

Plusieurs travaux ont été réalisés dans ce domaine, on peut citer les travaux de Daniel Salber, qui s'est intéressé dans son travail [17] à la présentation du contexte et des applications sensibles au contexte. Il a introduit un modèle implémenté dans le Contexte Toolkit et développé dans le but d'aider à traiter les informations issues du contexte. Puis il réalise une analyse entre les systèmes multimodaux et les systèmes sensibles au contexte et a conclu qu'en effet les applications sensibles au contexte sont une extension des systèmes multimodaux par le fait qu'ils permettent de prendre en charge un plus grand nombre de modalités différentes.

Nous pouvons aussi citer le modèle d'architecture logicielle « PAC-Amodeus» présenté dans [18] qui est une application qui prend en compte le contexte physique de l'utilisateur dans la conception logicielle de systèmes interactifs pour des applications multimodales.

D'autre part, nous trouvons dans [19] une présentation du contexte en informatique diffuse et celle du contexte en intelligence artificielle. La différence entre les deux, réside dans le fait que le contexte dans l'informatique diffuse est formé sur les besoins d'une application particulière, tandis que dans l'intelligence artificielle le contexte est déterminé par la source de l'information. Ensuite, les auteurs donnent une présentation du modèle qu'ils ont utilisé pour la représentation des informations du contexte utilisé dans l'informatique diffuse et qui sont basées sur les langages du web sémantique.

I.2 Cahier des charges du travail demandé:

Le travail présenté dans ce mémoire de magistère rentre dans le cadre de la robotique médicale. Il consiste en la conception et la mise en œuvre d'une plate-forme d'aide aux handicapés moteurs. Le système proposé est une interface multimodale qui va permettre le dialogue entre l'utilisateur et le robot. En effet, ce système doit être capable de prendre en compte l'état du contexte et répondre à des demandes d'un patient, émises via la parole, le clic de la souris d'un ordinateur, ou le geste. Le système robotique utilisé lors de notre étude est composé d'un fauteuil roulant commandé par la parole et d'un bras manipulateur munis d'une caméra relié à un écran dédié à la commande du système. La commande du dispositif pourra se faire:

- Soit par la parole seule
- Soit par la parole et la désignation de l'endroit par la souris ou le geste.

Le bras devra être commandé par la parole suivi par une bonne désignation du clic de la souris ou geste, dans le cas contraire le système répondra par une absence de localisation (désignation).

De plus, l'interface proposée doit permettre:

- Le déplacement du fauteuil roulant par exemple par une commande vocale.
- L'arrêt du fauteuil lorsque le patient le désire ou que le patient a des problèmes de santé détectée par des capteurs, ainsi que le redémarrage.
- La réception de la commande vocale.
- D'accéder à des objets via le bras manipulateur en cliquant sur l'objet sélectionné sur une image ou en le désignant par le geste.

En effet, nous nous intéressons dans ce mémoire à la conception d'une plate-forme composée d'une architecture multimodale intelligente et d'un ensemble de services ubiquitaires adaptatifs. Cette architecture multimodale sera modélisée à base de Réseaux de Petri Colorés Temporisés Stochastiques (RPCTS). Cette plate-forme sera déployée sur un robot d'assistance à l'être humain. Nous appellerons ce robot: Ubiquitous Robotic Companion (URC). Ce robot se veut autonome et capable d'adapter les services aux profils, préférences et contextes des utilisateurs. Il devra être capable de fournir une multitude de services via le réseau n'importe où et n'importe quand. La plate-forme globale sera composée de quatre éléments:

- Le contexte.
- La composition du service.
- La base de connaissance
- L'ontologie du domaine.

Ceci dit, nous nous intéresserons dans notre travail à la conception d'une architecture qui prendra en charge le contexte, les trois autres éléments de la plate-forme seront traités dans un travail ultérieur. De ce fait, nous avons choisi d'utiliser les informations relatives au contexte utilisateur, et au contexte environnement (Figure 1).

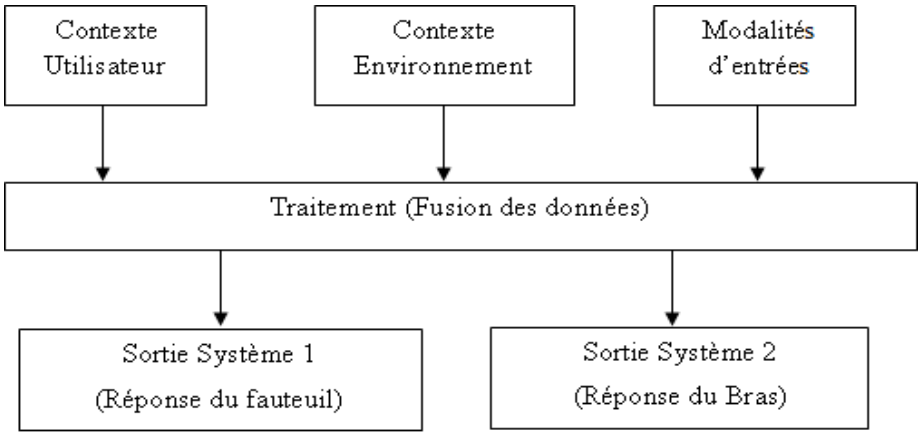

Figure 1: Représentation globale du système

- Le contexte utilisateur représente la partie qui prend en charge les informations relatives à l'état du patient, et qui sont sa température corporelle et sa tension artérielle.

- Le contexte environnement permet la saisie et le traitement des informations reçues des différents capteurs disposés sur le robot et qui permettent par exemple de donner des informations sur la température ambiante, la position du patient,… etc. Dans notre étude nous ne prendrons en charge que l'information relative à l'emplacement du patient et cela grâce à un GPS.

Les modalités d'entrées utilisées dans notre système sont:

- La parole qui permet la commande vocale du robot. Le vocabulaire, la grammaire et les phrases que l'utilisateur peut exprimer pour dicter des commandes à l'application sont les suivantes: Avance, stop, arrière, droite, gauche, prend et pose.
- Le geste ou le clic qui vont permettre de commander le robot. Cela va permettre de commander les mouvements du bras par la parole en précisant l'emplacement par un simple clic de la souris sur un écran d'ordinateur ou par une détection de geste.

La fusion des informations relatives au contexte et ceux reçues par les modalités vont permettre de commander le fauteuil selon le choix de l'utilisateur en prenant compte l'état de l'environnement pour assurer un guidage en toute sécurité.

Nous avons présenté dans ce chapitre un état de l'art relatif à nos travaux de conception d'un système multimodal qui prend en compte le contexte dans le cadre d'une commande d'un robot mobile muni d'un bras manipulateur.

Nous nous sommes tout d'abord orientés vers l'analyse des travaux relatifs aux robots mobiles munis de bras manipulateur ainsi que leurs différents systèmes de commande. Puis, nous nous sommes orientés vers l'étude des systèmes multimodaux existants. Cela nous a permis de voir l'apport de ces derniers dans la communication entre l'homme et la machine en la rendant similaire à une communication interpersonnelle. De plus, vu que la prise en compte du contexte est devenue une nécessité ces dernières années dans la conception de systèmes de commande, nous avons donné dans ce chapitre une présentation de différents travaux de recherches portant sur le contexte.

De notre étude bibliographique nous remarquons que les systèmes cités, sont conçus pour des applications diverses mais ne rentrent pas encore dans le cadre de l'aide à la personne dépendante. D'où l'intérêt de notre travail qui consiste à la conception d'un système multimodal qui prend en compte le contexte pour la commande d'un fauteuil roulant muni d'un bras manipulateur. En s'inspirant des travaux existants nous avons établis le cahier des charges auquel devra répondre le système mis en œuvre. Nous présenterons dans le chapitre suivant des notions sur les systèmes multimodaux et le contexte

Chapitre II:
Notions sur les systèmes multimodaux et le contexte

Une interaction multimodale est une interaction homme-machine qui combine plusieurs modalités en entrée. Dans notre vie de tous les jours, nous utilisons différents moyens de communication tels que la vision, le toucher, le geste … etc. Aussi, les modalités généralement utilisés pour une interaction homme-machine sont la souris et le clavier. La multimodalité a permis de rajouter d'autres moyens de communication avec la machine qui la rapproche d'une communication entre êtres humains, comme: la parole, le geste manuel, le mouvement de la tête, le toucher… etc. Ce nouveau concept donne à l'utilisateur une sensation de communication naturelle avec sa machine comme celle qu'il a l'habitude d'utiliser. En effet, l'utilisateur peut par exemple parler avec une machine et en même temps montrer un endroit avec sa main. Cette nouvelle manière d'interagir avec la machine offre une facilité d'utilisation et une simplicité accessible à tout type d'utilisateur.

D'autre part, l'homme étant continuellement en contact avec son environnement, il est nécessaire de prendre en compte l'état de ce dernier lors de l'interaction avec une machine. En effet, si un utilisateur interagit avec une machine dans un environnement bruité ou avec une présence d'autres personnes, cette interaction ne donnera pas forcément les mêmes résultats que ceux obtenus dans un environnement calme. Pour ce faire, il est nécessaire d'intégrer dans la conception de systèmes multimodaux la prise en compte du contexte. Cette prise en compte va permettre de concevoir des systèmes capables d'offrir à l'utilisateur une facilité d'interaction avec la machine grâce à la multimodalité tout en ayant une idée sur un ensemble d'informations sur des entités particulières.

II.1 **Les systèmes multimodaux**:

D'après l'encyclopédie Encarta, un système multimodal utilise une informatique dont les propriétés interactives permettent de faciliter les échanges entre l'outil informatique et l'utilisateur par des interfaces multimodales qui permettent à la machine de reconnaître la voix ou l'écriture de l'utilisateur …etc. Aussi, la multimodalité permet à plusieurs médias de coopérer entre eux et de faciliter la communication homme-machine en la rendant plus naturelle.

II.2 **L'intégration multimodale** :

Pour avoir un système multimodal, il est nécessaire:

- D'intégrer différentes modalités,
- De procéder à la mise en relation et la combinaison des différents types d'informations reçues par les modalités

D'après Djenidi 2007 [8], trois approches peuvent nous aider à réaliser une intégration correcte de modalités:

- Approche de Martin et al.
- Approche de Coutaz et Nigay
- Approche de Bellik

1. L'approche de Martin et al [8]: Cette approche est basée sur deux dimensions: les types et les buts entre les modalités. De ce fait, six types de coopérations possibles sont distingués :
    a. Complémentarité
    b. Redondance
    c. Equivalence
    d. Spécialisation
    e. Concurrence
    f. Transfert

2. Approche de Coutaz et Nigay: «Coutaz et Nigay (Nigay 1993; Nigay 1994; Nigay 2001» caractérisent les systèmes multimodaux à travers trois dimensions qui sont: le niveau d'abstraction du traitement, l'emploi des modalités et le type de fusion.
    a. Le niveau d'abstraction représente un des multiples niveaux de traitement des données par un système particulier. Ces niveaux d'abstraction s'étendent depuis le niveau signal jusqu'au niveau sémantique de l'information.
    b. L'emploi des modalités concerne la disponibilité temporelle des modalités qui peuvent être employées séquentiellement et en parallèle.
    c. Le type de fusion est la possibilité de combiner plusieurs événements en entrée du système. Coutaz et Nigay distinguent trois types de fusions selon le niveau d'abstraction de l'information.

- La fusion sémantique : C'est une combinaison de commandes pour obtenir une autre commande ou fonction. Les commandes prises séparément ont un sens et combinées entre elles donnent un autre sens.
- La fusion syntaxique : C'est une combinaison d'unités d'information pour obtenir une commande ou un effet. Considérées de façon isolée, les unités d'information ne sont pas porteuses de sens.
- La fusion lexicale : C'est une combinaison d'actions physiques pour obtenir des unités d'information au niveau signal.

3. Approche de Bellik: Bellik (Bellik 1995) a affiné et complété la classification présenté dans l'approche de Coutaz et Nigay en distinguant des critères de fusion selon trois paramètres:
   a. La production des énoncés qui indique si les énoncés (en entrée ou en sortie) doivent être produits séquentiellement ou s'il est possible que plusieurs énoncés indépendants soient produits en parallèle;
   b. L'usage des média qui indique si l'usage des média doit être exclusif, c'est à dire qu'à un instant donné un seul média peut être utilisé ou si au contraire, il est possible d'en utiliser plusieurs simultanément;
   c. Le nombre de média par énoncé qui indique si lors de la production d'un énoncé, il faut utiliser un seul média ou s'il est possible d'en utiliser plusieurs.

D'autres travaux se sont orientés vers la fusion multimodale comme c'est le cas dans [20], où l'on trouve un exemple de fusion multimodale pour faciliter l'accès aux services web.

II.3 **Le contexte :**

II.3.1 **Définition du contexte** :

Nous trouvons dans la littérature, différentes définitions sur le contexte, comme c'est le cas dans [17], [21], [22] et [23]. Nous citons celle de Daniel Salber [17] et qui est:

> « Nous définissons le contexte comme toute information sur l'environnement qui se rapportent à l'interaction entre l'utilisateur et l'application, et qui peut être détecté par les applications. Comme un classement préliminaire des informations du contexte, nous avons identifié les entités suivantes qui fournissent des informations sur le contexte,

1. Fournisseurs d''information du contexte: les gens, les lieux, objets physiques, objets informatiques.
2. Attributs du contexte: emplacement, l'identité, l'activité, l'état.

> Par exemple, l'emplacement et l'identité de l'utilisateur (personnes) sont utilisés par une application de guide d'un musée par le guide lors d'un tour de présentation dans un musée. Ou encore, un système de surveillance d'un bureau permet et assure un suivi des réunions et activités présentes dans différents emplacement (bureau) dans la société. »

Aussi, nous trouvons dans les travaux d'Anders Kofod-Petersen et Marius Mikalsen [24] la définition suivante :

> « Le contexte est l'ensemble des états et paramètres de l'environnement qui concernent l'utilisateur, et qui sont pertinents pour une application dédiée à offrir des services et informations à un l'utilisateur »

De ce fait, nous pouvons dire que le contexte est un ensemble d'informations relatives à l'état du patient, à son environnement, à l'état du système… etc. Les informations du contexte dépendront de ce que l'application aura pour but. En effet, il est très difficile de pouvoir définir toutes les informations qui entourent l'utilisateur et de pouvoir toutes les prendre en charge. Par conséquent, lors de l'élaboration d'une architecture de commande qui prend en compte le contexte il est nécessaire de bien choisir les entités que l'on veut prendre en charge et de les définir.

II.3.2 **Le contexte et la multimodalité :**

Les systèmes multimodaux peuvent être étendus à des systèmes sensibles au contexte sans aucune difficulté. En effet, la multiplication des moyens de communications entre l'utilisateur et la machine permet de prendre en charge plusieurs facteurs relatifs à cet utilisateur. La prise en compte des informations entourant ce même utilisateur rend les systèmes multimodaux encore plus efficaces. Le contexte étant un ensemble d'information relative à un utilisateur, on peut en conclure que des différents types de contexte peuvent être pris en charge. Nous trouvons dans [24] une définition sur ces différents types de contexte. En effet, le contexte peut être divisé en cinq sous- catégories :

1. Le contexte environnemental: Il prend en charge l'entourage de l'utilisateur (objet, personne, la luminosité…etc.)

2. Le contexte personnel: Il décrit les informations relatives à l'aspect mental et physique de l'utilisateur.
3. Le contexte social: Il prend en charge l'état social de l'utilisateur, comme ces collègues, ces amis, …etc.)
4. Le contexte de l'action: Il décrit ce que l'utilisateur est en train de faire, ses activités, son objectif, les tâches en cours, …etc.
5. Le contexte spatio-temporel: Cette partie du contexte est concerné par des attributs tels que: le temps, la position, les mouvements, …etc.

Compte tenu de cette catégorisation du contexte, nous pouvons conclure que la prise en compte du contexte dans des systèmes multimodaux nécessite une fusion des données d'entrées selon le choix des informations pris en compte.

II.4 **Conception d'un dialogue multimodal**:

II.4.1 **Méthode de représentation d'événements multimodaux** :

La représentation des événements multimodaux est caractérisée dans la littérature par différentes méthodes formelles et informelles. Nous trouvons dans les travaux de Djenidi(2007) [8], qu'il existe plusieurs méthodes, parmi elles nous pouvons citer:

1- **Types Feature Structures**: Cette approche permet une transformation des événements multimodaux en structures typées représentant les contributions sémantiques des entrées sans prendre en compte l'aspect temporel. Puis, ces structures vont être combinées par des opérations d'unifications.
2- **Représentation syntaxique**: Cette approche permet une représentation des modalités d'entrée par un triplet (verbe, objet, lieu). Cette représentation est suffisante pour un système qui a comme modalité d'entrée la parole et le geste. Par contre, elle ne l'est plus lorsqu'on désire rajouter d'autres modalités en entrées.
3- **Structure d'actions partielles**: Dans cette approche, les modalités d'entrées sont traitées séparément puis transformées en structures sémantiques (frames) contenant des labels qui caractérisent les paramètres de commande. L'événement multimodal global en entrée sera déterminé par un algorithme dynamique.

II.4.2 **Formalisme de spécification du dialogue multimodal**:

Les systèmes multimodaux interactifs nécessitent une modélisation des événements en prenant en compte le parallélisme et les contraintes temporelles ainsi que le dynamisme (la prise en compte en temps réel des événements). Pour ce faire, il existe plusieurs techniques formelles de description qui peuvent prendre en charge ces contraintes. Nous retrouvons dans [8] une comparaison entre ces différents formalismes. Nous reprenons du travail de Djenidi [8] le tableau récapitulatif des différences entre ces formalismes. Le tableau 1 donne une synthèse sur les différents outils et formalismes de spécification de systèmes multimodaux.

***Remarque**:* Dans ce tableau, certains formalismes sont représentés sous forme d'acronyme, nous les précisons dans ce qui suit :

- CSP : Communicating Sequential Processes
- CCS : Calculus Communicating Systems
- LOTOS : Language of Temporal Ordering Specification
- UML : Unified Modeling Language
- Z : Méthode de spécification en Z
- RTA : Réseaux de Transition Augmentés
- RPCT : Réseau de Pétri Coloré Temporisé

Propriété et Caractéristiques Permises ou offerte par la spécification		Prise en compte des paramètres temporels pour le dialogue			Langage Formel			Représentation												Graduée
		Dater les événements	Trier l'arrivée des événements	Durée de traitement des événements	Oui	Non	+ logique formelle	Des données échangées dans le dialogue (objets, symbole, messages,…)	Des supports et des structures de données (files d'attentes, ressources, canaux, localités,…) et de leurs répartitions et interconnexions.	Des aspects dynamiques du dialogue				Distribuée :		Graphiques Interactives	Hiérarchiée	Graduée Detaillée
										Asynchronisme de communication	Synchronisme de communication	Paralléliseme Dénotationnel	Paralléliseme Non-entrelacement	Fonctionnelle	Spatiale			
Formalismes pour la spécification de l'interaction des événements multimodaux	Algèbres Des Processus (Xavier 1992)	CSP			×		×	×			×	×						×
		CCS			×		×	×			×	×						×
		LOTOS			×		×	×			×	×				×		×
	UML	Diagrammes de collaboration				×		×	×	×	×	×				×		
		Diagramme de séquence	×	×		×				×	×	×				×		
		Diagrammes d'états transitions				×		×	×	×	×	×				×	×	
		Diagramme d'activité				×				×	×		×			×		
	Z			×		×	×	×	×								×	×
	RTA (Krus 1993)							×	×	×		×						
	RPCT (Jensen 1995)	×	×	×	×		×	×	×	×	×	×	×	×	×	×	×	×
	Structures des caractéristiques typées					×	×	×	×									
	Le langage de spécification des applications multimodales				×		×	×										
	Technique du creuset	×	×	×	×			×	×	×	×	×						
	Structure d'actions partielles				×			×	×									
	Représentations Syntaxiques							×										

Tableau 1: Comparaison des formalismes de spécification du dialogue multimodal [8]

Nous remarquons que l'outil de formalisme le plus approprié pour la modélisation multimodale est le réseau de pétri coloré temporisé (RPCT) [25], [26] [27]. En effet, c'est un langage formel qui:

1. Permet de prendre en charge les paramètres temporels
2. Il peut prendre en compte les aspects dynamiques d'un dialogue
3. Il peut représenter des données échangées dans un dialogue ainsi que des supports et des structures de données
4. Il peut avoir une représentation distribuée, graphique et hiérarchisée

Compte tenu de ces caractéristiques, les systèmes à automates finis (les RPCT) semblent être le formalisme le plus adéquat pour les applications multimodales. En effet, il est bien adapté aux protocoles de communication statique et dynamique. Ceci dit, nous avons choisi d'utiliser dans la suite de notre travail de conception d'une architecture multimodale, les RPCT grâce à l'outil de modélisation CPNTools.

CPNTools est un éditeur de graphe et un simulateur de réseau de Pétri coloré temporisé stochastique [28] [29] [30] [31]. Développé à l'université d'Aarhus (Danemark), il est basé sur une manipulation directe des menus et des boites de dialogue. Il peut produire des réseaux avec plus de mille places, transitions et arc structurés en une centaine de modules ou plus. Aussi, l'outil CPNTools permet d'avoir plusieurs sous réseaux dans la même fenêtre et avoir une liaison entre eux, grâce à la possibilité d'avoir des places spatialement distribuées. Par conséquent, cette caractéristique peut permettre de réaliser des structures de systèmes multi agents en représentant chaque agent séparément et faire une liaison entre eux.

II.5 **L'agent et les systèmes multi-agents**:

L'agent en informatique est une entité qui est doté d'une certaine autonomie, et peut représenter un programme, un processus, une personne…etc. Nous trouvons dans [8], une définition détaillée sur « la situation », « l'autonomie » et « la flexibilité » comme suit:

a. S*ituation*: l'agent est capable d'agir sur son environnement à partir des entrées sensorielles qu'il reçoit de ce même environnement. Exemples: Systèmes de contrôle de processus, systèmes embarqués, etc.;
b. *Autonomie*: l'agent est capable d'agir sans l'intervention d'un tiers (humain ou agent) et contrôle ses propres actions ainsi que son état interne;

c. *Flexibilité*: l'agent dans ce cas est:
   - *capable de répondre à temps:* l'agent doit être capable de percevoir son environnement et d'élaborer une réponse dans les temps requis par l'application;
   - *proactif*: l'agent doit exhiber un comportement proactif et opportuniste. Il doit être capable de prendre la bonne initiative au "bon" moment;
   - *social*: l'agent doit être capable d'interagir avec les autres agents (entités logicielles et humaines) quand la situation l'exige afin de compléter ses tâches ou aider ces agents à accomplir les leurs.

Ces définitions sont relatives aux agents dits « cognitifs » (ou délibératifs), et qui sont capables, suivant la définition préalable de leur environnement, de prendre des décisions et de planifier des actions. Néanmoins, on trouve aussi d'autres types d'agents, comme les agents réactifs qui agissent en fonction de l'état de l'environnement où ils se situent. C'est-à-dire, que n'ayant pas un modèle symbolique de leurs environnement, ces agents réactifs fonctionnent de façon élémentaire et qui est: stimulus-réponse. Aussi, on peut trouver une approche qui combine les agents cognitifs et réactifs et qui est dite hybride. Cette approche est une structuration d'agent en couches (couche bas niveau, couche intermédiaire et couche supérieur).

La combinaison de plusieurs agents donne des systèmes multi agents (SMA). Les agents dans ces systèmes coopèrent et coexistent dans le but de former des systèmes distribués. Ce domaine est celui de l'IAD ou Intelligence Artificielle Distribuée ; « il y a l'idée que plutôt de tenter de résoudre un problème par une seule entité intelligente monolithique, il serait plus simple de faire coopérer plusieurs intelligences entre elles, afin de résoudre le problème par une équipe, ce qui simplifie la construction, la maintenance et l'exécution de ces systèmes complexes » [32].

Aussi, la distribution fournie par les SMA permet d'accomplir des tâches complexes grâces à la coopération des agents. Aussi, ces systèmes permettent de faire une analyse théorique et expérimentale des mécanismes qui se produisent lors de cette interaction. Les SMA permettent aussi une facilité de communication et de coordination d'action. Par conséquent, ces systèmes sont très bien adaptés dans le cadre d'application multimodale. Grâce à leurs propriétés, ils vont permettre de réaliser le dialogue entre la personne et la machine. Ainsi, les RPCTS sont un formalisme qui permettent la modélisation de système

sous forme de graphe, la combinaison des deux (les RPCTS et SMA) va nous permettre de modéliser un système multimodal grâce à une interaction entre agents représentés par des réseaux de Pétri distribués.

Nous avons présenté dans ce chapitre des définitions et propriétés des systèmes multimodaux et du contexte. Aussi, nous avons présenté quelques formalismes cités dans la littérature et qui peuvent permettre la modélisation des systèmes multimodaux. Après comparaison de ces formalismes, nous avons opté pour l'utilisation des réseaux de Pétri coloré temporisé stochastique (RPCTS) compte tenus de leurs différentes propriétés, entre autres leurs capacité de formalisation de la solution et la possibilité de pouvoir réaliser une vérification graphique grâce à l'outil de modélisation CPNTools.

Nous avons donné une idée sur le paradigme agent ainsi que l'intérêt obtenu de leur combinaison dans des systèmes multi agents (SMA). En effet, les SMA sont bien adaptés dans des situations où la communication est complexe et diversifiée, ainsi qu'à des situations nécessitant une communication et une coopération entre plusieurs entités. Aussi, ils permettent une reconfiguration facile par la possibilité d'ajouter des agents, comme par exemple ajouter un média dans un système de communication. Par conséquent, les RPCTS et les SMA sont très bien adaptés à des applications multimodales multi agents qui peuvent prendre en compte le contexte.

Nous nous intéresserons dans le chapitre suivant à la conception d'un système multimodale pour une application robotique.

Chapitre III :
Conception d'un système multimodale pour une application robotique

De nos jours nous observons un intérêt croissant vers la conception de systèmes orientés services, ouvrant vers des domaines applicatifs tels que la maison intelligente, la télématique et les réseaux de nouvelles générations. Les dispositifs mobiles, entités centrales de ces systèmes, sont en pleine évolution, tant en termes de performances, de connectivité, ou d'usages. Ils deviennent de plus en plus omniprésents et nécessaires au confort de vie et aux besoins quotidiens des utilisateurs, à la maison comme à l'extérieur. Dans ces environnements, les applications et les services doivent être intelligents, multi-plateformes, multiservices et multi-usages et être ouverts sur une multitude d'architectures matérielles, de plateformes logicielles, et de réseaux hétérogènes. Ces services appelés omniprésents (ubiquitaires) sont ainsi capables d'accompagner l'utilisateur partout et de lui offrir un ensemble de services quel que soit l'environnement (habitation, lieu publique: gare, hôpital, etc.) et selon différents contextes d'utilisation.

La plate-forme à concevoir, dans le cadre de ce mémoire de magister, doit se baser sur les préférences et les besoins des utilisateurs, ainsi que sur un ensemble de processus qui coordonnent le fonctionnement pour permettre au robot de fournir à un utilisateur, le service le mieux adapté à son profil et à son environnement. L'expérimentation de la plateforme se basera sur le scénario de traitement de l'Handicap. Les besoins en matière de sécurité et d'assistance s'expriment de plus en plus fortement pour les personnes âgées et les handicapés. Dans ce contexte, la plateforme visée devra permettre la mise en œuvre de services accessibles à des personnes dépendantes et surtout adaptés à la diversité des handicapés et des maladies de ces derniers (paralysie, homéopathies, Parkinson, Alzheimer, etc.).

Ceci dit, nous proposons dans ce chapitre une architecture logicielle multimodale que l'on modélisera par un réseau de pétri coloré stochastique. Cette architecture sera dédiée à la commande d'un robot manipulateur d'aide aux personnes dépendantes. Dans le cas de notre application nous considérons comme système robotique un fauteuil roulant muni d'un bras manipulateur.

III.1 **Architecture du système:**

Nous proposons dans cette section une architecture logicielle d'un système multimodale de type multi-agents (SMA), qui prend en compte le contexte. Ce système aura pour entrée plusieurs modalités qui peuvent être: la parole, le geste, la commande tactile, la commande par souris… etc. Ce système doit donner en sortie une réponse à ces entrées en prenant en compte le contexte utilisateur ainsi que le contexte environnement. Le contexte utilisateur prendra en compte les informations qui concerne l'utilisateur comme: sa température, les mouvements de sa tête, sa tension … etc. D'autre part, le contexte environnement prendra en compte les informations relatives à l'environnement où évolue l'utilisateur comme: la température ambiante, le bruit, la localisation de l'utilisateur … etc.

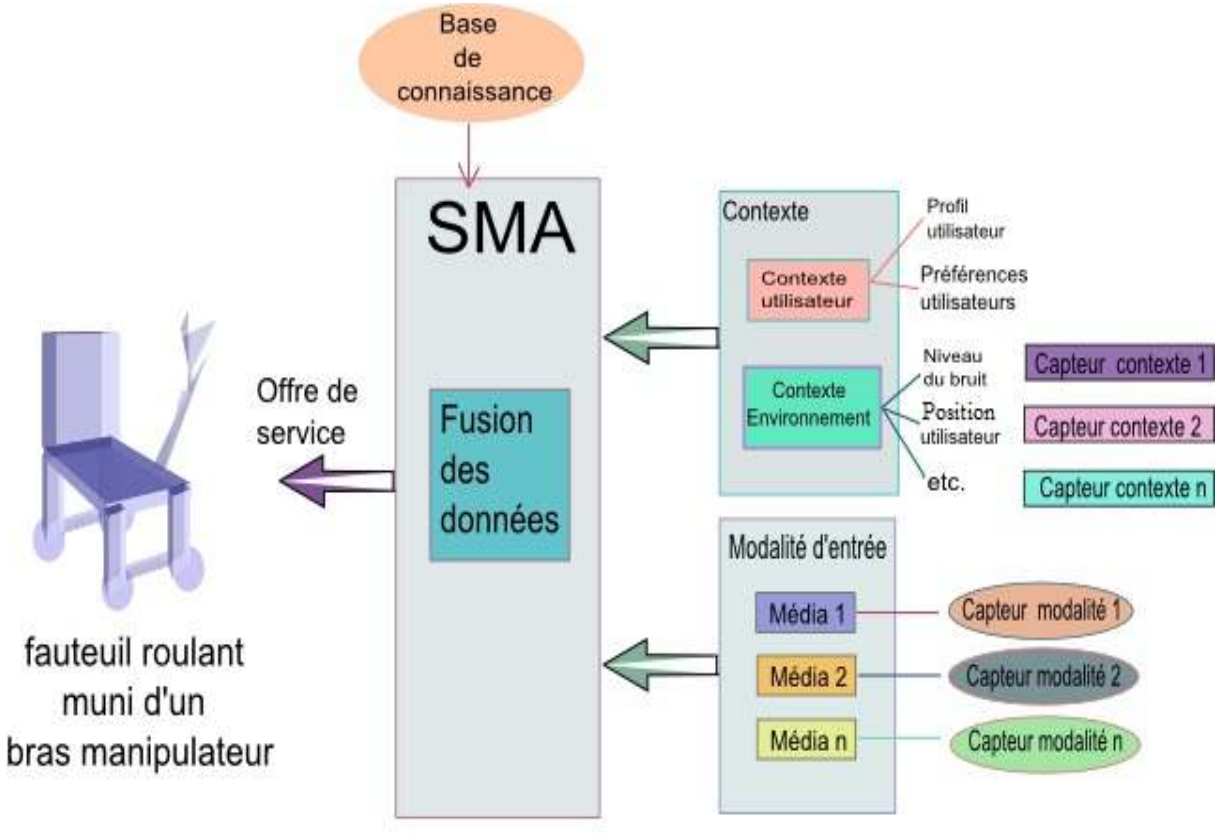

Figure 2: Architecture logicielle multimodale qui prend en compte le contexte

La figure 2 montre l'aspect général de notre système et reflète notre prise en compte du parallélisme dans la conception d'une architecture multimodale. Le système multi-agent (SMA) a pour rôle de fusionner les données issues des capteurs des différentes modalités ainsi que ceux du contexte. Cette fusion permettra de répondre à une requête de l'utilisateur tout en prenant en compte son état ainsi que l'état de son environnement.

Dans la présente étude nous avons choisi de prendre comme modalités d'entrée les médias suivant:

1- **La parole:** Cette modalité offre une facilité d'utilisation. En effet, l'utilisateur pourra communiquer avec le robot d'une façon naturelle avec une rapidité d'interaction.

2- **La commande par la souris:** L'utilisateur pourra par exemple sélectionner un objet sur l'écran de l'ordinateur et le système robotisé répondra par la saisie de cet objet.

3- **Le geste:** L'utilisateur pourra demander d'apporter un objet ou se déplacer vers un endroit par le simple fait de pointer le doigt vers la direction désirée.

Ces modalités seront capturées par différents capteurs spécifiques à chaque modalité on cite:

- Un micro pour la commande vocale.
- Une souris pour la commande par la souris
- Une caméra pour la capture du geste.

Aussi, pour le contexte nous prendrons en compte les cas suivants:

1- **Le contexte utilisateur:** Dans cette partie nous nous intéresserons aux données qui concernent l'utilisateur en tant que personne pour identifier son état physique et réagir en conséquence. Ainsi nous utiliserons:

   - Un capteur de température qui permettra de mesurer sa température
   - Un capteur de tension artérielle.

2- **Le contexte environnement:** Dans cette partie nous nous intéresserons aux données relatives à l'environnement où évolue le patient. Les capteurs utilisés sont:

   - Un GPS qui va permettre de donner la localisation de l'utilisateur.
   - Une caméra qui va afficher sur l'écran les objets à proximité du patient.

Afin de valider cette architecture, nous la modéliserons en utilisant les réseaux de Pétri coloré temporisé stochastique (RPTS). Ainsi nous assurerons une formalisation de la solution et nous exploiterons ces résultats pour une vérification graphique. En effet, ces réseaux

stochastiques ont le mérite de prendre en compte le parallélisme ainsi que les contraintes temporelles. Les réseaux de pétri colorés évoluent par le déplacement des marquages colorés dans ces réseaux. Ces déplacements obéissent à des règles et déterminent quand une transition peut être franchie.

Nous considérons que les entrées issues des modalités représentent l'action demandée et ceux issues des contextes représentent l'observation. Par conséquent:

Si action **ET** observation **alors** réponse du système

Cette réponse se situe dans la fusion des données issues des capteurs d'entrée. Nous devons donc déterminer les différents états des actions demandées et des observations et en déduire les actions qui doivent être fournies en sortie. Dans le cas de notre application, le vocabulaire utilisé dans la commande par la parole est:

- Avance
- Stop
- Gauche
- Droite
- Arrière
- Prend
- Pose

En plus de la modalité parole, nous avons choisi de considérer la commande par la souris ou le geste comme modalités d'entrées, ces dernières vont être représentées par les mots:

- Geste : qui désigne l'information reçue lors de la détection d'un geste
- Clic : qui désigne l'information reçue après un clic de souris

Remarque

> Les informations reçues de ces modalités représentent la localisation de la désignation d'un lieu ou d'un objet.

Pour le contexte nous prendrons en compte les données suivantes:

- **La température du patient:** Tm
- **La tension artérielle du patient:** Sm
- **La localisation du patient:** GPS

Le signal de sortie est déterminé en fusionnant la modalité (l'action demandée) avec les données du contexte. Nous pouvons donc écrire:

$$\text{Sortie} = \text{Modalité f (Tm, Sm)}$$

Nous établissons les hypothèses suivantes:

- Les informations issues des capteurs sont de types binaires. C'est-à-dire que si une donnée est stable le capteur donne **1** sinon il donne **0**.
- Dans le cas où une donnée n'est pas stable, le système mis en œuvre doit ordonner l'arrêt immédiat du robot et envoyer le signal de localisation fournit par le GPS.

Dans notre cas, la simulation par CPNTools nous conduit au codage des entrées (modalités) pour pouvoir les exploiter avec ce logiciel. De ce fait, nous avons défini les codes suivants que nous utiliserons:

Modalités	Code correspondant
Clic	15
Avance	1
Droite	2
Gauche	3
Arrière	4
Prend	5
Pose	6
GPS	7
Stop	0
Tm	(1, 0)
Sm	(1, 0)
Gest	8

Tableau 2: Codage utilisé dans l'approche proposée

Le logiciel CPNTools permet de traiter et de fusionner les données d'entrées pour générer une commande en sortie. Nous utilisons pour la fusion, une fonction mathématique constituée de combinaisons logique des données d'entrées générés aléatoirement par l'outil CPNTools. Ainsi:

Sortie = (parole **Ou** événement) **Et** (Tm **Et** Sm) ;

Le code relatif à la fusion obtenu, est le résultat de la loi suivante :

Code Sortie (Fusion) = (Code parole + Code événement) * (Code Tm) * (Code Sm);

Le tableau 3 donne l'ensemble des combinaisons des entrées prises en compte dans l'architecture proposée, ainsi que les sorties obtenues après l'opération de fusion.

Entrées	Désignation	Codes correspondants	Code de sortie
Avance (Tm, Sm)	Commande « Avance » avec un état stable du patient	1 (1, 1)	1
Avance (Tm, Sm)	Commande « Avance » avec un état instable du patient (problèmes de santé)	1 (1, 0) ou 1 (0, 1) ou 1 (0, 0)	0
(Avance, geste) (Tm, Sm)	Commande « Avance » jusqu'à la désignation Gest avec un état stable du patient	(1, 8) (1, 1)	9
(Avance, geste) (Tm, Sm)	Commande « Avance » jusqu'à la désignation Gest » avec un état instable du patient	(1, 8) (1,0) ou (1, 8) (0,1) ou (1, 8) (0, 0)	0
(Avance, clic) (Tm, Sm)	Commande « Avance » jusqu'à la désignation du Clic avec un état stable du patient	(1, 15) (1, 1)	16
(Avance, clic) (Tm, Sm)	Commande « Avance » jusqu'à la désignation du Clic avec un état instable du patient	(1, 15) (1,0) ou (1, 15) (0,1) ou (1, 15) (0, 0)	0
Droite (Tm, Sm)	Commande « Tourner vers la droite » avec un état stable du patient	2 (1, 1)	2
Droite (Tm, Sm)	Commande « Tourner vers la droite » avec un état instable du patient	2 (1,0) ou 2 (0,1) ou 2 (0, 0)	0
(Droite, geste) (Tm, Sm)	Commande « Tourner vers la droite » jusqu'à la désignation Gest avec un état stable du patient	(2, 8) (1, 1)	10
(Droite, geste) (Tm, Sm)	Commande « Tourner vers la droite » jusqu'à la désignation Gest avec un état instable du patient	(2, 8) (1,0) ou (2, 8) (0,1) ou (2, 8) (0, 0)	0
(Droite, clic) (Tm, Sm)	Commande « Tourner vers la droite » jusqu'à la désignation du Clic avec un état stable du patient	(2, 15) (1, 1)	17
(Droite, clic) (Tm, Sm)	Commande « Tourner vers la droite » jusqu'à la désignation du Clic avec un état instable du patient	(2, 15) (1,0) ou (2, 15) (0,1) ou (2, 15) (0, 0)	0
Gauche (Tm, Sm)	Commande « Tourner vers la gauche » avec un état stable du patient	3 (1, 1)	3

Tableau 3 (1/4): Liste des combinaisons des entrées et sorties obtenues après fusion dans l'architecture proposée

Entrées	Désignation	Codes correspondant	Code de sortie
Gauche (Tm, Sm)	Commande « Tourner vers la gauche » avec un état instable du patient	3 (1,0) ou 3 (0,1) ou 3 (0, 0)	0
(Gauche, geste) (Tm, Sm)	Commande « Tourner vers la gauche » jusqu'à la désignation Gest avec un état stable du patient	(3, 8) (1, 1)	11
(Gauche, geste) (Tm, Sm)	Commande « Tourner vers la gauche » jusqu'à la désignation Gest avec un état instable du patient	(3, 8) (1,0) ou (3,8) (0,1) ou (3, 8) (0, 0)	0
(Gauche, clic) (Tm, Sm)	Commande « Tourner vers la gauche » jusqu'à la désignation du Clic avec un état stable du patient	(3, 15) (1, 1)	18
(Gauche, clic) (Tm, Sm)	Commande « Tourner vers la gauche » jusqu'à la désignation du Clic avec un état instable du patient	(3, 15) (1,0) ou (3,15) (0,1) ou (3, 15) (0, 0)	0
Arrière (Tm, Sm)	Commande « Arrière » avec un état stable du patient	4 (1, 1)	4
Arrière (Tm, Sm)	Commande « Arrière » avec un état instable du patient	4 (1,0) ou 4 (0,1) ou 4 (0, 0)	0
(Arrière, geste) (Tm, Sm)	Commande « Arrière » jusqu'à la désignation Gest avec un état stable du patient	(4, 8) (1, 1)	12
(Arrière, geste) (Tm, Sm)	Commande « Arrière » jusqu'à la désignation Gest avec un état instable du patient	(4,8) (1,0) ou (4, 8) (0,1) ou (4, 8) (0, 0)	0
(Arrière, clic) (Tm, Sm)	Commande « Arrière » jusqu'à la désignation du Clic avec un état stable du patient	(4, 15) (1, 1)	19
(Arrière, clic) (Tm, Sm)	Commande « Arrière » jusqu'à la désignation du Clic avec un état instable du patient	(4,15) (1,0) ou (4, 15) (0,1) ou (4, 15) (0, 0)	0
(Prend, clic) (Tm, Sm)	Commande « Prendre » l'objet désigné par le clic avec un état stable du patient	(5, 15) (1, 1)	20
(Prend, clic) (Tm, Sm)	Commande « Prendre » l'objet désigné par le clic avec un état stable du patient	(5, 15) (1,0) ou (5, 15) (0,1) ou (5, 15) (0, 0)	0

Tableau 3 (2/4): Liste des combinaisons des entrées et sorties obtenues après fusion dans l'architecture proposée

Entrées	Désignation	Codes correspondants	Code de sortie
(Pose, clic) (Tm, Sm)	Commande « Poser » l'objet à l'emplacement désigné par le clic avec un état stable du patient	(6, 15) (1, 1)	21
(Pose, clic) (Tm, Sm)	Commande « Poser » l'objet à l'emplacement désigné par le clic avec un état instable du patient	(6, 15) (1,0) ou (6, 15) (0,1) ou (6, 15) (0, 0)	0
(Prend, geste) (Tm, Sm)	Commande « Prendre » l'objet désigné par le geste avec un état stable du patient	(5, 8) (1,1)	13
(Prend, geste) (Tm, Sm)	Commande « Prendre » l'objet désigné par le geste avec un état instable du patient	(5, 8) (1,0) ou (5, 8) (0,1) ou (5, 8) (0, 0)	0
(Pose, geste) (Tm, Sm)	Commande « Poser » l'objet à l'emplacement désigné par le geste avec un état stable du patient	(6, 8) (1,1)	14
(Pose, geste) (Tm, Sm)	Commande « Poser » l'objet à l'emplacement désigné par le geste avec un état instable du patient	(6, 8) (1,0) ou (6, 8) (0,1) ou (6, 8) (0, 0)	0
(Prend) (Tm, Sm)	Commande « Prendre » un objet sans une désignation par le geste ou le clic avec un état stable du patient → Erreur de localisation	(5,0) (1,1)	5
(Prend) (Tm, Sm)	Commande « Prendre » un objet sans une désignation par le geste ou le clic avec un état instable du patient → Erreur de localisation	(5, 0) (1,0) ou (5, 0) (0,1) ou (5, 0) (0, 0)	0
(Pose) (Tm, Sm)	Commande « Poser » un objet sans une désignation par le geste ou le clic avec un état stable du patient → Erreur de localisation	(6, 0) (1,1)	6
(Pose) (Tm, Sm)	Commande « Poser » un objet sans une désignation on par le geste ou le clic avec un état instable du patient → Erreur de localisation	(6, 0) (1,0) ou (6, 0) (0,1) ou (6, 0) (0, 0)	0

Tableau 3 (3/4): Liste des combinaisons des entrées et sorties obtenues après fusion dans l'architecture proposée

Entrées	Désignation	Codes correspondants	Code de sortie
Stop (Tm, Sm)	Commande « Arrêt » du système avec un état stable du patient	0 (1, 1)	0
Stop (Tm, Sm)	Commande « Arrêt » du système avec un état instable du patient	0 (1,0) ou 0 (0,1) ou 0 (0, 0)	0

Tableau 3 (4/4): Liste des combinaisons des entrées et sorties obtenues après fusion dans l'architecture proposée

Remarque:

> *Le système robotique s'arrête (sortie = 0) si l'utilisateur demande l'arrêt ou si l'un des signaux provenant du contexte utilisateur détecte que le patient a des problèmes de santé. Dans ce cas, le système mis en œuvre, transmet la localisation du patient qui est la donnée GPS.*

III.2 Modélisation de l'architecture par réseau de Pétri:

Le formalisme des réseaux de Pétri est un outil permettant l'étude de systèmes dynamiques et discrets. Les réseaux de Pétri sont une représentation mathématique qui permet la modélisation d'un système par un graphe composé de place, transition et arc [25]. L'analyse d'un réseau de Pétri peut révéler des caractéristiques importantes du système concernant sa structure et son comportement dynamique. Elle permet d'apporter des modifications, si nécessaire, à une architecture proposée. Dans une structure dynamique, nous pouvons modéliser un dialogue multimodal par le fait que ces réseaux apportent une possibilité de parallélisme tout en prenant en compte les contraintes temporelles. Nous pouvons donc assimiler chaque modalité d'entrée comme étant une file d'attente d'un signal correspondant à une information. Par conséquent toutes les entrées multimodales constituent un ensemble de files d'attente parallèles qui caractérise un environnement variable.

En outre, nous allons construire un système multi-agent ou chaque agent prend en compte une ou plusieurs données du système. Le système mis en œuvre observe continuellement l'état des données des différents agents et agit en conséquence. Les états des données d'entrée de chaque agent sont représentés dans une file d'attente.

Dans l'approche proposée, nous modéliserons chaque agent par un réseau de Pétri géré par un ensemble de règles. Les règles déterminent quand est-ce qu'une transition peut être franchie et ainsi changer l'état du système. Chaque place qui contient des marques porte l'information et l'instant d'arrivée de chaque information et peut par conséquent modéliser une file d'attente. L'aspect multi agent permet:

- La modélisation des modalités d'entrée et le contexte par des agents spécifiques distincts
- La fusion des résultats du traitement des différents agents.

Les agents utilisés sont:

- **Agent parole:** Cet agent prend en charge la modalité parole. Il reçoit en entrée les paroles de l'utilisateur et effectue la reconnaissance et l'envoie du code de commande correspondant au mot identifié.
- **Agent GestClic:** Cet agent prend en charge la modalité du geste ou du clic. En effet, ce dernier reçoit en entrée les signaux émis par ces modalités, identifie la modalité utilisée (« geste » ou « clic »), localise la position désigné et transmet cette information à l'agent fusion.
- **Agent TmSm:** Cet agent prend en charge les informations issues du contexte utilisateur qui sont la température du patient et sa tension artérielle. Il vérifie l'état du patient et envoie un signal qui désigne si le patient va bien ou non.
- **Agent fusion:** Cet agent reçoit et fusionne les données issues des trois agents cités ci dessus et envoie en sortie la commande adéquate.

III.3 **Modélisation des agents:**

Dans la structure multimodale proposée chaque agent est modélisé par un réseau de Pétri. Les différentes déclarations employées dans ces RPCTS pour les variables, fonctions, constantes etc..., sont données en annexe1. Les réseaux de Pétri colorés temporisés stochastiques nous permettent de prendre en charge le temps et d'introduire des fonctions aléatoires.

Remarque:

- Sur les figures représentants les réseaux de pétri des différents agents le texte figurant à proximité des places, transitions, et arcs est un script écrit en langage ML (Méta Langage [29]).
- Les inscriptions figurant sur les arcs entrants représentent les données qui doivent exister dans les places d'entrées (le marquage des places) pour que la transition soit franchie.
- Sur certains arcs, le franchissement de la transition est conditionné par une temporisation qui est représentée par exemple par **@+TempsArrive**.
- Une fois la transition franchie, on retrouve sur les arcs de sortie une ou plusieurs marques qui ont été retirées des places d'entrées et qui peuvent subir des modifications de type. Ces modifications sont réalisées par un code écrit en ML à proximité de la transition (par exemple **input ( ) output ( ) action ( )**).
- L'état du système change par l'addition de marquages ou le changement de type apporté au marquage dans les places de sorties.

III.3.1 **L'agent parole** :

L'agent parole est modélisé par un réseau de Pétri qui prend en charge la parole. Ce dernier reçoit en entrée les différents mots émis par l'utilisateur. Nous rappelons que le vocabulaire utilisé est prédéfini dans la partie III.1 de ce chapitre. Cet agent doit pouvoir reconnaître le mot et envoyer un code correspondant pour que le reste de l'architecture réagisse en conséquence. La figure 3 illustre le réseau le modélisant.

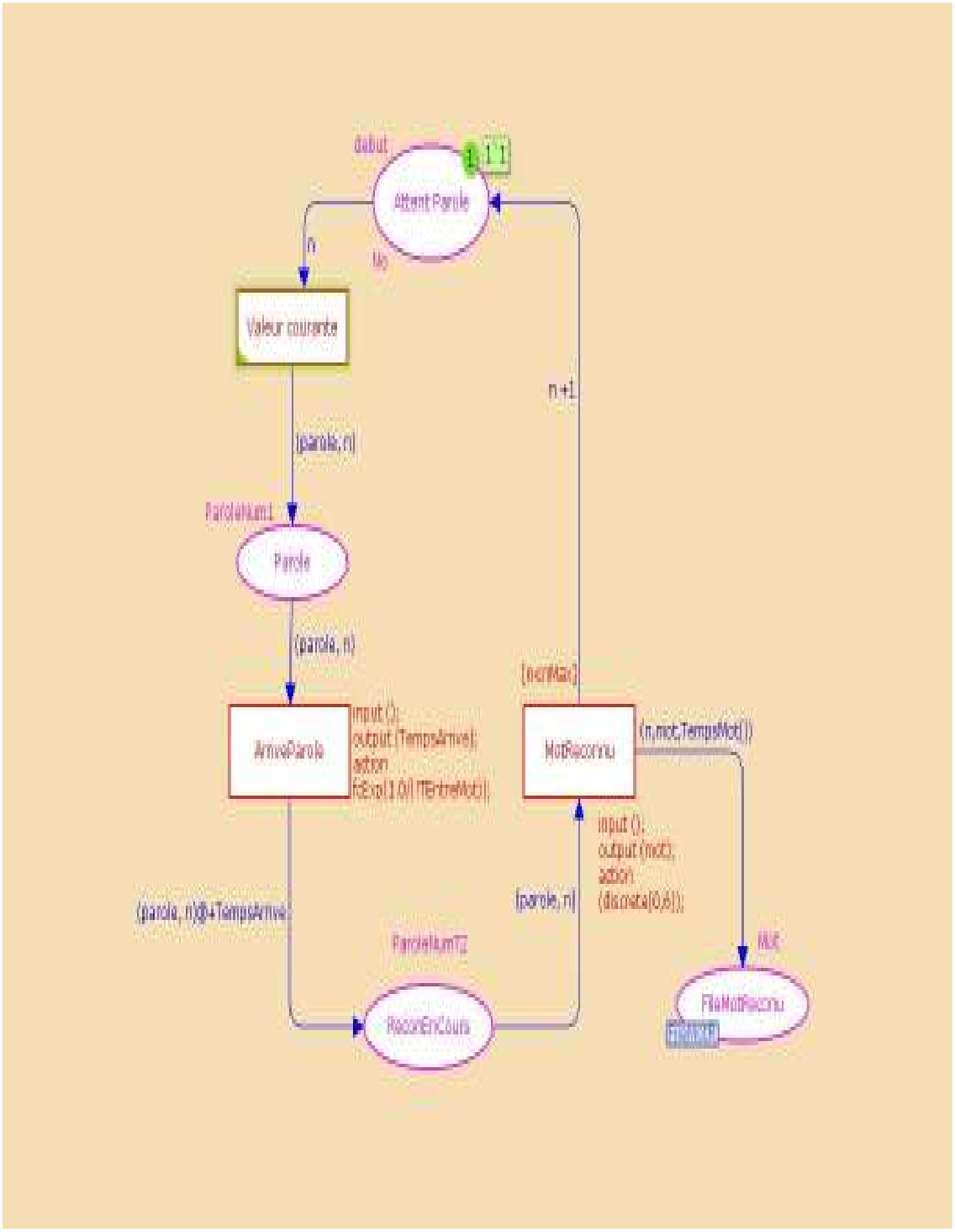

Figure 3: Réseau de Pétri modélisant l'agent parole

L'agent parole attend qu'un mot se présente en entrée où il gère une file d'attente de mots prononcés. Le travail réalisé dans le cadre de ce mémoire de magister est théorique. Les mots sont générés aléatoirement et non récupérées par un logiciel de reconnaissance vocale.

En effet, nous avons choisi de générer les mots aléatoirement et de donner un temps pour l'arrivée de chaque mot. Ceci rend le réseau proposé proche de la réalité sachant qu'un utilisateur est libre de prononcer n'importe quel mot de commande à n'importe quel instant. Les déclarations relatives au texte écrit en CPN-ML sont définies en annexe1. Le texte qui accompagne chaque place définit le type de cette place. Le texte à proximité des transitions constitue soit la condition pour que la transition soit franchie (comme par exemple (n<nMax) ou l'appel d'une fonction comme dans (input ( )…). La place **FileMotReconnu** porte l'étiquette FileMotR. Ceci indique que cette place est distribuée dans le réseau modélisant l'architecture proposée. En d'autres termes, nous retrouverons cette place ailleurs en l'occurrence dans le cas de notre application, dans le réseau modélisant l'agent fusion. Cette fonctionnalité nous permet d'assurer une continuité entre les RPCTS en les distribuant spatialement dans différentes fenêtres. Nous modéliserons donc chaque agent dans une fenêtre distincte et nous établirons les liens entre eux. Ceci permet de rendre les réseaux moins encombrés et plus faciles à comprendre.

III.3.2 **L'agent GestClic** :

Cet agent (figure 4) est modélisé par un réseau de Pétri qui prend en charge la génération aléatoire d'un événement. En effet, nous avons introduit dans ce réseau une fonction aléatoire qui permet aux événements d'arriver à des temps aléatoires comme c'est le cas dans la vie courante, où l'utilisateur peut demander une action via la parole et désigner un endroit en même temps, ou à un instant plus tard.

L'événement est considéré comme étant une modalité d'entrée. Le réseau attend l'arrivée d'un événement ou peut avoir en entrée, une file d'attente d'événement.

Dans notre modélisation, les signaux reçus sont directement la localisation de l'endroit désigné par le geste de l'utilisateur ou le clic d'une souris. Ceci dit, lorsque l'on parle de geste ou du clic on se réfère en réalité à l'endroit où l'utilisateur désire effectuer une action.

La place FileEventReconu est spatialement distribuée dans notre réseau car elle porte l'étiquette FileEvent. Cette place sera retrouvée dans le réseau qui modélise la fusion.

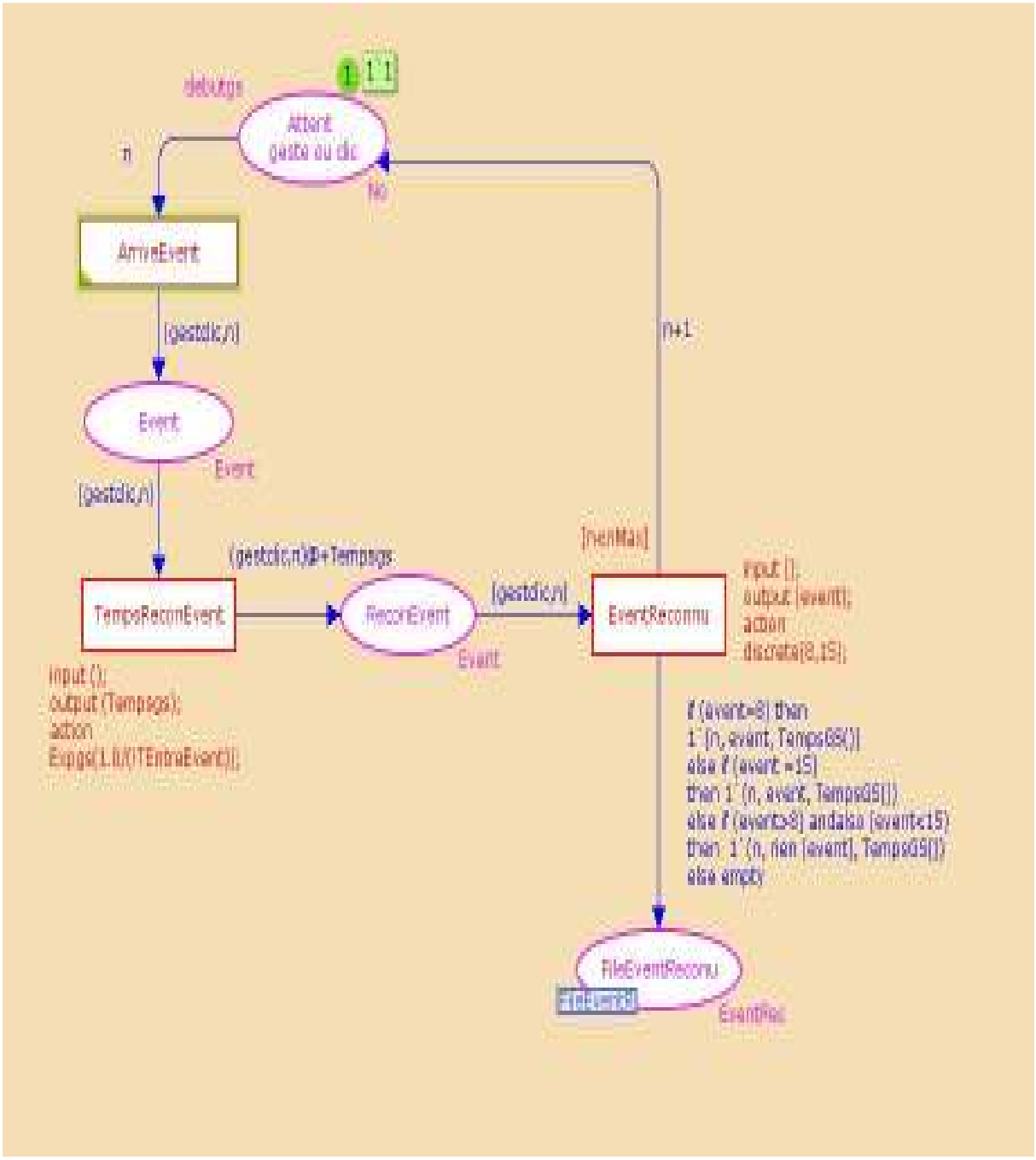

Figure 4 : Réseau de Pétri modélisant l'agent Gestclic

III.3.3 **L'agent TmSm** :

L'architecture de commande mise en œuvre doit permettre de guider le robot en prenant en compte l'état physique du patient et son environnement. Ceci dit, L'agent « TmSm » prend en charge les informations issues du contexte utilisateur et du contexte environnement. Ces informations représentent la température du patient, sa tension artérielle et la localisation par le GPS du fauteuil roulant. Le réseau reçoit continuellement en entrée la

localisation du patient fourni par un GPS. Ainsi cette information pourra être transmise si le système détecte des problèmes de santé chez le patient. Nous considérons que les informations sur la température et sur la tension artérielle sont de type binaire. Si le patient est en bonne santé les capteurs donnent l'information « 1 », sinon ils donnent « 0 ». La figure 5 illustre le RPCTS qui modélise l'agent TmSm.

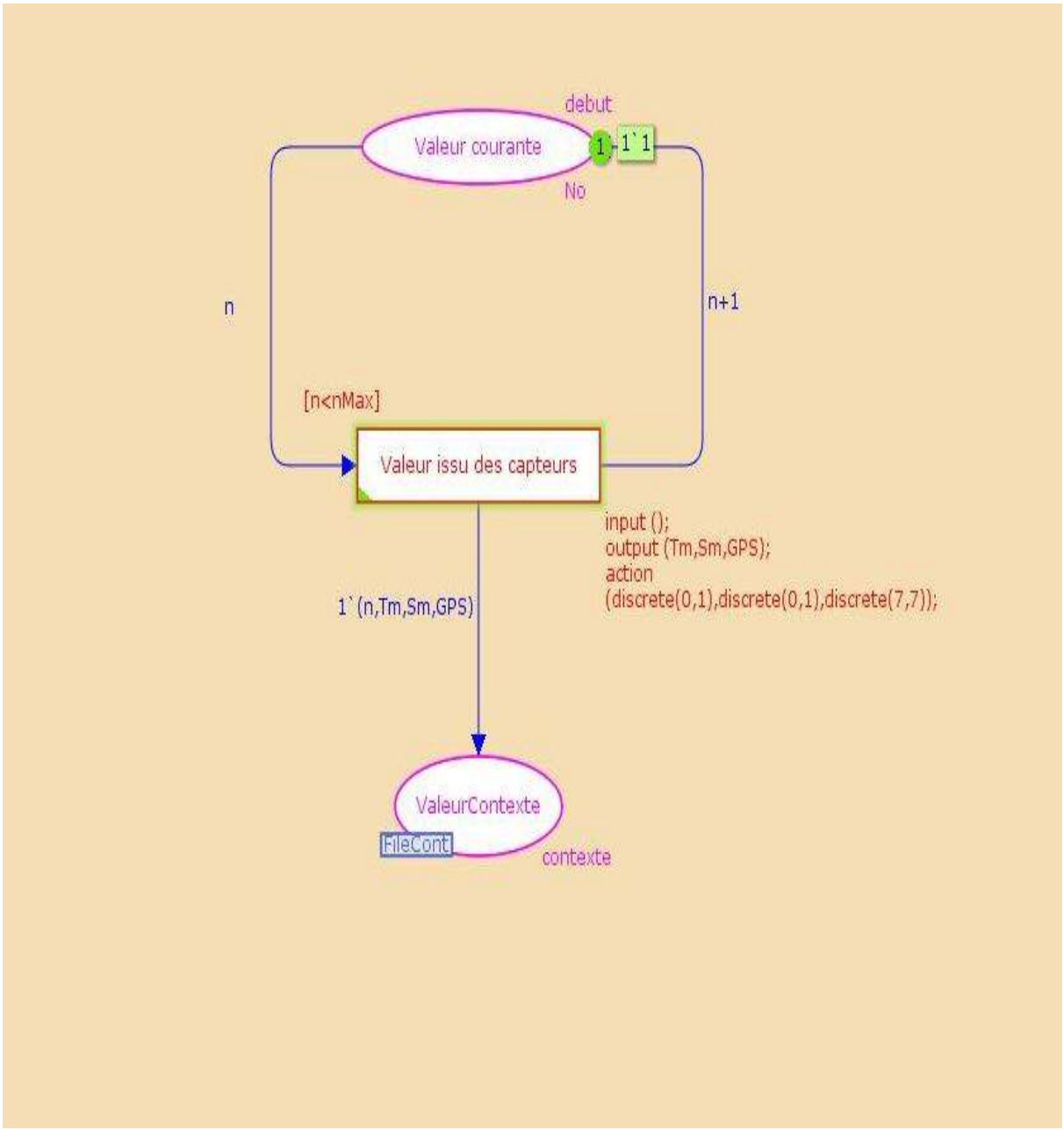

Figure 5 : Réseau de Pétri modélisant l'agent TmSm

Nous retrouverons dans le réseau correspondant à l'agent fusion la place (ValeurContexte) qui porte l'étiquette FileCont.

III.3.4 **L'agent Fusion** :

L'agent fusion combine les informations issues des trois autres agents et donne en sortie la commande en prenant en compte le contexte. En effet, l'agent fusion a en entrée (sous forme d'une file d'attente) les informations issues des sorties des autres agents. Les commandes de sorties obtenues seront le résultat d'une combinaison des commandes vocales en prenant en compte le contexte.

Dans l'architecture proposée nous avons choisi de prendre en considération les commandes par évènement (geste ou clic) que si la commande vocale précède. Ainsi deux variables « Temps1 » et « Temps2 » sont défini. Elles désignent respectivement le temps d'arrivée de la parole et le temps d'arrivée d'un événement. Afin de vérifier si l'événement est bien arrivé après une commande vocale la transition « **Fusion des Données** » a pour condition de franchissement:

[Temps1 < Temps 2]

Dans le cas contraire, nous considérons que l'événement détecté ne correspond pas à une commande et par conséquent il sera ignoré.

Le RPCTS correspondant à ce réseau est représenté dans la figure 6. Nous retrouvons sur cette figure les trois places distribuées spatialement sur les autres RPCTS propre aux agents parole, GestClic et TmSm qui sont: **FileMotReconu**, **FileEventreconu** et **ValeurContexte**. La définition de chaque type de place, des fonctions et code utilisés pour cet agent est donné en annexe1.

A chaque place de sortie de ce réseau nous avons associé un nom correspondant au mot utilisé par la commande vocale. La réponse trouvée en sortie de chaque place dépendra de la condition de l'arc correspondant. Par exemple, si le code généré par la fusion des données des trois autres agents est le 12, la commande de sortie sera donc de faire marche arrière jusqu'à la position indiquée par le geste.

Dans le cas où l'utilisateur a un problème de santé, l'architecture conçue impose un arrêt du système et transmet la position du patient fournie par le GPS. En effet, dans la place

correspondante à la sortie stop, la condition de passage vérifie si le code est « 0 » et l'information correspondante à Tm et Sm.

- Si le code est 0 et que Tm =1 et Sm = 1, cela voudra dire que le patient va bien donc le code = 0 correspond à une demande d'arrêt émise par l'utilisateur et le système répondra par une commande d'arrêt.

- Si le code est égale à 0 et que par exemple Tm = 0 et Sm =1, cela voudra dire que le système a détecté une hausse de température et une bonne tension. Donc le système doit donner l'ordre d'arrêt et envoyer la localisation GPS en précisant la nature du problème détecté. Nous avons modélisé cette commande par un message qui accompagne l'arrêt du système et qui est « **GPS, ProbTm** ».

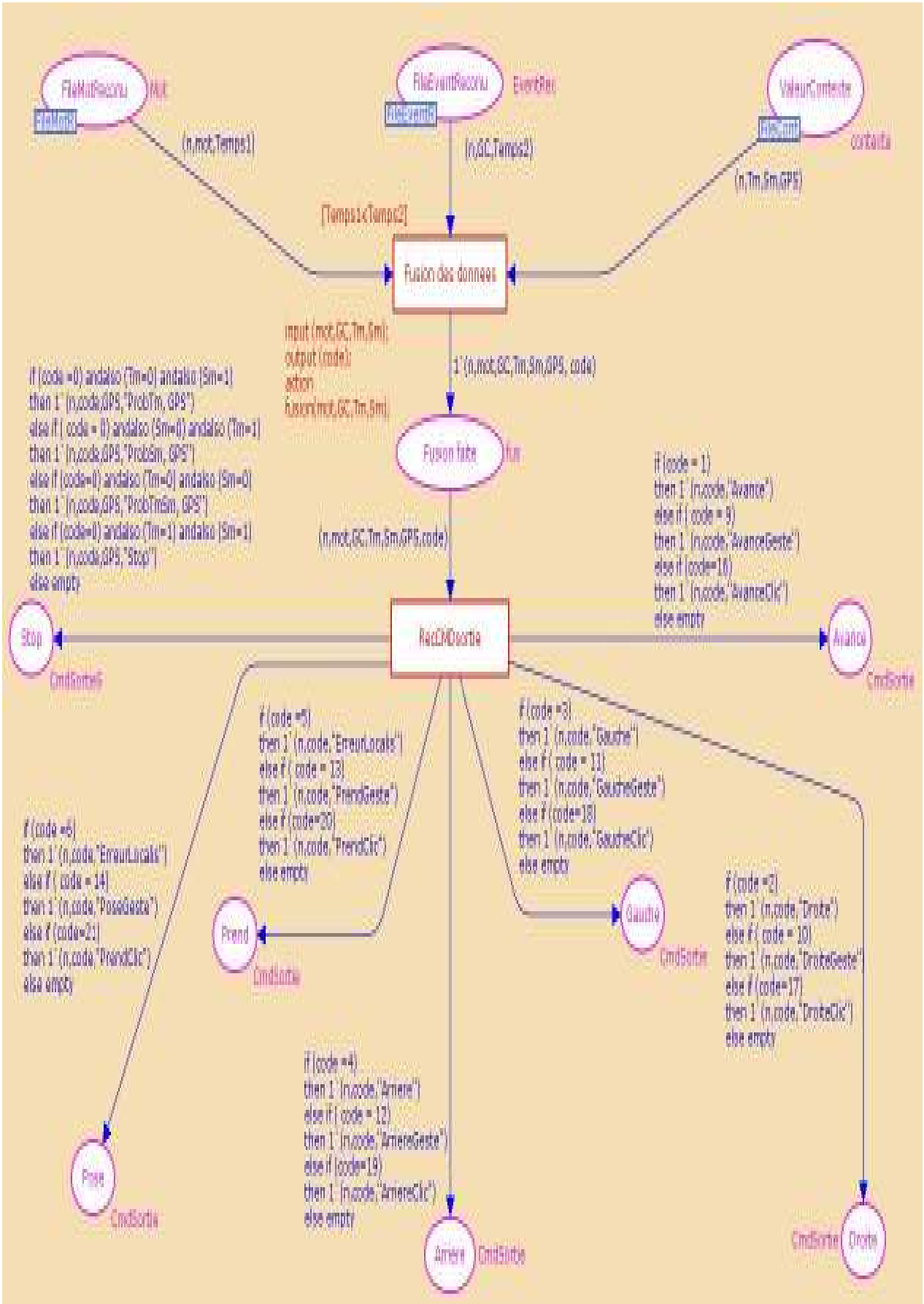

Figure 6: Réseau de Pétri modélisant l'agent fusion

III.4 **Mise en œuvre**:

Afin d'évaluer les performances des architectures à base de RPCTS qui modélisent les agents responsables de la commande du robot d'aide à la personne, nous présentons dans cette section les résultats des simulations obtenues après déroulement de ces programmes.

La figure 7 illustre les résultats lors de la simulation sur CPNTools. Nous remarquons que le premier échantillon ne passe pas par la transition fusion des données. En effet, le temps (Temps 2) d'arrivée du code qui donne l'état de l'événement (geste ou clic) est inférieur au temps (Temps 1) d'arrivée de la commande vocale émise par l'utilisateur (Temps 1>Temps 2). Cette action n'est pas acceptée dans notre approche car nous supposons que l'utilisateur doit toujours émettre une commande vocale en premier. Cette action est donc estimée comme une fausse manœuvre et n'est en conséquence pas prise en compte.

A chaque code de sortie est attribué un message qui correspond à l'action à réalisée. Par exemple, pour l'échantillon « 8 », le message qui l'accompagne est « Gauche ». L'action en sortie sera « tourner à gauche ». Pour ce qui est du message « erreur de localisation » qui accompagne les échantillons « 22 » et « 56 », il indique qu'il y'a eu un problème lors de la désignation ou une absence de désignation d'un événement (geste ou du clic). Dans ce cas l'architecture mise en œuvre ne génère pas la commande du bras manipulateur car la désignation de l'endroit est imprécise ou n'existe pas. Toute fois, elle délivre le message « erreur de localisation ». Nous avons choisi de donner des messages courts et précis. Les messages choisis donnent une idée immédiate sur l'action demandée, comme par exemple « droiteclic » signifie « tourner à droite jusqu'à la position désignée par le clic », et « probTm » signifie « Arrêt du système car un problème de température du patient est détecté ». Nous donnons dans la liste des abréviations plus d'information concernant les différents messages.

La simulation à l'aide de l'outil CPN Tools nous permet de visualiser les différents états du système en indiquant à chaque fois le numéro de l'étape et le temps correspondant. Afin de mieux vérifier les résultats de notre simulation, nous avons récupéré les codes en entrées et en sorties de l'architecture proposée et nous les avons comparé. Nous donnons les codes obtenus dans le tableau 1 de l'annexe 2.

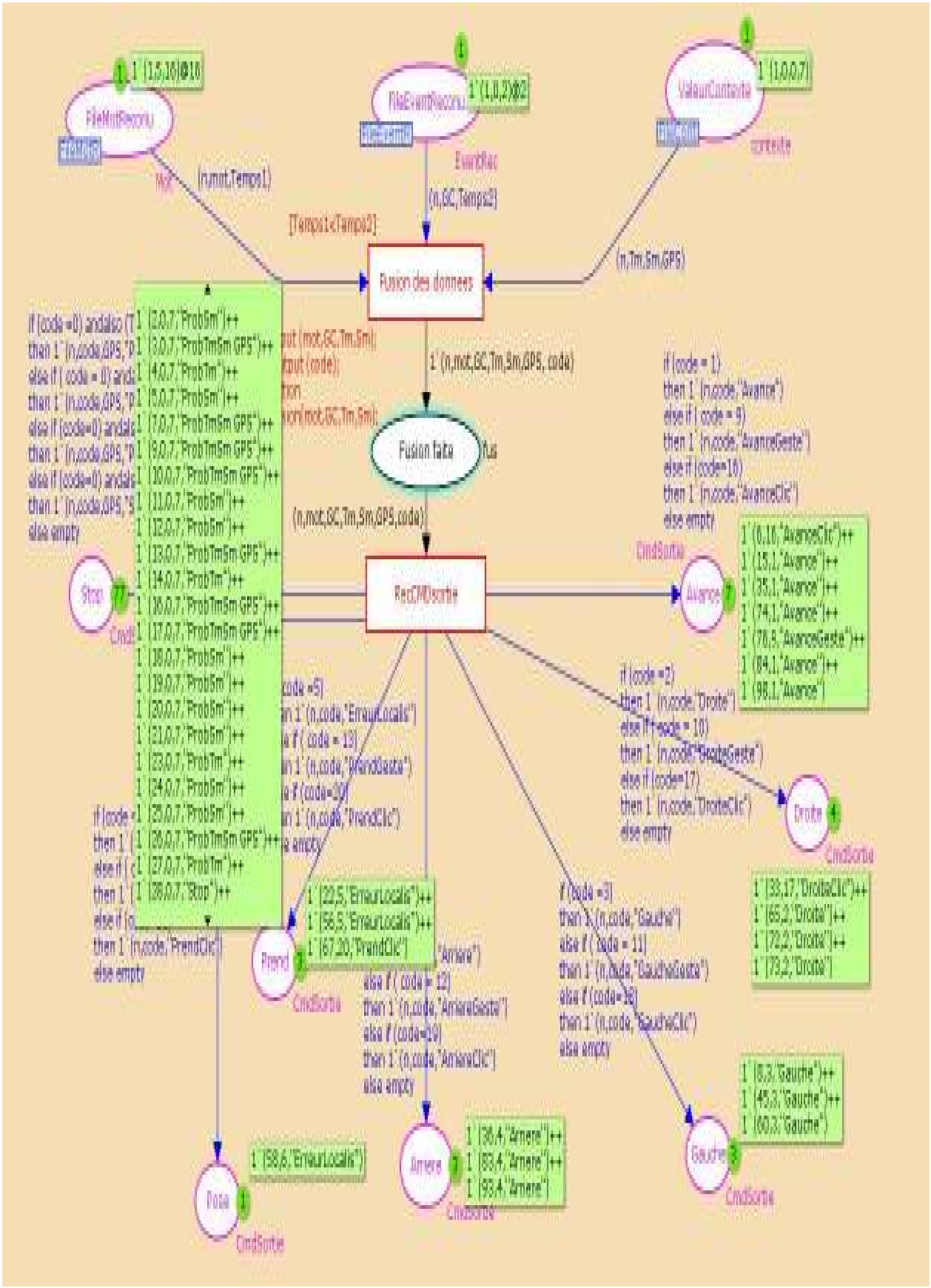

Figure 7 : Résultats obtenus par la simulation sur CPNTools

La simulation à l'aide de l'outil CPN Tools nous permet de visualiser les différents états du système en indiquant à chaque fois le numéro de l'étape et le temps correspondant. Une représentation graphiques sur des figures distinctes de ces résultats permet une vue globale, ce qui facilite l'analyse complète du système. Le logiciel CPNTools n'offrant pas la possibilité de représentation graphique des résultats, nous avons récupérer ces derniers en sortie du système et les avons présenté graphiquement à l'aide du logiciel MATLAB. Ces graphes sont illustrés sur la figure 8. Ils montrent les résultats d'une simulation pour cent échantillons (représenté par N). Cette représentation permet une analyse plus rapide sur chaque étape et le suivi de l'évolution des événements qui lui sont associés et la vérification de la commande obtenue après leurs traitements. En effet, sur cette figure nous observons cinq graphes qui représentent, les données obtenues en simulation après une génération aléatoire pour cent échantillons de la parole, du geste ou du clic, de la température et de la tension artérielle du patient, et enfin la fusion des données.

Nous remarquons par exemple pour l'échantillon 22 (indiqué par un trait en pointillé sur la figure 8) que le traitement des données d'entrées issues des modalités parole et geste-clic ont donné en sortie une fusion selon les critères que nous avions prédéfinie. En effet, pour le code « 5 » (Prend) de la modalité parole et un code « 0 » (problème de localisation) de la modalité geste-clic, nous obtenons en sortie le code « 5 » (erreur de localisation) tout en prenant en compte le contexte utilisateur avec un code égal à « 1 » pour la température et la tension artérielle du patient. Après vérification de l'ensemble des échantillons, nous pouvons conclure que le système répond correctement selon les critères établis pour tous les échantillons. L'architecture mise en œuvre permet donc de prendre en charge les entrées, et suivant le contexte, donne en sortie la commande adéquate. La condition indispensable pour le franchissement de la transition « **Fusion des données »** est que la parole arrive avant le geste ou clic. Le franchissement de cette condition permet le passage vers la place « **Fusionfaite** » représentée en noir sur la figure.

La figure 9 illustre le temps T1 d'arrivée de la parole et le temps T2 d'arrivée du geste ou clic. Nous remarquons que le temps T1 est toujours inférieur au temps T2 (arrivée de parole avant la détection de geste ou de clic), sauf pour le premier échantillon. Ceci est considéré comme une fausse manœuvre ou une erreur du système. Ainsi pour cet échantillon la commande ne sera pas prise en compte et le système reste bloqué au niveau de la transition Fusion des données comme le montre la figure 7.

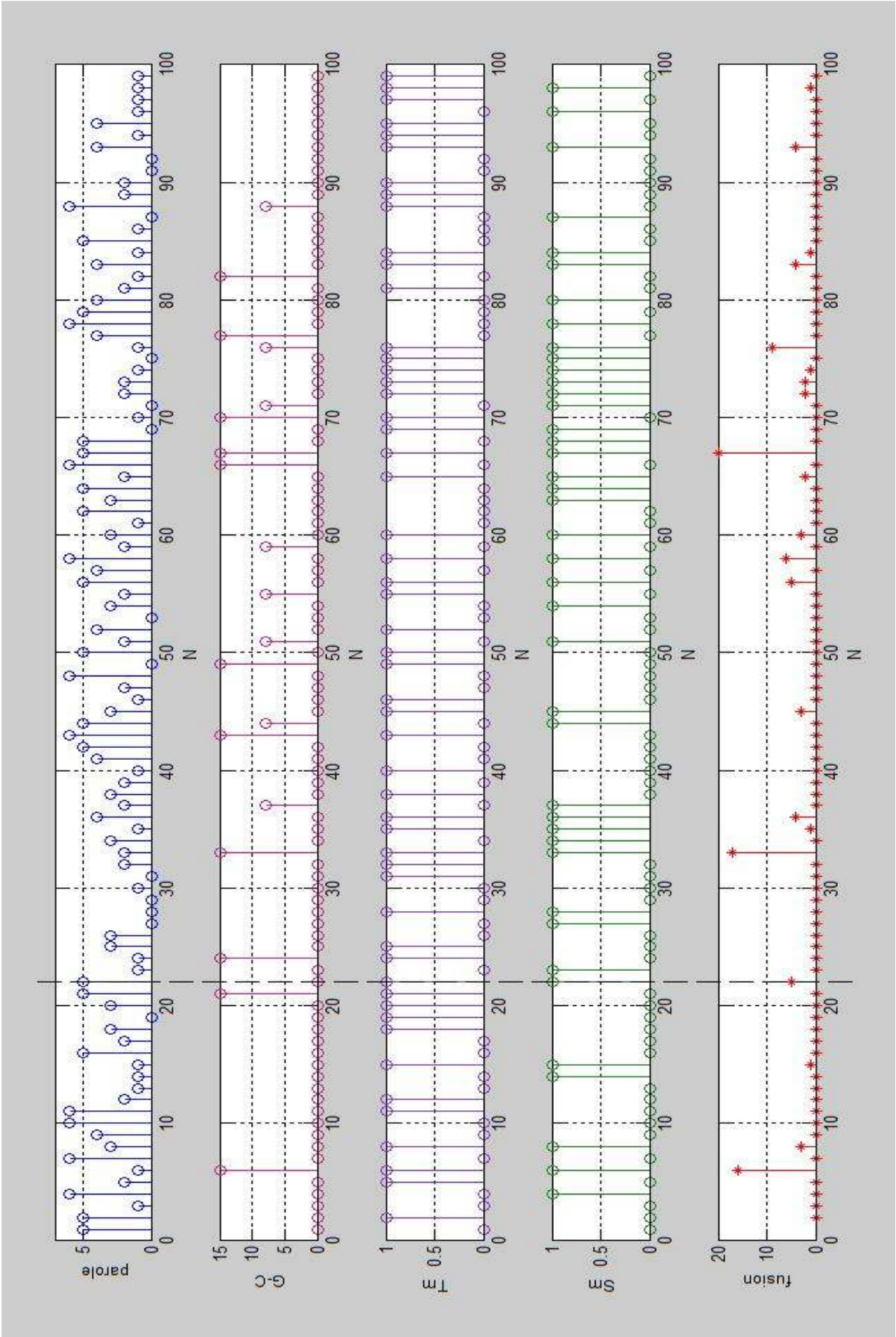

Figure 8 : Représentation graphique des résultats

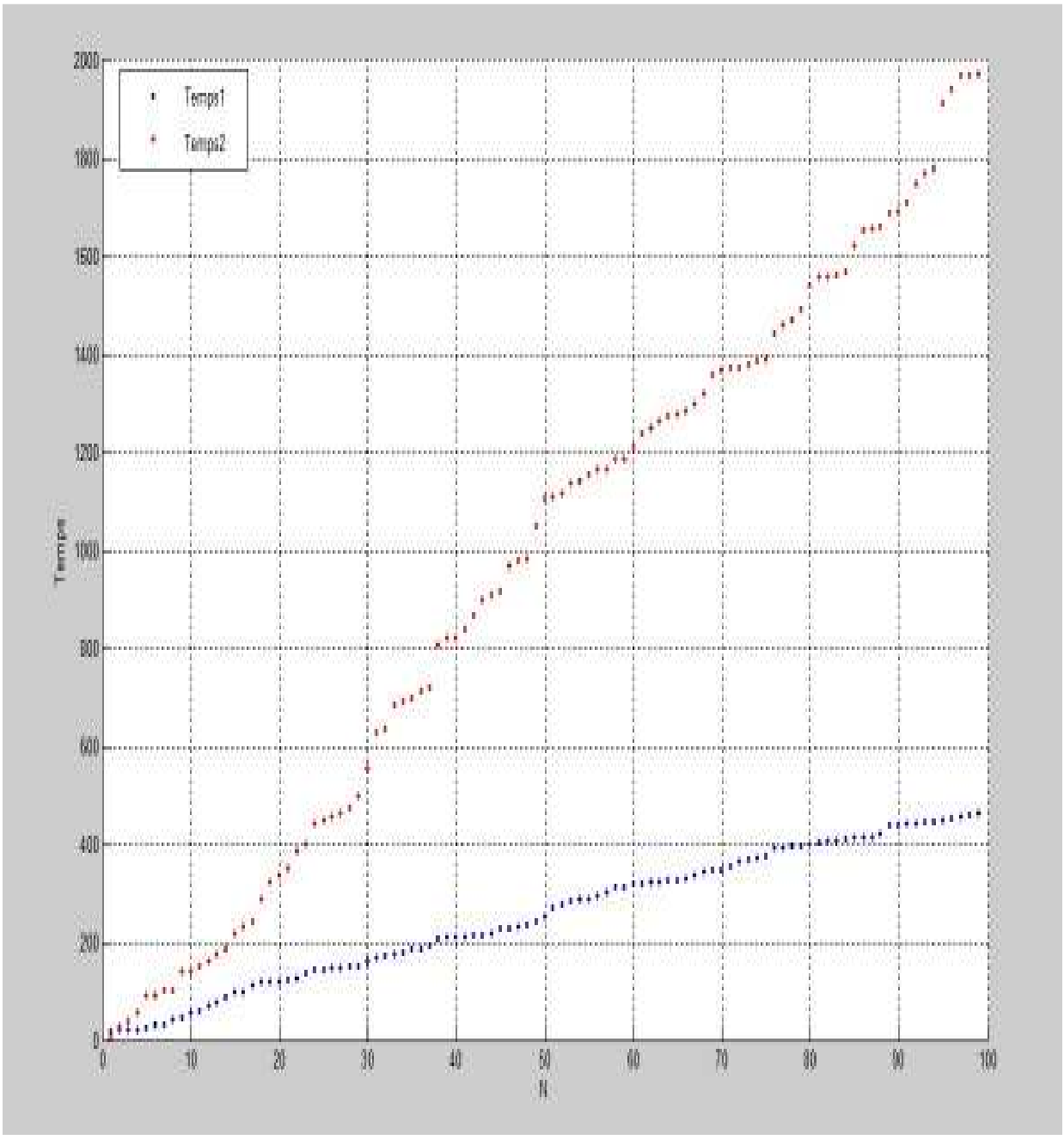

Figure 9 : Représentation des temps d'arrivée de la parole « Temps1 » et de l'arrivée de l'événement geste ou clic« Temps 2 »

Nous avons proposé dans ce chapitre une architecture multimodale qui prend en compte le contexte. Cette Architecture est dédiée à la commande d'un robot mobile muni d'un bras manipulateur. Afin de modéliser l'architecture proposée, nous avons utilisé les réseaux de Pétri colorés temporisés stochastiques pour leur facilité d'emploi dans les systèmes nécessitant du parallélisme et la prise en compte du temps. En effet, grâce à ce paradigme, nous avons géré les entrées de notre système en les traitants séparément comme étant des agents distincts responsables de chaque modalité. Nous avons ainsi pu traiter les modalités d'entrée en prenant en compte, l'état du contexte et donner en sortie, des commandes spécifiques à chaque cas.

Les tests réalisés sur CPNTools ont montré la bonne corrélation entre les entrées et les sorties du système. En effet, les sorties générés par l'architecture multimodale proposée correspondent exactement aux demandes émises par l'utilisateur en prenant en compte pour chaque commande, l'état de santé de l'utilisateur. De ce fait, nous pouvons dire que les résultats obtenus permettent de valider l'approche proposée.

Néanmoins, dans cette première approche les entrées du contexte et des modalités événements sont considérées comme des variables de type binaire. Les informations sur l'état du patient est soit vrai (1) si le patient va bien, ou fausse (0) dans le cas contraire. De plus les capteurs d'événement donnent une position sans erreur. Dans la vie réelle ces hypothèses sont fausses. En effet, l'état du patient est représenté par des variables de type réel et lors de la localisation, les capteurs fournissent des valeurs de type réel qui sont souvent entachées d'erreurs.

Afin de pallier à ce problème, nous avons choisi d'introduire une seconde approche qui prendra en considération des données plus réalistes issues de ces capteurs. Pour ce faire, nous avons choisi d'utiliser le concept de la logique floue dans le traitement de ces données. Cette approche est présentée au chapitre suivant.

Chapitre IV : Structure multimodale basée sur la logique floue

Dans le chapitre précédent, nous avons proposé une approche de commande qui consiste en l'élaboration d'une architecture multimodale qui prend en compte le contexte en utilisant le concept d'agent. Dans cette première approche, nous avons choisi de considérer que les capteurs de température et de tension du patient fournissaient des valeurs binaires. Aussi, nous avons supposé que les capteurs utilisés dans la détection du geste ou du clic donnent une valeur juste et nous n'avions pas pris en compte les erreurs possibles lors de cette mesure. Nous avions fait ce choix d'utiliser ces valeurs car notre souci principal était d'élaborer une architecture capable de prendre en charge les modalités d'entrées et les valeurs issues du contexte, de les traiter par une fusion des données et de donner en sortie la commande adéquate à chaque cas de figure.

Dans ce chapitre, nous exposerons une seconde architecture de commande basée sur la logique floue. Nous avons choisi d'introduire la logique floue en raison de sa capacité de traiter des données suivant un domaine de variation. Ceci dit, l'architecture proposée permet de prendre en charge les valeurs réelles de température et de tension de l'utilisateur et réagir en conséquence. De plus, elle utilise la logique floue pour gérer les erreurs de désignation et décider si elles sont acceptables ou non.

Nous commencerons par une présentation générale de la logique floue et expliquerons la méthodologie de mise en œuvre de cette théorie dans le traitement des données. Puis, nous exposerons les modalités traités par cette approche. Enfin, nous présenterons et commenterons les résultats graphiques obtenus lors des tests des programmes développés.

IV.1 **Généralité sur la logique floue** :

La logique floue est née du fait que la plupart des phénomènes ne peuvent pas être décrit correctement par la logique booléenne, c'est-à-dire que le fait de dire qu'un phénomène est 100% juste ou 100% faux n'est pas toujours vrai. En effet, un phénomène peut être vrai à un certain degré ou faux à un certain degré, par exemple, une eau à 20 degrés ne peut ni être considéré comme froide ni comme chaude, par contre, elle peut être décrite comme étant tiède. La logique floue considère qu'un phénomène peut appartenir à un ensemble et non plus à une fonction tout ou rien et peut ainsi prendre des valeurs comprises entre un intervalle de données ou entre 0 et 1.

Nous avons choisi d'introduire la logique floue dans notre raisonnement lors de la conception de notre deuxième approche pour pouvoir exploiter cette notion d'appartenance

qu'elle offre. Nous utiliserons cette approche dans le traitement des données issu du contexte. Ainsi, le traitement sera choisi selon le degré d'appartenance des modalités et du contexte aux ensembles flous correspondants. Dans cette approche nous utilisons la logique floue pour traiter l'état réel du patient (température et tension) et l'erreur résultante d'une désignation par le geste ou le clic d'un endroit (ou d'un objet).

IV.2 **Prise en compte des données d'état du patient par la logique floue**:

Les données issues des capteurs d'état du patient donnent sa température et sa tension artérielle. Les valeurs normales de ces grandeurs varient selon l'âge du patient, par conséquent nous ne pouvons pas les généraliser. Dans la présente étude nous avons choisi de prendre comme exemple des valeurs de température et de tension artérielle normaux pour une personne âgée d'une vingtaine d'années. Ainsi, selon l'état du patient sa température peut être comprise entre 34° et 40°. De plus, nous avons choisi de vérifier les deux tensions artérielles du patient: la pression maximale au moment de la contraction du cœur (systole), et la pression minimale au moment du « relâchement » du cœur (diastole). La pression maximale d'une personne âgée d'une vingtaine d'années est comprise entre 8 et 16 (80 mmHg et 160 mmHg). La pression minimale est comprise entre 3 et 10 (30 mmHg et 100 mmHg).

Remarque:

> *Si notre système doit être utilisé par une personne dans une autre tranche d'âge il suffira d'apporter des modifications pour l'adapter à son cas.*

Les figures 10, 11 et 12 représentent les domaines de variation des différentes données issues des capteurs d'état du patient. Ces données seront traitées par la logique floue. Lors de la mise en œuvre nous avons choisi d'utiliser des fonctions d'appartenance de type trapézoïdales en grande partie pour leurs simplicités et la facilité de leurs mises en œuvre.

Nous avons partagé le domaine de variation de la température en trois partie chacune représenté par un trapèze (figures 10). Les températures comprises dans l'intervalle [34°, 36°[sont considérées comme basse et sont représentées par « B » sur la figure. Les températures comprises dans l'intervalle [36°, 38°] sont considérées comme normal et sont représentées par « N ». Les températures comprises dans l'intervalle]38°, 40°] sont considérées comme haute et sont représentées par « F ».

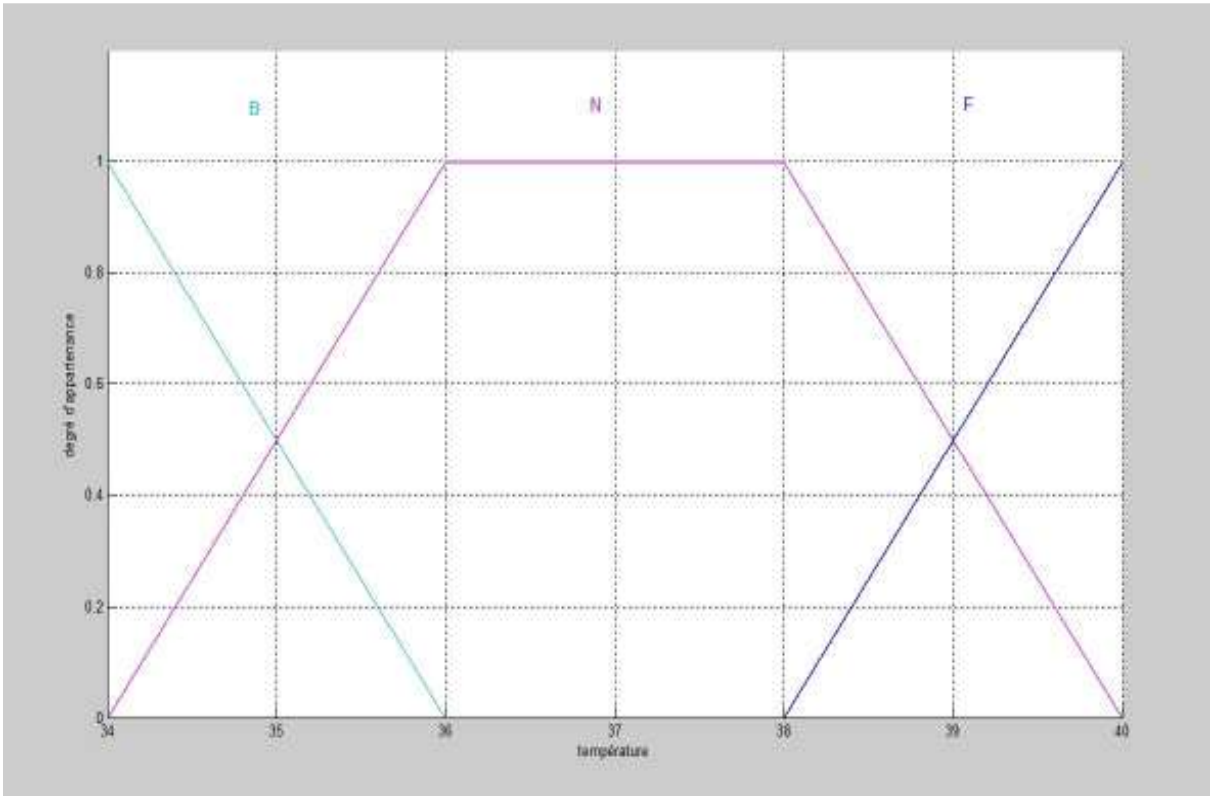

Figure 10 : Univers de discours de la température

Les figures 11 et 12 illustrent respectivement les variations de la tension maximale et minimale. Les tensions maximale comprises dans les intervalles [8, 10[et]14, 16] sont considérées comme mauvaise et sont représentés sur la figure 11 par MT1 et MT2. Les tensions minimale comprises dans les intervalles [3, 5[et]8,10] sont considérées comme mauvaises et sont représentés respectivement sur la figure 12 par MT3 et MT4. Les mesures comprises dans l'intervalle [10, 14] pour la tension maximale et [5, 8] pour la tension minimale sont considérées comme étant des bonnes tensions et sont représentées par « BT » sur les deux figures.

Dans l'architecture de commande proposée nous considérerons que l'état de santé du patient est bonne si sa température et sa tension artérielle (maximale et minimale) sont normales (Températures: [36, 38], tensions maximale: [10, 14], tension minimale: [5, 8]). Dans le cas contraire nous considérons que le patient a des problèmes de santé. Dans ce cas, l'architecture multimodale proposée prend en compte ces problèmes et réagit en conséquence en imposant un arrêt du système robotique et en envoyant la localisation du patient fournie par le GPS.

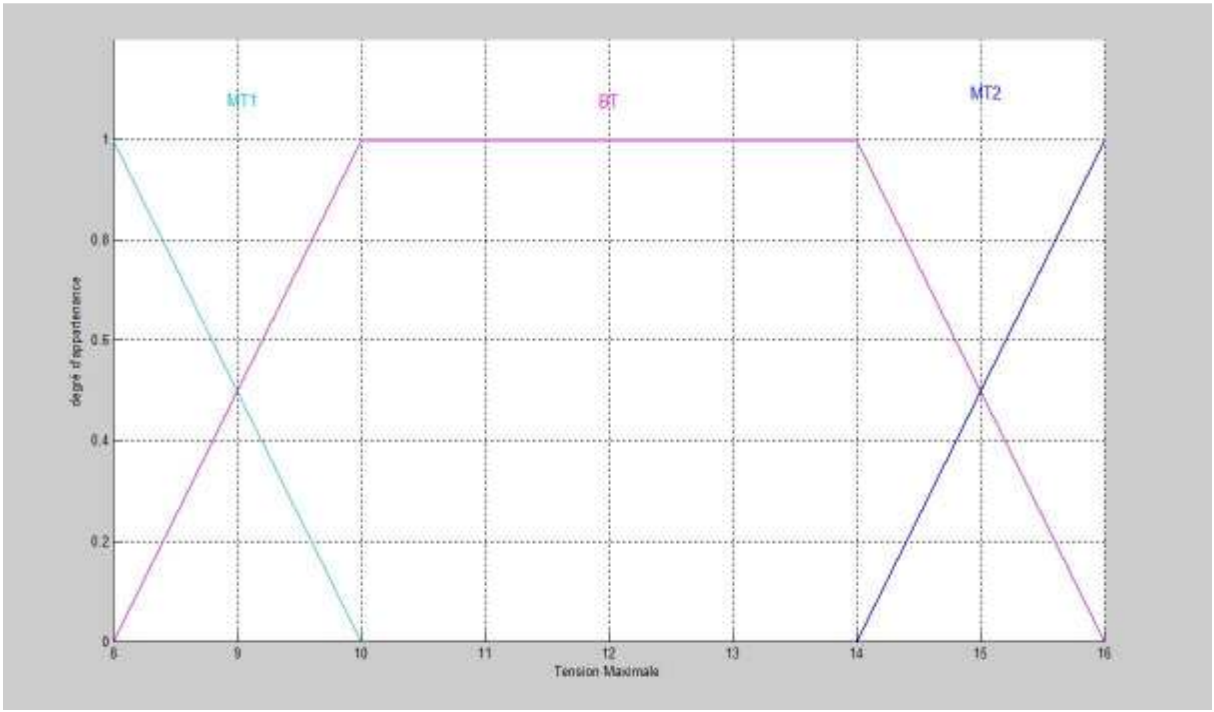

Figure 11 : Univers de discours de la tension maximale

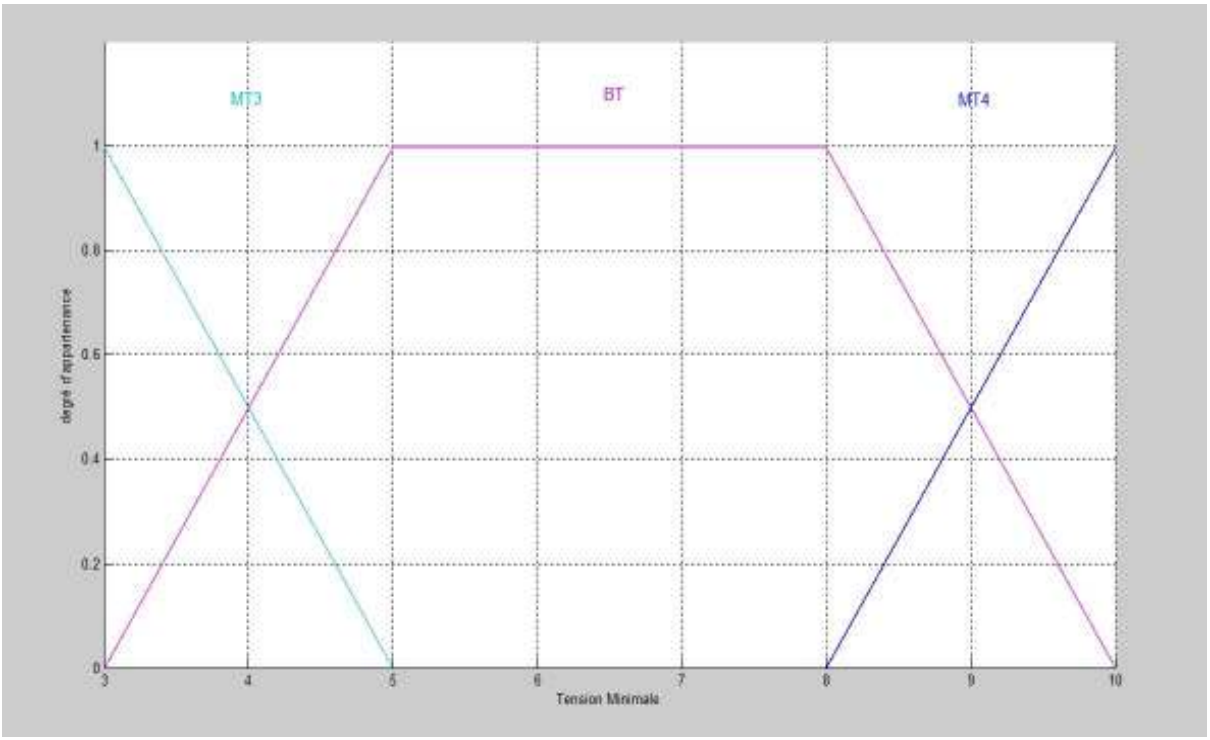

Figure 12 : Univers de discours de la tension minimale

Nous avons utilisé dans notre étude l'approche de Mamdani comme système de fuzzification [33]. Les règles choisies pour notre architecture multimodale sont comme suit:

	Tmf ϵ B			**Tmf ϵ N**			**Tmf ϵ F**		
	Smpf ϵ MT1	**Smpf ϵ BT**	**Smpf ϵ MT2**	**Smpf ϵ MT1**	**Smpf ϵ BT**	**Smpf ϵ MT2**	**Smpf ϵ MT1**	**Smpf ϵ BT**	**Smpf ϵ MT2**
SmGf ϵ MT3	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes
SmGf ϵ BT	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient est en bonne santé	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes
SmGf ϵ MT4	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes	le patient a des problèmes

Tableau 4 : Règles de la logique floue choisies pour les données correspondantes aux états du patient

IV.3 **Prise en compte de l'erreur de désignation par la logique floue**:

Le système mis en œuvre est conçu pour être utilisé par des personnes dépendantes. Ces personnes étant malades, elles peuvent lors de la désignation d'un endroit engendrer une erreur due à des tremblements de la main ou à une mauvaise maitrise des mouvements. Par conséquent, cette désignation ne sera pas toujours juste et sera entachée d'erreur. De ce fait, nous avons choisi d'introduire dans cette seconde approche une prise en compte de cette erreur dans la prise de décision. Nous utiliserons la logique floue pour traiter ses variations. La figure 13 illustre les domaines de variations choisis de l'erreur de désignation. Trois domaines apparaissent:

- la partie TB (très bonne) où l'erreur de positionnement est comprise entre 0cm et 3cm ([0, 3[)
- la partie B (bonne) où l'erreur de positionnement est comprise entre 3cm et 5cm ([3, 5])
- et enfin, la partie M (mauvaise) où l'erreur de positionnement est comprise entre 5cm et 10cm, ( ]5, 10]).

Dans l'architecture multimodale proposée nous avons choisi d'accepter les erreurs qui se situent dans les deux parties « TB » et « B », et de ne considérer que l'erreur qui se situe au-delà de 5cm comme étant une erreur de désignation trop importante. Dans ce cas nous ne devons pas prendre en charge les informations relatives à la commande demandée par le patient. Lors du traitement de l'erreur de désignation par la logique floue nous avons choisi, comme précédemment, d'utiliser le système de Mamdani. Les règles utilisées sont les suivantes:

	Geste	**Clic**	**Pas de détection**
Erreur € TB	bonne désignation	bonne désignation	mauvaise désignation
Erreur € B	bonne désignation	bonne désignation	mauvaise désignation
Erreur € M	mauvaise désignation	mauvaise désignation	mauvaise désignation

Tableau 5 : Règles de la logique floue choisies pour la prise en compte de l'erreur de désignation

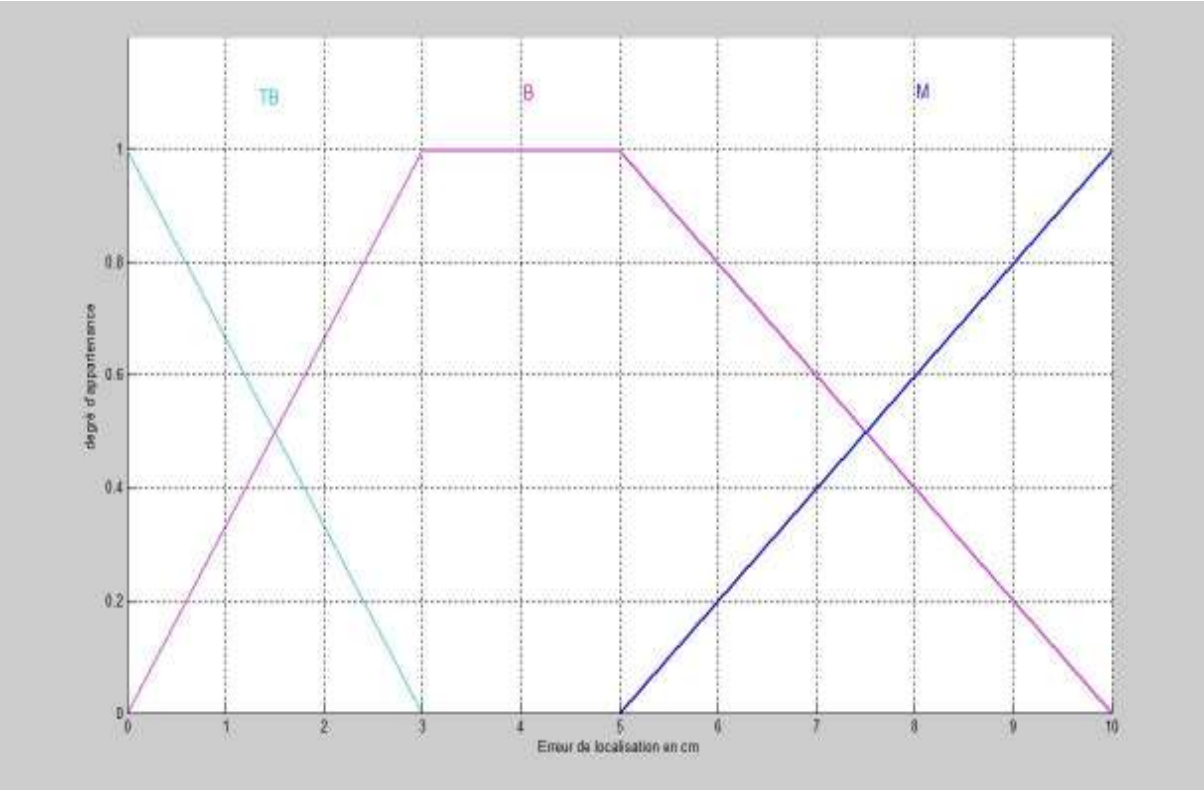

Figure 13 : Univers de discours de l'erreur de désignation en centimètre

IV.4 **Modélisation des agents par réseau de Pétri et par la logique floue :**

Par rapport à la première approche (chapitre III) aucune modification n'a été apportée aux modalités parole et fusion des données, par conséquent nous ne les aborderons pas dans ce chapitre. Cette seconde approche utilise la logique floue dans le traitement des données issues des modalités spécifiques à l'état du patient ainsi que celles relatives à la désignation par le geste ou le clic.

IV.4.1 **L'agent TmSm** :

Le réseau de Pétri qui prend en charge l'agent « TmSm » qui permet de traiter les modalités issues des capteurs d'état du patient utilise la logique floue. La figure 14 illustre l'architecture mise en œuvre

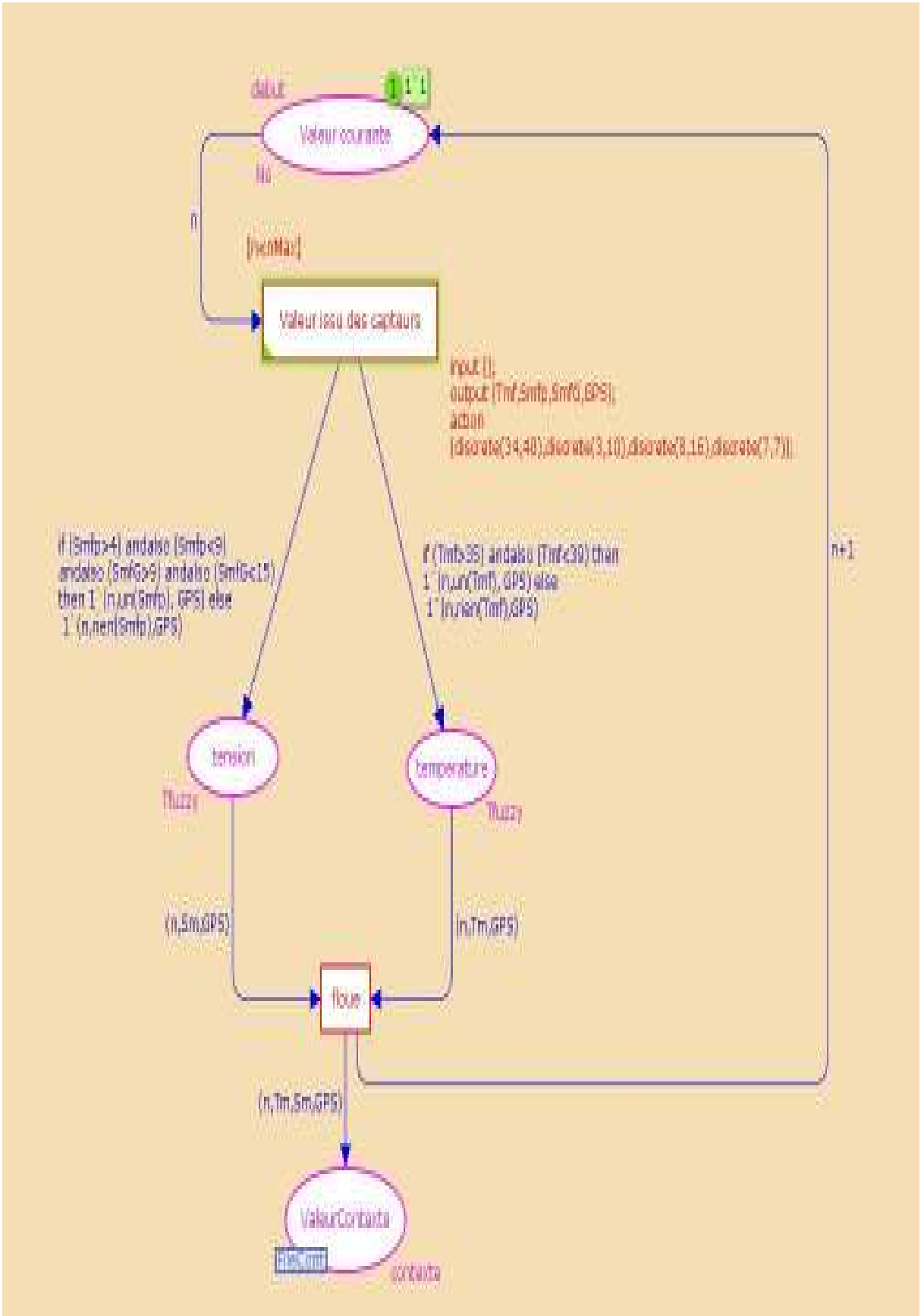

Figure 14 : Réseau de Pétri qui modélise l'agent TmSm en utilisant la logique floue

Nous remarquons que le traitement des données issues des capteurs de température et de tension artérielle a changé. En effet, l'introduction de la logique floue nous permet d'être plus proche de la réalité traitant différents cas et de réagir en conséquence. Dans cette nouvelle architecture, nous avons rajouté des branches composées de transition, place et arc qui vont modéliser le traitement réalisé par la logique floue.

Nous avons changé l'appellation des variables pour les distinguer de celles utilisées dans la première approche. Nous précisons que « Tmf » désigne la température du patient dans le domaine floue, « Smfp » est la tension minimale du patient dans le domaine floue et « SmpG » est la tension maximale du patient dans le domaine de la logique floue. Pour plus de détails sur les déclarations des nouvelles appellations, on avise le lecteur à se référer à la liste des abréviations.

Ces nouvelles informations vont être envoyées vers l'agent fusion via la transition « ValeurContexte », qui est une place spatialement distribuée dans notre réseau.

IV.4.2 **L'agent GestClic** :

Nous avons aussi introduit la logique floue dans le traitement des erreurs issues de la désignation d'un endroit ou d'un objet par le geste ou le clic. La nouvelle architecture est représentée sur la figure 15. Sur cette figure, nous retrouvons l'agent responsable du traitement des événements (geste ou clic) avec une prise en compte de l'erreur possible engendrée lors de la désignation de ces événements.

L'architecture proposée permet de répondre à une demande de l'utilisateur qui peut être une combinaison entre une commande vocale suivie par une désignation d'un endroit (ou d'un objet) par le geste ou le clic. En effet, l'utilisateur peut par exemple dire « pose ça ici », et fait suivre cette commande vocale par une désignation d'un endroit par un geste capté par une caméra ou un clic de souris sur l'image renvoyée par la caméra. Ou encore dire par exemple « avance » et faire suivre cette commande par la désignation de l'endroit où il désire arriver par le geste ou le clic.

Nous avons choisi de prendre un intervalle de variation de l'erreur de désignation d'un endroit ou d'un objet sur l'écran comprise entre 0cm et 10cm. Afin de tester en simulations tous les cas possibles nous avons généré ces valeurs d'erreurs aléatoirement. Pour ce faire, nous avons introduit une commande qui permet la génération aléatoire de l'erreur, en plus de

la commande qui permet la génération aléatoire de l'événement. Aussi, nous avons rajouté une place « VerrErreur » qui permet la vérification de l'erreur ainsi que l'événement correspondant et d'envoyer cette information à l'agent fusion. Cette information sera envoyée à l'agent fusion par le biais de la place « FileEventReconnu ». Cette dernière est spatialement distribuée dans le réseau pour être traitée et ainsi donner en sortie la commande relative à l'événement en prenant en compte l'erreur engendrée lors de la désignation d'un endroit ou d'un objet.

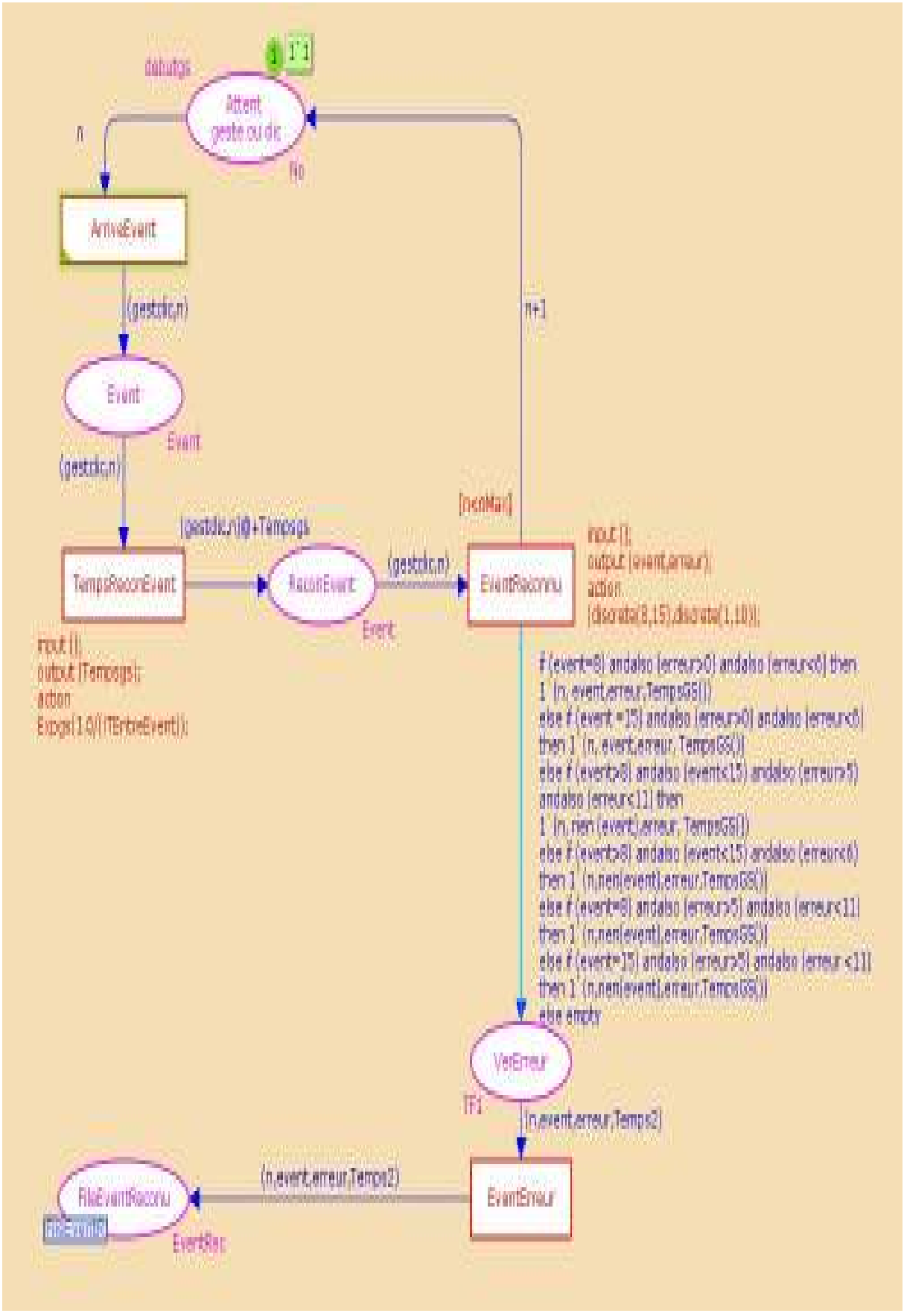

Figure 15 : Réseau de Pétri qui modélise l'agent GestClic en utilisant la logique floue

Tests de l'architecture proposée:

Afin de tester et de valider l'architecture proposée nous l'avons simulée sur CPNTools Nous présenterons tout d'abord la simulation de l'agent fusion relative à l'introduction de la logique floue dans le traitement de la température et la tension artérielle uniquement. Puis nous nous intéresserons aux résultats obtenus après traitement par la logique floue des agents TmSm, ainsi que l'agent GestClic..

IV.5.1 Résultats obtenus après le traitement par la logique floue des données relatives à l'état du patient:

Lors de la mise en œuvre, nous avons introduit la logique floue dans le traitement des variables issues du contexte, étape par étape. Nous avons commencé par son introduction juste dans l'agent TmSm relatif à l'état du patient. La figure représentative de l'agent TmSm a été présenté précédemment (figure 14).

La figure 16 illustre les résultats obtenus au niveau de l'agent fusion après simulation sur CPNTools pour 100 échantillons. Elle donne les résultats obtenus après une génération aléatoire des données issues des modalités et du contexte. Ces résultats sont représentés devant chacune des places de sortie dans un carré. Nous y retrouvons les commandes résultantes de la fusion de chaque modalité d'entrée avec la prise en compte du contexte.

Nous remarquons, que toutes les commandes ont pu passer par la transition « Fusion des données » ce qui veut dire que la condition de franchissement (arrivée de la commande vocale avant l'évènement geste ou clic: **Temps1<Temps2)** est vérifiée pour tous les échantillons, comme le montre la figure 17.

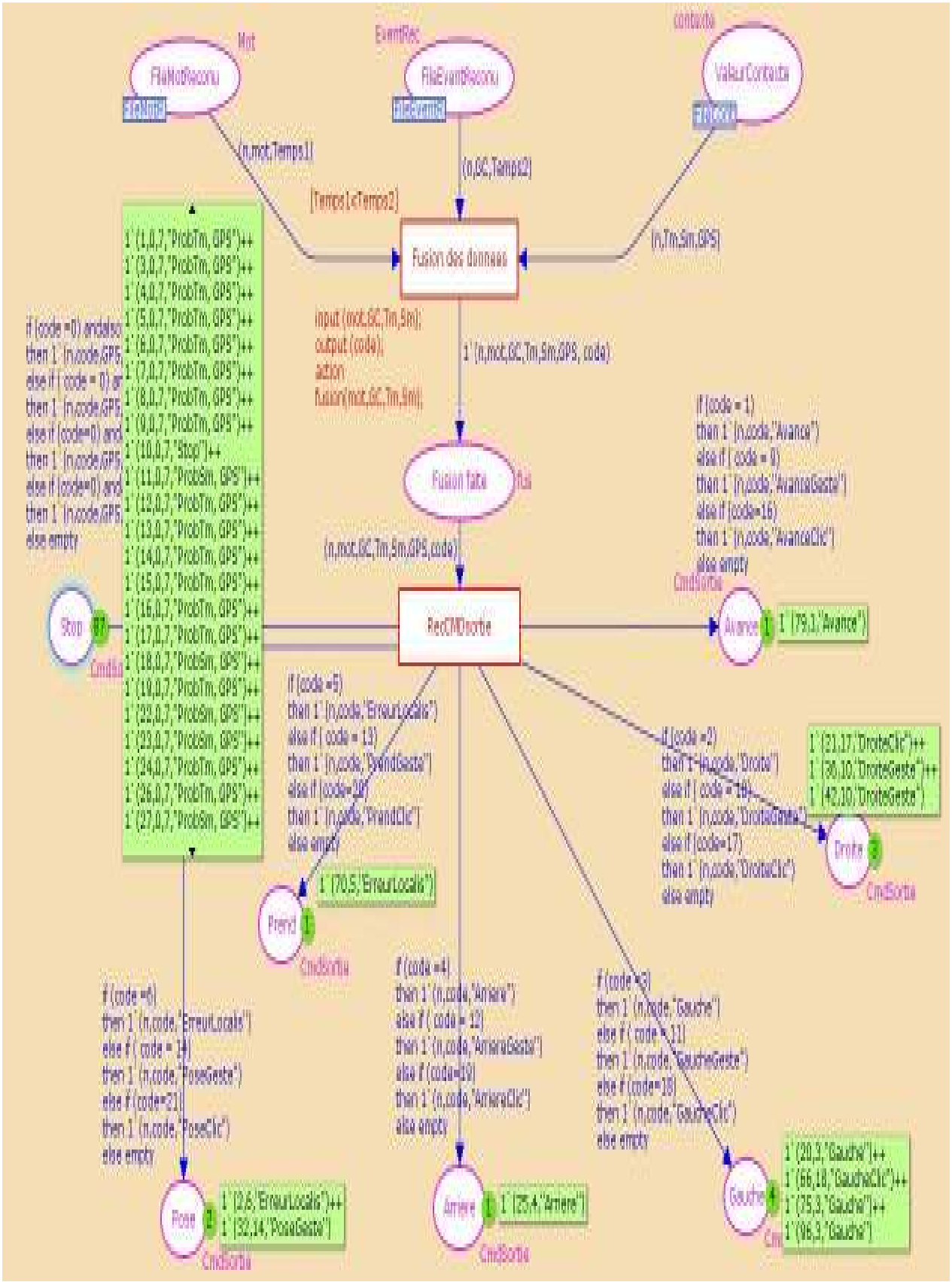

Figure 16 : Résultats obtenus après l'introduction de la logique floue dans le traitement des données de l'agent TmSm

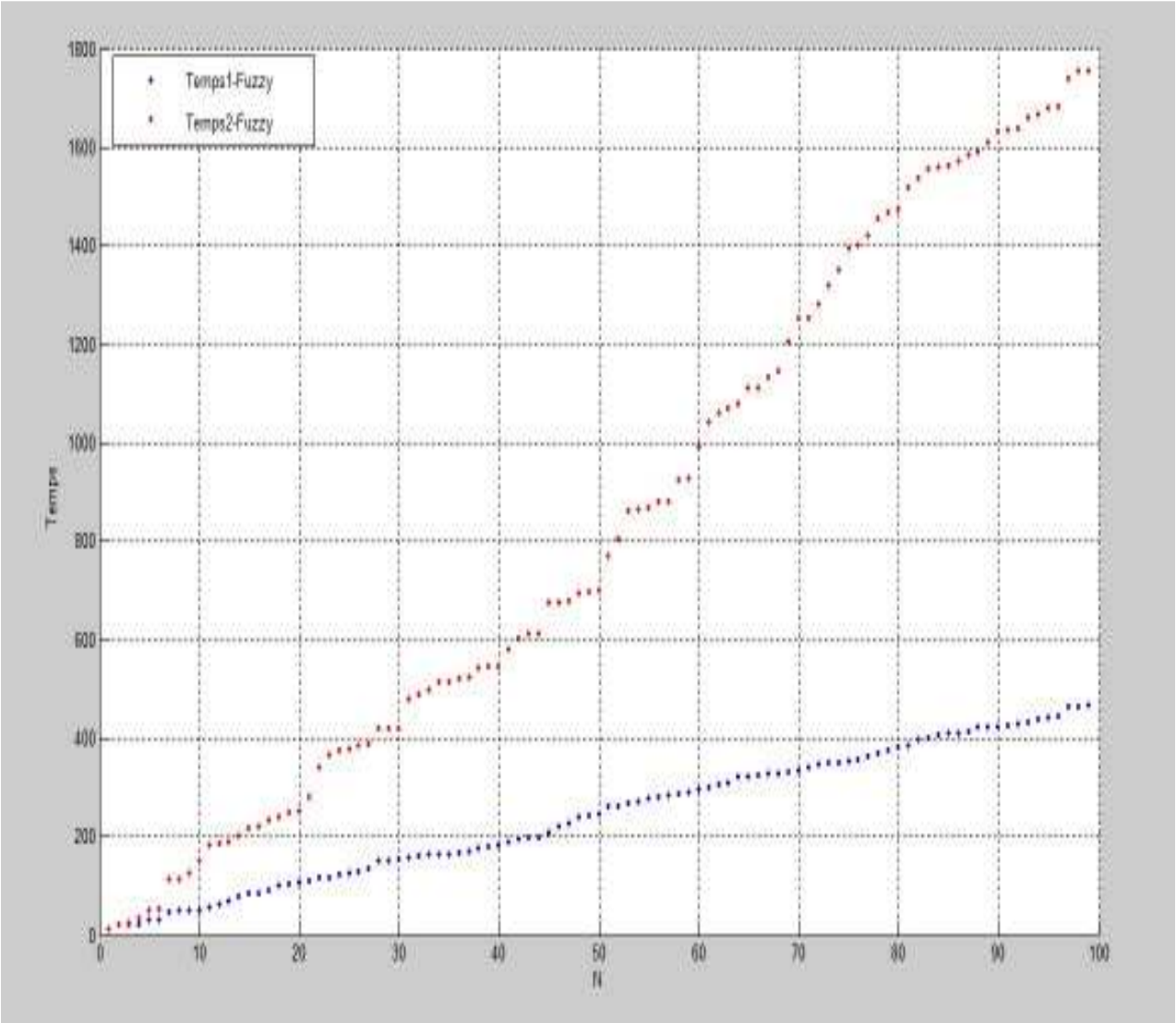

Figure 17 : Représentation des temps d'arrivée de la parole (Temps1) et de l'événement (Temps2) dans le cas de l'introduction de la logique floue des données de l'agent TmSm

Dans la figure 18 nous présentons une vue plus générale sur les résultats obtenus par la simulation sur CPNTools. Nous remarquons que l'introduction de la logique floue dans le traitement des données relatif à l'état du patient a permis d'avoir un plus grand domaine de variation de ces données et une meilleur exploitation de ces informations. Les données étant plus réalistes, elles permettent une estimation plus juste de l'état du patient. L'architecture mise en œuvre peut ainsi offrir une meilleure prise en charge. En effet, nous pouvons, par exemple, vérifier la commande de sortie de l'échantillon 42 (marquée par un trait noir discontinu sur la figure 18). Nous avons pour cet échantillon le code de la parole « 2 », qui représentent le mot « **Droite** » (qui correspond à une demande de tourner à droite), le code

« 8 » correspondant à un geste dans la détection d'un événement. Pour ce qui est de la température corporelle elle est de 37°, et les deux tensions maximales et minimales sont respectivement égales à 10 et à 6. D'après ces données nous pouvons dire que le patient n'a pas de problème de santé, par conséquent, la commande vocale suivie de la désignation par le geste peut être prise en compte. L'architecture mise en œuvre, donne en sortie une commande au robot qui a pour code « 10 ». Ce dernier correspond à la commande de tourner à droite jusqu'à la position désignée par le geste et qu'on retrouve sur la figure 16 dans le carré à proximité de la sortie « droite » où l'on retrouve le message « DroiteGeste ».

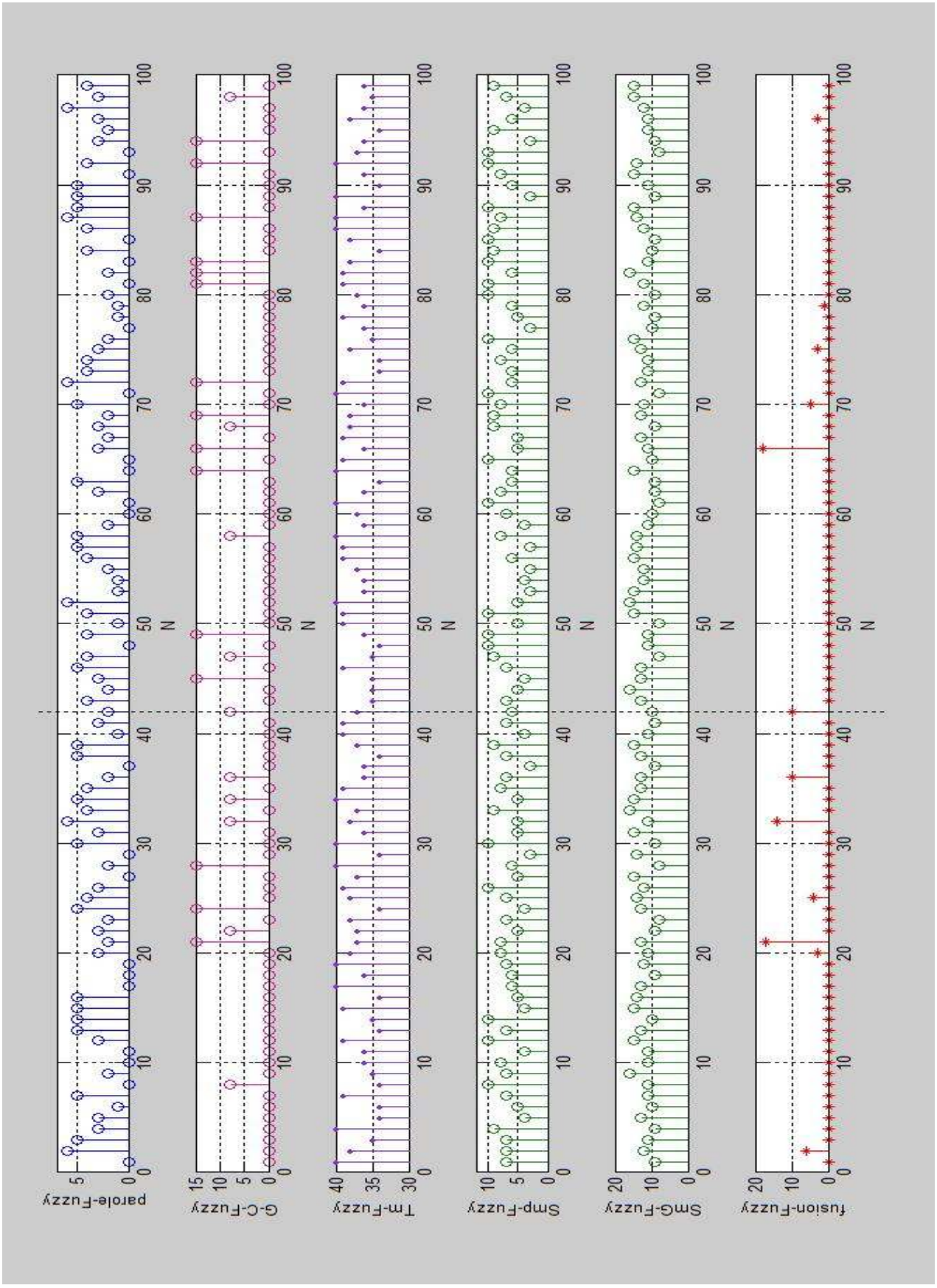

Figure 18 : Représentation graphique des résultats obtenus après introduction de la logique floue dans le traitement de données issues des capteurs d'état du patient

IV.5.2 **Résultats obtenus après traitement par la logique floue des données d'état du patient et l'erreur engendrée lors de la désignation d'un endroit ou d'un objet**:

Nous introduirons dans cette phase de mise en œuvre la logique floue dans le traitement des données relative à l'état du patient ainsi que les événements de désignation d'un endroit ou d'un objet par le geste ou le clic. Nous présenterons les résultats de la fusion des différentes modalités obtenus après déroulement de la simulation sur CPNTools, puis, nous donnerons les représentations graphiques des données pour 100 échantillons.

La figure 19 illustre les résultats obtenus lors de la simulation sur CPNTools. Sur cette figure, nous remarquons que les deux premiers échantillons n'ont pas pu passer la transition fusion des données car le temps d'arrivée de ces deux événements est inférieur au temps d'arrivée de la parole correspondante (voir figure 20). Conformément au critère que nous avons établi, ces deux demandes de l'utilisateur ne seront pas prises en compte par notre architecture.

Nous remarquons que les sorties « **Prend** » et « **Arrière** » (qui représentent respectivement la réponse à une commande de prendre un objet ou de faire une marche arrière) n'ont pas de message qui les accompagne. Cela veut tout simplement dire que lors de cette simulation les codes correspondant à ces deux demandes n'ont pas été générés (génération aléatoire des codes).

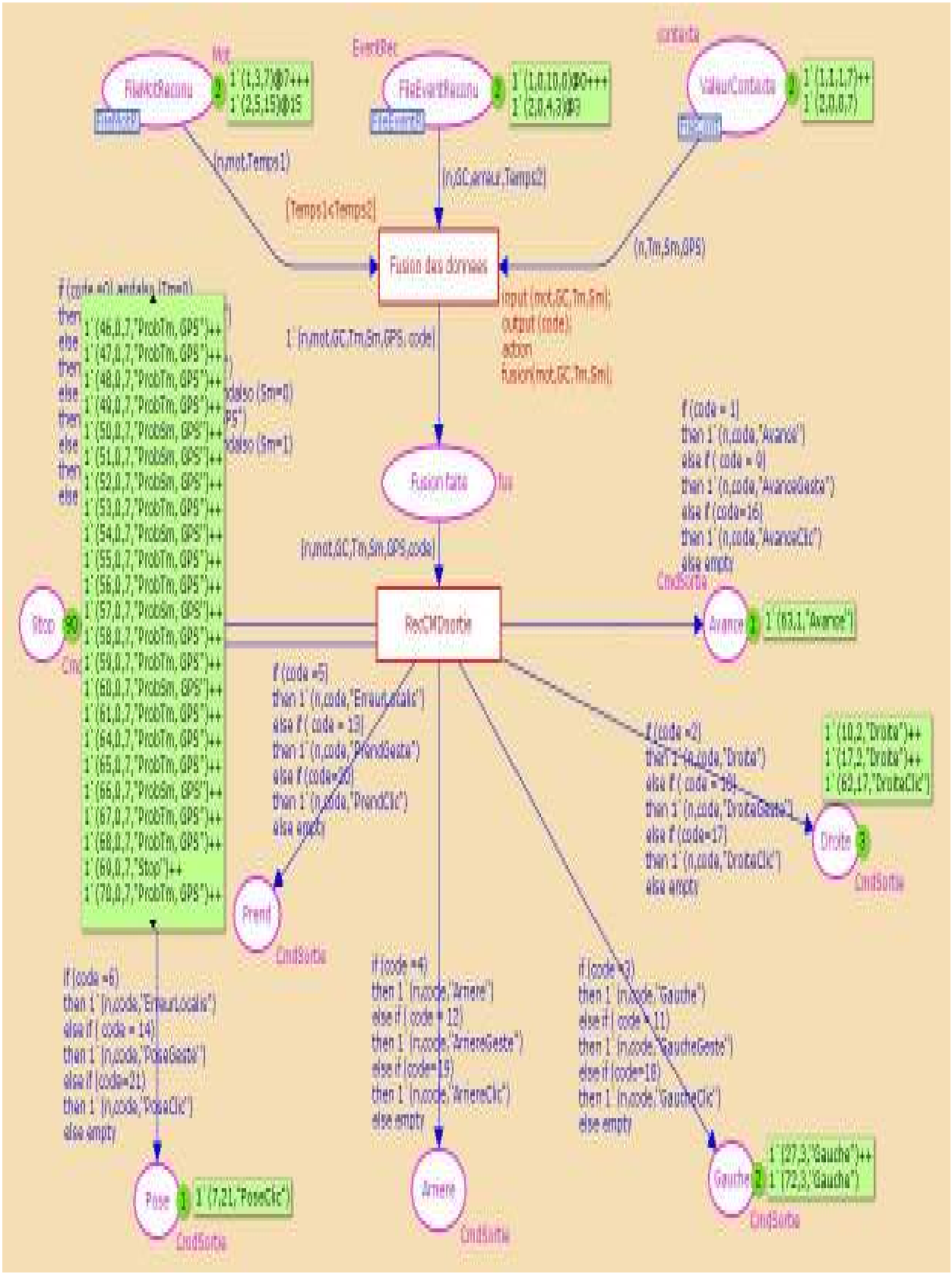

Figure 19 : Résultats obtenus lors de l'introduction de la logique floue dans le traitement des données relatif à l'état du patient ainsi que l'erreur de désignation d'un endroit ou d'un objet par les événements (geste ou clic).

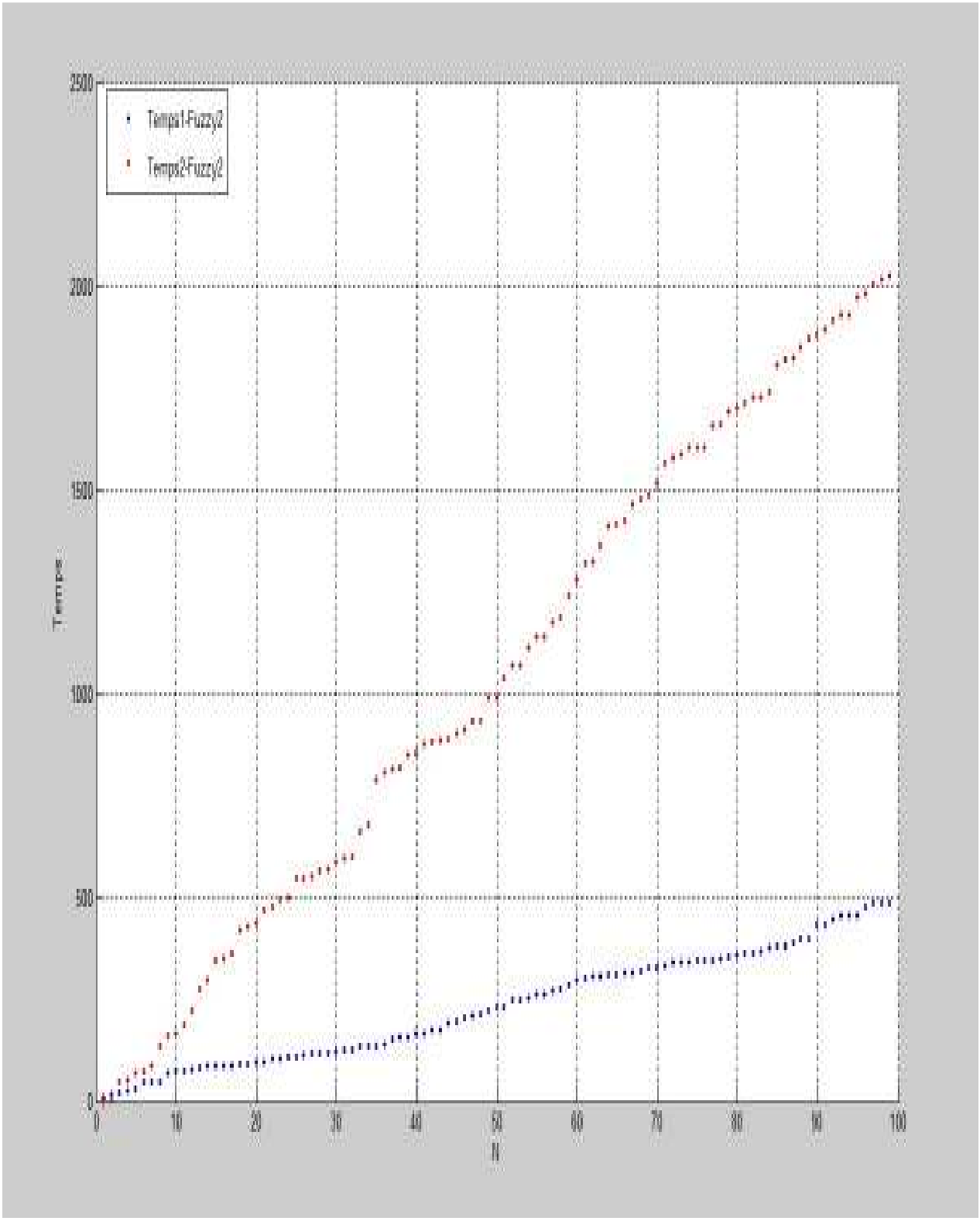

Figure 20: Représentation des temps d'arrivée de la parole (Temps1) et de l'événement (Temps2) Lors du traitement par la logique floue des données relatives à l'état du patient et de l'événement geste ou clic

La figure 21 illustre une représentation graphique des résultats obtenus lors du traitement des données par notre système et leurs fusions. Nous pouvons par exemple vérifier les données des entrées et voir ce que l'architecture mise en œuvre a donné en sortie pour l'échantillon 60 marqué par un trait en pointillé noir sur la figure 21. Nous avons pour l'entrée parole le code « 4 » et qui est le code de la commande vocale « Arrière ». Elle correspond à une demande de marche arrière. Le code correspondant à l'événement est de « 0 » car l'erreur est de « 9 » ce qui désigne une mauvaise désignation d'un endroit ou d'un objet. Les valeurs d'état du patient sont « 36 » pour la température du patient, « 5 » pour la tension minimale et « 15 » pour la tension maximale. Cette dernière valeur est considérée comme étant une valeur élevée. Elle indique que le patient a des problèmes de santé. Par conséquent, on retrouve en sortie le code « 0 » qui correspond à une demande d'arrêt du système en précisant qu'il y a un problème dans de tension artérielle chez le patient, et l'envoie de la localisation du patient fourni par le GPS. L'ensemble de ces informations est représenté par le message correspondant « ProbSm, GPS ».

Le traitement par la logique floue de l'erreur résultante lors de la capture d'un événement (geste ou clic) en plus du traitement de l'état du patient permet de considérer un plus grand domaine de variations de l'ensemble des entrées issues de ces modalités. Ainsi l'architecture mise en œuvre permet de traiter les données relatives au contexte avec plus de précision. L'introduction de la logique floue dans le traitement de ces modalités permet de rendre le système plus flexible. En effet, il suffit de changer les limites des intervalles de variations des fonctions d'appartenances pour les adapter aux conditions physiques spécifiques à chaque patient.

Remarque :

> Nous avons établi des priorités dans le traitement des données. Sachant que l'état de l'utilisateur est primordial et plus important, nous avons choisi de ne fournir en sortie que l'alerte sur son état de santé sans préciser l'erreur survenue lors de la désignation d'un endroit ou d'un objet. Ceci dit nous ne précisons pas dans le message de sortie « ProbSm, GPS » qu'il y a eu une erreur de désignation (erreur = 9).

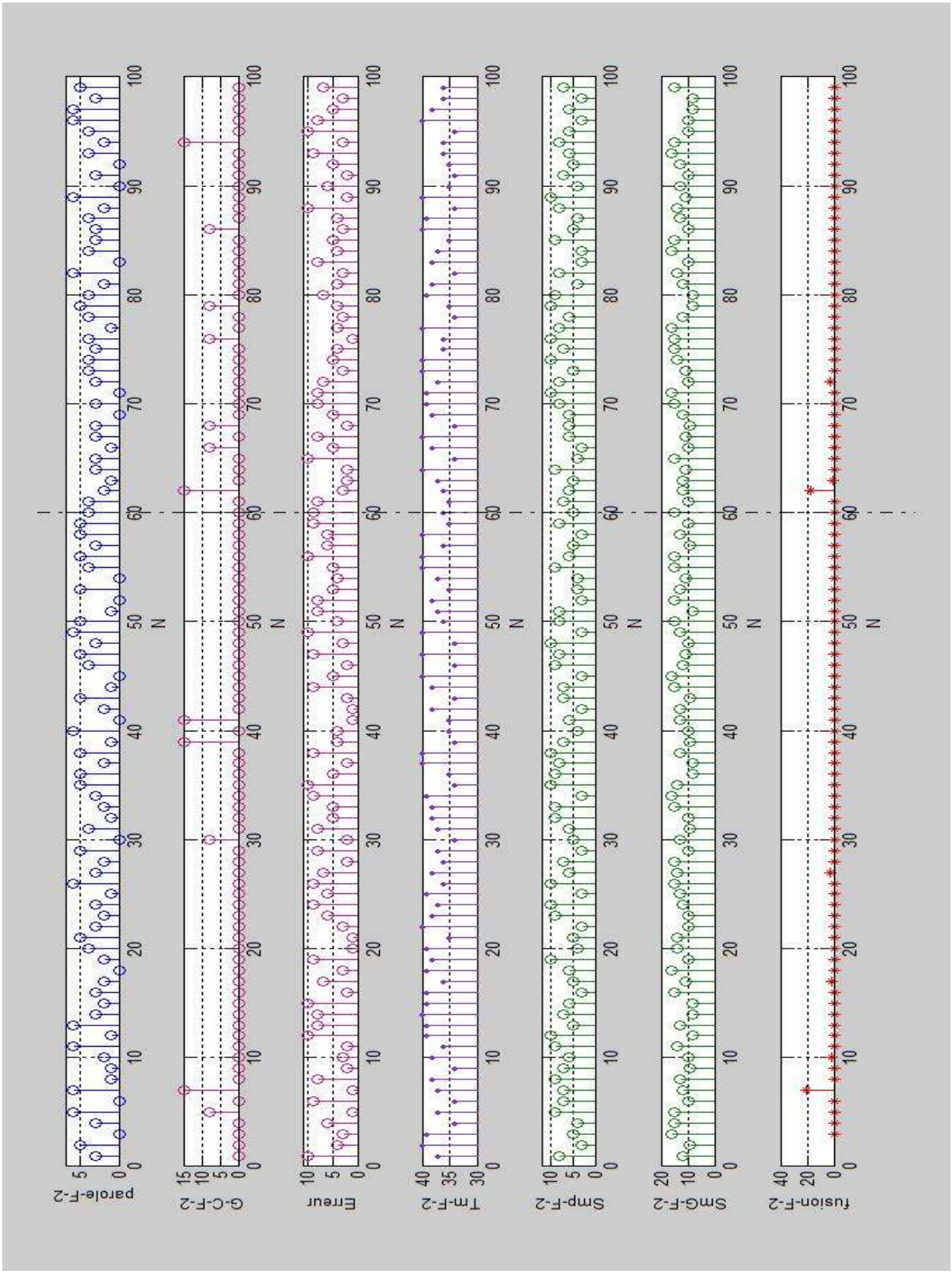

Figure 21 : Représentation graphique des résultats obtenus après introduction de la logique floue dans le traitement de données issues des capteurs d'état du patient et des événements geste ou clic.

Nous avons présenté dans ce chapitre une seconde architecture multimodale pour la commande d'un fauteuil roulant muni d'un bras manipulateur. L'architecture proposée utilise la logique floue dans le traitement des données issues des capteurs d'état du patient ainsi que de l'erreur résultante de la désignation d'un endroit ou d'un objet *par le geste ou le clic.*

Cette approche est plus réaliste que la première. En effet, elle permet de prendre en compte, contrairement à la première (chapitre III), l'état réel de l'utilisateur. Le partage en domaines qu'offre la logique floue lors du traitement des données nous a permis de mieux considérer chaque cas et de réagir en conséquence.

Lors de la mise en œuvre de notre approche les limitations des domaines des fonctions d'appartenances de l'état du patient ont été choisi comme étant des valeurs descriptives pour une personne âgée, d'une vingtaine d'années. Par conséquent, ces valeurs devront être revue pour un utilisateur se situant dans une autre tranche d'âge. De même pour ce qui est des limites de variations de l'erreur. En effet, on ne peut pas considérer que l'erreur est la même pour toutes les personnes. Par exemple, on pourra avoir comme utilisateurs des personnes qui peuvent gérer leurs mouvements, comme on peut avoir des personnes présentant des difficultés (tremblement, problème de concentration, ...).

L'introduction de la logique floue à permis de rendre l'architecture proposée plus flexible et adaptable à chaque cas de figure. En effet, il est possible d'adapter les limites des fonctions d'appartenances aux conditions réelles de travail. Nous pouvons dans le cadre des travaux futurs, envisager une adaptation automatique des valeurs limites de ces intervalles selon l'état de chaque personne.

Lors des tests en simulations sur CPNTools, nous avons choisi de générer les entrées aléatoirement pour un nombre conséquent d'échantillons (dans notre cas 100 échantillons). Ce choix nous a permis de prendre en compte toutes les combinaisons pouvant être rencontrées dans la pratique. Ainsi, au vu des résultats obtenus, nous pouvons dire que l'architecture mise en œuvre permet de prendre en charge correctement les commandes émises par l'utilisateur, de prendre en compte son état de santé et de fusionnerl'ensemble de ces informations pour donner en sortie la commande adéquate à la situation et l'état de l'utilisateur.

Ce système donne ainsi une opportunité à des personnes dépendantes de pouvoir facilement et en toute sécurité se déplacer et d'accéder à des objets. De plus, l'utilisation de la multimodalité nous a permis de rendre notre système facile à utiliser. En effet, la multimodalité offre une facilité d'interaction entre l'utilisateur et le robot, en la rendant semblable à une interaction entre personnes. La prise en charge du contexte utilisateur, rend l'utilisation du robot plus sécurisé. Cette sécurité se présente dans la capacité du système mis en œuvre à détecter chez l'utilisateur un problème de santé et d'offrir la possibilité d'arrêter le robot en transmettant sa localisation ce qui permet d'apporter une aide médicale si nécessaire.

Conclusion Générale

Conclusion Générale

Nous avons développé dans ce mémoire deux architectures multimodales dédiées à la commande d'un robot mobile muni d'un bras manipulateur. Notre travail se situe dans la combinaison de plusieurs disciplines et domaines de recherches. En effet, nous avons proposé des architectures logicielles dynamiques dans le cadre de la robotique d'aide à la personne dépendante en utilisant les interactions multimodales et la prise en compte du contexte. Nous avons pu par ce travail et en comparaison aux travaux déjà effectués sur les interfaces multimodales et sur les systèmes multi agents, proposer un nouveau système qui prend en compte le contexte dans sa prise de décision. La contribution principale de notre travail se situe dans la modélisation d'une architecture multimodale en utilisant le concept d'agent par des réseaux de Pétri colorés, temporisés et stochastiques. Nous utilisons le concept d'agent dans notre modélisation car nous avons choisi de traiter chaque modalité d'entrée et les données du contexte par des agents distincts, ce qui rend l'architecture proposée plus flexible et adaptable en offrant la possibilité de rajouter des agents pour réaliser différents traitements.

Nous avons subdivisé notre travail en deux parties. Dans la première nous avons conçu un système qui prend en compte des données binaires. Dans cette première phase notre souci principal a été de réaliser une architecture qui récupère des données en entrées et les fusionnent pour générer les commandes adéquates à la situation du contexte réel de l'utilisateur. Compte tenu des avantages que présentent les réseaux de pétri colorés, nous les avons sélectionnés pour modéliser l'architecture proposée sur CPNTools. L'un des principaux intérêts de l'utilisation des réseaux de Pétri est la possibilité d'une vérification automatique lors de la simulation des comportements de chaque agent, et de vérifier si les conditions et les lois prédéfinies ont bien été prises en compte. Les résultats obtenus lors des tests en simulation de cette première architecture a permis sa validation. Toutefois, les hypothèses émises lors de sa mise en œuvre ne permettent pas d'envisager son implémentation sur un système réel. En effet, les variables température et pression artérielle de l'utilisateur ainsi que l'erreur de désignation d'un endroit ou d'un objet ne peuvent pas être modélisées dans la vie réelle par des variables binaires. Afin de remédier à ce problème nous avons proposé dans le chapitre IV une seconde approche qui utilise la logique floue pour traiter ces grandeurs.

En définissant des domaines de variations de ces grandeurs nous avons pu prendre en considération les données caractéristiques de l'état réel de l'utilisateur. Ceci a permis une meilleure exploitation des informations du contexte afin de réagir selon l'état de l'utilisateur.

Assurer la sécurité de l'utilisateur est considéré comme une tâche prioritaire lors de la prise de décision et la génération des commandes. De ce fait, l'approche proposée a permis de donner une architecture qui a comme particularité :

- La prise en compte du contexte utilisateur
- La prise en compte du temps d'arrivée de la commande vocale et de l'événement geste ou clic
- Le parallélisme par le traitement simultané des données d'entrées
- La distribution. En effet, les données sont prises en compte par différents agents composés de places spatialement distribuées dans le réseau ce qui le rend plus simple et plus facile à comprendre
- L'extensibilité du système. Il est possible de changer les paramètres relatifs à l'état du patient en modifiant les paramètres des fonctions d'appartenances et d'ajouter des modalités et des données du contexte par le simple ajout d'agents spécifiques.

Nous avons testé cette seconde architecture de commande modélisée par des réseaux de Pétri sur CPNTools en réalisant une fusion entre les différentes modalités d'entrées et les données correspondant au contexte. Les résultats obtenus après simulation montrent que l'architecture mise en œuvre permet de répondre correctement aux demandes émises par l'utilisateur. De plus, elle offre une possibilité d'interaction multimodale avec un robot proche de celle utilisée entre des êtres humains. La prise en compte du contexte, dans notre cas la vérification de l'état de santé de l'utilisateur et sa localisation, permet de sécuriser le système et la localisation de l'utilisateur dans le cas d'une détection d'un problème de santé.

Parmi les perspectives offertes par notre travail nous pouvons citer:

- La possibilité d'ajouter plus de mode de commande en augmentant les modalités d'entrée.
- L'implémentation d'un dispositif d'adaptation en ligne afin de modifier automatiquement les paramètres des fonctions d'appartenances utilisées dans le traitement par logique floue des données du contexte utilisateur.

Nous rappelons en outre, que le travail réalisé dans le cas de ce mémoire de magister ne constitue qu'une première partie d'une proposition de plateforme globale composé des éléments suivants:

- Le contexte.
- La composition du service.
- La base de connaissance
- L'ontologie du domaine.

Du fait que notre travail s'est concentré sur la partie Contexte, nous pouvons donc, avoir comme perspective la continuité de ce travail par la conception des autres éléments afin d'aboutir à une plateforme complète de commande d'un robot manipulateur mobile.

Annexe 1:

Déclarations des variables des réseaux de Pétri mis en œuvre dans les chapitres III et IV

Nous présentons dans cette annexe les déclarations globales des variables utilisées dans les réseaux de Pétri présentés dans le chapitre III et chapitre IV. Le langage utilisé lors de la mise en œuvre est le CPN-ML (University of Aarhus 2006).

A. Variables utilisées dans le chapitre III:

- Colset No = int ;
- Var TempsArrive, n, mot, Tempsgs, event, Tm, Sm, GPS, Temps1, GC, Temps2, code : No ;
- Colset Parole = with parole timed;
- Colset ParoleNum1 = product Parole * No;
- Colset ParoleNumT2 = product Parole * No timed;
- Val TEntreMot = ref 5.0;
- fun round x = floor (x+ 0.5);
- fun fcExp x = round (exponential (x));
- val nMax = 100;
- val debut = 1`1;
- val debutgs = 1`1;
- colset Mot = product No * No * No timed;
- fun TempsMot() = IntInf.toInt(time());
- colset GestClic = with gestclic timed;
- val TEntreEvent = ref 20.0;
- fun Expgs x = round (exponential (x));
- colset Event = product GestClic * No timed;
- colset EventRzc = product No * No * No timed;
- fun TempsGS () = IntInf.toInt (time());
- fun rien (x) = 0;
- colset context = product No * No * No * No;
- fun fusion (a,b,c,d) = (a + b) * (c * d);
- colset fus = product No * No * No * No * No * No * No;
- colset DATA = string;
- colset CmdSortie = Product No * No* DATA;
- colset CmdSorties = product No * No * No * DATA;

B. Variables utilisées dans le chapitre IV:

- Colset No = int ;
- Var TempsArrive, n, mot, Tempsgs, event, Tm, Sm, GPS, Temps1, GC, Temps2, code, Tmf, Smf, Smfp, SmfG, erreur : No ;
- Colset Parole = with parole timed;
- Colset ParoleNum1 = product Parole * No;
- Colset ParoleNumT2 = product Parole * No timed;
- Val TEntreMot = ref 5.0;
- fun round x = floor (x+ 0.5);
- fun fcExp x = round (exponential (x));
- val nMax = 100;
- val debut = 1`1;
- val debutgs = 1`1;
- colset Mot = product No * No * No timed;
- fun TempsMot() = IntInf.toInt(time());
- colset GestClic = with gestclic timed;
- val TEntreEvent = ref 20.0;
- fun Expgs x = round (exponential (x));
- colset Event = product GestClic * No timed;
- colset EventRec = product No * No * No timed;
- fun TempsGS () = IntInf.toInt (time());
- fun rien (x) = 0;
- fun un (x) = 1;
- colset context = product No * No * No * No;
- fun fusion (a,b,c,d) = (a + b) * (c * d);
- colset fus = product No * No * No * No * No * No * No;
- colset DATA = string;
- colset CmdSortie = Product No * No* DATA;
- colset CmdSorties = product No * No * No * DATA;
- colset TF1 = product No * No * No * No;
- colset Tfuzzy = product No * No * No;

Annexe 2:

Codes obtenus lors des simulations des architectures proposées

Afin de mieux vérifier les résultats des simulations sur CPNTools des architectures proposées nous avons récupéré et comparé les codes des entrées et des sorties. Les tableaux suivants donnent les différents codes obtenus après simulation sur CPNTools. Cette présentation nous permet de vérifier pour chaque échantillon les actions demandées, le contexte et les actions générées en sorties.

Remarque:

> *Nous avons utilisé les mêmes abréviations que dans le programme (voir la liste des abréviations).*

Le tableau A1 présente les codes obtenus lors de la simulation de la première architecture (voir chapitre III). Dans cette dernière nous considérons que les variables température et tension sont de type binaire et nous supposons qu'il n'y a pas d'erreur lors de la désignation par les évènements geste ou clic d'un lieu ou d'un objet.

Les codes obtenus lors de la simulation de la seconde architecture (voir chapitre IV) sont donnés dans les Tableaux A2 et A3. La mise en œuvre de cette architecture a été réalisée en deux phases:

- Dans la première phase nous considérons que les variables température et tension sont de types réels et nous les traitons par la logique floue. Nous supposons qu'il n'y a pas d'erreur lors de la désignation d'un lieu ou d'un objet (Tableau A2).
- Dans la seconde phase nous considérons que les variables température et tension sont de types réels et nous prenons en compte les éventuelles erreurs engendrées lors de la désignation d'un lieu ou d'un objet. Le traitement de l'ensemble de ces variables est réalisé par une approche floue (Tableau A3).

Tableau A1: Codes obtenus lors de la simulation de la première architecture proposée
(Température et tension de type binaire, Pas de prise en charge de l'erreur lors de la désignation) (1/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code de l'agent TmSm avec sa signification		Code obtenu de l'agent Fusion avec sa signification
			Tm	Sm	
1	5 : Prend	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	Pas de fusion en sortie
2	5 : Prend	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
3	1 : Avance	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob Sm,GPS
4	6 : Pose	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
5	2 : Droite	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
6	1 : Avance	15 : Clic	1 : Bonne Tm	1 : Bonne Sm	16 : Avance Clic
7	6 : Pose	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
8	3 : Gauche	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	3 : Gauche
9	4 : Arrière	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
10	6 : Pose	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
11	6 : Pose	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
12	2 : Droite	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
13	1 : Avance	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
14	1 : Avance	0 : Prob loc	0 : Prob Tm	1: Bonne Sm	0: Stop, Prob Tm, GPS
15	1 : Avance	0 : Prob loc	1 : Bonne Tm	1: Bonne Sm	1: Avance

Tableau A1: Codes obtenus lors de la simulation de la première architecture proposée
(Température et tension de type binaire, Pas de prise en charge de l'erreur lors de la désignation) (2/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code de l'agent TmSm avec sa signification		Code obtenu de l'agent Fusion avec sa signification
			Tm	Sm	
16	5 : Prend	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
17	2 : Droite	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
18	3 : Gauche	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
19	0 : Stop	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
20	3 : Gauche	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
21	5 : Prend	15 : Clic	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
22	5 : Prend	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	5 : Prend
23	1 : Avance	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
24	1 : Avance	15 : Clic	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
25	3 : Gauche	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
26	3 : Gauche	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
27	0 : Stop	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
28	0 : Stop	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	0 : Stop
29	1: Avance	0 : Prob.loc	0: Prob Tm	1: Bonne Sm	0: Stop, Prob Tm, GPS
30	1 : Avance	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
31	0 : Stop	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS

Tableau A1: Codes obtenus lors de la simulation de la première architecture proposée
(Température et tension de type binaire, Pas de prise en charge de l'erreur lors de la désignation) (3/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code de l'agent TmSm avec sa signification		Code obtenu de l'agent Fusion avec sa signification
			Tm	Sm	
32	2 : Droite	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
33	2 : Droite	15 : Clic	1 : Bonne Tm	1 : Bonne Sm	17 : Droite Clic
34	3 : Gauche	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
35	1 : Avance	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	1 : Avance
36	4 : Arrière	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	4 : Arrière
37	2 : Droite	8 : Geste	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
38	3 : Gauche	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
39	2 : Droite	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
40	1 : Avance	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
41	4 : Arrière	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
42	5 : Prend	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
43	6 : Pose	15 : Clic	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
44	5 : Prend	8 : Geste	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
45	3 : Gauche	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	3 : Gauche
46	1 : Avance	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
47	2 : Droite	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS

Tableau A1: Codes obtenus lors de la simulation de la première architecture proposée
(Température et tension de type binaire, Pas de prise en charge de l'erreur lors de la désignation) (4/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code de l'agent TmSm avec sa signification		Code obtenu de l'agent Fusion avec sa signification
			Tm	Sm	
48	6 : Pose	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
49	0 : Stop	15 : Clic	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
50	5 : Prend	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
51	2 : Droite	8 : Geste	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
52	4 : Arrière	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
53	0 : Stop	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
54	3 : Gauche	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
55	2 : Droite	8 : Geste	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
56	5 : Prend	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	5 : Prend
57	4 : Arrière	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
58	6 : Pose	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	6 : Pose
59	2 : Droite	8 : Geste	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
60	3 : Gauche	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	3 : Gauche
61	1 : Avance	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0: Stop, Prob TmSm, GPS
62	5 : Prend	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
63	3 : Arrière	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS

Tableau A1: Codes obtenus lors de la simulation de la première architecture proposée
(Température et tension de type binaire, Pas de prise en charge de l'erreur lors de la désignation) (5/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code de l'agent TmSm avec sa signification		Code de l'agent Fusion avec sa signification
			Tm	Sm	
64	5 : Prend	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
65	2 : Droite	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	2 : Droite
66	6 : Pose	15 : Clic	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
67	5 : Prend	15 : Clic	1 : Bonne Tm	1 : Bonne Sm	20 : Prend Clic
68	5 : Prend	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
69	0 : Stop	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	0 : Stop
70	1 : Avance	15 : Clic	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
71	0 : Stop	8 : Geste	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
72	2 : Droite	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	2 : Droite
73	2 : Droite	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	2 : Droite
74	1 : Avance	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	1 : Avance
75	0 : Stop	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	0 : Stop
76	1 : Avance	8 : Geste	1 : Bonne Tm	1 : Bonne Sm	9 : Avance Geste
77	4 : Arrière	15 : Clic	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
78	6 : Pose	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
79	5 : Prend	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS

Tableau A1: Codes obtenus lors de la simulation de la première architecture proposée
(Température et tension de type binaire, Pas de prise en charge de l'erreur lors de la désignation) (6/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code de l'agent TmSm avec sa signification		Code obtenu de l'agent Fusion avec sa signification
			Tm	Sm	
80	4 : Arrière	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
81	2 : Droite	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
82	1 : Avance	15 : Geste	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
83	4 : Arrière	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	4 : Arrière
84	1 : Avance	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	1: Avance
85	5 : Prend	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
86	1 : Avance	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
87	0 : Stop	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
88	6 : Pose	8 : Geste	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
89	2 : Droite	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
90	2 : Droite	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
91	0 : Stop	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
92	0 : Stop	0 : Prob.loc	0 : Prob Tm	0 : Prob Sm	0 : Stop, Prob TmSm, GPS
93	4 : Arrière	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	4 : Arrière
94	1 : Avance	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
95	4 : Arrière	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS

Tableau A1: Codes obtenus lors de la simulation de la première architecture proposée
(Température et tension de type binaire, Pas de prise en charge de l'erreur lors de la désignation) (7/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code de l'agent TmSm avec sa signification		Code obtenu de l'agent Fusion avec sa signification
			Tm	Sm	
96	1 : Avance	0 : Prob.loc	0 : Prob Tm	1 : Bonne Sm	0 : Stop, Prob Tm, GPS
97	1 : Avance	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS
98	1 : Avance	0 : Prob.loc	1 : Bonne Tm	1 : Bonne Sm	1 : Avance
99	1 : Avance	0 : Prob.loc	1 : Bonne Tm	0 : Prob Sm	0 : Stop, Prob Sm, GPS

Tableau A2: Codes obtenus après simulation de la seconde structure
(Phase 1: Traitement par la logique floue de la température et de la tension) (1/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
			TmF	SmpF	SmGF	
1	0 : Stop	0 : Prob.loc	40 : F	7 : BT	9 : MT1	0 : Stop, Prob TmSm, GPS
2	6 : Pose	0 : Prob.loc	38 : N	7 : BT	12 : BT	6 : Erreur de localisation
3	5 : Prend	0 : Prob.loc	35 : B	7 : BT	11 : BT	0 : Stop, Prob
4	3 : Gauche	0 : Prob.loc	40 : F	9 : MT4	9 : MT1	0 : Stop, Prob TmSm, GPS
5	3 : Gauche	0 : Prob.loc	34 : B	4 : MT3	13 : BT	0 : Stop, Prob TmSm, GPS
6	1 : Avance	0 : Prob.loc	34 : B	5 : BT	10 : BT	0 : Stop, Prob Tm, GPS
7	5 : Prend	0 : Prob.loc	39 : F	7 : BT	11 : BT	0 : Stop, Prob Tm, GPS
8	0 : Stop	8 : Geste	34 : B	10 : MT4	11 : BT	0 : Stop, Prob TmSm, GPS
9	2 : Droite	0 : Prob.loc	35 : B	7 : BT	16 : MT2	0 : Stop, Prob TmSm, GPS
10	0 : Stop	0 : Prob.loc	36 : N	8 : BT	11 : BT	0 : Stop
11	0 : Stop	0 : Prob.loc	36 : N	4 : MT3	11 : BT	0 : Stop, Prob Sm, GPS
12	3 : Gauche	0 : Prob.loc	39 : F	10 : MT4	15 : MT2	0 : Stop, Prob TmSm, GPS
13	5 : Prend	0 : Prob.loc	34 : B	7 : BT	13 : BT	0 : Stop, Prob Tm, GPS
14	5 : Prend	0 : Prob.loc	35 : B	10 : MT4	10 : BT	0 : Stop, Prob TmSm, GPS
15	5 : Prend	0 : Prob.loc	39 : F	4 : MT3	15 : MT2	0 : Stop, Prob TmSm, GPS

Tableau A2: Codes obtenus après simulation de la seconde structure
(Phase 1: Traitement par la logique floue de la température et de la tension) (2/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
			TmF	SmpF	SmGF	
16	5 : Prend	0 : Prob.loc	34 : B	5 : BT	14 : BT	0 : Stop, Prob Tm, GPS
17	0 : Stop	0 : Prob.loc	40 : F	6 : BT	13 : BT	0 : Stop, Prob Tm, GPS
18	0 : Stop	0 : Prob.loc	36 : N	6 : BT	9 : MT1	0 : Stop, Prob Sm, GPS
19	0 : Stop	0 : Prob.loc	40 : F	7 : BT	12 : BT	0 : Stop, Prob Tm, GPS
20	3 : Gauche	0 : Prob.loc	38 : N	8 : BT	11 : BT	3 : Gauche
21	2 : Droite	15 : Clic	37 : N	8 : BT	13 : BT	17 : Droite Clic
22	3 : Gauche	8 : Geste	37 : N	5 : BT	9 : MT1	0 : Stop, Prob Sm, GPS
23	2 : Droite	0 : Prob.loc	38 : N	7 : BT	8 : MT1	0 : Stop, Prob Sm, GPS
24	5 : Prend	15 : Clic	34 : B	4 : MT3	13 : BT	0 : Stop, Prob TmSm, GPS
25	4 : Arrière	0 : Prob.loc	38 : N	7 : BT	14 : BT	4 : Arrière
26	3 : Gauche	0 : Prob.loc	39 : F	10 : MT4	12 : BT	0 : Stop, Prob TmSm, GPS
27	0 : Stop	0 : Prob.loc	37 : N	5 : BT	15 : MT2	0 : Stop, Prob Sm, GPS
28	2 : Droite	15 : Clic	40 : F	6 : BT	8 : MT1	0 : Stop, Prob TmSm, GPS
29	0 : Stop	0 : Prob.loc	34 : B	3 : MT3	14 : BT	0 : Stop, Prob TmSm, GPS
30	5 : Prend	0 : Prob.loc	40 : F	10 : MT4	9 : MT1	0 : Stop, Prob TmSm, GPS

Tableau A2: Codes obtenus après simulation de la seconde structure
(Phase 1: Traitement par la logique floue de la température et de la tension) (3/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
			TmF	SmpF	SmGF	
31	3 : Gauche	0 : Prob.loc	36 : N	5 : BT	15 : MT2	0 : Stop, Prob Sm, GPS
32	6 : Pose	8 : Geste	38 : N	5 : BT	11 : BT	14 : Pose Geste
33	4 : Arrière	0 : Prob.loc	37 : N	9 : MT4	16 : MT2	0 : Stop, Prob Sm, GPS
34	5 : Prend	8 : Geste	40 : F	5 : BT	15 : MT2	0 : Stop, Prob TmSm, GPS
35	4 : Arrière	0 : Prob.loc	39 : F	8 : BT	13 : BT	0 : Stop, Prob Tm, GPS
36	2 : Droite	8 : Geste	36 : N	7 : BT	13 : BT	10 : Droite Geste
37	0 : Stop	0 : Prob.loc	36 : N	3 : MT3	9 : MT1	0 : Stop, Prob Sm, GPS
38	5 : Prend	0 : Prob.loc	34 : B	7 : BT	13 : BT	0 : Stop, Prob Tm, GPS
39	5 : Prend	0 : Prob.loc	37 : N	9 : MT4	15 : MT2	0 : Stop, Prob Sm, GPS
40	1 : Avance	0 : Prob.loc	39 : F	4 : MT3	11 : BT	0 : Stop, Prob TmSm, GPS
41	3 : Gauche	0 : Prob.loc	39 : F	7 : BT	9 : MT1	0 : Stop, Prob TmSm, GPS
42	2 : Droite	8 : Geste	37 : N	6 : BT	10 : BT	10 : Droite Geste
43	4 : Arrière	0 : Prob.loc	35 : B	7 : BT	13 : BT	0 : Stop, Prob Tm, GPS
44	2 : Droite	0 : Prob.loc	35 : B	5 : BT	16 : MT2	0 : Stop, Prob TmSm, GPS
45	3 : Gauche	15 : Clic	35 : B	4 : MT3	13 : BT	0 : Stop, Prob TmSm, GPS

Tableau A2: Codes obtenus après simulation de la seconde structure

(Phase 1: Traitement par la logique floue de la température et de la tension) (4/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification	Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
			TmF	SmpF	SmGF	
46	5 : Prend	0 : Prob.loc	39 : F	7 : BT	13 : BT	0 : Stop, Prob Tm, GPS
47	3 : Arrière	8 : Geste	35 : B	9 : MT4	8 : MT1	0 : Stop, Prob TmSm, GPS
48	0 : Stop	0 : Prob.loc	34 : B	10 : MT4	11 : BT	0 : Stop, Prob TmSm, GPS
49	4 : Arrière	15 : Clic	36 : N	10 : MT4	11 : BT	0 : Stop, Prob Sm, GPS
50	1 : Avance	0 : Prob.loc	39 : F	5 : BT	8 : MT1	0 : Stop, Prob TmSm, GPS
51	4 : Arrière	0 : Prob.loc	39 : F	10 : MT4	15 : MT2	0 : Stop, Prob TmSm, GPS
52	6 : Pose	0 : Prob.loc	40 : F	5 : BT	16 : MT2	0 : Stop, Prob TmSm, GPS
53	1 : Avance	0 : Prob.loc	36 : N	3 : MT3	15 : MT2	0 : Stop, Prob Sm, GPS
54	1 : Avance	0 : Prob.loc	36 : N	4 : MT3	12 : BT	0 : Stop, Prob Sm, GPS
55	2 : Droite	0 : Prob.loc	37 : N	3 : MT3	12 : BT	0 : Stop, Prob Sm, GPS
56	4 : Arrière	0 : Prob.loc	39 : F	6 : BT	15 : MT2	0 : Stop, Prob TmSm, GPS
57	5 : Prend	0 : Prob.loc	39 : F	3 : MT3	14 : BT	0 : Stop, Prob TmSm, GPS
58	5 : Prend	8 : Geste	40 : F	8 : BT	14 : BT	0 : Stop, Prob Tm, GPS
59	2 : Droite	0 : Prob.loc	36 : N	4 : MT3	11 : BT	0 : Stop, Prob Sm, GPS
60	4 0 : Stop	0 : Prob.loc	37 : N	7 : BT	10 : BT	0 : Stop

Tableau A2: Codes obtenus après simulation de la seconde structure

(Phase 1: Traitement par la logique floue de la température et de la tension) (5/7)

Numéro de l'échantillon	Code obtenu de l'agent parole avec sa signification	Code obtenu de l'agent GestClic avec sa signification	Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
			TmF	SmpF	SmGF	
61	0 : Stop	0 : Prob.loc	40 : F	10 : MT4	8 : MT1	0 : Stop, Prob TmSm, GPS
62	3 : Gauche	0 : Prob.loc	36 : N	8 : BT	9 : MT1	0 : Stop, Prob Sm, GPS
63	5 : Prend	0 : Prob.loc	34 : B	6 : BT	9 : MT1	0 : Stop, Prob TmSm, GPS
64	0 : Stop	15 : Clic	40 : F	6 : BT	15 : MT2	0 : Stop, Prob TmSm, GPS
65	0 : Stop	0 : Prob.loc	39 : F	10 : MT4	10 : BT	0 : Stop, Prob TmSm, GPS
66	3 : Gauche	15 : Clic	36 : N	5 : BT	11 : BT	18 : Gauche Clic
67	2 : Droite	0 : Prob.loc	39 : F	5 : BT	13 : BT	0 : Stop, Prob Tm, GPS
68	3 : Gauche	8 : Geste	38 : N	9 : MT4	9 : MT1	0 : Stop, Prob Sm, GPS
69	2 : Droite	15 : Clic	38 : N	9 : MT4	13 : BT	0 : Stop, Prob Sm, GPS
70	5 : Prend	0 : Prob.loc	36 : N	8 : BT	12 : BT	5 : Erreur de localisation
71	0 : Stop	0 : Prob.loc	40 : F	10 : MT4	8 : MT1	0 : Stop, Prob TmSm, GPS
72	6 : Pose	15 : Clic	39 : F	6 : BT	13 : BT	0 : Stop, Prob Tm, GPS
73	4 : Arrière	0 : Prob.loc 0 :	34 : B 34 :	6 : BT	11 : BT	0 : Stop, Prob Tm, GPS
74	4 : Arrière	Prob.loc	B	8 : BT	11 : BT	0 : Stop, Prob Tm, GPS
75	3 : Gauche	0 : Prob.loc	38: N	6 : BT	13 : BT	3 : Gauche

Tabl

eau A2: Codes obtenus après simulation de la seconde structure

(Phase 1: Traitement par la logique floue de la température et de la tension) (6/7)

Numéro de l'échantillon	Code obtenu de l'agent parole avec sa signification	Code obtenu de l'agent GestClic avec sa signification	Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
			TmF	SmpF	SmGF	
76	2 : Droite	0 : Prob.loc	35 : B	10 : MT4	15 : MT2	0 : Stop, Prob TmSm, GPS
77	0 : Stop	0 : Prob.loc	36 : N	3 : MT3	10 : BT	0 : Stop, Prob Sm, GPS
78	1 : Avance	0 : Prob.loc	39 : F	5 : BT	9 : MT1	0 : Stop, Prob TmSm, GPS
79	1 : Avance	0 : Prob.loc	36 : N	6 : BT	12 : BT	1 : Avance
80	2 : Droite	0 : Prob.loc	37 : N	10 : MT4	9 : MT1	0 : Stop, Prob Sm, GPS
81	0 : Stop	15 : Clic	39 : F	10 : MT4	12 : BT	0 : Stop, Prob TmSm, GPS
82	2 : Droite	15 : Clic	39 : F	6 : BT	16 : MT2	0 : Stop, Prob TmSm, GPS
83	0 : Stop	15 : Clic	38 : N	10 : MT4	11 : BT	0 : Stop, Prob Sm, GPS
84	4 : Arrière	0 : Prob.loc	34 : B	9 : MT4	10 : BT	0 : Stop, Prob TmSm, GPS
85	0 : Stop	0 : Prob.loc	38 : N	10 : MT4	9 : MT1	0 : Stop, Prob Sm, GPS
86	4 : Arrière	0 : Prob.loc	40 : F	9 : MT4	12 : BT	0 : Stop, Prob TmSm, GPS
87	6 : Pose	15 : Clic	40 : F	8 : BT	14 : BT	0 : Stop, Prob Tm, GPS
88	5 : Prend	0 : Prob.loc	36 : N	10 : MT4	15 : MT2	0 : Stop, Prob Sm, GPS
89	5 : Prend	0 : Prob.loc	40 : F	3 : MT3	9 : MT1	0 : Stop, Prob TmSm, GPS
90	5 : Prend	0 : Prob.loc	34 : B	6 : BT	11 : BT	0 : Stop, Prob Tm, GPS

Tableau A2: Codes obtenus après simulation de la seconde structure
(Phase 1: Traitement par la logique floue de la température et de la tension) (7/7)

Numéro de l'échantillon	Code obtenu de l'agent parole avec sa signification	Code obtenu de l'agent GestClic avec sa signification	Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
			TmF	SmpF	SmGF	
91	0 : Stop	0 : Prob.loc	36 : N	8 : BT	15 : MT2	0 : Stop, Prob Sm, GPS
92	4 : Arrière	15 : Clic	40 : F	10 : MT4	14 : BT	0 : Stop, Prob TmSm, GPS
93	0 : Stop	0 : Prob.loc	37 : N	10 : MT4	8 : MT1	0 : Stop, Prob Sm, GPS
94	3 : Gauche	15 : Clic	36 : N	3 : MT3	9 : MT1	0 : Stop, Prob Sm, GPS
95	2 : Droite	0 : Prob.loc	34 : B	9 : MT4	11 : BT	0 : Stop, Prob TmSm, GPS
96	3 : Gauche	0 : Prob.loc	38 : N	6 : BT	11 : BT	3 : Gauche
97	6 : Pose	0 : Prob.loc	36 : N	4 : MT3	12 : BT	0 : Stop, Prob Sm, GPS
98	3 : Gauche	8 : Geste	35 : B	7 : BT	15 : MT2	0 : Stop, Prob TmSm, GPS
99	4 : Arrière	0 : Prob.loc	36 : N	9 : MT4	15 : MT2	0 : Stop, Prob Sm, GPS

Tableau A3: Codes obtenus après simulation de la seconde structure
(Phase 2: Traitement par la logique floue de la température, de la tension et de l'erreur lors de la désignation (1/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification en considérant l'erreur		Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
		GC	Err	TmF	SmpF	SmGF	
1	3 : Gauche	0 : P.L	10 : M	37 : N	8 : BT	12 : BT	Pas de Fusion
2	5 : Prend	0 : P.L	4 : B	40 : F	3 : MT3	9 : MT1	Pas de Fusion
3	0 : Stop	0 : P.L	3 : B	39 : F	5 : BT	16 : MT2	0 : Stop, Prob TmSm, GPS
4	3 : Gauche	0 : P.L	6 : M	34 : B	4 : MT3	15 : MT2	0 : Stop, Prob TmSm, GPS
5	6 : Pose	8 : Geste	1 : TB	37 : N	9 : MT4	15 : MT2	0 : Stop, Prob Sm, GPS
6	0 : Stop	0 : P.L	9 : M	34 : B	7 : BT	10 : BT	0 : Stop, Prob Tm, GPS
7	6 : Pose	15 : Clic	1 : TB	37 : N	7 : BT	12 : BT	21 : Pose Clic
8	1 : Avance	0 : P.L	8 : M	38 : N	9 : MT4	13 : BT	0 : Stop, Prob Sm, GPS
9	1 : Avance	0 : P.L	2 : TB	34 : B	7 : BT	9 : MT1	0 : Stop, Prob TmSm, GPS
10	2 : Droite	0 : P.L	3 : B	38 : N	6 : BT	10 : BT	2 : Droite
11	6 : Pose	0 : P.L	2 : TB	36 : N	9 : MT4	14 : BT	0 : Stop, Prob Sm, GPS
12	1 : Avance	0 : P.L	10 : M	39 : F	10 : MT4	8 : MT1	0 : Stop, Prob TmSm, GPS
13	6 : Pose	0 : P.L	8 : M	39 : F	5 : BT	13 : BT	0 : Stop, Prob Tm, GPS
14	3 : Gauche	0 : P.L	8 : M	40 : F	7 : BT	8 : MT1	0 : Stop, Prob TmSm, GPS
15	2 : Droite	0 : P.L	10 : M	39 : F	6 : BT	8 : MT1	0 : Stop, Prob TmSm, GPS

Tableau A3: Codes obtenus après simulation de la seconde structure
(Phase 2: Traitement par la logique floue de la température, de la tension et de l'erreur de désignation (2/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification en considérant l'erreur		Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
		GC	Err	TmF	SmpF	SmGF	
16	3 : Gauche	0 : P.L	2 : TB	39 : F	3 : MT3	15 : MT2	0 : Stop, Prob TmSm, GPS
17	2 : Droite	0 : P.L	7 : M	36 : N	5 : BT	11 : BT	2 : Droite
18	0 : Stop	0 : P.L	3 : B	39 : F	6 : BT	16 : MT2	0 : Stop, Prob TmSm, GPS
19	2 : Droite	0 : P.L	9 : M	38 : N	10 : MT4	10 : BT	0 : Stop, Prob Sm, GPS
20	4 : Arrière	0 : P.L	1 : TB	39 : F	4 : MT3	14 : BT	0 : Stop, Prob TmSm, GPS
21	5 : Prend	0 : P.L	1 : TB	35 : B	5 : BT	14 : BT	0 : Stop, Prob Tm, GPS
22	3 : Gauche	0 : P.L	3 : B	40 : F	3 : MT3	10 : BT	0 : Stop, Prob TmSm, GPS
23	2 : Droite	0 : P.L	6 : M	38 : N	9 : MT4	10 : BT	0 : Stop, Prob Sm, GPS
24	3 : Gauche	0 : P.L	9 : M	37 : N	10 : MT4	12 : BT	0 : Stop, Prob Sm, GPS
25	1 : Avance	0 : P.L	6 : M	39 : F	3 : MT3	13 : BT	0 : Stop, Prob TmSm, GPS
26	6 : Pose	0 : P.L	9 : M	36 : N	10 : MT4	15 : MT2	0 : Stop, Prob Sm, GPS
27	3 : Gauche	0 : P.L	7 : M	38 : N	6 : BT	14 : BT	3 : Gauche
28	2 : Droite	0 : P.L	2 : TB	36 : N	7 : BT	15 : MT2	0 : Stop, Prob Sm, GPS
29	5 : Prend	0 : P.L	8 : M	37 : N	3 : MT3	10 : BT	0 : Stop, Prob Sm, GPS
30	0 : Stop	0 : P.L	2 : TB	34 : B	5 : BT	13 : BT	0 : Stop, Prob Tm, GPS

Tableau A3: Codes obtenus après simulation de la seconde structure
(Phase 2: Traitement par la logique floue de la température, de la tension et de l'erreur de désignation (3/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification en considérant l'erreur		Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
		GC	Err	TmF	SmpF	SmGF	
31	4 : Arrière	0 : P.L	8 : M	37 : N	6 : BT	9 : MT1	0 : Stop, Prob Sm, GPS
32	1 : Avance	0 : P.L	5 : B	38 : N	9 : MT4	10 : BT	0 : Stop, Prob Sm, GPS
33	2 : Droite	0 : P.L	5 : B	38 : N	9 : MT4	15 : MT2	0 : Stop, Prob Sm, GPS
34	3 : Gauche	0 : P.L	9 : M	39 : F	3 : MT3	16 : MT2	0 : Stop, Prob TmSm, GPS
35	5 : Prend	0 : P.L	10 : M	34 : B	10 : MT4	14 : BT	0 : Stop, Prob TmSm, GPS
36	5 : Prend	0 : P.L	5 : B	35 : B	9 : MT4	8 : MT1	0 : Stop, Prob TmSm, GPS
37	2 : Droite	0 : P.L	2 : TB	40 : F	8 : BT	8 : MT1	0 : Stop, Prob TmSm, GPS
38	5 : Prend	0 : P.L	9 : M	40 : F	10 : MT4	13 : BT	0 : Stop, Prob TmSm, GPS
39	1 : Avance	15 : Clic	4 : B	34 : B	7 : BT	9 : MT1	0 : Stop, Prob TmSm, GPS
40	6 : Pose	0 : P.L	4 : B	35 : B	4 : MT3	10 : BT	0 : Stop, Prob TmSm, GPS
41	0 : Stop	15 : Clic	1 : TB	35 : B	6 : BT	13 : BT	0 : Stop, Prob Tm, GPS
42	2 : Droite	0 : P.L	1 : TB	38 : N	3 : MT3	13 : BT	0 : Stop, Prob Sm, GPS
43	5 : Prend	0 : P.L	2 : TB	34 : B	7 : BT	9 : MT1	0 : Stop, Prob TmSm, GPS
44	1 : Avance	0 : P.L	9 : M	38 : N	7 : BT	15 : MT2	0 : Stop, Prob Sm, GPS
45	0 : Stop	0 : P.L	5 : B	40 : F	3 : MT3	16 : MT2	0 : Stop, Prob TmSm, GPS

Tableau A3: Codes obtenus après simulation de la seconde structure
(Phase 2: Traitement par la logique floue de la température, de la tension et de l'erreur de désignation (4/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification en considérant l'erreur		Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
		GC	Err	TmF	SmpF	SmGF	
46	4 : Arrière	0 : P.L	2 : TB	34 : B	9 : MT4	12 : BT	0 : Stop, Prob TmSm, GPS
47	5 : Prend	0 : P.L	9 : M	40 : F	8 : BT	11 : BT	0 : Stop, Prob Tm, GPS
48	3 : Gauche	0 : P.L	3 : B	34 : B	10 : MT4	10 : BT	0 : Stop, Prob TmSm, GPS
49	6 : Pose	0 : P.L	1 : TB	40 : F	3 : MT3	13 : BT	0 : Stop, Prob TmSm, GPS
50	5 : Prend	0 : P.L	4 : B	36 : N	8 : BT	15 : MT2	0 : Stop, Prob Sm, GPS
51	1 : Avance	0 : P.L	8 : M	37 : N	8 : BT	8 : MT1	0 : Stop, Prob Sm, GPS
52	0 : Stop	0 : P.L	8 : M	38 : N	3 : MT3	15 : MT2	0 : Stop, Prob Sm, GPS
53	5 : Prend	0 : P.L	5 : B	35 : B	4 : MT3	13 : BT	0 : Stop, Prob TmSm, GPS
54	0 : Stop	0 : P.L	4 : B	37 : N	4 : MT3	11 : BT	0 : Stop, Prob Sm, GPS
55	4 : Arrière	0 : P.L	5 : B	40 : F	9 : MT4	15 : MT2	0 : Stop, Prob TmSm, GPS
56	5 : Prend	0 : P.L	10 : M	40 : F	6 : BT	15 : MT2	0 : Stop, Prob TmSm, GPS
57	3 : Gauche	0 : P.L	6 : M	36 : N	5 : BT	9 : MT1	0 : Stop, Prob Sm, GPS
58	5 : Prend	0 : P.L	6 : M	40 : F	3 : MT3	13 : BT	0 : Stop, Prob TmSm, GPS
59	5 : Prend	0 : P.L	9 : M	35 : B	8 : BT	10 : BT	0 : Stop, Prob Tm, GPS
60	4 : Arrière	0 : P.L	9 : M	36 : N	5 : BT	15 : MT2	0 : Stop, Prob Sm, GPS

Tableau A3: Codes obtenus après simulation de la seconde structure
(Phase 2: Traitement par la logique floue de la température, de la tension et de l'erreur de désignation (5/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification en considérant l'erreur		Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
		GC	Err	TmF	SmpF	SmGF	
61	4 : Arrière	0 : P.L	8 : M	35 : B	7 : BT	10 : BT	0 : Stop, Prob Tm, GPS
62	2 : Droite	15 : Clic	3 : B	36 : N	6 : BT	12 : BT	17 : Droite Clic
63	1 : Avance	0 : P.L	2 : TB	37 : N	5 : BT	12 : BT	1: Avance
64	3 : Gauche	0 : P.L	2 : TB	40 : F	9 : MT4	11 : BT	0 : Stop, Prob TmSm, GPS
65	3 : Gauche	0 : P.L	10 : M	34 : B	4 : MT3	15 : MT2	0 : Stop, Prob TmSm, GPS
66	1 : Avance	8 : Geste	5 : B	38 : N	3 : MT3	10 : BT	0 : Stop, Prob Sm, GPS
67	3 : Gauche	0 : P.L	8 : M	40 : F	6 : BT	11 : BT	0 : Stop, Prob Tm, GPS
68	3 : Gauche	8 : Geste	2 : TB	34 : B	6 : BT	9 : MT1	0 : Stop, Prob TmSm, GPS
69	0 : Stop	0 : P.L	5 : B	38 : N	6 : BT	12 : BT	0 : Stop
70	3 : Gauche	0 : P.L	8 : M	39 : F	8 : BT	15 : MT2	0 : Stop, Prob TmSm, GPS
71	0 : Stop	0 : P.L	8 : M	39 : F	10 : MT4	16 : MT2	0 : Stop, Prob TmSm, GPS
72	3 : Gauche	0 : P.L	7 : M	37 : N	8 : BT	10 : BT	3 : Gauche
73	4 : Arrière	0 : P.L	3 : B	40 : F	5 : BT	11 : BT	0 : Stop, Prob Tm, GPS
74	4 : Arrière	0 : P.L	5 : B	40 : F	10 : MT4	14 : BT	0 : Stop, Prob TmSm, GPS
75	3 : Gauche	0 : P.L	4 : B	36 : N	7 : BT	15 : MT2	0 : Stop, Prob Sm, GPS

Tableau A3: Codes obtenus après simulation de la seconde structure
(Phase 2: Traitement par la logique floue de la température, de la tension et de l'erreur de désignation (6/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification en considérant l'erreur		Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
		GC	Err	TmF	SmpF	SmGF	
76	4 : Arrière	8 : Geste	1 : TB	36 : N	10 : MT4	15 : MT2	0 : Stop, Prob Sm, GPS
77	1 : Avance	0 : P.L	4 : B	40 : F	8 : BT	16 : MT2	0 : Stop, Prob TmSm, GPS
78	4 : Arrière	0 : P.L	3 : B	34 : B	6 : BT	12 : BT	0 : Stop, Prob Tm, GPS
79	5 : Prend	8 : Geste	4 : B	35 : B	10 : MT4	8 : MT1	0 : Stop, Prob TmSm, GPS
80	4 : Arrière	0 : P.L	7 : M	39 : F	9 : MT4	8 : MT1	0 : Stop, Prob TmSm, GPS
81	2 : Droite	0 : P.L	4 : B	38 : N	4 : MT3	12 : BT	0 : Stop, Prob Sm, GPS
82	6 : Pose	0 : P.L	3 : B	34 : B	8 : BT	14 : BT	0 : Stop, Prob Tm, GPS
83	0 : Stop	0 : P.L	8 : M	38 : N	3 : MT3	10 : BT	0 : Stop, Prob Sm, GPS
84	4 : Arrière	0 : P.L	4 : B	37 : N	3 : MT3	16 : MT2	0 : Stop, Prob Sm, GPS
85	3 : Gauche	0 : P.L	5 : B	35 : B	9 : MT4	15 : MT2	0 : Stop, Prob TmSm, GPS
86	3 : Gauche	8 : Geste	3 : B	40 : F	5 : BT	10 : BT	0 : Stop, Prob Tm, GPS
87	4 : Arrière	0 : P.L	4 : B	39 : F	4 : MT3	13 : BT	0 : Stop, Prob TmSm, GPS
88	2 : Droite	0 : P.L	10 : M	34 : B	8 : BT	14 : BT	0 : Stop, Prob Tm, GPS
89	6 : Pose	0 : P.L	2 : TB	40 : F	10 : MT4	11 : BT	0 : Stop, Prob TmSm, GPS
90	0 : Stop	0 : P.L	6 : M	35 : B	4 : MT3	13 : BT	0 : Stop, Prob TmSm, GPS

Tableau A3: Codes obtenus après simulation de la seconde structure
(Phase 2: Traitement par la logique floue de la température, de la tension et de l'erreur de désignation (7/7)

Numéro de l'échantillon	Code de l'agent parole avec sa signification	Code de l'agent GestClic avec sa signification en considérant l'erreur		Code obtenu de l'agent TmSm avec sa signification			Code obtenu de l'agent Fusion avec sa signification
		GC	Err	TmF	SmpF	SmGF	
91	3 : Gauche	0 : P.L	2 : TB	34 : B	7 : BT	10 : BT	0 : Stop, Prob Tm, GPS
92	0 : Stop	0 : P.L	5 : B	35 : B	5 : BT	13 : BT	0 : Stop, Prob Tm, GPS
93	4 : Arrière	0 : P.L	9 : M	36 : N	6 : BT	16 : MT2	0 : Stop, Prob Sm, GPS
94	2 : Droite	15 : Clic	3 : B	36 : N	8 : BT	15 : MT2	0 : Stop, Prob Sm, GPS
95	4 : Arrière	0 : P.L	10 : M	34 : B	6 : BT	10 : BT	0 : Stop, Prob Tm, GPS
96	6 : Pose	0 : P.L	8 : M	40 : F	3 : MT3	10 : BT	0 : Stop, Prob TmSm, GPS
97	6 : Pose	0 : P.L	5 : B	38 : N	6 : BT	8 : MT1	0 : Stop, Prob Sm, GPS
98	3 : Gauche	0 : P.L	3 : B	36 : N	3 : MT3	8 : MT1	0 : Stop, Prob Sm, GPS
99	5 : Prend	0 : P.L	7 : M	36 : N	7 : BT	15 : MT2	0 : Stop, Prob Sm, GPS

Annexe 3:

Présentation de l'outil CPNTools et du langage de spécification ML

Nous présentons dans cette annexe l'outil CPNTools utilisé lors de la mise en œuvre des architectures proposées. Nous donnerons une introduction à la syntaxe ML utilisé lors de la programmation.

1 **Introduction :**

Le logiciel CPNTools est un outil de représentation graphique de réseau de Pétri. Il permet de concevoir et de simuler des réseaux qui modélisent des systèmes. Le choix de cet outil est dû au fait de plusieurs facteurs, nous pouvons citer :

- L'outil est gratuit et peut être obtenu par une demande de licence à partir du site web consacré à cet outil [34]
- La prise en compte du temps et du parallélisme
- Possibilité d'offrir des graphes hiérarchiques
- Possibilité d'introduire l'aspect stochastique dans les réseaux

De plus, CPNTools est un outil informatique interactif pour la modélisation et la simulation des réseaux de Pétri [35]. Il fournit :

- Un éditeur pour créer et manipuler des réseaux de Pétri (RP).
- Un vérificateur de syntaxe pour la validation des RP.
- Un simulateur pour l'exécution des RP.
- Une vérification interactive et possibilité de débogage.
- Organisation facile des RP en un ensemble hiérarchique de modules.
- Possibilité d'afficher les résultats de la simulation.

Ces capacités permettent de pouvoir facilement créer des RP et de les modifier selon le choix du concepteur.

2 **Présentation du logiciel CPNTool :**

2.1 **Description de l'interface de CPNTools** :

L'interface CPNTools est une fenêtre qui contient un index sur la gauche et un espace de travail sur la droite comme le montre la figure suivante :

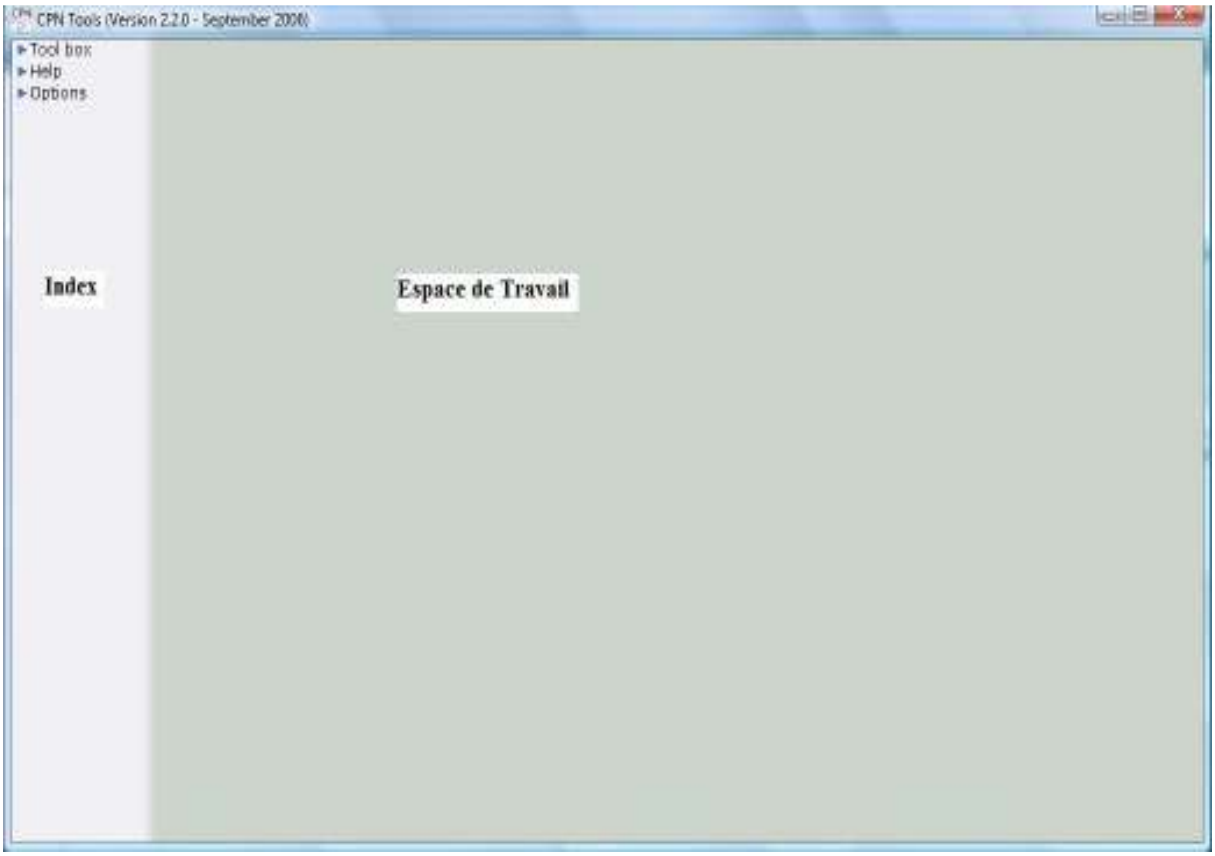

Figure A1 : Fenêtre obtenu après lancement de CPNTools

Pour pouvoir ouvrir ou charger un réseau il suffit de cliquer sur l'espace de travail avec le bouton droit. Ou encore, pour pouvoir faire des modifications sur ces réseaux il suffit de défiler dans les options que nous offre l'index. Les concepteurs de CPNTools nous ont permis d'avoir accès à une bonne documentation qui est disponible gratuitement sur leur site (veuillez consulter l'adresse fournie dans [36]. Nous avons dans [30] une bonne description de la façon d'utiliser cet outil et une présentation de quelques options que nous offre CPNTools, un exemple est donné dans la figure suivante :

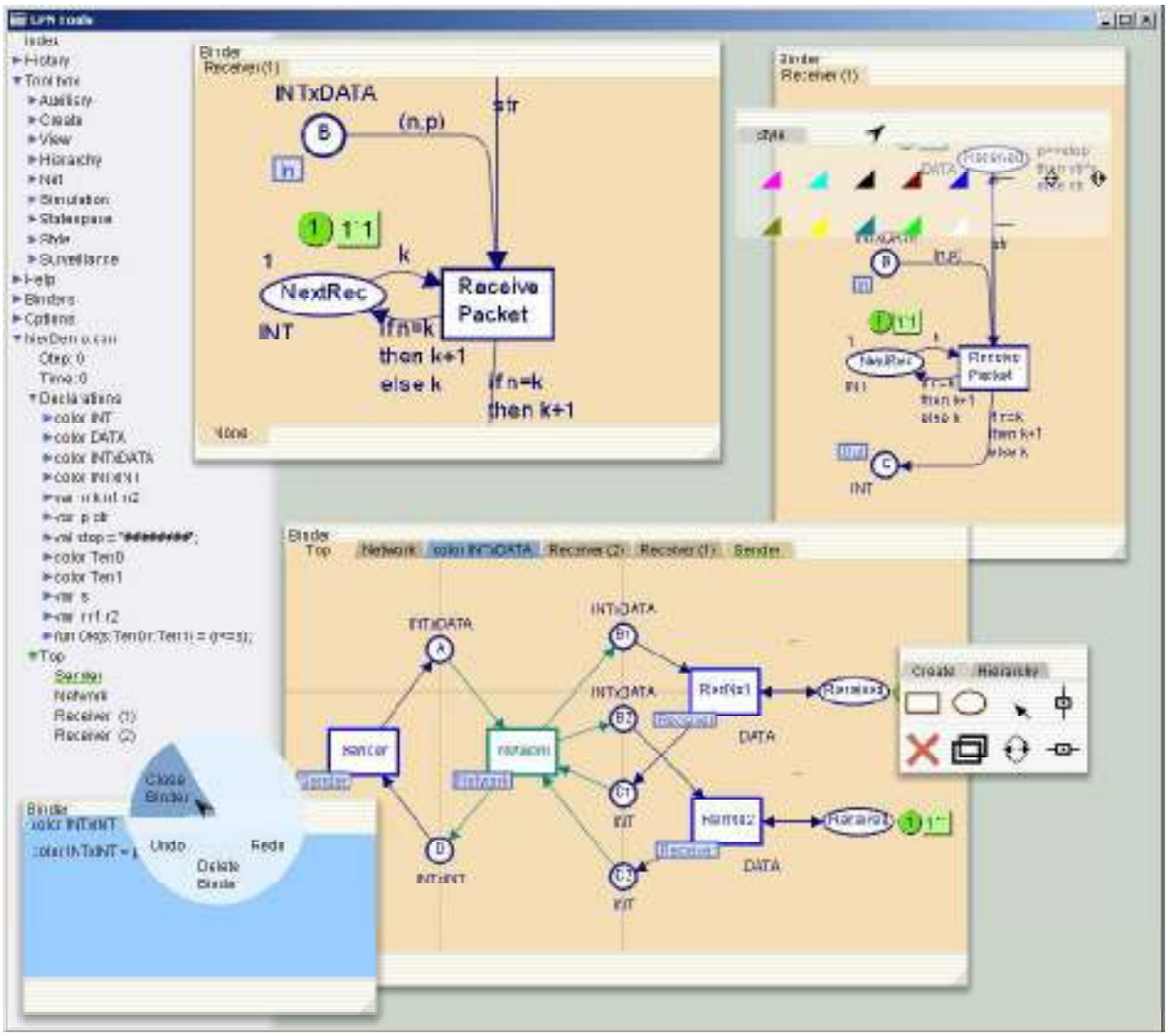

Figure A2 : Présentation de l'interface de CPNTools

2.2 **Composant d'un réseau de Pétri** :

Un réseau de Pétri est une structure graphique associé à un langage informatique de déclaration. Les principales composantes d'un réseau de Pétri sont :

- Les données: Les RP utilisent des types de données, des fonctions et des variables qui représentent les valeurs des données. Les données CPN sont définies par un langage informatique appelé CPN ML.
- Places: Location de conservation des données.
- Transitions: activités qui transforment les données.
- Arcs: Connecte les places avec les transitions pour pouvoir préciser le chemin des flux de données.

- Inscription des arcs d'entrées : Spécifie les données qui doivent exister pour qu'une transition soit franchie.
- Guards: Définit les conditions qui doivent être remplies pour qu'une activité peut se produire.
- Inscriptions des arcs de sortie : Spécifie les données qui seront produites si une transition est franchie.

3 **Introduction à la syntaxe ML** :

Nous allons présenter dans cette partie le langage de programmation CPN ML utilisé dans la déclaration des variables, fonctions, inscription …etc. Nous précisons que nous allons nous intéresser à quelques aspects utiles à notre travail. Plus de précision sont fourni dans d'autres travaux comme c'est le cas dans [28] et [35]. Notamment, tous les types de données CPN et les variables doivent être déclarées. Ils le sont dans une boîte de déclaration appelé « global déclaration node » et qui d'après le manuel de référence [35], peut contenir les éléments suivants

3.1 Colorset : Les données d'un réseau peuvent être de tous les types de données

généralement disponibles dans un langage informatique: entiers, réels, les chaînes, les booléens, les listes, et ainsi de suite. Dans le contexte des réseaux de Pétri, ces types sont appelés **colorsets**. Tout les colorset utilisé dans un réseau doivent être déclaré, la syntaxe utilisée dans ce cas est :

Colset nom = définition

Où :

- nom : représente le nom de la variable que l'utilisateur à choisi
- définition : représente le type de cette variable : int, string…etc.

Ainsi, nous pouvons avoir des jetons qui ont une série de déclarations prédéfinie de valeurs littérales. Ces valeurs sont déclarées dans un **colorset énuméré**, la syntaxe utilisée est la suivante :

colset nom = **with** valeur {| valeur}...;

Nous pouvons donner comme exemple les cas suivant :

- **colset** Parole = **with** parole timed ; (parole est un symbole dépendant du temps)
- **colset** Booln **=** **with** oui | non;

Remarque:

> Le langage voit une différence entre les majuscules et les minuscules.

De plus, nous avons des colorset qui sont composé de plusieurs autre colorsets. CPN ML peut fournir plusieurs compositions de colorset : records, lists, tuple…etc. Le colorset tuple est un colorset composé qui permet de réaliser un produit cartésien de deux ou plusieurs autres colorsets. La syntaxe utilisée est la suivante :

colset nom = **product** colset1 ***** colset2

Exemple:

colset No = Int;

colset Mot = **product** No ***** No ***** No timed;

Remarque :

> Nous précisions que le mot réservé « timed » est ajouté à la fin du colorset pour rendre le marquage de la place temporisé. Cet apport est possible par le fait que le simulateur possède un compteur qui indique la valeur courante du temps depuis le début de la simulation. Donc une transition est franchissable lorsque son étiquette temporelle à une valeur inférieur ou égale à celle du compteur.

3.2 Valeur: Dans CPN, les réseaux ont souvent besoin d'accéder à des valeurs de jetons pour être utilisés de différentes façons. Ces valeurs doivent être prédéfinies dans le réseau. La syntaxe utilisée est la suivante :

val nom de la valeur = valeur

Exemple :

val nMax = 100 ;

3.3 Variable : Les variables sont utilisées dans les RP comme dans tout programme. Ce sont des déclarations qui vont porter des types de données différents et qui vont permettre le déroulement du programme. La syntaxe utilisée est la suivante :

var nom de la variable : type de la variable

Exemple :

var GC : No ;

3.4 Fonction : Dans les réseaux de Pétri, lorsqu'un jeton franchie une transition, ce dernier peut être assujetti à des modifications de type ou de valeur. Cette modification ce produit lors de l'exécution d'un code écrit en ML. Ce code peut être une suite d'instruction ou des fonctions. La syntaxe utilisée pour la déclaration de fonction est la suivante :

fun nom de la fonction = la fonction

Exemple :

fun fcExp (x) = round (exponentiel (x)).

Lors de son utilisation dans le réseau, nous avons la syntaxe suivante :

fcExp (1.0/(! TEntreMot)) ;

Cette fonction fait appel à une valeur TEntreMot qui est précisé dans les déclarations par le mot clé **ref** (correspond à un pointeur). Cette valeur est employée précédé par un point d'exclamation pour pouvoir y accéder.

Remarque:

> Nous précisions que nous avons utilisé dans notre travail des fonctions prédéfinies dans CPN-Tools, comme :

- La fonction exponentiel : fun exponentiel (r : real) : real
  Cette fonction permet d'avoir une distribution exponentielle de moyenne 1/r (avec r : réel).
- La fonction floor : cette fonction permet d'arrondir un réel en entier
- La fonction IntInf.toInt( ) : cette fonction permet la conversion d'un réel en entier. Comme par exemple : fun TempsMot( ) = IntInf.toInt (time()) ;

Où time ( ) est une fonction sans argument qui permet de retourner la valeur courante du temps.

3.5 Temporisation des arcs : pour pouvoir prendre en compte le temps dans les RP, CPNTools offre une possibilité de temporisation des arcs. Nous avons deux types d'arcs :

- Arc d'entré : ce sont les arcs qui arrivent sur une transition
- Arc de sortie : ce sont les arcs qui sortent d'une transition

Nous pouvons rajouter une temporisation sur ces arcs, cette temporisation est écrite sur l'arc par la syntaxe « @ + Temps », ou Temps est une unité de temps.

Bibliographie

BIBLIOGRAPHIE:

[1] : **Hok KWEE et Jan DUIMEL, HV Hoensbroek, Jan Smits et Arjen Tuinhof de Moed Produktcentrum-TNO, Delft, Jan van Woerden, LW. val. Kolk et J.O. (Rosier TPD-TNO, Delft, Pays-Bas. J.-C. Cunin, AFM, 91000 Evry, France)**: *« MANUS - Robot adapté sur fauteuil, premières expériences et collaboration internationale* », Actes du colloque « Tome I » HANDITEC. Les Technologies au service des personnes handicapées Moteurs. (décembre 1989)

[2]: **Nicolas BIARD (Université René Descartes Paris 5) : «** *Implémentation d'une commande référencée vision sur le bras manipulateur Manus : Evaluation du projet AVISO* **».** Mémoire de Master 2 Science de la Motricité Mention Vieillissement, Handicap : Mouvement et Adaptation Robotique d'assistance et compensation des limitations des préhensions**.** (2007-2008).

[3]: **Jong-Hwan Kim, Yong-Duk Kim, and Kang-Hee Lee (Robot Intelligence Laboratory, KAIST, Yuseong-gu, Daejeon 305-701, Republic of Korea)**: **«** *The Third Generation of Robotics: Ubiquitous Robot* **».** 2nd International Conference on Autonomous Robots and Agents, Palmerston North, New Zealand. (December 13-15, 2004).

[4]: **ENNAJI Mourad (LASC Université de Metz Ile du Saulcy. METZ) : «** *Modélisation agent d'une Architecture Logicielle de commande d'un Véhicule Autonome* », Publication IRISA. (2000)

[5] : **Alexandre Abellard, Mohamed Ben Khelifa, Moez Bouchouicha, Patrick Abellard (Université de Toulon-Var, France) :** « *Modélisation par réseaux de Pétri pour une programmation VHDL. Exemple d'application en robotique mobile d'assistance au handicap* », Laboratoire SIS/AI. International Journal of Info & Com Sciences for decision making. ISSN : 1265- 499X, N° 13. (Janvier 2004)

[6] : **Emrah Akin Sisbot (LAAS/CNRS)** : « *Planification de Mouvement pour l'Interaction Homme-Robot (Motion Planning for Human Robot Interaction)* », Colloque des Doctorants EDIT, Toulouse France. (2007)

[7] : **Colle Etienne, (IBISC, CNRS)** : « *Interaction homme-robot dans le cadre de la personne handicapée ou dépendante*», CNRS UEVE-FRE 2873. (JNRR '07, 9-12 octobre 2007).

[8] : **HICHAM DJENIDI (Ecole de technologie supérieure Université du QUÉBEC)**: « *Architectures Logicielles Dynamiques Dédiées aux Applications Multimodales* », thèse en cotutelle avec l'Université de Versailles Saint-Quentin en Yvelines, pour l'obtention du doctorat en génie de l'ETS et du doctorat en informatique de l'Université de Versailles Saint-Quentin en Yvelines, France. (2007)

[9]: **Richard A. Bolt:** «*Voice and Gesture at the Graphics Interface*». Computer Graphics Journal of the Association of Computing and Machinery **14**(3): 262- 270. (1980)

[10]: **Philip COHEN, David McGEE, Josh CLOW (Center for Human-Computer Communication Oregon Graduate Institute of Science & Technology):** «The Efficiency of Multimodal Interaction for a Map-based Task», Published in: Proceeding ANLC '00 Proceedings of the sixth conference on applied natural language processing (2000).

[11]: **Dennis Perzanowski, Alan C. Schultz, William Adams, Elaine Marsh, and Magda Bugajska. (Naval Research Laboratory, Washington)***:* **«** *Building a Multimodal Human–Robot Interface* », Journal IEEE Intelligent Systems, archive volume 16 Issue 1, (Janvier 2001)

[12]: **Manolo Dulva HINA (Ecole de Technologie Supérieure Université du QUEBEC)**: « *A Paradigm of an Interaction Context-Aware Pervasive Multimodal Multimedia Computing System*», Manuscript-based thesis presented to Ecole de Technologie Supérieure Université de Versailles-Saint-Quentin-en-Yvelines (Cotutorship). In partial fulfillment of the requirements for the degree of doctor of philosophy PHD. (Juin 2010)

[13] : **Stephane. Lallee et Peter Ford Dominey (Stem Cell and Brain Research Institute, INSERM U846. France), Anthony Mallet et Eiichi Yoshida (Laboratoire d'Analyse et Architecture des Systèmes, CNRS), Francesco Nori, Lorenzo Natale et Giorgio Metta (Italian Institute of Technology, Central Research Labs, Genoa Headquarter. Italy), Felix Warneken (Department of**

Developmental and Comparative Psychology, Max Planck Institute for Evolutionary Anthropology, Germany) : « *Human-Robot Cooperation Based on Interaction Learning*», in From Motor Learning to Interaction Learning in Robots, Studies in Computational Intelligence SCI 264, pp. 491–536, Springer-Verlag Berlin Heidelberg. (2010)

[14]: **Michael Johnston (AT&T Labs Research, Florham Park, NJ): «** *Building Multimodal Applications with EMMA*», *ICMI-MLMI'09,* November 2–4, 2009, Cambridge, MA, USA. Copyright 2009 ACM 978-1-60558-772-1/09/11. Proceedings of the 2009 international conference on Multimodal interfaces.

[15]: **Frédéric Landragin (LATTICE, Montrouge, France):** « *Physical, semantic and pragmatic levels for multimodal fusion and fission*», halshs-00138503, version 1 - 26 Mar 2007 (Author manuscript, published in "(2007)").

[16]: **Manuel Giuliani, Alois Knoll** (Robotics and Embedded Systems Group Department of Informatics Technische Universität München, Germany: « *MultiML – A General Purpose Representation Language for Multimodal Human Utterances* », ICMI'08, October 20–22, 2008, Chania, Crete, Greece. Copyright 2008 ACM 978-1-60558-198-9/08/10 (2008).

[17]: **Daniel Salber (IBM T.J. Watson Research Center, NY, USA): «** *Context-Awareness and Multimodality* », *Colloque sur la multimodalité, IMAG, Grenoble* (Mai 2000)

[18]: **Laurence Nigay1, Phil Gray (Computing Science Dept, University of Glasgow, Scotland**: « *Architecture Logicielle Conceptuelle pour la Capture de Contexte*», *IHM 2002*, November 26-29, 2002, Poitiers, FRANCE. Copyright 2002 ACM 1-58113-615-3/02/0011.

[19] : **Jérôme Euzenat (1INRIA Rhône-Alpes), Jérôme Pierson et Fano Ramparany (France Telecom R&D)** : « *Gestion dynamique de contexte pour l'informatique diffuse*», Conférence en Reconnaissance des Formes et Intelligence Artificielle (rfia 2006).

[20] : **Atef Zaguia (Ecole de technologie supérieure université du QUÉBEC) : «** *Fusion Multimodale et Services Web* », mémoire présenté à l'école de technologie supérieure

comme exigence partielle à l'obtention de la maîtrise en génie de la production automatisée, M. Ing. (Octobre 2010).

[21]: **Gaëtan Rey, Joëlle Coutaz (CLIPS-IMAG, Grenoble, France) :** « *Le Contexteur : Capture et distribution dynamique d'information contextuelle* », *Mobilité & Ubiquité '04*, June 1-3, 2004, Nice, France Copyright ACM 1-58113-915-2/04/0006. (2004)

[22]: **Patrick Brézillon (Université Paris 6)** : « *Représentation de pratiques dans le formalisme des graphes contextuels* », EPIQUE. (2003)

[23] : **Patrick Brézillon, Emilie Marquois (Université Paris 6)** : « *Une approche centrée contexte de l'activité* », In J.M.C. Bastien (eds), actes dy Symposium "Tâche, Activité et Contexte". 2èmes Journées d'Etudes en Psychologie Eregonomique, pp. 263--268, INRIA. (2003)

[24]: **Anders Kofod-Petersen, Marius Mikalsen (Norwegian University of Science and Technology, Trondheim, Norway)**: « *Context: Representation and Reasoning. Representing and Reasoning about Context in a Mobile Environment*», revue d'intelligence artificielle. (2005)

[25]: **Rene David, Hassane Alla (INP Grenoble Laboratoire d'Automatique de Grenoble ENSIEG, France)** : « *Discrete, Continuous, and Hybrid Petri Nets* », ISBN 3-540-22480-7 Springer Berlin Heidelberg New York. Library of Congress Control Number: 2004110950. (2005)

[26]: **Michel ACHARD, Eric LEE, David LOPES, David MILLET (Direction des Organes / PSA PEUGEOT CITROEN, Direction de l'Innovation et de la Qualité PSA PEUGEOT CITROËN, France)**: «*Les réseaux de Pétri et les plans d'expériences pour l'allocation d'objectifs de fiabilité et le choix de maintenance préventive* », Article du congrès QUALITA 2001.

[27]: **Nicolas Rivière (Université Paul Sabatier de Toulouse)** : « *Modélisation et analyse temporelle par réseaux de Pétri et logique linéaire»*, Thèse préparée au Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS en vue de l'obtention du Doctorat de l'Institut National des Sciences Appliquées de Toulouse. Spécialité : Systèmes Informatiques. (2003)

[28]: **Kurt Jensen, Lars Michael Kristensen and Lisa Wells Department of Computer Science University of Aarhus, DENMARK)**: «*Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems* », (Int J Softw Tools Technol Transfer, DOI 10.1007/s10009-007-0038-x. (2007)

[29]: **Robert Harper (Carnegie Mellon University)**: «*Programming in Standard ML* », Working draft of August 20. (2009)

[30]: **Anne Vinter Ratzer, Lisa Wells, Henry Michael Lassen, Mads Laursen, Jacob Frank Qvortrup, Martin Stig Stissing, Michael Westergaard, Søren Christensen, Kurt Jensen (Department of Computer Science, University of Aarhus, Denmark)**: «*CPN Tools for Editing, Simulating, and Analysing Coloured Petri Nets*», Applications and Theory of Petri Nets 2003: 24th International Conference, ICATPN 2003.

[31]: **Michel Beaudouin-Lafon, Wendy E. Mackay, Peter Andersen, Paul Janecek, Mads Jensen, Michael Lassen, Kasper Lund, Kjeld Mortensen, Stephanie Munck, Anne Ratzer, Katrine Ravn, Søren Christensen and Kurt Jensen (Department of Computer Science University of Aarhus – Denmark)**: «*CPN/Tools: A Post-WIMP Interface for Editing and Simulating Coloured Petri Nets*», Coloured Petri Nets, Application and Theory of Petri Nets 2001, Proceedings of the 22nd International Conference, ICATPN 2001 Newcastle upon Tyne.

[32]: **Pierre-Michel Ricordel (Institut National Polytechnique de Grenoble)**: « *Programmation Orientée Multi-Agents : Développement et Déploiement de Systèmes Multi-Agents Voyelles* », T h e s e pour obtenir le grade de Docteur de l'INPG spécialité : « informatique : systèmes et communication » Préparée au laboratoire Leibniz dans le cadre de l'école doctorale « Mathématiques, Sciences et Technologies de l'Information, Informatique » (25 octobre 2001).

[33]: **Timothy Ross (Stanford University Professor, Dept. Of civil engineering,**

University of New Mexico): «Fuzzy Logic for Engineering Applications-2nd Edition», Wiley Edition, UK, ISBN 0-470-86074-X (2004)

[34]: http://wiki.daimi.au.dk/cpntools/cpntools.wiki

[35]: **Design/CPN (a trademark of Meta Software Corporation), Meta Software Corporation Cambridge, MA, USA**: « *Design/CPN Reference Manual for X-Windows Version 2.0* ». (1993)

[36]: http://wiki.daimi.au.dk/cpntools-help/getting_started_with_cpn_.wiki

Résumé :

Avec l'essor des applications dites « intelligentes », un des défis de la recherche concernant la multimodalité est l'élaboration de solutions architecturales qui répondent et qui s'adaptent à différentes types de contraintes dans le contexte d'interaction en robotique.

Aujourd'hui encore, de nombreuses questions se posent quant à la résolution de l'interaction multimodale personne-robot aussi bien au niveau conceptuel et logiciel qu'au niveau matériel. Les solutions proposées sont souvent ad hoc et relativement peu de travaux offrent des propositions « d'architectures logicielles » pour de telles applications. Lors de la conception d'une application intelligente multimodale il est nécessaire d'analyser les tâches effectuées par le robot (ou qu'il aura à effectuer), d'identifier ses besoins et de spécifier toutes les contraintes éventuellement, matérielles et/ou autres (exemples: environnement bruité, accès difficiles, comportement, etc.). Ceci permet d'élaborer les différents liens qu'entretiennent les messages en entrée du robot et, de les combiner de façon intelligente dans une architecture du dialogue « personne-robot ». Ces architectures doivent s'adapter continuellement aux changements dus aux perturbations externes ou aux actions de l'utilisateur. Elles sont donc assujetties à des contraintes en cours d'utilisation (en temps réel) lors du dialogue de robot avec l'utilisateur en particulier et avec l'environnment en général.

L'objectif principal de ce sujet est de proposer un modèle d'architecture logicielle adaptable et qui permettra au robot d'utiliser plusieurs modalités et de faire la fusion entre ces données pour augmenter son interaction avec l'environnement tout en prenant compte du contexte. Nous avons aussi introduit la logique floue dans le traitement des données issues de plusieurs modalités d'entrées. Cette architecture logicielle multimodale qui prend en compte le contexte, a été modélisée à base de réseaux de Petri colorés temporisés stochastiques (RPCTS). La simulation se fera avec l'outil CPN-Tools.