République Démocratique et Populaire d'Algérie
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Mohamed Khider - BISKRA
Faculté des sciences exactes, des sciences naturelles et de la vie
Département informatique

Numéro de commande : GLSD2/M2/2018

Mémoire

Présenté pour obtenir le diplôme de Master académique en

Informatique

Option : **Génie logiciel et systèmes distribués**

Spécification et vérification d'un Système mobile

---

par :
Bida Yassine

Soutenu le 25/06/2018, devant le jury composé de :

Bourekkache Samir	MAA	Président
Mohammedi Amira	MAA	Superviseur
Zouiouche Amina	MAA	examinateur

Année universitaire : 2017/2018

Remerciements

Louange à ALLAH, le Compatissant, le Miséricordieux. Paix et bénédiction sur le Messager d'Allah, Muhammad le prophète (paix soit sur lui). Je souhaite exprimer ma gratitude à ALLAH pour sa bénédiction et son inspiration qui m'ont conduit à terminer ce travail.

Mes remerciements et mon appréciation à mon superviseur **Mme.Mohammedi Amira** pour ses encouragements continus, ses conseils et pour sa patience et ses précieux conseils.

Je voudrais exprimer mes plus sincères remerciements aux membres du jury : **M.Bourekkach Samir** et **M. Zouioueche Amina** pour avoir lu et évalué ma dissertation.

Je n'oublie jamais de remercier tous mes professeurs du département d'informatique qui m'ont enseigné les principes de base de l'informatique.

Un grand merci à toute ma famille grands et petits (mes chères parents mes beaux-parents, mes frères et soeurs et tous les autres), pour leurs amour et encouragements.

tiens également à remercier tous ceux qui m'ont encouragée et aidée de près ou de loin tout au long de mon travail, et tout particulièrement, ma collègue et amie,Melle.Belaiche Laila Rania Hamlaoui et mon collègue Allia Abd ennacer et Chebira Abderrahman pour sa compagnie, ses critiques, ses conseils.

Résumé

Les systèmes mobiles sont des systèmes où les entités, qui les composent, peuvent changer de localité durant leurs existences. Nous avons étudié la spécification et la vérification formelles de migration et de la communication dans un réseau mobiles.Le Régulateur de vitesse adaptatif(adaptatif cruise control ACC) est utilisé pour maintenir la vitesse d'un véhicule dans un intervalle prédéfini.Nous nous somme basé sur les travaux présentés dans (G. Ciobanu et.RUSU)[37].qui fournissent des spécifications incomplètes.Nous avons complété les spécifications système(ACC) en utilisant l'algèbre de processus π-calcul basée sur la mobilité de lien .Puis nous avons expérimenté la vérification de certaine propriétés. Nous avons présenté des diagrammes de séquence,la sémantique formelle,quelques propriétés, vérification par l'analyse de Mobility Workbench.Mobility Workbench [12] est utilisé comme un outil logiciel pour la vérification.

Table des matières

Liste des tableaux

Table des figures

Introduction générale

L'évolution des réseaux à grande échelle a permis la naissance d'un grand nombre de nouvelles applications.Ces applications exigent une forte interaction entre différentes entités distribuées dans le réseau, celles-ci partagent les mêmes ressources et visent à atteindre les mêmes buts. Ces applications mobiles adoptent une multitude de modèles d'exécution distribués, proposés dans la littérature[9]. Ces systèmes sont apparus depuis longtemps, et ont connu un grand développement avec l'évolution en informatique, électronique, et mécanique. Les systèmes de robots mobiles, les réseaux mobiles cellulaires, et enfin les logiciels ou agents mobiles sont les exemples les plus remarquable dans le domaine de la mobilité.

Les algèbres de processus on retrouve parmi lesquelles CCS[11] et leπ-calcul[14] représentent un cadre naturel pour décrire et analyser les systèmes répartis et mobiles. Elles fournissent plusieurs descriptions d'un même système à des niveaux d'abstraction différents, et des techniques permettant de montrer leur équivalence.

Le π-calcul est une extension de CCS, dans la mesure où les processus peuvent échanger des noms de canaux. Cette possibilité augmente le pouvoir expressif de ce langage, et permet de décrire les systèmes dont la topologie du réseau de communications change dynamiquement. Le prix à payer pour ce gain d'expressivité est la complexité de la vérification et d'analyse de ces systèmes. De l'autre coté une implémentation distribuée du CCS ou du π-calcul est loin d'être évidente, les deux modèles utilisent la communication par rendez-vous, ce qui donne une synchronisation distribuée, qui n'est pas adéquate à la programmation répartie[9]. Le π-calcul permet de rendre compte des interactions mais aussi d'une forme de mobilité.On entend ici par mobilité la capacité d'un système à reconfigurer dynamiquement la topologie de son réseau d'interconnexions. Celui-ci est représenté par les noms de canaux de communications et par les processus qui en ont connaissance. La mobilité s'exprime alors par l'échange de ces noms entre processus et donc par la création de nouvelles connexions.Cependant, cette forme de mobilité ne permet pas de rendre compte de l'existence de machines et de la migration de programmes entre celles-ci tels que dans les réseaux réels[5].

Nous nous somme basé sur les travaux présentés dans (G. Ciobanu et.RUSU)[37].Qui fournissent des spécifications incomplètes.Nous avons complété les spécifications système(ACC) en utilisant l'algèbre de processus π-calcul basée sur la mobilité de lien.Nous avons complété les spécifications puis nous avons expérimenté la vérification de certaine propriétés. Il faut cependant citer un outils de vérification : " Mobility Workbench (MWB) [12]" . Cette outil supporte des modèles de systèmes en π-calcul.
La suite de ce mémoire suit le plan suivant :

Dans Le premier chapitre de ce mémoire nous avons présenté un état de l'art sur la mobilité. On tente de retracer le chemin dans lequel les idées ont évolué. Cependant, si on commence par un aspect large de la mobilité.

Dans le deuxième chapitre nous avons commencé par les approche formelle algèbre de processus et réseau de pétri est l'étude détaillée de différents algèbres de processus (CCS, π-calcul) de spécification de systèmes répartis et mobiles.

Dans le troisième chapitre nous décrivons le système mobile régulateur de vitesse adaptatif (ACC).Depuis nous nous somme basé sur les travaux présentés dans (G.Ciobanu et.RUSU 2008)[37].Qui fournissent des spécifications incomplètes.Nous avons complété les spécifications système(ACC) en utilisant l'algèbre de processus π-calcul basée sur la mobilité de lien.On utilise l'outil MWB[12] pour la vérification des unités de systèmes.

Nous terminons le présent manuscrit par une conclusion général qui récapitule le travail effectué ainsi que quelques perspectives.

Première partie

Etat de l'art

Chapitre 1

Systèmes mobiles et agent mobiles

Chapitre 1

Système mobile et agent mobile

1.1 Introduction

Avec la naissance des premiers systèmes distribués, le développement des réseaux informatiques sans fil, et la diffusion de dispositifs portables, on a vécu l'émergence de l'informatique mobile.Dans ce contexte, la mobilité concerne le déplacement de dispositifs matériels ou la migration d'applications logicielles[20] .Ces applications réparties adoptent une multitude de modèles d'exécution distribués, proposés dans la littérature[21]. Le but du déplacement est généralement d'accéder localement à des données ou à des ressources initialement distantes, d'effectuer le traitement en local et de ne déplacer que les données utiles. Nous distinguons deux rôles essentiels pour un agent mobile : les échanges de données et l'agent mobile calculateur. Dans une application donnée, un agent peut jouer l'un des deux rôles ou les deux en même temps.

Nous commençons, dans ce chapitre, par présenter brièvement la mobilité et les caractéristiques de mobilité basée sur l'agent mobile . Nous insisterons sur la description du modèle d'agent mobile, qui propose une solution facilitant le développement des applications mobiles sur des réseaux à grande échelle. Cette description permet d'appréhender la complexité de ce domaine qui est exprimée par la variété des aspects à considérer dans un tel système.

1.2 Systèmes mobiles

Des systèmes où les entités, qui les composent, peuvent changer de localité durant leurs existences.Elle se déplacent[20],ou encore quelles entités sont capables de se reconfigurer[6]. Ainsi, dans le cadre des systèmes communicants, on peut considérer que les entités mobiles sont des processus informatiques qui bougent dans l'espace formé par des liens qui existent entre-eux.

La mobilité des processus se traduit alors par la reconfiguration de la topologie de leurs liens [9][44].Par exemple, les connexions entre des téléphones mobiles et des antennes-relais peuvent apparaître et disparaître en fonction des déplacements des téléphones telles que la mobilité des processus, des dispositifs et des agents, concerne le déplacement de dispositifs matériels ou la migration d'applications logicielles sur des environnements différents (hétérogènes)[1] .On peut aussi considérer que les liens entre les processus se reconfigurent d'eux-mêmes dans l'espace virtuel formé par l'ensemble des processus liés, les processus restant fixes. On peut encore considérer que les processus se déplacent entre des espaces appelées localités, sièges des calculs qui permettent à ces processus d'évoluer. Un exemple

concret d'un tel système mobile serait un programme informatique transitant de nœuds en nœuds dans un système distribué[9]. Ces systèmes sont apparus depuis longtemps, et ont connu un grand développement avec l'évolution en informatique, électronique, et mécanique. Les systèmes de robots mobiles, les réseaux mobiles cellulaires, et enfin les logiciels ou agents mobiles.Cependant, les applications mobiles sont difficiles à développer [8][45].

1.3 Type de mobilité

On a trois types dans les application mobiles :
- **Mobilité physique (Mobile computing).**
- **Code mobile.**
- **Agents mobiles.**

1.3.1 Mobilité physique :

Définie comme étant un paradigme dans lequel les utilisateurs du réseau partagé se connectent via des machines déplaçables .

La diffusion des réseaux sans fils, et les réseaux de télécommunication cellulaires, expliquent La motivation de ce paradigme. Ces réseaux nécessitent de grandes capacités de calcul et de traitement de l'information[1].

1.3.2 Code mobile :

La capacité dynamique d'échange entre les fragments du code à exécuter et les locations sur lesquels s'exécutera ce code, définit le code mobile[20].
Un équilibrage de charge entre les processeurs connectés sur le réseau est assuré par la migration du code (processus, objet, ou une procédure). Celle-ci permet également d'améliorer les performances dans la communication (rassembler les objets qui se communiquent intensivement sur les mêmes nœuds)[20].

1.3.3 Agent mobile :

Un agent mobile est un objet capable de migrer de manière autonome d'un site à un autre afin d'exécuter des actions décrites par son créateur[4].

1.4 Architecture de systèmes mobiles

Les composants sont les éléments composant d'une architecture,Ils peuvent être divisés en :

1.4.1 Composants de ressource des éléments

représentant des données passives ou physiques dispositifs, par exemple un fichier, un pilote de périphérique réseau ou un pilote d'imprimante ,et il y'a un type particulier de ressource est représenté par des composants de code qui contiennent le savoir-faire nécessaire à l'exécution d'une tâche particulière[1].

1.4.2 Composants informatiques(processus,fil)

Ils sont caractérisés par un état, qui comprend données privées, l'état de leur exécution, et des liaisons avec d'autres composants, particulier pour coder les composants et les ressources Composants[6].

1.4.3 Interactions

Des événements qui impliquent deux composants ou plus. Par exemple, un message échangé entre deux composants de calcul peuvent être considérés comme une interaction entre eux[7].

1.4.4 Sites

Ils hébergent des composants et fournir un soutien pour l'exécution de calcul composants. Dans nos paradigmes, les sites incarner l'avis intuitif de localisation.
Les interactions parmi les composants résidant dans le même site sont considérés comme moins chers que les interactions se déroulant entre les composants situés dans différents sites[6].

1.4.5 Environnement d'exécution

Fournir un support pour l'exécution du calcul Composants[11].

1.5 Structures organisationnelles

Parmi les structures organisationnelles de la mobilité, nous présentons l'évaluation à distance, le code à la demande et le modèle à base d'Agent Mobile :

1.5.1 Évaluation distante

Dans une interaction par évaluation distante (voir Figure1.1), un client envoie un code à un site distant. Le site récepteur utilise ses ressources pour exécuter le programme envoyé. Éventuellement,une interaction additionnelle délivre ensuite les résultats au client[7]. Dans ce schéma, seul le code est transmis au serveur et l'exécution du code se déroule uniquement sur ce dernier. Le code d'une requête SQL émis vers un serveur de base de données représente un exemple d'évaluation distante.

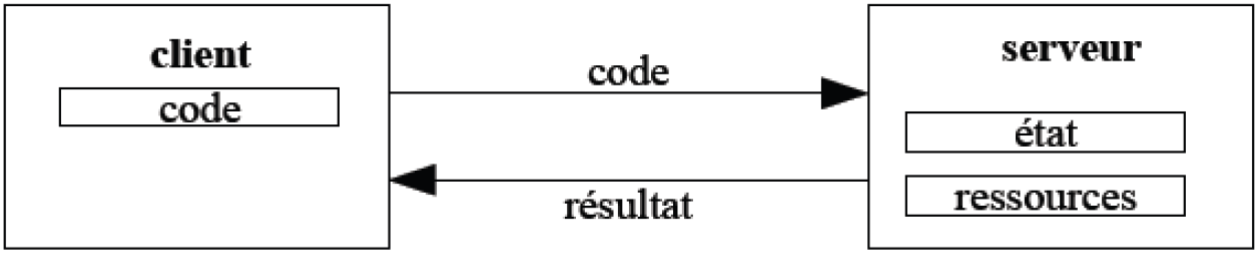


FIGURE 1.1 – Evaluation à distance[4]

1.5.2 Code à la demande

Dans ce schéma, le processus client interagit avec un site distant afin de récupérer un savoir-faire qui sera exécuté sur la machine cliente. Ainsi, le client télécharge le code nécessaire à la réalisation d'un service[11].
Le rôle du site distant est de fournir le code du service qui sera exécuté sur le site client (voir Figure 1.2). Les Applets Java reposent sur cette technologie de code mobile, il s'agit d'un programme chargé à partir d'une page Web pour être exécuté sur la machine du client.

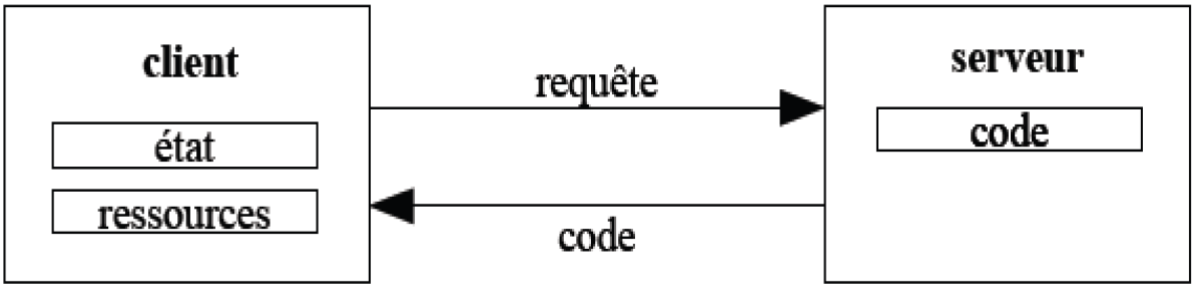


FIGURE 1.2 – Code à la demande[4]

1.5.3 Agent mobile

Les agents mobiles ont été introduits initialement en 1994 avec l'environnement Téléscript qui permettait aux processus de choisir eux-mêmes de se déplacer sur les sites d'un réseau afin de travailler localement sur les ressources.

Dans ce schéma(voir Figure 1.3), le savoir-faire appartient au client, l'exécution du code est initiée côté client et continuée sur les différentes machines visitées.
L'exécution du processus débute sur le site client. Dans la mesure où le client a besoin d'interagir avec le serveur, ce même processus (code, état d'exécution et données) se déplace à travers le réseau pour continuer son exécution et pour interagir localement avec les ressources

du serveur . Après exécution, l'agent mobile retourne éventuellement vers son client afin de lui fournir les résultats de son exécution.

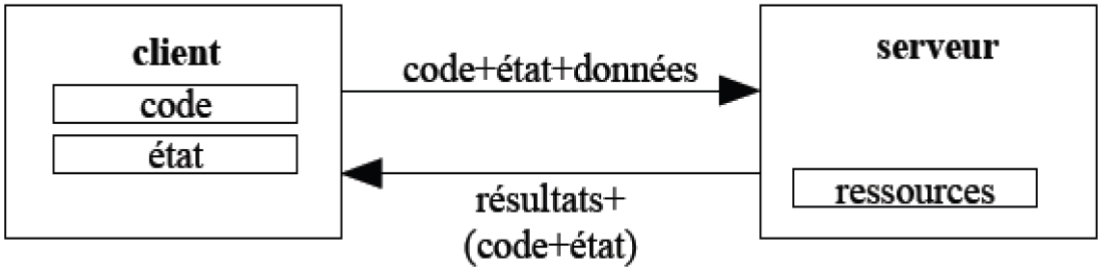


FIGURE 1.3 – paradigme informatique basé sur les agents mobiles[7]

1.5.4 pourquoi la mobilité ?

Il ya beaucoup de bonnes raisons motivant l'utilisation de mobilité :

1) **Recherche de performance** :
La migration du code vers les données offerte par l'utilisation de mobilité [38] :

a) **Réduction des coûts de communication :**
L'informatique distribuée nécessite des interactions entre différents ordinateurs via un réseau[20]. La latence et le trafic réseau[5] des interactions souvent sérieusement affecter la qualité et la coordination de deux programmes fonctionnant sur des ordinateurs différents.
Comme nous pouvons le voir sur la figure(1.4) , si l'un des programmes est un agent mobile, il peut migrer à l'ordinateur, l'autre est en train de communiquer avec lui localement. C'est, agent mobile La technologie permet aux communications à distance de fonctionner comme des communications locales[2].

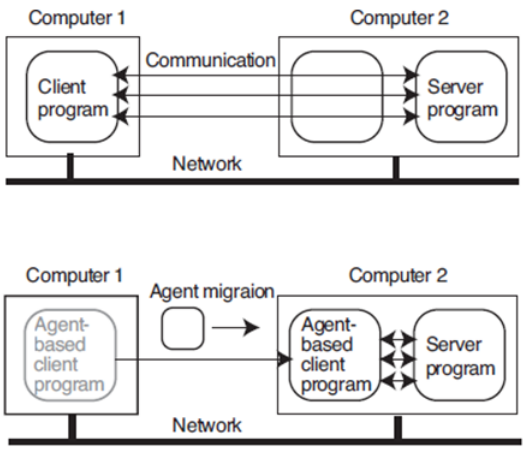


FIGURE 1.4 – réduction de communication[20].

b) **Éviter les transferts de données intermédiaires** :

à travers le réseau lors d'un calcul, permet de continuer l'exécution du calcul malgré la présence de coupures du réseau[28].
Si ces différents avantages permettent le plus souvent à un agent mobile de réaliser des calculs plus rapidement qu'avec une solution traditionnelle de type client/serveur, il arrive que l'utilisation de mobilité puisse ralentir l'exécution d'un calcul[4].
En effet, le code de l'agent peut être plus important (en nombre de bytes) que les données avec lesquelles il travaille. Dans ce cas le transfert de l'agent est plus long que le transfert des données. De la offre des temps de transfert rapides alors l'exécution d'un agent mobile peut être plus lente que même façon, si le réseau le transfert des données. Ceci est dû au fait que les systèmes mobiles sont souvent implantés à l'aide de langages interprétés pour des raisons de portabilité et de sécurité[7], l'interprétation du code peut alors être plus lente que le transfert de données[10]. L'apport des agents mobiles dans un système distribué est donc important si le réseau n'est pas rapide, ce qui est souvent le cas dans un environnement comme une grille de calcul[19].

2) **Indépendance des calculs :**
Certains services, nécessitant de longues phases de traitement, ne supportent pas toujours les ruptures de connexion avec les clients.
Par ailleurs, le maintien des liens des communications dans les réseaux à large échelle ou sans fil, est très difficile. Ceci rend, le modèle d'évaluation à distance où le client et le serveur doivent rester connectés tant que le service est en cours d'exécution, très peu adéquat[3].

3) **Efficacité** :
Les Systèmes mobiles consomment moins de ressources réseau car ils déplacent le calcul vers les données au lieu des données vers le calcul[10].

4) **Moins de bande passante** :
La plupart des protocoles de communication impliquent plusieurs interactions, ce qui cause beaucoup de trafic sur le réseau.
Les applications mobiles consomment de la bande passante uniquement lorsqu'elles se déplacent[5].

5) **Robustesse et tolérance aux pannes** :

La capacité des agents mobiles à réagir de façon dynamique aux situations défavorables facilite la construction d'un comportement de tolérance aux pannes dans les systèmes distribués complexes[10].

6) **Soutien au commerce électronique** :
Les systèmes (agents) mobiles sont utilisés pour construire des marchés électroniques, car ils incarnent les intentions, les désirs et les ressources des participants sur le marché[39].

7) **Paradigme de développement plus facile** :
La construction de systèmes distribués peut être facilitée avec des agents mobiles.
Les systèmes mobiles sont intrinsèquement distribués dans la nature. Par conséquent, ils sont une vue naturelle d'un système distribué[6].

8) **Facilité d'adaptation** :
Une application construite à base d'agents mobiles peut se re-configurer dynamique-

ment en réaction à une situation particulière pour améliorer la performance, satisfaire la tolérance aux pannes ou réduire le trafic réseau.
En effet,la capacité d'adaptation individuelle des agents mobiles permet de faire évoluer un système distribué pour répondre aux besoins d'adaptation qu'ils soient opérationnels (décentralisation des données, hétérogénéité matérielle et logicielle des composants . . .) ou fonctionnels (spécialisation des calculs à réaliser au niveau des données ou avec les autres agents). Bien entendu, l'adaptation d'un agent doit s'effectuer de manière transparente du point de vue de l'application, et donc de l'utilisateur. Les apports majeurs de ce type d'architecture concernent donc la facilité de mise en œuvre de l'adaptation dynamique aux conditions d'exécution et au calcul à effectuer[5].

1.6 Limites des systemes mobile

Le principal problème avec les systèmes mobile est :

— Sécurité :
Qui reste un domaine de recherche en soi. Dans un système d'agent avec un faible niveau de sécurité, l'agent mobile peut endommager l'hôte ou l'hôte peut endommager l'agent mobile [20].
— Nécessité d'installer une plate-forme de mobilité de support sur chaque hôte que les composants de systèmes doivent visiter. De plus, dans ces plateformes, le code de mobilité est généralement interprété [3] Manque de conscience de la mobilité par les applications [11].
— Modèle de programmation intrinsèquement transparent (objet, axé sur les composants, mais pas sur les aspects)[2].
— Manque de sensibilisation à la mobilité par le système[10].
— Réseau : les protocoles de transport existants sont inefficaces à utiliser mélange hétérogène de réseaux fixes / sans fil[4].

— Session et présentation : inappropriée pour le sans fil environnement et pour la mobilité[6].

1.7 Conclusion

Dans ce chapitre, nous avons présenté la mobilité informatique et Les systèmes mobiles qui fournissent une nouvelle approche du logiciel basé sur l'adaptabilité du système par rapport à son contexte d'exécution.La mobilité utilise pour interagir avec la ressource pour extraire des données. Si la ressource est manquante, il peut adapter son comportement et passer sur un autre ordinateur pour effectuer sa tâche.

Chapitre 2

Formalismes pour les Systèmes Mobiles

2.1 Introduction

Nous présentons les approches basées sur des langages pour la modélisation de systèmes concurrentiels tels que les algèbres de processus.Il existe une variété de formalismes qui peuvent être utilisés pour spécifier les systèmes mobiles et de vérifier leurs propriétés.

L'objectif de ce chapitre est de présenter quelques formalismes proposés pour la spécification est la vérification des aspects mobiles en informatique. Nous allons décrite tout d'abord l'algèbre de processus avec sa version de base LOTOS, CCS (où, il n y a pas de mobilité) et son extension π-calcul qui a introduit la mobilité des liens de communication entre processus. Enfin son extension le plus convenable à la spécification de processus mobiles.

2.1.1 Méthodes formelles

Les méthodes formelles (MFs) [35] sont des techniques basées sur les mathématiques pour décrire des propriétés de systèmes. Elles fournissent un cadre très rigoureux pour spécifier, développer et vérifier les systèmes d'une façon systématique,afin de démontrer leur validité par rapport à une certaine spécification. L'absence des incohérences, des contradictions et des failles dans la conception ainsi que la détermination de la correction l'implantation d'un système, sont tous des éléments très fortement assurés par ces méthodes qui sont désormais, l'objet de nombreuses recherches afin d'élargir leurs champs d'application.Parmi les plus importantes de ces méthodes les algèbres de processus.

2.1.2 Pour quoi utiliser ?

•Analyse et spécification [35] :

— Description formelle des concepts clés du problème (domaine d'application).

— Spécification formelle de la solution (comportement du système).

— Prototypage.

• **Conception architecturale et détaillée :**

— Spécification formelle de l'interface des modules.

— Méthode rigoureuse de rafinement.

•**Tests [35]** :

— Génération automatique des tests.

2.1.3 Limites et inconvénients des méthodes formelles

Pourtant, les méthodes formelles sont souvent critiquées. D'une part, elles sont jugées difficiles pour des non-experts, tels que les clients.La difficulté de formalisation s'accompagne d'une difficulté de lecture et relecture pour les intervenants extérieurs.

Ceci limite les échanges possibles entre clients et concepteurs pendant les phases initiales importantes qui décident du succès ou de l'échec d'un projet.De plus, les clients peuvent participer aux étapes de validation des spécifications.

D'autre part, ces méthodes ont un champ d'application limité et ne couvrent que certains aspects des systèmes, et de plus, n'offrent pas nécessairement une garantie totale sur ces aspects. D'un point de vue théorique, il existe une limitation d'in-décidabilité faisant que plusieurs propriétés ne peuvent être vérifiées. D'un point de vue pratique, comme nous le montrons dans les paragraphes suivants, une activité de modélisation et de vérification formelles ne peut faire l'impasse sur une activité de validation informelle des modèles proposés. Cette validation (conduite par relectures, simulations informelles ou tests) est alors sujette aux failles. L'existence d'une preuve formelle sur les modèles peut laisser une illusion de correction, alors que cette preuve n'a pas d'intérêt si les modèles eux-mêmes ne sont des modèles fidèles de ce qu'ils sont censés représenter.

2.2 Algèbres de processus

Les algèbre de processus sont des formalismes de description formelle pour la spécification de systèmes logiciels, en particulier pour les systèmes mobiles[11].

Parmi les algèbres des processus existants, on peut citer LOTOS[44], CSP (Communicating Sequential Process) définis par Hoare [46], CCS (Calculus of Communicating Systems) définis par Milner [14], ACP (Algebra of Communicating Process) définis par Bergstra et Klop [30] et le π-calcul défini par Robin Milner, Joachim Parrow et David Walker [13] [14]. Pour ce faire,π-calcul prend en charge le passage de canaux de communication (channels) en tant que données à travers d'autres canaux de communication. La structure du système concurrentiel peut alors changer en cours d'exécution.

Les algèbres de processus utilisent des structures simples telles que l'émission ou la réception de message par un processus, le séquence-ment de processus, le choix non déterministe ou encore l'exécution parallèle de processus[9]. Par exemple, π-calcul comprend des processus, des canaux de communication et des opérateurs[11].

Dans une algèbre des processus, nous pouvons raisonner en utilisant les propriétés algébriques de ce système pour vérifier ou pour démontrer si certaines propriétés sont satisfaites ou non [34].

2.2.1 Algèbre de processus à synchronisation multiple : LOTOS

LOTOS (Language of Temporal Ordering Specification) est une technique de description formelle, promue au rang de norme ISO en 1988 [44]. Il s'appuie sur le langage CCS de Milner (étendu par un mécanisme de synchronisation multiple hérité de CSP [46] de Hoare) pour la spécification de la partie comportementale ; la partie description des structures de données est inspirée de Act-One , un formalisme de description des types de données abstraits algébriques.

Le concept sous-jacent à LOTOS est que tout système peut être spécifié en exprimant les relations qui existent entre les interactions constituant le comportement observable des composantes du système. En LOTOS, un système est vu comme un processus, qui peut être constitué de sous-processus, un sous-processus étant un processus en lui-même. Une spécification LOTOS décrit ainsi un système par hiérarchie de processus. Un processus représente une entité capable de réaliser des actions internes (non-observable) et d'interagir avec d'autres processus qui forment son environnement.

2.2.2 Algèbre CCS pour les processus communiquants

Ce langage est développé par Milner fin des année 70 et début des années 80.Expliqué dans son livre "Communication and Concurrency"[8]. En effet, c'est lui qui a initié la recherche sur les systèmes concurrents communicants.[20] Il a voulu mettre en évidence une description abstraite des mécanismes fondamentaux de la concurrence ainsi qu'un calcul pour raisonner sur les programmes concurrents. A sa suite, c'est développée toute une approche algébrique, connue sous le nom d'algèbre de processus[15].

Présentation

Definition 1 : L'algèbre CCS (Calculus of Communicating Systems) [14] va permettre de décrire des processus communicante,comme des ensembles d'automates non-déterministes (appelés agents ou processus) qui interagissent par le biais de synchronisations[20].

Definition 2 : Le comportement d'un agent n'est décrit que partiellement par l'ensemble de ses traces(une trace est une suite de transitions) et la notion de bisimulation propre à cette théorie permet de raffiner la notion d'égalité des agents.Le CCS est proposé en particulier pour la spécification des systèmes réactifs[15].

Definition 3 : Un système réactif est un système dont le comportement le plus important est la communication avec d'autres systèmes où des réactions aux stimulus de leurs environnements. Un système d'exploitation, système de contrôle, système de téléphones mobiles, ... sont des exemples de systèmes réactifs[15]. Est caractérisé par Un processus CCS est construit à partir d'actions atomiques et d'opérateurs de composition.Le mécanisme d'interaction inter-processus est celui des rendez-vous (synchronisation binaire a/ā).

Definition 4 : Un processus CCS est vu abstraitement comme une boite noire[20] avec une interface (des ports de communication) pour interagir avec son environnement.
la communication[9] est réduite à une synchronisation sur des ports (canaux). Les processus ne se communiquent pas de l'information via ces ports. Deux processus qui ont les mêmes noms de ports se synchronisent et les communications sont bloquantes.

Definition 5 : Un processus CCS [8]peut exécuter un ensemble d'actions Act=A$\cup\bar{A}\cup\tau$.

1) A = a, b, c, d, ... est un ensemble de canaux d'entrée (appelés : actions d'entrée ou de réception).

Definition 6 : Un processus P peut être soit le processus inerte 0, soit un processus préfixé par une action a, soit une composition parallèle de processus P |Q, ou une restriction(νx) c'est à dire seul le processus P à accès sur le nom x.

Definition 7 : La signification de la syntaxe des processus est donnée par une relation de transition ternaire $P \xrightarrow{a} Q$, qui signifie le processus "P peut effectuer l'action a et devenir le processus Q".

2) $\bar{A}$= ā, ... est un ensemble de canaux de sortie (appelés : actions de sortie ou d'émissions).

3) τ : représente une action silencieuse (inobservable par les autres processus).

La syntaxe de CCS [12]

Un terme CCS peut être processus ou une action.Il existe cinq opérateurs en CCS[20] :
. (Opérateur de suffixe)

+ (opérateur de choix)

| (Opérateur de parallélisme)

/ (opérateur de restriction)

$[f]$ (opérateur de re-nommage, où f est une fonction).

La sémantique du CSS

Le CSS est un langage formelle est donc il dispose d'une sémantique opérationnelle formelle. Cette formelle est basée sur des interactions atomiques système/environnement[20].

La sémantique opérationnelle d'une algèbre de processus associe un modèle à chaque terme de cette algèbre, ce modèle se présente sous la forme d'un système de transitions étiquetées (LTS) ou automate à états finis[8].
La sémantique opérationnelle [21] exploite une relation de transition notée " $\longrightarrow$ " qui met en relation les éléments du langage. Cette relation de transition est utilisée dans les règles d'inférence qui décrivent l'évolution du système modélisé.

Les règles de transition élémentaires sont [21] :

$$\frac{}{act.P \xrightarrow{act} P}$$

$$\frac{P \xrightarrow{act} P'}{P+Q \xrightarrow{act} P'} \quad \text{et} \quad \frac{Q \xrightarrow{act} Q'}{P+Q \xrightarrow{act} Q'}$$

$$\frac{P \xrightarrow{act} P'}{P|Q \xrightarrow{act} P'|Q} \qquad \frac{Q \xrightarrow{act} Q'}{P|Q \xrightarrow{act} P|Q'} \qquad \frac{P \xrightarrow{act_1} P' Q \xrightarrow{act_2} Q'}{P|Q \xrightarrow{act_1 \wedge act_2} P'|Q'}$$

Exemple

Pour représenter graphiquement des processus décrits en CCS, on utilise une classe de systèmes de transitions étiquetées qui sont LTS (labeled transition système)[20].

Les étiquettes des transitions évoqueront les émissions et les réceptions.Voici le LTS d'une machine délivrant du thé et du café (figure2.1) (elle a été volontairement simplifie dans le sens o'u la transmission du choix de la boisson par l'utilisateur a été occulté) :
Cette conception déterministe de la machine (il n'y a pas deux transitions étiquetés)[20].

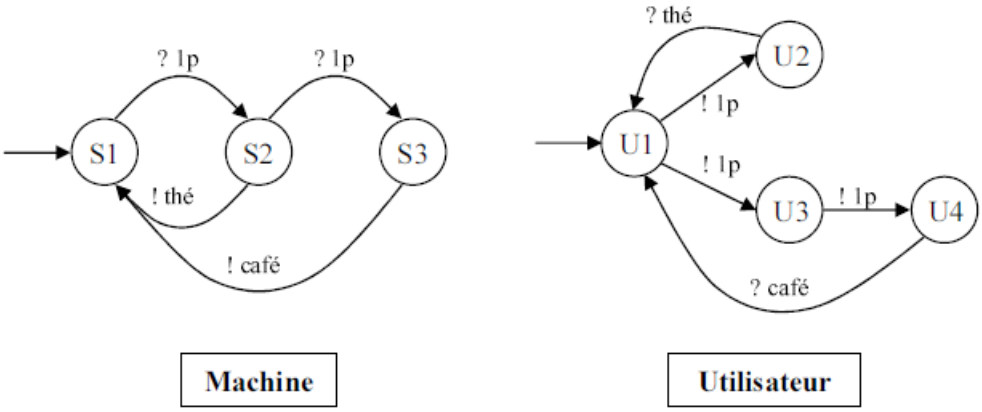


FIGURE 2.1 – Machine à café déterministe[20]

identiques partant d'un même état) possède l'équation CCS suivante[20] :

S1=1P.($\overline{thé}$.S1 + 1p.$\overline{café}$.S1)

Cette modélisation peut correspondre au mode de fonctionnement suivant :
l'utilisateur introduit d'abord les pièces puis effectue le choix de sa boisson.
L'équation le montre bien : réception de la première pièce, puis choix entre fournir du thé ou recevoir d'une deuxième pièce et fournir du café.

2.3 *π–calcul*

R. Milner[13].A poursuivi ses travaux sur les processus communicants et rajoute la notion de mobilité [13] [14] [15]. La problématique étudiée est donc la communication de plusieurs processus au travers d'un réseau dont la topologie change dynamiquement.

Cette algèbre[14][15][7] est une extension de l'algèbre de processus CCS, et suit la même idée que les travaux de Engbergg et Nielsen qui ont ajouté de la mobilité à CCS tout en préservant ses propriétés algébriques.

En particulier,le *π–calcul* est un calcul pour modéliser des processus communicants qui sont capable de changer leurs interconnexions d'interaction est principalement utilisé pour établir des preuves d'équivalence entre des modèles de systèmes distribués[20]. Le calcul s'étend en permettant un processus de transfert noms utilisés pour les liens de communication.

Le processus de réception peut utiliser le lien vers interagir avec d'autres parties du système.[9].Par conséquent, les noms des ports de communications peuvent changer durant le déroulement du système.

Les processus (dits agents) peuvent échanger leurs ports de communication, comme ils échangent des valeurs entre eux[20].le π–*calcul* permet de rendre compte de la notion de localité et de migration.

Ce formalisme spécifie des réseaux de processus interagissants. Les communications se font sur des canaux, eux-même définis par des noms. Le contenu des communications peut être le nom d'un canal se qui provoque un échange des noms de canaux [44].

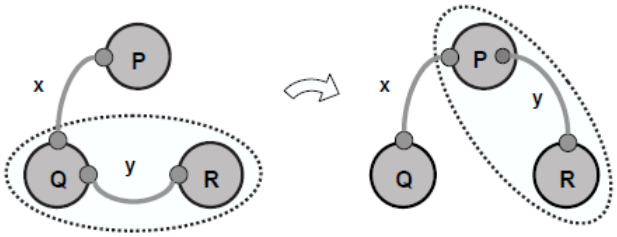


FIGURE 2.2 – exemple du changement d'interconnexion entre les processus de communication[44]

Nous appelons un processus un agent, et appelons un nom utilisé pour les liens de communication Port.La Figure(2.2) illustre comment les agents communiquent et modifient leurs interconnexions.

Dans cet exemple,l'agent P communique avec l'agent Q via le port x.Dans cette interaction,Q transfère le port y vers P. Cela rend les interconnexions changé,et P peut communiquer avec l'agent R.

•Exemple de processus dans le langage pi-calcul :

P : := 0 | αi.P | P1 + P2 | P1|P2 | (νx)P |$[x = y]P|A(x1, ..., xn)$.

2.3.1 Syntaxe π-Calcul

La syntaxe du langage pi-calcul est donnée selon la grammaire dans la table [2.1]suivante :

Préfixes	α	::=	$\overline{a}\langle x \rangle$	Émission
			$a(x)$	Réception
			τ	Interne
Agents	P	::=	0	Nul
			$\alpha . P$	Préfixation
			$P + P$	Somme
			$P \mid P$	Parallèle
			$[x = y]P$	Égalité
			$[x \neq y]P$	Inégalité
			$(\nu x)\ P$	Restriction
			$!P$	Réplication
			$A(y_1, \ldots, y_n)$	Identifiant
Définitions			$A(x_1, \ldots, x_n) \stackrel{def}{=} P$	où $i \neq j \Rightarrow x_i \neq x_j$

TABLE 2.1 – Syntaxe du π-calcul [21]

Cette syntaxe formelle peut être interprétée par[8] :

1. **Le processus vide 0** :
c'est un processus qui ne fait rien.

2. **Un préfixe de sortie ($\bar{a}$x.P)** :
c'est un processus qui envoie le nom x sur le canal a et se comporte ensuite comme P. a est un sujet positif et x est l'objet de l'action , $\bar{a}$x est appelé préfixe négatif.'

3.**Un préfixe d'entrée (a(x).P) :**
c'est un processus qui reçoit un nom sur le canal a, ensuite se comporte comme P dans lequel x est remplacé par le nom reçu .
a est un sujet négatif et x est l'objet de l'action, a(x) est appelé préfixe positif.

4.**Un préfixe silencieux (τ.P) :**
c'est un préfixe qui fait une transition silencieuse τ (tau) et se comporte ensuite comme P.

5.**Une somme (P+Q) :**
c'est un processus qui se comporte comme P ou comme Q.

6. **Une composition parallèle (P|Q) :**
c'est un processus représentant la composition parallèle de P et Q.
Les processus P et Q peuvent évoluer séparément, de plus les communications entre P et

Q peuvent se produire si l'un des processus émet sur un canal et l'autre reçoit sur le même canal .Ces communications se traduisent par des transitions silencieuses τ.

7. **Le match (if x=y then P) :**
c'est un processus qui se comporte comme P si x=y , sinon ne fait rien .

8.**Le mismatch (if x $\neq$ y then P) :**
c'est un processus qui se comporte comme P si x et y n'ont pas la même valeur, sinon ne fait rien .

9. **Une restriction ((vx)P) :**
c'est un processus qui agit comme P mais le nom x est privé à P et ne peut pas être utilisé comme canal dans les communications avec l'environnement du processus (ie, avec les autres processus) .x est donc lié dans P (ie, x est un nom local dont la portée est P).

10.**Un identificateur de processus (A(y , y ,...yn) 1 2) :**
tel que n est l'arité du processus

defini par une équation de la forme P= A (x1,x2,...xn) , où x1,x2,...xn sont des noms distincts et les seuls noms libres dans P.

Le processus A(y1, y2,...yn) se comporte comme P dans lequel chaque xi a été substitué par yi .
les xi peuvent être considérés comme des paramètres formels de A et les yi comme des paramètres d'appel dans A.

11.**Un identificateur de processus (Une Réplication (!P) = P| !P :**

L'opérateur « ! » est appelé réplicateur, définie par !P = P | !P .
Le processus P peut être appliqué une infinité de fois [17].

2.3.2 Noms dans le π-calcul

Forme normale[20]

On dit qu'un processus P est sous une forme normale de tête s'il est égal à une somme de préfixes.

P=$\sum$ αi.Pi (ou αi est un préfixe) [21].

On note par n(P) l'ensemble des noms apparaissant dans le processus P .

Les noms liés dans un processus P notés bn(P) sont les noms liés par un préfixe d'entrée ou par l'opérateur de restriction[20].

Les noms liés traduisent uniquement les calculs internes.
On écrit P Q pour exprimer que P et Q sont alpha-équivalents (i.e, ils diffèrent seulement par les noms liés)[15].

Les noms libres dans un processus P notés fn(P) sont des noms qui ne sont pas liés dans P.

Les noms libres d'un processus P représentent l'environnement de ce processus ou sa liaison avec les autres processus de son environnement.
Réciproquement, les noms libres d'un processus P représentent ce que l'environnement sait sur ce processus. Lorsque x, l'objet d'un préfixe de sortie, est lié on dit qu'on a un préfixe de sortie lié et on le note « $\bar{a}$(x) »[16].

Les noms libres d'un préfixe α fn(α) et les noms liés bn(α) sont présentés dans la table [2.2] suivante :

α	type	fn(α)	bn(α)
τ	silencieuse	$\{\ \}$	$\{\ \}$
$\bar{x}y$	Préfixe de sortie libre	$\{x, y\}$	$\{\ \}$
$\bar{x}(y)$	Préfixe de sortie lié	$\{x\}$	$\{y\}$
xy	Préfixe d'entrée libre	$\{x, y\}$	$\{\ \}$
x(y)	Préfixe d'entrée lié	$\{x\}$	$\{y\}$

TABLE 2.2 – noms liés et noms libres [21]

On dit qu'une action est libre si tous les noms apparaissant dans cette action sont libres. On note par P{y \ x } le résultat de la substitution de toutes les occurrences libres de x par y dans P, avec un renommage des noms liés ceci est nécessaire pour éviter la capture de y (ie, la liaison de y) dans P[13].

Le processus Pσ est P dans lequel tous les noms libres x sont remplacés par σ(x), où σ est une fonction de N vers N.

Considérons le processus ($\bar{x}$y.P | (νx)x(z).Q).

Le nom x apparaît dans les deux parties de la composition parallèle, mais ne représente aucune liaison entre $\bar{x}$y.P et (νx)x(z).Q
car x est libre dans $\bar{x}$y.P mais lié dans (vx)x(z).Q .[17]

En particulier, ces processus ne peuvent pas communiquer le long du canal x.

Même si le nom x est transmis comme objet à un processus possédant lui-même un nom local x (phénomène appelé scope intrusion), les deux x demeurent distincts comme le montre la transition suivante :

$\bar{x}$y.P | (νx)x(z).Q $\xrightarrow{\tau}$P | (νx')Q$\{x'/x\}${x/ z}

Le renommage du x local par x' est une conséquence de la définition de la substitution, les noms liés sont renommés si nécessaire afin d'éviter les captures. La seule situation où l'environnement de (vx)P peut connaître le nom local x, est lorsque P émet x comme objet

dans une communication à l'environnement. La portée de x s'accroit alors en s'étendant de x (mais pas aux autres processus). Ce phénomène est appelé scope extrusion et est illustré par l'exemple suivant[20].

Deux processus P et Q qui ne diffèrent que par leurs noms liés, sont appelés des processus α-convertibles, et ils ne sont pas distingués. Dans ce cas, on a l'habitude de noter P α Q.[30]

Les noms libres d'un processus circonscrivent les capacités de ce processus à communiquer,et donc interagir avec d'autres processus de son environnement.

Pour pouvoir émettre ou recevoir un nom, un processus doit posséder des noms libres qui permettent les échanges.

Dans la figure(2.3), les processus P, Q, et R peuvent communiquer parce qu'ils partagent les noms libres x, y, z et w.

Le nom x est partagé par les trois, alors que y est exclusivement réservé aux processus P et Q : on dit qu'il est restreint, ou interne, ou, plus habituellement,lié au processus (P|Q). Aucun autre processus du système ne peut l'utiliser[16][39].

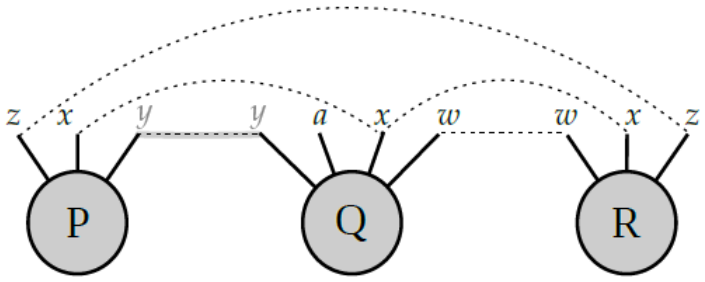
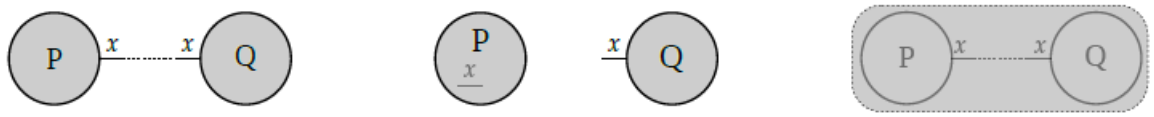


FIGURE 2.3 – Communications entre processus en π-calcul. y est lié au processus P|Q[8].

Un processus qui n'a pas de noms libres, ne peut pas réaliser d'actions d'entrées-sorties avec d'autres processus de son environnement : il est dit clos ou fermé. Par extension, un système clos est un système qui ne partage aucun nom libre avec son environnement, ce qui revient à dire qu'il ne peut pas réaliser d'actions d'entrées/sorties avec lui Par contre, les interactions entre les composants de ce processus ou de ce système qui, depuis son environnement sont modélisées par des actions silencieuses, restent possibles.

La figure (2.4) illustre les possibilités de communication entre deux processus P et Q [39].



(a) Le système $P \mid Q$. Les processus P et Q partagent le nom *libre* x : ils peuvent communiquer.

(b) Le système $(\nu x)P \mid Q$. Le nom x est *lié* (ou restreint) à P, et utilisé par certains de ses composants non représentés. Les deux processus ne peuvent pas communiquer.

(c) Le système $(\nu x)(P \mid Q)$. Le nom x est lié au processus $P \mid Q$. Les processus P et Q peuvent communiquer par cette voie qui leurs est exclusivement réservée.

FIGURE 2.4 – Illustration de la capacité d'échange d'un processus en π-calcul[39].

Finalement, l'analyse de l'attache des occurrences des noms d'un processus permet de connaître, dans une certaine mesure seulement, les comportements de celui-ci.

Dans x(z).P,les occurrences libres de z dans P indiquent les places où le nom libre reçu via x sera substitué après une interaction. C'est grâce à de tels échanges de noms que la topologie de communication d'un processus peut évoluer.

Si fn(P) = ϕ, alors P est clos, et il ne peut pas communiquer avec son environnement. Donc, si son comportement est dynamique, alors ce dynamisme provient nécessairement de l'interaction entre ses composants. D'où l'intérêt de la vérification formelle des systèmes clos : on isole le système de son environnement, avant d'étudier son évolution comportementale.

Substitutions de noms

Une substitution,[20] sous-entendue de noms, est une fonction σ définie de N dans N par l'identité, l'image d'un nom étant lui même,sauf pour un sous-ensemble fini de noms.

La substitution[20] σ appliquée au nom x est notée xσ. L'image du sous-ensemble de noms {x1,..,xn} par σ, notée {y1/x1, . . . , yn/xn},est telle que quel que soit i $\in$ [1..n] xiσ = yi et xσ = x pour x$\notin$ {x1, . . . , xn}[21] .

On note P{y1/x1,...,yn/xn}, ou P{yi/xi} 1$\preceq$ i $\preceq$ n,le processus obtenu en substituant simultanément chaque occurrence libre de xi par yi dans P avec, si nécessaire,le changement des noms liés portant la même appellation.

On retrouve ce changement nécessaire de nom dans l'intrusion dans la portée d'un nom privé présentée au paragraphe suivant :
Une substitution[8] σ peut également être appliquée à une action α, ce que l'on dénote par α σ. On a alors : $\tau\sigma = \tau$, (x(y))σ = σ(x)(y) si y $\notin$ n(σ),($\bar{x}$y)σ = $\sigma(\bar{x})\sigma$(y),
et ($\bar{x}$(y))σ = $\sigma(\bar{x})$(y) si y $\notin$ n(σ).
Dans les cas de la réception et de l'émission d'une référence,la substitution n'est définie que lorsque la substitution ne change pas les noms liés[44].

2.3.3 Exemples d'interactions entre processus

Les exemples de ce paragraphe, tirés de [17], illustrent quelques interactions élémentaires entre processus du π-calcul[21].

— **Le passage d'un lien** :

Considérons le système composé des processus P, Q et R. P partage le nom x vers Q, qu'il souhaite envoyer à R via le lien y. R souhaite le recevoir [9]. La transition,illustrée par la figure (2.5), est donc :

$$\bar{y}x.P' \mid y(z).R' \mid Q \xrightarrow{\tau} P' \mid R'\{x/z\} \mid Q$$

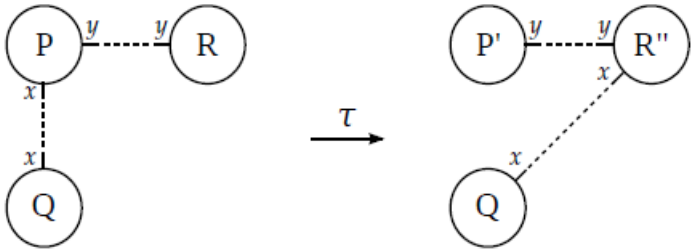


FIGURE 2.5 – Illustration du passage d'un lien entre deux processus en π-calcul.[9]

Ce schéma illustre le cas où x $\notin$ fn(R) et x $\notin$ fn(P). Ces deux conditions n'affectent pas la transition.

— **L'intrusion dans une portée d'un nom privé[8]** :

Considérons le système composé par les processus P, Q, R et S.P partage le lien x avec Q,et le lien y avec R [11].
Par ailleurs, R un accès exclusif au processus S, via le lien privé x qu'ils partagent[11].
Le système global comprend donc deux liens x :
l'un est public, alors que l'autre est privé au sous-système R | S [21].
On suppose que P veut envoyer son lien x à R, qui souhaite le recevoir.
Il est alors nécessaire de renommer le lien privé x de ce dernier, pour éviter une confusion. Ce cas de figure se résume en disant que P présente une intrusion dans la portée, ou s'immisce dans la portée, du lien privé x entre R et S.
Les interactions entre les processus de ce système sont représentées par la figure (), et se modélisent par :suppose que P veut envoyer son lien x à R, qui souhaite le recevoir. Il est alors nécessaire de renommer le lien privé x de ce dernier, pour éviter une confusion. Ce cas de figure se résume en disant que P présente une intrusion dans la portée, ou s'immisce dans la portée, du lien privé x entre R et S. Les interactions entre les processus de ce système sont représentées par la figure (2.6), et se modélisent par :

$$\bar{y}x.P' \mid Q \mid (\nu x)(y(z).R' \mid S) \xrightarrow{\tau} P' \mid Q \mid (\nu x')(R'\{x'/x\}\{x/z\} \mid S\{x'/x\})$$

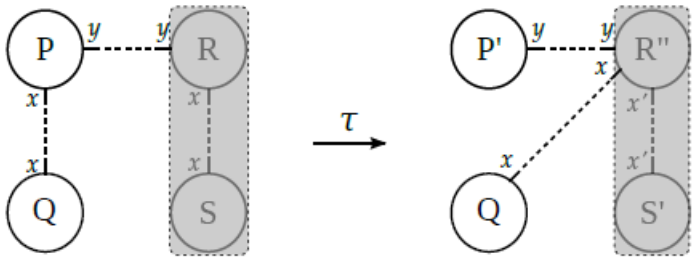


FIGURE 2.6 – Illustration l'intrusion dans d'un nom privé en π-calcul[17].

— **L'extrusion et la migration de portée[8]** :
Considérons maintenant le système composé par les processus P, Q et R. P a un accès exclusif à Q via x, un lien privé au processus P | Q.
Par ailleurs, P et R partagent le lien public y. On suppose que P veut envoyer x à R qui veut le recevoir.
Les interactions entre les processus de ce système sont représentées par la figure(2.7), et se modélisent par :

$$(\nu x)(\bar{y}x.P' \mid Q) \mid y(z).R' \xrightarrow{\tau} (\nu x)(P' \mid Q \mid R'\{x/z\})$$

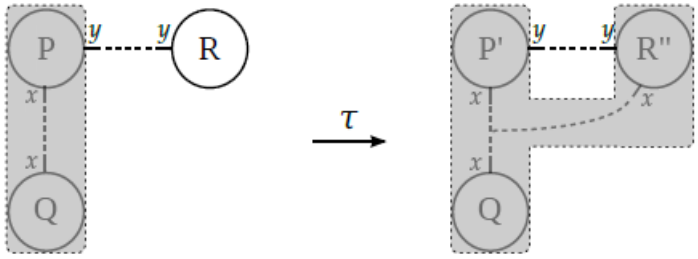


FIGURE 2.7 – IIllustration de la notion d'extrusion de portée d'un nom privé en π-calcul[21].

La portée du caractère privée de x à P | Q est étendue au processus R. Deux cas peuvent se présenter[20][9] :

– R possède déjà un lien public x, c'est-à-dire x $\in$ fn(R). Dans ce cas, le lien privé x doit être renommé avant l'interaction pour le distinguer du lien public existant.

– x n'est pas libre dans P' (x $\notin$ fn(P')). Dans ce cas, le système global évolue en :

P' | (νx)(Q | R' {x / z}),par application de la règle Strict.Cela revient à considérer que p' ne peut plus communiquer via x puisqu'il ne le possède plus. Ce cas de figure(2.7) correspond à la migration de portée de x, qui est un cas particulier d'extrusion [29].

2.3.4 Sémantique

Sémantique opérationnelle

Une sémantique opérationnelle[8] est très utile pour calculer l'évolution d'un terme. La sémantique opérationnelle de π-calcul est donnée par un système de transitions étiquetées, où les transitions [10] sont du type p$\xrightarrow{\alpha}$p' avec α un ensemble d'actions pouvant être l'action interne τ, l'action de réception a(x) et l'action d'émission $\bar{a}$(x).
Cette relation de réduction signifie que P est transformé en P' suite à l'action α.

La transition a(x).P$\longrightarrow$a(u) Pu/x signifie que si le nom u est transmis à travers le canal a alors le processus a(x).P qui attend un nom sur le canal a reçoit le nom u et effectue une substitution de x par u et se comporte ensuite comme P, où toutes les occurrences de x sont remplacées par[14] .

Les règles de communication sont les plus spécifiques pour un système ayant un besoin de mobilité.
L'interaction entre deux processus est donnée par les règles COM1 et COM2 de communication :

$$Com_1 : \quad \frac{P \xrightarrow{\bar{a}x} P', Q \xrightarrow{a(y)} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'\{x/y\}}$$

$$Com_2 : \quad \frac{P \xrightarrow{a(y)} P', Q \xrightarrow{\bar{a}x} Q'}{P \mid Q \xrightarrow{\tau} P'\{x/y\} \mid Q'}$$

La sémantique opérationnelle duπ-calcul [7] est donnée par la table des transitions suivante(2.3) :

STRUCT	$\dfrac{P' \equiv P, P \xrightarrow{\alpha} Q, Q \equiv Q'}{P' \xrightarrow{\alpha} Q'}$
PREFIX	$\dfrac{}{\alpha.P \xrightarrow{\alpha} P}$
SUM	$\dfrac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}$
MATCH	$\dfrac{P \xrightarrow{\alpha} P'}{if x = x then P \xrightarrow{\alpha} P'}$
MISMATCH	$\dfrac{P \xrightarrow{\alpha} P', x \neq y}{if x \neq y then P \xrightarrow{\alpha} P'}$
PAR	$\dfrac{P \xrightarrow{\alpha} P', bn(\alpha) \cap fn(Q) = \{\}}{P \mid Q \xrightarrow{\alpha} P' \mid Q}$
COM	$\dfrac{P \xrightarrow{a(x)} P', Q \xrightarrow{\bar{a}u} Q'}{P \mid Q \xrightarrow{\tau} P'\{u / x\} \mid Q'}$
RES	$\dfrac{P \xrightarrow{\alpha} P', x \notin \alpha}{(\nu x)P \xrightarrow{\alpha} (\nu x)P'}$

TABLE 2.3 – La sémantique opérationnelle du π-calcul[21].

Les différentes valeurs possibles de α sont :

•l'action interne τ .

•l'action de sortie (libre) de type $\bar{a}$x.

•l'action d'entrée de type a(x).

•l'action de sortie liée $\bar{a}$(x).

2.3.5 Extensions du π-calcul

a) **•π-calcul asynchrone** :
dans le cas des systèmes distribués le synchronisme n'est pas réaliste[43].
C'est pourquoi, on se fixe la contrainte suivante[42][20] :
lors des envois de messages, on envoie le message et aucune action situées après ne doit être tributaire de cet envoi [9]. On le note $\bar{c}$(v).0.L'envoi de message devient une action terminale.

b) **•π-calcul polyadique** :
il autorise dans une seule et même action l'envoi, la communication de plusieurs noms :
$\bar{c}$(v1, ..., vn).P
et c(w1, ...,wn).Q.
Toutefois, ceci peut être simulé en calcul monoadique en passant le nom d'un canal privé par lequel sera envoyée la séquence de noms[15].

c) **•π- calcul d'ordre supérieur[20]** :
Dans le τ -calcul d'ordre supérieur les objets transmis dans une communication peuvent être des processus.
La forme d'un préfixe de sortie d'ordre supérieur est $\bar{a}$ <P>.Q
ceci signifie « envoyer l'agent P à travers a ensuite continuer comme Q ».
Le préfixe d'entrée d'ordre supérieure est de genre a(X).Q,
ce qui signifie « recevoir un agent X ensuite continuer comme Q ».
Dans [14] un système de sorte d'ordre supérieure a été défini pour le π-calcul d'ordre supérieur afin de résoudre les problèmes de sorte.

d) **•π-calcul stochastique [20]** :
le π-calcul classique permet de d'écrire des systèmes sous des aspects qualitatifs tels la vérification d'équivalence, la détermination de blocages... Mais les aspects de nature quantitative et l'analyse des performances ne sont pas abordés. Alors que dans le π-calcul toutes les communications ont des chances égales de se produisent (sémantique non-déterministe), sa version stochastique introduit le taux d'activité qui représente la probabilité de réalisation d'une action.
Ainsi, à chaque action est accolé le taux d'activité :

$$(act, taux).P \overset{(act,taux)}{\rightarrow} P$$

2.4 Bisimulation

2.4.1 Bisimilarité forte et équivalence

Definition[40] : (simulation, bisimulation, bisimilarité) : Une relation binaire S sur les agents est une simulation(forte) s'il satisfait le suivant :

1. *Si* $P \xrightarrow{a} P'$ *et a action libre, alors* $Q \xrightarrow{a} Q'$ *, et* $P' \mathcal{S} Q'$.
2. *Si* $P \xrightarrow{x(y)} P'$ *et* $y \notin n(P,Q)$*, alors ,* $Q \xrightarrow{x(y)} Q'$ *et pour tout w,* $P'\{w/y\}\mathcal{S}Q'\{w/y\}$.
3. *Si* $P \xrightarrow{\overline{x}(y)} P'$ *et* $y \notin n(P,Q)$*, alors ,* $Q \xrightarrow{\overline{x}(y)} Q'$ *et* $P'\mathcal{S}Q'$.

La relation S est une bisimulation(forte) si S et son inverse sont des simulations. La relation $\sim$, bisimilarité(forte), sur les agents est définie par P $\sim$ Q si et seulement si existe une bisimulation S tel que P S Q.

2.4.2 Propriétés modèles et équivalences

Utiliser des équivalences de bisimulation[17] pour vérifier un système est utile et intéressant, mais son applicabilité est limitée. Cette approche ne supporte pas les processus lâches spécification9 . Nous avons besoin d'une logique capable de décrire les propriétés des processus pour ce genre de spécification.

Dam en [40] a proposé une logique temporelle pour le π-calcul polyadique basé sur extensions de point de HML Il fournit des fonctionnalités de paramétrage, génération, et passage des noms. Cela inclut l'utilisation de la somme dépendante et du produit tenir compte des entrées et des sorties (non-localisées) et du paramétrage explicite noms utilisant lambda-abstraction et application[40]. La syntaxe des formules est donnée comme suit :

$$\begin{aligned}\phi \quad ::= \quad & x = y \mid x \neq y \mid \phi \wedge \phi \mid \phi \vee \phi \mid \langle\alpha\rangle\phi \mid [\alpha]\phi \mid \\ & X \mid \nu x.\phi \mid \mu x.\phi \mid \lambda x.\phi \mid \phi x \mid \Sigma\phi \mid \forall\phi \mid \exists\phi\end{aligned}$$

Les formules, réparties par ϕ,ψ, sont interprétées comme des ensembles d'agents paramétrés sur les noms. Les lettres X, Y, Z vont sur des variables propositionnelles où chaque assigné une arité n $\in \omega$, écrite X : n. Les connecteurs logiques dans la syntaxe ci-dessus peut être expliqué comme suit : $\wedge$ et $\vee$ sont les connecteurs booléens pour et et ou, respectivement, $< \alpha >$ et $[\alpha]$ sont des connecteurs modaux étiquetés, ν est le plus grand opérateur de point fixe utilisé pour les propriétés invariantes,μ est le moins opérateur de point fixe utilisé pour les éventualités,λet l'application est utilisée pour le nom paramétrage Σ est la somme dépendante utilisée pour les concrétions ; $\forall$ et $\exists$ sont des quantificateurs exprimer les propriétés des abstractions [40].

2.5 Model-checking

Le model-checking[40] est une technique automatique de vérification formelle d'un système fini. Il est tr'es populaire dans le monde de la recherche et est notamment utilisé pour les systèmes réactifs[9], les systèmes critiques, les systèmes temps réel et les systèmes embarqués, complexes ou contraints. De bonnes implémentations (académiques et industrielles) de cette technique sont disponibles et plusieurs outils de model-checking commencent à être utilisés dans l'industrie (les entreprises Dassault et RATP par exemple) pour des applications critiques en sûreté[44].
Le principe de cette technique [40] est le suivant :
étant donné un ensemble de machines à états finis (ou automates, ou systèmes de transitions, ou IOLTS) et une propriété désirée, formalisée dans une logique temporelle, l'algorithme explore l'ensemble des états de ces machines afin de vérifier que la propriété désirée est bien satisfaite (figure 2.8).
Si ce n'est pas le cas, la séquence de transitions d'état menant à la violation du système est générée en guise de contre-exemple, ce qui montre que le système est incorrect[42]. L'algorithme permettant de vérifier si une machine M satisfait la propriété s'appelle un model-checker. Ces propriétés sont de plusieurs types : les logiques temporelles peuvent exprimer les notions d'accessibilité d'un état, de sûreté (le comportement spécifié ne se produira jamais), de vivacité (le comportement spécifié aura lieu), d'équité (le comportement spécifié pourra se produire une infinité de fois) et des propriétés temporelles[39].

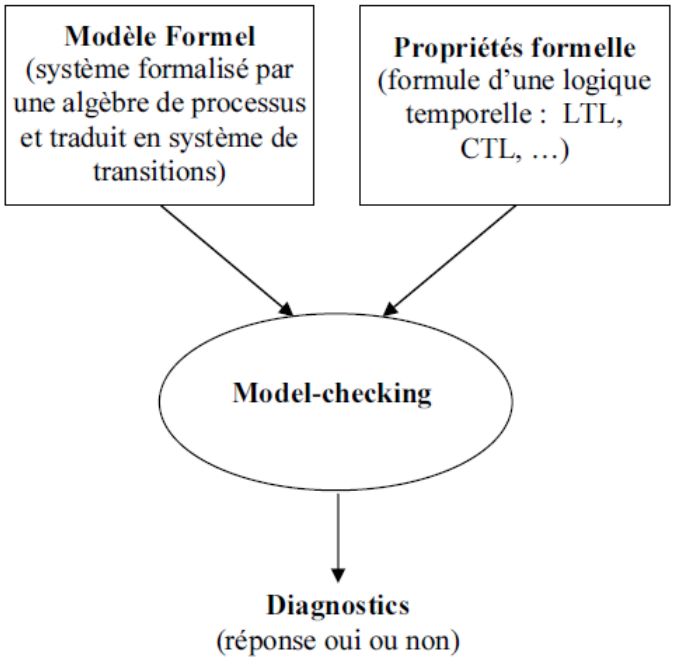


FIGURE 2.8 – Model-checking [40]

2.5.1 Limites

dans [9] [11] citent ces deux major limites du model-checking :
La première est de nature calculatoire et elle concerne la composition de plusieurs automates. La seconde est une limite de conception de la preuve.

2.6 Conclusion

Dans ce chapitre nous avons exploré plusieurs formalismes qui adapte les besoins de la programmation répartie et mobile, nous avons vu que le formalisme CCS c'était le point de départ, il n'inclu pas la notion de mobilité. π-calcul : un calcul de processus algébrique avec la mobilité. Il y a eu un nombre des formalismes qui traitent la mobilité, mais n'ont pas développé leurs théories algébriques spécifiques[8].

Chapitre 3

Spécification et vérification système ACC

Introduction

La spécification est l'une des phases importantes dans le processus de développement.Permet la description d'un système par moyens d'un langage formel. Dans ce chapitre,nous intéressons, en particulier, à la spécification formelle offrant les moyens de vérification et de validation des systèmes.Cependant La réalisation d'une spécification formelle exige la mise en place d'un formalisme dédié à la mobilité. De nos jours, il existe une variété de formalismes qui peuvent être utilisés pour spécifier les systèmes mobiles et de vérifier leurs propriétés[37].Les langages de spécification et les méthodes se concentrent plutôt sur un résumé, description de haut niveau qui n'est pas forcément exécutable [37].

Le régulateur de vitesse adaptatif (ACC)[23] est une caractéristique de l'automobile qui permet le contrôle de la vitesse d'un véhicule utilisé pour maintenir la vitesse d'un véhicule dans un intervalle prédéfini.Il est également capable de calculer et de conserver une distance de sécurité avec le véhicule qui le précède sur la même voie [24] .

Dans ce chapitre,nous décrivons d'abord ACC (Adaptive Cruise Control)en utilisant l'algèbre de processusπ-calcul, une telle description formelle consiste en des diagrammes, équations, sémantique formelle, propriétés,spécification du système et vérification.Nous nous somme basé sur les travaux présentés dans (G. Ciobanu et.RUSU)[37].Qui fournissent des spécifications incomplètes.Nous avons complété les spécifications système(ACC) en utilisant l'algèbre de processus π-calcul . MWB est utilisé comme un outil logiciel pour la vérification.

3.1 Régulateur de vitesse adaptatif

Au cours des dernières années, de nombreuses études sur les véhicules intelligents ont été consacrées à la résolution de problèmes tels que la réduction du fardeau du conducteur, la prévention des accidents, le lissage du trafic[33].

Chaque minute, en moyenne, au moins une personne meurt dans un accident[33]. Mentalement, la conduite est une activité très exigeante un conducteur doit maintenir un haut niveau de concentration pendant de longues périodes et être prêt à réagir en une fraction de seconde à des situations changeantes[32].

Le système de régulateur de vitesse (CC) a été développé pour aider le conducteur à conduire de longues distances sur les autoroutes. Le régulateur de vitesse peut effectuer uniquement le contrôle de la vitesse . Le RV(CC) conventionnel devient moins utile en cas de congestion du trafic. Cet inconvénient peut être surmonté par le régulateur de vitesse adaptatif (ACC). Le but de l'ACC est d'éviter la collision arrière en maintenant une distance de sécurité [31].

ACC réduit le stress de la conduite dans un trafic dense en agissant comme un pilote de contrôle . Le système permet d'adapter la distance à la voiture à venir sans l'intervention du conducteur, soulageant efficacement le conducteur.

ACC adapter la vitesse du véhicule à l'environnement de la circulation. Un système radar fixé à l'avant du véhicule est utilisé pour détecter si les véhicules qui circulent plus lentement se trouvent sur le trajet du véhicule ACC.

Si un véhicule qui circule plus lentement est détecté, le système ACC ralentira le véhicule et contrôlera le jeu, ou l'intervalle de temps, entre le véhicule ACC et le véhicule avancé. Si le système détecte que le véhicule avant n'est plus sur le chemin du véhicule ACC, le système ACC accélérera le véhicule jusqu'à sa vitesse de régulateur de vitesse réglée. Cette opération permet au véhicule ACC de ralentir et d'accélérer de manière autonome avec le trafic sans intervention du conducteur.

La méthode de contrôle de la vitesse du véhicule ACC est la commande de l'accélérateur du moteur et le fonctionnement limité du frein[23].

3.1.1 Avantages ACC

Avant tout, le régulateur de vitesse adaptatif améliore le confort de conduite du chauffeur. Du fait que l'ACC veille aux distances de sécurité avec le véhicule qui le précède, la conduite n'en est que plus sereine, ce qui réduit les symptômes de fatigue[23].

•ACC dispose d'une fonction d'alerte de collision qui avertit le chauffeur s'il doit intervenir manuellement[33].

•Le système de freinage d'urgence avancé (AEBS) réduit encore le risque de collision avec le véhicule qui précède en appliquant, si nécessaire, la puissance de freinage maximum disponible[33].

3.1.2 Fonctionnement

Un capteur radar [23] situé derrière la calandre détecte les obstacles situés à l'avant du véhicule et contrôle leurs vitesse et distance relatives et détecter la vitesse et la distance du véhicule qui le précède[24].

Les données sont envoyées aux processeurs qui calculent plusieurs variables (accélération, décélération, changement de voie, distance de rupture) lié à la voiture en face de vous[37].

Les processeurs envoient ces variables à un traitement central ACC unité qui dicte aux autres composants (freins, accélérateur, boîte de vitesses, voyants, systèmes de sécurité etc.) que faire dans la situation actuelle[32].
Si le véhicule principal ralentit ou si un autre objet est détecté, le système envoie un signal

au moteur ou au système de freinage pour décélérer. Ensuite, lorsque la route est dégagée, le système se réaccélère le véhicule à la vitesse préréglée[33] voir figure(3.1).

Les systèmes ACC sont améliorés pour inclure des capacités d'avertissement de collision avertir les conducteurs au moyen de signaux visuels et / ou sonores qu'un accident est imminent et que le direction évasive est nécessaire. Il y a aussi des détecteurs d'angles morts, des aides au stationnement, un pré-armement déploiement de l'airbag et tension de la ceinture de sécurité. L'angle de braquage et les capteurs spéciaux détectent les voies et prédire les courbes de la route, en s'assurant que tout véhicule devant se trouve dans la même voie que le véhicule sujet équipé du système ACC.

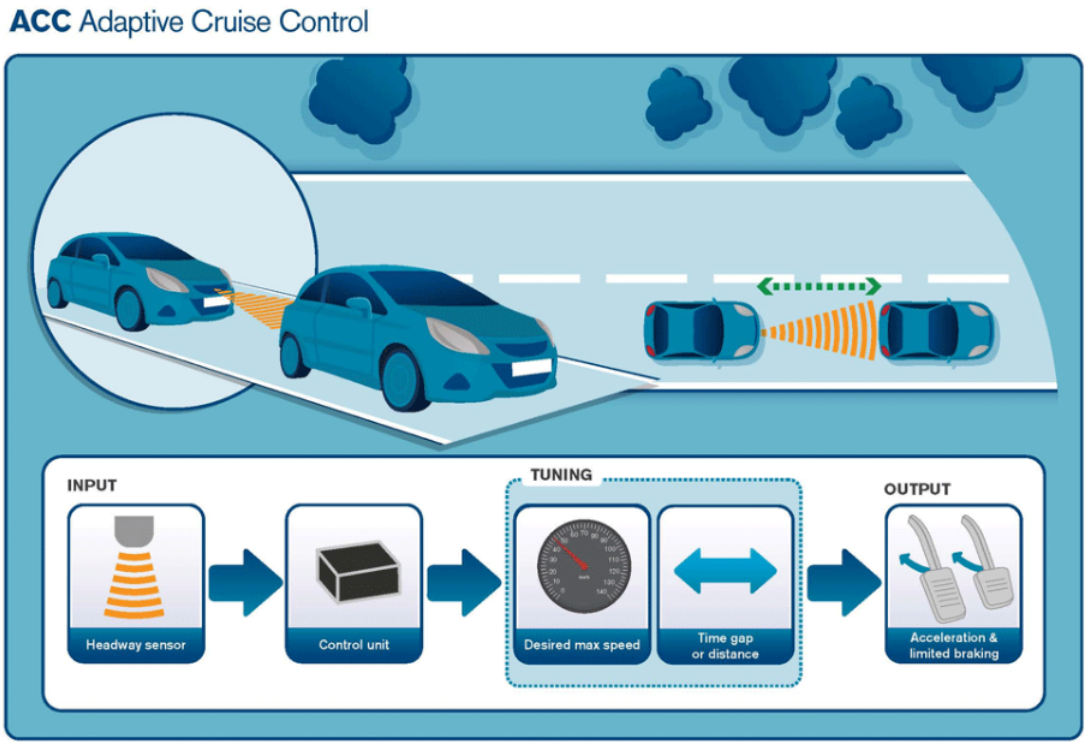


FIGURE 3.1 – ACC Régulateur de vitesse adaptif [23]

3.1.3 Dans quelles circonstances le régulateur de vitesse adaptatif réagit-il ?

Le régulateur de vitesse adaptatif réagit en présence[24] :

•D'obstacles mobiles se rapprochant de l'avant du véhicule, comme par exemple un véhicule le précédant et roulant à une vitesse moindre.

•D'obstacles immobiles détectés précédemment comme étant en mouvement, tels que par exemple une file roulant au ralenti et venant à s'arrêter totalement.

Le régulateur de vitesse adaptatif ne réagit pas en présence :

•D'obstacles s'éloignant du véhicule, tels que des véhicules en cours de dépassement.

•D'obstacles immobiles, comme par exemple un embouteillage à l'arrêt complet lors de sa détection.

•De véhicules roulant en sens inverse.

3.1.4 Limitations

Par rapport à la conduite normale d'un automobiliste, l'acc présente plusieurs limitations restreignant son usage à des conditions bien déterminées[31] :

— L'utilisation de l'acc est réservée à l'autoroute.

— Il existe une vitesse minimum et une vitesse maximum de fonctionnement.

— la décélération est limitée (à environ 1/5 à 1/3 de la puissance maximale) lors d'un freinage

— Les obstacles fixes présents sur la chaussée sont ignorés par le système.

— L'acc doit être compris comme une amélioration du régulateur de vitesse standard pour assister le conducteur, et non pas comme un système de conduite automatisée se substituant à celui-ci. Il s'agit d'une prestation de confort apportant un agrément de conduite appréciable sur autoroute et non pas d'une prestation de sécurité.

3.1.5 Shéma Géneral

Dans le shéma[37] présenté à la figure[3.2], nous incluons tous les composants automobiles interagir sous une unité de contrôle ACC.

Le shéma fournit la fonctionnalité d'un système ACC, modélisé ensuite en utilisant le pi-calcul. Les flèches représentent les canaux de communication entre certaines unités de l'ensemble, en indiquant les directions des signaux et / ou des informations d'un composant à l'autre.

En outre, nous regroupons les unités d'envoi et de réception d'entrée en colonnes séparées, les premiers à gauche du shéma et les seconds à le côté droit.

L'unité matérielle ACC est en haut, visuellement différencié du reste.
Au dessous de sont l'ECU et le VCU, le dernier est tiré deux fois afin de souligner complètement deux rôles différents :
il reçoit des ensembles complets d'informations provenant des capteurs du véhicule et puis, après les avoir édités, il envoie de nouvelles informations et commandes le long d'autres unités, par conséquent, les deux VCU présentés dans notre shéma est en fait un et le même, mais nous préférons inclure deux comme le VCU est un composant très complexe de notre système[37].

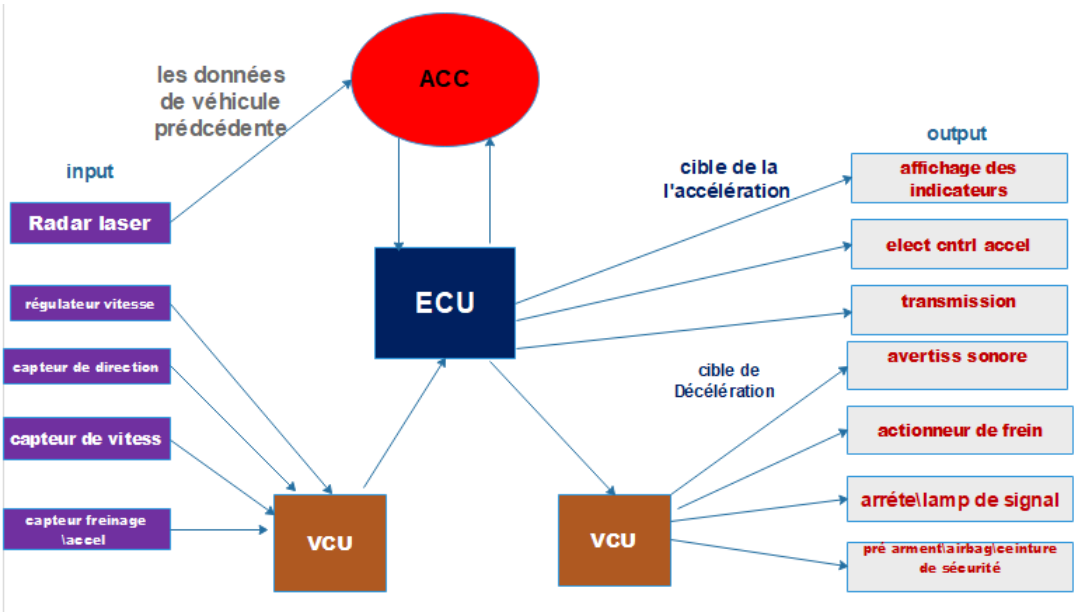


FIGURE 3.2 – Le Shéma général du système ACC[37]

3.1.6 Shéma Fonctionnel du système ACC

À partir du shéma général[37], nous construisons un diagramme fonctionnel qui aide à mieux visualiser l'interaction objets.

Nous regroupons plusieurs systèmes dans groupes fonctionnels connexes :
les capteurs sur le côté gauche.

Le formulaire de régulateur de vitesse une équipe qui envoie des informations sur la statut de la voiture de l'utilisateur, et les deux groupes sur le côté droit sont liés à l'action, séparés par les différentes situations dans lesquelles ils agissent[31].

Nous nommons ces équipes sous le générique noms des environnements, et ils peuvent être facilement identifié dans le shéma présenté à la figure[3.3] suivante :

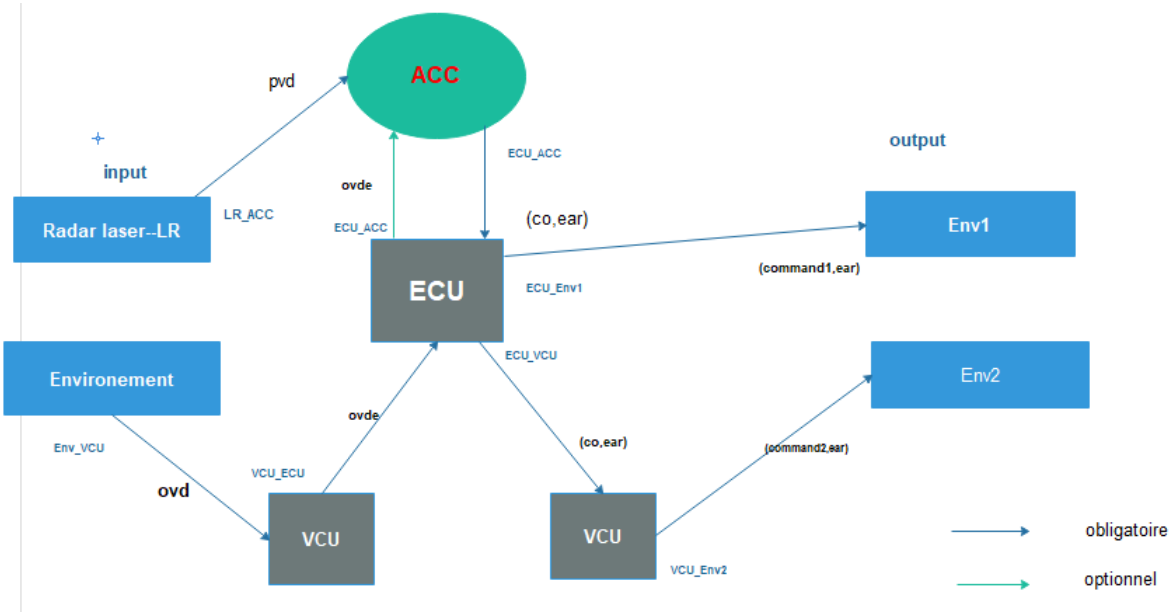


FIGURE 3.3 – Le shéma fonctionnel du système ACC [23]

Nous définissons les groupes fonctionnels suivants[37] :

— **1)Signaux et informations** :

- pvd – (preceding vehicle data)données précédentes du véhicule ;

- ovd – (own vehicle data) données sur le propre véhicule ;

- ovde – (own vehicle data edited) données du véhicule propre éditées ;

- co – ordre de commande ;

- ear (emergency alerte)- Alerte d'urgence :0-pas de commande ;

1 - faible (accélération) ;

2 - normal (décélération) ;

3 - haut (décélération d'urgence) ;

- commande1, commande2 - commandes ou ensembles de commandes de ECU / VCU vers Env1 / Env2.

2) Les canaux de communication entre les unités[37] :

- ACC_ECU, ECU_ACC, ... Canaux de communications entre les dispositifs.

3) System components/units : :

•Radar Laser [32] :

Il y a trois radars dans le système, un à l'avant du véhicule, et un de chaque côté du véhicule. Les radars détectent les obstacles physiques cela peut être sur le chemin du véhicule en mouvement.
Les radars latéraux sont seulement utilisé lorsque le système demande au conducteur de se dérouter vers une voie adjacente et doit s'assurer que la manœuvre est sans danger pour le conducteur.

•Environement :

Entrées, sorties et effets liés à la voiture.

•Capteur / Actionneur d'écran de tableau de bord :

Il y a un écran de tableau de bord dans le véhicule, qui est utilisé pour communiquer avec le conducteur.
Les messages seront affichés sur l'écran pour le conducteur à examiner.
L'écran activera le message du VC communication, ainsi que le sens de la saisie du conducteur[32].

•Régulateur de vitesse (ou cruise control ou autocruise ou autocroisière) :

est un système destiné à stabiliser automatiquement la vitesse des véhicules automobiles[23].

•Capteur de direction

sont les capteurs du système qui sont des boutons sur le volant qui peut être appuyé par le conducteur.

•Capteur de vitesse

est souvent le même que celui utilisé pour le compteur de vitesse. Le capteur le plus commun produit une fréquence de signal pulsé, qui est proportionnelle à la vitesse du véhicule.[23]

•**ACC** :
émetteur permanent avec décision (s'il est activé, il transmet les commandes sans pause[23], il prend également des décisions dans l'exécution des commandes) .Voir le diagramme de séquence dans la figure(3.4) suivant :

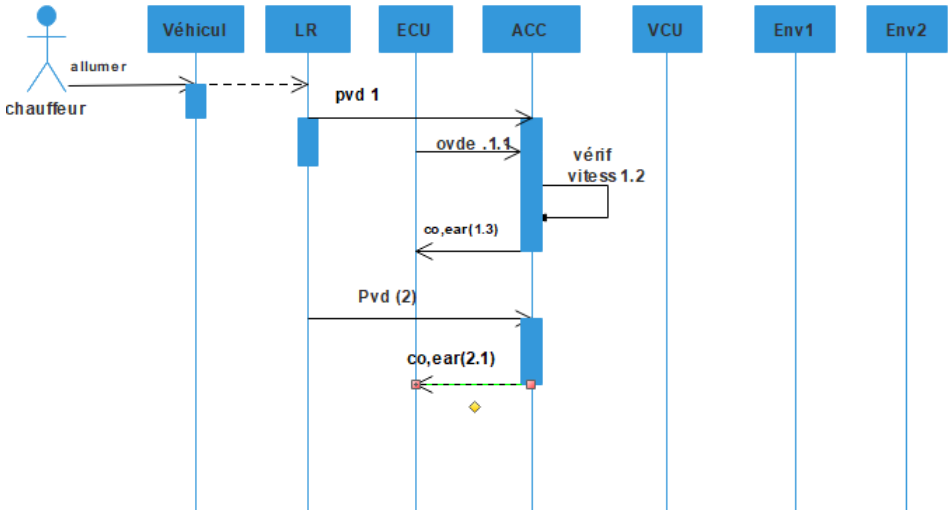


FIGURE 3.4 – Diagramme de séquence agent ACC

.

- **ECU [32] :**

Recevoir informations provenant du module ACC et de la grappe d'instruments et contrôler la vitesse du véhicule sur cette information.

Le module de commande du moteur contrôle la vitesse du véhicule en contrôlant l'accélérateur du moteur.
émetteur d'informations non permanent (vers ACC).voir le diagramme de séquence dans la figure[3.5] suivante :

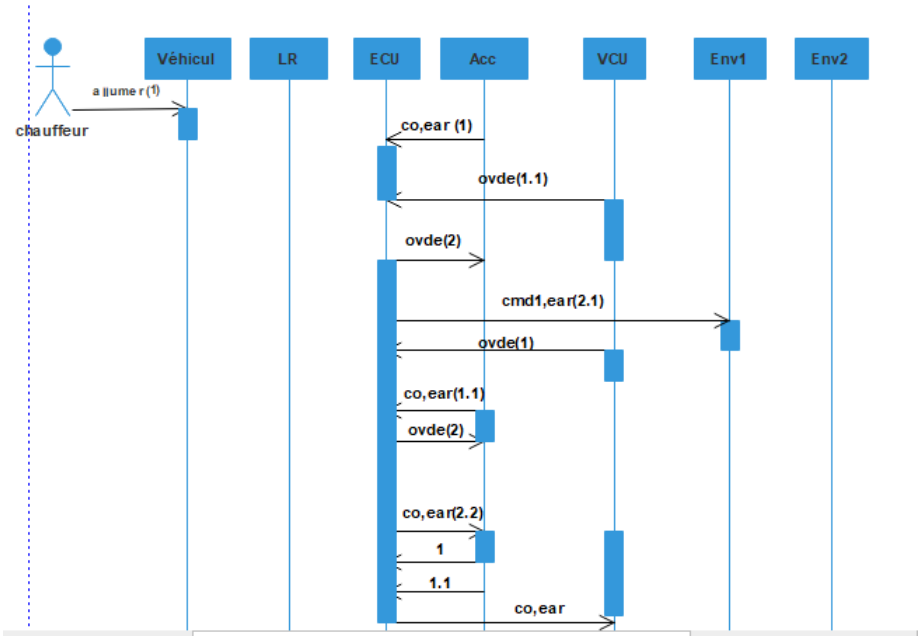


FIGURE 3.5 – Diagramme de séquence agent ECU

• **VCU [33] :**

unités cntrole de véhicule il reçoit des ensembles complets d'informations provenant des capteurs du véhicule et puis, après les avoir édités, il envoie de nouvelles informations et commandes le long d'autres unités.voir le diagramme de séquence dans la figure[3.6] suivante :

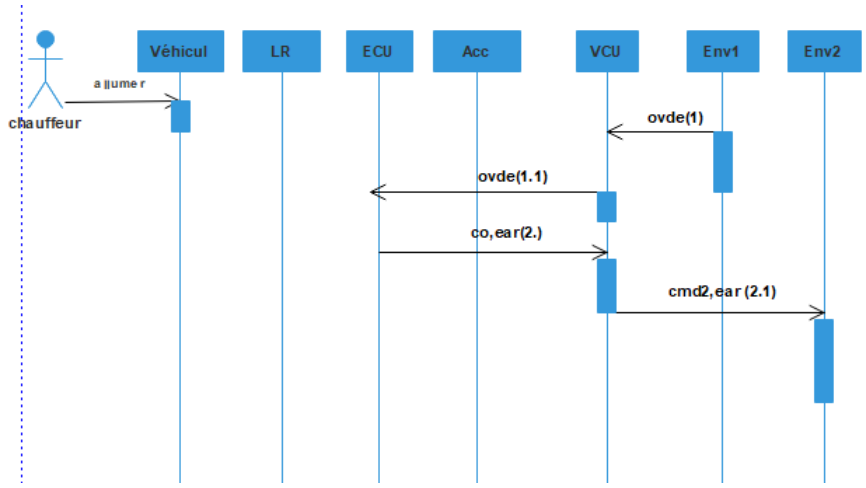


FIGURE 3.6 – Diagramme de séquence agent VCU

• **ECU, VCU [20] :**
unités de commande (ECU envoie commande1 à exécuter et VCU envoie commande2 être éxecuté).

3.1.7 Diagramme de séquence géneral

Les diagrammes de séquences sont des modèles largement utilisés en début de conception des systèmes. Ce couplage fournira à des concepteurs des moyens pour évaluer la cohérence de différents types de modèles comportementaux.
Voici les diagrammes de séquence pour le système ACC. Ces diagrammes représentent le comportement séquentiel du système. Les diagrammes de séquence sont modélisés après le système scénario.

•Scénario
Le conducteur active le véhicule et augmente leur vitesse manuellement.Le conducteur appuie sur le bouton d'activation sur le volant, ce qui permet croisière.La vitesse de croisière du véhicule est réglée sur la vitesse actuelle du véhicule.Il marche par une vitesse normale Le conducteur détient le bouton d'accélération pour augmenter la vitesse de croisière à certaine vitesse.Il y a un ou des véhicules devant va plus lentement.Laser radar détecté un obstacle et envoie comme un information a ACC qui reçoit aussi la vitesse la distance et les propres données de véhicule et Acc décide d'augmenter ou de diminuer la vitesse.Si il accéléré envoie ordre de commande vers Ecu.Et Ecu envoie la commande1 d'accélération a l'environnement d'accéléré.Si non Ecu envoi L' ordre de commande vers Vcu et Vcu envoie la commande2 pour décélérer a l'environnement de freinage et diminuer la vitesse.

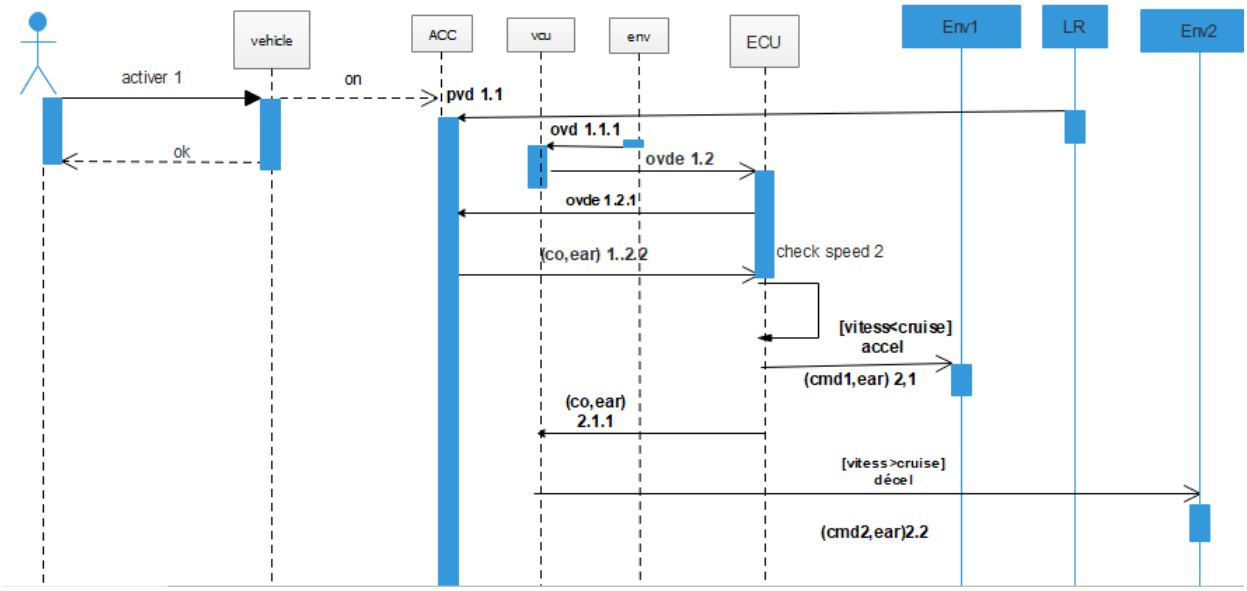


FIGURE 3.7 – Diagramme de séquence de systeme Acc

- **Accélération du véhicule [23]**
  si La route est vide et il n'y a pas d'obstacles, ou une voiture à l'avant accélère, l'ACC accélère aussi notre véhicule au préréglagela vitesse. Par conséquent, aucune assistance de freinage n'est nécessaire.

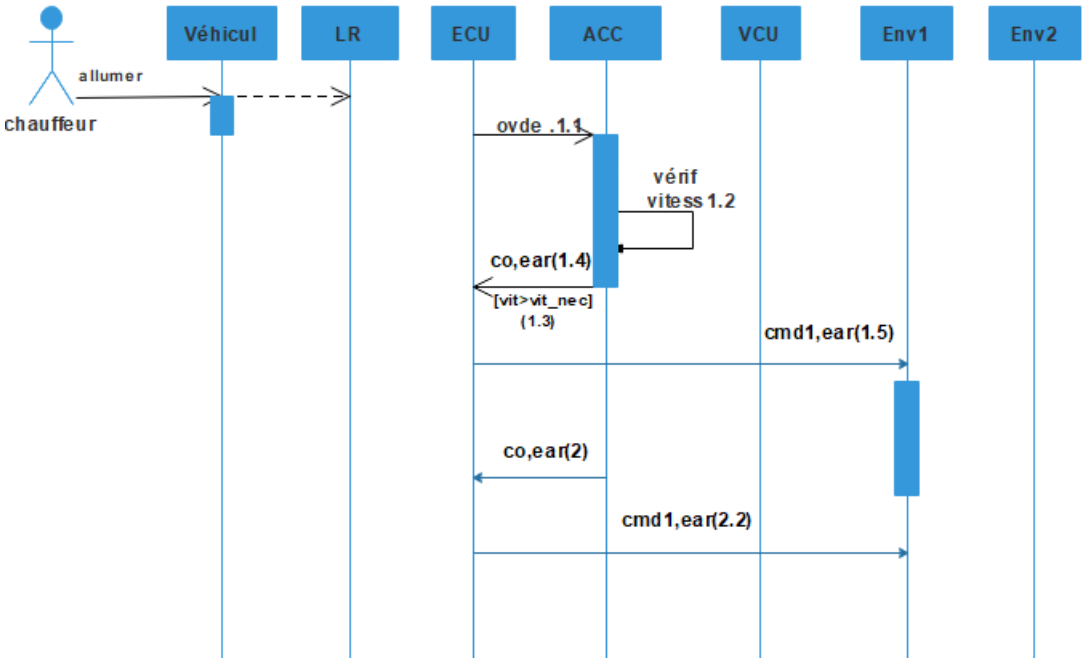


FIGURE 3.8 – Diagramme séquence agent accéleration

- **Décélération du véhicule [23]**
  Si le véhicule à l'avant décélère,ou il y'a d'obstacles l'ACC ralentit également notre véhicule. Donc, là est nécessaire pour l'assistance au freinage, que ce soit par la gestion du moteur ou par le freinage système.

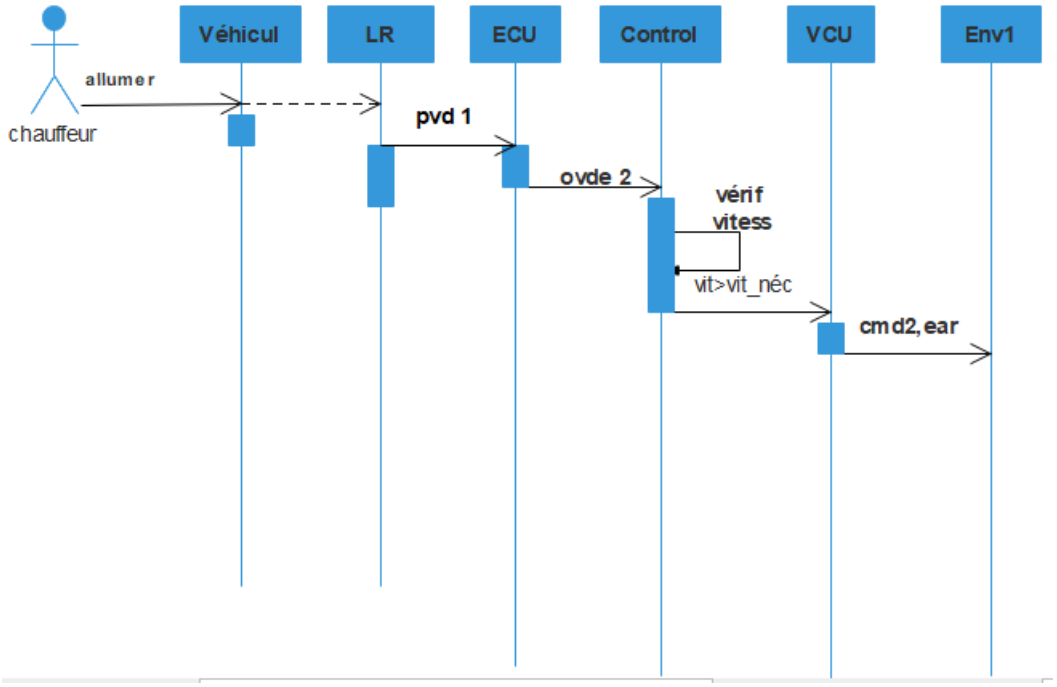


FIGURE 3.9 – Diagramme séquence Agent Déceleration

- **L'état Normal(Norm) [20]**
  état normal pour l'ACC lorsque ni l'accélération ni la décélération ne sont présentes.

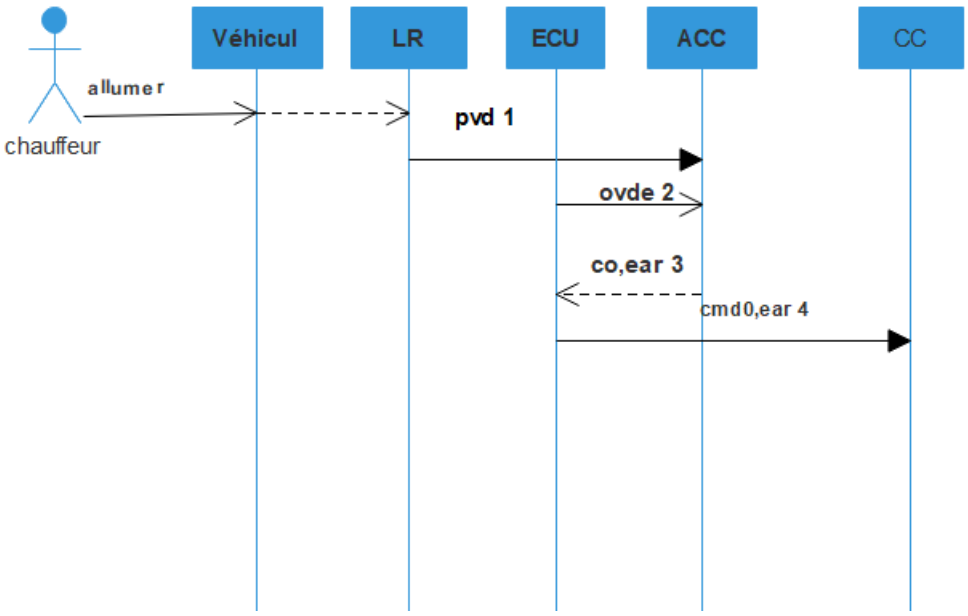


FIGURE 3.10 – Diagramme séquence agent Norm

- **le contrôleur cordinateur)[23]**
  Coordonne tous les sous-systèmes, Détecte la vitesse du véhicule, la vitesse du véhicule principal et s'ajuste pour maintenir une distance de sécurité,ou pour l'accélération ou la décélération voir figure(3.11).

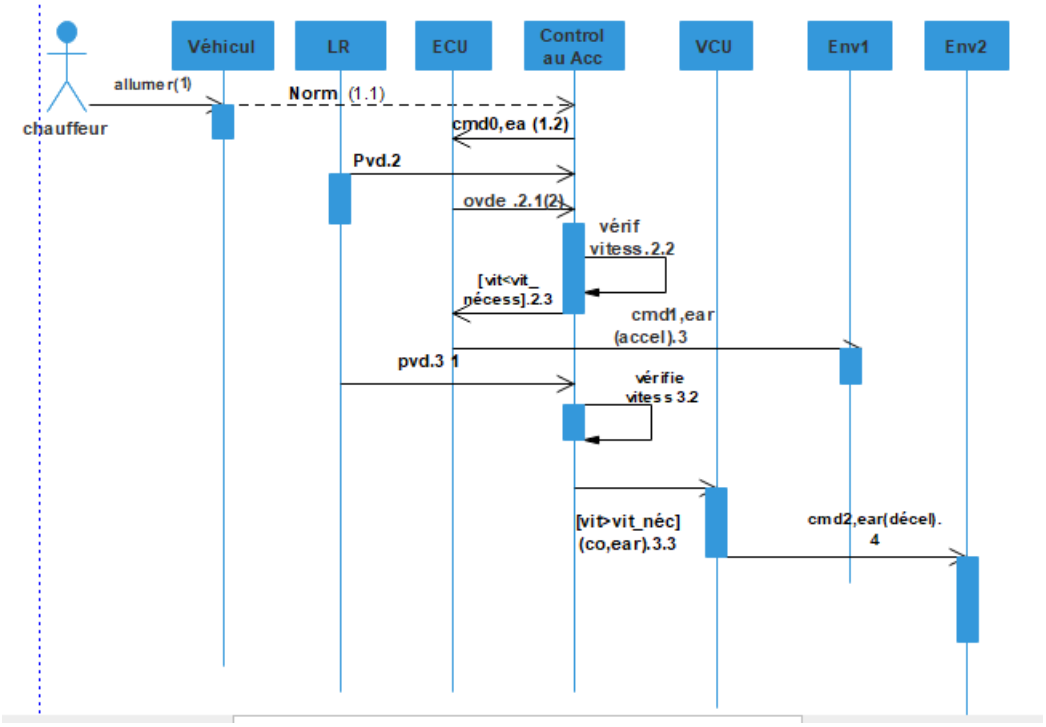


FIGURE 3.11 – Diagramme séquence de controleur Acc

3.2 Spécification à l'aide du π-Calcul

la spécification est l'une des phases importantes dans le processus de développement,la spécification formelle offrant les moyens de vérification et de validation des systèmes. La réalisation d'une spécification formelle exige la mise en place d'un formalisme dédié à la mobilité.

Afin d'illustrer la fonctionnalité de l'interaction des signaux et des données dans les automobiles équipées de régulateur de vitesse adaptatif, nous utilisons une algèbre de processus appelée π-calcul [14] pour la modélisation.

Une algèbre de processus [13] est un formalisme utilisé pour spécifier le comportement des systèmes dans un manière précise, modulaire et hiérarchique. Les blocs de construction de base sont des processus et des canaux.

Nous nous référons aux algèbres de processus comme le Calcule système de Communication (ccs) [13] et le π-calcul [14] [15] , et soutient l'analyse des processus d'interaction. Nous pouvons dessiner des shémas [20] de les processus en tant que cercles, et les interactions entre eux en tant que lignes de connexion. Dans CCS, le les processus peuvent mourir ou se diviser en deux, mais de nouveaux liens ne peuvent pas être créés.

En revanche, le π-calcul est capable de créer de nouveaux liens[16]. Le π-calcul généralise le CCS en transmettant des noms plutôt que des valeurs sur les canaux

d'envoi-réception, tout en préservant la correspondance en tant que contrôle structure pour la communication. Puisque les variables peuvent être des noms de canaux, le calcul peut changer la topologie du réseau, et donc la mobilité est supportée.

Le π-calcul est un modèle largement accepté de systèmes en interaction avec dynamiquement évolution de la topologie de la communication. Il permet aux canaux d'être transmis en tant que données le long d'autres canaux, et cela introduit la mobilité des canaux.

Une caractéristique importante du π-calcul est que la mobilité est exprimé par l'évolution de la configuration et de la connectivité entre les processus. Cette mobilité augmente le pouvoir expressif, permettant la description de nombreuses fonctionnalités concurrentes de haut niveau.

Le π-calcul a une sémantique simple et une théorie algébrique traitable. Le calcul Le monde du π-calcul contient seulement des processus (aussi appelés agents) et des canaux (appelés aussi noms ou ports). Il modélise les réseaux dans lesquels les messages sont envoyés d'un site à un autre et peut contenir des liens vers des processus actifs ou vers d'autres sites. Le π-calcul est un modèle général de calcul qui prend l'interaction comme une primitive. Pour la simplicité, nous présentons le poladique version dπ-calcul [14] : cela signifie qu'un message comporte exactement un nom. Nous utilisons la version monadique poladique [14] [16].

• On peut remarquer que les noms des canaux sont choisis de telle sorte que la direction de la transmission est facilement compréhensible.
Par exemple, l'utilisation du nom de chaîne LR_ACC indique qu'il y a des données envoyées de LR à ACC.

3.2.1 Spécification des agents du système

•Agent Acc[37] :

ACC = (LR_ACC(pvd) |ECU_ACC(ovde)).$\overline{Acc\_Ecu}$ <co, ear>.ACC

+ LR_ACC(pvd) . $\overline{Acc\_Ecu}$ <co, ear>.ACC

•agent Ecu :

ECU = (VCU_ECU(ovde) | ACC_ECU(co, ear)).

.($\overline{Ecu\_Acc}$ <ovde> | $\overline{Ecu\_Env1}$ <command1,ear>.ECU+

+ (VCU_ECU(ovde) | ACC_ECU(co, ear))

.($\overline{Ecu\_Acc}$ <ovde> | $\overline{Ecu\_Vcu}$ $<co, ear>$).ECU

+ (VCU_ECU(ovde) | ACC_ECU(co, ear)) . $\overline{Ecu_Env1}$$<command1, ear>$.$ECU$

+ (VCU_ECU(ovde) | ACC_ECU(co, ear)). $\overline{Ecu\_Vcu}$ <co, ear>.ECU

•agent Vcu :

VCU = Env_VCU(ovd). $\overline{Vcu_Ecu}$$<ovde>$.$VCU$

+ $ECU\_VCU(co, ear)$.$\overline{Vcu_Env2}$$<command2, ear>$.$VCU$

•agent LR :

LR = $\overline{LR\_Acc}$<pvd>.LR

•agent Environement :

Environement = Env | Env1 | Env2

•agent Env :

Env = $\overline{Env\_VCU}$ <ovd>.ECU_Env1(command1, ear).Env

+$\overline{Env\_VCU}$ <ovd>.VCU_Env2(command2, ear).Env

•agent Env1 :

Env1(ecu_env1,commande1,ear)=ecu_env1(commande1,ear).

.Env1(ecu_env1,commande1,ear)

•agent Env2 :

Env2(vcu_env2,commande2,ear)=vcu_env2(commande2,ear).

.Env2(vcu_env2,commande2,ear)

•agent Norm :

Norm(vcu_ecu,acc_ecu,ecu_acc,vcu_ecu,lr_acc,pvd,ovde,co,ear) =

$Acc(lr\_acc, ecu\_acc, acc\_ecu, pvd, ovde, co)$

•agent Contrôleur :

agent Controller(acc_ecu,ecu_acc, ecu_vcu,ecu_env1,vcu_env2, vcu_ecu, co,ear,ovde,comman
,command2 ,lr_acc, pvd)=

= Accel(acc_ecu,ecu_acc,ecu_env1,co,ear,ovde,command1, lr_acc,pvd) |

| Decel(acc_ecu, ecu_acc, ecu_vcu, ecu_env1, vcu_env2,co,ear,ovde,command1,command2,lr_

| Norm(vcu_ecu, acc_ecu, ecu_acc, vcu_ecu, lr_acc, pvd, ovde, co, ear)

•agent Système[37 :

agent Système= (Lr + Env) | Contrôleur | (Env1 + Env2)

•agent Accéleration :

ACCEL = ACC ($\overline{Acc\_ECU}$ <co, ear>.ECU_ACC(ovde) +

($\overline{Acc\_ECU}$ <co, ear>) .ECU(ACC_ECU(co, ear).($\overline{Ecu\_Acc}$ <ovde> | ($\overline{Ecu\_Env1}$

<$command1, ear$>) + $ACC\_ECU(co, ear)$.

$\overline{Ecu\_Env1}$ <command1,ear>)). Env1(ECU_Env1(command1, ear))

•agent Decel :

DECEL =ACC ($\overline{Acc\_ECU}$ <co, ear>.ECU_ACC(ovde) +$\overline{Acc\_ECU}$ <co, ear>)

.ECU (ACC_ECU(co, ear).($\overline{Ecu\_Acc}$ <ovde> | $\overline{Ecu\_Vcu}$ <co, ear>)+$ACC\_ECU(co, ear)$.

$\overline{Ecu\_Vcu}$ <co, ear>)

.$VCU(ECU\_VCU(co, ear).VCU_Env2 < command2, ear > \overline{Vcu\_Env2}$ <command2, ear>)

.Env2(VCU_Env2(command2,ear))

3.2.2 propriétés

•**Propriété(1) [20]** :

Si ear = 3, alors la séquence d'actions commence par ACC (qui envoie ear) et continue tout droit à travers ECU à VCU,ui finalise la séquence en lançant commande2 avec alerte d'urgence maximale.
Dans cette situation, il n'y a pas de transmission sur le tube ECU_ACC, en raison de l'état d'urgence émis par ACC à savoir ear = 3. L'équation de pi-calcul pour cette situation est :

EAR_3 = ACC ($\overline{Acc\_Ecu}$ <co, ear>).ECU (ACC_ECU(co, ear).

. $\overline{Ecu\_Vcu}$ <co, ear>).$VCU(ECU\_VCU(co, ear).VCU\_Env2 < command2, ear >$)

•**Propriété(2) [20]** :

Si ear = 2, la séquence d'actions suit précisément le modèle décrit dans le équations de la sous-section 3.2.1.

l'équation de π-calcul pour un EAR_2 est identique à DECEL.

•**Propriété(3)** [**20**] :

Si ear = 1, alors la séquence d'actions commence par ACC (qui envoie ear) et continue à ECU, qui lance command1.
L'équation de π-calcul pour un événement EAR_1 est identique à ACCEL.

•**Propriété(4)** :

Si ear = 0, aucun ordre de commande n'est émis et il n'y a que des transmissions des données à ACC de LR et ECU (qui reçoit de Env à travers VCU), jusqu'à ce que l'ACC décide, sur la base des données reçues, s'il est nécessaire d'envoyer un ordre de commande.

$\mathrm{EAR}_0 = (LR(\overline{LR\_ACC}<pvd>).ACC(LR\_ACC(pvd)))|$

| (ECU(VCU_ECU(ovde). $\overline{Ecu\_ACC}$ <ovde>).ACC(ECU_ACC(ovde)))

3.3 Mobilité Workbench (MWB)

Le Mobility Workbench (MWB)[17] [9] [17] [18] est un outil de manipulation et d'analyse système concurrent mobile qui sont définis en π-calcul. L'outil a été développé par B.Victor et F.Moller dans [12] à l'université d'Upsalla, en Suède. Il est écrit en ML standard, et fonctionne le compilateur SML du New Jersey.

Certaines caractéristiques de cet outil sont la décision de bisimulation ouverte [12] [9], la recherche de blocages [17], simulation d'agents et vérification de modèle [33]. Dans la version actuelle, l'outil prend en charge commandes pour le π-calcul.

Victor dans [12] donne quelques informations sur l'algorithme qui sont implémentées dans MWB. L'algorithme est utilisé pour décider de la bisimulation ouverte de Sangiorgi [33] pour les agents dans leπ-calcul avec le positif original match opérateur. L'algorithme est basé sur les caractérisations alternatives efficaces des équivalences dans [9] [35], et génère l'espace d'état à la volée. Le la mise en œuvre d'algorithmes comprend à la fois des équivalences fortes et faibles. Pour la fonctionnalité de vérification de modèle, l'outil implémente l'algorithme lié à le travail de Dam [40].
On utilise MWB comme modele checking .

3.3.1 Syntaxe du script MWB

•La syntaxe pour les agents π-calcul

la syntaxe qui définir les agents en (MWB)[12] correspond dans ce tableau[3.1] :

Syntax	Meaning
0	The null process
α . P	Action prefix
$prefix$. P	Prefix
[a = b] P	Match
P_1 | P_2	Parallel
P_1 + P_2	Summation
Id < $nlist$ >	Application
(^ $nlist$) P	Restriction
(\ $nlist$) P	Abstraction
[$nlist$] P	Concretion
(P)	Parenthesis

TABLE 3.1 – La syntaxe MWB pour la définition d'agents [17].

Dans la syntaxe, la liste nlist représente une liste de noms séparés par des virgules non vides [17]. Les noms doivent être démarrés avec une lettre minuscule, mais après peuvent inclure des caractères spéciaux , lettres et chiffres. Le α représente un nom, un co-nom ou un silencieux action (τaction).
Nous utilisons un guillemet simple (') pour représenter une barre sur un co-nom, et utilisez "t" pour représenter l'action τ. Le préfixe représente l'entrée ou nom de sortie

avec paramètre. L'identifiant est un nom commençant par une majuscule lettre [17] [18].

Nous utilisons " ˆ " pour représenter "ν" ou d'autres symboles de restriction. "\" est représentation du symbole "λ" dans l'abstraction. Nous pouvons diviser les lignes d'entrée en utilisant le caractère de continuation " \ " à la fin d'une ligne d'entrée.L'opérateur parallèle "|" se lie plus fort que sommation "+". Les deux se lient plus faible que le préfixe "." Et correspondent à "[...]".

• Syntaxe pour les formules modales

La syntaxe MWB pour définir les formules [30] correspond à la syntaxe présentée en 2.7 La syntaxe des formules est donnée dans le tableau [3.2].

Syntax	Meaning
`TT`	Truth
`FF`	Falsity
name `=` *name*	Equality between names
name `#` *name*	Inequality between names
formula `&` *formula*	Conjunction
formula `|` *formula*	Disjunction
`<`*action*`>`*formula*	Possibility modality
`[`*action*`]`*formula*	Necessity modality
`Sigma` *name* `.` *formula*	Sigma expression
`Bsigma` *name* `.` *formula*	Bound sigma expression
`Pi` *name* `.` *formula*	Universal quantification
`exists` *name* `.` *formula*	Existential quantification
op id `.` *formula*	Fixpoint expression
`(`*formula*`)`	Parenthesis

TABLE 3.2 – La syntaxe MWB pour la définition des formules [30].

Dans la syntaxe, l'op représente l'opérateur de point fixe mu (représentant "μ"), min, nu (représentant "ν"), et max. L'id représente l'identificateur de point fixe qui devrait être commencé avec une lettre majuscule.

3.4 spécification MWB

Nous utilisons les spécifications de π-calcul du système ACC présentées dans la figure[3.3] .Les processus π-calcul sont appelés agents dans MWB.
Nous présentons le code MWB pour certaines unités représentatives du système (agents nommés) ACC et ECU,VCU,ENV,Env1,Env2 Accel, Decel et Norm agents, qui décrivent des événements pour l'accélération, la décélération et un état normal pour l'ACC (lorsque ni l'accélération ni la décélération ne sont présentes), sont définis dans de la même manière et sont regroupés dans l'agent du système avec l'utilisation d'un commutateur représenté par l'agent du contrôleur.

•agent ACC :

Acc(lr_*acc*, *ecu*_acc,acc_*ecu*, *pvd*, *ovde*, *co*, *ear*) = *lr*_acc(pvd).ecu_acc(ovde).'acc_*ecu* < *co*, *ear* > .
.*Acc*(*lr*_acc,ecu_*acc*, *acc*_ecu,pvd,ovde,co,ear)

•agent ECU :

Ecu(vcu_ecu,acc_ecu,ecu_acc,ecu_vcu,ecu_env1,ovde,co,ear,env1,command1)=

vcu_ecu(ovde).Ecu(vcu_ecu,acc_ecu,ecu_acc,ecu_vcu,ecu_env1,ovde,co,ear,env1,command1)

|(acc_ecu(co,ear).'ecu_acc<ovde>).

.Ecu(vcu_ecu,acc_ecu,ecu_acc,ecu_vcu,ecu_env1,ovde,co,ear,env1,command1)

|'ecu_env1<command1,ear>.Ecu(vcu_ecu,acc_ecu,ecu_acc,ecu_vcu,ecu_env1,ovde,co,ear,env1,c

.Ecu(vcu_ecu,acc_ecu,ecu_acc,ecu_vcu,ecu_env1,ovde,co,ear,env1,command1)

•agent VCU :

Vcu(env_vcu,vcu_ecu,ecu_vcu,vcu_env2,ovd,ovde,co,ear,command2)=env_vcu(ovd).'vcu_ecu<c

. Vcu(env_vcu,vcu_ecu,ecu_vcu,vcu_env2,ovd,ovde,co,ear,command2)

+ecu_vcu(co,ear).'vcu_env2<*command*2, *ear*>.

.*Vcu*(*env_vcu*, *vcu_ecu*, *ecu_vcu*, *vcu_env*2, *ovd*, *ovde*, *co*, *ear*, *command*2)

•agent Controller :

agent Controller(acc_ecu,ecu_acc, ecu_vcu,ecu_env1,vcu_env2, vcu_ecu, co,ear,ovde,command1 ,command2 ,lr_acc, pvd)=

= Accel(acc_ecu,ecu_acc,ecu_env1,co,ear,ovde,command1, lr_acc,pvd) |

| Decel(acc_ecu, ecu_acc, ecu_vcu, ecu_env1, vcu_env2,co,ear,ovde,command1,command2,lr_acc,pvd)

| Norm(vcu_ecu, acc_ecu, ecu_acc, vcu_ecu, lr_acc, pvd, ovde, co, ear)

•agent Systèm :

agent System= (Lr + Env) | Controller | (Env1 + Env2)

La spécificaion de π dans l'outil Mwb dans les figures suivntes :

•démarrer MWB

Ouvrez l'outil MWB a partir le fichier bin qui se trouve dans le SLMNJ Figure [3.12] :

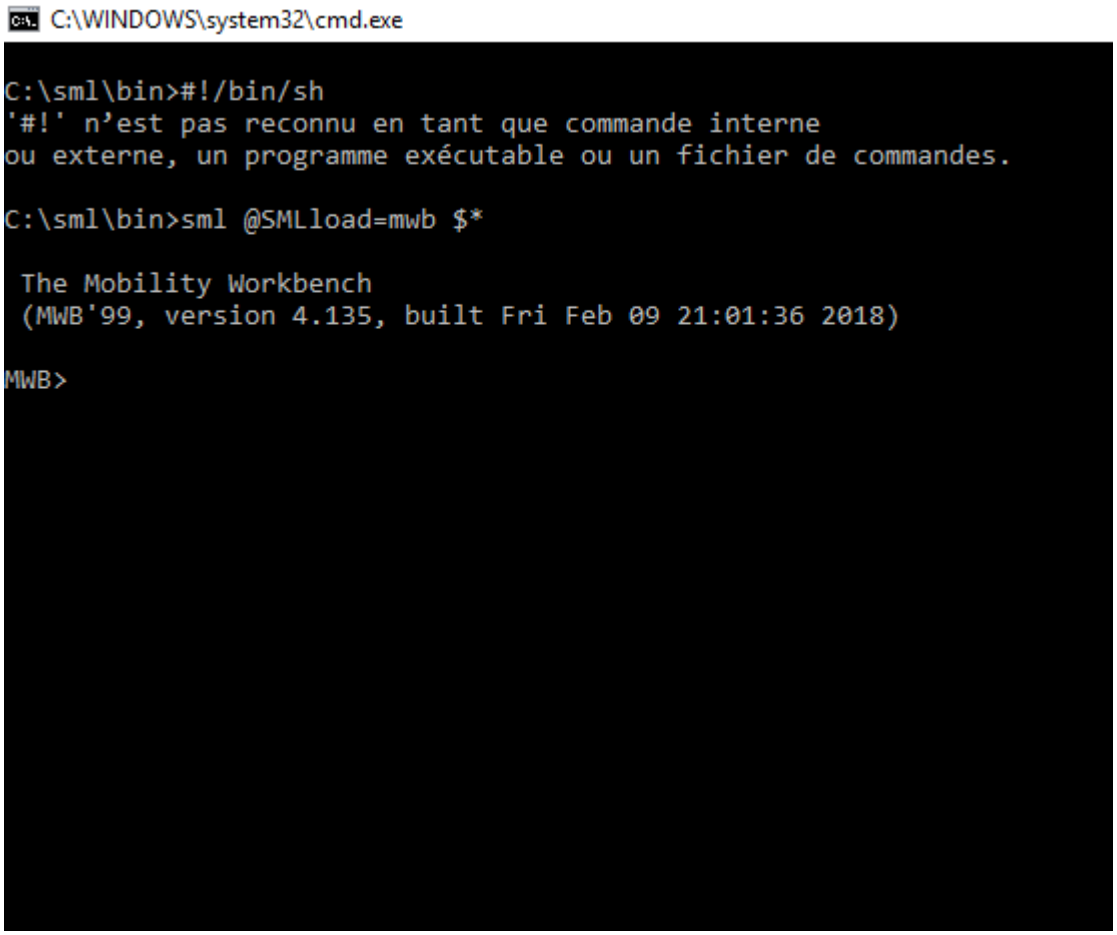


FIGURE 3.12 – Démarrer Mwb

•importer et afficher les agents

Ouvrez l'outil MWB et entrez la commande : "Acc.txt" pour importer les agents , et à travers la commande "env" tout les agents définis sont affichés. L'opération est montrée dans Figure [3.13] :

FIGURE 3.13 – Spécification systeme ACC en Mwb

3.5 Vérification

La vérification du logiciel peut être considérée comme le processus de vérification de certaines propriétés P L'approche de vérification de modèle pour la vérification [40] consiste à extraire la clé éléments du logiciel et de vérifier seulement ces éléments. Ces abstractions clés sont des prédicats binaires, et diverses techniques et structures ont été développé pour vérifier automatiquement et efficacement les éléments abstraits contre propriétés spécifiées. Compte tenu de la dépendance sous-jacente aux abstractions binaires, n'est pas surprenant que la vérification du modèle soit utilisée dans l'analyse des circuits électroniques, mais il s'est également avéré efficace dans le domaine du logiciel, en particulier dans les domaines de l'analyse de protocole, le comportement des systèmes réactifs, et pour vérifier les systèmes mobiles[9]. On utilise MWB comme un model checking .

Nous avons l'intention d'appliquer cette approche au modèle π-calcul d'ACC.
Nous avons également réalisé des expériences pour équivalence observationnelle dans laquelle nous montrons que le système décrit a le même comportement avec un système idéal (un système sans échecs).

Une fonctionnalité importante de la MWB est de décider de la bisimilarité ouverte forte et faible entre deux systèmes décrits comme des agents, ainsi que la vérification des blocages et autres propriétés exprimées en formules π-calcul.

3.5.1 Step Agents

utiliser la commande "step" pour la simulation et implémentation des agents[17] . Chaque agent du modèle ACC est vérifié selon aux étapes ci-dessus, et les résultats montrent que chaque agent

Step Acc

Vérifier les étapes d'exécution d'agent Acc et vérifier si le modèle de plainte est capable de s'exécuter correctement.les résultats d'opération sont montrés dans figure[3.14]

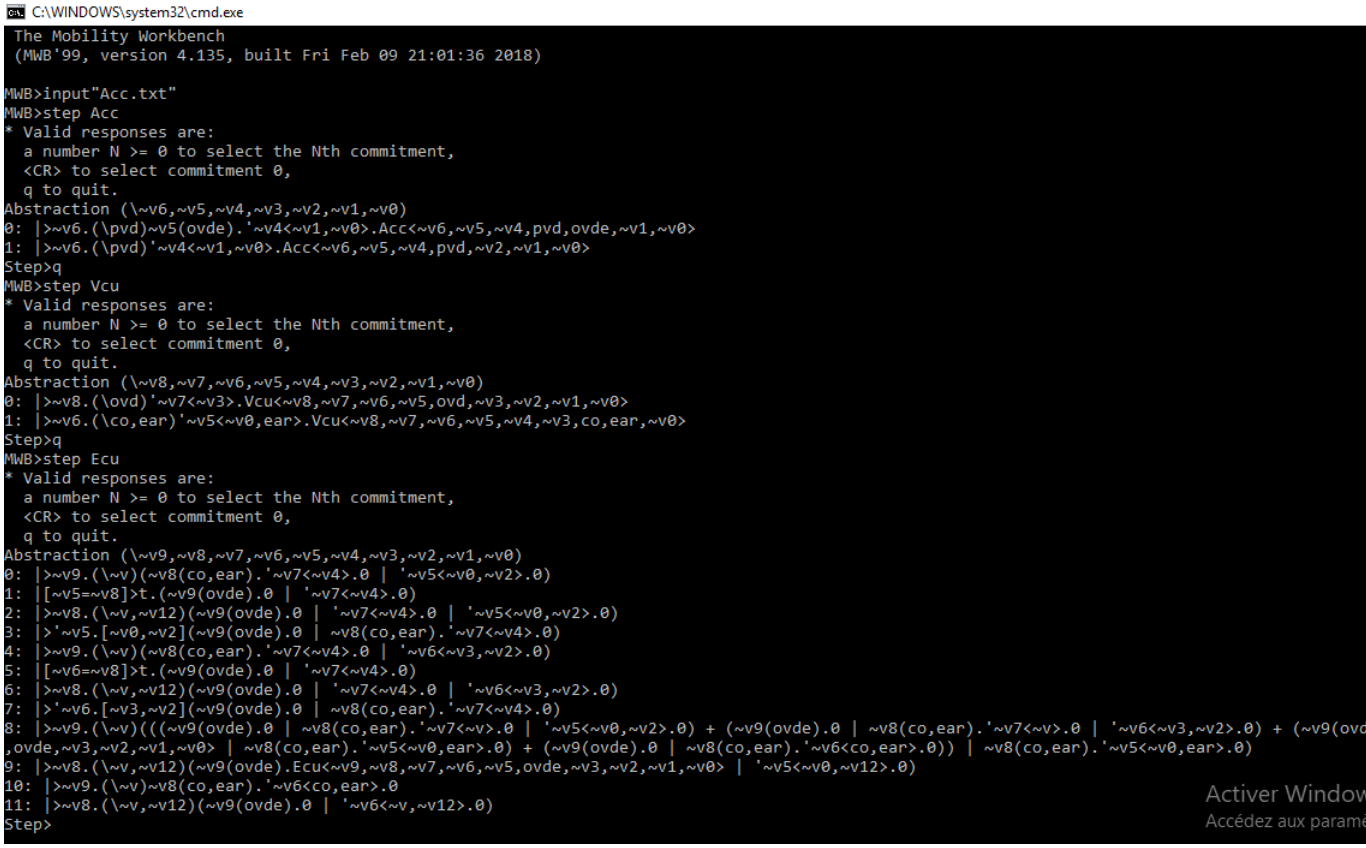


FIGURE 3.14 – Étapes exécutives à vérifier pour des'agent de systeme ACC Mwb

Step Accel

Vérifier les étapes d'exécution d'agent Accel .Et vérifier si le modèle de plainte est capable de s'exécuter correctement.Les résultats d'opération sont montrés dans figure[3.15]

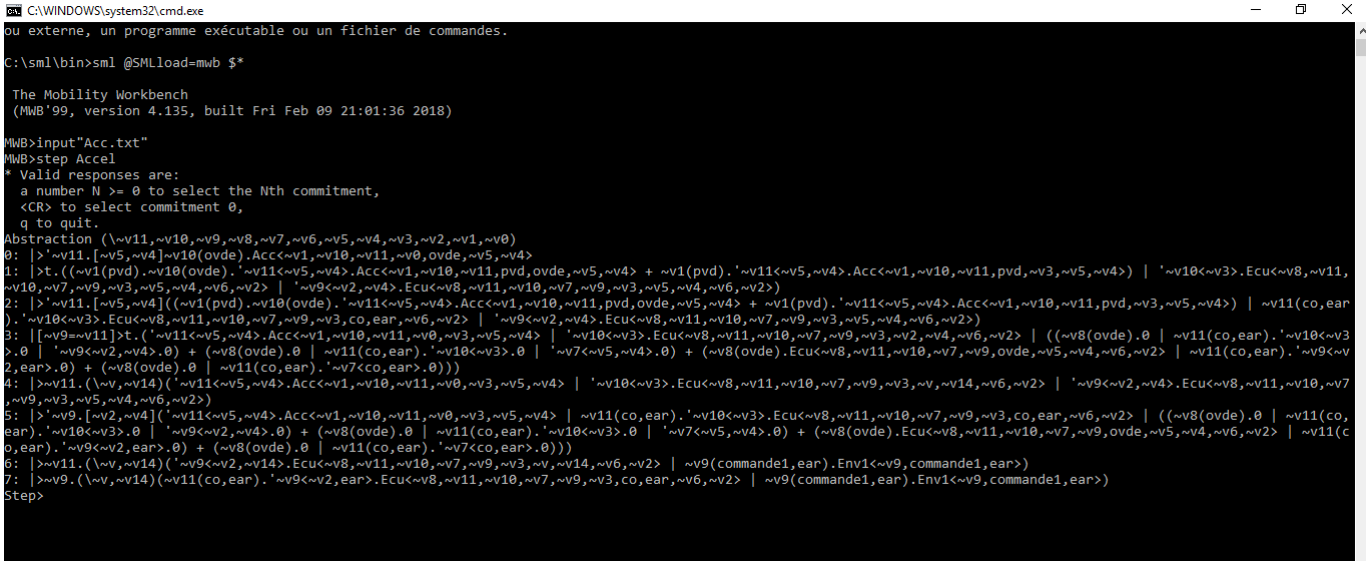


FIGURE 3.15 – Étapes exécutives à vérifier pour l'agent Accel(Mwb)

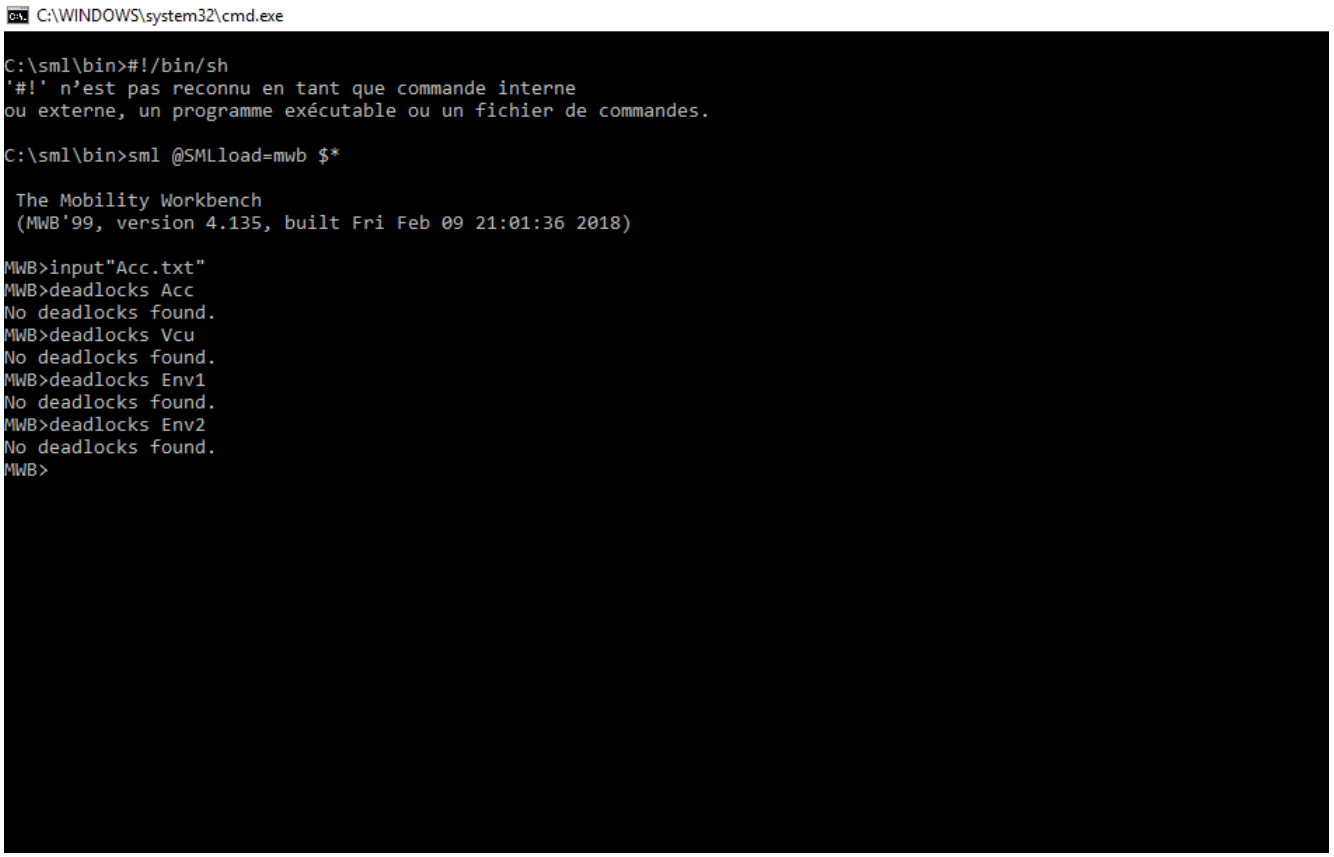
Step Decel

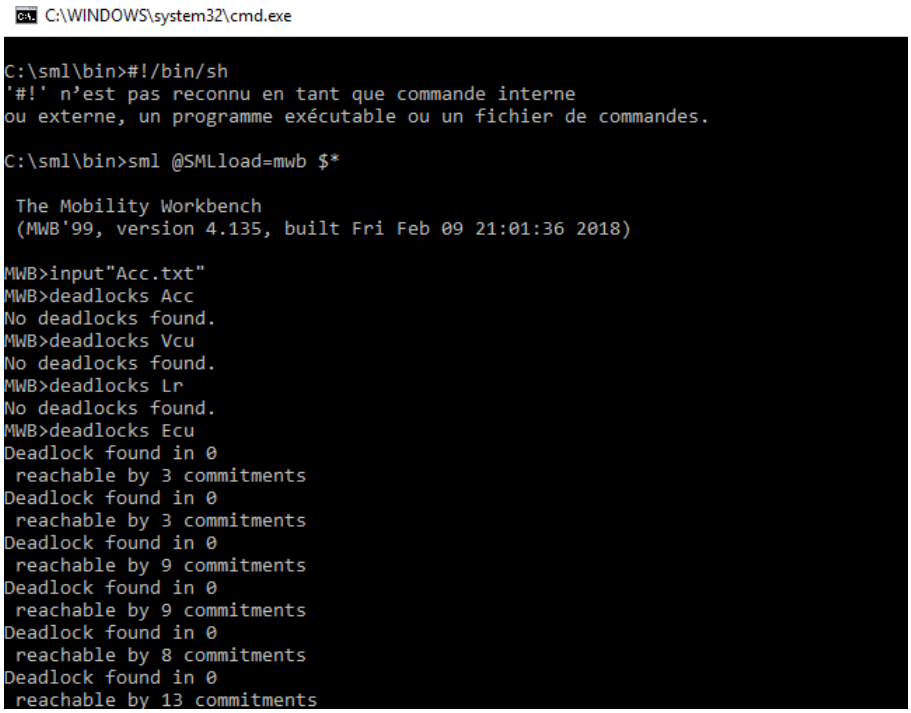
Vérifier les étapes d'exécution d'agent Decel .Et vérifier si le modèle de plainte est capable de s'exécuter correctement.Les résultats d'opération sont montrés dans figure[3.16]

FIGURE 3.16 – Étapes exécutives à vérifier pour l'agent Decel (Mwb)

3.5.2 Blocages Agents (deadlocks Agents)

trouve et décrit les blocages dans l'agent donné comme argument il affiche l'agent dans lequel les blocages est trouvé[16] . Les blocages sont affichés comme ils sont trouvés ce qui rend la commande utile même si l'espace d'état est infini.
Le blocage de chaque agent a été examinée dans le Modèle ACC, et les résultats prouvent que chaque agent n'existe pas de blocage les résultats dans les figures suivantes(3.17) :





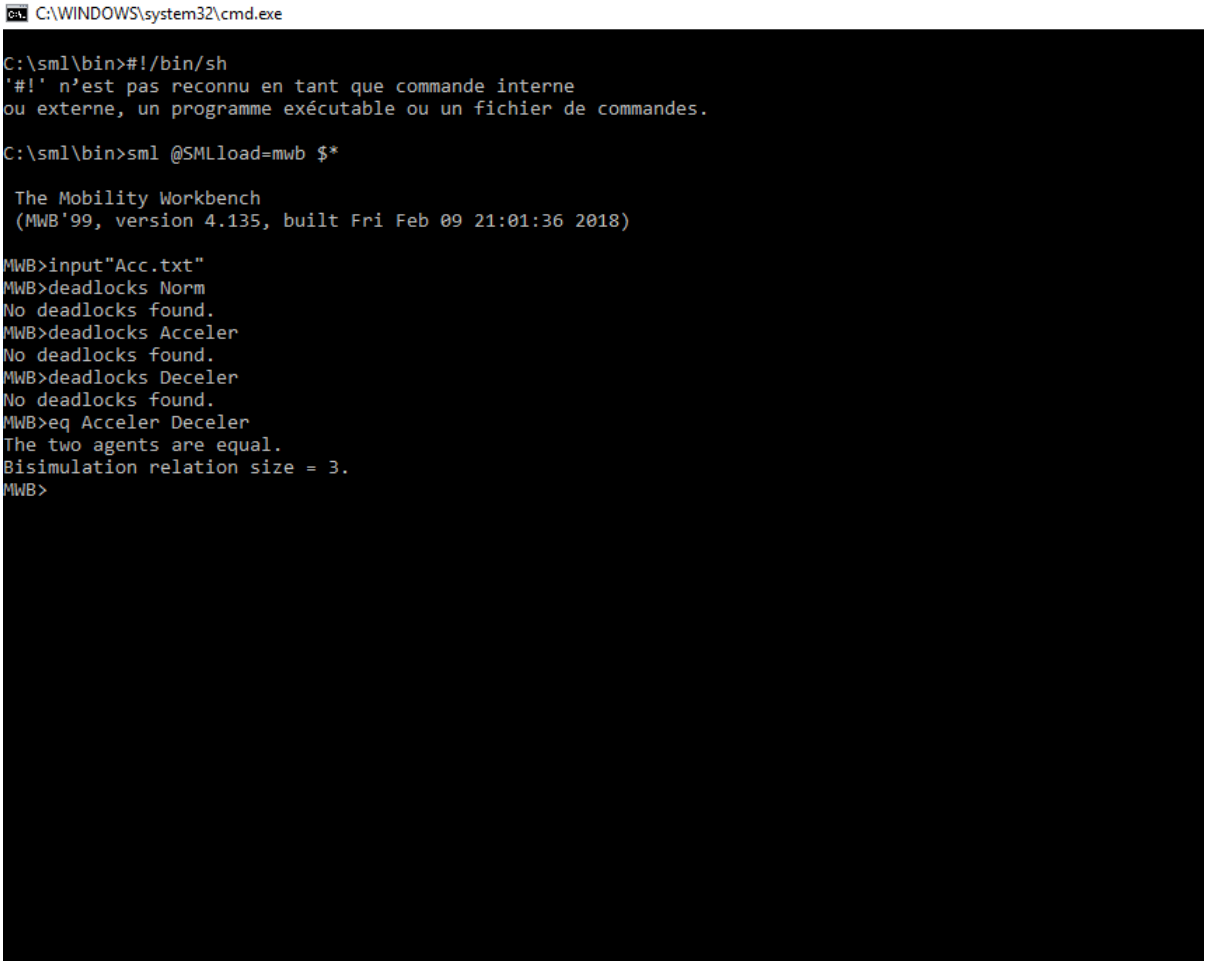


FIGURE 3.17 – vérifier et tester les blocage(deadlock) d'agents ACC(Mwb)

3.5.3 Équivalence Agent(eq)

Nous utilisons la commande "eq" pour vérifier si deux agents sont équivalents ouverts forts (fortement bisimilaire) [12].les résultats d'opération sont montrés dans la figure[3.18] suivante :

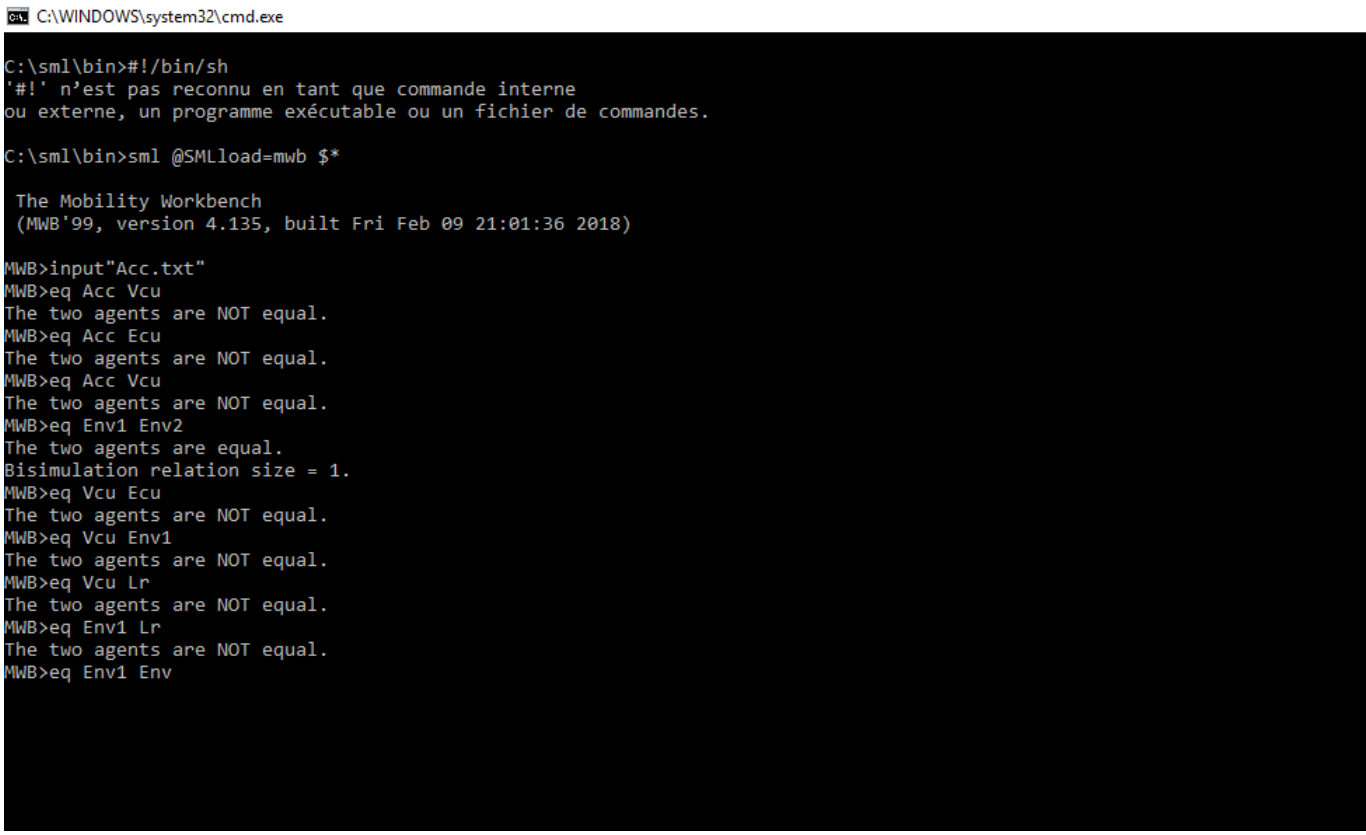


FIGURE 3.18 – vérifier l'équivalence d'agents ACC(Mwb)

3.5.4 Bisimulation ouvert faible Agent systeme Acc(weq)

vérifie si les agents de systeme ACC sont équivalent de bisimulation ouvert faible[9]. Les résultats de quelques exemples dans la figure[3.19] suivante :

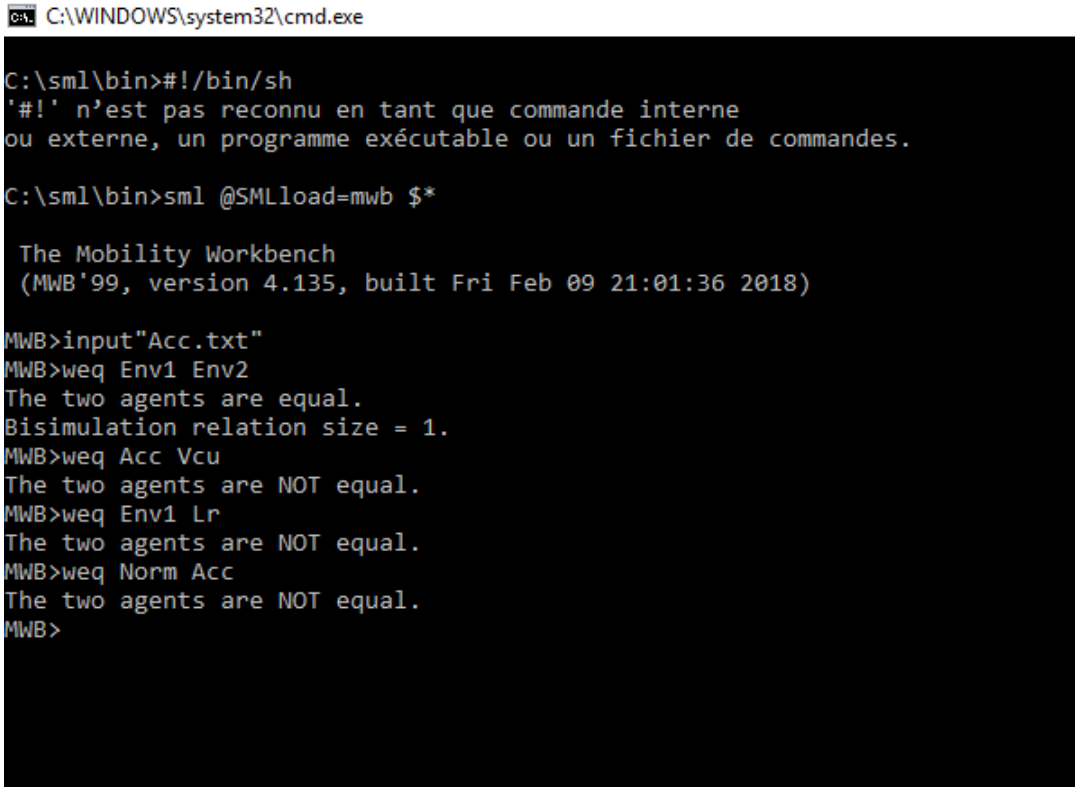


FIGURE 3.19 – vérifier l'équivalence de bisimulation faible de systeme ACC(Mwb)

3.5.5 Taille agent ACC(size)

donne une faible mesure de la taille du graphique de l'agent ce n'est pas toujours minimal mais l'espace d'agent exploré par les commandes de vérification d'équivalence est probablement plus grand[12].Les résultats de quelques exemples dans la figure[3.20] suivante :

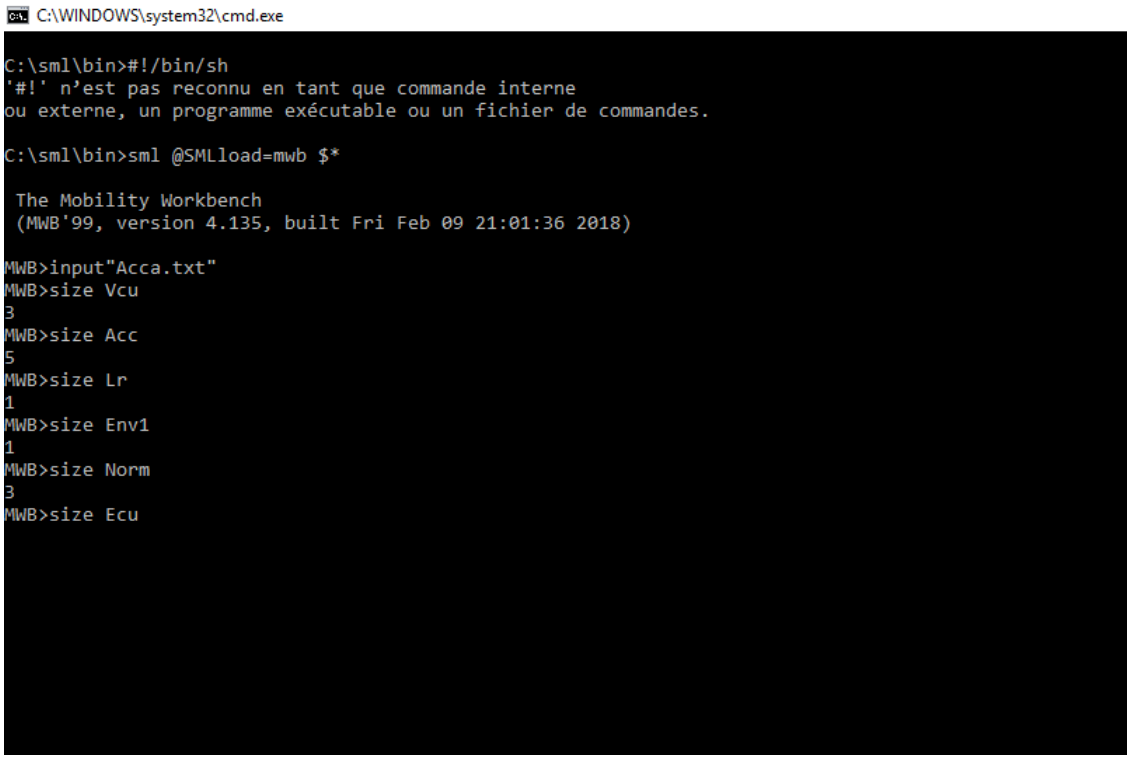


FIGURE 3.20 – la taille de quelqu'agent de systeme ACC(Mwb)

3.5.6 Propriétés

Nous avons également testé un ensemble de propriétés spécifiques MWB. La syntaxe de leur utilisation [40] est :
prouver la propriété agent_nom, écrite dans l'invite de commande MWB ou par une entrée du fichier, leur code et les explications connexes sont énumérées ci-dessous.
Les propriétés ont été enregistrées dans des fichiers (propriété.txt) contenant la syntaxe ci-dessus, pour faciliter leur utilisation.

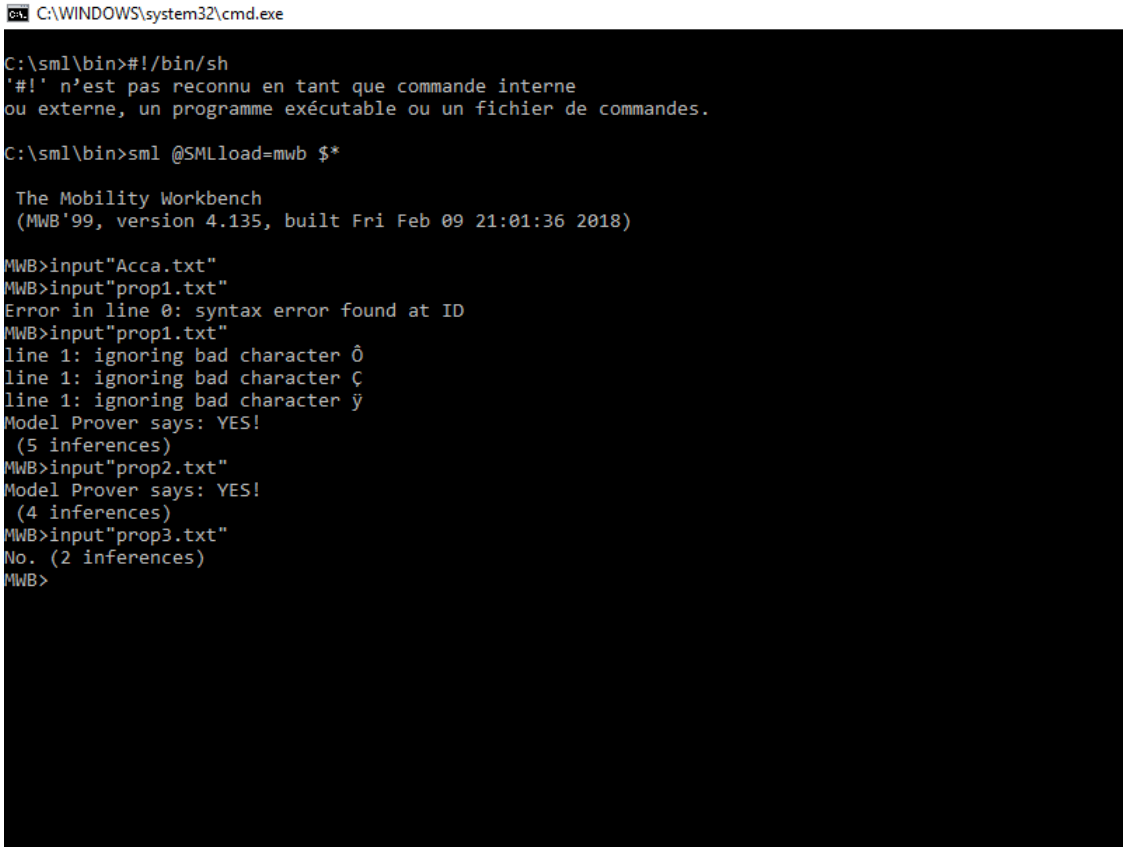


FIGURE 3.21 – propriétés de systeme ACC(MWB)

propriété(1) :

En bref, cette propriété indique que il ne peut y avoir aucune sortie ultérieure sans une entrée initiale, une situation valable pour tous les agents présents dans le système ACC :

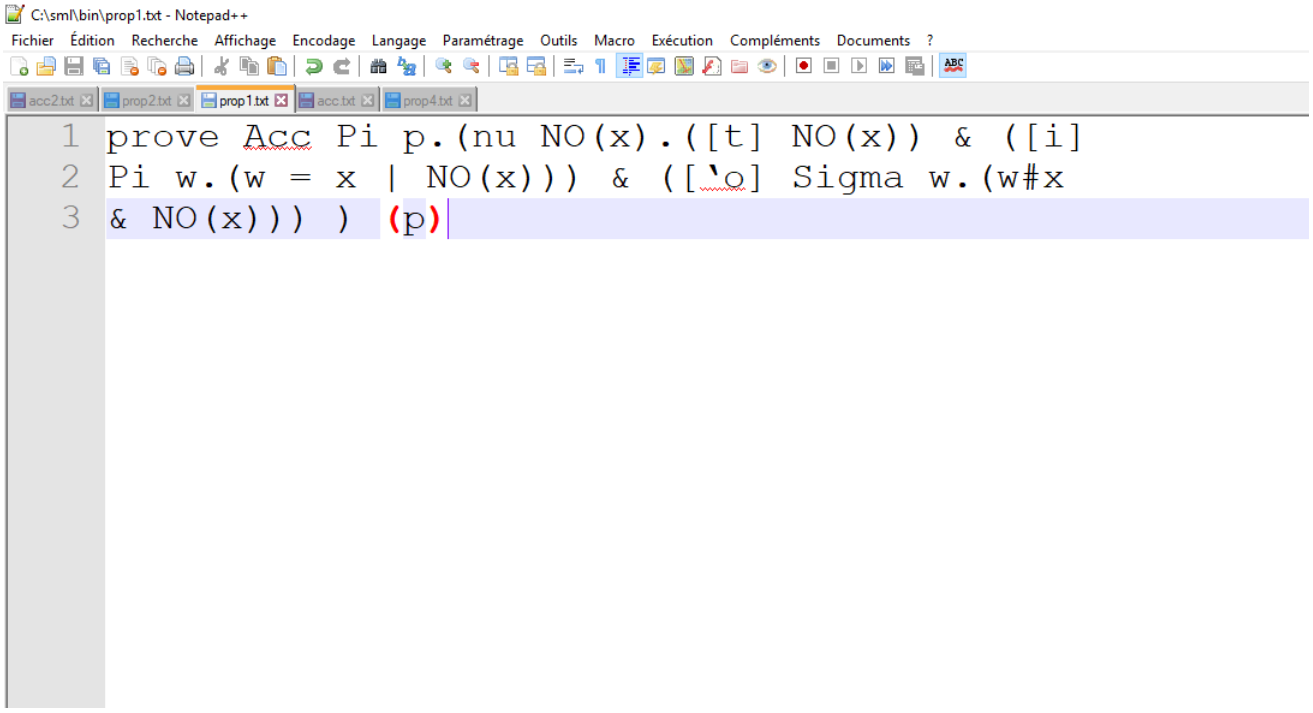


FIGURE 3.22 – propriété(1) de systeme ACC

propriété(2) :

Norm doit tenir après tout transitions, sauf pour une entrée co quand Accel (ou Decel) détient :

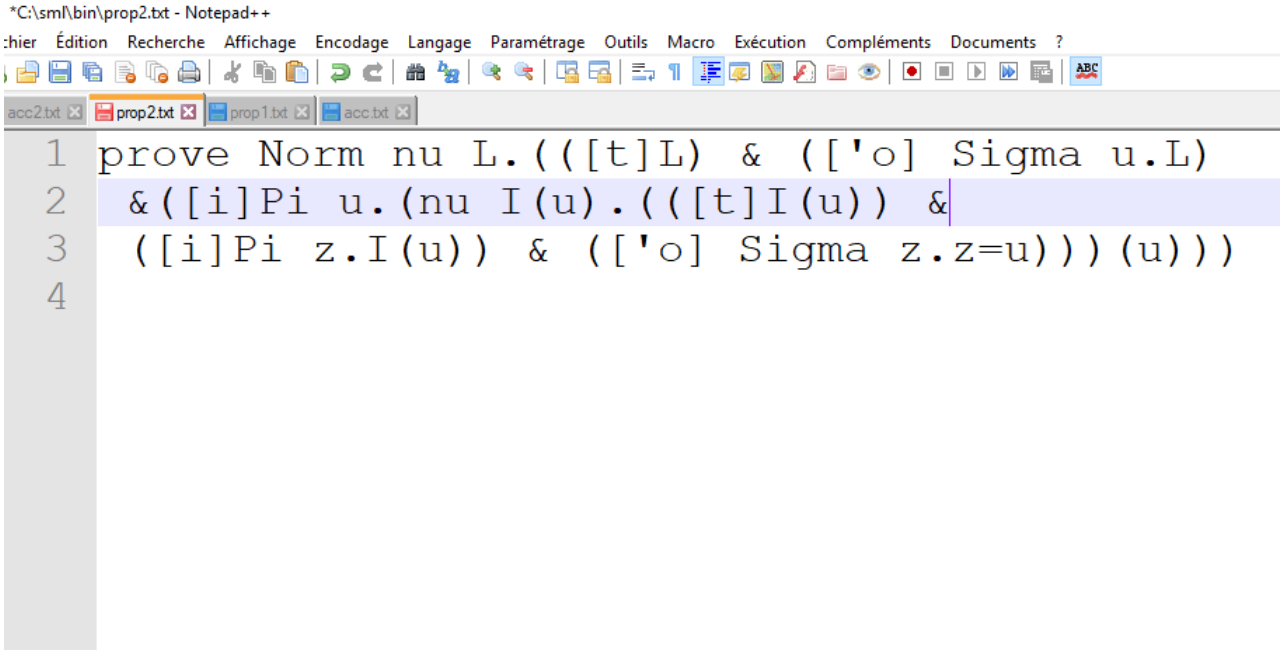


FIGURE 3.23 – propriété(2) de systeme ACC

propriété(3) :

La formule dit qu'il est garanti que les actions silencieuses d'agent Décel sont effectuées infiniment [9] souvent. Il est exempt de blocages.

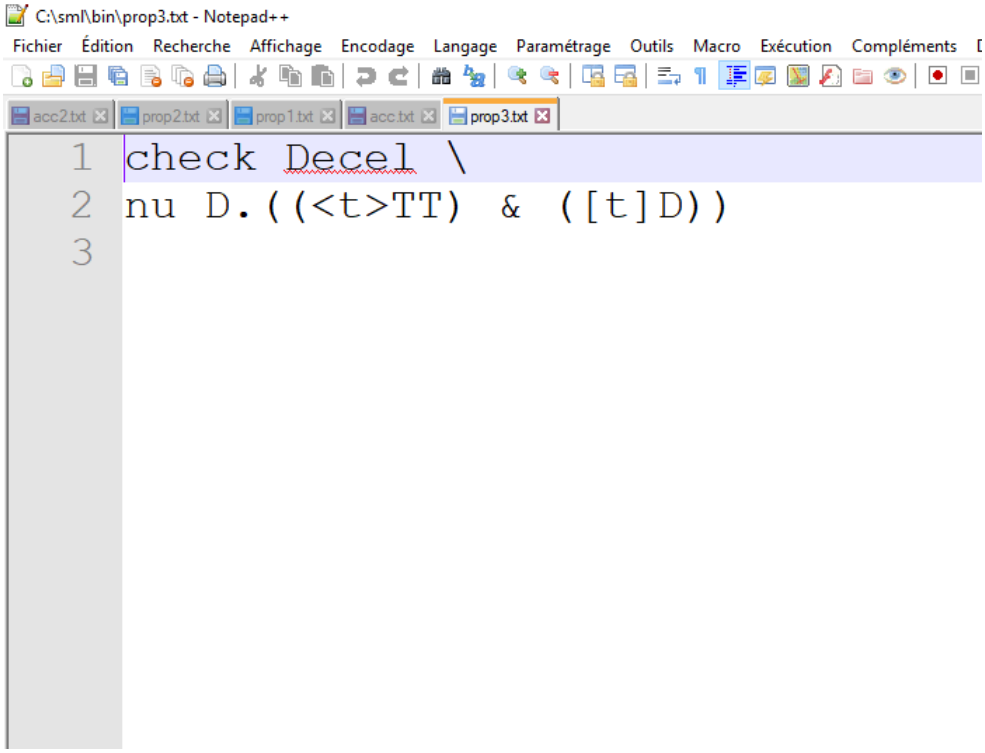


FIGURE 3.24 – propriété(3) de systeme ACC

3.6 Conclusion

La technologie ACC est une composante clé de toute future générations de voitures intelligentes.
Ce chapitre à présenté le régulateur de vitesse adaptatif en utilisant une approche formelle. Il explique également ce que c'est et comment cela fonctionne. Le but de ce travail est la spécification et vérification des composants et des actions avec l'utilisation d'une algèbre de processus appelée π-calcul qui supporte la mobilitée et son éxtension .
Nous avons étudié en détaille le fonctionnement et la dynamique de système. De plus,nous nous somme basé sur les travaux présentés dans (G. Ciobanu et.RUSU)[37].qui fournissent des spécifications incomplètes de système.Depuis la vérification formelle du système décrit en utilisan le model_cheking mobilité workbench(MWB).Il eSt utilisé pour importer la description des agents(processu),l'application des instructions pour la vérification automatique .Il permet aussi de prouver les propriétés désirées.

Conclusion générale

Avec la naissance des premiers systèmes distribués, le développement des réseaux informatiques sans fil, et la diffusion de dispositifs portables, on a vécu l'émergence de l'informatique mobile [20].Notre projet a permis de présenter la modélisation et la spécification des systèmes de régulateur de vitesse adaptive(Acc) par un langage formel qui est le π-calcul. Ce dernier a été défini pour analyser et décrire les systèmes mobiles [9].

L'utilisation d'un langage formel pour la modélisation de systeme ACC offerte des moyens rigoureux pour étudier et comparer les agents de systèmes et permet également la vérification de certaines propriétés du système modélisé.On utilise l'outil mobility workbench MWB.

Pendant la réalisation du projet, nous avons acquis des connaissances sur :

- Les systèmes mobiles et l'informatique mobile.
- Les formalismes les plus exploités pour la spécification et la vérification des systèmes mobiles.
- Le système ACC et leur spécification en utilisant le π-calcul et les diagramme de séquences.
- Certaines propriétés, décrivent la dynamique de système ACC.
- vérificateur du modèle(model-checking) appelé Mobilité Workbench qui est un outil automatique de vérification pour le π-calcul a vérifier l'exactitude de certaines propriétés telles que le blocage(deadlock), livelock, incohérences ... etc[8].

Nous nous somme basé sur les travaux présentés dans (G. Ciobanu et.RUSU)[37].Qui fournissent des spécifications incomplètes.Nous avons complété les spécifications système(ACC) en utilisant l'algèbre de processus π-calcul basée sur la mobilité de lien.Nous avons complété les spécifications puis nous avons expérimenté la vérification de certaine propriétés.

Dans nos recherches futures, nous entendons nous concentrer sur d'autres questions qui restent à résoudre, dont certains sont :

- Nous avons l'intention d'ajouter des tests en utilisant l'algèbre Processus π-calcul stochastique . La caractéristique particulière de ce type est l'utilisation des taux associés aux actions[17]. Chaque action a un délai stochastique qui décide quand il a lieu.
- Afin travailler avec l'algèbre Processus d'évaluation des performances, nous pouvons utiliser le proverif Outil (développé à l'Université d'Edimbourg).
- La spécification avec des des autres extensions de π-calcul en [17].

Bibliographie

[1] G. Gardarin O.Gardarin, rapport architecture Client-Serveur. Eyrolles, mars 1993.

[2] Russell Beale and Andrew Wood, Agent-Based Interaction. HCI pp.239-245,1994.

[3] J. Ferber, Les systèmes multi-agents Vers une intelligence collective, Masson, 1995.

[4] DEHIMI NARDJESS TISSILIA Un Cadre Formel pour La Modélisation et L'analyse Des Agents Mobiles,2011.

[5] Mâamoun Bernichi,Surveillance logicielle à base d'une communauté d'agents mobiles 4 May 2010.

[6] Antonio Carzaniga,Gian Pietro Picco,Giovanni Designing Distributed Applications with Mobile Code Paradigms 1999.

[7] Salah El Falou,Programmation répartie, optimisation par agent mobile 2007.

[8] Robert Abo, Approches formelles pour l'analyse de la performabilité des systèmes communicants mobiles : Applications aux réseaux de capteurs sans fil 2013 .

[9] Muhammad Fadlisyah,Using the π-Calculus for Modeling and Verifying Processes on Web Services 2004 .

[10] Gordan Jezic and Ignac Lovrek,USING π-CALCULUS FOR SPECIFICATION OF MOBILE AGENT COMMUNICATION University of Zagreb Faculty of Electrical Engineering and Computing 2014.

[11] Mikael BOURHIS Algèbres de processus et modélisation de phénomènes biologiques 2005.

[12] Victor, B., Moller, F. The Mobility Workbench - a tool for the pi-calculus. Lecture Notes in Computer Science vol.818, Springer, 428-440, 1994.

[13] R. Milner. Communication and Concurrency. Prentice Hall, 1989.

[14] R. Milner. The polyadic pi-calculus : a tutorial. Rapport technique, Logic and Algebra of Specification,University Press, 1991.

[15] R. Milner. Communication and Mobile Systems : the pi-Calculus. Cambridge University Press, 1999.

[16] R. Milner, J. Parrow et D. Walker. A calculus of mobile processes, parts i and ii.Journal of Information and Computation, 100 :1–77, 1992.

[17] BJORN victor Faron moller,A tool for the pi calculus in Computer Science sweeden vol.818, 1996.

[18] BJORN VICTOR,guide mwb département informatique Upsala sweeden 1994.

[19] S. Misljencevic. Jini Service Container. Thèse de doctorat, Universiteit Antwerpen,2006.

[20] Dr.Laid Kahloul Support de Cours Module : Sémantique des Systèmes Mobiles,Option Génie Logiciel et Systèmes Distribués (GLSD) 2014.

[21] ISHAK BOUSHAKI,Modélisation et simulation des processus biologiques dans le π-calcul, 2006/2007.

[22] Flemming Nielson : The typed lambda-calculus with first-class processes. In Eddy Odijk, Martin Rem et Jean-Claude Syre, éditeurs : PARLE (2), volume 366 de Lecture Notes in Computer Science, pages 357373. Springer, 1989.

[23] http ://wiki.fot-net.eu/index.php/ACC(adaptive Cruise_Control).

[24] https ://www.eetimes.com/document.aspdoc_id=1137811.

[25] MASIF, The OMG Mobile Agent System Interoperability Facility,1997, URL www.masif.org.

[26] Foundation for Intelligent Physical Agents,1998, URL,(www.fipa.org).

[27] chaoui.alaoua Un Cadre Formel pour La Modélisation et L'analyse Des Agents Mobiles université constanatine 2 2014.

[28] J. Gomoluch and M. Schroeder, Information agents on the move : A survey on loadbalancing with mobile agents, 2001.

[29] Alfonso Fuguetta and Gian petro,understanding Code mobility.IEEE softwr engineering,1998.

[30] Fredrick B.Beste. The Model Prover : A Sequent Calculus Based Modal -Calculus Tool for Finite Control -Calculus Agents. ftp ://ftp.docs.uu.se/pub/mwb/x4.ps.gz, January 1998.

[31] Cruise Control International Journal of Scientific and Engineering Research Volume 3, Issue 8, August-2012.

[32] Cruise Control System Overview 5th Meeting of the U.S. Software System Safety Working Group April 12th-14th 2005 @ Anaheim, California USA.

[33] Design of vehicle control unit based on DSP for a parallel HEV.

[34] Carl Hewitt et Henry G. Baker : Laws for communicating parallel processes. In IFIP Congress, pages 987992, 1977.

[35] Davide Sangiorgi and David Walker. The π-Calculus : a Theory of Mobile Processes.Cambridge University Press, 2001.

[36] TACOMA - Operating system support for agents.www.tacoma.cs.uit.no/

[37] G. Ciobanu and S¸. Rusu raport Verifying adaptive Cruise Control by p-calculus and Mobility Workbench,Roanian academy,2008.

[38] Alberto Montresor and Hein Meling and Ozalp Babaoglu, Messor : Load-Balancing through a Swarm of Autonomous Agents. AP2PC pp.125–137, 2002.

[39] Cédric LHOUSSAINE,Réceptivité, Mobilité et p-Calcul,L'UNIVERSITÉ D'AIX-MARSEILLE .au sein du Laboratoire d'Informatique Fondamentale de Marseille 2002.

[40] Mads Dam Model Checking Mobile Processes (1996).

[41] JADE Java Agent Development Framework. ¡URL : http ://www.jade.cselt.it/¿.

[42] Gérard Boudol,Towards a lambda-calculus for concurrent and communicating systems. In Proceedings of the International Joint Conference on Theory and Practice of Software Development, Volume 1 : Advanced Seminar on Foundations of Innovative Software Development I and Colloquium on Trees in Algebra and Programming, TAPSOFT '89/CAAP '89, pages 149161, London, UK, 1989. Springer-Verlag.

[43] J.C.M. Baeten. A Brief History of Process Algebra.Report CSR, 04-02. Eindhoven, Netherlands : Vakgroep Informatica, Technische Universiteit Eindhoven, 2004 .

[44] Toufik Messaoud MAAROUK,Spécification formelle des systèmes mobiles temps réel.Université de Mentouri, 25000 Constantine 2012.

[45] M. Breugst and I. Busse and S. Covaci and T.Magedanz, Grasshopper – A Mobile Agent Platform for IN Based Service Environments. Proceedings of IEEE IN Workshop, pp.279–290, 1998.

[46] C. A. R. Hoare : Communicating Sequential Processes (Prentice Hall International Series in Computing Science). 1985.