A Model-driven Approach for the QoC-Awareness of Ubiquitous Applications

Sophie Chabridon, Zied Abid, Chantal Taconet and Denis Conan
Institut TELECOM, TELECOM SudParis, CNRS UMR Samovar
9 rue Charles Fourier, 91011 Évry cedex, France
Email: Firstname.Lastname@telecom-sudparis.eu

Abstract *- Context-aware ubiquitous applications are entering everyday life. However, their implementation remains challenging as there exist very few models and tools to guide application designers and developers in mastering the complexity of context information. This becomes even more crucial as the context of a user is by nature imperfect. One way to address this issue is to associate to context information some meta-data representing its quality. We propose a generic and extensible design process for context-aware applications taking into account the quality of context (QoC). We demonstrate its use on a prototype application for sending flash sale offers to mobile users. Through this example, we show how the addition of a context-awareness aspect in an application design process leverages the overall quality of mobile and ubiquitous applications.*

Keywords - MDE; context; QoC; pervasive applications

I. INTRODUCTION

For more than a decade, we have been witnessing a very fast evolution of mobile computing and ubiquitous services. Universal access to information is now an implicit requirement of distributed applications running on mobile devices as users expect data to be brought to them anywhere at anytime. However, their implementation remains challenging as there exist very few models and tools to guide application designers and developers in mastering the complexity of context information.

Context information was identified several years ago as a corner stone for mobile, ubiquitous or pervasive applications [8], [11]. Context managers have been proposed to infer high-level context data from low-level raw data extracted from several distributed sources such as operating systems, user profiles, knowledge bases and environment sensors [2], [8], [12], [25]. But only a few context-managers and consequently context-aware applications do pay attention to the Quality of the Context information (QoC). The importance of QoC as a first-class concept for context-aware services has first been identified by [5] defining it as "any information describing the quality of information that is used as context", and considering that it is intrinsic to the information as opposed to the computing process (e.g., quality of service) or to the hardware equipment (e.g., quality of device). The notion of worth has then been added to introduce the point of view of the targeted applications [19]. Context data are indeed known to be inherently uncertain due to the imperfection of physical sensors and the real world itself [3], [13]. As context data are by nature dynamic and very heterogeneous, they also tend to be incorrect, they indeed do not exactly reflect the real state of the modeled entity, inconsistent, with the risk to have contradictory information from different context sources, or incomplete when some aspects of the context are missing [15]. Therefore, taking into account the knowledge of the quality of context information appears to be essential to reach an effective and efficient context management. We propose a generic and extensible design process for context-aware applications taking into account the quality ofcontext (QoC) and a context manager which manages QoC. We demonstrate QoC modeling and QoC management through an implemented prototype Flash sale application for mobile users. Through this example, we show how the addition of a context-awareness aspect in the application design process leverages the overall quality of mobile and ubiquitous applications.

The organization of the paper is the following. We present in Section II the motivations of our work through a location-aware flash sale scenario. In Section III, we explain in detail the role of QoC in the design process, and for instance we present the resulting context-awareness model of the illustrating scenario. Next, in Section IV, we describe the implementation of our QoC-aware context-manager. In Section V, we discuss related work concerning both QoC management and context-awareness design before concluding the paper in Section VI.

II. MOTIVATING SCENARIO

In this section, we introduce the location-aware flash sale scenario for which we have developed a prototype application to illustrate the role and significance of QoC-aware context management. Essential to this scenario is the knowledge of the location of the mobile user and furthermore of the quality of this location information. We determine the location QoC through the management of additional context meta-data during the context management process.

At 10.00AM, Celina drives to the largest mall of the region for some shopping. She has her new mobile phone with GPS navigation, 3G and Wi-Fi communication. When she arrives on the outdoor parking lot of the mall, she receives a short message informing her of the availability of the new Flash sale offer service and inviting her to download this new application. As Celina is a frequent client of this mall, she has already registered with the mall office and has given her consumer profile mentioning her product preferences. Right after downloading the application, she receives a notification indicating that she has still 1 hour to benefit from a flash sale running in her favorite beauty shop. As Celina location accuracy is not good enough, only Celina's and the shop

positions are shown on the map. Later in the afternoon, Celina is inside the mall at the grocery store. She gets another alert for a flash sale offer proposed by a new shop that just opened and that she does not know yet. The flash sale is running for only 15 minutes. The radio coverage being currently very good, the location of Celina inside the mall can be determined with a very high quality level. She then receives on her phone a detailed and focused map of the mall indicating the path to the shop. All along the way, she is informed of the remaining time until the end of the flash sale and the map on her phone gets refreshed. Thanks to the guiding, she reaches the shop before the end of the flash sale.

Through this scenario, we show that it may be essential to provide applications with information on the quality of the context information they depend on. The context manager chooses the location which provides the best QoC, and furthermore, different services or different behaviors are delivered to the user according to the QoC.

III. DESIGNING CONTEXT-AWARENESS WITH QOC MANAGEMENT

We follow a model-driven approach to define the context-awareness of the application. As stipulated by the model-driven approach, designers write models conforming to meta-models. Since we target ubiquitous applications, besides classical application models, we need additional context-awareness models [34]. In this paper, we focus on the treatment of the QoC in the context-awareness management process. So, in this section, we start by describing the concepts manipulated in the context-awareness meta-model with a specific focus on QoC. The characteristics of this meta-model are to be independent of the application. We then describe the Flash sale offer application introduced in the scenario presented in Section II. Finally, we show how we use the context-awareness meta-model to derive the context-awareness model of the Flash sale application.

A. Context-awareness meta-model with QoC

The context-awareness meta-model is central to the design process we propose to build ubiquitous applications for mobile phones. We show in Figure 1 the subset of this meta-model which presents context-awareness contracts. The basis of our meta-model is the one presented in [34] in which a context- aware system is composed of observable entities, observables, and context-awareness contracts defined for these observables. The *observable entities* (not displayed in the view presented in Figure 1) are logical or physical elements to be observed. The context data types of the information observed on these entities are called *observables*. The contribution of this paper lies in the context-awareness contracts, and more especially, in the QoC aspects of these contracts.

A *context-awareness contract* defines a contract for a given observable and a given application to be fulfilled by the context management middleware service. In these contracts, through the *QoCRequirement* concept, the application designer specifies both the type of QoC (the list of *QoCParameters*) and the level of QoC (*QoCLevel*). The QoC parameters correspond to the meta-data we associate to context data. A first set of these parameters is directly collected from context sources, depending on the information available at the sources, and additional parameters can be computed at the acquisition step or even later during the inference process by the context management framework [1]. A *QoCLevel* defines the expected value of some QoC parameters. For instance, three levels of QoC (*high*, *medium* and *low*), which may be associated to a *QoCRequirement* for a given context-awareness contract.

We introduce three subclasses of context-awareness contracts that express three ways of modeling context-awareness requirements. Firstly, when an application designer wants to synchronously observe context data, she specifies an *ObservationContract*. A context-aware middleware, such as the one presented in [35], uses these contracts to instantiate observation artefacts able to provide the right level of QoC. Secondly, by using a *NotificationContract*, the application designer can specify subscriptions to events that occur for instance when a numerical observable value reaches a fixed threshold or when an enumerated observable value changes from one enumeration to another one with a given QoC. The condition is defined via the *triggerCondition* attribute. The designer is also asked to indicate the application entity that will receive the notifications. Note that observation and notification contracts managed by the context management framework do not deal with the adaptation decisions which are under the responsability of the application. Then, the third kind of context-awareness contracts, namely the *AdaptationContract*, puts in action the adaptation decisions taken by the application following the detection of a specific adaptation situation. A *ServiceContract* is an example of an *AdaptationContract* which allows to trigger the activation of a service. This kind of contract may require the support from the middleware layer to find, instantiate and deploy the required service. Through the contract, the chosen service may depend on the QoC level.

B. Flash sale offer application

We now discuss the design of the Flash sale offer application from Section II. One characteristic of this application is to exploit the knowledge of the quality of the context information. The application sends attractive messages to the mobile clients present at the mall to inform them about on-going special commercial offers that might interest them. This potential interest is evaluated by the system according to the context information available and to the quality of this context information. For this application, we have chosen three QoC parameters to characterize the

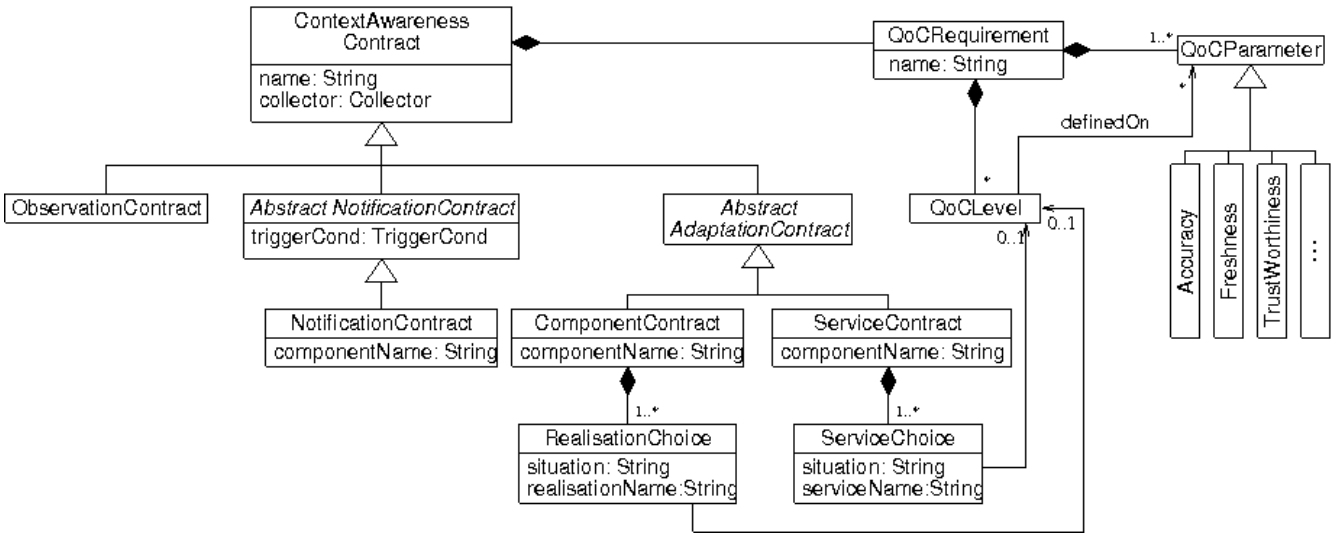

Figure 1. Meta-model of the context-awareness contract augmented with QoC.

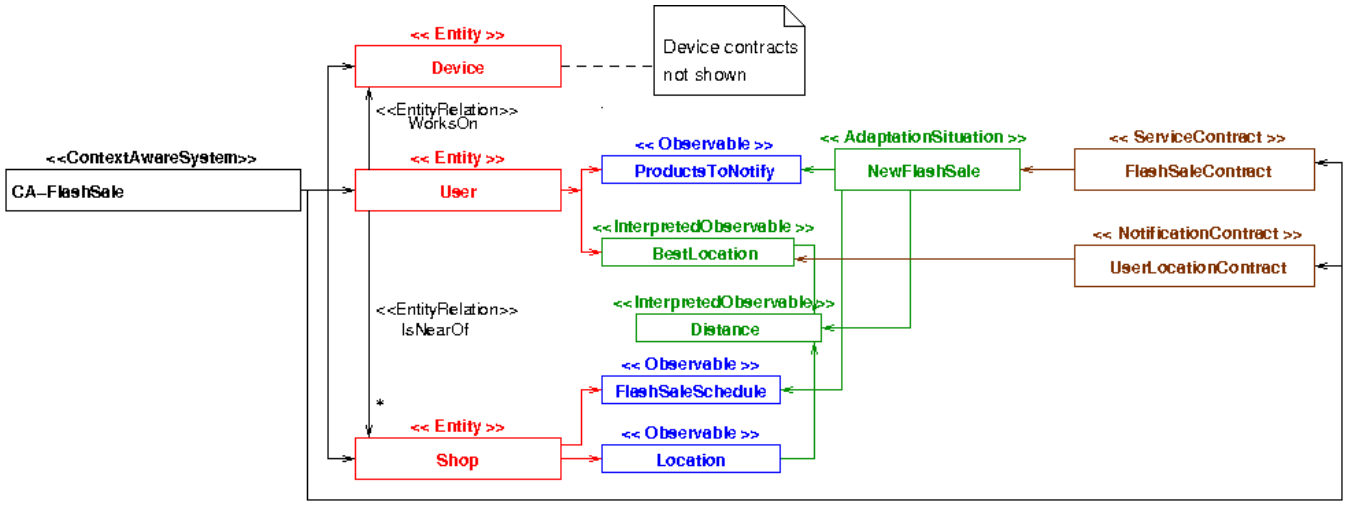

Figure 2. Model excerpt of the context-awareness of the Flash sale application

quality of the location information as discussed in [6]: the accuracy (expressed in *m*) indicating how close the estimated value is to the real one, the freshness or up-to-dateness (between 0 and 1) representing how old is the context information relatively to its lifetime, and the trustworthiness (between 0 and 1) that depends on how context data are collected or computed and on how much the context source can be trusted (see [1] for computation details). We manipulate multiple context data, some being static (i.e., values are given directly by the user) like the user's profile including their preferred products, the characteristics of the device, and some being dynamic such as the location and the movement speed of the user.

As the application is expected to be used outdoor (on a parking lot, on the street) or indoor (in a mall), we consider multiple methods of localization but with the constraint that they do not require the deployment of a specific infrastructure. As we want to privilege low-cost solutions, the quality of the location information becomes a key issue. It is important to master the QoC of the location in order to choose the best available location information. Depending on the capabilities of the mobile devices, several localization means can possibly be available. In our prototype implementation, we have considered three different technologies: GPS, Telephone Cellular network (GSM) and WLAN radio communication.

C. Modeling the context-awareness of the Flash sale offer application

The application scenario presented in Section II results in a context-awareness model conforming to the context-awareness meta-model. We show in Figure 2 the excerpt of this model focusing on the flash sale. The context manager takes advantage of the presence of the QoC attached to context information in order to provide the best location (*BestLocation*, i.e. location with the best level of QoC among

the available ones at a given time). We define two contracts related to the flash sale. The first one is the *UserLocationContract* notification contract. Thanks to this contract, the application component responsible for displaying the map of the mall receives the user location updates as soon as the user moves significantly. The second contract is the *FlashSaleContract* service contract. It allows a flash sale service to be triggered when there is a flash sale of interest in the user's vicinity. The Flash sale service is chosen according to the QoC level.

Figure 3 gives the definitions of three *QoCLevels*. The QoC level named *high* corresponds to the case where the level of each of the three QoC parameters that we consider in the scenario is high. The freshness is only 20% below the maximum; the accuracy indicates that the estimated position is less than 10m away from the real position; and the trustworthiness is larger than 90%. With the QoC level named *medium*, only the trustworthiness is below what is expected in the previous case. Finally, the level named *low* is even below. More levels could be defined, but by experience these three levels are realistic and represent sufficiently discriminating cases. The Flash sale service associated to the different QoC levels differs with the added value it provides to the user. When the QoC level is *high*, the application has a sufficient confidence in the estimation of the location of the user to guide them precisely. It displays an alert message with the distance and the remaining time to reach the shop where the flash sale takes place and also draws the route to follow on the map. When the QoC level is *medium*, the route is not displayed on the map. Finally, with a *low* QoC level, the distance to walk is not displayed for the user and only the remaining time is indicated.

```
quality high (freshness : Freshness,
              accuracy : Accuracy,
              trustworthiness : Trustworthiness) {
      freshness >= 0.8;
      accuracy < 10;
      trustworthiness >= 0,9; }

quality medium (freshness : Freshness,
              accuracy : Accuracy,
              trustworthiness : Trustworthiness) {
      freshness >= 0.8;
      accuracy < 10;
      trustworthiness < 0,9 ; }

quality low (freshness : Freshness,
              accuracy : Accuracy,
              trustworthiness : Trustworthiness) {
      freshness < 0.8;
      accuracy < 10;
      trustworthiness < 0,9; }
```

Figure 3. QoC levels for the Flash sale application

IV. MANAGING THE CONTEXT AND ITS ASSOCIATED QUALITY

We present in this section the implementation we have realized on Android mobile phones for the Flash sale application using the COSMOS framework [7]. As in the previous sections, we focus on QoC concerns. Section IV-A presents the context framework that we have complemented with QoC capabilities. The component-based orientation of the approach demonstrates the interest of the model-driven engineering (MDE) approach: As the designers of the context framework, we provide application designers with libraries of components for instance to process the context data and the QoC meta-data; the context framework is extensible so that other designers can develop and provide off-the-shelf components that get integrated into the context framework for instance to take into account new QoC parameters; application designers do not program inference treatments of the raw context data and QoC meta-data they collect but rather compose the architecture of the context manager that fits their needs. Section IV-B presents such a design: The application designer models the inference treatments both of the context data and of the QoC meta-data; these treatments are organized into a graph; the context framework being a process-oriented component-based context framework, the nodes of the graph correspond to components and the graph corresponds to an architecture. Finally, Section IV-C provides some details on the realization of the illustrative scenario on mobile phones.

A. Context and QoC management with COSMOS

COSMOS is a process-oriented context manager that collects raw context data from the different context sources and transforms them to higher-level context data. COSMOS can both be responsible for inferring high-level context data and situations, or supply other inference engines with low-level context data, for instance to ontology-based context managers [4]. The processing is organized into a graph representing a context policy which is a hierarchy of context nodes. These nodes are implemented as software components and can be shared across several context policies. They perform basic context-related operations (e.g., gathering data from a system or network probe, computing threshold or average values) and are assembled with a set of well-identified architectural design patterns [29]. A library of context operators allows designers to define new COSMOS nodes by composition.

Every context node of a context policy can be finely tuned in order to control the flow of context data and to control the operating system resources consumed for context processing, more especially threads and memory space. Therefore, COSMOS is available on a large number of mobile devices including J2ME phones and Android phones. COSMOS is implemented as an open source framework (http://picolibre.int-evry.fr/projects/cosmos). As shown in [1], COSMOS manages QoC thanks to a QoC context node composed of a QoCAwareOperator and QoCParameter components. We manipulate QoC separately from context data since we consider QoC as an additional concern of

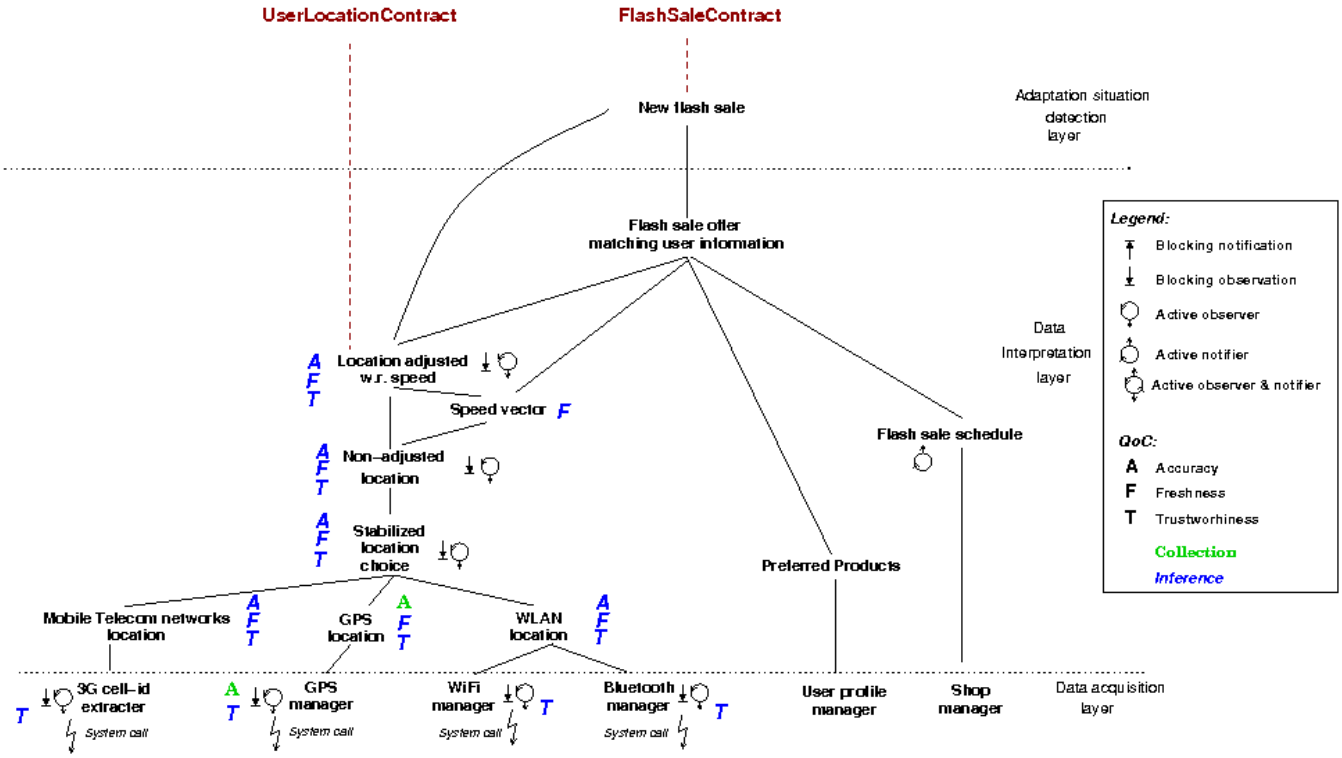

Figure 4. COSMOS context policy with QoC of the Flash sale application

context management. This allows for a flexible QoC management, which is activated only when necessary. This further opens the way for performance optimization like computing the QoC on a set of context data and not simply for each sensed data. Moreover, each QoCParameter component computes a specific QoC parameter such as accuracy, freshness, etc. As a consequence, for a given application, application designers select the relevant QoC parameters in the library of COSMOS QoCParameter components and compose their QoC context nodes.

B. COSMOS context policy of the Flash sale application

We show on Figure 4 the COSMOS context policy defined for the Flash sale application. It takes advantage of the available positioning technologies, such as cellular networks, wireless radio and GPS, and determines a stabilized location choice. This choice is guided by the QoC of the location information. As introduced in section III-B, we consider three QoC parameters for the location information, that are the accuracy (noted A), the freshness (noted F) and the trustworthiness (noted T). We distinguish the QoC parameters that are collected from context sources by the data acquisition layer from the QoC parameters that are computed by inference following the paths of the context policy. With satellite positioning technologies, an accuracy measure may be provided with the position. This is the case for Assisted-GPS [30]. Therefore, we indicate on Figure 4 that the accuracy parameter is collected by the GPS manager. For the other positioning technologies, the accuracy is measured via statistics derived from experimental observations. The trustworthiness is computed by the COSMOS framework and depends on the location source. For instance, with Wi-Fi communication, we derive a trustworthiness measure from the strength of the received radio signals [6]. The freshness parameter is also computed by the COSMOS framework at the time the context data is exploited, that is in the data interpretation layer. During the inference process, the Stabilized location choice context node determines the best location according to the values of the trustworthiness, the freshness and the accuracy parameters, in this order.

As users are mobile, we take into account in the context policy the speed of the movement of the user. The speed vector is deduced from the position history of the user as stored on the phone. This allows to adjust the location of the users at a given time and to determine how much time they might need to reach a given flash sale location. A freshness QoC parameter is computed by COSMOS and associated to the speed vector. The remaining part of the COSMOS context policy directly derives from the context-awareness model of the Flash sale application (see Figure 2). When a flash sale is scheduled in the short term (information coming from the Flash sale schedule node), the Flash sale offer context node makes use of the location of the user, the location of the shop where the flash sale is taking place and the user's movement speed to determine whether the flash sale offer matches the user's situation. If it is the case, the New flash sale adaptation situation is detected and the Flash sale service gets activated. Then, depending on the QoC

level of the user's location, the appropriate service is proposed to the user according to the Flash sale contract.

The context nodes of the context policy shown on Figure 4 are to be deployed entirely on the mobile phone of the user. We have designed the Flash sale offer application as an autonomous application that can entirely run on the mobile phone. Therefore, the deployment of the components managing the global map of the mall center and the list of the flash sales that are scheduled within the next hours occurs when the user arrives nearby the mall. The map of the mall center is displayed on the user's phone screen with a focus on the current location of the user. There is also a zoom feature allowing the user to change the scale of the map. This design removes any concern the users might have with regard to the preservation of their privacy. The Mall center information system does not get access to the location information of the different clients. The users control the knowledge of their own location which is stored only on their mobile phone.

C. Implementation on a mobile phone

We present in this subsection the prototype we implemented to validate the Flash sale application. In this section, we use the Siafu open source context simulator (http://siafusimulator.sourceforge.net/) to generate context information. By using a context simulator, we can better experiment and measure the adequacy of our prototype with more complex scenarios. We use it also for integration tests before the validation tests with end-users. We have prepared a map of the Évry 2 mall center and defined 17 product types on which flash sales can be proposed. Network overlays can also be defined and several access points can be positioned in the mall. Moreover, the context simulator allows to simulate very precisely the behavior of agents. Some anonymous agents have a random behavior and other named agents, like Celina, do have a well-defined behavior. Figure 5 shows the full map of the mall center that is deployed on the mobile phone. This map is then displayed on the screen with a sufficient zoom level to be readable, centered on the user's location.

We have developed the mobile application in Java and tested it on Android phones. Figure 6 shows a screen copy of the wireless toolkit phone emulator. As this is a prototype built for demonstration purpose, we display the different locations available on the phone which correspond actually to internal information. A real application would only show the chosen estimated location to the user. During the simulation, an alert message gets displayed on the phone screen when a Flash sale notification is received. Depending on the QoC level, additional information is given in order to guide the user towards the Flash sale.

We show on Figure 7 a screen copy of the case where a Flash sale offer occurs and the location information is of a *high* QoC level. This is the optimal case where full information on the flash sale is given to the user with the distance to the shop, the remaining time to go there and also the route to follow displayed with dotted lines.

Demonstrations were made to our industrial partners who welcomed the new shopping experience brought to the users.

Figure. 5. Map of the mall center

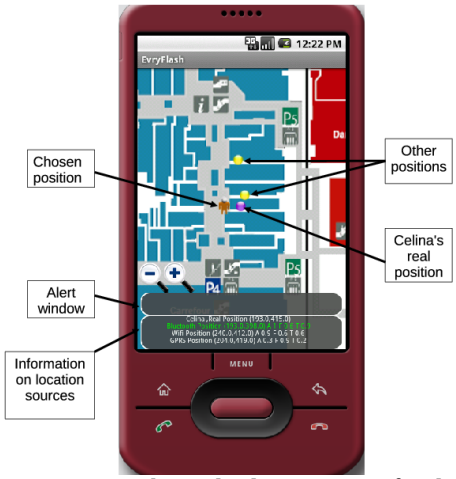

Figure. 6. The multiple positions of Celina

Figure. 7. Flash sale offer notification

As malls are becoming very large, any information on shops location is expected to be relevant and correct. Knowing the quality of the location information brings an added-value to the Flash sale application in terms of the quality of the user experience.

V. Related Work

Producing context-aware software is a very complex task. MDE for context-awareness and context-management is essential to ease the production of context-aware applications. We present below related works for context-awareness modeling and context management. This paper focusing on QoC, we highlight when relevant the QoC aspects of the related works.

Due to the variety of context data to be collected and analyzed, context management requires the support of abstract context modeling. The main families of context modeling are profiling (e.g. CC/PP [18]), databases (e.g., CML [14]), ontologies (e.g., CONON [36]) and MDE. Our work aims at using MDE for defining links between context modeling to express complex context situations, and context-awareness modeling to link context situations to application entities.

ContextUML [32] is one of the first domain specific model for context-awareness. It defines a meta-model for modeling context-awareness of web services. Consequently, web services elements such as Service, Operation and Message are represented in the model as well as related adaptation mechanisms of type Binding or Triggering. CAPPUCINE [24] describes an MDE approach for dynamically producing product lines according to context information. CAPPUCINE and ContextUML put the stress on adaptation mechanisms rather than on context modeling. Our work enables application designers to express interpreted context such as situations computed from distributed context observations and include the analysis of the QoC for driving the context-awareness of the application.

MLContext [16] is a DSL for modeling context. [17] presents an extension of this DSL which integrates the QoC in the expression of situations. A situation is then detected if the context data it depends on fulfill the required QoC. The model presented in our paper takes also into consideration the context-awareness aspect of the business application. Therefore, the middleware which uses the model is also able to connect detected situations to appropriate business services according to the QoC level.

[22] proposes an objective view of QoC, independent of any application requirement, and a subjective view of QoC considering its worth for a specific context requirement. Our context-awareness meta-model is also generic and independent of the applications, while the context-awareness model takes into account the requirements of a given application. However, the differentiating aspect of our work lies in a model-driven approach to guide application developers all along the software lifecycle from the design phase to the execution phase as the context-awareness meta-model and model can be accessed at runtime [35].

Concerning context management, many frameworks have been proposed and have become references in the domain of ubiquitous computing like the Context Toolkit [12], the Contextor [9], [28], Draco [26], MoCA [10] or MoCoA [31]. However, context management frameworks integrating and manipulating QoC are only beginning to appear and mainly concern location information.

Middlewhere [27] relies on three metrics for determining the quality of the location information: resolution, confidence and freshness. It proposes an uncertainty model based on a predicate representation of contexts allowing to use mechanisms such as probabilistic logic, fuzzy logic and Bayesian networks to fuse multiple sensor readings. However, the resulting quality of location information is not exposed to the applications and the models cannot easily be extended by application developers.

Nexus [20,23] is an open platform to ease the development of location-aware applications. It considers three quality aspects through degradation, consistency and trust. Nexus considers uncertainty as a key factor of location information and proposes a generic mathematical uncertainty model for position information [21]. This model is very powerful but requires applications to specify probabilities in order to perform position queries. We propose a more user-friendly solution where the framework informs the user of the obtained context quality rather than requiring the user to restrict the search domain.

The LOC8 framework [33] is a recent effort to provide application developers with easy access to location information. LOC8 defines a quality matrix consisting of granularity, frequency, coverage and a list of accuracy and precision pairs. LOC8 also relies on a sensor fusion method, with a default implementation based on fuzzy logic integrating the confidence on location data. While our work results from a similar effort to manipulate different sensor data and to expose the knowledge of its quality, we promote a fusion process that considers a larger set of quality criteria, and not only confidence.

VI. Conclusion

In this paper, we promote a model-driven approach for designing QoC-aware ubiquitous applications and introduce the QoC-enabled part of the COSMOS context manager. We demonstrate these two contributions through a Flash sale offers application. This application has been experimented as one the demonstrator applications of the CAPPUCINO project (http://www.cappucino.fr) in which were involved two large French chain stores. As for numerous mobile distributed applications, location-awareness is essential for the Flash sale application. We consider that location information requires specific care to deal with its inherent uncertainty and that applications need to have the knowledge of this uncertainty level. We identify accuracy, freshness and trustworthiness, as being the quality criteria that are particularly relevant for location information, but other QoC parameters are provided by our COSMOS context management framework and this list can be extended at will. With this prototype, we show that QoC may be used at different levels: at the context management level, for instance to choose the best location among several ones, and at the application level, for instance to trigger the appropriate service according to the current context situation and its QoC level. This justifies the model-driven approach that we have followed from the specification of context-awareness contracts to the design of the architecture of the context manager that runs on the mobile phone of the end-user.

To assert a given level of quality while using rapid prototyping approaches, we show how well the model-driven approach is adequate for designing context-aware ubiquitous applications. As future work, we will keep applying the model-driven approach to other concerns of the context-awareness management for instance to enhance the privacy of personal context data during the whole context management process in multi-scale networks (ambient environment, Internet, clouds) and over the Internet of things. In addition, since it is a process-oriented component-based context manager, we view COSMOS as the basis for distributing both processing and flows of context data and their QoC meta-data. This clearly opens new issues for QoC management.

REFERENCES

[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.

[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] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni. A survey of context modelling and reasoning techniques. Pervasive and Mobile Computing, Elsevier, 6(2):161-180, 2010.

[4] A. Bouzeghoub, C. Taconet, A. Jarraya, N.K. Do, and D. Conan. Complementarity of Process-oriented and Ontology-based Context Managers to Identify Situations. In Proc. 5th ICDIM, Thunder Bay, Canada, June 2010.

[5] 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 HPOVUA, Geneva, Switzerland, July 2003.

[6] S. Chabridon, C-C. Ngo, Z. Abid, D. Conan, C. Taconet, and A. Ozanne. Towards QoC-aware location-based services, Proc. 11th IFIP DAIS, vol. 6723 of LNCS, Reykjavik, Iceland, June 2011, Springer.

[7] D. Conan, R. Rouvoy, and L. Seinturier. Scalable Processing of Context Information with COSMOS. In Proc. 6th IFIP DAIS, volume 4531 of LNCS, pp. 210–224, Cyprus, June 2007. Springer.

[8] J. Coutaz, J.L. Crowley, S. Dobson, and D. Garlan. Context is key. CACM, 48(3):53, 2005.

[9] J. Coutaz and G. Rey. Foundations for a Theory of Contextors. In C. Kolski and J. Vanderdonckt, editors, Proc. 4th Int. Conf. on Computer-Aided Design of User Interfaces, pp. 13–34, Valenciennes (France), May 2002. Kluwer.

[10] R.C.A. da Rocha and M. Endler. Evolutionary and Efficient Context Management in Heterogeneous Environments. In Proc. 3rd Int. Workshop on Middleware for Pervasive and Ad-hoc Computing, Grenoble (France), Nov. 2005.

[11] A.K. Dey and G.D. Abowd. Towards a better understanding of context and context-awareness. In Proc. CHI Workshop on the what, who, where, when, and how of context-awareness, pp.304–307, 2000.

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

[13] K. Henricksen and J. Indulska. Modelling and using Imperfect Context Information. In Proc. 1st PerCom Workshop CoMoRea, pp. 33–37, March 2004.

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

[15] K. Henricksen, J. Indulska, and A. Rakotonirainy. Modeling Context Information in Pervasive Computing Systems. Proc. 1st IEEE PerCom, LNCS 2414, pp. 167–180, Zurich (Switzerland), Aug. 2002.

[16] J.R. Hoyos, J. Garca-Molina, and J.A. Bota. MLContext: A Context-Modeling Language for Context-Aware Systems. In 3rd DisCoTec CAMPUS Workshop, Amsterdam Pays-Bas, Jun. 2010.

[17] .J.R. Hoyos, D. Preuveneers, J.J. Garcia-Molina, and Y. Berbers. A DSL for Context Quality Modeling in Context-aware Applications, Proc. 2nd ISAMI, Springer, April 2011.

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

[19] M. Krause and I. Hochstatter. Challenges in Modelling and Using Quality of Context (QoC),;Mobility Aware Technologies and Applications, volume 3744 of LNCS, pp. 324–333. Springer, 2005.

[20] R. Lange and al. Making the World Wide Space Happen: New Challenges for the Nexus Context Platform. In Proc. 7th IEEE PerCom, pp. 300–303, Galveston, TX, USA, March 2009.

[21] R. Lange and al. On a generic uncertainty model for position information. In Proc. of 1st Int. Workshop on Quality of Context, pp. 76–87, Stuttgart, Germany, June 2009. Springer.

[22] A. Manzoor, H.-L. Truong, and S. Dustdar. Quality of Context: Models and Applications for Context-aware Systems in Pervasive Environments, The Knowledge Engineering Review, Special Issue on Web and Mobile Information Services, 2011.

[23] Nexus Team. Reference Model for the Quality of Context Information. Univ. Stuttgart, Feb. 2010.

[24] C. Parra, X. Blanc, and L. Duchien. Context Awareness for Dynamic Service-Oriented Product Lines. In Proc. 13th SPLC, San Francisco, CA, USA, August 2009.

[25] N. Paspallis, R. Rouvoy, P. Barone, G.A. Papadopoulos, F. Eliassen, and A. Mamelli. A Pluggable and Reconfigurable Architecture for a Context-aware Enabling Middleware System. In Proc. 10th DOA, LNCS 5331, pp. 553–570, Monterrey, Mexico, Nov. 2008. Springer.

[26] D. Preuveneers and Y. Berbers. Adaptive Context Management Using a Component-Based Approach, Proc. 5th IFIP DAIS, volume 3543 of LNCS, pp.14–26, Athens (Greece), June 2005. Springer.

[27] [27] A. Ranganathan, J. Al-Muhtadi, S. Chetan, R. Campbell, and M.D. Mickunas. MiddleWhere: A Middleware for Location Awareness in Ubiquitous Computing Applications. Proc. IFIP/ACM/USENIX Middleware, LNCS 3231, Toronto, Canada, Oct. 2004. Springer.

[28] G. Rey and J. Coutaz. The Contextor Infrastructure for Context-Aware Computing. In Proc. ECOOP Workshop on Component-oriented Approaches to Context-aware Computing, Oslo, June 2004.

[29] R. Rouvoy, D. Conan, and L. Seinturier. Software Architecture Patterns for a Context Processing Middleware Framework. IEEE DSO, 9(6), June 2008.

[30] N. Samama. Global Positioning: Technologies and Performance. Wiley-Interscience, 2008.

[31] A. Senart, R. Cunningham, M. Bouroche, N. O'Connor, V. Reynolds, and V. Cahill. MoCoA: Customisable Middleware for Context-Aware Mobile Applications. In Proc. 8th DOA, LNCS 4275, Montpellier (France), Nov. 2006. Springer.

[32] Q.Z. Sheng and B. Benatallah. ContextUML: A UML-Based Modeling Language for Model-Driven Development of Context-Aware Web Services. In Proc. 4th IEEE ICMB, Sydney, 2005.

[33] G. Stevenson, J. Ye, S. Dobson, and P. Nixon. LOC8: A Location Model and Extensible Framework for Programming with Location. IEEE Pervasive Computing, 9:28–37, 2010.

[34] C. Taconet and Z. Kazi-Aoul. Building Context-Awareness Models for Mobile Applications. JDIM, 8(2):78–87, April 2010.

[35] C. Taconet, Z. Kazi-Aoul, M. Zaier, and D. Conan. CA3M: A Runtime Model and a Middleware for Dynamic Context Management, Proc. 11th DOA, LNCS 5870, Portugal, Nov. 2009.

[36] X. H. Wang, D. Q. Zhang, T. Gu, and H. K. Pung. Ontology Based Context Modeling and Reasoning using OWL. In Proc. 2nd IEEE PerCom, pp.18–22, Orlando, FL, USA, March 2004.