Middleware Support for Quality of Context in Pervasive Context-Aware Systems

Kamran Sheikh, Maarten Wegdam, Marten van Sinderen
Center for Telematics and Information Technology, ASNA group, University of Twente,
P.O. Box 217, 7500 AE Enschede, Netherlands
{k.sheikh, wegdam, sinderen}@cs.utwente.nl

Abstract

Middleware support for pervasive context-aware systems relieves context-aware applications from dealing with the complexity of context-specific operations such as context acquisition, aggregation, reasoning and distribution. The middleware decouples applications from the underlying heterogeneous context sensors, and offers advantages such as rapid development of context-aware applications and efficient usage of the context sensors. Context sensors have inherent limitations with respect to the quality of the context information they produce. Without breaking the decoupling, the middleware needs to explicitly model and quantify this Quality of Context in order to allow application to adapt their behavior based on the Quality of Context, for efficiency reasons and to enable Quality-of-Context-aware privacy policies. In this paper we identify and define five Quality-of-Context indicators for context-aware middleware, and discuss different alternatives for their quantification. These Quality-of-Context indicators are: precision, freshness, spatial resolution, temporal resolution and probability of correctness.

1. Introduction

Context-aware systems offer personalized services to its users based on their context. We define 'context' as 'information that describes the situation of a human user either directly or indirectly'. This is a somewhat more limited definition than others tend to use (e.g., Dey et al. [2]), since we focus on human users contrary to entities in general. As context information represents real-world situations, it is associated with certain quality indicators, such as precision and freshness. Buchholz et al. are probably the first to discuss this, and they named this Quality of Context (QoC) [3].

Context sensors collect raw context information, which can be aggregated to improve the quality, or reasoned with to produce higher level context information. Although context-aware applications can be developed to be tightly coupled with the context sensors, several advantages of having a context-aware middleware layer that decouples sensors and applications have been identified [10][11][12]. There are, however, some issues that such a middleware inherits with the responsibility of context acquisition, aggregation, reasoning and distribution, in particular the need to explicitly consider QoC. As discussed in Section 4, existing work on context and context-aware middleware does not sufficiently address this issue.

The rest of this paper is structured as follows. In Section 2 we further motivate the role of QoC for context-aware systems. In Section 3 we present our five QoC indicators and discuss how they can be quantified. In Section 4 we compare our work with related work on QoC. Finally, the paper is concluded in Section 5.

2. Motivating Quality of Context

In this section we identify three main reasons why QoC is necessary for pervasive context-aware applications and middleware. Since this research is applied in the context-aware mobile health domain (see [14]), we take an application we're developing in this project as an example: tele-monitoring of epileptic patients. For this application vital signs, such as heart rate, are constantly monitored using wearable sensors and a body area network and sent over mobile or wireless networks to a back-end that analyses these vital signs. If this analysis indicates a likely epileptic seizure, the application notifies a voluntary or professional healthcare giver to check if the patient is ok. Who to notify, is based on the location of the patient and the location and availability of the involved healthcare givers.

The three main reasons to explicitly consider QoC are:

QoC-based application adaptation – context-aware applications by definition adapt their behavior based on the user's context. As noted before, context information is inherently imperfect, due to sensor limitations and other reasons. Real-world context-aware applications, therefore, should adapt their behavior to QoC information too. Since this adaptation is highly application dependent, the middleware should pass the QoC along with the context information to the application, in a consistent manner independent of receiving applications and low-level sensors. In the above tele-monitoring application, the QoC is important when selecting a healthcare giver, e.g., if two health-care givers are approximately equally far away from the patient, the healthcare giver whose status is 'available' with the highest probability is notified (before others).

Middleware efficiency - In a typical situation, there are several context sources available that can produce a

Proceedings of the Fifth Annual IEEE International Conference
on Pervasive Computing and Communications Workshops(PerComW'07)
0-7695-2788-4/07 $20.00 © 2007 IEEE

certain type of context, and there can still be cached context available. By comparing the required QoC with the QoC of the available context sources and the cached context information, the middleware can optimize the selection of which context source (not) to use. In the above tele-monitoring example, the location of the patient is required with high precision, since the caregiver needs to be directed to the patient. But the location of the caregiver may not be as precise (e.g., granularity of 100 meters or more), and a cached value is often good enough.

Users' privacy enforcement - There is a relationship between the privacy sensitiveness of context information and its QoC. The middleware must therefore provide users (context owners) with the means to limit the QoC information provided to different requesters. An obvious example is location, i.e. providing the complete address where a user is present is more privacy sensitive than providing only the name of the city. In the tele-monitoring scenario, healthcare givers may restrict the precision, with which the tele-monitoring application can determine their location, to a maximum of street level. This way it is more difficult to, for instance, infer that they are in a bar.

3. QoC Indicators and their Quantification

3.1. Precision

We define precision as the **'granularity with which context information describes a real world situation'**. For the purpose of quantification of precision, context information values can be classified into four types.

- *Boolean:* These are context types that can have only a true or a false value. This type of context information can not possess different levels of precision.
- *Numeric:* This is the type of context information that can be completely represented by one or more numerical value(s), e.g. speed, distance, temperature. Precision of these can be quantified in two different ways. (1) Ranges: The complete spectrum of values that the context variable can possibly attain is divided into a set of discrete ranges. Then, instead of providing the real value, the context information is said to lie in one of the ranges, e.g. available credit in account is \$0-1000, 1001-\$10000, \$10001 or above. (2) Significant figures: The number of significant figures in a numeric value provides a fairly accurate representation of its precision, e.g. a doctor requires patients' body temperature with at least three significant figures precision (such as 36.3°C).
- *Incremental sets:* Context types that have non-numerical, discrete values can be partitioned into a series of sets representing increasing information value (i.e. precision). For example, a European civic location can be completely stated using 4 of the standard notations in IETF RFC 4119 [4] namely, Country, A3 (city), A6-STS (street), HNO-HNS (House number with suffix). These can be arranged into a series which progressively represent higher information value.
  - (Country)
  - (Country, A3)
  - (Country, A3, A6-STS)
  - (Country, A3, A6-STS, HNO-HNS)
- *Weighted Sets:* Another strategy can be used to quantify the precision of context types that are represented by sets of discrete values but can not be arranged in a series of increasing precision. Each member of the set that comprises of all possible values of the context type can be assigned a weight that shows its information/privacy value. Then context owners can specify the maximum weight that can be provided to a requester to protect privacy. For example, the following weights, ranging from 1 to 10, can be assigned to the different parts of a person's identity; name=8, address=7, telephone number=7, type of employment=3, place of employment=4, etc. Users can then specify that items whose weight sums up to 10 or less may be provided to requesters. In the same way, applications can request context information with a minimum weight.

3.2. Freshness

We define freshness as **'the time that elapses between the determination of context information and its delivery to a requester'**. This is an important requirement as context that is too old may not be useful to a requester. Also, the freshness of context is directly proportional to its privacy sensitiveness, i.e. users may restrain recent context information from being provided to applications. The middleware would evaluate application requirements and user privacy preferences regarding freshness to decide on its caching strategy.

Quantifying freshness is a relatively simple task as it represents a period of time. Services can express their requirement of minimum freshness (i.e. maximum age) and users can state a minimum age (i.e. maximum freshness) of provisioned context using any suitable unit of time. For example, a service can constrain incoming user context to be no more that 1 hour old, while a user may restrict context information any newer than 30 minutes from be provided to the service. Then, context information that is between 30 and 60 minutes may be provided to the service.

3.3. Temporal Resolution

We define temporal resolution as **'the period of time to which a single instance of context information is applicable'**. This varies due to two main reasons. (1) The context source can be limited by its sampling rate [1] owing to which the occurrence of an event (context information) is reported with a certain delay. In this case,

Proceedings of the Fifth Annual IEEE International Conference
on Pervasive Computing and Communications Workshops(PerComW'07)
0-7695-2788-4/07 $20.00 © 2007 IEEE

the temporal resolution would represent the maximum delay in reporting the event due to the sampling rate, e.g. the temperature of a room collected every 8 hours is valid for a period of 8 hours after it is collected. (2) To protect user privacy, the time of collection of context information might be expressed with limited precision. For example, the time when an employee entered the office may be shown to colleagues with 'minutes' precision (e.g. 04-Sep-2006 0832hrs) but the time of leaving the office may be obfuscated to 'day' precision (e.g. 04-Sep-2006) for privacy reasons.

The difference between the two QoC indicators, temporal resolution and freshness, is illustrated in Figure 1. In the above example, the temporal resolution of context information about the employee leaving the office is manipulated, but the freshness can not be changed.

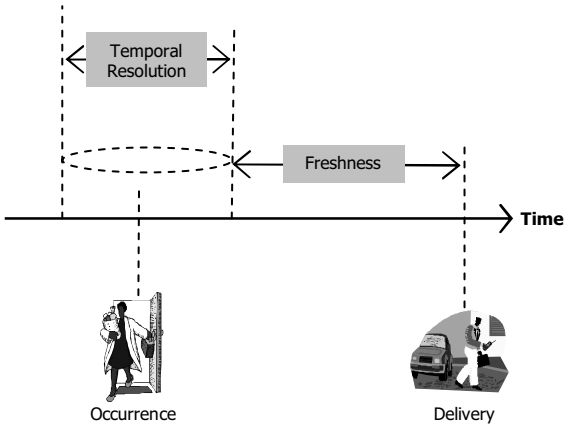

Figure 1. Freshness and temporal resolution

Notably both temporal resolution and freshness may be expressed implicitly, e.g. through a timestamp that shows when the context was determined. But they have to be quantified and stated explicitly in user privacy preferences, which set an upper bound on all QoC indicators, and in the application requirement which sets a lower bound on QoC.

Temporal resolution represents the *precision* of the time of occurrence/determination of a piece of context information. Therefore, the strategies presented in section 3.1 can be used for its quantification. Below we provide three examples.

- *Unix timestamp:* A Unix timestamp is the number of seconds that have elapsed from 01-Jan-1970 0000hrs to the second that the event has occurred. This can be considered as a numerical value and its precision can be quantified as explained earlier.
- *Standard date/time:* constitutes of year, month, day, hours, minutes and seconds. These can be arranged as a series of sets that represent increasing information value.
- *Time period:* Instead of depending on the particular format of the timestamp the context owner could just specify a minimum period of time in the range of which the time of occurrence/determination can be expressed. For example, when an application requests an event that occurred at 1530hrs and the context owner has specified a maximum temporal resolution of 8 hrs, the requester could be told that the event occurred between 1400hrs and 2200hrs. The bounds of this time period can be random as long as the minimum length is 8 hrs and the actual time of occurrence of the event falls within the range.

3.4. Spatial Resolution

We define spatial resolution as **'the precision with which the physical area, to which an instance of context information is applicable, is expressed'**. Spatial resolution is an important QoC indicator to protect user privacy as more localized information (higher spatial resolution) signifies higher privacy sensitiveness. For example, users may allow a building security system access to the number of people in their building but not in the room in which they are present. This will prevent the system from deducing whether the user is in a meeting. The quantification of spatial resolution is highly dependent on how 'space' is expressed in particular applications. For instance, when an application requests the location of a user from a mobile service provider, the spatial resolution may be expressed as a radius around a GPS coordinate. On the other hand, a building security system that keeps track of the number of people present in the building may provide this information with the spatial resolution of a room, a floor, a section of the building or the whole building. A number of location expressive models are available that can be used for this purpose, for example, IETF RFC 4119 [4] specifies additional civic location elements such as FLR (floor of building) and LOC (additional location information such as room number) that may be used to identify parts of a building. The Open Geography Markup Language [5] can be used to formally express more complex geographical topologies.

3.5. Probability of Correctness

We define probability of correctness as **'the probability that an instance of context accurately represents the corresponding real world situation, as assessed by the context source, at the time it was determined'**. Two reasons why the confidence of a context source in the accuracy of its collected context may be reduced are as follows. (1) Partial unavailability of the sensory infrastructure, e.g. if the location of a user is determined by a mobile service provider using only two Base Transceiver Stations (BTS) instead of three, the probability of this location being inaccurate is higher than if three BTSs are available for this purpose. (2) Availability of contradictory context information, e.g. the

fact that a user has used his key card to enter his office 10 seconds ago shows that his current location is his office, while his mobile service provider reports that his location is different. In this case it might be concluded that the user's location is his office but with low probability of correctness. Notably, probability of correctness is often inversely proportional to other QoC indicators including precision, spatial and temporal resolution. For example, the probability-of-correctness of context information that an employee entered the office would be much higher at a temporal resolution of hours (e.g. between 8-9AM) than seconds (e.g. 08:36:19AM) due to limitations in the capabilities of sensors. In the same way a user's location at country precision can be determined with higher probability of correctness than at street level by a mobile service provider.

Different applications require context information at different levels of probability of correctness, for example, a security service that opens the building gates when an employee is standing outside the door would need the employee's location with very high probability of correctness. On the other hand, a location-aware weather service can work with a location sample of lower probability of correctness. With the user's privacy protection point of view, probability of correctness of context information will play its part for 'plausible deniability' [6]. For example, users may be obliged to allow certain requesters access to their location but they can 'artificially' reduce its probability of correctness so that they can later deny being at certain locations. Lederer et al. [7] argue that users should not be expected to deviate from normal social practices just because the current technology works differently. Falsifying information about oneself is an established practice in social interactions, e.g. screening phone calls. Software applications that disseminate presence related information, such as instant messaging clients, allow their users to set their own presence status enabling them, for instance, to falsely set their status to 'away' while they are at their computer. Context-aware systems that take away this right from their users by disseminating accurate information to others at all times may seriously jeopardize their social acceptability.

Context sources can express their level of confidence in an instance of context information in several ways. Some examples are,

- As a percentage value between 0% and 100%.
- A more course grained approach would be to define levels such as low, medium and high.

4. Related Work

[3] is most related to the research described in this paper, and part of our work is based on this paper. A major difference however is that we go one step further in modeling QoC by discussing how our QoC indicators can be actually quantified. Besides this, there are differences in the motivation for the need for QoC, and the QoC indicators that are identified. In [3], six reasons are described that motivate having QoC as a notion of quality, in addition to 'quality of service' or 'quality of device'. These reasons somewhat correspond to ours, but we take a more generalized and categorized approach to motivate the need for QoC. From [3], *QoC agreements* and *Reconstructing service behavior* reasons are covered by our *QoC-based application adaptation* reason, since [3] limits them to using SLAs while there are other methods to express quality requirements, e.g., on a per request basis. From [3], *selection of context providers* and *adapting context refinement and dissemination* reasons are part of our *middleware efficiency* reason. The analysis presented in [3] is too restrictive in this regard as more methods for increasing middleware efficiency through QoC are possible, e.g., adapting sampling frequency of sensors to application QoC requirements. Finally, our *users' privacy protection* reason corresponds to *fine-grained privacy policies in* [3]. Besides the motivation for QoC, several QoC indicators are identified in [3]: *precision, probability of correctness, trustworthiness, resolution* and *up-to-dateness*. We based our definitions for precision, probability of correctness and resolution on the ones in [3], but concluded that resolution needs to be separated in spatial and temporal resolution. In addition, contrary to [3], we have discussed the relationship between our reasons for QoC and the actual identified QoC indicators, i.e. how do each of the indicators correspond to the reasons we need QoC.

In [1], six QoC indicators that represent the ambiguity introduced in context information due to sensor limitations are described. These indicators are *coverage, resolution, accuracy, repeatability, frequency* and *timeliness*. The notion of enforcing fine grained privacy policies using QoC and quantification of these indicators has not been explored. There are some basic differences between the QoC indicators chosen by [1] and our approach. We don't have *coverage* as a QoC indicator, since determining and expressing the complete range of context values that a source can deliver is very complex and does not provide a substantial advantage. We have introduced the notion of *temporal resolution* (Section 3.3) which is a broader concept than *sampling frequency,* proposed in [1]. Similarly, we have adopted *probability of correctness* (Section 3.5) over *repeatability* because the latter is only one way of measuring the *probability of correctness* of a context source, among others, e.g. checking for consistency with redundant context sources.

In [8], a middleware for context-aware applications that adopts most of its QoC indicators from [3] is proposed. The *Trustworthiness* indicator has been ignored while *Refresh Rate* has been added. In their middleware, the 'adaptation engine' can use any number of QoC indicators for activities such as service adaptation and discovery.

Proceedings of the Fifth Annual IEEE International Conference
on Pervasive Computing and Communications Workshops(PerComW'07)
0-7695-2788-4/07 $20.00 © 2007 IEEE

But how each of the QoC indicators that they have adopted from [3] can be used in the proposed middleware has not been described. [9] describes 'Quality of Information' as a design issue for pervasive systems but mentions only two indicators, *freshness* and *confidence*. In both [8] and [9], user's privacy protection point of view and quantification of QoC indicators have not been considered. Also, the QoC indicators presented are a subset of our QoC indicators.

Finally, in [6] a very good technique for enforcing privacy in context-aware systems using *capturing confidence* (probability of correctness) of sensors and *representational accuracy* (precision) of the identity of users has been presented. But these indicators are insufficient to ensure infrastructure efficiency or for QoC-aware application adaptation.

Summarizing this, to the best of our knowledge, we are the first to explore quantification of QoC indicators. In addition, we provide a more thorough motivation for QoC in pervasive context-aware middleware, and relate our motivation to the QoC indicators we have identified.

5. Conclusion and Future Work

During the development of the AWARENESS context-aware middleware [11] we concluded that there are three main reasons due to which a structured approach towards defining Quality-of-Context is necessary. These are *application adaptation, middleware efficiency* and *users' privacy enforcement*. We identified five indicators to represent the different aspects of QoC, presented precise definitions and discussed available techniques to quantify them.

In the future we plan to develop generic methodologies to accurately determine QoC values. While we expect that quality of raw context collected directly from sensors may be determined in a fairly straightforward manner, this does not apply to context that is the result of reasoning or aggregation of two or more instances of context information. The computation of QoC values for reasoned or aggregated context information will need to be part of the aggregation/reasoning algorithm. Another issue that will arise while determining QoC values would be to express values of Probability of Correctness (e.g. high, 84%) in a manner that allows all receiving parties to unequivocally comprehend what the originator means by it. This would be very difficult unless all of these parties have an agreement in advance. Values for the other indicators can be expressed in fairly unambiguous terms.

Trustworthiness [3] is a possible sixth QoC indicator, which we left out of scope for this paper because it is inherently more complex to quantify and support. This is because, unlike the other indicators, each entity in a context-aware system has a different view of how it trusts other entities. The same instance of context information would thus have a trustworthiness indicator that is different depending on entity receiving that instance of context information.

The techniques for quantification of the QoC indicators presented in this paper will be used to express (1) Application requirements (2) Context source capabilities and (3) Users' privacy preferences. These three values can then be used to negotiate the final level of QoC that is communicated to requesters. The method presented in the GeoPriv Common Policy format [13] would be very useful for such a negotiation.

6. References

[1] P. Gray and D. Salber. Modeling and Using Sensed Context Information in the design of Interactive Applications. In proceedings of 8th IFIP Working Conference on Engineering for Human-Computer Interaction (EHCI 01), Toronto, Canada, May 2001.

[2] A.K Dey, "Understanding and Using Context," J. Personal and Ubiquitous Computing, vol. 5, no. 1, Feb. 2001, pp. 4-7.

[3] T. Buchholz, A. Küpper, and M. Schiffers. Quality of context: What it is and why we need it. In Proceedings of the Workshop of the HP OpenView University Association 2003 (HPOVUA 2003), Geneva, 2003.

[4] IETF RFC (4119). A Presence-based GEOPRIV Location Object Format. http://www.ietf.org/rfc/rfc4119.txt?number=4119

[5] OpenGIS, "Open Geography Markup Language (GML) Implementation Specification", OGC 02-023r4, January 2003. http://www.opengeospatial.org/specs/?page=specs

[6] X. Jiang and J.A. Landay, "Modeling Privacy Control in Context-Aware Systems", in 1(3), pp. 59-63, IEEE Pervasive Computing, 2002.

[7] Lederer, S., J.I. Hong, A. Dey, and J.A. Landay, 2004. Personal Privacy through Understanding and Action: Five Pitfalls for Designers. Personal and Ubiquitous Computing 8, 6, 440 - 454.

[8] M. C. Huebscher and J. A. McCann. Adaptive middleware for context-aware applications in smart-homes. In Proceedings of the 2nd Workshop on Middleware for Pervasive and Ad-Hoc Computing (MPAC), Oct. 2004.

[9] Ebling, M., Hunt, G.D.H., Lei, H.: Issues for context services for pervasive computing. In: Middleware 2001 Workshop on Middleware for Mobile Computing, Heidelberg (2001)

[10] Jason I. Hong and James A. Landay. An infrastructure approach to context-aware computing. Human-Computer Interaction, 16(287–303), 2001.

[11] M.J. van Sinderen, A.T. van Halteren, M. Wegdam, H.B. Meeuwissen, E.H. Eertink, Supporting Context-aware Mobile Applications: an Infrastructure Approach, IEEE Communication Magazine, September 2006.

[12] Salber, D., Dey, A.K., and Abowd, G.D. The Context Toolkit: Aiding the development of context enabled applications. In Proceedings of CHI'99 (1999) 434-441.

[13] IETF Internet Draft, Common Policy: A Document Format for Expressing Privacy Preferences, http://tools.ietf.org/wg/geopriv/draft-ietf-geopriv-common-policy/

[14] T. Broens, A. van Halteren, M. van Sinderen, K. Wac, Towards an application framework for context-aware m-health applications, Proceedings of 11th Open European Summer School (EUNICE 2005), 6-8 July 2005, Colmenarejo, Spain.

Proceedings of the Fifth Annual IEEE International Conference
on Pervasive Computing and Communications Workshops(PerComW'07)
0-7695-2788-4/07 $20.00 © 2007 IEEE