UNIVERSITÉ DE BOURGOGNE,
Dijon,

Modèle d'abstraction des données de contexte dans la configuration d'un reseau d'application

MÉMOIRE

MASTER RECHERCHE IMAGE INFORMATIQUE ET INGÉNIERIE

par

Ghislain Loaec

Tuteurs:
Nader Mbarek
Emmanuel Garette

2014

Table des matières

Table des figures

Liste des tableaux

Page

LISTE DES ALGORITHMES

Page

REMERCIEMENTS

Je souaiterais remercier...

RÉSUMÉ

Modèle d'abstraction des données de contexte dans la configuration d'un reseau d'application

Par

Ghislain Loaec

Master Recherche Image Informatique et Ingénierie in Informatique

Unversité de Bourgogne, Dijon, 2014

Nader Mbarek

L'objectif fondamental de l'informatique ubiquitaire est de faciliter l'utilisation de l'ordinateur. Cela passe par extraire le maximum de bénéfices de l'environement numérique. Les défaillances logicielles deviennent monnaie courante à mesure que les systèmes informatiques et leur complexité continuent de croître. Le problème réside principalement dans l'absence de standards ou de modèles réutilisables pour la gestion des informations de contexte.

Chapitre 1

État de l'art

1.1 Introduction

La configuration des composantes logicielles impose un coût majeur dans l'administration d'un système. Des erreurs de configuration peuvent se traduire par des vulnérabilités en termes de sécurité, de sévères pertubations dans le fonctionnement de la brique logicielle, ou purement et simplement provoquer un déni de service. La prise en considération du contexte pourrait permettre une abstraction partielle ou complète de cette couche très technique et extrêmement pénible à configurer.

Un système sensible au contexte doit être capable de mimer la capacité humaine à reconaître et exploiter l'information implicitement présente dans l'environement. Cela implique une configuration dynamique de chacune des composantes de l'architecture, de manière à pouvoir ajuster leur comportement respectif en fonction de la situation. Identifier l'activité humaine est un défi, il est essentiel que les applications opèrent en transmettant l'information appropriée au bon endroit et au bon moment par inférence de l'intention des utilisateurs. L'infomatique sensible au contexte est un paradigme dans lequel les application peuvent découvrir et tirer profit d' informations de circonstance telles que la position actuelle, l'heure de la journée, les personnes et périphériques dans l'environement et leurs activités.

Dans ce mémoire, nous aborderons les principes communs à chacunes des architectures existantes, desquels nous detaillerons le framework conceptuel dérivé (!) par couches. Nous présenterons un certaine variété d'intergiciels et d'infrastructures reconnus pour faciliter la configuration d'applications et de services basés sur le contexte.

1.2 Background

De nombreux débats ont eu lieu sur .. Alors que la plupart des gens comprènent de manière tacite ce qu'est le contexte, ils le trouve par ailleurs particulièrement difficile à élucider.

"... toute information pouvant être utilisée pour caractériser la situation d'une entité.
Une entité peut être une personne, un lieu ou un objet considéré pertinent dans l'interaction entre un utilisateur et une application, notamment l'utilisateur et l'application eux même." [1]
Gregory D. Abowd, Christopher G. Atkeson, Jason Hong, Sue Long, Rob Kooper, and Mike Pinkerton. Cyberguide : a mobile context-aware tour guide. Wireless Networks, 3(5) :421–433, October 1997.

FIGURE 1.1: Le context défini par Abowd et. al.

Cette définition rend la tâche plus facile à un développeur d'application pour énumérer le contexte pour un scénario d'application donné. Si un fragment d'information peut être utilisé pour caractériser la situation d'un participant dans une quelquonque interaction, alors cette information appartient au contexte.

1.3 Vue d'ensemble sur le contexte

"Le contexte est efficace, seulement lorsqu'il est partagé." [13]
Terry Winograd. Architectures for Context.
Human-Computer Interaction, 16(2) :401–419, December 2001.

FIGURE 1.2: Winograd a propos du contexte

Pour s'assurer que le contexte soit partagé, il doit d'abord être receilli et rigoureusement traité. Cela implique qu'un système sensible au contexte doit être en mesure de comprendre ce qu'est le contexte avant d'aller à la recherche de ces informations et de pouvoir les catégoriser.

1.3.1 Classes de contexte

Schilit et. al. proposent la classification suivante des informations de contexte :

- **Contexte Informatique** - Connectivité réseau, bande passante, and ressources à proximité telles que des imprimantes, des affichages ou des postes de travail.
- **Contexte Utilistateur** - Le profil utilisateur, sa situation géographique, sa situation sociale actuelle et les individus qui l'entourent.
- **Contexte Physique** - L'éclairage, le niveau de bruit, les condition de circulation ou la température.

Chacune de ces catégories contiennent une richesse d'informations pertinantes pour le système sensible au contexte. Elle ne peuvent cependant pas être traitées de manière isolée pour pouvoir en extraire le meilleur. L'intention du système sensible au contexte est de rassembler et de fusionner ces informations pour aboutir à une vue d'ensemble de la situation. Une fois le contexte mis en tampon ou en base, le système doit alors filtrer les informations pertinantes pour l'utilisateur, dans le moment présent.

Les informations de contexte peuvent alternativement être subdivisées en 2 catégories bien distinctes : contexte virtuel ou physique.

1.3.1.1 Contexte virtuel

Le contexte virtuel inclut la version du système d'exploitation, les possibilités d'interface, la technologie en charge de l'accomplissement des communications, les emails envoyés et reçus, et les documents édités.

1.3.1.2 Contexte physique

Les contexte physique d'un autre coté peut être la présence d'une autre entité, qu'elle soit utilisateur ou périphérique, la proximité d'un imprimante en particulier, une indication que l'utilisateur est debout, en train de marcher ou assis ou les conditions météorologiques actuelles. En d'autres termes, le contexte physique peut être défini comme toute donnée aquierable par le biais d'une sonde.

1.3.1.3 Contexte historique

Les contextes mémorisés au cours d'un certain laps de temps. Cette information est considérée très utile, mais n'est que très rarement utilisée, sauf pour les applications mobiles. Le système doit être en mesure d'estimer les informations valant la peine d'être conservées. Cette évaluation est excessivement coûteuse et nécéssite donc des algorithmes très performants.

1.3.2 Les caractéristiques des informations de contexte

Les chercheurs l'université de Queensland ont classifié quatre caractéristiques majeures : [3]

1. **Les informations de contexte présente une gamme de caractéristiques temporelles**
   L'information de contexte est d'hors et déjà catégorisée selon l'environement auquel il appartient : virtuel ou physique ; elle peut en outre est subdivisée selon un critère de temporalité :

– Information statique : toute information apparentée à l'environement de l'utilisateur qui ne varie pas.
– Information dynamique : l'information accumulée continuellement, fréquemment et automatiquement.

De plus, l'information de contexte passé semble indispensable pour la compréhension de l'état global de l'environnement.

2. **L'information de contexte n'est pas parfaite**

   Cela considère la validité du contexte, majoritairement concernant les informations de contexte dynamique. La vitesse et la fréquence à laquelle l'information varie soulève de sérieuses raisons de douter de sa solidité. Ce "délai entre la production et l'utilisation de l'information de contexte" [3] est une préoccupation non-négligeable. D'autres sources d'inquiétudes quant au bien-fondé de l'information de contexte incluent la fiabilité des informations fournies par les producteurs de contexte : défaillance d'un capteur, chemin rompu entre les producteurs ou n'importe quelle source qui fournirait une information erronée ou désuette.

3. **L'information incarne un grand nombre de représentations**

   Les données brutes recueillies depuis l'environemment physique et virtuel peuvent prendre de nombreuses formes et doivent être traitées pour se mêler à d'autres informations de contexte. La probabilité pour qu'un système sensible au contexte obtienne un taux de succès de 100% lors d'une "capture des relations existantes entre les représentations alternatives" [3] de l'information et celle apte à la situation courante est quasi-nulle.

4. **Les informations de contexte sont très fortement correlées**

   L'information contextuelle provenant d'une origine particulière peut avoir un lien très étroit avec sa source, si bien qu'elle est dépendante de l'origine. Le contexte peut ne pas être fiable "là où les caractéristiques de l'information dérivée sont intimement liées aux propriétés de l'information dont il est issu." [3]

1.3.3 Système sensible au contexte

Un système est dit sensible au contexte s'il utilise le contexte pour fournir des informations pertinentes et/ou des services à l'utilisateur, où la pertinence dépend de la tâche de l'utilisateur. Les systèmes sensibles au contexte peuvent être implémentés sous plusieures formes, mais pour accomplir cet objectif, le système doit d'un manière générale :

– Recceuillir l'information
– Sérialiser cette information
– Fusionner l'information pour générer un contexte de plus haut niveau
– Prendre automatiquement des mesures basées sur l'information recceuillie

– Rendre l'information disponible à l'utilistateur, dans l'immédiat, dans le futur ou au moment approprié pour améliorer et aider à la completion de la tâche de l'utilisateur.

Les chercheurs du Context Toolkit à l'université de Berkley proposent 3 fonctionnalités qu'une application sensible au contexte doit impérativement supporter : [5]

1. Présentation de l'information et des services à l'utilistateur
2. Éxécution automatique d'un service pour l'utilisateur
3. Etiquetage de l'information de contexte pour une extraction ultérieure

Les développeurs du systems Kimura ont conçu des composants distribués destinés à un système sensible au contexte global. Ces composants se divisent en trois classes [12], qui sont les suivantes :

1. Aquisition du contexte - Le système récupère l'information de contexte et l'ajoute dans un dépôt dédié.
2. Interprétation du contexte - Les sytème convertit l'information recceuillie en un contexte de travail.
3. Interaction de l'utilisateur - Le système affiche le contexte de travail à l'uilisateur.

1.3.4 Recceuillir l'information

Certaines informations de contexte sont explicitement données au sytème, comme le nom de l'utilisateur, son âge, son addresse email, les variables d'environemment or le registre. D'autres informations physiques primitives comme la lumière, la température ou des lectures de pression peuvent être acquises par l'intérmédiaire d'un capteur.

La situation géographique et l'identité sont les deux fragments de contexte les plus fréquemment détéctés.

> "Les capteurs ne sont pas toujours précis ou fiables à 100%,
> tout particuièrement s'ils sont à usage unique. Le processus de
> récupération de l'information doit être tolérent à la défaillance
> protentielle d'un capteur. Toutes les informations recueillies doivent
> être assujetties à des contrôles de validité pour vérifier leur exactitude.
> La fusion des capteur est un moyen de remédier à cette difficulté." [10]
> *Albrecht Schmidt, Michael Beigl, and Hans-W Gellersen.*
> *There is more to context than location. Computers & Graphics, 23(6) :893–901, 1999.*

FIGURE 1.3: La fiabilité des capteurs par Schmidt

Prenez l'exemple d'une centrale éléctrique, possédant de nombreuses sondes de témpérature implantées à diverses endroits d'un reacteur nucléaire. Le système doit prendre la déscision de refroidir plus ou moins généreusement en fonction de ces informations sondées. La sous-estimation du contexte de température aurait des conséquences catastrophiques. Il semblerait judicieux de simplement calculer un valeur moyenne et/ou d'écarter les valeurs reportées qui manifestemment diffèreraient trop des autres mesures. Cette méthode éviterait des prises de mesures drastiques du système sesnsible au contexte dans le but de corriger ce qu'il considère comme une variation de température, mais qui en réalité n'est que le résultat d'une panne de capteur.

1.3.5 Récupérer l'information

- Modèle "Push" - La source de contexte recupère l'information avant même qu'elle soit requise. Cela améliore indéniablement les performances, mais nécéssite une consommation des resources fréquente et considérable pour une information qui pourrait bien ne jamais être exploitée.
- Modèle "Pull" - Collecte l'information de contexte au moment opportin. Cela autorise le recueil d'information à la demande, mais expose le système aux latences réseaux et aux indisponibilités potentielles de certains services.

La méthode de d'acquisiton des données de contexte est très importante lorsque l'on conçoit un système de cette nature, puisque c'est ce qui prédéfinit le style architectural du système. Chen (2003) [4] présente trois approches différentes pour acquérir l'information de contexte.

1.3.5.1 Capteur en accès direct

Cette approche est souvent utilisée dans les periphériques avec capteurs intégrés. Le logiciel client récupère l'information désirée directement depuis les sondes, i.e., il n'y a pas de couche supplémentaire pour obtenir et traiter les données du capteur. Les pilotes pour les capteurs sont raccordés à l'application, donc cette méthode de couplage étroit n'est utilisable que dans des cas très rares. Par conséquent, elle n'est pas adapté pour les sysèmes distribués.

1.3.5.2 Infrastructure intergicielle

L'approche intergicielle introduit un architecture en couches avec l'intention d'abstraire les détails de bas-niveau du processus de détection. La méthode est équivalente au modèle client-serveur, plus flexible que le widget puisqu'elle favorise l'indépendance de chacunes des composantes du système. Chaque composante doit être en mesure d'effectuer les opération suivantes : établir des connexions, envoyer et recevoir des messages et gérer les erreurs. Cet modèle est plus complexe de manière

significative, mais l'approche est plutôt aisée puisqu'elle support un grand nombre de périphériques et d'applications pas l'utilisation de normes de codage et des protocoles réseaux standards.

1.3.5.3 Serveur de contexte

Cette approche distribuée élargie l'infrastructure basée sur les intergiciels en introduisant un gestionnaire d'accès distant. Les données recceuillies par les capteur sont déplacées vers ce dit "serveur de contexte" dans le but de faciliter les accès currents.

1.3.6 Les architectures de gestion de contexte

Winograde (2001) [13] décrit trois modèles différents afin de coordiner les proccessus et composantes multiples :

- **Widgets** L'objectif clé du widget est distinguer l'application du processus d'acquisition de contexte, de manière d'abstraire la complexité que représente le recueil et la gestion de l'information de contexte. Le widget est considéré comme un médiateur qui transmet exclusivement des informations pertinentes à l'appplication. Un widget de contexte fonctionne totalement indépendament de l'application, cet qui permet à plusieures applications d'en faire usage simultanément. Le widget est notamment responsable de l'entretien d'un historique complet du contexte sondé au fil du temps. C'est le modèle le plus répandu.
- **Services réseaux** Cette approche plus flexible, comme l'argumente Hong and Landay (2001) [8], ressemble à l'architecture de serveur de contexte. Au lieu d'un gestionnaire de widget centralisé, des techniques sont utilisées pour découvir les services réseaux dans l'infrastructure. Cette approche basée sur les services n'est aussi performante que l'architecture basée sur les widgets à cause de la complexité des composants orientés réseaux, mais fournit une certaine robustesse.
- **Tableau noir (Blackboard)** En contraste avec la vue orientée processus du widget et le modèle orienté services, le tableau noir représente un approche orientée sur les données. Cete approche donne un focus tout particulier aux données, correspondant des motifs spécifiques dans les données. Dans ce modèle asymétrique, les processus envoient de messages dans un média partagé, le dit "tableau noir", et souscrivent à recevoir des notifications lorsque certains évenements se produisent. Les avantages de ce modèles résident dans la simplicité de configuration et d'ajout de nouvelles sources de contexte.

1.3.6.1 Critères d'arbitrage

Afin de définir au mieux quel modèle choisir dans la mise en place d'un système sensible au contexte, il est avisé de considérer les caractéristiques suivantes :

- **Efficacité** : accélérer le débit de l'information compte tenu de la bande passante et de la latence causée par l'explosion du nombre d'applications et de périphériques dans le réseau.
- **Effort de configuration** : compte tenu de la quantité variable de composants, effectuer des changements dans l'état de la configuration, sans engendrer des pertubations ou mettre le système en échec, n'est pas une tâche fastidieuse. Le modèle doit s'assurer que l'édition est sans danger.
- **Robustesse** : Le degré auquel le système peut faire face à la défaillance
- **Simplicité** :

  > un système qui nécéssite une compréhension avancée des méchanismes qu'il implémente pour faire l'usage, ne sera utilisé que par ceux qui auront le dévouement et la motivation de le maîtriser [13]

- **Extensibilité** :

  > Les services supportant la notion générale de contexte doivent être facilement extensible pour accueillir toute nouvelle source d'information de contexte non-anticipée. [7]

Ces critères peuvent être utilisés pour comparer et contraster les différents modèles de gestion de contexte présentés précédemment.

> Le modèle widget a lien très étroit avec les composantes système, ce qui en fait le modèle de plus efficace dans certaines circonstances.
> Il souffre malheureusement d'une configuration très complexe et est impuissant face à l'échec. Le modèle d'infrastructure, constitué de composants indépendants, est très convoluté, mais cela ne semble pas etre un facteur de résignation pour de nombreux développeurs. Toutefois, la configuration est simple et le système est robuste, ce qui compense le critère négatif de difficulté de compréhension. Pour finir, le modèle tableau noir avec des composants très faiblement liés présente des problèmes d'efficacité, mais il reste simple, robuste et très facilement configurable. [13]
> *Terry Winograd. Architectures for Context.*
> *Human-Computer Interaction, 16(2) :401–419, December 2001.*

FIGURE 1.4: Comparaison des différents modèles de gestion de contexte

1.3.7 Représentation du contexte

Il existe plusieures manières de modéliser l'information de contexte :

- **Clé-Valeur** : La structure de données la plus simple pour modéliser le contexte. Elle est très fréquemment utilisées dans les framework de services, où les paires clé-valeur servent à décrire les aptitudes d'un service.
- **Langage de balisage** : Stucture de données consisté de balises avec des attributs et des contenus associés.
- **Graphique** : Langage de Modélisation Unifié (UML), extension de la Modélisation Role Objet (ORM) par le contexte.
- **Orienté objet** : Exploite tout la puissance de modélisation objet : l'encapsulation, la réutilisabilité, l'héritage. Les approches existantes utilisent des objets variés pour représenter différents types de contextes, et encapsule les détails du traitement et de la représentation du contexte.
- **Logique** : Haut degré de formalité. Typiquemment, des faits, des expressions et des règles sont utilisés pour définir le modèle de contexte.
- **Basé sur les ontologies** : Représente une description des concepts et des relations. Cet outil est très prometeur pour modéliser l'information contextuelle, grâce à son expressivité élevée et très formelle et les possibilités d'appliquer des techniques de raisonemment propres aux ontologies.

La conclusion de l'évaluation présentée par Strang et Linnhoff-Popien [11], basé sur six critères d'exigences, montre que les ontologies incarnent de loin le modèle de plus expressif et comblent la plupart de ces exigences.

Korpipää et al. [9] présente un ensemble de conditions requises et d'objectifs dans la conception d'une ontologie de contexte :

- **Simplicité** : Les expressions et relations définies doivent être les plus simples possible pour ne pas décourager les développeurs d'appplications.
- **Flexibilité et extensibilité** : L'ontologie doit supporter l'ajout de nouveaux éléments de contextes et de relations.
- **Généricité** : L'ontologie ne doit pas être limitée à un seul type d'atome de context, mais au contraire supporter un maximum de formats pour l'information de contexte.
- **Expressivité** : L'ontologie doit permettre de décrire le plus d'états de contexte possibles et de manière la plus détaillée qu'il soit.

Un atome de contexte peut être decrits à l'aide de quelques attributs seulement. Les deux attributs les plus évidents sont :

- **Type du contexte** : Catégorie dans laquelle s'incrit le contexte, qu'il s'agisse d'une donnée de température, temporelle, de vitesse ou d'accélération, etc. L'information de genre peut être utilisée en tant que paramètre dans une requête ou une souscription pour certain type de contexte.
- **Valeur du contexte** : Les données brutes recueillies par une sonde. L'unité dépend très étroiement du type de contexte et du capteur dont l'information provient, e.g., degré Celcius, kilomètres par heure, mégabit, etc.

Le type et la valeur du contexte ne sont néanmoins pas des informations suffisantes obtenir un système sensible au contexte opérationel. En effet, d'autres attributs supplémentaires doivent être mis en oeuvre dans un atome de contexte pour améliorer sa concision :

- **Horodatage** : Valeur de type date/heure réprésentative de l'instant à laquelle l'information de contexte a été capturée. Elle est requise dans l'élaboration d'un historique du contexte ou même le traitement des conflits de détéction.
- **Source** : Comment l'information a-t-elle été reccueillie ? Dans le cas d'un capteur matériel, l'identifiant de la sonde doit être conservé pour permettre à une application de préférer les information provenant de ce capteur en particulier.
- **Confiance** : L'incertitude confiée à l'information sondée. Toutes les sources de données ne fournissent pas une information juste, les données de situation géoprahique souffrent d'une impréscision certaine intrinsèquement liée à la puce GPS utilisée.

1.3.8 Interprétation du contexte

> L'interprétation fait référence au processus d'élévation
> du niveau d'abtraction d'un fragment de contexte [6]
> *Anind Dey, Gregory Abowd, and Daniel Salber.*
> *A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications.*
> *Human-Computer Interaction, 16(2) :97–166, December 2001.*

FIGURE 1.5: Intérprétation du contexte par Dey (2001)

Après avoir détecté, récupéré et sauvegardé le contexte provenant de sources variées, le système sensible au contexte devra implémenter des méchanismes d'intérprétation du contexte, de sorte que les informations recueillies soit utilisées de manière convenable. Cette étape d'interprétation implique l'integration des plusieurs contexte en un seul et unique contexte de plus haut nivea. Ainsi, l'interprète modifie les informations de contexte en augmentant son niveau d'abstraction. Une apporoche est d'utiliser la fusion de contexte pour convertir des informations de bas niveau en un contexte global directemment à la portée des applications.

1.3.9 Framework conceptuel en couches

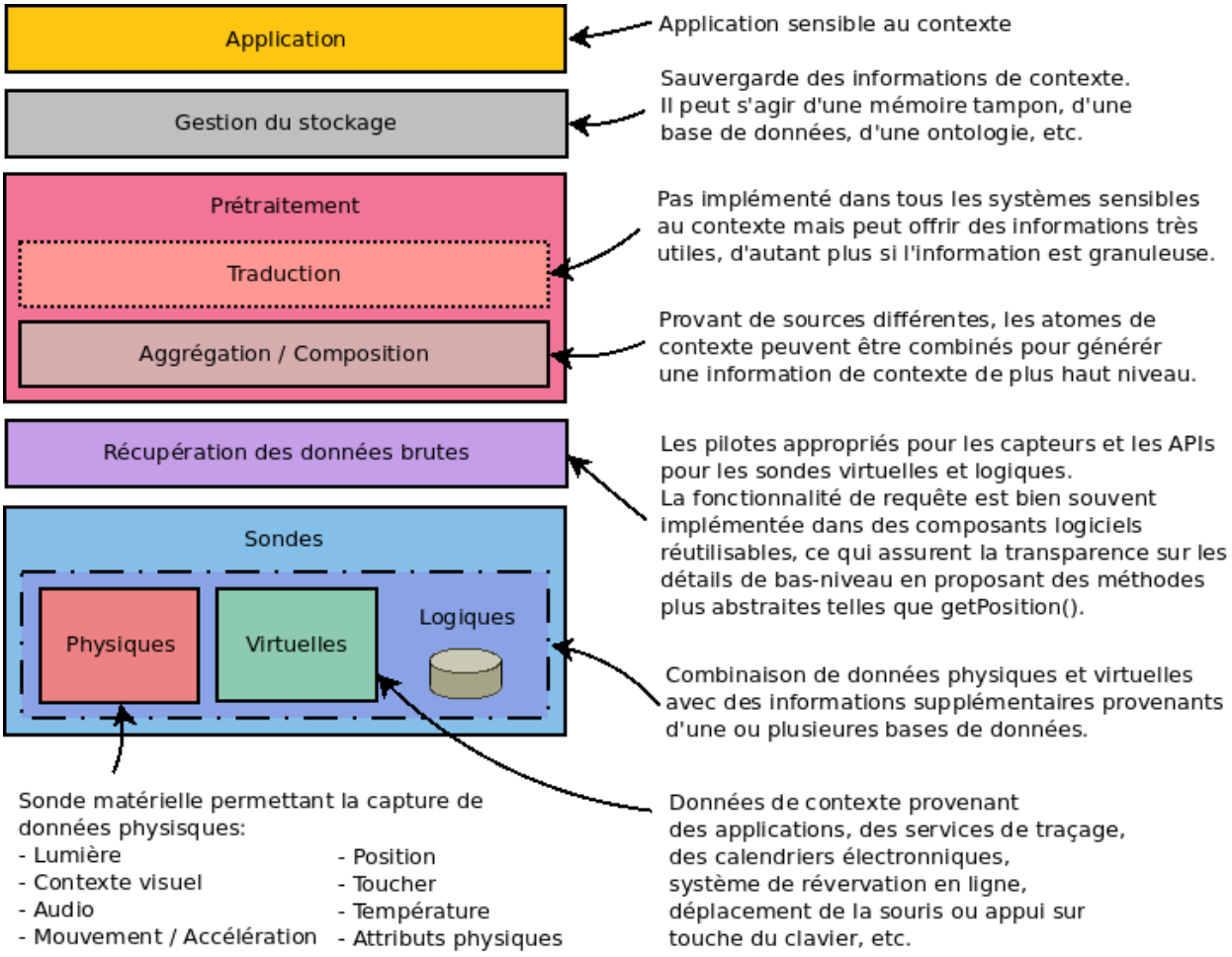

FIGURE 1.6: Framework conceptuel d'un système sensible au contexte

1.3.10 Sécurité et confidentialité

> La confidentialité est intrinsèquement liée au contrôle. [2]
> *Mark Ackerman, Trevor Darrell, and Daniel Weitzner. Privacy in Context. Human-Computer Interaction, 16(2) :167–176, December 2001.*

FIGURE 1.7: La confidentialité par Ackerman

Comme le contexte est susceptible de contenir des informations sensible sur des personnes, leur situation géographique ou leur activité par example, il est nécéssaire de protéger le caractère personnel de ces données. Les systèmes sensibles au contexte soulèvent des défis en termes de confidentialité inhabituels et inconsidérés par les systèmes traditionnels. La question quant au contrôle des différents capteurs et services n'est pas abordée dans Dey et al. [6], et bien souvent, des applications telles

que le Context Toolkit sont supposées être sous le contrôle d'un "utilisateur". Les utilistateurs sont assignés comme propriétaires des données sondées par leurs soins. Ils sont par ailleurs autorisés à octroyer l'accès à ces données à d'autres utilisateurs.

Néamoins, en ce qui concernent les sondes capables de déterminer la présence d'individus dans une pièce ou à tout autre endroit, il n'existe aucunes préconisations à ce sujet. C'est un point toutefois critique.

D'une manière générale, dans une environemment mettant à disposition un certain eventail d'applications sensibles au contexte, un individu opère à travers de mutliples environnemments sociaux, qui eux même font l'usage des données d'un grand nombre d'autres individus.

epackagefloat Aspects souvent négligés, la sécurité et la confidentialité font partie des composantes les plus importantes d'un système sensible au contexte, car la protection des données sensibles doit être garantie.

1.4 Systèmes et frameworks existants

1.4.1 Technologies de détection

1.4.2 Représentations du contexte

1.4.3 Découverte des ressouces

1.4.4 Gestion du contexte historique

1.4.5 Sécurité et confidentialité

1.4.6 Conclusion

Chapitre 2

Problématiques émergentes

L'une des principales innovations à apporter dans les infrastructures d'applications sensibles au contexte réside dans l'introduction d'une interface présentant un niveau d'abstraction élevé. Cela permetterait de représenter la connectivité des composants applicatifs avec les politiques de haut niveau qui la régissent. Cette couche doit rester simple d'utilisation pour les développeurs d'applications, et le modèle améliorer simultanément l'automatisation et la sécurité.

2.1 Ontologie de contexte

2.2 Théorie de promesse

Le modèle permettant la définition des politiques d'administration serait un modèle orienté sur les ontologies et basé sur la théorie de la promesse. Celle-ci s'appuie sur un contrôle évolutif des objets intelligents, contrairement aux modèles impératifs plus traditionnels pensés comme des systèmes de gestion descendants. Dans ces derniers, le gestionnaire central doit être informé des commandes de configuration des objets sous-jacents et de l'état actuel de ces objets.

Au sein du contexte, le modèle fournit une série d'objets qui définissent l'application. Les objets englobent les terminaux, les groupes de terminaux et les politiques qui définissent leur relation.

L'infrastructure conçoit un modèle d'objet pour le déploiement d'applications, ces dernières constituant le point central. Historiquement, les applications étaient limitées par les capacités du réseau et par des configurations visant à prévenir leur utilisation abusive. Des concepts tels que l'adressage, le VLAN et la sécurité sont depuis toujours intimement liés, ce qui limite l'évolutivité et la mobilité des applications. Alors que les applications sont redessinées pour la mobilité et l'évolutivité web,

cette approche traditionnelle empêche leur déploiement rapide et homogène.

2.2.1 Principes de fonctionnement

2.3 Algorithmes de consensus

2.3.1 Algorithme de Paxos

2.3.2 Algorithme de Raft

Raft est un algorithme de consensus qui est conçu pour être facile à comprendre. Il est équivalent à Paxos dans la tolérance aux pannes et en termes de performance. La différence, c'est qu'il est décomposé en sous-problèmes relativement indépendants, et il traite de manière rigoureuse toutes les pièces majeures nécessaires pour obtenir un système cohérent.

Le consensus est un problème fondamental dans les systèmes distribués tolérants aux pannes. Consensus implique de multiples serveurs acceptant des valeurs. Une fois qu'ils atteignent une décision sur une valeur, cette décision est définitive. Les algorithmes de consensus typiques sont amenés à faire des progrès lorque la majorité de leurs serveurs sont disponibles, par exemple, un cluster de 5 serveurs peut continuer à fonctionner même si deux serveurs ne sont plus disponibles. Si plusieurs serveurs échouent, ils cessent de faire des progrès (mais ils ne retournerons jamais de valeurs érronées)

2.4 Vue d'ensemble

Comme on le voit 2.1

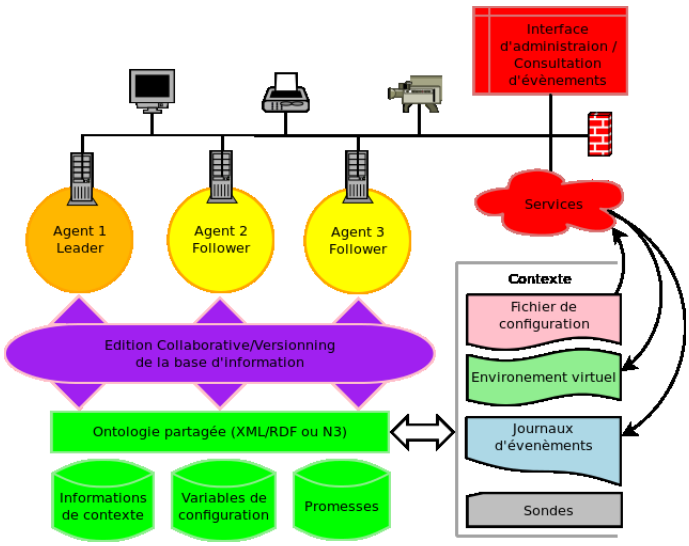

FIGURE 2.1: Schéma d'implémentation du système multi-agents

Chapitre 3

Simulation

Bibliographie

[1] G. D. Abowd, C. G. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. Baltzer journals september 23, 1996. 1997.

[2] M. Ackerman, T. Darrell, and D. J. Weitzner. Privacy in context. *Human–Computer Interaction*, 16(2-4) :167–176, 2001.

[3] C. Catharina and Kari. Context aware management architecture, 2002.

[4] H. Chen, T. Finin, and A. Joshi. An intelligent broker for context-aware systems. In *Adjunct proceedings of Ubicomp*, volume 3, page 183–184, 2003.

[5] A. K. Dey. *Providing architectural support for building context-aware applications*. PhD thesis, Georgia Institute of Technology, 2000.

[6] A. K. Dey, G. D. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-computer interaction*, 16(2) :97–166, 2001.

[7] M. Ebling, G. D. Hunt, and H. Lei. Issues for context services for pervasive computing. In *Middleware 2001 Workshop on Middleware for Mobile Computing, Heidelberg*, 2001.

[8] J. I. Hong and J. A. Landay. An infrastructure approach to context-aware computing. *Human-Computer Interaction*, 16(2) :287–303, 2001.

[9] Korpipää and Mäntyjärvi. An ontology for mobile device sensor-based context awareness. 2003.

[10] A. Schmidt, M. Beigl, and H.-W. Gellersen. There is more to context than location. *Computers & Graphics*, 23(6) :893–901, 1999.

[11] T. Strang and C. Linnhoff-Popien. A context modeling survey. In *Workshop Proceedings*, 2004.

[12] S. Voida, E. D. Mynatt, B. MacIntyre, and G. M. Corso. Integrating virtual and physical context to support knowledge workers. *IEEE Pervasive Computing*, 1(3) :73–79, 2002.

[13] T. Winograd. Architectures for context. *Human-Computer Interaction*, 16(2) :401–419, 2001.

Annexe A

Appendix Title

Supplementary material goes here. See for instance Figure A.1.

A.1 Lorem Ipsum

dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

"I am glad I was up so late,
for that's the reason I was up so early."
William Shakespeare (1564-1616), British dramatist, poet.
Cloten, in Cymbeline, act 2, sc. 3, l. 33-4.

FIGURE A.1: A deep quote.