
بسم الله الرحمن الرحيم

ملخص :

ان الانظمة الحساسة للسياق تتطلب دراسة معمقة، وهذا لتحديد طرائق استنباط وصياغة المعلومات السياقية ، كذلك فان تطوير هذه الانظمة يتطلب الاستعانة بالتكنولوجيات الحديثة ، والوسائط المعقدة للوصول الى الحلول المثالية.

تبعا لهذا المنظور فان انظمك العملاء المحيطية بخواصها التفاعلية، والاستنتاجية والاستقلالية، بالإضافة الي التكافل بين عناصرها، يجعل منها الخيار الامثل لصياغة الانظمة الحساسة للسياق .

هذا العمل، يستعرض التحديات المستخلصة خلال عملية تصميم، ونمذجة نظام متعدد العملاء هدا النظ يتجسد في تطبيق للهواتف الذكية ، يتكون من عدة عملاء تتكامل عناصرها فيما بينها، و كذا تتفاعل مع محيط لاستغلال خصائص الجهاز، لتقديم خدمة ادراكية للمستخدم ، كذلك فان هذا العمل يعرض بناء متعدد المستويات ونمذجة بلغة AUML .

إن الاخذ بهذه المعطيات، قادنا الي تقديم تصميم تطبيق "خشوعي" مستعينين بتكنولوجيا

JADE-LEAP و ANDRIOD-STUDIO .

الكلمات المفتاحية: سياق ,أنظمة حساسة للسياق، عميل محيط، نظم متعددة العملاء، نمذجة AUML.

Abstract

The context sensitive systems require consideration of context to make a response adapted to the users, for there, a representation of contextual information is required. The development of such systems requires the integration of new technologies dedicated to complex environments to ensure the best solutions.

In this sense, ambient multi-agent systems offer special features, which make them suitable for the development of ambient systems sensitive to context, such as reactivity, reasoning, autonomy and social skills.

Which aims to use this technique for taking into account the context in nomadic environments sensitive to the context, first, it was suggested in this paper, an approach based on environmental agents dedicated to smart-phones, which consists of several environmental agents. Its environmental agents, interact with their environment using devices capacity to reach a goal to offer assistance to the user. It was also used to model the architecture AUML layers of our approach. In the end, we have implemented a prototype arrived in our approach called "KHOCHO3I" using android studio and JADE-LEAP technology.

KEYWORDS*:*

Context, context awareness, environmental agents, AUML, multi agent system, environmental system.

RESUME

Les systèmes sensibles au contexte exigent la prise en compte du contexte pour rendre une réponse adapté aux utilisateurs, pour ce là, une représentation de l'information contextuelle est requise. Le développement de ce type de systèmes a besoin de l'intégration des technologies nouvelles dédiés aux environnements complexes, pour garantir des meilleures solutions.

Dans ce sens, les systèmes multi-agents ambiants offrent des caractéristiques spéciales, qui les rendent appropriées pour le développement des systèmes ambiants sensible au contexte, telles que la réactivité, le raisonnement, l'autonomie et les aptitudes sociales.

Dont l'objectif d'utiliser cette technique pour la prise en compte du contexte dans les environnements nomade sensible au contexte, tout d'abord, on a proposé dans ce mémoire, une approche à base d'agents ambiants dédiée aux smart-phones, qui se compose de plusieurs agents ambiants. Ses agents ambiants, agissent avec leurs environnement en utilisent les capacités des dispositifs pour arriver à offrir une aide objectif à l'utilisateur. On a aussi, utilisé AUML pour modéliser l'architecture de couches de notre approche. En fin, on a arrivé à implémenté un prototype de notre approche qui s'appelle «KHOCHO3I », en utilisant les technologies android studio et JADE-LEAP.

LES MOTS CLES :

Contexte, sensibilité au contexte, agents ambiants, AUML, système multi agent, système ambiant.

DÉDICACE

A nos pères et nos mères

En témoignage de nos immenses affections et nos grandes gratitudes

Ni nos dédicaces, ni nos remerciements ne pourront égaler vos sacrifices

Votre amour et sollicitude que vous avez toujours nous réservés

Que Dieu vous prête longue vie et bonne santé et qu´il garde notre solidarité et

notre indéfectible attachement familial.

Nous dédions ce modeste travail à :

Nos frères et nos sœurs, que nous aimons beaucoup.

Notre grande famille.

Nos cher ami (e) s, et enseignants.

Tout qu'ont collaboré de près ou de loin à l'élaboration de ce travail.

Ghassem sidi aly /Benmedakhene Slimane

REMERCIEMENT

On tient, au terme de ce travail, à présenter nos vifs remerciement à tous les personnes qui ont contribué, de près ou loin, à son bon déroulement.

On tient à présenter tous nos respects et notre gratitude aux personnes pour leurs suivis et encouragement tout au long de ce travail.

Notre gratitude s'adresse également à Mm S. Bourogaa pour son encadrement et pour l'aide qu'il nous prodigué durant ce rapport, ainsi qu'a tous notre formateurs.

On remercie également les membres des jurys qui ont accepté d'évaluer notre travail.

Ghassem sidi aly /Benmedakhene Slimane

INTRODUCTION GENERALE

INTRODUCTION GENERALE

Au début, l'utilisation de l'Informatique prenait une forme standard. Un utilisateur d'une application est associé à un ordinateur situé dans un bureau. L'utilisateur est déterminé par des caractéristiques communes avec tous les autres utilisateurs de la même application. En général, il est défini par un processeur supportant un grand nombre de traitements, une connexion filaire et une alimentation continue. Ces caractéristiques ont caractérisé l'environnement de travail comme étant un environnement stable.

Avec l'avènement des technologies mobiles, on parle de l'Informatique ubiquitaire introduite initialement par M. Weiser 1997. L'Informatique ubiquitaire rend l'utilisation de l'Informatique possible en tout moment et indépendamment de positionnement de l'utilisateur.

Actuellement, il existe dans la recherche en informatique une idée importante pour l'utilisateur : l'intégration de différentes applications techniques dans nos environnements habituels comme la maison, le lieu de travail, les centres de loisir, la ville, etc. Dans un futur proche, ces environnements connaîtront nos goûts, nos préférences, nous aideront à trouver nos objectifs et satisfaire nos besoins. Les personnes interagiront avec un environnement qui sera conscient de leur présence et du contexte général et qui pourra s'adapter et répondre aux nécessités, coutumes et émotions de l'utilisateur. Ces environnements sont définis comme les systèmes sensibles au contexte

Il y a aucun doute que les systèmes sensibles au contexte sont un domaine de recherche très actif dans l'intelligence ambiante dans laquelle les systèmes multi agents ambiants jouent un rôle essentiel car Ils offrent des propriétés fonctionnelles et non fonctionnelles telles que l'autonomie, le raisonnement réactif et proactif. Ainsi qu'ils peuvent être vus en tant que agents autonomes qui extraient, traitent, changent et partagent l'information disponible de contexte [01].

Les systèmes multi agent ambiant sensible au contexte est composé de plusieurs agents ambiants qui utilisent les informations contextuelles collecter à partir des diverses ressources pour interagir avec leurs comportements, l'utilisation des agents ambiants lui donne l'avantage d'utiliser leur mécanisme d'auto-organisation et leurs capacités pour résoudre certains problèmes qui sont apparus plus difficile à résoudrai par des autres conceptions.

Pour atteindre ces buts il faut rependre les questions :

Comment utiliser les agents ambiants dans les systèmes sensibles au contexte et quel sont les avantages gagner dans ce cas ?

A cause que le contexte du système est changeable avec le temps, il y a le risque de perdre certaines ressources.

- **Comment rendre ce système sensible au contexte d'avoir la possibilité de remplacer ces ressources et d'être capable de prendre la décision à partir les ressources disponibles ?**

Pour cela on va faire une conception d'un système multi agent ambiant pour les environnements sensibles au contexte. Le travail va consister un rapport qui s'organise selon les parties suivantes :

ORGANISATION DU MEMOIRE :

- Un état de l'art qui va introduire le contexte, la sensibilité au contexte, les agents ambiants, et fini par citer les déférents travaux réalisés à ce Domain et les discutera.

- Une contribution qui va contenir la conception d'une application multi agents ambiant sensible au contexte qu'il doit :

  - Réaliser son traitement d'une façon transparente, afin de gêne le moins possible l'utilisateur, et de fonctionner en dépit de l'apparition et de la disparition de certaines sources, notamment à cause de la mobilité de certains de ces dispositifs ou manque de ressources.

Cette étape va contenir une conception du système à partir des scenarios et des schémas, une modélisation par le langage AUML et se termine avec une implémentation présentée par la plate-forme JADE-LEAP et l'outil de développement ANDRIOD STUDIO.

CHAPITRE I

LA SENSIBILITE AU CONTEXTE

I. LE CONTEXTE

Avant de construire un système sensible au contexte, il faut d'abord passer par une définition

1. Définition

Ils existent plusieurs études et recherches visant à donner une définition claire et précise à la notion contexte. Nous présentons ici une liste non exhaustive des définitions disponibles dans la littérature, ordonnées selon leur ordre chronologique [02].

- [03], définissent le contexte comme les informations répondant aux questions « Where are you (où vous êtes) », « whoyou are with (qui sont avec vous) » et « what ressources are nearby(qui les ressources sont proches de vous) ». Ils considèrent que le contexte n'est pas limité à la localisation de l'humaine, et puis incluent la luminosité, le niveau de bruit, l'architecture du réseau, le coût et la bande passante de communication, et même la situation sociale de l'utilisateur.

- [04], proposé une définition plus précise du contexte. Il commence par le définir en tant qu'étant la totalité de toutes les entités qui influencent le comportement cognitif de l'humain (ou du système) dans une occasion particulière.

- [05], considérer le contexte comme un concept particulier défini par l'entité qui le perçoit. Pour une entité, le contexte pourrait être sa localisation spatiale, alors que pour des autres, il sera au sujet de sa localisation temporelle, ou de son état émotionnel.

- [06], apportent une définition plus générale du contexte, qui est la définition la plus utilisée par les concepteurs des systèmes sensibles au contexte. Ils considèrent qu'il est l'ensemble de toutes les informations qui peuvent être utilisée pour caractériser la situation d'une entité.

- Pour [07], le contexte est l'ensemble des paramètres externes à l'application qui peut influencer son comportement en définissant de nouvelles vues sur ses données et ses services. Ils ajoutent que l'ensemble des paramètres du contexte peut être couverte par cinq axes

: le mode de communication, l'utilisateur, le terminal, la localisation et l'environnement. Un changement de valeur sur l'un de ces axes définit alors une nouvelle situation contextuelle à laquelle l'application doit s'adapter. La **figure1.1** montre une représentation de deux situations contextuelles différentes pour la même action (ici, pour signaler le dossier médical d'un patient) par trois de ces axes. Le contexte n'est en effet pas le même si l'utilisateur est une infirmière qui regarde le dossier en dehors du cabinet sur un PDA, ou s'il est le médecin qui le regarde sur le PC du cabinet.

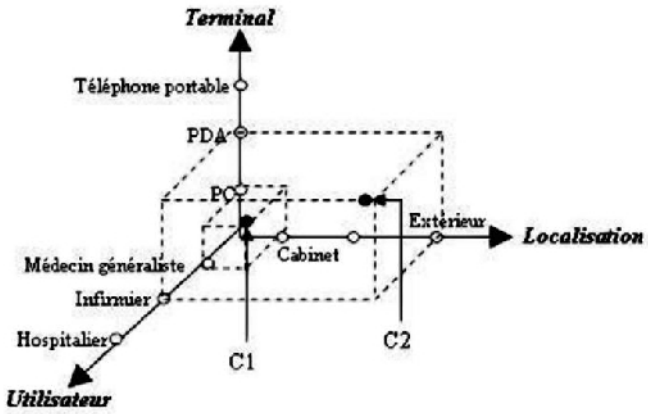


Figure1.1 : Représentation multiaxiale du contexte [07]

- [08], préserver l'idée du contexte liée à une entité suggérée près [Pascoe98], en le définissant comme l'ensemble des connaissances mesurées (les faits tels qu'ils sont perçus) ou conclus (résultant d'une formation et d'un raisonnement informatique appliqués aux contextes passés et présent) qui décrivent l'état et l'environnement dans lequel une entité existe ou a existé.

- [09] définit le contexte comme ce qui contraint le focus d'un acteur sans y intervenir explicitement. Il complète cette définition en précisant que "parler de contexte n'a de sens que relativement au focus" d'un acteur, que "comme le focus évolue, le contexte a une dynamique" et que "le contexte est ancré dans un domaine".

Discussion

Déférents aspects sortent de ces définitions. Comme [10] est soulignée, le contexte ne se rapporte pas seulement à la localisation de l'utilisateur, où même celle des différents éléments

composant l'environnement ambiant (les composants micros et macro du système), mais un plus large panneau d'information. La sensibilité à la localisation est seulement un sous-problème plus limité de la sensibilité au contexte.

Il est possible de lister les divers types d'information contextuelle, comme le font plusieurs des définitions indiquées précédemment. Le contexte peut alors être décrit comme l'information au sujet de la localisation, de l'heure, du profil d'utilisateur, de la température, etc. l'énumération explicite de tous les types de données contextuelles donne une solution limitée, parce que cela implique le risque d'exclure certaines données contextuelles puisqu'elles n'écrivent pas une des catégories établies, ou cela elles apparaîtraient plus tard à la conception de systèmes. D'ailleurs, comme le rapport près [11], le contexte n'est pas limité à une liste de propriétés, mais peut également se composer des associations entre les déférents informations qui, prise séparément, ne sont pas incluses dans les données contextuelles. Pour cette raison, une définition correcte du contexte doit être assez générale pour inclure n'importe quel type de données, voilà ce qui était apparent dans la définition de [06].

La notion de la particularité proposée notamment par [05] et [08] ils semblent tout aussi appropriés, parce que dans un système aussi vaste qu'un système ambiant, il est impossible de caractériser le contexte de la totalité du système. Il nous inspire que le contexte doit être défini selon le point de vue local de chaque entité composant le système.

Finalement, le lien défini par [09] entre le focus d'une entité, en d'autres termes la tâche qu'elle doit réaliser et le contexte de cette entité, nous semble également essentiel.

Pour ces raisons, la définition que nous choisissons et qui couvre tous les axes possibles du contexte, est la définition de dey qui est la plus adapté.

2. Les caractéristiques du contexte

Le contexte est fortuit :

L'homme perçoit la réalité non pas comme elle existe en tant que telle mais à partir des propres informations qui relèvent d'une présentation cognitive (modèle), le contexte est alors considéré comme "l'état d'esprit" de cette présentation interne, Le contexte d'une entité (humain

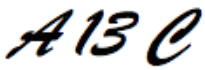
ou système) est alors constitué des informations issues de la représentation interne que l'entité se fait de son environnement plutôt que de l'état réel de cet environnement [04].

Le contexte est changeable avec le temps, dynamique :

L'information de contexte change continuellement de valeur avec le temps. Par exemple, le déplacement de l'utilisateur implique que le contexte localisation de l'utilisateur change de valeur (exemple : cordonnées GPS) [04].

Le contexte n'a pas de frontière clairement définie :

[04], a fait par un exemple simple portant sur la reconnaissance de caractères. Si nous considérons l'ensemble composé des caractères suivants :

Exemple 1 : 

La majorité des gens reconnaîtront les caractères A, B et C. Tandis que dans l'ensemble suivant :

Exemple 2 : 

Il est probable qu'ils reconnaissent bien les caractères 9, 13 et 7. Les mêmes symboles 13 auront donc été reconnus de façon différente selon leur contexte. Dans un contexte encore plus large l'interprétation va être différemment. Il est donc difficile d'établir les frontières entre les données qui influencent la tâche courante (les informations contextuelles), ici la tâche de reconnaissance de caractères, et celles qui ne le font pas.

L'information de contexte est hétérogène :

Cette caractéristique est issue vue que les informations du contexte capturées à partir d'une variété de sources [12]. Notamment le contexte peut être : une information attirée directement à partir des capteurs physique ou récupérés à partir de composants logiciels, une

donnée par l'utilisateur ou bien prévenu en synthétisant plusieurs sources ou une interprétation d'une seule ressource

L'information de contexte est imparfaite :

Cette caractéristique de contexte est déterminée selon la source de cette information : le contexte sera ambigu si des ressources séparées fournissent une même information avec des niveaux de granularités différents, le contexte peut être imprécis ou bien erroné ou même inconnu [12].

L'information de contexte est interdépendante :

L'information contextuelle peut être dépendante d'une autre, le changement de valeur d'une information de contexte peut influencer sur une autre valeur de contexte [12].

3. Catégorisation de contexte

Pour faciliter La collection et la présentation des informations contextuelles dans un système sensible au contexte il faut faire une catégorisation, plusieurs travaux ont proposé une catégorisation de contexte.

- [13] classifie le contexte en deux catégories, un contexte physique de l'utilisateur et un contexte organisationnel. Le contexte physique couvre les paramètres de l'utilisateur tels que : la localisation, les caractéristiques du matériel et de l'application. Le contexte organisationnel représente les informations relatives aux processus de collaboration dans lequel l'utilisateur est impliqué tels que le les fonctions et le calendrier.

- Selon [14], le contexte est catégorisé en trois axes, un contexte physique, un contexte utilisateur et un contexte social. Le contexte physique est présenté par les caractéristiques des dispositifs physiques tels que la bande passante. Le contexte utilisateur réunis les préférences de l'utilisateur. Le contexte social apparus en cas des interactions entre les utilisateurs existants et les nouveaux arrivants.

La combinaison entre les deux catégorisations offre que les informations de contexte peuvent être classifiées selon le contexte utilisateur, le contexte spatio-temporel, le contexte physique et le contexte organisationnel [15] :

- Contexte Utilisateur : décrit les préférences de l'utilisateur telles que la langue, la résolution de l'écran …etc. Ces informations sont généralement imposées par l'utilisateur lui-même.

- Contexte Physique : inclut les paramètres physiques de l'environnement de travail. Cette catégorie décrit les caractéristiques du terminal (mémoire disponible, vitesse du processeur, etc.) et du réseau de connexion (bande passante, mode de connexion, canal de connexion.). D'autres paramètres physiques peuvent aussi être inclut tels que la température.

- Contexte Organisationnel : se compose des informations relatives à la collaboration dans laquelle un utilisateur est engagé tels que le groupe auquel il appartient, le rôle, l'activité en cours, les ressources partagées entre les membres du groupe, les autorisations possibles, etc.

II. LA SENSIBILITE AU CONTEXTE

1. Définition

Apres l'étude des différentes définitions de la notion contexte et choisir l'un de les plus utiliser par les chercheurs, nous sommes en train de faire un spot sur les déférents définitions proposée pour décrire la notion de "sensibilité au contexte"

- A travers de la conception d'un système "active map service" [16] a présente la première introduction de la sensibilité au contexte, ce premier système propose des informations a un utilisateur à partir de sa localisation. Alors ils définissent la sensibilité au contexte comme la capacité des applications d'un utilisateur mobile à découvrir et réagir au changement survenant dans l'environnement où ils sont situés. Ensuite [03] devisaient les applications sensibles au contexte en 4 catégories selon la manière de réaliser la tache manuellement ou automatiquement et que cette tâche consiste uniquement à obtenir des informations ou à réaliser des actions (**voir tableau 1.1).**

	Manuelle	automatique
Information	Sélection à proximité	Reconfiguration contextuelle automatique
commande	Commande contextuelle	Actions contextuellement déclenchées

Tableau 1.1 : *catégorisation d'application sensible au contexte* [03]

- [17] la capacité d'un système à être diminue sensible au contexte dans quatre niveaux :

➢ La perception contextuelle qui donne la possibilité a un système de capturer les informations du contexte afin de pouvoir présenter à l'utilisateur. Le système peut alors percevoir son environnement

➢ L'adaptation contextuelle qui rend également possibilité au système de modifier son comportement et intégrer pour être transparente dans l'environnement de l'utilisateur.

➢ La découverte des ressources contextuelles pour avoir une vision complète sur son contexte le système découvrir et utiliser des ressources supplémentaires d'autres entités. Le système est alors capable d'interagir avec son environnement

➢ L'augmentation contextuelle qui permet à partir des expérimentations antécédentes ou un contexte donné, d'enrichir l'environnement en associant des données. Le système peut alors enrichir son environnement

- [17], disant que la sensibilité au contexte décrit la capacité d'un système d'être perçu et acte sur les informations sur son environnement, tel que le temps, la température, la localisation ou l'identité de l'utilisateur.

- Selon [06], une application est sensible au contexte s'il peut distinguer son contexte et si elle fait guider un comportement dynamique par sa connaissance sur son environnement.

- Pour [18], la sensibilité au contexte c'est à dire toute donnée n'appartenant pas aux entrées ou aux sorties explicites d'une application. Une application sensible au contexte tient compte de données qui ne lui sont pas fournies explicitement ce sont des données qu'il perçoit ou récupère par lui-même contrairement aux applications classique ou les systèmes reçoit en entrée des données fournies explicitement par l'humain et il se base sur ces entrées pour produire des sorties. Alors que entries ou sorties d'un système sensibles au contexte influençant néanmoins son traitement appartient alors à son contexte

- [19] les conditions pour dire qu'un système est sensible au contexte s'il avoir la capacité d'extraire, interpréter et utiliser des informations de contexte et adapter sa fonctionnalité au contexte d'utilisation courant.

- [19], un système sensible au contexte et déférent qu'un système classique car il :

  - Adaptatif : le système avoir la capacité à apprendre les préférences de l'utilisateur et à s'ajuster en conséquence.

  - Réceptif : il anticipe les besoins de l'utilisateur dans un environnement évolutif

  - Proactif : il possède un objectif et est capable de prendre des initiatives, plutôt que de simplement réagir à l'environnement.

  - Autonome : il peut agir indépendamment, sans intervention humaine, le système utilise des expérimentations et données entrées pour prendre des décisions.

- Une définition proche de celle de [16] est proposée par [39], il établit que la sensibilité de contexte est autonome de détecter et réagir aux variables de l'environnement.

Selon ces définitions qui couvre les différents axes de la sensibilité au contexte il est possible d'attirer deux aspects principaux caractérisent la sensibilité au contexte, le premier est la capacité de percevoir son contexte et le second est la capacité de l'exploitation du contexte perçu, tous ces aspects apparus dans la définition de [19], « la capacité à percevoir, interpréter et

utiliser les différentes informations relatives au contexte courant pour adapter dynamiquement sa fonctionnalité »

III. ARCHITECTURES GENERALE D'UN SYSTEME SENSIBLE AU CONTEXTE

Il a y eu une différence entre les systèmes classiques et les systèmes sensible au contexte

Selon [06], la différence entre les deux catégories est au niveau des données explicites manipulées, les données des applications d'interaction classique peuvent être des variables internes de l'application où peut être des données explicites des utilisateurs

Mais les applications sensibles au contexte en plus des données internes et utilisateurs il traite aussi les informations de contexte

Il y a beaucoup de travaux ont proposé des architectures pour les systèmes sensibles au contexte [21] [22] [23] [24].

Un consensus dans ces travaux sur la séparation entre la capture des informations de contexte et utilisation de ces informations afin d'assurer l'extensibilité et la réutilisation du système.

La **figure1.2** présentée l'architecture générale d'un système sensible au contexte. Elle est composée des quatre couches : Capture de contexte, Interprétation de contexte, Gestion de contexte et Adaptation au contexte.

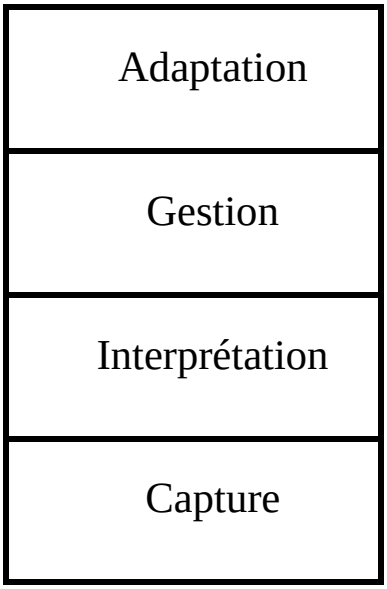


Figure1.2 : Architecture de système sensible au contexte [22]

Chacun de ces couches joue un rôle dans le système, dont on va en présenter les couches de l'extérieur vers l'intérieur avec le rôle de chacune.

1. Capteur de contexte

Les systèmes sensibles au contexte sont désignés pour être à l'écoute aux changements de l'environnement de système. La couche de capture est composée d'un ensemble de capteurs. Ces capteurs peuvent être classifiés en physiques ou logiques.

Afin d'assurer la capture des données de contexte le système doit transforme ces données pour être lisible par le système, la transformation est une sorte d'interprétation assure par la couche d'interprétation de contexte [22].

2. Interprétation de contexte

Le rôle de cette couche est d'interpréter les données contextuelles fournies par les capteurs. Elle fait l'analyse et la transformation des données brutes fournies par la couche du capteur, les opérations effectuées sur les données brutes concernent l'extraction, la quantification et l'agrégation.

Par exemple, les coordonnées GPS d'une personne peuvent être moins significatives qu'une adresse sous forme de numéro de rue et de ville. Cette couche peut aussi avoir le rôle de résolutions de conflits causés par l'utilisation de plusieurs sources de contexte. Ces sources peuvent donner des résultats contradictoires comme ils peuvent aboutir à des situations imprécises. Cette couche doit donc avoir des moyens pour résoudre ces conflits [21].

3. Gestion de contexte

A ce niveau, le contexte capturé et interprété doit être bien géré pour faciliter l'utilisation. La gestion de contexte contient le stockage et la représentation formelle des informations de contexte. Le stockage peut être centralisé ou distribué [22].

Après la capture des données contextuelles et les interpréter pour donner une information lisible par le système, la couche gestion de contexte ne fait pas juste la représentation formelle des informations contextuel mais il fait aussi le stockage des données

La représentation des informations de contexte est la structuration de ces informations se fait selon un modèle. Le choix du modèle dépend fortement de mécanismes choisis pour adapter les actions du système au contexte. Il existe Plusieurs méthodes de représentation proposées, les données contextuelles peuvent être présentées en :

- **Représentations des données contextuel à base de pair (*attribut, valeur*) :** c'est la structure des données la plus simple pour la modélisation des informations contextuelles. L'*attribut* représente un élément de contexte tel que la localisation de l'utilisateur. La *valeur* est la valeur actuelle de cette information.

  - Par exemple {Localisation =”La gare”, Terminal =”PDA”, Heure =”t1”}. Cette représentation est caractérisée par sa facilité d'implémentation. La représentation la plus connue illustrant ce modèle est le modèle ContextToolkit de [25].

- **Représentations des données contextuelles basées sur XML :** ces modèles utilisent une structure de données hiérarchique. Plusieurs modèles de description des informations de contexte se dérivent à partir de ce langage, à savoir UAProf (User Agent Profile) [le web site] et CC/PP (Composite Capabilities/Preference Profile).

Le langage CC/PP est une recommandation de W3C (World Wide Web Consortium) Il est basé sur RDF (Resource Description Framework) qui est utilisé afin de créer des profils décrivant les capacités des terminaux et les préférences d'un agent utilisateur. CC/PP est utilisé pour personnaliser le contenu sur la base des capacités d'un terminal et les préférences de l'utilisateur [26].

- **Représentations graphiques des données contextuelles :** Ces modèles consistent à modéliser les informations contextuelles selon un graphe conceptuel. [27] ont utilisé une représentation graphique basée sur UML (UnifiedModelingLanguage) pour la modélisation des informations contextuelles pour la gestion des services web.

- **Représentation orientée objet des données contextuel :** l'approche objet est utilisée pour pouvoir intégrer facilement la représentation du contexte au sein de l'application qui en dépend. Cette représentation du contexte s'appuie sur les propriétés de nommage, d'encapsulation, de réutilisation et d'héritage. Chaque type de contexte utilisé est composé de plusieurs objets contexte qui constitue la superclasse de plusieurs éléments du contexte, notamment : le temps, le réseau, la localisation, l'utilisateur, la machine. les autres éléments peuvent être ajoutés par l'héritage de la super classe [27].

- **Représentations des données contextuelles basées sur des ontologies** : Une ontologie est un ensemble structuré de concepts. Les concepts sont organisés dans un graphe dont les relations peuvent être des relations sémantiques ou des relations de composition et d'héritage. Elle fournit un vocabulaire représentatif pour un domaine donné, un ensemble de définitions et d'axiomes qui contraignent le sens des termes de ce vocabulaire de manière suffisante pour permettre une interprétation consistante des données représentées au moyen de ce vocabulaire. Parmi les travaux qui ont utilisés des ontologies, le travail de [28]. Il définit une collection d'ontologies appelée COBRA-ONT pour la modélisation de contexte dans un environnement d'une salle de rencontre intelligente. COBRA-ONT des concepts typiques associés avec des places, des agents et des événements.

4. Adaptation au contexte

L'adaptation au contexte est l'ensemble des réactions prévues suites aux changements de contexte. L'adaptation se base sur un ensemble de règles. Ces règles sont implémentées selon des langages de programmations traditionnels ou bien la logique de prédicats. Pour remédier au caractère incertain de contexte, certains travaux adoptent la logique floue ou la logique probabiliste. Plusieurs formes d'adaptation peuvent être se présentées [29]

- ***Adaptation architecturale*** **:** désigne les changements faits, en temps réel, dans la structure des composants du système et/ou dans les interactions entre eux en utilisant un modèle architectural du système.

- ***Adaptation compositionnelle*** **:** concerne les modifications de la structure et du comportement d'une entité logiciel, en temps réel, en réponse aux changements arrivés à son environnement d'exécution.

- ***Adaptation structurelle*** **:** Une adaptation structurelle signifie le changement dynamique de type des composants de l'application tel qu'une signature d'une méthode.

- ***Adaptation*** **comportementale :** désigne les changements dynamiques dans la phase de l'exécution d'un composant logiciel (par exemple : en changeant sa configuration ou bien en interceptant ses requêtes et réponses).

- ***Adaptation de contenu*** **:** concerne la transformation et la manipulation des contenus en se basant sur les caractéristiques de l'application et des terminaux utilisés.

5. Conclusion

Dans ce chapitre nous avons abordé une station essentielle dans le chemin pour bien comprendre les systèmes sensibles au contexte, Nous avons fait un tour d'horizon sur les définitions du contexte les plus utilisés et ses caractéristiques, et les différentes catégories du contexte reconnus, ensuite on a présenté la sensibilité au contexte. A la fin nous avons exposés l'architecture générale d'un système sensible au contexte.

Dans le chapitre suivant on va étudier les agents ambiants, on peut voir la déférence entre les systèmes SMA ordinaire et les systèmes ambiants, leurs propriétés et leurs mécanismes d'auto-organisation, et enfin on va terminer par le différent model existant des agents ambiants.

CHAPITRE II
LES AGENTS AMBIANTS

I. INTRODUCTION

Après l'étude de la sensibilité au contexte dans le chapitre précédent et la présentation des différents points de vus on va définir et conceptualiser les agents ambiants. On montrera la motivation pour l'utilisation des agents ambiants, on décrira les prospérités requises pour la sensibilisation au contexte de l'environnement ambiant et aussi les mécanismes d'auto-organisation et on terminera par leurs modèles.

II. DEFINITION DES AGENTS

La définition classique du concept d'agent est donnée par **[**30], qui le décrivent comme une *"entité virtuelle ou réelle qui est capable d'agir sur son environnement, qui possède des moyens de perception et de représentation partielle de son environnement, qui est capable de communiquer avec d'autres agents et qui est autonome dans sa prise de décision".*

Donc les propriétés d'un agent sont [30] :

- Il évolue dans un environnement (physique ou virtuel) depuis lequel il perçoit des Informations et sur lequel il est capable d'agir ;

- Il est autonome, ce qui signifie qu'il possède une représentation interne de son environnement auquel les autres agents n'ont pas accès, grâce à laquelle il décide seul de ses actions sans l'intervention d'un utilisateur extérieur ou d'un autre agent ;

- Il est capable d'interagir avec d'autres agents, soit directement, soit par envoi de messages, soit par inscription dans l'environnement.

Un système multi-agent est alors composé d'un ensemble d'agents interagissant les uns avec les autres, chaque agent possédant son propre objectif et ses propres connaissances et compétences. Ces connaissances et compétences étant réparties entre les différents agents. Le comportement du système multi-agent résulte des comportements locaux de chacun des agents qui le composent et de leurs interactions. L'observation d'un système multi-agent peut alors se faire selon deux niveaux [31] :

- Le niveau *macro* où l'observateur extérieur au système perçoit la totalité du système en ignorant les interactions internes au système.

- Le niveau *micro* où l'observateur perçoit les parties internes du système et leurs interactions.

La fonctionnalité globale du système, observée au niveau *macro*, dépend du comportement local des agents qui composent ce système au niveau *micro*, mais aussi de l'organisation de ces agents et de leurs interactions.

III. AGENT AMBIANT

1. Définition

Un agent ambiant est un agent autonome qui est embarqué sur un appareil électronique de telle façon que cette union est appelée *Ambient Intelligence Device (AmID)*. Ces agents sont capables d'interagir avec l'environnement réel, avec d'autres agents et de coopérer entre eux pour atteindre à un résultat de la tâche globale. Tous ces AmID rendent possible l'existence de l'AmI, ils ont des capacités et des comportements pour que l'interaction soit un succès [32].

2. Les comportements d'un agent AMBIANT:

Selon [32], un agent ambiant a les comportements suivants :

- **Avoir une représentation de son environnement :** pour un agent son environnement est là son vision locale du monde ; c'est-à-dire ; tout ce qu'il perçoit avec ses capteurs et tout ce qu'il reçoit à travers de l'interaction avec les autres agents.

- **Être capable d'interagir avec les autres :** l'agent n'est pas un système isolé et inerte, il est une unité autonome et comme il est une partie de tout le système. Par conséquent, l'agent interagit continuellement, il doit s'intégrer dans le système en envoyant et recevant des informations des autres agents ou échangeant des services.

- **Être relié à un appareil électronique :** l'agent est embarqué sur l'appareil pour qu'il puisse accéder aux services, capteurs et actuateurs de l'appareil. Par conséquent, l'agent

ambiant a les capacités de l'appareil mais avec le bénéfice que l'agent les utilise de façon intelligente.

- **Survivre au mieux :** le cycle de vie d'un agent sensé être infini (percevoir, décider et agir) mais l'agent est dépendant de l'autonomie d'énergie de l'appareil qui le supporte. Par conséquent, l'agent doit considérer cela dans son cycle de vie de manière à s'adapter aux caractéristiques de l'appareil en ralentissant et/ou suspendant ses activités. De même, il doit déléguer ses fonctions aux autres agents qui ont des capacités supérieures.

- **Intégrer un nouveau système :** le potentiel de la mobilité de l'agent ambiant peut changer l'environnement et par conséquent changer de système. L'agent doit être spontanément coopératif pour que le fonctionnement du système soit efficace ; c'est-à-dire que l'agent doit se présenter quand il arrive pour la première fois dans un système. Cette présentation se comporte la raconter toutes ses caractéristiques principales et fonctionnalités comme les services qu'il peut offrir, les « langues » qu'il comprend, l'appareil sur lequel il est embarqué, sa capacité d'énergie, etc.

IV. MOTIVATION POUR L'UTILISATION DES AGENTS AMBIANT

A l'étude de la plupart des agents coopératifs dans les systèmes multi-agents coopératifs, [32] a différencie que les agents ambiants doivent avoir des caractéristiques et des capacités nécessaires au domaine de l'intelligence ambiante. Très souvent, les agents classiques sont Intégrés dès le départ dans un même système. Ils communiquent avec un même langage et les mêmes protocoles de communication.

Souvent ils ne sont pas destinés à fonctionner sur des appareils mobiles. Par contre aux agents classiques, les agents ambiants coopératifs n'ont généralement Aucune connaissance initiale sur leurs voisins, ni de leur existence, ni de leurs caractéristiques. Ils doivent donc avoir la capacité de s'intégrer dans un système, lui-même en changement. Ces agents peuvent avoir différents langages et protocoles de communication. Ils sont conçus pour fonctionner sur des appareils mobiles avec les contraintes en découlant (vu la **figure2.1**). Nous détaillons ici les deux spécificités principales des agents ambiants.

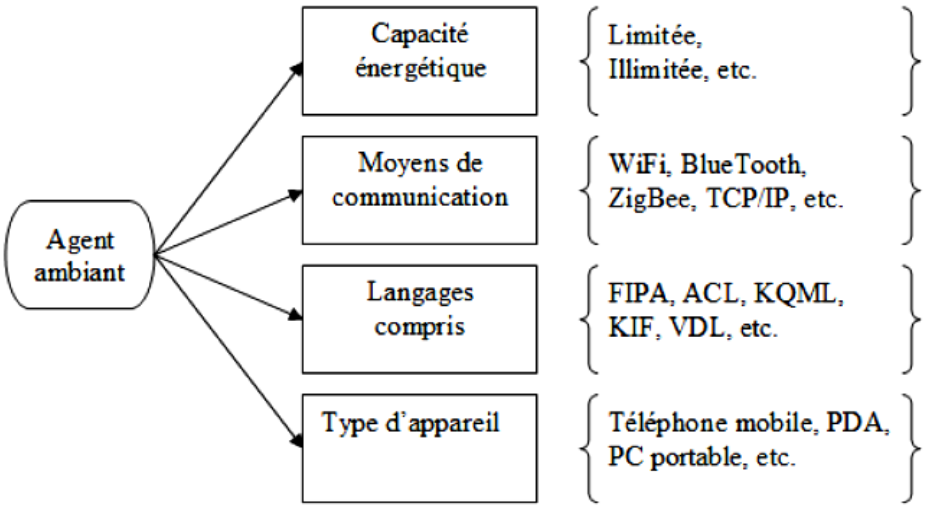


Figure 2.1 : caractéristiques d'un agent ambiant [32]

V. PROPRIETES REQUISES POUR LA SENSIBILISATION AU CONTEXTE EN ENVIRONNEMENT AMBIANT

Les systèmes ambiants tiennent des caractéristiques particulières qui font de leur sensibilisation au contexte un problème complexe mais aussi fournit une maximum d'exploitation des ressources accessible : ils sont composés de dispositifs distribués et hétérogènes, certains de ces dispositifs pouvant apparaître et disparaître en cours de fonctionnement, et reliés entre eux par de nombreuses interconnexions.

Pour permettre à un système ambiant d'être sensible au contexte, certaines propriétés doivent être respectées. Plusieurs propriétés ont été proposées dans des plusieurs études [25],[33], nous présentons ici les plus adoptées:

- **Généricité** : La représentation des données contextuelles peut être très variée, allant d'un simple doublet <nom – valeur> jusqu'à une ontologie complète. Un système est dit générique si sa représentation des données contextuelles lui permet d'intégrer n'importe quel type de données [25].

- **Expressivité** : Une donnée contextuelle ne se limite pas à une valeur, elle est aussi associée à une sémantique. En fonction de la modélisation du contexte utilisée, il est possible de disposer d'une sémantique de données plus ou moins riche [25].

- **Distribution** : Un système distribué est physiquement et/ou logiquement décomposé en différents processus fonctionnant sans exiger la présence d'un processus central (un serveur, par exemple), et limitant leurs échanges aux données nécessaires et suffisantes à leur fonctionnement [25].

- **Ouverture** : Un système est ouvert s'il peut intégrer et exploiter de nouvelles données en cours de fonctionnement sans nécessiter d'être (re)configuré et/ou de recommencer la totalité de ses traitements. Il s'agit aussi de pouvoir supporter la disparition de données connues en cours d'exécution [33].

- **Pro-activité** : La pro-activité d'un système désigne sa capacité à anticiper et à prendre des décisions concernant une situation. Un système proactif ne se limite pas à la perception et à l'interprétation de la situation dans laquelle il se trouve ; il est aussi capable d'agir pour améliorer la situation courante au lieu d'attendre qu'un utilisateur le fasse par lui-même [33].

- **Confidentialité** : La confidentialité permet de garantir à l'utilisateur que ses données contextuelles privée sont sécurisées et indisponible a des tiers et qu'il en maîtrise le contenu et l'usage [33].

- **Explicabilité** : L'explicabilité d'un système se manifester a son habileté de donner, en continu ou en cas de demande d'un utilisateur, les faits et le raisonnement qui expliquent son comportement. Pour évaluer les différents systèmes de sensibilisation au contexte décrits dans la section suivante, nous nous basons sur des critères issus de ces propriétés [33].

VI. MECANISMES D'AUTO-ORGANISATION

Les mécanismes d'auto-organisation sont les solutions et les réponses aux situations non coopératives (SNC). Puisque les environnements sont instables les situations non coopératives

sont fréquentes. Pour chaque situation non coopérative il y a des solutions et des réponses possibles fournir par les agents [34].

1. SNC de perception

Les agents peuvent parler différentes langues comme ils peuvent avoir différents protocoles de communication parce que, simplement, l'intelligence ambiante exige l'hétérogénéité entre ses agents. Ainsi, ils peuvent avoir des difficultés qui sont : l'incompréhension et l'ambiguïté.

- **Incompréhension :** un agent a reçu un message qu'il ne comprend pas. Il peut résolue Cette incompréhension avec la collaboration d'un autre agent qui peut traduire le message. Par conséquent, un agent traducteurs intervenus s'il y a des différences entre agent au sujet de la communication.

- **Ambiguïté :** un agent a reçu un message qu'il peut interpréter de plusieurs manières. L'ambiguïté peut être résolue de deux manières différentes. La première réside en la participation d'un autre agent qui peut préciser le sens du message selon le domaine. La seconde manière consiste à demander un éclaircissement à l'agent qui a envo*yé* le message.

2. SNC de décision

Parfois l'agent reçoit un message qu'il comprend mais il ne sait pas que faire ou il n'a pas de réponse pour ce message. Ce cas est appelé : **incompétence** et **improductivité** respectivement. Les deux difficultés peuvent généralement être résolues de façon identique : l'agent ayant reçu le message et ne sachant pas que faire répond à l'émetteur qu'il ne peut rien faire. De toute façon ces situations doivent être évitées, idéalement tous les agents doivent savoir que faire avec un message qu'ils ont compris [31].

Il faut développer correctement les agents pour éviter que les situations non coopératives de décision arrivent.

3. SNC d'action

A cause de la mobilité des agents ambiants ils peuvent changer d'environnement et leurs actions peuvent ne pas avoir d'effet ; ce que mis l'agent dans la situation non coopérative **d'inutilité**. Pour résoudre, cette situation chaque fois que l'agent change d'environnement il doit se présenter pour le découvrir et aussi peut ralentir ses activités s'il n'y a pas plus rien à faire puisqu'il se doit éviter de consommer son énergie [34].

VII. LES MODELES DES AGENTS AMBIANTS

1. Modèle schématique d'un agent ambiant

Le model schématique nous donne une vue sur le cycle de vie d'un agent d'une façon clair. Puis on va voir le model fonctionnel avec la description et classification de chaque phase et comportement comportements [35]

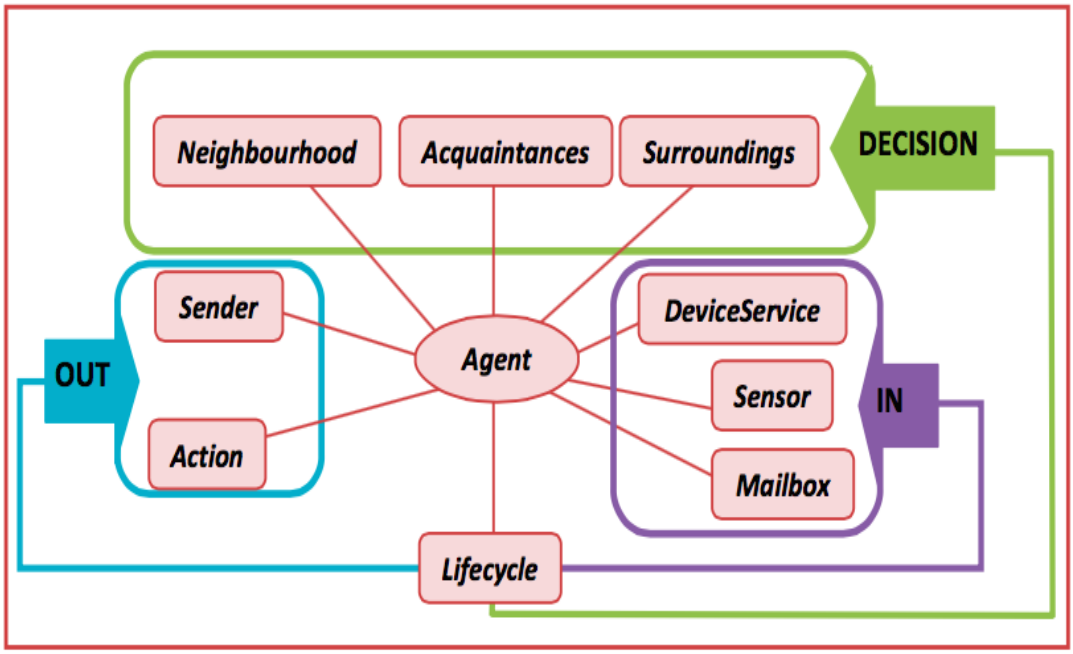


Figure 2.2 : model schématique d'un agent ambiant [35]

2. Modèle fonctionnel d'un agent

Le model fonctionnel expose d'une les comportements et les microcomposants d'un agent ambiant. Le comportement de l'agent comprend l'intelligence, la prise de décisions, l'adaptation et toutes les capacités qui ne sont pas supportées par le niveau opératoire et également les mécanismes montrés précédemment pour résoudre les situations non coopératives [35].

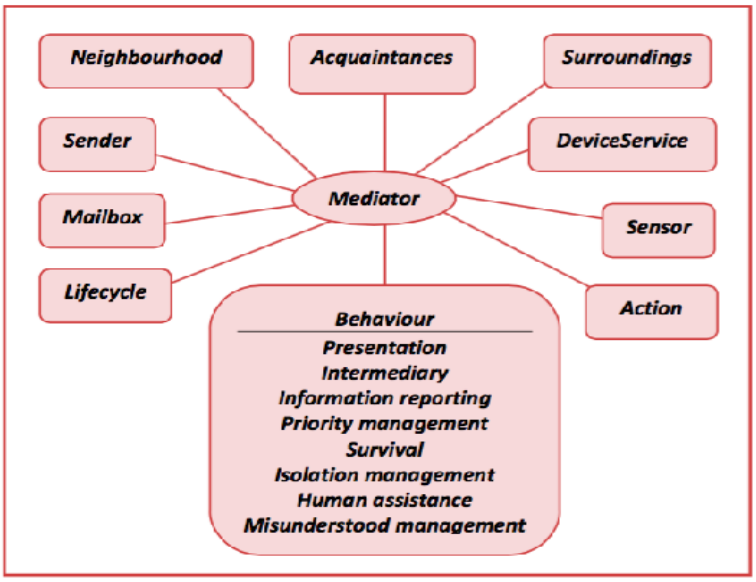


Figure 2.3. Modèle fonctionnel d'un agent ambiant [35].

- **Lifecycle**: c'est un microcomposant représente le cycle de vie de l'agent : (perception, décision et action). L'agent perçoit ce qui se passe dans son environnement grâce aux messages reçus et les informations obtenues des capteurs et des services de l'appareil où il est embarqué.

- **Mailbox :** c'est une boîte aux lettres ; tous les messages que l'agent reçoit y sont stockés. Il offre un service *getMessages()*, pour que les autres microcomposants puissent consulter et accéder aux messages reçus. Par conséquent avec le microcomposant précédent ils forment l'ensemble des moyens de communications de l'agent.

- **Sender :** ce microcomposant permet l'envoi de messages ; c'est-à-dire ; la communication avec les autres agents. Il offre un service *send()* pour que les autres microcomposants puissent envoyer des informations.

- **Neighbourhood :** l'agent a besoin de communiquer avec les autres agents donc il doit connaître ses voisins. L'agent a une représentation de son environnement pour pouvoir prendre des décisions donc une partie importante pour lui est son voisinage. Le voisinage peut changer, c'est pourquoi il fournit des services aux autres microcomposants pour ajouter ou effacer des voisins.

- **Acquaintances :** tous les agents connus ne sont pas dans le voisinage et tout le voisinage n'est pas d'égale importance pour l'agent. De cette façon il garde ceux qui sont les plus importants parmi les agents rencontrés au cours du temps : les amis, le groupe d'intérêt, etc. Toute cette information est utilisée pour prendre des décisions.

- **Surroundings:** l'agent doit sauvegarder certaines informations de son environnement qu'il connaît pour prendre de bonnes décisions. Chaque agent pour prendre des décisions nécessite différentes informations de son environnement.

- **Devise-Services:** avec les deux derniers microcomposants l'agent peut obtenir l'information de l'environnement et y agir, mais l'appareil sur lequel l'agent s'exécute peut avoir des services supplémentaires qui sont aussi importants. Comme ce microcomposant fait signal aux caractéristiques de l'appareil concret, il comprend tous les services de l'appareil.

- **Sensor :** l'agent est embarqué sur un appareil, cet appareil peut avoir des capteurs et l'agent doit pouvoir accéder à l'information que les capteurs ont de l'environnement (*getInformationSensor()*).

- **Action :** si l'agent a accès aux capteurs de l'appareil, il a également accès aux actuateurs pour pouvoir changer son environnement (actions simples) et il peut ordonner à l'appareil de faire des actions plus complexes.

Toutes les capacités ne sont pas prises en compte par les microcomposants. Les restes capacités manquantes se présenté comme des comportements.

Les comportements des agents ambiants pour ce scénario sont ceux de la figure précédente : présentation, intermédiaire, rapport d'information, gestion de priorité, survie, gestion d'isolement, gestion des incompréhensions.

- **Présentation :** L'agent ambiant a la capacité de s'intégrer dans un nouveau système et par conséquent quand un agent arrive dans un nouvel environnement il doit se présenter aux autres agents. Cette présentation consiste à dire quelles sont ses fonctions, les caractéristiques de l'appareil sur lequel il est embarqué, etc.

- **Intermediary:** L'agent ambiant est coopératif et doit aider à ce que les messages qui sont envoyés pour son voisinage arrivent au destinataire ; c'est-à-dire ; il doit éviter les situations non coopératives de décision. Par conséquent, chaque fois qu'il reçoit un message dont le destinataire n'est pas lui il doit participer au relais pour que l'information remonte aux agents pertinents.

- **Information reporting:** L'agent ambiant ne communique que les informations importantes pour résoudre la tâche et être coopératif (pas de SNC d'action). Les informations importantes se classent en :

  - *Information de secours*: le scénario concerne le feu de forêt où le but global est de sauver toutes les victimes et éteindre le feu. Par conséquent, l'information importante est la localisation du feu et la localisation et le nombre des victimes.

  - *Information de sauvetage*: quand un pompier prend la décision d'aller sauver un groupe de victimes ou d'éteindre un feu il doit communiquer où il va et quelle en est la raison. Ainsi les autres pompiers décideront où ils vont pour que toutes les victimes soient sauvées et tous les feus soient éteints.

  - *Information d'arrêt* : l'agent n'a pas une vie illimitée et pour être coopératif il doit communiquer son arrêt et diffuser tous ses connaissances non connues des autres.

  - *Information d'aide directe* : Toute décision prise par un robot relativement à l'aide directe aux blessés (prendre le blessé, lui donner un médicament, lui mettre un masque

d'oxygène, etc.) doit être communiquée au voisinage pour qu'il n'y ait pas de concurrence ou de conflit avec les voisins.

- **Survival:** La capacité des agents est de survivre au maximum. Cette survie est liée aux caractéristiques des appareils. Étant donné que la plupart des appareils sont mobiles et ont des batteries, les agents n'ont donc pas de vie illimitée. Il y a également du feu et tous les appareils ne supportent pas le feu donc les agents peuvent mourir. Pour que les agents survivent le plus possible, ils doivent fuir le feu, contrôler leur cycle de vie et déléguer leurs fonctionnalités ; ainsi les agents augmentent leur durée de vie.

- **Isolation management :** Un agent ne doit pas rester tout seul parce que les informations qu'il envoie et les informations pertinentes pour lui n'arriveront pas. Donc on doit éviter que les agents soient isolés. Ce comportement peut être découpé en deux :

  - **Escort :** Un agent doit se garder d'être isolé donc s'il peut être accompagné, il doit faire ce choix. Pour éviter qu'un agent soit isolé il faut éviter de fuir les voisins qui ont un seul voisin, il faut les accompagner.

  - **Repair**: Si un agent perd tout contact avec son voisinage, il doit chercher un autre contact pour ne pas être isolé.

- **Priority management** : L'agent ambiant reçoit les informations de l'environnement au moyen de messages et de ses capteurs, mais toutes les informations n'ont pas la même priorité. Par exemple l'information de présentation d'un agent est moins importante que l'information de localisation du feu. De même, un agent qui va communiquer la localisation d'un feu s'il n'y pas de voisinage doit solutionner cette situation avant l'autre.

- **Misunderstanding management :** Comme l'agent peut recevoir un message qu'il ne comprend pas il doit chercher un agent qui le traduise où qui l'éclaircisse. C'est pourquoi l'agent peut comprendre le message reçu et pouvoir décider que faire. De cette manière on résout les difficultés de perception vues auparavant.

VIII. CONCLUSION

Dans ce chapitre nous avons présenté quatre grandes idées. Tout d'abord, les agents ambiants en analysant leurs capacités et en les comparant aux autres agents coopératifs étudiés par l'équipe SMAC ont été déduits. Ensuite, nous avons exploré les comportements de ces agents ambiants. Par la suite, nous avons analysé le mécanisme d'auto-organisation des agents ambiants en explorant les situations de non-coopération qui peuvent se présenter à ces agents et comment ils peuvent les résoudre. Enfin, nous avons indiqué les modèles des agents ambiants : le model schématique et le model fonctionnels qui nous donne une vue claire sur les comportements et les microcomposants des agents ambiants.

CHAPITRE III

LES TRAVEAUX RELIÉS

I. INTRODUCTION :

Il n'y a aucun doute que les systèmes sensibles au contexte sont un domaine de recherche très actif dans l'intelligence ambiante dans laquelle les systèmes des agents ambiants jouent un rôle essentiel [36]. Dans cette section, nous avons concentré notre attention d'aperçu sur quelques approches multi-agent qui ont été employés dans l'AMI. Chacun de ces approches multi-agent a ces propres dispositifs, conception unique, domaine d'application et environnement. La plupart de ces approches partagent certaines caractéristiques communes. Nous essayons de présenter une brève vue d'ensemble des approches que nous avons choisies avec un tableau qui fournit l'essentiel de ces approches discutées.

II. LES TRAVAUX EXISTANTS

1. AmbieAgents

L'infrastructure d'AmbieAgents est proposée comme une solution évolutive pour les services des informations sensibles au contexte mobile. L'agent du contexte dans cette infrastructure gère les informations de contexte en tenant compte des questions de confidentialité. Ils reçoivent des informations de contexte anonymes et exécute des requêtes afin de recevoir des informations pertinentes dans un contexte donné. L'agent de recommandation dans cette infrastructure utilise l'ontologie et le raisonnement les plus avancées afin d'effectuer des requêtes plus spécifiques [11].

2. AmbieSense

Le système AmbieSense possède une architecture multi-agent pour les services des informations sensibles au contexte, spécialement pour les utilisateurs mobiles qui se compose des étiquettes sans fil de contexte, des plateformes contenantes des dispositifs et des utilisateurs mobiles. Le but du projet AmbieSense est de développer une plateforme répondant aux besoins des voyageurs en matière de l'information et fournissant des informations ambiantes et personnalisées d'une façon sensible au contexte. Le système AmbieSense est constitué de quatre agents. Le premier, *agent de contenu*, fournit le contenu de bas niveau, *agent de contexte*, responsable de l'interaction vise- avis de l'environnement, *agent de Recommandation*, utilise les mécanismes de raisonnement pour fournir des contenus appropriés aux utilisateurs et, un *agent*

d'intégration, qui permet l'emballage et traite des possibilités d'interaction avec l'environnement ambiant et non-ambiant [36].

3. CARISMA

CARISMA se réfère au système d'intergiciel réflectif sensible au contexte. Il est concentré sur les systèmes mobiles qui sont extrêmement dynamiques. L'adaptation (également appelée la réflexion) est le rôle principal de CARISMA. Le contexte est stocké comme des profils d'application (base XML), qui permet à chaque application de maintenir les métadonnées au-dessous de deux catégories : passif et active [36].

La catégorie passive est définie à partir des règles de l'état intermédiaire quand certains événements spécifiques se produisent, telles que l'arrêt si la batterie est basse.

Cependant, des conflits pourraient se produisent quand deux profiles définissent les mêmes règles. La catégorie active maintenir les relations entre les services des applications d'être employés, aussi les politiques, et les configurations de contexte. Cette capacité indique comment se comporte dans les différentes conditions environnementales et utilisateurs. Un mécanisme de résolution du conflit est également présenté dans CARISMA basé sur des techniques macro-économiques. Un protocole d'enchère est employé pour manipuler la décision pendant qu'elles soutiennent les plus grands degrés d'hétérogénéité au-dessus d'autres solutions de rechange. Des décisions finales sont prises afin de maximiser le bien-être social parmi les agents [36].

4. CARMEN

CARMEN est un logiciel intermédiaire pour la gestion des ressources dans les systèmes sensibles au contexte. Ce système prend en charge la reconfiguration automatique de l'accès Internet basé sur le contexte dans un environnement sans fil. Il a la capacité de contrôler les ressources même s'il est débranché soudainement. On peut aussi utiliser CARMEN en tant qu'agents mobiles. Cette capacité permette l'utilisateur d'accès aux ressources dans l'environnement sans fil. La responsabilité de la capacité inclut l'appui de la migration à travers l'environnement divergent avec la gestion des ressources [36].

5. Cobra

Le cobra (*context broker architecture* : architecture de courtier de contexte) est une architecture centrale d'agent courtier qui fournit le partage de la connaissance et le raisonnement de contexte pour les smart-espaces [28]. Elle est particulièrement concentrée sur les endroits de réunion. Le cobra aborde deux thèmes principaux [37] :

- Soutien les appareils ordinateurs mobiles qui ont des ressources limitées
- Répondre aux soucis de la confidentialité d'utilisateur.

L'information de contexte est modelée par l'utilisation des ontologies de HIBOU. L'élément principal du cobra ou le courtier de contexte est un agent de serveur. Un courtier de contexte comporte quatre composants fonctionnels :

- Base de connaissance de contexte (fournit le stockage persistant pour l'information de contexte).
- Moteur de raisonnement de contexte (exécute le raisonnement au-dessus de l'information contextuel stockée).
- Module d'acquisition de contexte (récupérer le contexte à partir des sources contextuelles).
- Module de gestion de politique (contrôle des politiques, telles que l'accès aux données).

Bien que l'architecture soit centralisée, plusieurs courtiers peuvent fonctionner ensemble par une fédération de courtier.

6. Cyberguide

Cyberguide est l'un des premiers systèmes qui ont fournis des guides touristiques sensible au contexte pour les utilisateurs mobiles. Ce système incorpore les l'utilisateur et la localisation, il fait un historique des endroits passés pour fournir les différents types des services considérés comme des vrais guides touristiques. Il présente un modèle et une architecture de multi-agent appelés le SKELETONAGENT, le but est de résoudre les problèmes dans les

domaines de Web tourisme, par l'intégration des techniques de rassemblement et de planification des informations [36].

7. SIM

SIM (*Sensor Information Management* : gestion de l'information du capture) est concentré sur le domaine ***domotique***. SIM utilise une architecture basée selon la spécification standard fournie dans la fondation pour les agents physiques intelligents. Dans son architecture SIM contient les composants suivants : Système de gestion d'agent, permettant de contrôler les agents, de maintenir l'annuaire des agents et de tenir les services et les descriptions des agents. Il a aussi la voie de transmission respective d'agent permettant de transférer des messages dans l'environnement. L'objectif de SIM est de collecter des données de capteur provenant à partir des sources multiples et les agréger ensemble pour analyser et obtenir des informations plus précises. SIM rassemble deux types d'informations : niveau de nœud et niveau d'attribut. Au niveau de nœud : nœud ID, l'emplacement et la priorité sont collectées. Les attributs et leurs mesures sont stockés dans la base de l'information d'attribut. Un algorithme de localisation est introduit à l'aide d'un dispositif de positionnement mobile. Un gestionnaire de position gère la localisation. SIM est capable de résoudre les conflits des informations reçues selon la priorité de capture. La résolution du conflit est faite par un directeur de contexte à l'aide de l'agrégation, la classification, et des composants de décision. Même si SIM ne concentre pas sur la gestion du contexte au niveau du matériel, leur approche est plus contiguë de bas niveau au lieu de haut niveau par rapport à d'autres projets. Ce mécanisme est extrêmement puissant et efficace pour gérer et éviter les informations conflit [37].

8. X.MAS

A. Addis et autres, [37] propose X.MAS comme architecture multi générique d'agent, elle est concentrée sur les tâches de recherche documentaire qui sont visées pour la rechercher, le filtrage, et la restructuration des données pour fournir des informations selon les intérêts de l'utilisateur. Il se compose par les catégories d'agents logiciels suivants: *Agents de l'information*, adaptés aux besoins des clients pour accéder à la source d'informations ; Agents de filtre pour transformer l'information selon les préférences d'utilisateur ; Agents de tâche pour d'aider les utilisateurs à effectuer des tâches typiquement en coopération avec d'autres agents ; l'agent d'interface, responsable des interactions d'utilisateurs , et l'agent Moyen qui agit comme un

établissement de communication entre les agents demandeurs et les agents fournisseurs des services. Les créateurs présentent WIKI.MAS qui classifie le contenu de Wikipedia.

9. DALICA

DALICA est un système multi-agent qui utilise des données de localisation pour la diffusion des informations sur les biens culturels. DALICA présente une architecture plus intéressante, elle est une combinaison de l'ontologie (le positionnement continu le long de Galileo) et le profil utilisateur. Ce système est capable de surveiller les visiteurs et le transport de ces actifs. DALICA est une partie du projet européen CUSPIS. L'application CUSPISse compose d'un environnement d'application, d'un agent générateur, d'un agent de profil d'utilisateur et un agent de sortie. L'agent générateur génère le profil de l'utilisateur quand un utilisateur lance une visite dans l'application. L'agent de profil d'utilisateur extrait les intérêts d'utilisateur et surveille leur comportement. Enfin, l'agent de sortie gère la communication entre les appareils des utilisateurs et les infrastructures [11].

10.Fusion@

Fusion @ [32] fondé sur une architecture décentralisée dans laquelle les agents recherchent des services pour les utilisateurs. L'architecture multi-agent modulaire est mis en œuvre Fusion @. Dans cette architecture, les applications et les services sont contrôlés par des agents délibératifs BDI (Belief, Desire, Intention). Voici les différents types d'agents de Fusion @: *CommApp Agent, CommSer Agent, Admin Agent, Supervisor Agent, Directory Agent, Security Agent, et Interface Agent*. Chaque agent a des rôles spécifiques, des capacités et des caractéristiques. Agent CommApp est responsable des communications entre les plates-formes et les applications. Agent CommSer est responsable de la communication entre les services et les plates-formes. Les listes des services sont gérées par *Directory Agent*. *Supervisor Agent* surveille les fonctionnalités des autres agents dans le système. Les questions de sécurité sont prises en charge par *Security Agent*. Un type spécial d'agent de faible empreinte appelé *Interface Agen* réside sur les appareils mobiles des utilisateurs pour profiter des services de l'environnement [36].

III. TABLEAU DE COMPARAISON

Num	L'approche	Platform / environment	Fonctionnalités	Domain d'application	Structure d'agent
01	CARISMA 2003	mobile	L'adaptation dynamique	Smart espace de conférence	Distribué
02	CARMEN 2003	mobile	reconfiguration automatique	La gestion des ressources	Distribué
03	AmbieSense 2004	mobile	Augmentation des informations numériques à des objets physiques	informations voyageurs	fixe
04	CoBrA 2004	mobile	Prise en charge les appareils mobiles d ressources limitées	Les espaces smart	centralisée
05	AmbieAgents 2005	mobile	évolutive	livraison en fonction du l'information contextuel	fixe
06	CyberGuide 2005	web	Le premier système fournir un guide touristique	E- guide touristique	Distribué
07	SIM 2007	captures	puissant et efficace pour gérer et évite les conflits d'informations	Système de suivi la localisation	Distribué
08	X.MAX 2008	web	Dynamique	Tâches de Recherche d'information	Distribué
09	DALICA 2008	PDA (ordinateur de poche) et web	représentation des Ontologies et la diffusion de l'information	biens culturels	centralisée
10	Fusion 2009	mobile	services orientes	Convergence des dispositifs	fixe

Tableau 3.1 : comparaison des approches multi-agents ambiants

IV. DISCUSSION

AmI a évolué en raison les possibilités croissantes offertes par les récentes technologiques avancées. Un système sensible au contexte ou CAS (*Context-aware system*) est l'un des domaines des systèmes AmI. CAS est équipé par des dispositifs qui reconnaissent le contexte et agissent en conséquence pour soutenir les individus. Une grande partie de ces systèmes fonde sur une infrastructure multi agent ambiant. Les principaux avantages des systèmes fondés sur cette infrastructure sont les fonctionnalités intégrées telles que le comportement autonome, réactif et proactif. Un autre avantage important est l'intelligence du système. Cela signifie que le système et les entités qui le composent peuvent apprendre et raisonner par eux-mêmes, sans qu'il soit nécessaire de les programmer pour chaque fonction. Dans ce chapitre, nous avons examiné plusieurs domaines d'application de la sensibilité au contexte, y compris la domotique, aide à la vie, l'industrie, le tourisme, etc... Dans ceux-ci, nous avons examiné globalement les caractéristiques, le rôle de l'agent, la plate-forme et les défis. L'objectif principal est de connaître les différentes exigences couvertes par les approches des Agents ambiants [32].

Il y a beaucoup de défis et de la complexité de la conception d'un système CAS qui peuvent être traitées efficacement en utilisant des systèmes multi-agent [32]. Comme nous avons étudié les différentes approches multi Agent, nous trouvons qu'une application CAS bien structuré doit être adaptable, intelligent, sensible, transparente et omniprésente. Ces capacités seront donc fournir les humains une assistance objectif. Voici les autres observations faites par le biais de cette enquête:

- certains systèmes ne pas utiliser des agents comme un paradigme de calcul distribué

- les agents sont utilisés dans plusieurs projets sur la sensibilisation au contexte, mais il n'y a pas de travail direct sur l'interopérabilité des sources de contexte et les utilisateurs. [11]

Trois phases principales sont à considérer au court de la réalisation d'une application sensible au contexte:

- La valeur sociale aux utilisateurs
- L'inférence des mécanismes
- Les systèmes distribués sensibles au contexte [38]

Du point de vue de la mise en œuvre, le développement de ces systèmes cotes logiciels et matériels est une tâche difficile. Il nécessite des outils et des procédures adéquates pour eux-mêmes et des technologies de communication adaptable. Sur la base de l'étude des différentes approches, nous avons identifié les concepts suivants pour la mise en service effective de multi agents dans un environnement sensible au contexte :

- Les systèmes sensibles au contexte :

- Devraient obtenir l'information et répondre à l'utilisateur ainsi les dispositifs dans l'environnement.
- Devrait soutenir temporelle, spatiale et continue activités de déduire le contexte actuel.
- Le système doit être capable d'adapter à l'utilisateur, et changements de l'environnement.
- Capable d'anticiper les souhaits des utilisateurs en fonction de leur comportement passé, intégrant soit implicite ou explicite des commentaires
- En outre, le contexte temporel, l'incertitude et la confidentialité des informations doivent également être pris en compte dans la conception l'approche multi agent pour la sensibilité au contexte. Ainsi, la conception du CAS basée sur les caractéristiques ci-dessus intègre les connaissances de plusieurs domaines et agit comme un environnement intelligent.

V. CONCLUSION

Ce chapitre examine les différentes approches multi-Agent pour la sensibilisation au contexte dans l'intelligence ambiante et identifié leurs exigences générales. Les concepteurs, les chercheurs et les fondateurs préfèrent utiliser l'approche multi agent sur l'approche traditionnelle en raison de ses caractéristiques telles que l'adaptabilité, la collaboration, la mobilité...

Cette enquête [36] arpentée certaines des zones d'application des approches Multi Agent telles que les services de guide touristique pour les voyageurs, la livraison en fonction du contexte de l'information, domaine de la maison intelligente (domotique)... Ce chapitre a également abordé les défis à prendre en compte dans la conception d'une approche générique efficace multi Agent pour la sensibilisation du contexte. En outre, sensibilisation de contexte de multi agent doit soutenir une gestion efficace des ressources hétérogènes et les fonctionnalités de l'environnement ambiant. En raison de la nécessité d'échanger énorme volume de données, il est nécessaire d'avoir un système évolutif, fiable et efficace pour assurer l'information et les flux messages d'être ininterrompu. Finalement une conception de l'architecture multi-agent dynamique, adaptative et autonome doit être considérée.

CHAPITRE IV
LA CONCEPTION

I. INTRODUCTION

La première partie de notre étude, nous a permis de mieux comprendre ce domaine, nous a mis sur le rail et nous a donné une future vision sur notre système.

Dans ce chapitre on va présenter notre conception du système multi agent ambiant, son rôle et son architecture et notre vision d'utiliser les agents ambiants pour atteindre les mieux résultats.

II. SCENARIO

La première étape à réaliser dans un projet de développement en intelligence ambiante est la description textuelle du scénario de l'application. En effet, la plupart des travaux de développement en ce domaine commencent par cette étape, où les concepteurs présentent l'objectif de leurs applications et décrivent rapidement l'enchaînement des interactions entre l'utilisateur et les appareils.

Notre but à travers cette étape est que d'une part, le concepteur établisse une première idée sur le fonctionnement de son application et que d'autre part il commence à réfléchir à la nature de l'environnement où son application va s'exécuter, ainsi qu'aux différents équipements dont il aura besoin pour la faire marcher.

Notre système multi agents ambiants «qui s'appelle **khochou3i**» va être une application pour les smart phones, le rôle de cette application et de rendre l'appareil en mode silencieux quand l'utilisateur va être en prière , il utilise l'horloge du système et une méthode de calcul du temps de la prière pour définir le temps , et utilise la localisation à partir du système GPS pour définir la localisation de la «mosquée», le problème en Algérie et la plupart des pays arabes et islamiques est que les emplacement non pas identifies dans les systèmes des localisation civile utilisé (exemple : google-maps),et en plus de ce là ,la valeur incertain utiliser dans ces systèmes est 20 mètres alors on ajoutent certain ressources qui vont aider notre application à prendre des décisions plus efficaces en cas d'avoir des information non faibles , cette ressource va être le mouvement de l'utilisateur,

À partir des détecteurs de mouvement, le système va contrôler la position de l'appareil dans la poche de l'utilisateur, quand l'utilisateur est en cours de prier, la position va changer entre la position verticale et horizontale plusieurs fois dans un court laps de temps. Si L'agent contrôleur du mouvement détecte certaines séquences de mouvement il va informer l'agent du service que l'utilisateur peut être en cours de prier, ce dernier va confirmer cette information, ii va envoyer un message à l'agent contrôleur de localisation pour vérifier l'emplacement de l'appareil s'il est dans une mosquée à partir du GPS service. Aussi il va confirmer avec l'agent contrôleur du temps s'il est le temps d'une prière.

Le rôle de l'agent service est la coordination entre les agents pour éviter d'être en position de conflit des informations et le deuxième rôle c'est effectuer les actions du système (mètre l'appareil en mode silencieux).

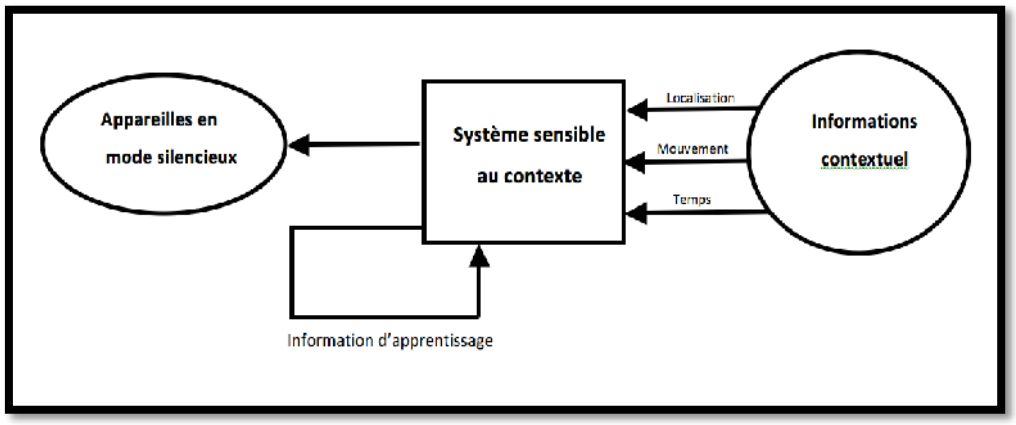


Figure 4.1 : architecture générale du système **khochou3i**

III. SYSTEME DE POSITIONNEMENT ET CAPTEURS DES SMART-PHONES

Pour comprendre le système de positionnement sur un Smartphone, nous donnons un exemple du **Nexus 4**. **Nexus 4** est équipé d'un processeur **quad-coreKraitQualcomm** qui fonctionne à une fréquence de 1,5 GHz et 2 Go de mémoire RAM. Il est approvisionné avec le **GNSS de Qualcomm** module qui prend en charge le positionnement global utilisant GPS et GLONASS satellites. Ces modules sont utilisés pour le positionnement global et sont utiles pour

la navigation à l'extérieur, donc nous ne parlons pas de ces modules dans cette thèse. Dans cette section, nous parlons en détail sur les capteurs de positionnement disponible sur les Smartphones.

1. Systèmes de positionnement

Un système de positionnement tente de déterminer l'emplacement d'un objet dans l'espace. Il permet de localiser un objet avec différentes précisions. Un tel système de positionnement utilise la coordonnée système à 2 ou 3 dimensions. Il existe différents types de système de coordonnées comme le Système de coordonnées cartésiennes, système de coordonnées polaires et système de coordonnées sphérique. Dans le système de coordonnées cartésien [39], les axes sont déterminés en tant que X, Y et Z axe de coordonnées. Un point P dans ce système a X, Y et Z représente P = (X, Y, Z).

Le point P peut également être représenté comme un vecteur de l'origine (0,0,0) au point P. Le système de coordonnées peut être relatif à un "cadre de référence". Un cadre de référence est une petite région de l'espace où l'on prévoit que le système de coordonnées de fonctionner. Les coordonnées peuvent être par rapport à un cadre mondial, local ou corps. Nous expliquons tous les types de cadres que nous utilisons dans notre recherche ci-dessous.

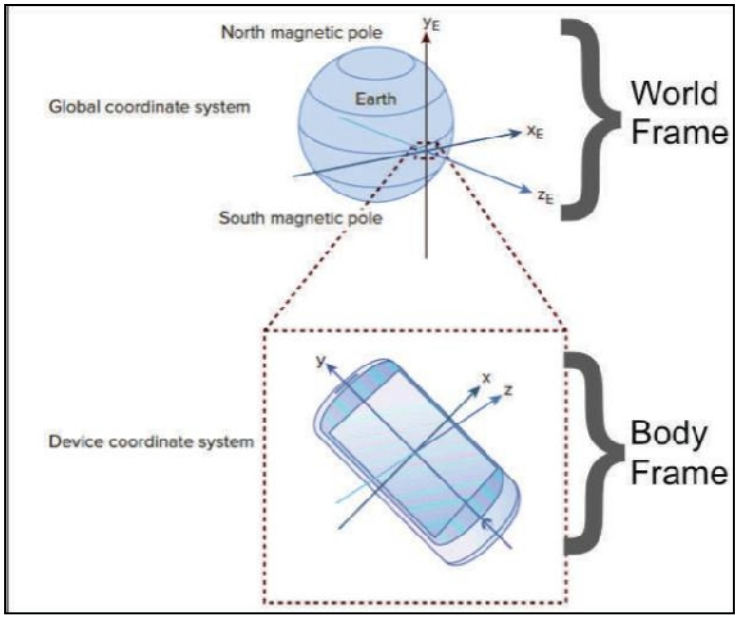


Figure 4.2 : coordination du système andriod [39]

a. Cadre mondial et Global Système de coordonnées

Dans un cadre mondial de tous les capteurs et les méthodes se réfèrent à une orientation absolue par rapport à la terre en utilisant le système de coordonnées global. Dans la **figure 4.2** world frame signifie le cadre mondial. Ce système de coordonnées se réfère au cadre mondial est connu en tant que système global. Dans le système de coordonnées global les axes sont dirigés comme suit [39] :

- YE pointe vers le nord magnétique, qui est le vrai nord.
- XE est de 90 degrés à partir de YE et est parallèle à la surface de pointage à l'est de la Terre.
- ZE souligne l'écart du centre de la terre.

b. Cadre de corps et dispositif de système de coordonnées

Les capteurs Android de positionnement (à savoir, l'accéléromètre, magnétomètre et gyroscope) les valeurs de rapport correspondant à l'appareil ou le corps smartphone. D'où la trame est connue sous le châssis et le système de coordonnées en se référant au cadre de corps en tant que dispositif du système de coordonnées. Un dispositif de smartphone peut avoir deux orientations différentes, la valeur par défaut étant l'orientation portrait et l'autre étant l'orientation paysage [40].

Le smartphone à la **figure 4.2** signifie le dispositif système de coordonnées. Pour une orientation paysage par défaut les axes de l'appareil système de coordonnées sont dirigées comme suit :

• L'axe horizontal X avec des valeurs positives dans la bonne direction.

• L'axe Y est vertical avec des valeurs positives dans le haut.

• L'axe Z est des valeurs positives en face de l'écran.

Dans le dispositif du système de coordonnées est fixé au dispositif de coordonnées. Le système de coordonnées de l'appareil ne change pas lorsque l'orientation du dispositif change. L'origine du dispositif

Système de coordonnées est au centre de l'écran. Les quantités angulaires autour des axes sont données par une matrice de rotation 3-vecteur ou un quaternion qui mappe le dispositif du système de coordonnées du système de coordonnées global comme le montre la **figure 4.2.**

c. Cadre local ou système de coordonnées relatif

Pour les systèmes de navigation intérieure du cadre mondial est très grande et nous utilisons donc quelque chose qui est local à une petite zone. Un tel système de coordonnées par rapport au repère local est appelé le rapport système dans lequel l'origine est fixée au début et toutes les autres positions sont par rapport à cette origine des coordonnées. Habituellement, le X et l'axe Y sont utilisés pour définir la position sur la carte et l'axe Z permet de définir l'altitude au-dessus du sol. Ce système est principalement utilisé pour définir des positions à l'intérieur d'un bâtiment ou d'une carte. Dans ces systèmes, la carte est d'abord tournée pour correspondre au Nord vrai dans le cadre du monde, puis l'axe du bâtiment est aligné sur lui [40].

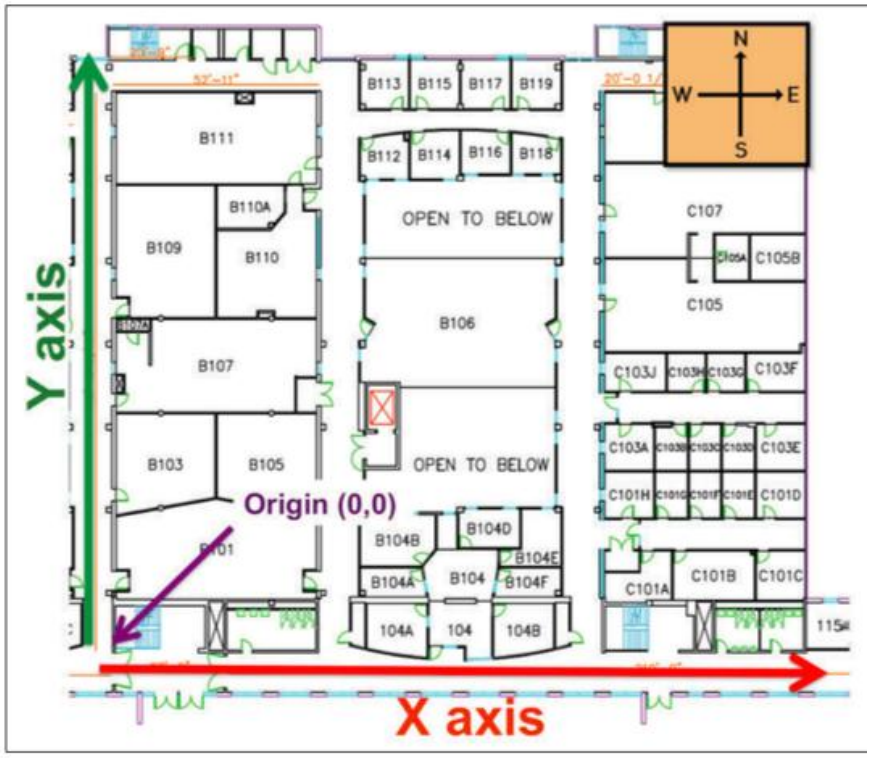


Figure 4.3 : coordination relative locale [40].

2. Capteurs de position

À partir d'Android 1.5 comme un ensemble standard de capteurs et associés API Capteurs [41] a été mis à disposition. Les capteurs standards incluent maintenant l'accéléromètre, gyroscope, magnétomètre, capteur de lumière, capteur de proximité, l'humidité et capteur de pression. Ces capteurs premiers sont des capteurs électromécaniques Micro- (MEMS), qui sont faites sur une petite échelle sur des puces de silicium. Ils ont la capacité de détecter, capturer et analyser le mouvement et normalement contenir une partie qui se déplace ou vibre physiquement. Les principaux capteurs de position sont l'accéléromètre, un gyroscope et de magnétomètre. Figure 2.6 montre un capteur MEMS Gyroscope et Accéléromètre sur le Nexus 4 carte mère [42]. Ces capteurs MEMS sont testés et parés de façon qu'aucun étalonnage du capteur supplémentaire ne soit nécessaire usine.

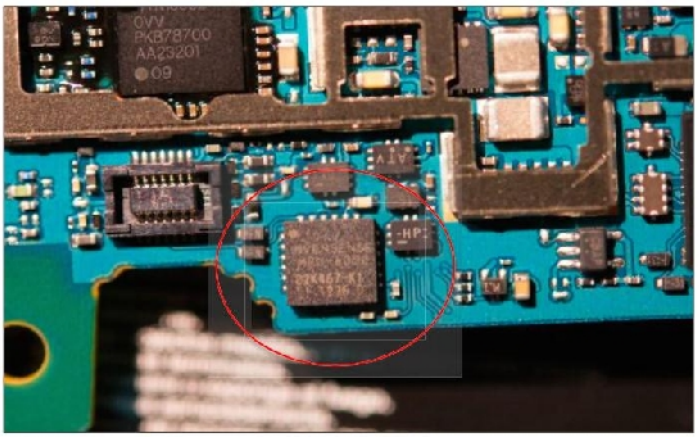


Figure 4.4 : les capteurs des appareils portables [42]

IV. L'ARCHITECTURE D'UN SYSTEME SENSIBLE AU CONTEXTE

Chaque système sensible au contexte peut être présenté avec quatre éléments : un environnement du travail (l'appareil ou la plateforme), des éléments de contexte (l'utilisateur, la localisation… etc.), un adaptateur au contexte et un ensemble d'actions (résultats des actions des agents ambiants) [15].

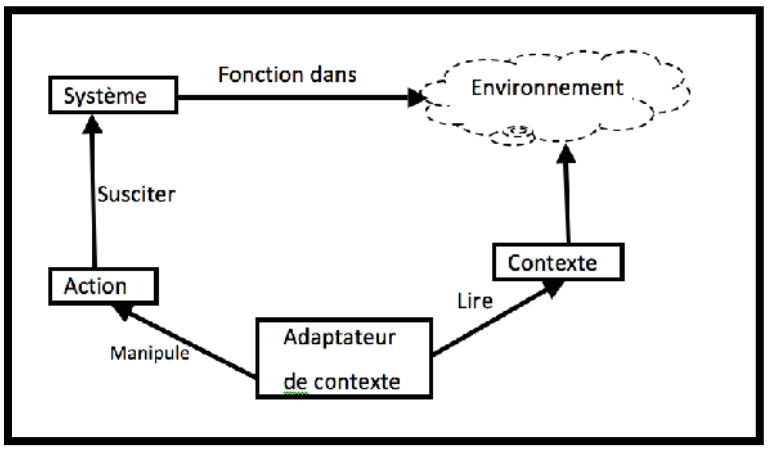


Figure 4.5: composants d'un système sensible au contexte [15]

L'environnement d'une entité représente tout ce qui n'est pas cette entité. Du point de vue du système, l'environnement représente donc tout ce qui lui est extérieur. Concernant l'environnement d'un agent au sein du système, il inclut son environnement auquel se rajoute le reste du système (incluant les autres agents et leurs connexions). Les agents sont considérés des entités, ce qui signifie qu'ils sentent leur environnement et agissent en conséquence, nous pouvons appeler les informations liées à cet environnement comme étant le contexte environnemental de l'agent. A l'exception de leur environnement, les agents sont entourés par et interagissent avec d'autres agents. C'est pourquoi, il existe des interactions et autres valeurs du contexte qui doivent également être prises en compte. Certains agents sont directement reliés aux besoins de l'utilisateur et devrait donc tenir compte de son contexte lorsqu'ils prennent des décisions (exemple : contrôler le mouvement) d'autre agents sont reliés aux mouvements de l'appareil [06].

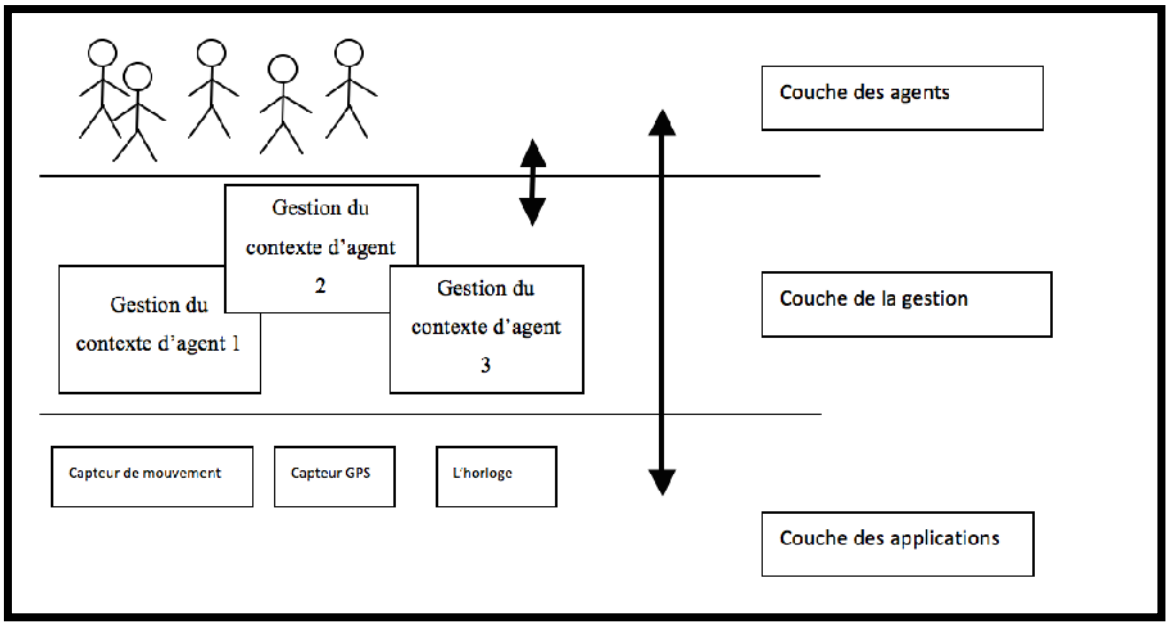


Figure 4.6 : les couches du système sensible au contexte (KHOCHO3I)

Cette approche nous donne la vision fonctionnelle, ainsi nous avançons une proposition d'une architecture de couches. Il se compose de 3 couches : Sources de contexte (couche 0), gestion de contexte (couche 1), couche des agents (couche 2). Les agents interagissent avec la couche gestionnaire, qui est en mesure de répondre aux questions liées au contexte. Au cours de la dernière couche on trouve des applications dédiées au système, qui utilisent la capacité des agents.

La deuxième couche est composée de plusieurs gestionnaires de contexte, un pour chaque agent. Le gestionnaire du contexte est en mesure de traiter les informations de contexte disponible. Un Agent opère à la troisième couche, qui utilise le gestionnaire de contexte pour récupérer la connaissance contextuelle, qui les aidera à décider et à résoudre les problèmes. Pour améliorer leurs capacités, en utilisant les expériences des autres agents, qui peuvent partager entre eux, la connaissance de contexte dans la résolution des problèmes similaires. Aussi, ces agents peuvent stocker ces données, nouvelles et anciennes, dans une base de données pour les réutiliser.

V. LES AGENTS DU SYSTEME : MECANISMES ET ALGORITHMES

Dans notre approche, il y a quatre agents qui agissent avec leurs environnements, un agent pour chaque capteur et un autre agent « agents du service » qui va être le gérant du système

3. Agent contexte

Lorsqu'un utilisateur réalise une action dans une certaine situation (position, mouvement), nous pouvons considérer que cette action est pertinente pour cette situation spécifique. Par conséquent, une des hypothèses de notre étude est que si cette même situation se présente à nouveau, alors la même action sera à nouveau pertinente. Cette connaissance est représentée au travers des agents *contexte*.

Un agent *contexte* associe une situation spécifique, dite "situation de validité", à une action à réaliser sur un effecteur donné. Son objectif est alors de déterminer si la situation courante telle qu'il la perçoit est similaire à sa situation de validité, à laquelle il est considéré pertinent que son action soit réalisée. Ne pouvant pas réaliser lui-même cette action, il la propose à l'agent *contrôleur* en charge du contrôle de l'effecteur sur lequel s'effectue son action.

a. Agent contexte -localisation :

Il utilise le service de localisation GPS de l'appareil pour définir la localisation de l'utilisateur, et quand il détecte qu'il est situé à l'intérieur de la mosquée il envoie un message vers l'agent service pour que ce dernier mette l'appareil en mode silencieux

Algorithme agent-contexte-localisation
Déclaration
Pu : localisation utilisateur
Pm : localisation mosquée
Début
Pm= Nulle Pu= Nulle
Tanque (Pm <>mosquée) et (Pu<>Pm)**faire**
Lire localisation Pu
Lire localisation Pm
Fin tanque
Envoyé un message de localisation valide vers l'agent contrôleur
Fin

b. Agent contexte-mouvement :

A partir du capteur du mouvement de l'appareil, l'agent va contrôler la séquence du changement de la position de l'appareil. Quand l'utilisateur va prier la position de l'appareil change en séquence (vertical, horizontale, vertical, horizontale), si l'agent détecte cette séquence dans un court laps de temps, il envoie un message vers l'agent du service pour l'informer que l'utilisateur est en cours de prière.

Algorithme : Agent -contexte-Mouvement

Déclaration

M= séquencement valide

(Verticale, Horizontale Verticale, Horizontale, Verticale)

T= maxtime = 2 mint

Début

Pour i aller de 0 a T Faire

Lire séquencement de mouvement

Fin pour

M1 ⟵ nouvelle séquencement de mouvement

Si M1=M alors

Envoie un message de localisation vers l'agent contrôle

Finsi

Fin

4. Agent contexte-time :

A partir d'une méthode de calcul du temps de prière choisi, cet agent va contrôler le temps du système et quand il détecte le temps valide, il va envoyer un message vers l'agent service pour l'informer (confirmer) sur le temps de la prière pour mettre le dispositif en mode silencieux

Algorithme agent-contexte-time

Déclaration

T : temps du système

Tp :tableau du 5 case de type time

Debut

Lire la date

Calculer temps du preyer

Tp[1] ⟵ fajer-time

Tp[2] ⟵ doher-time

Tp[3] ⟵ asser-time

Tp[4] ⟵ maghrib-time

Tp[5] ⟵ ichaa-time

Pour **i** *aller de* **1** *a* **5** *faires*

Si Tp[i]= t *alors*

envoie message de validation time-prayer à l'agent contrôleur

Fin si

Fin.

5. Agent contrôleur

L'objectif de l'agent *contrôleur* est de confirmer la situation validité des informations arrivées à partir des agents contexte et décider quelle action réaliser.

Un agent *contrôleur* connaît un ensemble d'agents *contexte* qui lui sont associés et, qui en fonction de la situation courante, lui proposent différentes actions possibles. Il connait aussi un unique agent *service*, à qui transmettre ses décisions d'actions.

Un agent *contrôleur* est capable d'envoyer des messages aux agents *contexte* qui lui sont associés, ainsi qu'à son agent *service* (afin de lui envoyer des demandes d'actions).

Un agent *contrôleur* possède la compétence de déterminer quelle est la meilleure action à réaliser, parmi un ensemble de propositions d'actions. Cette décision se base notamment sur le gain estimé et la confiance qui accompagnent chaque proposition d'action.

Algorithme agent-contrôleur

Déclaration

Pa : ancien validation position

La : ancien validation localisation

Ta : ancien validation prayer-time

Pn : nouvelle validation position

Ln : nouvelle validation localisation

Tn : nouvelle validation prayer-time

Debut

Si Pa ≠ Pn alors Pa ⟵ Pnfinsi

Si La ≠ Ln alors La ⟵ Ln finsi

Si Ta ≠ Tn alors Ta ⟵ Tnfinsi

Tanque (Pa et La ≠ valide) *ou* (Pa et Ta ≠ valide) *ou* (La et Ta ≠ valide) faire

Demande de mise à jour d'état validation

Message de nouvelle d'action (désactiver mode silencieux) envoyé à l'agent service

Fin tanque

Message de nouvelle d'action (activer mode silencieux) envoyé à l'agent service

Fin.

6. Agent service :

Son objectif est de répondre aux sollicitations de l'agent *contrôleur* qui lui est associé lorsque celui-ci lui demande d'affecter un nouvel état, il est aussi capable d'interagir avec les autres agents pour modifier son état.

Il n'appartient pas à l'agent *service*de décidé de modifier l'état de son effecteur, cette décision appartenant à l'agent *contrôleur* qui lui est associé. Le comportement nominal de l'agent *service*sera de rajouter le fait de modifier l'état de son effecteur si l'agent *contrôleur* associé le lui demande. Il prévient aussi l'agent *contrôleur* qui lui est associé en cas de changement d'état de l'effecteur. Le comportement nominal de l'agent *service* est illustré par l'algorithme

Algorithme : Agent -service

Déclaration

état-effecteur : mode silencieux, mode sonnerie

Début

Si réception nouvelle action envoyer par agent contrôleur

Alorsmodifier-état-effecteur (nouvel état)

Fin si

Etat-effecteur ⟵ mise-a-jour ()

Si valeur état-effecteur modifier

Alors envoyer message vers agent contrôleur (état modifier)

Fin si

Fin

VI. CONCLUSION

Dans ce chapitre nous avons présenté la conception de notre système multi-agent ambiant à partir d'un scenario. On a proposé aussi, une architecture du système qui montre notre vu d'utilisation des agents ambiant, on a détaillé notre système multi-agent en présentant l'objectif et la tâche de chaque agent avec leurs algorithmes.

Dans le chapitre suivant, on a voir une description du langage de modalisation « agent-UML », ensuite on l'utilisera pour modéliser notre système.

CHAPITRE V
MODELISATION

I. INTRODUCTION

Dans ce chapitre, nous présentons la modélisation de notre système. Nous avons utilisé le langage AUML qui est une extension de UML spécial pour modélisé les systèmes multi agent, nous avons montré a partir les diagrammes de séquence les interactions entre les agents et les devers contextes,

II. AGENT UML

1. Définition

AUML (Agent Unified Modeling Language) est un langage de modélisation graphique qui est étant normalisé par la Fondation pour les agents physiques intelligents (FIPA) Modélisation Comité technique (Modeling TC) suivant l'idée qu'un agent est "un objet qui peut dire «go» et «no». AUML a été proposé comme une extension de la modélisation unifiée Langage (UML)[43].

AUML joue un rôle très important à stimuler les discussions dans la communauté du génie logiciel Agent-Oriented que expériences, évalue et contribue pour sa spécification.

AUML consiste à utiliser le langage de modélisation UML et de son extension afin de représenter les agents, leurs comportements et leurs interactions entre eux.

L'équipe FIPA(Modeling Technical Committee) est développée le AUML pour but [44]. :

- Faire progresser l'état de l'art de la modélisation orientée agent.
- Permettre aux développeurs de mieux comprendre comment modéliser des applications orientées agent.
- Être en accord avec les spécifications existantes de FIPA (Foundation for Intelligent Physical Agents) et OMG (Object Management Group**) [43].**
- Avoir un aperçu rapide du déroulement d'un protocole. Rappelons que les protocoles d'interaction du modèle FIPA-ACL sont spécifiés en AUML **[43].**

- Représentation du comportement interne à un agent**.**

Les déférents niveaux d'AUML [44] :

- *Niveau1:* Représentez le protocole global (diagramme de séquence, Package, Template).

- *Niveau2:* représentation des interactions entre les agents (diagramme de séquence, activités, collaboration, état)

- *Niveau3:* Représentation interne aux agents (diagramme d'activité, et d'état)

2. Représentation des interactions

Interactions agent peuvent être représentées dans le standard UML en utilisant des diagrammes de séquence

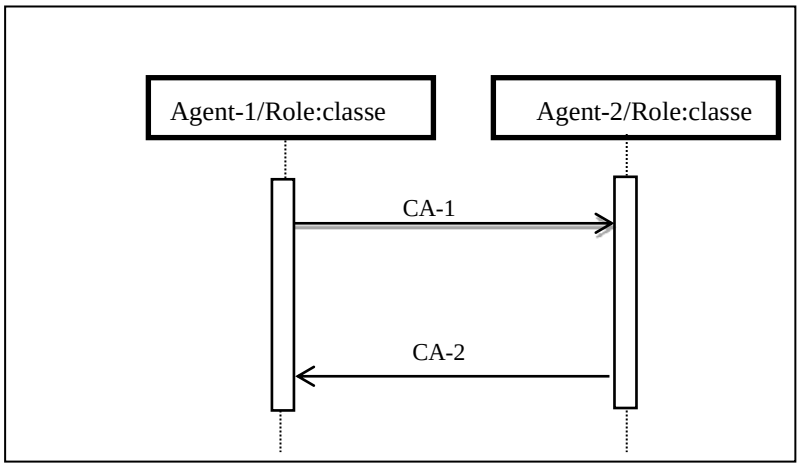


Figure 5.1 : représentation des interactions des agents par le diagramme de séquence

3. Interactions simultanées

UML a été étendu afin de représenter la communication simultanée des actes envoyés de l'expéditeur au destinataire.

- Les actes de communication simultanée de CA - 1à CA - n sont envoyés en parallèle.

- Une sélection de n actes est envoyé en parallèle (zéro ou plus).
- Choix exclusive : un seul des actes de communication est envoyé.

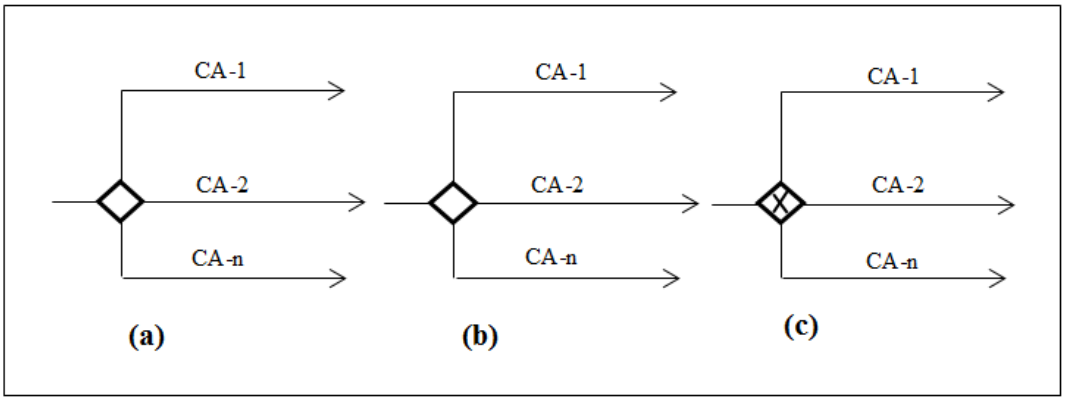


Figure 5.2 : Les actes de communication simultanée [44].

4. Diagramme de classe

Les diagrammes de classes UML peuvent être utilisé pour représenter la vision statique des agents [43].

Une capacité est composée des éléments suivants:

- **Input (Entrée)**

Ce que l'agent doit recevoir en entrée pour réaliser sa tâche

- **Output (Sortie)**

Ce que la capacité génère à la suite des travaux

- **Input contraints (Contraintes d'entrée)**

Contraintes qui sont attendus à tenir avant que l'action définie par la capacité peut être réalisée

- **Output contraints (Contraintes de sortie)**

Contraintes qui sont attendus à tenir dans la situation après l'action définie par la capacité a été réalisée

- **Input-output contraints (Les contraintes d'entrées-sorties)**

Les contraintes qui doivent être titulaires aux situations d'entrée et de sortie

- **Description**

Une description en langage naturel de la capacité

La figure 5.3 , suivante nous représente un exemple d'un agent qui a le rôle de faire des opérations arithmétiques

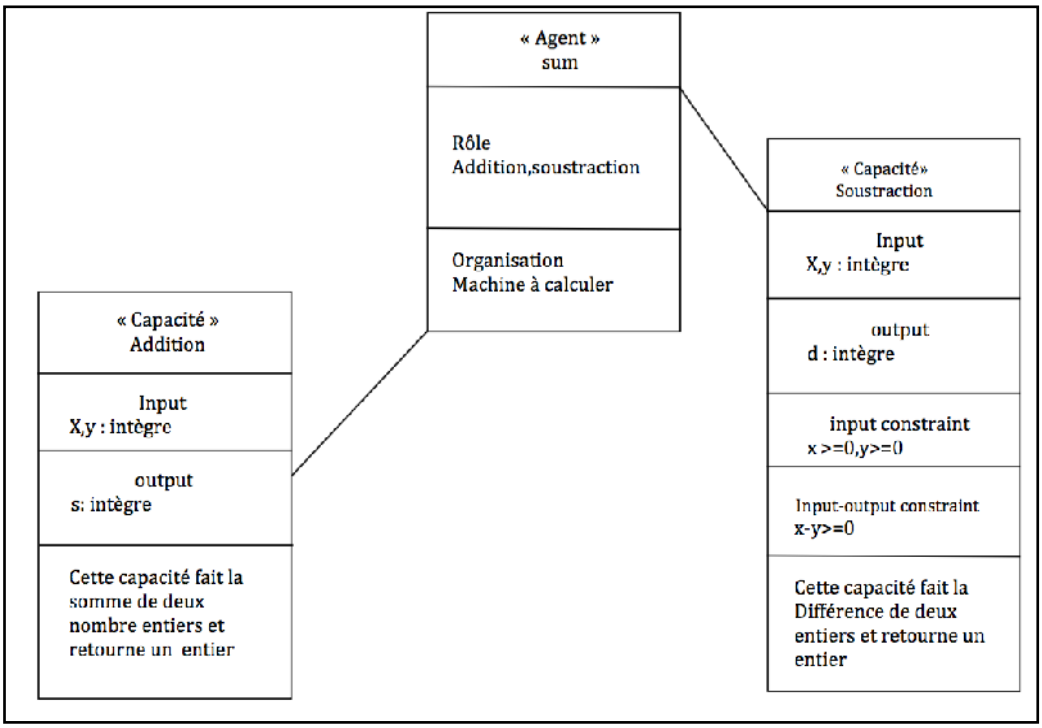


Figure 5.3 : exemple représentation d'un agent par le diagramme de classe « AUML » [43].

III. DIAGRAMME DE SEQUENCE : SCENARIO DE LA LOCALISATION

Diagramme de séquence qui traduit le scénario de la localisation par GPS, l'agent-contexte-localisation envoyer un message à l'agent-contexte-contrôleur qui indique que la position est valide le dernier va envoyer un message à l'agent-contexte-service pour activé le mode silencieux.

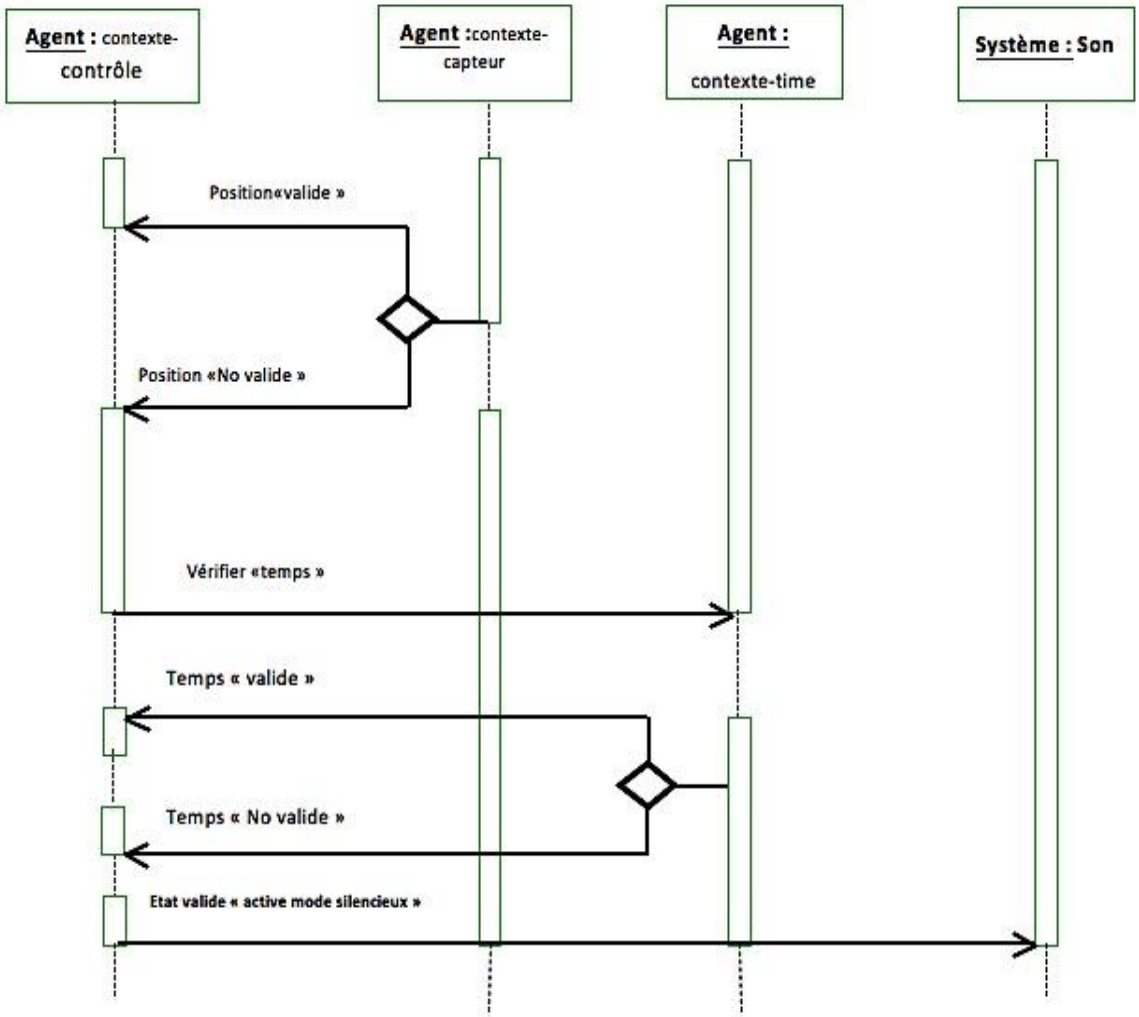


Figure 5.4 : Diagramme de séquence « scénario localisation GPS »

IV. DIAGRAMME DE SEQUENCE : SCENARIO DETECTION DE MOUVEMENT

Diagramme de séquence qui traduit le scénario détection par mouvement, l'agent-contexte-mouvement envoyer un message à l'agent-contexte-contrôleur qui introduit que le mouvement est valide le dernier vérifier que le temps et valide et envoyer un message à l'agent-contexte-service qui va activer le mode silencieux

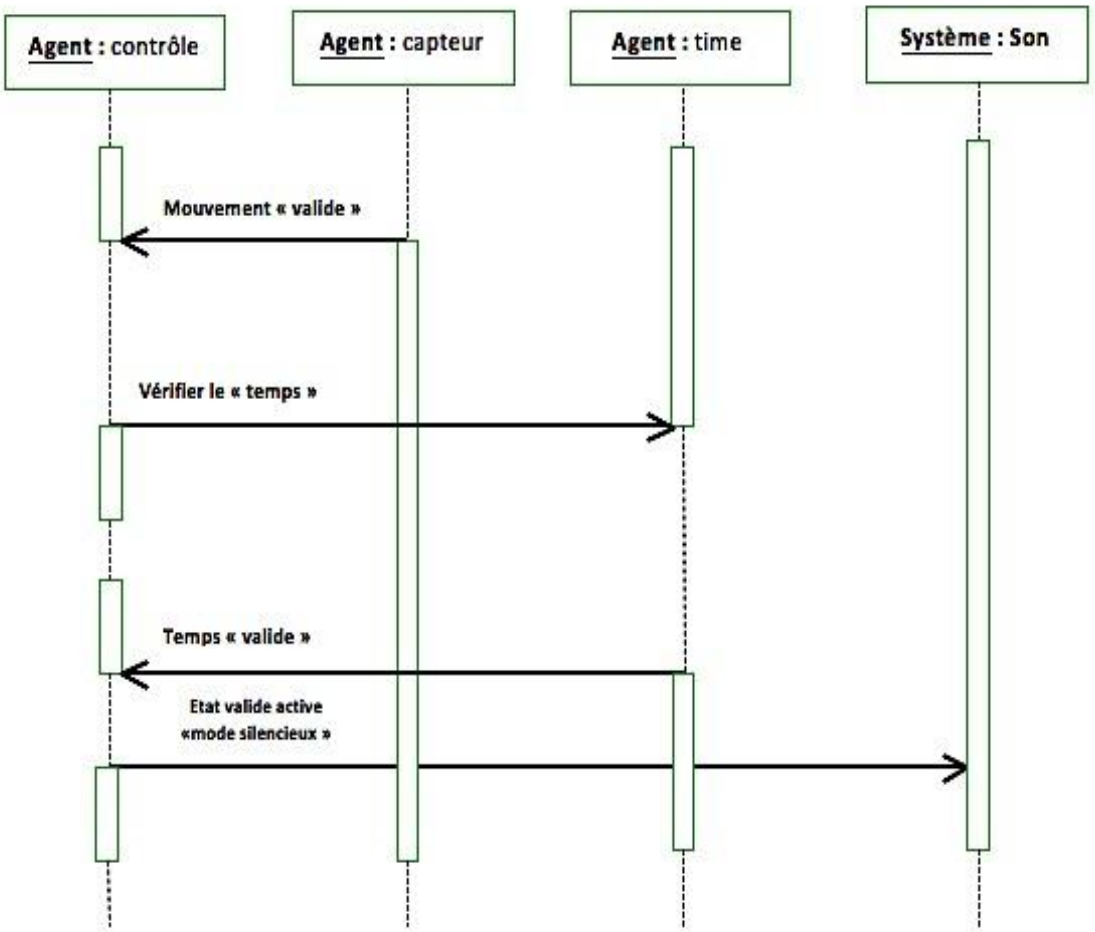


Figure 5.5 : Diagramme de séquence : « scénario détection de mouvement »

V. DIAGRAMME DE SEQUENCE : SCENARIO DETECTION PAR LE TEMPS

C'est le diagramme qui traduit le scénario de détection par temps, l'agent-contexte-time envoyer un message à l'agent-contexte-contrôleur introduit que le temps est valide le dernier va vérifier que la position est valide et pour confirmer l'activation du mode silencieux par l'agent-contexte-service.

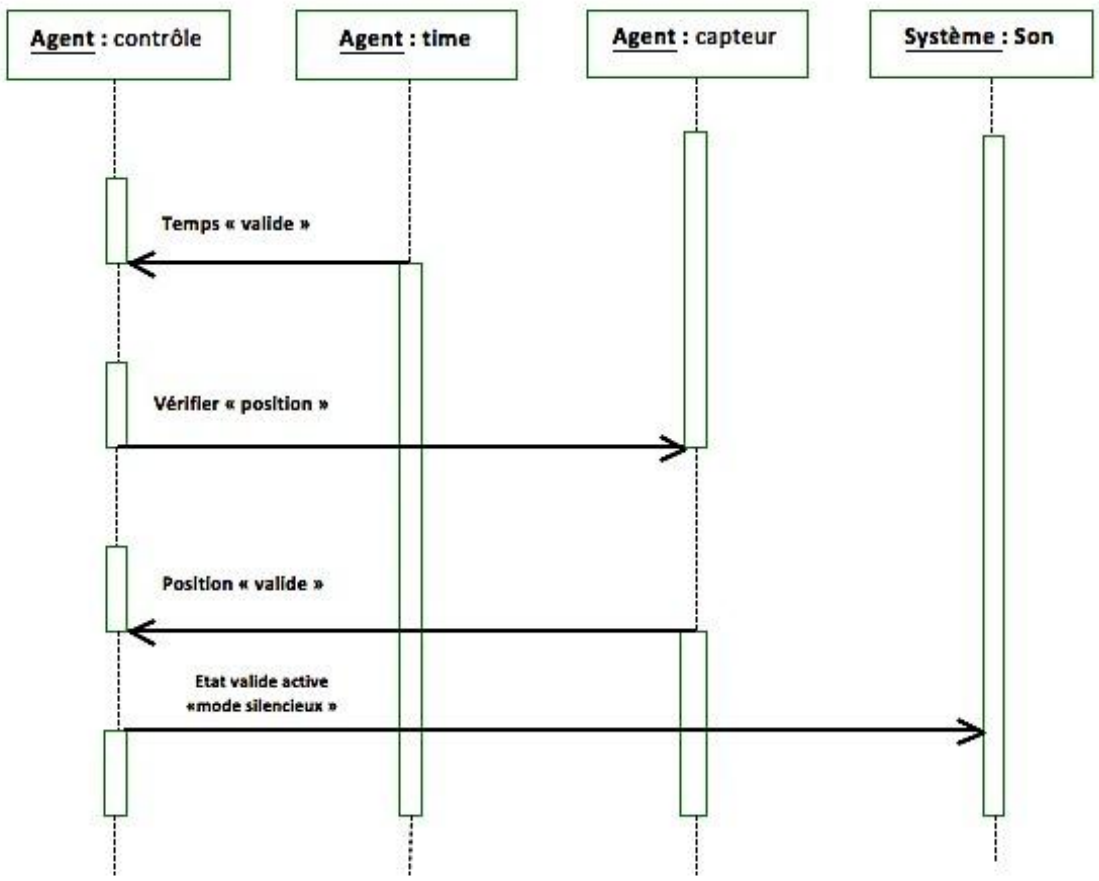


Figure 5.6 : Diagramme de séquence : « scenario détection par le temps »

VI. DIAGRAMME DE CLASSE

Un diagramme de classe est une réalisation d'une modélisation basée sur des composants. Une modélisation à base de composants peut supporter plusieurs types de réalisation. Dans ce travail, nous définissons l'une des réalisations possibles Dans la figure, nous identifions plusieurs classes, la classe « agent-contexte-localisation » utiliser le capteur GPS pour déterminer l'emplacement de l'utilisateur, et envoyer un message de validation a « l'agent-contexte-contrôleur » la classe « agent-contexte-mouvement » utilise le capteur de mouvement pour déterminer les actions de l'utilisateur, et envoyer un message qui vérifier la validité de mouvement a « l'agent-contexte- contrôleur » la classe « agent-contexte-Time » vérifier que le temps de prier et valide et envoyer un message de validation a « l'agent-contexte-contrôleur », la classe « agent-contexte-service » activé ou désactivé le mode sonnerie du diapositif par la réception d'un message de « l'agent-contexte-contrôleur », la classe « agent-contexte-contrôleur » qui contrôle les interaction entre les agents par l'envoi et la réception des messages.

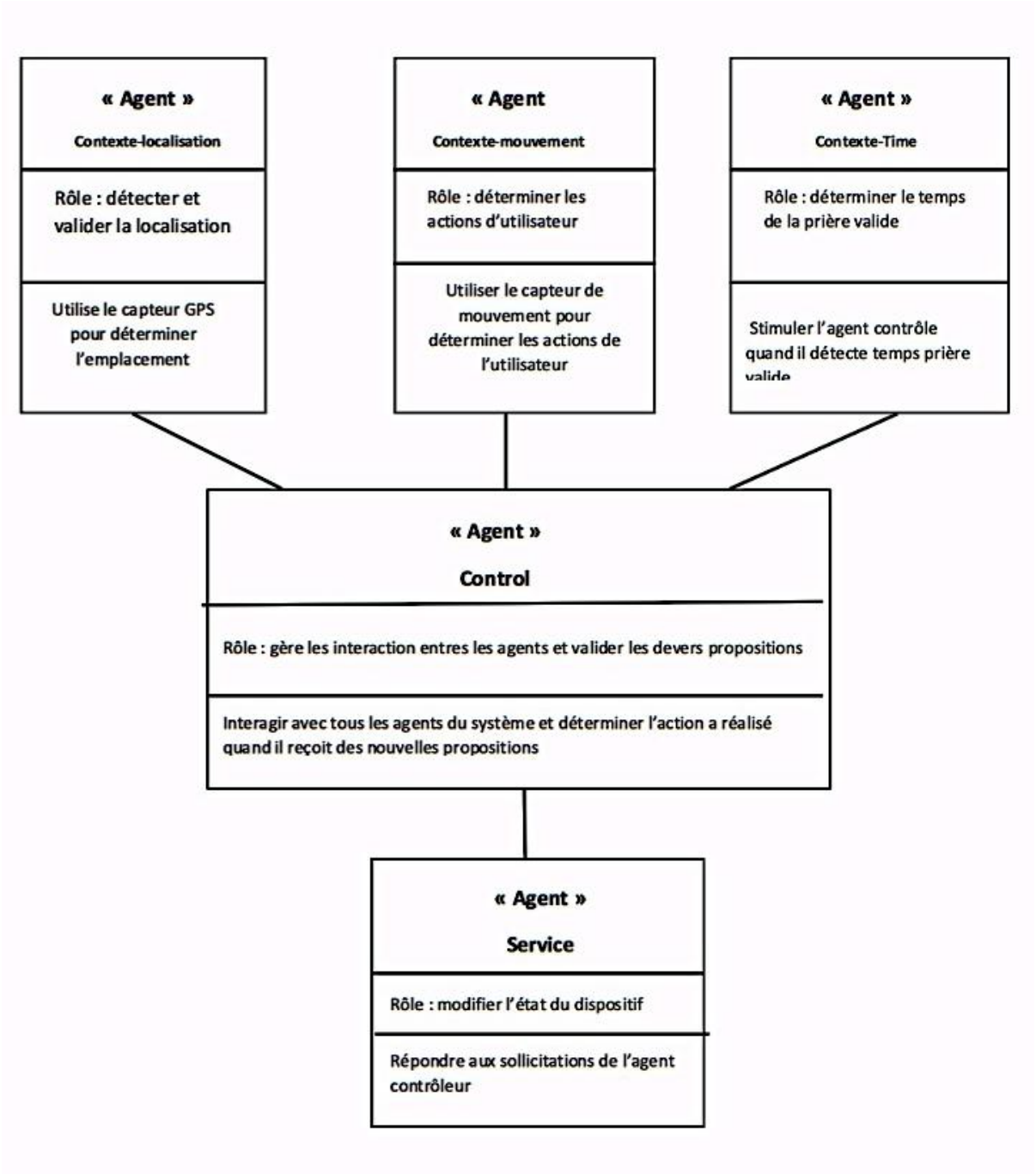


Figure 5.7: Diagramme de classe

VII. CONCLUSION

Dans ce chapitre nous avons présenté une description sur l'AUML (Agent Unified Modeling Language), et une présentation de notre modélisation en utilisent les diagrammes de séquence et de classe en détaillent les différentes interactions entre les agents, et le fonctionnement du système

CHAPITRE VI
IMPLEMENTATION

I. INTRODUCTION

Dans ce chapitre, on va présenter une approche à base d'agent pour l'apprentissage mobile. Nous avons utilisé des agents ambiants afin de profiter les points forts de ce paradigme tels que : autonomie, réactivité, et sociale. Nous avons opté de la disponibilité d'une extension de plateforme pour les agents ambiants, il s'agit de la plateforme Jade-LEAP (Lightweight and Extensible Agent Platform).

On va utiliser l'environnement de programmation L'Andriod Studio qui est présenté par Google pour réaliser les applications mobiles

II. PLATEFORME JADE-LEAP

1. Définition

JADE (Java Agent DEvelopment cadre) est le populaire frame-work open-source qui facilite le développement de Systèmes multi-agents interopérables. Il est l'un des outils les plus largement utilisés pour le développement de systèmes à base d'agents.

L'adoption de JADE dans des environnements mobiles remonte au début des années 2000 avec LEAP (Lightweight and Extensible Agent Platform). Il est intéressant de noter que JADE et JADE-LEAP étaient à l'origine deux plates-formes différentes qui ne sont pas compatibles. Mais l'introduction de la version 4.0 de JADE fusionné JADE et JADE-LEAP dans une plate-forme unique avec une interopérabilité totale [45].

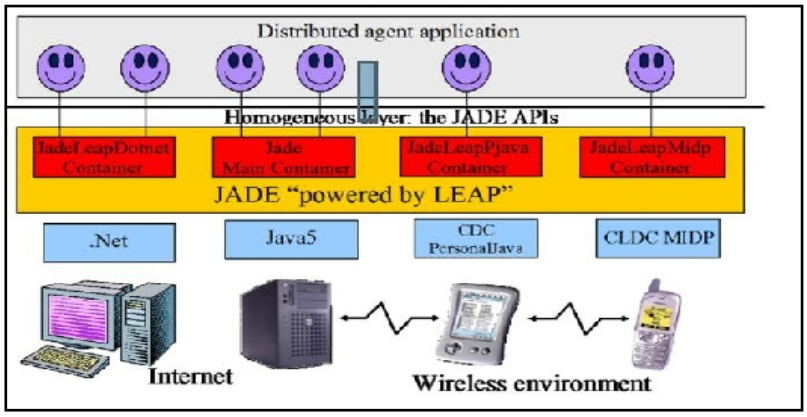


Figure 6.1 :l'environnement d'exécution de JADE-LEAP [45]

2. Motivation pour l'utilisation de la plate-forme JADE-LEAP

Il y a plusieurs plates-formes multi-agents pour les appareils mobiles tels que Le MobiAgent, AgentLight, plateforme d'agent MicroFIPA-OS, et le JADE-LEAP. Nous avons choisi la plate-forme jade-LEAP pour de nombreuses raisons telles que [46] [47] :

- Extension de JADE, qui écrit en Java, et ont des caractéristiques telles que la possibilité d'exécuter plusieurs tâches simultanées (comportements) dans un seul thread Java, correspond bien aux contraintes imposées par les dispositifs ayant des ressources limitées.
- Supporte une variété de périphériques tels que Java MIDP téléphones, PDA.
- La plus petite plate-forme disponible en termes de taille de l'encombrement.
- Open-source.

3. Les composants du JADE-LEAP : LEAP-IMTP

Bien que JADE et JADE-LEAP soient presque identiques à partir d'un point de vue externe, ils sont tout à fait différents à l'intérieur. Un changement important est que l'IMTP. Le IMTP JADE normale est basée sur Java RMI et ce n'est pas adapté pour les appareils mobiles. JADE-LEAP utilise donc un IMTP alternative basée sur un protocole propriétaire appelé JICP (Jade Inter Container Protocole) [46].

Le LEAP IMTP est composé d'un soi-disant « Command Dispatcher » et un ou plusieurs pairs de communication internes (PIC), qu'il est responsable de l'envoi et la réception de commandes en série sur le réseau en utilisant un protocole donné. La **figure6.2** illustre les principales composantes de l'IMTP LEAP.

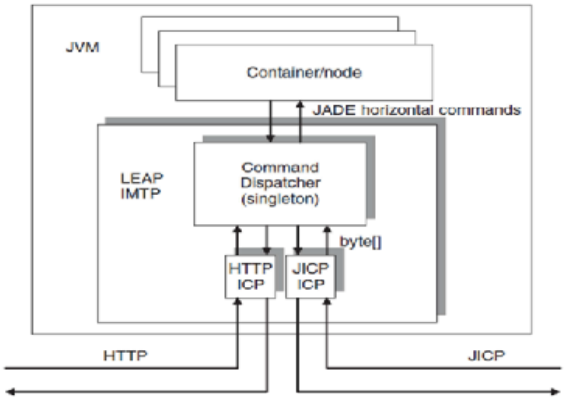


Figure 6.2 : les principales composantes de l'IMTP-LEAP[46]

4. Le mode d'exécution du JADE-LEAP

L'environnement d'exécution JADE peut être exécuté dans deux modes différents. Le mode d'exécution normale *"Stand-alone"*, où un conteneur complet est exécuté sur le host où l'exécution JADE est activée. Le mode d'exécution *"Split"* où le conteneur est divisé en un *FrontEnd* (exécuter sur le périphérique hôte où le JADE runtime est activée) et un *BackEnd* (exécuter sur un serveur distant) reliés entre eux par l'intermédiaire d'une connexion Le tableau suivant résume la façon dont les deux modes d'exécution sont prises en charge dans les différents environnements ciblés par JADE-LEAP [48].

	.NET	pjava	midp	Android
Stand-alone	Suggéré	Supporté	Non Supporter	Supporté
Split	Supporté	Suggéré	Obligatoire	Supporté

Tableau 6.1 : prise en charge des modes d'exécution dans JADE-LEAP [48]

Le mode d'exécution « *split* » simule une *architecture P2P hybride* avec les avantages suivants : connaissance des agents actifs, de leurs services et de leurs capacités; moins de trafic sur le réseau, un mécanisme de registre (souscription) et d'authentification permettant à un agent de reconnaître tous ses pairs.

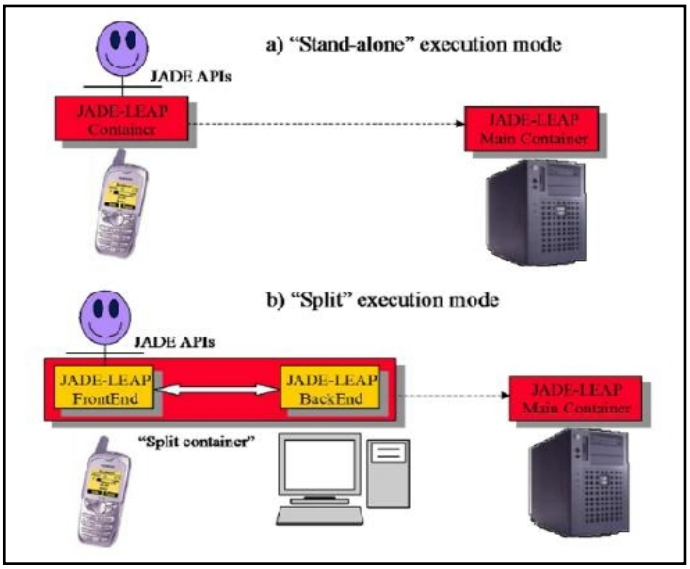


Figure 6.3 : les modes d'exécution dans JADE-LEAP [49]

III. II. JADE-LEAP POUR ANDROID

Le JADE exécution a été initialement conçue pour répondre une large gamme de mobiles. Afin de traiter correctement la mémoire et les limitations de puissance de traitement ces dispositifs. JADE peut être configurée pour s'adapter aux caractéristiques de l'environnement de déploiement [45].

La configuration du LEAP add-on pour les environnements andriod est comme suit:

android—to host JADE agents on devices that supports Android 2.1 (or later);

Android a une approche très particulière au développement des applications et il introduit également un nouveau concept. La première opération pour activer le JADE runtime d'une activité Android est de récupérer un MicroRuntimeServiceBinder objet en utilisant un sous-classe de ServiceConnection (dans le paquet android.content), comme suit :

```
serviceConnection = new ServiceConnection() {
public void onServiceConnected(
ComponentName className, IBinder service) {
// Bind successful
microRuntimeServiceBinder =
(MicroRuntimeServiceBinder) service;      }
public void onServiceDisconnected(
ComponentName className) {
// Bind unsuccessful
serviceBinder = null;
}          };
```

Figure 6.4 : activation du JADE runtime sou la plate-forme JADE-LEAP [49]

Après avoir récupéré l'objet MicroRuntimeServiceBinder il est maintenant possible de démarrer un conteneur JADE, Une fois le JADE runtime est en cours d'exécution, il est possible de commencer un agent comme le montre l'extrait de code ci-dessous :

```
serviceBinder.startAgent(nickname,
className,
new Object[] { getApplicationContext() },
new RuntimeCallback<Void>() {
@Override
public void onSuccess(Void thisIsNull) {
// Agent startup successful
 ...  }
@Override
public void onFailure(Throwable t) {
// Agent startup error
...
}  });
```

Figure 6.5 : création d'un agent sou la plate-forme JADE-LEAP [49]

IV. DEFINITION DE L'ANDRIOD STUDIO

Android Studio (illustré à la figure 6.6) est l'idée pour Android qui a été annoncé en 2013 au (Google I/O developerse vent), et est conçu comme une alternative à Eclipse 1001.

Actuellement Google est lancé la version la plus récent 1.0 du l'andriod studio, ce projet n'est pas encore prêt pour tous end-to-end de développement d'applications Android, mais devrait être prêt à la venue. Android Studio est basé sur l'IDE Java appelé IntelliJ. Qui a nous présenté précédemment Redmine, PyCharm, PhpStorm, WebStorm, ou AppCode. Tout IntelliJ produits partagent le même IDE shell. Malgré que Android-Studio est une nouvel IDE de la marque, il est important de noter que la plupart des compétences IDE de Eclipse appliquent à l'Android-studio aussi bien. La plupart de l'outillage dans Android-Studio est très similaire à Eclipse, tels que les raccourcis, les concepteurs et éditeurs de code. [50]

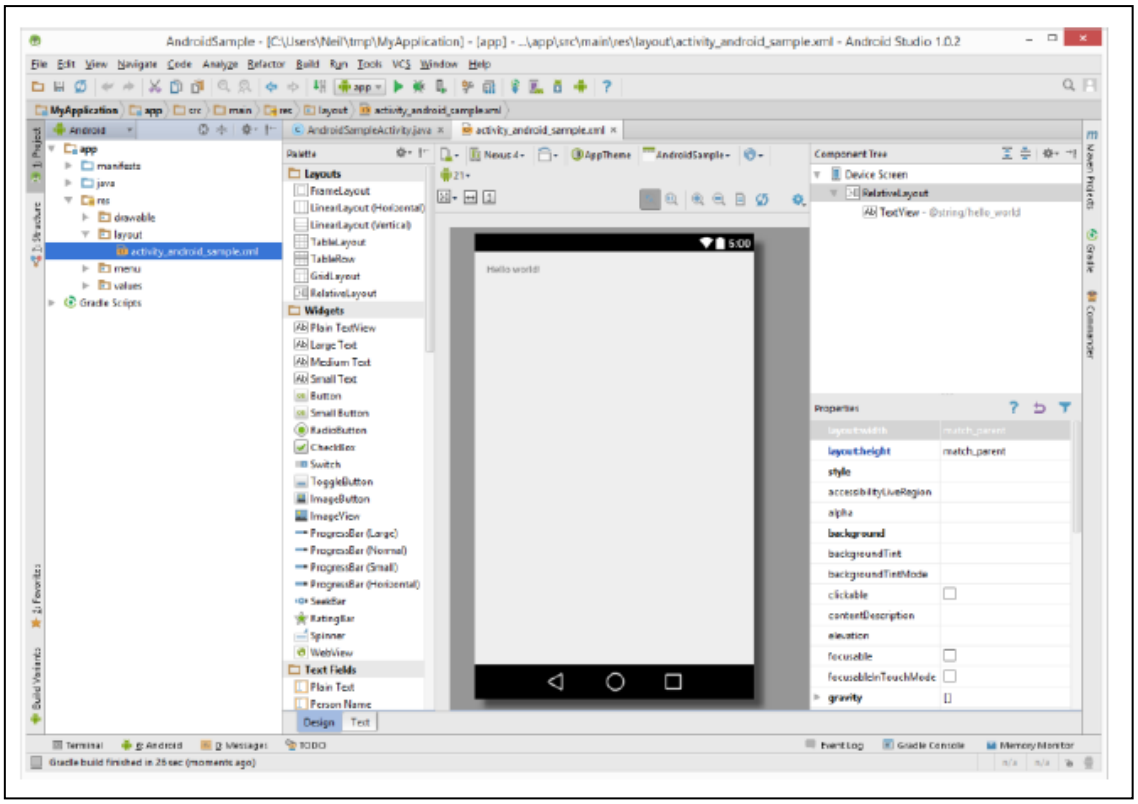


Figure 6.7 : interface de l'Andriod Studio [50]

V. CONCLUSION

Dans ce chapitre nous avons présentés la phase de l'implémentation du notre système sensible au contexte. Nous avons défini la plat forme JADE-LEAP, présenté leur composants, leur mode d'exécution et la façon de l'utilise pour l'environnement andriod ; aussi nous avons expliqué notre motivation d'utiliser le JADE-LEAP, finalement on a présenté la technologie Andriod-Studio qui présenté par Google pour le développement des applications Andriod

CONCLUSION GENERALE

CONCLUSION ET PERSPECTIVES

Notre Projet s'inscrit dans le domaine des systèmes ambiants sensible au contexte, nous avons présenté l'axe de l'utilisation des agents ambiant, on a commencé par une étude des concepts : le contexte, la sensibilité au contexte et les agents ambiants, chacun de ces concepts représente un grand axe de recherche.

Dans cette étude, nous avons essayé de présenter des travaux pertinents au domaine de la sensibilité au contexte et les agents ambiants, ces travaux proposent plusieurs méthodes et modèles des systèmes multi-agents ambiants. D'après l'étude de ces travaux, nous attirons que l'utilisation des agents ambiants, dont ils offrent plusieurs mécanismes et capacités, permet les réalisateurs de résoudre des grands problèmes, qui apparus très difficile à l'utilisation d'autre approche.

Notre travail a mené à la conception du système multi-agent ambiant sensible au contexte « KHOCHO3I », a l'étape de la conception de ce système, nous avons utilisé une architecture des couches, qui nous permettant d'expliquer les composant de défirent niveau du système, et le mécanisme de coopération entre les agents ambiants qui lui compose. Pour bien gérer les agents du système nous avons les effectués en trois catégories : agent service joue le rôle d'un gérant, agents capteurs contrôlent les capteurs du dispositif et agent effecteur réalise les actions du système.

Notre modélisation du système fait à travers le langage AUML qui est une extension de l'UML spécial pour modéliser les systèmes multi agent, à trévère les diagrammes de séquence et cas d'utilisation nous avons illustré les déférentes actions du système.

Finalement pour valide notre étude nous avons utilisé les technologies jade-leap et andriod-studio pour notre l'implémentation

REFERENCES
BIBLIOGRAPHIES

[01] Bikakis A, Antoniou G. Defeasible contextual reasoning with arguments in ambient intelligence. Knowledge and Data Engineering, IEEE Transactions on. 2010 Nov;22(11):1492-506.

[02] BEHLOULI, Nabiha Belhanafi, thèse de 3éme cycle, Université de Nice-Sophia Antipolis, Nice.(France) 2006.

[03] Schilit B, Adams N, Want R. Context-aware computing applications. InMobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on 1994 Dec 8 (pp. 85-90). IEEE.

[04] Kokinov B. A dynamic approach to context modeling. InProceedings of the IJCAI-95 workshop on modeling context in knowledge representation and reasoning. LAFORIA 1995 (Vol. 95, No. 11, pp. 199-209).

[05] Pascoe J. Adding generic contextual capabilities to wearable computers. InWearable Computers, 1998. Digest of Papers. Second International Symposium on 1998 Oct 19 (pp. 92-99). IEEE.

[06] Abowd GD, Dey AK, Brown PJ, Davies N, Smith M, Steggles P. Towards a better understanding of context and context-awareness. InHandheld and ubiquitous computing 1999 Sep 27 (pp. 304-307). Springer Berlin Heidelberg.

[07] Chaari T, Laforest F, Flory A. Adaptation des applications au contexte en utilisant les services web. InProceedings of the 2nd French-speaking conference on Mobility and ubiquity computing 2005 May 31 (pp. 111-118). ACM.

[08] Strassner J, van der Meer S, O'Sullivan D, Dobson S. The use of context-aware policies and ontologies to facilitate business-aware network management. Journal of Network and Systems Management. 2009 Sep 1;17(3):255-84.

[09] Bikakis A, Antoniou G. Defeasible contextual reasoning with arguments in ambient intelligence. Knowledge and Data Engineering, IEEE Transactions on. 2010 Nov;22(11):1492-506.

[10] Schmidt A, Beigl M, Gellersen HW. There is more to context than location. Computers & Graphics. 1999 Dec 31;23(6):893-901.

[11] A. Olaru, thèse de 3éme cycle, Université Pierre et Marie Curie, Paris, (France), 2011.

[12] Henricksen K, Indulska J. Developing context-aware pervasive computing applications: Models and approach. Pervasive and mobile computing. 2006 Feb 28;2(1):37-64.

[13] Kirsch-Pinheiro M, Villanova-Oliver M, Gensel J, Martin H. Context-aware filtering for collaborative web systems: adapting the awareness information to the user's context.

InProceedings of the 2005 ACM symposium on Applied computing 2005 Mar 13 (pp. 1668-1673). ACM.

[14] Amara-Hachimi N, El Fallah-Seghrouchni A. Modélisation d'informations contextuelles pour des agents mobiles sensibles au contexte. In NOTERE (Vol. 6)., 2006.

[15] Y. Elghayam, thèse de 3éme cycle, université Mohamed V- souisi , rabat, (Maroc), 2011.

[16] Schilit BN, Theimer MM. Disseminating active map information to mobile hosts. Network, IEEE. 1994 Sep;8(5):22-32.

[17] Ryan N, Pascoe J, Morse D. Enhanced reality fieldwork: the context aware archaeological assistant. Bar International Series. 1999;750:269-74.

[18] Lieberman H, Selker T. Out of context: Computer systems that adapt to, and learn from, context. IBM Systems Journal. 2000;39(3.4):617-32.

[19] Mari KORKEA-AHO: Context-aware applications survey, 2000. URL http://www. cse.tkk.fi/fi/opinnot/T-110.5190/2000/applications/context-aware. html.

[20] Weiser M. Some computer science issues in ubiquitous computing. Communications of the ACM. 1993 Jul 1;36(7):75-84.

[21] T. Chaari, thèse de3éme cycle, Institut National des Sciences Appliquées, Lyon, (France), 2007.

[22] M. Miraoui, thèse de 3éme cycle, université du Québec, Montréal, (Canada), 2009.

[23] Baldauf M, Dustdar S, Rosenberg F. A survey on context-aware systems. International Journal of Ad Hoc and Ubiquitous Computing. 2007 Jan 1;2(4):263-77.

[24] Indulska J, Sutton P. Location management in pervasive systems. InProceedings of the Australasian information security workshop conference on ACSW frontiers 2003-Volume 21 2003 Jan 1 (pp. 143-151). Australian Computer Society, Inc.

[25] Dey AK, Abowd GD, Salber D. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Human-computer interaction. 2001 Dec 1;16(2):97-166.

[26] W3C. Composite Capabilities / Preferences Profile (CC/PP). http://www.w3.org/Mobile/CCPP .

[27] Henricksen K, Indulska J. Modelling and using imperfect context information. InPervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on 2004 Mar 14 (pp. 33-37). IEEE.

[28] Chen H, Finin T, Joshi A. An ontology for context-aware pervasive computing environments. The Knowledge Engineering Review. 2003 Sep 1;18(03):197-207.

[29] Raibulet C. Facets of adaptivity. InSoftware Architecture 2008 Sep 29 (pp. 342-345). Springer Berlin Heidelberg.

[31] Cernuzzi L, Insfrán Pelozo CE, RODRIGUEZ VIRUEL ML. Requirements Modeling for Multi-Agent Systems. Multi-Agent Systems-Modeling, Control, Programming, Simulations and Applications. 2011:3-22.

[32] Tapia DI, Corchado JM. An ambient intelligence based multi-agent system for Alzheimer health care. International Journal of Ambient Computing and Intelligence (IJACI). 2009 Jan 1;1(1):15-26.

[33] Hofer T, Schwinger W, Pichler M, Leonhartsberger G, Altmann J, Retschitzegger W. Context-awareness on mobile devices-the hydrogen approach. InSystem Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on 2003 Jan 6 (pp. 10-pp). IEEE.

[34] Ayala I, Amor M, Fuentes L. A model driven engineering process of platform neutral agents for ambient intelligence devices. Autonomous agents and multi-agent systems. 2014 Mar 1;28(2):214-55.

[35] Rougemaille S, Migeon F, Maurel C, Gleizes MP. Model driven engineering for designing adaptive multi-agents systems. InEngineering Societies in the Agents World VIII 2007 Oct 22 (pp. 318-332). Springer Berlin Heidelberg.

[36] Babu AK, Nilavu D, Kumar RS. Development of Secured Context Rough Ontology Modelling for Ensuring Data Consistency and Integrity. Journal of Engineering Technology. 2015 Jul;3:105-13.

[37] V. Guivarch, thèse de 3éme cycle, université de Toulouse III- Paule Sbatier, (France), 2014

[38] Barkhuus L. Context information vs. sensor information: A model for categorizing context in context-aware mobile computing. SIMULATION SERIES. 2003 Jan;35(1):127-33.

[39] Cartesian Coordinates,

http://mathworld.wolfram.com/CartesianCoordinates.html

[30] Ferber J, Perrot JF. Les systèmes multi-agents: vers une intelligence collective. InterEditions; 1995.

[40] Milette, Greg, and Adam Stroud. Professional Android sensor programming. John Wiley & Sons, 2012.

[41] Android Sensors API,

http://developer.android.com/reference/android/hardware/Sensor.html

[42] AnandTech Nexus 4 Teardown,

http://www.anandtech.com/Gallery/Album/2435#6

[43] Peres J, Bergmann U. Experiencing auml for mas modeling: A critical view. Software Engineering for Agent-Oriented Systems, SEAS. 2005 Oct.

[44] Cabac L. Modeling agent interaction protocols with auml diagrams and petri nets. Hamburg, Germany. 2003 Dec 9.

[45] Caire G, Pieri F. Leap user guide. TILab, Jan. 2006.

[46] Bellifemine FL, Caire G, Greenwood D. Developing multi-agent systemswith JADE. John Wiley& Sons; 2007 Mar 13.

[47] Mahmoud Q. MobiAgent: A mobile agent-basedapproach to wireless information systems. InAgent-Oriented Information Systems 2001 May (pp. 87-90).

[48] Felder RM, Silverman LK. Learning and teaching styles in engineering education. Engineering education. 1988 Apr 1;78(7) : 674-81.

[49] Bergenti F, Caire G, Gotta D. Agents on the move: JADE for Androiddevices. InProcs. Workshop FromObjects to Agents 2014 Sep.

[50] Craig C, Gerber A. LearnAndroid Studio: BuildAndroid Apps Quickly and Effectively. Apress; 2015 May 11.