Service-Centric Inference and Utilization of Confidence on Context

Atif Manzoor, Hong-Linh Truong, Christoph Dorn, Schahram Dustdar
Distributed Systems Group, Vienna University of Technology, Vienna, Austria
{*manzoor, truong, dorn, dustdar*}*@infosys.tuwien.ac.at*

***Abstract*—The inadequate quality of context forces the context consumers in pervasive environments to reason about the quality and relevance of context to be confident of its worth to perform their functionality. The additional task of analyzing large volumes of context drastically affects the performance of the context consumers to adjust to dynamically changing situations. A single value that presents the quality and relevance of context information tailored to the needs of a particular context consumer may release them from spending resources on context quality analysis and let them concentrate on their main task. In this paper we present a novel technique to combine different Quality of Context (QoC) metrics to infer the value of confidence on context. Our technique also considers the requirements of a particular context consumer regarding QoC metrics while confidence inference. Confidence on context is further provided to the context consumers to select high quality context and use the confidence in their functionality. We have successfully evaluated our approach using two context consumer services and user context collected from a smart home pervasive environment.**

***Keywords*-context confidence; quality of context metrics; smart homes;**

I. Introduction

Pervasive environments consist of a plethora of sensing and actuating devices that observe the environment and adapt it to support people in carrying out their every day life activities in an easy and natural way. Many virtual sensor services (VSSs) and context management services (CMSs) also work as an intermediator between these devices [4]. VSSs extract context from sensor data and CMSs distribute high level context to context consumer services (CCSs). CCSs control the actuators to interact with the environment, e.g., CCSc control window blinds, ambiance, and other appliances in a smart home environment [1]. Despite the criticality of consistent and coherent context for the effectiveness of the CCSs, continuously emerging situations that involve various sensing modalities and context extraction algorithms result in a high volume of imperfect context information. Furthermore, most of the time only a small subset of the overall generated context is relevant to the functionality of an individual CCS. Without proper knowledge of the applicability and the quality of a piece of context, CCSs are prone to make unsuitable decisions which the user may perceive as inconvenient for them. Consequently, CMSs also evaluate Quality of Context (QoC) metrics for that context and provide them to context consumers. The CCSs subsequently reason about context and corresponding QoC metrics to infer whether a particular context object is of good-enough quality and relevant for performing their functionality. This extra effort affects the ability of CCSs to make timely decisions. A single metric — *Confidence on Context* — aggregating multiple QoC metrics in a client-driven fashion, enables services to easily receive high quality context information that is relevant to perform their functionality without performing any additional reasoning.

Various research efforts have proposed metrics to indicate the confidence on context and provide it to context consumers [3]. So far most of the works suggested the application of only a single QoC metric, such as timeliness [13], [3], precision [11], and probability of correctness [2], to present confidence. Single-dimension QoC metrics have proven insufficient to model the confidence on context information for multiple purposes [7]. QoC metrics, such as reliability, timeliness, and significance, need to be combined considering the requirements of a specific CCS to ultimately infer an easy to use confidence on context metric. In this paper we describe the aggregation of different QoC metrics using a fuzzy inference system to calculate the confidence on context information. As different CCSs have different quality requirements, we provide a means to specify relevant QoC input, minimum quality thresholds, and relevant context items to achieve service-centric confidence on context information. The confidence on context information is further utilized to decide which context object should be provided to which CCSs. Our single metric indicating the quality and relevance of context considering the requirements of a particular service empowers service engineers to easily design CCSs without having to worry about quality and relevant context — a crucial factor to succeed in smart environments.

The remainder of this paper is organized as follows: Section II outlines the problem based on a motivating scenario. Section III presents the approach to infer confidence on context. Section IV details the experiments that evaluate our approach. Section V presents an overview of related work and compares them to our approach. Finally we conclude our work and present future steps in Section VI.

II. Motivational Scenario

Figure 1 illustrates our motivational scenario where physical sensor services, virtual sensor services, context management services, and context consumer services interact with

each other to adopt a smart home environment to continuously emerging situation as described in [1]. Appliances Control (AC) and Ambiance Management (AM) are two typical CCSs deployed in a smart home to control actuators. The task of AC is the proactive anticipation of the future use of appliances in the home kitchen and switch them on or off to support the user. For example, when a user wants to start cooking, the AC should switch on and preheat the oven. The AC also switches appliances off when they are no longer in use to save power and avoid any injuries. For example, the AC should switch off the oven when no cooking activity is currently detected or expected to happen in near future. The AM controls the home ambiance by tuning temperature, light, and background music. For example, the AM decreases luminosity and the volume of background music when the user is relaxing. Both of the services heavily depend on the users' current activity to effectively perform their functionality. Examples of typical activities in a home environment include relaxing, coffee time, and sandwich time. Sensors collect the data from the environment and various Context Management Services (CMS) subscribe to those sensors to obtain sensor data. Ultimately AC and AM subscribe to CMS to get the required context (user activity).

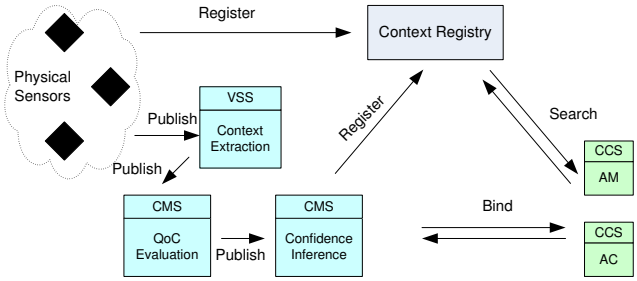

Figure 1. Scenario illustration

The QoC evaluator also analyzes the quality of high level context. The CMS then disseminates context annotated with QoC metrics to the subscribing CCS independent of the fact whether those context objects are of good-enough quality or are of relevance to the functionality performed by a particular service. AC and AM are interested only in a subset of the user's activities and suffice with different quality levels. For example, AC is interested in activities that the user performs in the kitchen area to keep the appliances ready for use. In contrast, AM is more interested in activities performed in other parts of the house. Large number of irrelevant context objects of inadequate quality increase the application's burden to reason on QoC metrics or context information itself and infer whether an underlying piece of context is of sufficient quality and relevant to carry out their task. This extra effort affects the main functionality of the services, i.e., to adapt to dynamically changing situations in the smart home.

A single metric that aggregates multiple QoC metrics — such as reliability, timeliness, and completeness indicates the relevance of context to the applications functionality — relieves the services from extra effort to reason on QoC metrics. A service merely needs to observe the value of the confidence on context metric to decide whether it needs to react to the received context information object. Alternatively, services may also provide the context manager with a threshold value to ensure that only those context objects are forwarded that have a sufficiently high confidence. Consequently the service only needs to react to context objects of high quality that are also relevant to the task of the service.

III. Confidence on Context

Confidence on context information is a multidimensional quantity that is used to present quality of context information from various aspects. *"Confidence on context indicates that how much context is free of errors, valid to use, and relevant to perform a specific task by a particular context consumer and liberates the context consumers from the extra effort to reason about the context or QoC metrics"*. Different QoC metrics such as reliability, timeliness, completeness, significance, and usability are also evaluated, along with extraction of high level context information, by the context producers and provided to context consumers, such as context-aware services in smart homes. These QoC metrics can be combined according to the requirements of a particular context consumer to infer the value of confidence on context information. In this section we will describe the sources of confidence on context, i.e., QoC metrics and the context consumer requirements, the confidence inference system that uses fuzzy logic to infer the value of confidence, and the dynamic rule generator that generates the rules according to context consumer requirements. These rules are used by our confidence inference system to infer the value of confidence on context.

A. QoC Metrics

Along with sensor data, sensors also provide meta-data about the sensor characteristics, such as accuracy, precision, and the granularity of sensor measurements. Context of a specific measurement, such as measurement time and sensor location, is also gathered by the lowest rung of QoC processing layer. *QoC Evaluator* uses this information is used to evaluate QoC metrics as numerical values in range $[0..1]$, where 0 indicate the lowest level of quality for that metric while 1 indicate the highest level of quality, and provided to context consumer as described in [6]. Many QoC metrics have been defined in different works. We considered the common quality concepts that have been used to define QoC metrics, such as temporal quality, correctness of context information, sensor observation level, amount of information contained by a context object, and trust on sensors that

have collected the sensor data. Considering these common quality concepts we have selected QoC metrics to use in our system. Here, we give a brief description of those QoC metrics that we have used in this work to infer confidence on context. *Timeliness* is defined as the belief in the validity of context information considering the time of collection of context. *Reliability* is the belief in correctness of context information. *Completeness* is the amount of information provided by a context object. *Significance* is the worth of context in a certain situation or far a particular service. Detail description about the evaluation of QoC metrics is presented in [6]. These QoC metrics are combined together to calculate confidence on context information. Confidence on context information is also calculated in range $[0..1]$, where 0 indicates the lowest level of confidence and 1 indicates the highest level of confidence.

B. Context Consumer Request

Every context consumer has different sets of requirements considering QoC metrics. For example, a context consumer service that has been installed in a smart home to deal with emergency situations is much more concerned about the reliability of context information than another service that maintains the home entertainment system. Therefore, considering the requirements of a specific consumer regarding QoC are indispensable to infer confidence. There must also be a simple mechanism for CCSs to specify their requirements. In our system CCSs specify their requirements by selecting the linguistic values of a QoC metric based fuzzy variable. For example, we define the linguistic values of *Reliability* based fuzzy variable as *Very High*, *High*, *Medium*, *Low*, *Very Low* and context consumers can indicate their requirements regarding *Reliability* of context using any of these linguistic values. Context consumer can also optionally specify a threshold value for the confidence on context. If a context consumer specifies the threshold value of confidence then CMS will only forward the context objects that have the confidence on context higher than the threshold value. Otherwise a context consumer will receive all context objects with value of confidence on context. Figure 2 shows a context consumer request for the context of type user's activity and QoC requirements *VeryHigh*, *Fresh*, and *Vital* for reliability, timeliness, and significance respectively, and threshold value of 0.8. The requirements are used to dynamically generate the rules to infer confidence on context information as described in Section III-C and Section III-D.

C. Confidence Inference System

We have used fuzzy logic [14] to infer the value of confidence on context as shown by the *Confidence Inference System* in Figure 3. The first step to use any fuzzy inference system is to define fuzzy variables corresponding to all input base numerical variables and the membership functions that maps the numerical value of base variables to linguistic

```
<ContextConsumerRequest
        contextType = ''UserActivity''>
        <QoCRequirement>
                <reliability>VeryHigh</reliability>
                <timeliness>Fresh</timeliness>
                <significance>Vital</significance>
        </QoCRequirement>
        <threshold>0.8</threshold>
</ContextConsumerRequest>
```

Figure 2. An example of context consumer request

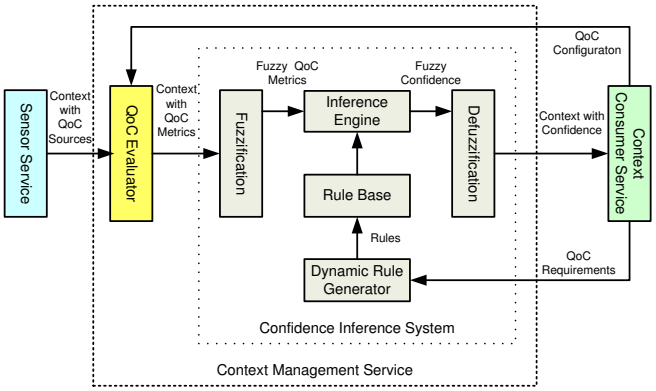

Figure 3. CMS with QoC Evaluator and Confidence Inference System

values of fuzzy variables. We define fuzzy variables and their membership function for every QoC metric that are used by the *Fuzzification* to map QoC metrics provided as an input to the system to their fuzzy values. The *Rule Base* holds the knowledge in the form of a set of rules to make the inference. These rules are generated by the *Dynamic Rule Generator* considering the requirements of the concerned application. Fuzzy values of QoC metrics are provided to the *Inference Engine* that uses the rules in the *Rule Base* with generalized modus ponen for making inference from input fuzzy variables QoC metrics to output fuzzy variable confidence. Defuzzification is the process of converting the fuzzy variable value back into a numerical value. The Center of Gravity (COG) is a popular technique to determine the numerical value from the fuzzy value and it is used to get the value of confidence as follows.

$$C = \frac{\int_{min}^{max} x\mu_C(x)dx}{\int_{min}^{max} \mu_C(x)dx} \quad (1)$$

where C is the numerical value of confidence that we will get after the process of defuzzification, x is the output variable and $\mu_C(x)$ is the confidence membership function for corresponding value of xi. The context manager passes the value of confidence along with context object to the context consumer.

D. Dynamic Rule Generator

The *Dynamic Rule Generator* takes QoC requirements, discussed in Section III-B, defined by a context consumer as an input to the system and dynamically generates the rules. These rules are added in the *Rule Base* and used by the *Inference Engine* to infer the value of confidence on context. Algorithm 1 provides the procedure for generating the rules. The algorithm starts with a nested *for* loop at Line 2 that will run for each fuzzy value of QoC metrics indicated in QoC requirements. The first block of Algorithm 1 at Lines 4-5 checks whether the current fuzzy value of a particular QoC metric is greater than or equal to required fuzzy value of that particular QoC metric. If the condition is true then that fuzzy value of QoC metric is mapped to maximum value of confidence on context. For example, for QoC requirement indicated in Figure 2 we will add the following rules to map significance of context to confidence.

IF `Significance is Vital` **THEN** `Confidence is VeryHigh`.

The second block of Algorithm 1 starting from Line 7 checks whether the current fuzzy value of a QoC metric is minimum for that QoC metric or not. If this condition is true then the algorithm generates the rule to map the minimum value of QoC metric to the minimum value of confidence on context. For example, the algorithm will generate the following rule to map the minimum value of significance of context to minimum value of confidence on context.

IF `Significance is Negligible` **THEN** `Confidence is VeryLow`.

The final block of Algorithm 1 starting from Line 10 maps all the remaining fuzzy values of QoC metrics to fuzzy values of confidence on context. These fuzzy values of QoC metric will be less than the QoC requirement set by a context consumer and greater than the minimum fuzzy value that can be assigned to that QoC metric. The algorithm gets the index of fuzzy value of confidence as the ratio of the difference between maximum and minimum fuzzy value of confidence to difference of current fuzzy value of a QoC metric value and fuzzy value of that QoC metric that has been assigned in QoC requirements. For example, for QoC requirements indicated in Figure 2, following rules will be generated to map significance of context information to confidence on context.

IF `Significance is MidVital` **THEN** `Confidence is High`

IF `Significance is MidNegligible` **THEN** `Confidence is Low`

Similarly, the algorithm generates the rule to map fuzzy values of all QoC metrics mentioned in QoC requirements. These rules are added to the *Rule Base*. The *Inference Engine* uses them to infer the value of confidence on context by combining the QoC metrics. As these rules are generated considering the QoC requirements set by a context consumer, the confidence value reflects quality and relevance of context for each particular context consumer.

Algorithm 1 Rule generation according to QoC requirement set by a context consumer

INPUT: QoCRequirement qocReq
OUTPUT: GeneratedRules gRules

1: GeneratedRules gRules ← null
2: **for** each QoCMetric qc_i in QoCRequirement **do**
3: **for** each FV (Fuzzy Value) fv_j in QoCMetric qc_i **do**
4: **if** $qc_i.fv_j >= qocReq.qc_i.FV$ **then**
5: gRules.addRule(IF qc_i is fv_j THEN Confidence is Confidence.FV.MAX)
6: **else**
7: **if** $qc_i.fv_j$ is MINIMUM **then**
8: gRules.addRule(IF qc_i is fv_j THEN Confidence is Confidence.FV.MIN)
9: **else**
10: i← $\frac{Confidence.FV.MAX.Index - Confidence.FV.MIN.Index}{qocReq.qc_i.FV.Index - qc_i.FV.Index}$
11: gRules.addRule(IF qc_i is fv_j THEN Confidence is Confidence.FV[i-1])
12: **end if**
13: **end if**
14: **end for**
15: **end for**
16: RETURN gRules

Table I
ACTIVITIES DESCRIPTION AND THEIR SINGLE RUN DURATION

Activities	*Description*	*Duration (seconds)*
Idle	Not performing any activity	583
Relaxing	Go outside and have a walk	157
Early morning	Move around in the room and casually check the objects	276
Coffee time	Prepare coffee with milk and sugar using coffee machine and drink it	129
Sandwich time	Prepare sandwich with bread, cheese, and salami using bread cutter, various knives, and plates and eat it	375
Clean up	Put objects used to original place or dish washer and cleanup the table	183

IV. EXPERIMENTS

In our experiments we evaluated the confidence on context information considering the requirements of two CCSs, AC and AM, as we have discussed in our motivating scenario in Section II. In the remaining section we describe the data that we used in the experiments. Then we show our evaluation of the QoC metrics inference of confidence on context considering the requirements of these two services and finally use confidence on context for context selection.

A. Data Description

In this experiment we used the data set that has been collected in the EU project OPPORTUNITY [12]. The data set was collected in a sensor-rich environment: a room simulating a studio flat with kitchen, deckchair, and outdoor access where subjects performed daily morning activities. 15 networked sensor systems with 72 sensors of 10 modalities were deployed integrated in the environment, objects, and on the body. The subjects have performed daily life morning

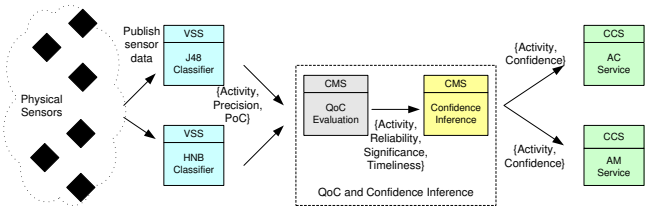

Figure 4. Experiments settings

time activities in this environment. They were instructed with the overall experimental protocol. They then executed 5 activity runs. These runs consisted of the early morning, coffee, sandwich, clean, and relax activities at higher level of abstraction. Table I provides a short description of those activities and their duration for a single run. Twelve subjects executed activities of daily living in this environment, yielding an average of 2 hours of effective data per subject, for a total twenty five hours of sensor data. In our experiments we used the action primitives extracted from the annotations performed on the data set to extract high level user activities. We have used five runs of a single subject from this data set. Considering better appearance and understanding, figures illustrating the results of our experiments present data evaluated for a chunk of thirty objects.

B. Experiment Setting

Figure 4 depicts the setting of our experiment. The environment is embedded with many physical sensors. They sense the environment and collect the data as described in Section IV-A. Physical sensors provide this data to virtual sensors. Virtual sensors classify the current user activity using machine learning classification algorithms as shown in Figure 4. In our experiment, the virtual sensors used machine learning algorithms J48 and Hidden Naive Bayes for the purpose of classification of user activity as presented in [8]. J48 is an algorithm that implements a C4.5 decision tree [10]. Hidden naive Bayes [15] is an extended form of naive Bayes and accommodates the attribute dependencies. We used their implementations available in WEKA [5]. Virtual sensors used the classification models of the aforementioned algorithms that have been trained with the sensor data. Precision and recall of classification methods to recognize user activity were also calculated to evaluate the accuracy of classification methods. Virtual sensors deployed as Web services provide the classified context (user activity) to the context management services that evaluate the QoC metrics and confidence on context and provide them to context consumer along with the context information items (user activity). We performed our experiments on a system having Intel Core 2 T5500 @1.66 GHz CPU and 2 GB RAM.

C. Evaluation of QoC metrics

In the first part of the experiment we have evaluated the QoC metrics of the context (user activity) recognized by the virtual sensors. The first QoC metric that we have evaluated was the *Reliability* of context. *Reliability* is defined as the belief in correctness of context information contained by a context object and is evaluated combining *Precision* and *Probability of Correctness (PoC)* of the context extracting process. WEKA implementation of J48 and HNB also provide PoC of context extracted from sensor data. *Precision* is the exactness of the context extracted from the context extraction process and is calculated as the ratio of true positives to total number of positives. Figure 5 shows the true positives, false positives and precision of context (activity) recognition chains using J48 and HNB classification algorithms. *PoC* is the probability of correctness of the predication made by a context (activity) recognition chain. We have combined *Precision* and *PoC* of context to calculate *Reliability* using following equation:

$$Reliability = 2 * \frac{Precision * PoC}{Precision + PoC} \quad (2)$$

The *Reliability* of context extracted from two different context recognition chain is shown in Figure 6.

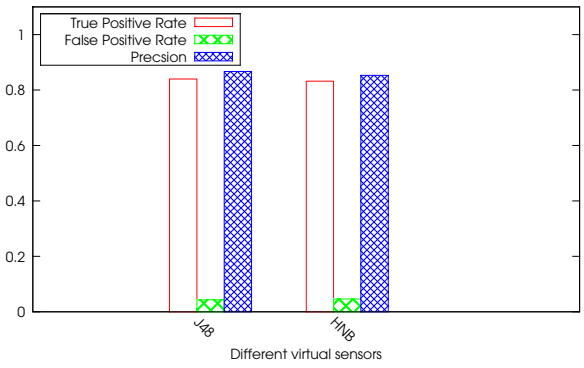

Figure 5. Precision of different virtual sensors

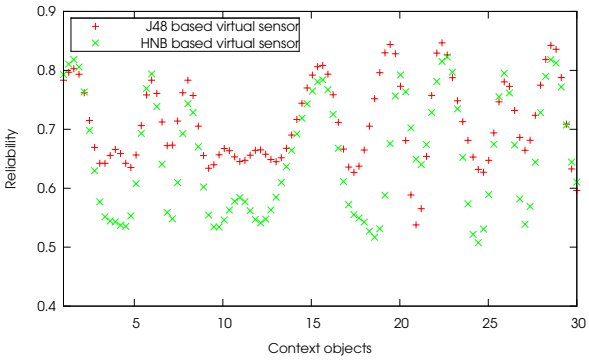

Figure 6. Reliability of context

The *Significance* of context is evaluated as the ratio of critical level of a particular context to maximum critical level that type of context can have. Applications configure the

critical level of different types of context according to their requirements as shown in Figure 3. We mentioned before that AC and AM services are interested in different values of context (user's activity). AC—that switch appliances on or off depending whether there is a chance that user will like to use these appliances— has assigned kitchen activities a higher critical value, so that when a context (user's activity) is recognized as one of kitchen activity, such as *Coffee_time*, *Sandwich_time*, and *Cleanup*, it has a high value of significance for the AC service. Similarly, the AM service is more interested in user's activities that have been performed outside kitchen area activities, such as *Idle*, *Relaxing*, and *Early Morning*, which have higher significance for this service. Figure 7 shows the significance of context for the different services.

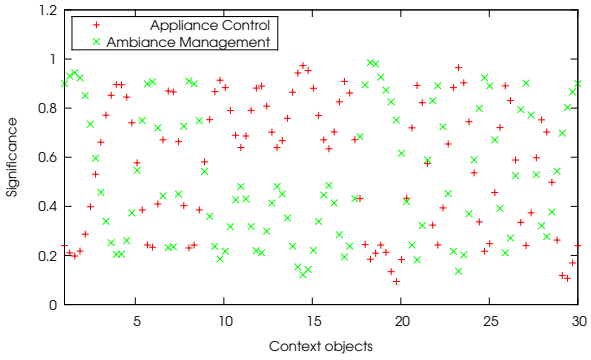

Figure 7. Significance of context

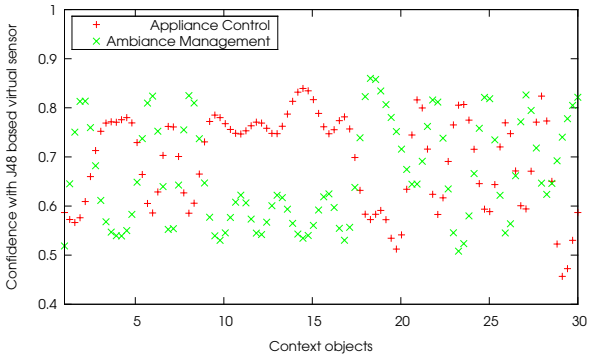

Figure 8. Confidence on context using J48

D. Evaluation of confidence

In our experiment QoC metrics *Reliability*, *Significance*, and *Timeliness* are used to evaluate confidence on context. QoC metrics having numerical value serve as the base variables for fuzzy QoC variables. Both context consumers specified QoC requirement values of *Reliability*, *Significance*, and *Timeliness* as *VeryHigh*, *Fresh*, and *Vital* respectively. We passed QoC metrics that have been evaluated in our experiments as described above to the *Confidence Inference System* that used the rules generated by the *Rule Engine* to infer the value of confidence for each context object considering the requirements of both applications. Figure 8 and Figure 9 show the confidence on context considering the requirements of both applications from context extraction chains using J48 and HNB classification algorithms. A careful examination of the values of confidence for different context objects in both of Figure 8 and Figure 9 depicts that the confidence on context is distinctively different for the two services for most of context objects. For example, context object 15 in Figure 8 and Figure 9—user activity *Coffee_time*—has higher value of confidence for AC than for AM. Considering higher value of confidence we can decide that this particular context object is more important to AC than AM to perform their functionality. This observation demonstrates that confidence on context successfully indicates the quality and relevance of context to both services.

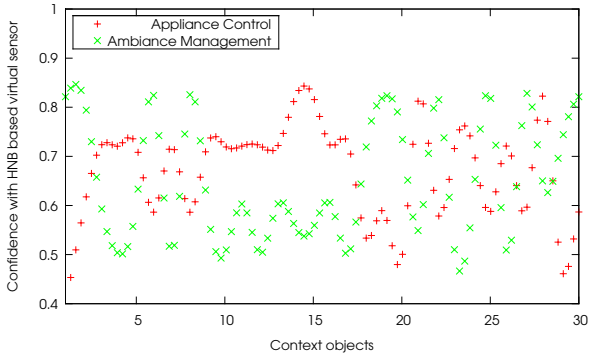

Figure 9. Confidence on context using HNB

E. Context consumers using confidence

In our experiments we have used the confidence on context information to filter the context objects that have been forwarded to services. These services can also provide the threshold value of the confidence on context information for interested context objects, i.e., context management service should only forward those context objects to applications that meet the required value of confidence. Figure 10 shows the number of context objects that have been provided to both applications using different threshold levels. The number of context object decreased as we increase the threshold level of confidence. Figure 11 shows the precision — as defined in information retrieval to indicate the relevant context objects — with rise in the threshold level of confidence. Precision is calculated as the proportion of true positives, i.e., objects that a particular application should receive, to total number of context objects received by an application. We can observe in Figure 11 that precision of the objects received increases with the rise in threshold level for confidence on context information. Rise in precision

depicts that the number of context objects that are not related to the functionality of a particular application decreases with increase in confidence threshold value.

Figure 12 shows the recall with increase in threshold level of confidence on context information. Recall is the fraction of context objects that are relevant to the functionality of a particular application and they have been retrieved successfully. Recall is calculated as the ratio of relevant context objects received by an application to the total relevant context objects generated. We observe in Figure 12 that the value of recall remains constant up to certain values of confidence and then it starts to decrease. This is because with low values of confidence threshold applications have been receiving all the context objects that have been generated. Increase in threshold decreased the number of context objects receive as shown in Figure 10 that eventually also decreased the number of relevant context objects received by the services. The service-specific confidence threshold value serves as a tradeoff between receiving all relevant but also unsuitable context items, and receiving most relevant context items and missing some of them.

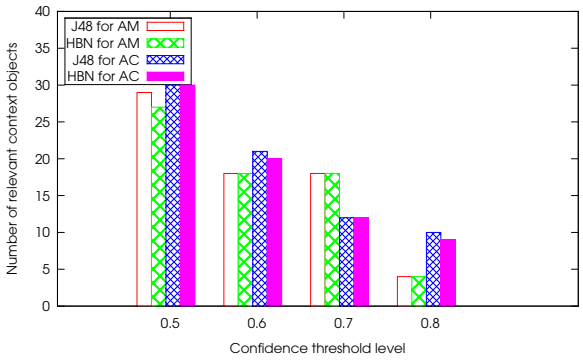

Figure 10. Number of context objects selected

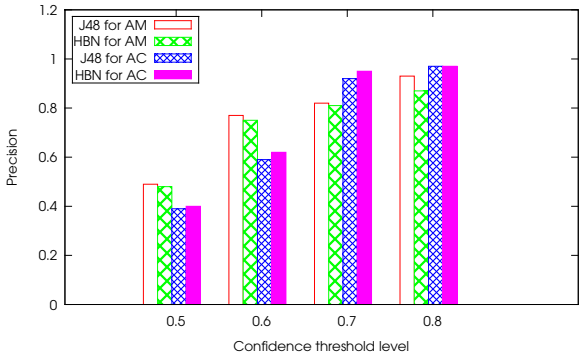

Figure 11. Precision of context selection

Figure 13 shows the time consumed by the context chain from sensor data generation to delivering the context object to context consumer services (CSSs) with different number

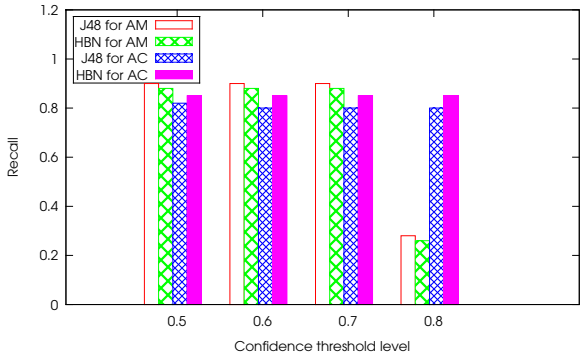

Figure 12. Recall of context selection

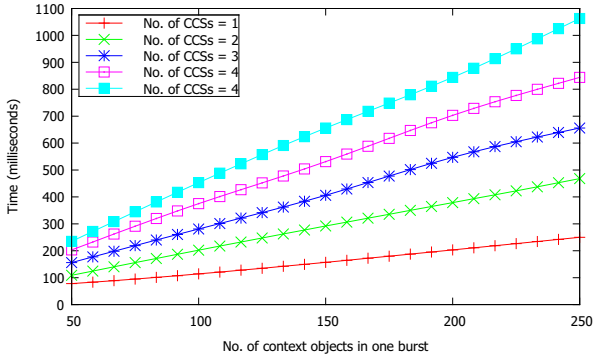

Figure 13. Time consumed in context delivery

of context objects in single burst and different number of CSSs subscribed for the context. In this experiment, we have taken CCSs that have different set of QoC requirements and context is distributed depending upon the confidence on context considering the requirements of a specific CCSs. Figure 13 shows that the context chain can accommodate a burst of two hundred and fifty context objects with CCSs of five different types of QoC requirements to distribute context within one second. In case we have more than two hundred and fifty context objects in a single burst and more than five CCSs with context delivery requirements of less than one second we may need more than one CMSs and better load balancing techniques to meet these requirements.

V. Related Work

Context information has been considered vague and ambiguous since the start of research in context-aware systems and different approaches have been proposed to present quality of context information as QoC metrics [6], [2], [3]. Previous research has also proposed to attach context information with confidence metadata to enable the applications in context-aware systems to select and use context with high value of confidence [3]. Most of the time, however, only a single quality metric measured confidence or uncertainty of context information. Bu et al. [3] proposed to indicate

confidence on context information based on the time of generation of context object. Schmidt [13] also used age of context information to indicate confidence and select among conflicting context information. In this case a most recently collected context object is always be selected and can result in ignoring context objects with highly reliability and accurate information generated at earlier stage in favor of less quality context object generated later.

Ranganathan et al. [11] used precision of sensor measurement to indicate uncertainty of context information. However, Mckeever et al. [9] proposed a model to combine different QoC metrics at context level to have the value of confidence that can be used at the application level but they do not forsee any mechanism to combine QoC metrics as required by the context consumer. Similarly, Brgulja et al. [2] also proposed to combine different QoC metrics to calculate confidence on context information without considering the context consumer requirements. Compared to these works, in this paper we have combined different QoC metrics to indicate confidence on context objects according to the requirements of application using context information. We have also presented a simple mechanism for higher level applications to indicate their requirements for quality of context objects. With the help of confidence metric, services can successfully select context objects without compromising the quality of context information from any aspect. The confidence value at application layer in context-aware systems also allows them to model quality in their logic and enhance their performance.

VI. Conclusion and Future Work

In this paper we presented a novel technique to infer confidence on context information by aggregating QoC metrics according to the requirements of a particular context consumer service for QoC. We used fuzzy logic for this purpose that suits quite well considering the uncertain and imperfect nature of context information and impreciseness of QoC metrics. The experiments demonstrated that confidence successfully depicts the worth of context for a specific context consumer considering quality and relevance of context for that particular context consumer. Our experiments also showed that confidence on context can be used to select the important context objects for a specific context consumer service to perform its task. The threshold value of confidence to select context objects can also be optimized as a tradeoff between precision and recall of relevant context objects. For the future, we plan to fuse context gathered from different sources to increase the confidence on context.

Acknowledgement

This work is supported by EU FP7 STREP Project SM4ALL (Smart hoMes for ALL), under Grant No. 224332. We also thank our colleagues working in EU FP7 FET project OPPORTUNITY, specially Alberto Calatroni, Claudia Villalonga, Daniel Roggen, and Gerhard Tröster, for their support to use the OPPORTUNITY smart home dataset.

References

[1] M. Aiello and S. Dustdar, "Are our homes ready for services? a domotic infrastructure based on the web service stack," *Pervasive and Mobile Computing*, vol. 4, pp. 506 – 525, 2008.

[2] N. Brgulja, R. Kusber, K. David, and M. Baumgarten, "Measuring the probability of correctness of contextual information in context aware systems," *IEEE International Symposium on Dependable, Autonomic and Secure Computing*, 2009.

[3] Y. Bu, T. Gu, X. Tao, J. Li, S. Chen, and J. Lu, "Managing quality of context in pervasive computing," in *QSIC '06: Proceedings of the Sixth International Conference on Quality Software*. IEEE Computer Society, 2006, pp. 193–200.

[4] X. Chu and R. Buyya, "Service oriented sensor web," in *Sensor Networks and Configuration*, N. P. Mahalik, Ed. Springer Berlin Heidelberg, 2007, pp. 51–74.

[5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, 2009.

[6] A. Manzoor, H.-L. Truong, and S. Dustdar, "On the evaluation of quality of context," in *EuroSSC '08: Proceedings of the 3rd European Conference on Smart Sensing and Context*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 140–153.

[7] A. Manzoor, H. L. Truong, and S. Dustdar, "Using quality of context to resolve conflicts in context-aware systems," in *Proceedings of First International Workshop on Quality of Context, Stuttgart, Germany*. Springer, 2009, pp. 144–155.

[8] A. Manzoor, C. Villalonga, A. Calatroni, H.-L. Truong, D. Roggen, S. Dustdar, and G. Tröster, "Identifying important action primitives for high level activity recognition," in *EuroSSC '10: Proceedings of the 5th European Conference on Smart Sensing and Context*. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 149–162.

[9] S. McKeever, J. Ye, L. Coyle, and S. Dobson, "Using dempster-shafer theory of evidence for situation inference," in *Proceedings of the 4th European conference on Smart Sensing and Context*. Springer, 2009, pp. 149–162.

[10] R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann Publishers, 1993.

[11] A. Ranganathan, J. Al-Muhtadi, and R. H. Campbell, "Reasoning about uncertain contexts in pervasive computing environments," *IEEE Pervasive Computing*, vol. 3, pp. 62–70, 2004.

[12] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Föorster, G. Tröoster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, J. Doppler, C. Holzmann, M. Kurz, G. Holl, R. Chavarriaga, M. Creatura, and J. del R. Millàn, "Collecting complex activity data sets in highly rich networked sensor environments," in *In Proceedings of the 7th International Conference on Networked Sensing Systems, INSS 2010*. IEEE, 2010.

[13] A. Schmidt, "Ontology-based user context management: The challenges of imperfection and time-dependence," in *Proceedings of On the Move to Meaningful Internet Systems 2006, Part I*. Springer, 2006, pp. 995–1011.

[14] L. A. Zadeh, "Fuzzy logic = computing with words," *IEEE Transactions on Fuzzy Systems*, vol. 4, no. 2, pp. 103–111, May 1996.

[15] H. Zhang, L. Jiang, and J. Su, "Hidden naive bayes," in *Twentieth National Conference on Artificial Intelligence*. AAAI Press, 2005, pp. 919–924.