Ingénierie dirigée par les modèles pour la construction d'applications ubiquitaires sensibles à la qualité du contexte

Sophie Chabridon, Denis Conan, Zied Abid and Chantal Taconet

Institut Télécom, Télécom SudParis, UMR CNRS 5157 SAMOVAR
9 rue Charles Fourier
Evry ; F-91011 ; France
Prenom.Nom@telecom-sudparis.eu

Résumé

Résumé : *Les dispositifs mobiles équipés de capteurs se multipliant, il est naturel de s'attendre à ce que les applications ubiquitaires exploitent la richesse de ces informations de contexte. Cependant, la conception et l'implantation de telles applications sensibles au contexte reste un défi car peu de modèles et d'outils existent pour guider les architectes, les concepteurs et les développeurs. Cette difficulté est exacerbée par la nature imparfaite des informations de contexte. Dans nos travaux, nous complétons les informations de contexte par des méta-informations représentant leur qualité. Nous proposons un processus outillé générique et extensible de conception d'applications sensibles au contexte prenant en compte la qualité des informations de contexte. Nous illustrons sa mise en œuvre sur un prototype d'application dans le domaine du commerce mobile.*

Mots clefs : *Ingénierie dirigée par les modèles, ubiquité, contexte, qualité de contexte*

1 Introduction

Ces dernières années, la gestion de contexte en environnement pervasif suscite l'intérêt d'une communauté de recherche de plus en plus importante. Cependant, les informations de contexte sont par nature imparfaites puisque d'une part elles résultent d'un modèle représentant une abstraction de la réalité ne pouvant pas être exhaustive et d'autre part elles dépendent de dispositifs tels que des capteurs ayant des limitations physiques intrinsèques. Les informations de contexte ont donc tendance à être incorrectes car elles ne reflètent pas exactement l'état réel de l'entité observée, incomplètes lorsque certains aspects du contexte sont manquants, ou encore ambiguës si plusieurs valeurs sont collectées et ne correspondent pas entièrement entre elles [8]. Par exemple, deux dispositifs de localisation distincts peuvent fournir des valeurs correspondant à des régions se recouvrant, ayant des niveaux de précision (granularité) différents ou bien encore étant incohérentes si elles présentent des informations contradictoires. Lorsque les informations de contexte collectées sont imparfaites et incertaines, le risque existe de baser une décision sur des informations erronées [7]. De plus, raisonner sur des informations incertaines induit des coûts de raisonnement très élevés dûs à la complexité des solutions à mettre en œuvre [6, 19]. Il est donc important de prévoir les mécanismes nécessaires à la limitation de l'impact que peuvent avoir ces imperfections, notamment en permettant au système de prendre en compte l'existence

même de ces imperfections. C'est pourquoi la connaissance ainsi que la maîtrise de la qualité des informations de contexte deviennent primordiales pour la construction d'applications sensibles au contexte.

La notion de qualité de contexte (QoC) a été définie pour la première fois dans le cadre de la gestion de contexte par [3] en tant que concept à part entière pour décrire la qualité des informations utilisées comme informations de contexte, indépendamment du processus ou du composant matériel ayant fourni ces informations. Cette définition a été complétée par [14] en ajoutant la notion de valeur pour une application spécifique. La QoC apparaît donc comme un ensemble de méta-données caractérisant une information de contexte.

La suite de cet article est organisée de la manière suivante. Dans la section 2, nous présentons les motivations de nos travaux en les illustrant par une application de ventes flash dans un centre commercial. Puis, en section 3, nous présentons une vue globale du processus de conception de la sensibilité au contexte des applications ubiquitaires qui inclut la prise en compte des exigences de QoC et nous le relions aux plateformes logicielles que nous proposons. Puis, en section 4, nous décrivons un gestionnaire de contexte qui prend en compte les informations de QoC. En section 5, nous présentons l'évaluation du gestionnaire de contexte en terme de performance avec et sans prise en compte de la QoC, cette évaluation est réalisée sur terminal mobile. En section 6, nous discutons de la prise en compte de la QoC dans des travaux connexes en modélisation de la sensibilité au contexte et en gestion de contexte. Enfin, en section 7, nous concluons notre article.

2 Motivations

Considérons le scénario de vente Flash dans un centre commercial qui suit. *Marie se rend dans le plus grand centre commercial de la région. Elle possède un téléphone mobile avec navigation GPS, communication 3G, WiFi et Bluetooth. Quand elle se gare sur le parking extérieur à 10 heures, elle reçoit un message l'informant de la disponibilité d'un nouveau service d'annonce de* ventes flash *et lui proposant de le télécharger. Comme Marie est une cliente privilégiée de ce centre commercial, elle s'est déjà inscrite pour bénéficier de services et de conseils en indiquant ses goûts et produits préférés. Juste après avoir téléchargé la nouvelle application, elle reçoit une annonce indiquant qu'il lui reste 1 heure pour profiter d'une vente flash en cours dans son magasin de sport préféré. À ce moment-là, la localisation de Marie n'est pas connue de manière exacte. Dans ce cas, l'application indique à l'écran la position estimée de Marie ainsi que la position du magasion, sans indication supplémentaire. Dans l'après-midi, Marie reçoit alors une nouvelle annonce pour une vente flash qui va commencer dans un magasin venant juste d'ouvrir et qu'elle ne connaît pas encore. La vente flash est exceptionnelle et ne dure que 15 minutes. La couverture réseau est à ce moment-là très bonne et la localisation de Marie est estimée de manière très précise grâce aux informations récupérées par le téléphone. Une carte du centre commercial apparaît alors sur l'écran du téléphone, centrée sur la position actuelle de Marie. Un tracé sur la carte lui indique clairement le chemin à suivre pour se rendre dans le magasin où se déroule la vente flash. Marie décide alors de prendre la direction de ce nouveau magasin. Tout au long du parcours, elle est informée du temps qu'il reste avant la fin de la vente flash et la carte avec l'itinéraire est rafraîchie régulièrement. Grâce aux indications précises fournies par*

le service de vente flash Marie atteint le nouveau magasin dans les temps.

Ce scénario montre qu'il est essentiel qu'un service exploitant des informations décrivant le contexte d'un utilisateur puisse mesurer la qualité des informations de contexte. Par exemple, la localisation d'un utilisateur mobile peut être fournie par plusieurs techniques (p.ex. GPS, GPRS, Wifi), mais de manière plus ou moins fiable. Il est donc nécessaire de qualifier cette localisation. Nous proposons de prendre en compte des méta-données indiquant la qualité du contexte (QoC) à différentes étapes du processus de conception des applications sensibles au contexte. Plusieurs techniques de localisation pouvant être utilisées sur le téléphone, seule la position de l'utilisateur avec une qualité suffisante et la plus élevée parmi celles disponibles est effectivement affichée sur la carte. De plus, le service de vente flash est adapté en fonction de la qualité de l'information de contexte qu'il utilise, à savoir la localisation. Une annonce de vente flash est affichée sur le téléphone uniquement si la localisation est connue de manière suffisamment fiable. De plus, l'itinéraire précis est indiqué uniquement s'il peut être basé sur une bonne estimation de la position de départ de l'utilisateur. Il serait contre-productif de fournir des indications erronées à l'utilisateur et de l'envoyer dans une mauvaise direction. Dans le cas de la vente se déroulant dans le magasin de sport préféré de Marie, la qualité de la localisation de Marie n'est pas suffisante, le service n'affiche pas la carte détaillée avec l'itinéraire. Mais dans ce cas, Marie trouve facilement comment se rendre dans le magasin de sport et n'est pas gênée de ne pas avoir d'indications plus précises sur le chemin à suivre. Le niveau de service offert dépend ainsi de la qualité des informations de contexte.

Dans le reste de cet article, nous présentons un processus de conception et un cadriciel générique pour manipuler la QoC. Pour valider notre approche, nous utilisons comme exemple fil rouge le prototype de l'application que nous avons développé pour ce scénario de vente flash.

3 Processus de conception d'applications sensibles à la QoC

Dans cette section, nous décrivons brièvement le processus de conception que nous préconisons pour la construction d'applications sensibles au contexte, issu de nos travaux antérieurs [23]. Nous nous concentrons principalement ici sur les extensions apportées à ce processus pour la définition des exigences de QoC.

Le rôle de la gestion de contexte est de fournir des informations de contexte de haut niveau afin d'identifier des situations pouvant nécessiter une adaptation de l'application sensible au contexte. Cela consiste en trois tâches qui sont l'acquisition de données de contexte, le traitement de ces données par fusion, l'agrégation ou l'interprétation, et la présentation de situations de contexte pertinentes pour l'application. Notre processus de conception découple autant que faire se peut la conception de la gestion de contexte de la conception de l'application métier. Pour cela, nous ajoutons au domaine de la conception métier deux autres domaines dédiés : la gestion de contexte et la sensibilité au contexte, induisant les deux nouveaux rôles que sont le concepteur de la gestion de contexte et le concepteur de la sensibilité au contexte.

Nous avons suivi le modèle SPEM (Software Process Engineering MetaModel) [17] pour définir les rôles, les activités et les résultats produits dans le domaine de la gestion de contexte. Le méta-modèle du gestionnaire de contexte utilisé, qui est COSMOS [4] dans nos travaux, est

en entrée de toutes les activités. Par conséquent, tous les modèles et implantions résultants sont conformes au méta-modèle de COSMOS et à son API. Le concepteur de la gestion de contexte définit les inférences d'informations de contexte de haut niveau telles que la distance entre deux localisations à partir de données brutes collectées par ce que nous appelons des *collecteurs de contexte*. Cela implique de définir des *nœuds de contexte* en utilisant un langage dédié au domaine de la gestion de contexte, à savoir le DSL COSMOS, dans lequel une inférence d'informations de contexte de haut niveau est une expression impliquant des données de contexte de plus bas niveau mettant en œuvre ce que nous appelons des *opérateurs de contexte*.

Ainsi, la première activité est la spécification de collecteurs de contexte à l'aide du DSL COSMOS et leur implantion en quelques lignes de code Java. Par exemple, le concepteur peut définir plusieurs collecteurs de contexte pour récupérer des positions brutes provenant de différents capteurs (p.ex. GPS, GPRS, Wifi) et différents équipements (p.ex. téléphones Android, iPhones). La seconde activité concerne la conception des opérateurs de contexte. La caractéristique importante de notre approche est que ces opérateurs permettent le calcul de la qualité de l'information de contexte en complément de la valeur elle-même de cette information de contexte, mettant ainsi en œuvre des *opérateurs de contexte sensibles à la QoC* et des *nœuds de contexte sensibles à la QoC*. Afin de rendre incontournable la qualification des informations de contexte, nous introduisons une troisième activité spécifique à la conception d'*opérateurs sur paramètre de QoC* qui calculent les méta-données de QoC (p.ex. la fraîcheur d'une information de contexte, fonction de son âge depuis la date de collecte et de sa durée de vie). La quatrième et dernière activité est constituée par la conception des entités représentant les informations de contexte de haut niveau à l'aide du DSL COSMOS. Cela revient à décrire les expressions de traitement des données de contexte. Ces expressions réutilisent les collecteurs de contexte, opérateurs de contexte et opérateurs sur paramètre de QoC conçus au cours des activités précédentes. Cela se fait grâce au DSL COSMOS qui permet la réutilisation par composition et évite ainsi au concepteur de la gestion de contexte de programmer ces expressions (cf. Section 4, et plus particulièrement la Figure 2).

Nous décrivons maintenant les principales activités du concepteur de la sensibilité au contexte des applications. La première activité consiste en la spécification des entités devant être observées à l'exécution (que nous nommons *entités observables* ou *entités*), et la spécification de quelles données de contexte doivent être collectées ou inférées (appelées *observables* dans notre terminologie) sur ces entités. Par exemple, dans le cas de l'application de vente Flash, le client et les magasins sont des exemples d'entités à observer, la localisation du client et ses produits préférés étant des observables. Le méta-modèle de sensibilité au contexte CA3M [23] ainsi que le méta-modèle de la logique métier des applications sont des entrées de cette première activité. En effet, le rôle du concepteur dans cette activité est d'étiqueter par des stéréotypes les éléments du modèle UML de l'application représentant les entités et les observables. La seconde activité consiste à faire le lien entre les artefacts produits par le processus de conception de la gestion de contexte et les artefacts de l'application métier au travers de la définition de contrats spécifiques. Par exemple, un contrat de sensibilité au contexte indique qu'une annonce de vente Flash doit être déclenchée dès lors qu'une vente Flash correspondant aux préférences du client a lieu à proximité. Lors de la définition de ces contrats, le concepteur ajoute les exigences de QoC de l'application métier et configure les nœuds de contexte sensibles à la QoC avec les *niveaux de QoC* appropriés. Un niveau de QoC correspond à un niveau global de qualité calculé en agrégeant

plusieurs paramètres de QoC tels que la fraîcheur et la confiance. Étant donné que les exigences de QoC sont propres à une application, la configuration de ces niveaux de QoC fait partie du domaine de la sensibilité au contexte et non de celui de la gestion de contexte.

4 Gestion de la QoC

Même s'il n'existe pas de consensus sur les critères de qualité à utiliser, un certain nombre d'entre eux reviennent dans plusieurs travaux du domaine ([3, 12, 16, 18, 24]) :

- l'exactitude (ou estimation) décrit dans quelle mesure la valeur de l'information de contexte représente la réalité. L'exemple typique est celui de la localisation. Son exactitude dépend du mécanisme utilisé. Avec un récepteur GPS, une estimation d'environ 4 mètres est attendue, alors qu'un réseau cellulaire atteint des estimations d'au mieux 500 mètres en zone urbaine ;
- la probabilité de correction est liée à la présence d'erreurs non intentionnelles. La source de l'information indique au travers de la probabilité de correction combien de fois l'information de contexte qu'elle fournit risque d'être erronée en raison de problèmes internes ;
- la confiance concerne le point de vue du fournisseur de contexte pour qualifier les différentes sources d'informations qu'il utilise. Par exemple, le fournisseur de contexte A envoie une information de contexte à B. A indique que la probabilité de correction de cette information est de 100%. Cependant, B recevant souvent des informations incorrectes de la part de A, il transmettra les informations à ses clients avec une mise en garde sur le fait que la source de l'information n'est pas entièrement digne de confiance ;
- la résolution ou granularité est appelée précision par [18]. Un fournisseur de contexte annonce que la température d'une pièce est de 17°C. Ceci est vrai en moyenne, mais pas nécessairement à proximité du radiateur présent dans la pièce. La valeur pourra ainsi être complétée par des bornes min et max indiquant le niveau de détail de la mesure : 17$\pm$ 1°C ;
- la fraîcheur (ou âge des informations [18]) indique si les informations sont suffisamment récentes pour représenter la situation présente ;
- la portée géographique indique la zone géographique dans laquelle l'information peut être utilisée. La température fournie par une station météo extérieure n'est pas la plus adaptée pour contrôler le chauffage central à l'intérieur d'un bâtiment.

COSMOS gère la QoC grâce à un nœud de contexte étendu, un nœud de contexte sensible à la QoC, composé d'un composant opérateur de contexte sensible à la QoC et de un ou plusieurs composants opérateurs sur paramètres de QoC [1].

4.1 Politique de contexte pour l'application de vente Flash

La figure 1 montre le graphe de la politique de contexte de l'application de vente Flash avec un focus sur la partie concernant le calcul de la localisation ajustée. La figure 2 donne la description à l'aide du DSL COSMOS correspondante. Ce sous-arbre n'est pas décrit entièrement : nous présentons uniquement les parties spécifiques à l'application. De plus, les nœuds provenant de la

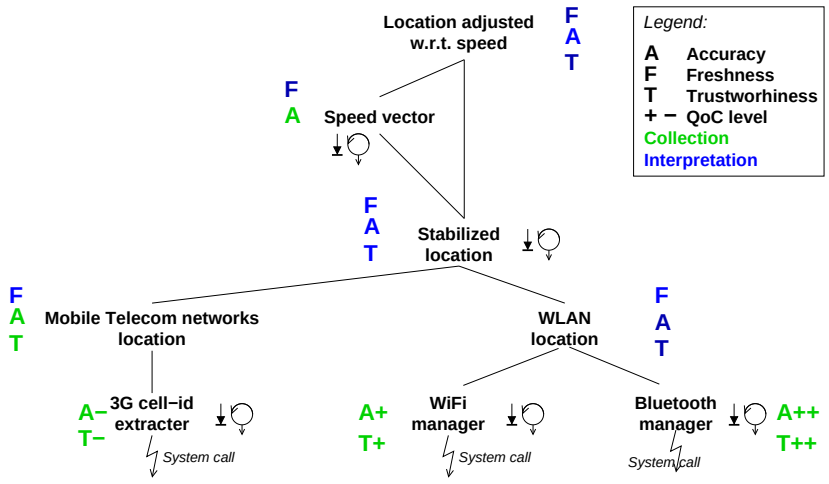

FIGURE 1 – Politique de contexte avec QoC pour l'ajustement de la localisation

bibliothèque générique de COSMOS (indiqués en italique) sont décrits ailleurs. Cette spécification est ensuite fournie en entrée à l'outil cosmos2code pour générer l'architecture logicielle de l'application sous forme de composants ainsi que le squelette des composants primitifs. Le programmeur de l'application n'a alors plus qu'à compléter le code de l'application en écrivant quelques lignes de code Java pour les composants primitifs.

La politique de contexte exploite les différentes techniques de localisation pouvant être accessibles depuis un téléphone mobile telles que les réseaux 3G, la radio sans fil et le GPS, et effectue le choix d'une localisation stabilisée. Ce choix est guidé par la QoC de l'information de localisation. Nous considérons trois paramètres de QoC dans cette politique de contexte : l'exactitude (notée A pour *accuracy* sur le graphe), la fraîcheur (notée F) et la confiance (notée T pour *trustworthiness*). Pendant le processus d'inférence de la localisation, le nœud de contexte Stabilized location choice détermine la meilleure localisation en fonction des valeurs de la confiance, de la fraîcheur et de l'exactitude, dans cet ordre.

5 Évaluation

Nous présentons dans cette section le résultat des expérimentations que nous avons menées pour évaluer les performances de l'application de vente Flash et mesurer le coût de la gestion de la qualité de contexte avec un téléphone Samsung Galaxy S2, avec le système d'exploitation Android 2.3.5, un processeur à deux cœurs ARM Cortex-A9 cadencés à $1.2GHz$, avec $16Go$ de ROM et $1Go$ de RAM ($660Mo$ initiallement disponibles), exécutant une machine virtuelle Java 1.5.

La table 1 indique la durée nécessaire pour créer l'ensemble de la politique de contexte de l'application de vente Flash. Avec ou sans gestion de la QoC, l'ensemble des nœuds de contexte nécessaires pour la politique de contexte de la vente Flash est créé en moins de $100ms$. Ce temps reste très raisonnable et n'impacte pas de manière significative le temps de déploiement sur un

```
package evry2mallqoc;
chunk position = {
    classname : evry2mallqoc.PositionChunk,
    typeparam : evry2mallqoc.PositionInfo
}
message location = {
    position, accuracy, freshness, trustworthiness
}
operator computeGprsLocation = {
    output : location,
    classname : evry2mallqoc.ComputeGprsPosition,
    input : gprs
}
node gprsLocation = {
    operator : computeGprsLocation,
    children : gprsExtracter
}
node stabilizedLocation = {
    operator : computeStabilization,
    children : gprsLocation gpsLocation wlanLocation
}
operator computeNonAdjustedLocation = {
    output : location,
    classname : evry2mallqoc.ComputeNonAdjustedLocation,
    input : location
}
node nonAdjustedLocation = {
    operator : computeNonAdjustedLocation,
    children : stabilizedLocation
}
operator computeAdjustedLocation = {
    output : location,
    classname : evry2mallqoc.ComputeAdjustedLocation,
    input : location speed
}
node adjustedLocation = {
    operator : computeAdjustedLocation,
    children : nonAdjustedLocation speedVector
}
configuration : gprsExtracter
    [observethrough=false] [periodobserve=10000]
configuration : stabilizedLocation
    [observethrough=false] [periodobserve=2000]
configuration : nonAdjustedLocation
    [observethrough=false] [periodobserve=3000]
configuration : adjustedLocation
    [observethrough=false] [periodobserve=5000]
```

FIGURE 2 – Politique de calcul de “adjusted location" en COSMOS DSL

téléphone mobile.

Cas de test	Moyenne (ms)	Intervalle de confiance à 95%
Sans QoC	69.53	[67.77..71.31]
Avec QoC	99.28	[97.04..101.53]

TABLE 1 – Durée de l'instantiation

La table 2 donne la durée d'une observation de la politique de contexte avec et sans gestion de la QoC dans le pire cas, c'est-à-dire lorsque tous les nœuds sont non bloquants (l'appel parcourt tous les nœuds du graphe jusqu'au nœuds feuilles). Le point à noter est que le surcoût lié à la gestion de la QoC est seulement d'environ $1ms$.

Cas de test	Moyenne (ms)	Intervalle de confiance à 95%
Sans QoC	15.2	[14.87..15.52]
Avec QoC	16.27	[15.54..17.00]

TABLE 2 – Durée d'une observation (cas défavorable)

Les résultats de l'évaluation de performance présentés dans cette section montrent que COSMOS permet d'effectuer la gestion de contexte entièrement sur un téléphone mobile. Le calcul des méta-données de QoC engendre un coût supplémentaire, expliqués par la nécessité d'ajouter des composants dans l'architecture mise en place, mais celui-ci reste faible. D'autres tests montrent que notre approche peut passer à l'échelle lorsqu'un grand grand nombre de nœuds de contexte est nécessaire. Par exemple, en moins de $100ms$, il est possible de parcourir 1 chemin de 121 nœuds ou 20 chemins de 5 nœuds. Il est aussi possible de créer près de 3000 nœuds de contexte sensibles au contexte avant de saturer la mémoire vive.

6 Travaux connexes

La production d'applications sensibles au contexte est une tâche complexe pour laquelle l'ingénierie dirigée par les modèles apporte des mécanismes et des outils essentiels. Nous présentons dans cette section les travaux en modélisation de la sensibilité au contexte et en gestion de contexte, et mentionnons si la QoC est prise en compte ou non.

Les principales familles de modélisation de contexte sont la définition de profils (p.ex. CC/PP [13]), les bases de données (p.ex. CML [9]), les ontologies (p.ex CONON [25]) et l'IDM. Dans notre approche, nous avons choisi de suivre les principes de l'IDM qui permet de définir des liens entre la modélisation de contexte, qui permet d'exprimer des situations de contexte, et la modélisation de la sensibilité au contexte, qui associe ces situations de contexte avec des entités de l'application.

ContextUML [22] est l'un des premiers modéles dédiés au domaine de la sensibilité au contexte. Il définit un méta-modèle pour la modélisation de services Web sensibles au contexte. Les éléments propres aux services Web, tels que *Service*, *Operation* et *Message*, sont représentés

dans le modèle ainsi que les mécanismes d'adaptation comme *Binding* ou *Triggering*. ContextUML met l'accent sur les mécanismes d'adaptation plutôt que sur la modélisation de contexte, et ne traite pas de la qualité de contexte. Notre travail, quant à lui, permet aux concepteurs de l'application d'exprimer des contextes de haut niveau tels que des situations déterminées à partir d'obervations de contexte distribuées et d'inclure dès le début du processus l'analyse de la QoC. Cela contribue à améliorer la sensibilité au contexte de l'application et par conséquent à améliorer la qualité des adaptations effectuées sur l'application.

MLContext [10] est un langage dédié à la modélisation de contexte. Il est étendu dans [11] pour intégrer la QoC dans l'expression des situations de contexte. Une situation est donc jugée comme pertinente si les données de contexte sur lesquelles elle repose satisfont à la QoC attendue. Le modèle que nous proposons prend également en considération la sensibilité au contexte de l'application métier.

Des états de l'art récents sur les systèmes sensibles au contexte ne traitent que succinctement des aspects qualité : [2] mentionne la nécessité pour un serveur de contexte de préciser les paramètres de qualité de service réseau, ce qui diffère de la QoC, et [15] indique que parmi tous les intergiciels passés en revue, seul MiddleWhere [20] traite explicitement de la qualité de contexte, mais limitée aux informations de localisation. Des travaux de recherche prenant en compte la qualité de contexte dans le cadre de l'intelligence ambiante commencent à apparaître mais n'en sont qu'aux prémisses ([5, 12, 18, 19, 21, 24, 26]) et la proposition de solutions intergicielles flexibles et performantes reste indispensable.

7 Conclusion

Cet article propose l'utilisation d'une approche dirigée par les modèles pour concevoir des applications sensibles au contexte. Nous nous appuyons sur une chaîne d'outillage complète et intégrée comprenant un méta-modèle de sensibilité au contexte (CA3M), un langage dédié à la gestion de contexte (COSMOS DSL) associé à un outil de génération de code (cosmos2code) des artefacts déployés à l'exécution.

Nous avons démontré nos contributions sur l'exemple d'une application de vente Flash. De même que pour de nombreuses applications mobiles, la connaissance de la localisation de l'utilisateur est essentielle pour une vente Flash, qui se déroule en un lieu précis et pendant une période de temps limitée. Nous considérons que cette information de localisation est au cœur de l'application. Nous donnons ainsi à l'utilisateur la maîtrise de sa localisation qui est collectée et mesurée uniquement sur le téléphone de l'utilisateur, ce qui garantit le respect de la vie privée. De plus, l'information de localisation pouvant être évaluée de manière plus ou moins précise, cela nécessite donc de prendre en compte les éventuelles incertitudes qui peuvent l'impacter. C'est pourquoi nous avons identifié trois critères de QoC comme étant particulièrement pertinents pour la localisation, à savoir l'exactitude, la fraîcheur et la confiance. D'autres paramètres de QoC pouvant être calculés par notre gestionnaire de contexte, des paramètres supplémentaires pourraient être ajoutés en tant que méta-données associées à la localisation.

Nous avons présenté les résultats des mesures de performances que nous avons réalisées, en termes de mémoire et de temps de calcul. Le coût de l'ajout de la gestion de la QoC est égale-

ment mesuré et se limite à quelques ms. Nous avons montré qu'une politique de contexte avec 120 nœuds sensibles à la QoC peut être traitée en moins de $100ms$ sur un téléphone mobile. Par ailleurs, nous avons pu créer une politique de près de 3000 nœuds avant d'épuiser les ressources mémoire du téléphone. Ceci permet de mettre en place des scenarios applicatifs très riches impliquant un grand nombre de nœuds de contexte et permet donc le développement d'applications ubiquitaires s'exécutant entièrement sur téléphone mobile.

Références

[1] Z. Abid, S. Chabridon, and D. Conan. A Framework for Quality of Context Management. In *Proc. 1st Int. Workshop on Quality of Context*, volume 5786 of *LNCS*, Stuttgart, Germany, June 2009. Springer-Verlag.

[2] M. Baldauf, S. Dustdar, and F. Rosenberg. A Survey on Context Aware Systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4) :263–277, 2007.

[3] T. Buchholz, A. Kupper, and M. Schiffers. Quality of context information : What it is and why we need it. In *10th Int. Workshop of the HP OpenView University Association (HPOVUA)*, Geneva, Switzerland, July 2003.

[4] D. Conan, R. Rouvoy, and L. Seinturier. Scalable Processing of Context Information with COSMOS. In *Proc. 6th IFIP International Conference on Distributed Applications and Interoperable Systems*, volume 4531 of *LNCS*, pages 210–224, Cyprus, June 2007. Springer-Verlag.

[5] W. Dargie and T. Hamann. A Distributed Architecture for Reasoning about a Higher-Level Context. In *Proc. IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, pages 268–275, June 2006.

[6] W. Dargie and T. Hamann. A distributed architecture for reasoning about a higher-level context. In *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, 2006. (WiMob'2006)*, pages 268– 275, 2006.

[7] A. Dey, G. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Special issue on context-aware computing in the Human-Computer Interaction Journal*, 16(2) :97–166, 2001.

[8] K. Henricksen and J. Indulska. Modelling and using Imperfect Context Information. In *Proc. 1st PerCom Workshop CoMoRea*, pages 33–37, Mar. 2004.

[9] K. Henricksen and J. Indulska. Developing context-aware pervasive computing applications : Models and approach. *Pervasive and Mobile Computing*, 2(1) :37–64, February 2006.

[10] J. Hoyos, J. Garcìa-Molina, and J. Bota. MLContext : A Context-Modeling Language for Context-Aware Systems. In *3rd DisCoTec Workshop on Context-aware Adaptation Mechanisms for Pervasive and Ubiquitous Services*, Amsterdam, Netherlands, June 2010.

[11] J. Hoyos, D. Preuveneers, J. García-Molina, and Y. Berbers. A DSL for Context Quality Modeling in Context-aware Applications. In *Ambient Intelligence - Software and Applications : 2nd International Symposium on Ambient Intelligence*, volume 92, pages 41–49. Springer-Verlag, April 2011.

[12] Y. Kim and K. Lee. A quality measurement method of context information in ubiquitous environments. *International Conference on Hybrid Information Technology*, 2 :576–581, 2006.

[13] G. Klyne and al. Composite Capability/Preference Profile (CC/PP) : Structure and vocabularies 2.0. W3C recommendation, april 2007.

[14] M. Krause and I. Hochstatter. Challenges in Modelling and Using Quality of Context (QoC). In T. M. et al., editor, *Mobility Aware Technologies and Applications*, volume 3744 of *LNCS*, pages 324–333. Springer-Verlag, 2005.

[15] Kristian Ellebæk Kjær. A survey of context-aware middleware. In *SE'07 : Proceedings of the 25th conference on IASTED International Multi-Conference*, pages 148–155, Anaheim, CA, USA, 2007. ACTA Press.

[16] A. Manzoor, H. Truong, and S. Dustdar. On the Evaluation of Quality of Context. In *IEEE EuroSCC, 3d European Conference on Smart Sensing and Context*, volume LNCS 5279. Springer, Oct. 2008.

[17] Object Management Group. Software & Systems Process Engineering Metamodel (SPEM) v2.0. formal/2008-04-01, Apr. 2008.

[18] D. Preuveneers and Y. Berbers. Architectural backpropagation support for managing ambiguous context in smart environments. In *Proc. 4th Universal Access in Human-Computer Interaction, Ambient Interaction, UAHCI 2007, Part of HCI'2007*, volume 4555 of *Lecture Notes in Computer Science (LNCS)*, pages 178–187. Springer-Verlag, July 2007.

[19] A. Ranganathan, J. Al-Muhtadi, and R. Campbell. Reasoning About Uncertain Contexts in Pervasive Computing Environments. *IEEE Pervasive Computing*, 3(2) :10–18, Apr. 2004.

[20] A. Ranganathan, J. Al-Muhtadi, S. Chetan, R. Campbell, and M. Mickunas. MiddleWhere : A Middleware for Location Awareness in Ubiquitous Computing Applications. In H.-A. Jacobsen, editor, *Proc. IFIP/ACM/USENIX Middleware*, volume 3231 of *LNCS*, pages 397–416, Toronto, Canada, Oct. 2004. Springer-Verlag.

[21] M. Razzaque, S. Dobson, and P. Nixon. Categorisation and modelling of quality in context information. In W. Roy Sterrit, S. Dobson, and M. Smirnov, editors, *Proceedings of the International Joint Conference on Artificial Intelligence, Workshop on AI and Autonomic Communications*, 2005.

[22] Q. Sheng and B. Benatallah. ContextUML : A UML-Based Modeling Language for Model-Driven Development of Context-Aware Web Services. In *Proc. 4th IEEE ICMB*, pages 206–212, Sydney, Australia, July 11-13 2005.

[23] C. Taconet, Z. Kazi-Aoul, M. Zaier, and D. Conan. CA3M : A Runtime Model and a Middleware for Dynamic Context Management. In R. Meersman, T. Dillon, and P. Herrero, editors, *Proc. 11th International Symposium on Distributed Objects and Applications*, volume 5870 of *LNCS*, pages 513–530, Vilamoura, Algarve, Portugal, Nov. 2009. Springer-Verlag.

[24] S. Tang, J. Yang, and Z. Wu. A context quality model for ubiquitous applications. In *IFIP International Conference on Network and Parallel Computing Workshops*, pages 282–287, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[25] X. H. Wang, D. Q. Zhang, T. Gu, and H. K. Pung. Ontology Based Context Modeling and Reasoning using OWL. In *Proc. 2nd IEEE International Conference on Pervasive Computing and Communications*, pages 18–22, Orlando, FL, USA, March 2004.

[26] A. Zimmermann. *Context Management and Personalization : A Tool Suite for Context- and User-Aware Computing*. PhD thesis, Westphalie Technical University - Fraunhofer Institute for Applied Information Technology, Germany, Oct. 2007.