

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

	Page
LISTE DES FIGURES	iv
LISTE DES TABLES	v
LISTE DES ALGORITHMES	vi
REMERCIEMENTS	vii
RÉSUMÉ	viii
1 État de l'art	1
1.1 Introduction	1
1.2 Background	2
1.3 Vue d'ensemble sur le contexte	2
1.3.1 Classes de contexte	3
1.3.2 Les caractéristiques des informations de contexte	4
1.3.3 Système sensible au contexte	5
1.3.4 Recueillir l'information	6
1.3.5 Récupérer l'information	7
1.3.6 Les architectures de gestion de contexte	8
1.3.7 Représentation du contexte	11
1.3.8 Interprétation du contexte	11
1.3.9 Framework conceptuel en couches	11
1.3.10 Sécurité et confidentialité	11
1.4 Systèmes et frameworks existants	11
1.4.1 Technologies de détection	11
1.4.2 Représentations du contexte	11
1.4.3 Découverte des ressources	11
1.4.4 Gestion du contexte historique	11
1.4.5 Sécurité et confidentialité	11
1.4.6 Conclusion	11
2 Problématiques émergentes	12
3 Simulation	13
Bibliographie	14

A	Appendix Title	15
A.1	Lorem Ipsum	15

Table des figures

	Page
1.1 Le context défini par Abowd et. al.	2
1.2 Winograd a propos du contexte	2
1.3 La fiabilité des capteurs par Schmidt	6
1.4 Comparaison des différents modèles de gestion de contexte	10
2.1 Schéma d'implémentation du système multi-agents	12

Liste des tableaux

Page

LISTE DES ALGORITHMES

Page

REMERCIEMENTS

Je souhaiterais 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

Université 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'environnement 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 perturbations 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 reconnaître et exploiter l’information implicitement présente dans l’environnement. 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’environnement et leurs activités.

Dans ce mémoire, nous aborderons les principes communs à chacune des architectures existantes, desquels nous détaillerons le framework conceptuel dérivé (!) par couches. Nous présenterons une 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 comprennent de manière tacite ce qu'est le contexte, ils le trouvent 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 quelconque interaction, alors cette information appartient au contexte.

1.3 Vue d'ensemble sur le contexte

“Le contexte est efficace, seulement lorsqu'il est partagé.” [9]
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 recueilli 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 Utilisateur** - 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 conditions de circulation ou la température.

Chacune de ces catégories contiennent une richesse d'informations pertinentes 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 pertinentes 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 édité.

1.3.1.2 Contexte physique

Le contexte physique d'un autre côté 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 acquiesable 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 nécessite 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 : [2]

1. Les informations de contexte présente une gamme de caractéristiques temporelles

L'information de contexte est d'hors et déjà catégorisée selon l'environnement auquel il appartient : virtuel ou physique ; elle peut en outre est subdivisée selon un critère de temporalité :

- Information statique : toute information apparentée à l'environnement 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" [2] 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 désuète.

3. L'information incarne un grand nombre de représentations

Les données brutes recueillies depuis l'environnement 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" [2] de l'information et celle apte à la situation courante est quasi-nulle.

4. Les informations de contexte sont très fortement corrélées

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." [2]

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 plusieurs formes, mais pour accomplir cet objectif, le système doit d'un manière générale :

- Recueillir l'information
- Sérialiser cette information
- Fusionner l'information pour générer un contexte de plus haut niveau
- Prendre automatiquement des mesures basées sur l'information recueillie
- Rendre l'information disponible à l'utilisateur, 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 : [4]

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

Les développeurs du système Kimura ont conçu des composants distribués destinés à un système sensible au contexte global. Ces composants se divisent en trois classes [8], qui sont les suivantes :

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

1.3.4 Recueillir l'information

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

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

“Les capteurs ne sont pas toujours précis ou fiables à 100%, tout particulièrement s'ils sont à usage unique. Le processus de récupération de l'information doit être tolérant à la défaillance potentielle d'un capteur. Toutes les informations recueillies doivent être assujetties à des contrôles de validité pour vérifier leur exactitude. La fusion des capteurs est un moyen de remédier à cette difficulté.” [7]
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 électrique, possédant de nombreuses sondes de température im-

plantées à divers endroits d'un reacteur nucléaire. Le système doit prendre la décision 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 une valeur moyenne et/ou d'écarter les valeurs reportées qui manifestement diffèreraient trop des autres mesures. Cette méthode éviterait des prises de mesures drastiques du système sensible 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 récupère l'information avant même qu'elle soit requise. Cela améliore indéniablement les performances, mais nécessite une consommation des ressources fréquente et considérable pour une information qui pourrait bien ne jamais être exploitée.
- Modèle "Pull" - Collecte l'information de contexte au moment opportun. 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'acquisition 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) [3] 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 périphériques 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 adaptée pour les systèmes distribués.

1.3.5.2 Infrastructure intergicielle

L'approche intergicielle introduit une 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 chacune des composantes du système. Chaque composante doit être en mesure d'effectuer les opérations 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 supporte un grand nombre de périphériques et d'applications sans l'utilisation de normes de codage et des protocoles réseaux standards.

1.3.5.3 Serveur de contexte

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

1.3.6 Les architectures de gestion de contexte

Winograd (2001) [9] décrit trois modèles différents afin de coordonner les processus 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'application. Un widget de contexte fonctionne totalement indépendamment de l'application, ce qui permet à plusieurs 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) [6], ressemble à l'architecture de serveur de contexte. Au lieu d'un gestionnaire de widget cen-

tralisé, des techniques sont utilisées pour découvrir 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 événements 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 perturbations 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 nécessite une compréhension avancée des mécanismes qu'il implémente pour faire l'usage, ne sera utilisé que par ceux qui auront le dévouement et la motivation de le maîtriser [9]

- **Extensibilité** :

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

[5]

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 en cas d'échec. Le modèle d'infrastructure constitué de composants indépendants est très convoluté, mais cela ne semble pas être 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. [9]

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

1.3.8 Interprétation du contexte

1.3.9 Framework conceptuel en couches

1.3.10 Sécurité et confidentialité

1.4 Systèmes et frameworks existants

1.4.1 Technologies de détection

1.4.2 Représentations du contexte

1.4.3 Découverte des ressources

1.4.4 Gestion du contexte historique

1.4.5 Sécurité et confidentialité

1.4.6 Conclusion

Chapitre 2

Problématiques émergentes

Comme on le voit 2.1

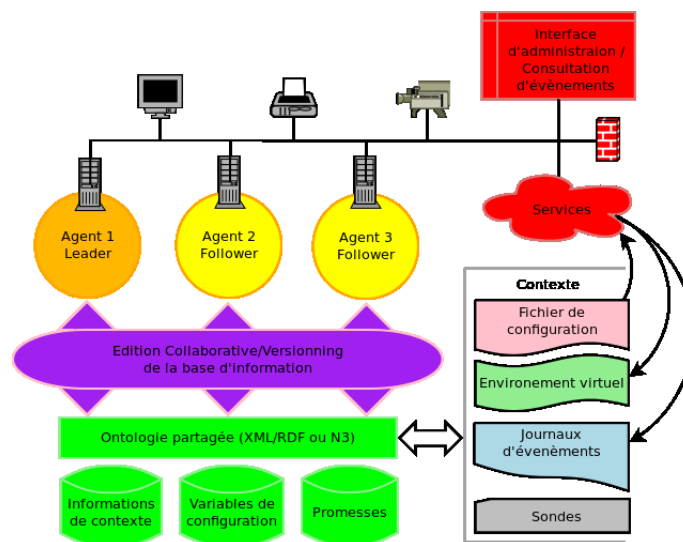


FIGURE 2.1: Schéma d'implémentation du système multi-agents

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] C. Catharina and Kari. Context aware management architecture, 2002.
- [3] 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.
- [4] A. K. Dey. *Providing architectural support for building context-aware applications*. PhD thesis, Georgia Institute of Technology, 2000.
- [5] 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.
- [6] J. I. Hong and J. A. Landay. An infrastructure approach to context-aware computing. *Human-Computer Interaction*, 16(2) :287–303, 2001.
- [7] A. Schmidt, M. Beigl, and H.-W. Gellersen. There is more to context than location. *Computers & Graphics*, 23(6) :893–901, 1999.
- [8] S. Voidsa, 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.
- [9] 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.