



HAL
open science

Context Classification and Context Analysis Approach for collaborative Ubiquitous Systems

Nesrine Khabou, Ismael Bouassida Rodriguez, Mohamed Jmaiel

► **To cite this version:**

Nesrine Khabou, Ismael Bouassida Rodriguez, Mohamed Jmaiel. Context Classification and Context Analysis Approach for collaborative Ubiquitous Systems. 2012. hal-00704197

HAL Id: hal-00704197

<https://hal.science/hal-00704197>

Submitted on 4 Jun 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Context classification and context analysis approach for collaborative ubiquitous systems

Nesrine Khabou*[§], Ismael Bouassida Rodriguez*[†] and Mohamed Jmaiel*[‡]

*ReDCAD, University of Sfax, B.P. 1173, 3038 Sfax, Tunisia

[†]CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France

Univ de Toulouse, LAAS, F-31400 Toulouse, France

[§] Email: nesrine.khabou@gmail.com

[†] Email: bouassida@redcad.org

[‡] Email: mohamed.jmaiel@enis.rnu.tn

Abstract—Taking advantage of contextual information is very important for context aware applications especially in a mobile collaborative environment. This kind of applications adapt their behavior (self-adaptability property) according to the user needs, the available resources and the surrounding environment. This property is enforced by four phases; monitoring the application and its environment, analyzing context changes, planning adaptation actions and executing the planned actions to respond to context changes. In this paper, we focus and discuss the second phase. First, we propose a novel context classification which covers almost all of the context aspects. Then, we propose an analysis approach for collaborative ubiquitous systems. Our approach relies on different kinds of thresholds. Once relevant context changes occur, the context aware application will be notified to trigger its suitable process dynamically in order to deal with the changes.

Keywords-Ubiquitous computing, context awareness, adaptation, context analysis techniques, trend, peak, burst.

I. INTRODUCTION AND RELATED WORK

With the great growth of mobile devices such as laptops, palm tops and smart phones and the evolution of the communication technologies, a new communication paradigm has been emerged. In this paradigm, the ubiquitous computing concept has been introduced by Mark Weiser in 1991 [1]. The vision of ubiquitous computing adopts the idea of building smart environments where computing and communication are everywhere, being embedded in almost every surrounding object and can be accessed every time. In this paradigm, the behavior of the devices is unpredictable due to the environmental changes (temperature, pressure, noise, etc.), the execution context changes such as the network connection failure, the devices joining and leaving the network, etc. Hence, the need for sensing and detecting context changes to adapt the application behavior remains a challenge. This ability to sense and detect context changes is generally referred to as context awareness. Moreover, context awareness is central to ubiquitous computing and a key issue for making the devices aware of their situations. Therefore, the development of context aware applications

requires the fulfillment of different phases. First, context monitoring which aims at collecting context information from different sources. In fact, some studies have addressed this issue by proposing monitoring techniques such as the work achieved in [2].

Because the diversity and the heterogeneity of these information, it is suggested to classify them in order to facilitate the context manipulation. Many researchers have proposed several classifications of context. Some studies classify the context into two categories like [3],[4],[5]. Schmidt et al. state two context classes such as physical environment and human factors. Mitchell [3] divides the context into two classes. A personal environment formed by the user's interest, the user's location and an environmental context formed by the weather context. Other studies classify context to several categories like [6] and [7]. Schilit et al. [7] define the context as the constantly changing execution environment. In fact, the execution environment is the computing context formed by the available processors, the network capacity, and the connectivity; the user environment which contains the user's location, his social situation. Finally, the physical environment. In order to achieve a better understanding of the concept across a time span, Chen et al. [8] add time context such as the time of the day, the week, the month and the season of the year.

Context classification is an important step to discover the possible context easily, simplify the context manipulation including context analysis. In fact, the context analysis's purpose is to analyze context information and identify context changes. In the context changes detection research direction, several techniques and models are proposed in order to reveal the context changes. In fact, in [9], Cioara et al propose to use the context entropy concept for detecting the context changes and determining the degree of fulfilling a predefined set of policies. Moreover, context situation entropy defines the level of the system's self and execution environment disorder which is measured by evaluating the degree of respecting a set of policies. Hence, once the context entropy

exceeds a fixed threshold, then the system is in a critical state and it must execute adaptation actions. Although this approach allows self adaptation following context changes, it does not consider many context parameters to study as it is restricted to external parameters such as temperature, humidity and light etc.

In other studies, context changes are picked up by comparing a context value saved in a repository with a new context value. In fact, in [10], Zheng et al have addressed the issue of context change detection by proposing a context-aware middleware which conforms to the CORBA component model. The proposed middleware is composed of context aware services such as a context collector, a context interpreter, a context repository and a context analyzer. The latter is in charge of filtering and analyzing context information to determine relevant context changes and notifies the application afterwards. Furthermore, context filtering is based on a comparison of the context values saved in the context repository with the new context value in order to detect context changes.

The proposed middleware enables to save the scarce resources. In fact, the component deployment is performed “just-in-time”. However, this middleware does not specify context information to take into account. Another approach for dynamic context management is proposed in [11]. Indeed, Taconet et al [11] present CA3M, a context aware middleware, which enables applications to adapt their behavior by dynamically taking into account context changes.

They model the application by “entities”, which represent a physical or logical phenomenon (person, concept, etc.) and “observable”, which defines something to observe. For instance, a mobile device state is an example of an observable which may take a finite number of values (e.g low battery, almost low battery or normal battery). They consider that the change of an “observable” state or even the observation goes past a given threshold from the last notified value leads to a different application behavior.

In [12], Bouassida et al proposed a model driven approach for collaborative ubiquitous systems. In order to detect context changes, they specify predefined thresholds. Then, once context values remain below/under the threshold values, a notification is raised. Although this approach enables to detect instantly context changes, it may cause false detections as well as missing alarms by using fixed thresholds.

The rest of the paper is organized as follows. In section II, we introduce a case study named “Smart campus system”. Its purpose is to provide suitable services to users according to the context changes. The proposed approach is presented in section III. Our approach defines a novel context parameter classification which takes into account the parameter evolution behavior. Then, we present the analysis approach that allows an application to analyze context and detect context changes using thresholds. In section VI, we motivate and illustrate the feasibility of our approach through an

illustrative scenario. The last section concludes the paper and gives some directions for future work.

II. CASE STUDY: SMART CAMPUS

To motivate the use of our approach, we introduce in the following an example of a smart campus system illustrated in Figure 1. Smart campuses, with the ability to collect and analyze data are built in order to benefit the institution, the students and the teachers by providing services which facilitate interaction between the actors such as the teachers, the students and the technicians. For instance, every actor is equipped with a personal device (PDA, smart phone, PC, tablet, etc) and the smart campus system provides different services to actors based on their current situations. To maintain the collaboration among students and teachers, the actors’ devices need to be aware of their environment, their execution context, etc . Consequently, smart campuses contain an infrastructure that allows devices and systems to be monitored and adapted automatically according to the changing context. Our work focuses on several parameters such as temperature, pressure, position and light which need to be monitored in response to the changing context. Furthermore, a multitude of resources such as the available memory, the energy, the CPU load and available bandwidth should be considered and supervised to assess both the devices and the communication state and react in a convenient way either by switching devices or improving the links quality.

An efficient smart campus relies on the cooperation of smart devices. Indeed, the campus architecture depicted in the Figure 1 involves different kinds of participants. On the one side, three controlling servers called Smart Campus Servers (SCS) namely SCS1, SCS2 and SCS3, which are Ethernet-connected and equipped with important storage and high computational capabilities. On the other side, fixed and mobile actors are equipped with sensors, accelerometers, thermometers, GPS, cameras, mics, etc. These mobile actors are usually resource constrained in terms of memory, bandwidth and energy for example. Hence, a periodic monitoring by the controlling servers is needed in order to check their state which changes according to context. Three gateways called Smart Campus Gateways (SCG), SCG1, SCG2 and SCG3 implementing software interfaces are used to connect devices to the corresponding controlling servers to exchange relevant information. The campus space is composed of separate buildings. Each one corresponds to a department such as physics department (Department2), biology department (Department3), computer science department (Department1), etc. In each department, there is a number of classrooms and offices. Each office and classroom is equipped with devices such as smart air-conditioner, a presence sensor and a luminosity sensor used to supervise the environment parameters of the department. Other devices (mobile phones, tablets, PDAs, etc.) are carried by mobile actors such as students and teachers.

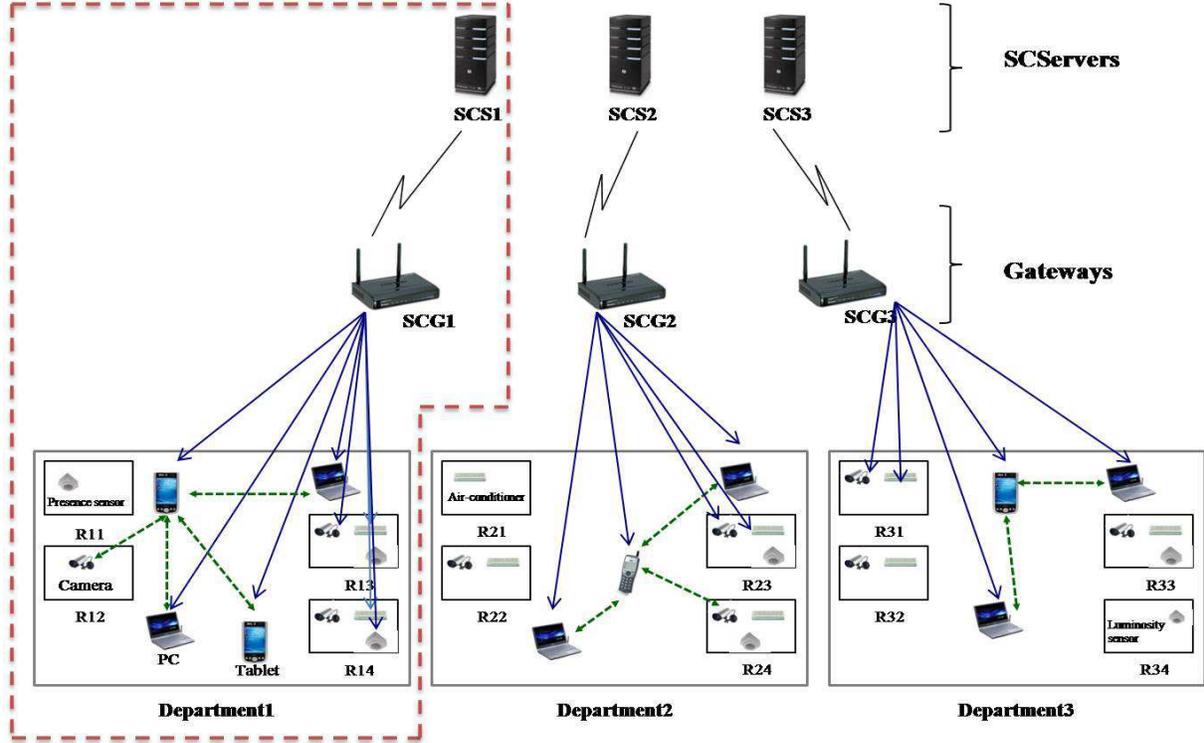


Figure 1. Case study: Smart campus

Because of the complexity of the interaction between the different entities, we propose to study a part of the smart campus highlighted in the Figure 1(dashed line). It consists in a gateway connected to both the SCS1 and the devices located in the Department1. In fact, the SCSs hold analysis algorithms and so they are able to receive monitored data parameters and to analyze them.

III. THE PROPOSED APPROACH

Towards detecting context changes, we propose a novel analysis approach which aims at raising notifications if context changes occur. In order to achieve the mentioned goal, the application should perform a set of tasks. First, it should collect information, then analyze contextual information. Finally, the application should trigger the appropriate adaptation actions. Moreover, our approach depicted in Figure 2 involves two main components. A context provider responsible for providing context information. A second component namely a context manager is in charge of managing and handling context information. Furthermore, the activity of the context manager is divided into four modules.

- A context collector module: Responsible for managing sensors and collecting context information.
- A context interpreter module: It aims at processing contextual information and shielding off the information which is not useful to the application.

- A context database module: Filtered and processed contextual data are stored into the context database module for further retrieval.
- A context analyzer module: Responsible for analyzing stored contextual information and detecting context changes.

In our work, we focus on the context analyzer module. It retrieves context information from the context database module, analyzes it and identifies the context changes using thresholds. Afterwards, the context analyzer module notifies the application in order to execute the appropriate adaptation actions.

- For instance, all SCSs namely SCS1, SCS2 and SCS3 mentioned in section II (case study) are equipped by context databases and context analyzer modules. In fact, the SCSs need a set of context information parameters retrieved from the context database module in order to analyze them and generate appropriate notifications.
- The gateways SCG1, SCG2 and SCG3, when forwarding information from the devices to the SCS, filter the received data. For that reason, the gateways used in the smart campus application hold context interpreter modules. Then, only meaningful contextual information are stored in the context database modules.
- Each mobile device such as a tablet, a PDA, a smart phone is equipped with a context collector module that

enable to capture context information such as light, temperature and forward the obtained data to the SCSs through the gateways.

For example, an actor equipped with its tablet holding a context collector (GPS for example) reports periodically the actor’s position to the SCS1 through SCG1 as illustrated in Figure 1. The SCS1 receives the filtered data from SCG1, stores it in the context database module. Later, these information are analyzed and the SCS1 takes the appropriate decision. For instance, according to the position of the actor equipped with a tablet as depicted in Figure 1, the SCS1 decides to switch off the lights in R14.

- Towards controlling their own state, mobile devices hold also a context analyzer module. In fact, each mobile device uses its context analyzer module to identify its resource context changes namely the battery level, the memory consumption, the CPU load, etc.

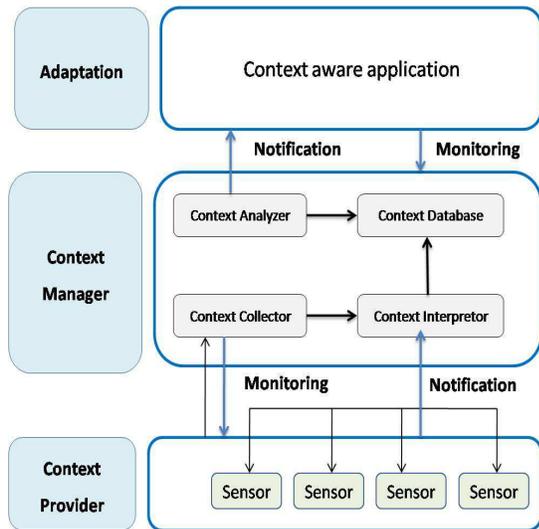


Figure 2. The proposed approach

A. Context parameter classification

With a wide range of context information, context parameters should be classified into categories towards using context effectively. We propose two classifications. A resource based classification and a quantitative based one.

a) Resource centric classification:

A first classification concerns resource context. In fact, resource context include the constraints imposed by the environment. We propose to classify resource context into two classes. Resource context related to the devices and resource context related to the network communication.

- **Resource context related to the device**

This class corresponds to the resources which characterize the device such as the available memory, the CPU frequency, the CPU load and the battery level.

- **Resource context related to the network communication**

This class deals with the resources that identify the network communications such the loss rate, the network bandwidth, the network connectivity, the communication link load and the latency.

Despite the simplicity and the ease of the resource based classification, it remains specific to some context parameter types. In fact, this classification is restricted only to computing context described in section I. Since our work addresses a large amount of context parameters, our interest focuses on quantitative context parameters

b) Quantitative centric classification:

Since context parameters evolution is highly dynamic especially in ubiquitous environments, we propose to divide the quantitative context parameters into three families according to the context parameter evolution. Indeed, this classification covers almost all the context parameter types. Three classes are identified. Parameters whose behavior is characterized by a trend, parameters whose behavior is characterized by peaks and parameters whose behavior is characterized by bursts. Afterwards, we present each family separately.

1) *Parameters whose behavior is characterized by a trend:* A trend is defined as the “long term” movement in a time series without irregular and abrupt effects. It is the underlying direction (an upward or downward trend) in a time series. This class deals with parameters whose behavior evolves in a constant way within time- That is the parameter model roughly coincides to a trendline as illustrated in Figure 3.

- For example, each mobile device such as PDA, mobile phone, etc. mentioned in section II periodically monitors its resource state. In fact, for each mobile device, the battery level, the memory consumption evolve in a relatively constant way. Besides, the latency defined as the time elapsed between the delivery of the message by the gateway and its reception by the controlling server can be modeled by a trend.

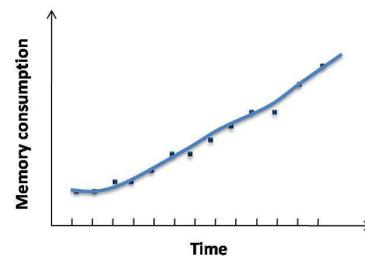


Figure 3. Memory consumption evolution within time

2) *Parameters whose behavior is characterized by peaks:* Peaks are defined as high values with sharp rise followed quickly by sharp fall implying a narrow period width [13]. Otherwise, we define a peak “ P ” by its amplitude “ A_p ” and its period “ T ” as depicted in Figure 4. In fact, the peak is a very narrow period of high values- That is, its amplitude “ A_p ” exceeds for n times the average amplitude “ A_{av} ” of the time series formed by the parameter behavior. Hence, the parameters belonging to this class are those characterized by abrupt changes in their behavior as depicted in the Figure 4. Otherwise, a peak is defined by the following formulas:

$$\begin{cases} A_p = n \times A_{av} \\ T \leq m \times TimeUnit \end{cases}$$

Where A_p defines the peak’s amplitude, A_{av} defines the average amplitude of the parameter behavior, n and m are constants fixed by the application designers.

The SCSs receive the monitored data from the mobile devices in order to analyze them. For that reason, the SCSs CPU load can rise suddenly reaching high values after a high rate of clients requests. Hence, the CPU load especially of the SCSs is modeled by a peaked function. The link load as well as the available bandwidth belong also to this context family.

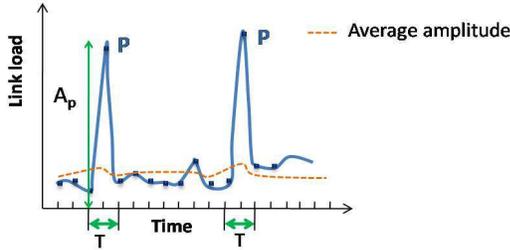


Figure 4. Link load evolution within time

3) *Parameters whose behavior is characterized by a burst:* A burst consists on a relatively wide contiguous region of values. Otherwise, a burst is defined as a large number of occurring events [14]. As depicted in the Figure 5, we can model the wide region by an ON period and the other by an OFF behavior [15]. The ON-period models a single flow such as the transfer of a single web page, and the OFF-period models the user’s thinking time. Noting that the ON-period and the OFF-period are strictly alternating. The message number received during an ON-period can be modeled as a burst.

Since our interest focuses on context parameters evolution behavior, in the following we detail the threshold calculation for the quantitative centric classification which takes into account the parameter evolution behavior.

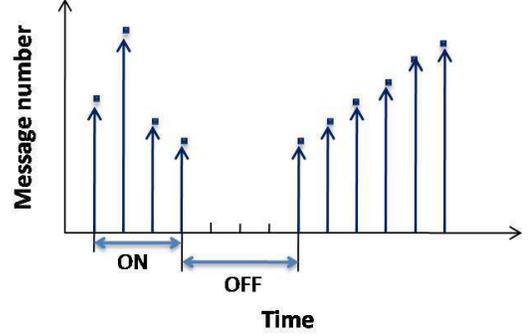


Figure 5. A bursty traffic model

B. Threshold calculation

We have proposed a parameter context classification which takes into account the parameter evolution behavior. As we base our approach on thresholds for detecting context changes, we present threshold calculation for each parameter class.

For each class parameter, we propose to assign a number of thresholds. For each threshold, we attribute a notification or a signal. The latter is defined by an expert and it corresponds to an emergency degree or a need for adaptation that depends on the need of an expert or the application itself. The threshold values can be either predefined or adaptive ones.

Afterwards, we explain some element about threshold calculation for each parameter class detailed previously.

1) *Threshold calculation for the parameters whose evolution behavior is characterized by a trend:* For this class, the parameter evolution behavior is described by a trend. In order to avoid false detections as well as missing alarms, we need to define thresholds which are uncorrelated with the parameter behavior. A notification is raised when the parameter behavior crosses the threshold as depicted in Figure 6.

Different kinds of thresholds can be applied for this class, such as fixed thresholds, adaptive thresholds and step function thresholds.

For instance, fixed thresholds may be defined by the application designers according to the parameter characteristic. Then, a notification is raised once the parameter value crosses the threshold as depicted in the Figure 6(a).

For the adaptive threshold denoted in the Figure 6(b), mathematical methods can be applied in order to update threshold values at runtime such as the Exponential Weighted Moving Average technique used in [16]. However, for this kind of resource context characterized by a trend, adaptive threshold must be uncorrelated with the resource evolution behavior in order to avoid false detections and missing alarms. Finally, for the step function threshold described in the Figure 6(c), thresholds are defined per period

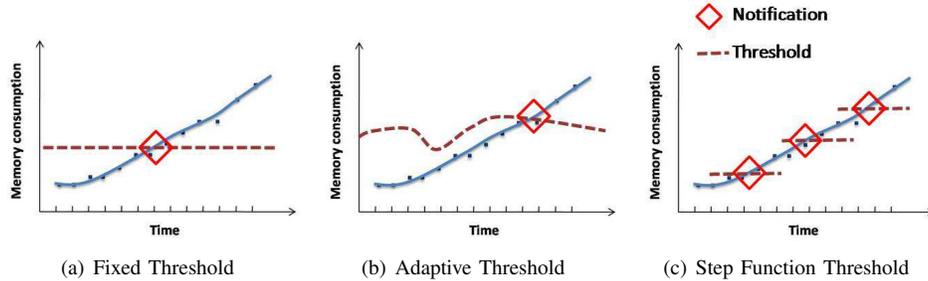


Figure 6. Threshold calculation for the resource whose evolution behavior is characterized by a trend

and notifications are raised when the resource behavior crosses the thresholds.

- The memory consumption can be modeled by a trend as mentioned in part III-B. Each mobile device needs to monitor and analyze the memory consumption to evaluate its state. So each mobile device compares periodically its memory consumption state with the threshold. A signal or a notification is raised when there is an intersection between the threshold and the parameter behavior as illustrated in Figure 6.

2) *Threshold calculation for the parameters whose evolution behavior is characterized by peaks:* Identifying and detecting peaks in a given time series is important in order to react accordingly and to avoid undesirable effects. The key question is how to set the correct threshold so as to minimize false positives. In this class parameter, the idea consists in specifying adaptive thresholds that are correlated with the parameter evolution behavior. In fact, using fixed thresholds in this class could maximize false alarms. Furthermore, since peaks characterize sudden changes from a normal behavior to an abnormal one, adaptive threshold correlated with the parameter evolution behavior remains under the resource shape. So, a violation of the threshold indicates a context changes.

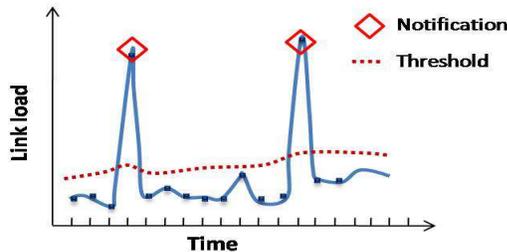


Figure 7. Threshold calculation for the resources whose evolution behavior is characterized by a peak

- As mentioned in section II (case study), the servers receive periodically contextual information from the devices through the gateways. In case of a large amount of data, the CPU load of the servers could be overloaded.

This overload is identified when the CPU load oversteps the adaptive threshold for this parameter class.

3) *Threshold calculation for the parameters whose evolution behavior is characterized by a burst:* Burst is an unusually large number of events occurring within a certain ON-period of time T_i . To the best of our knowledge, thresholds cannot be applied for this parameter class. So our idea consists on transforming a bursty model into a peaked model. So we propose to apply an aggregate function g in each ON-period. The aggregate function application is illustrated in Figure 8. As depicted in the Figure 8, the obtained model roughly coincides with a peaked function. Since we apply adaptive thresholds in this class parameter, a notification is raised when the parameter behavior crosses the threshold calculated in an ON-period.

For example, the gateway SCG1, is equipped with a context analyzer as described in section III. The maximum queue size for this device is set by the application designer. So that, if the traffic received by this gateway in a period T_i exceeds a maximum, then a burst is identified in T_i . Towards detecting bursty periods, applying the aggregate function g consists in calculating the slope of the scatter diagram obtained in each ON-period as depicted in the Figure 9(c). Second, in each ON-period, we compute the intensity of each slope formed in each ON-period. We obtain the figure 9(c). Consequently, if the slope intensity exceeds the threshold, then a burst is detected and appropriate reconfiguration actions are launched.

IV. ILLUSTRATIVE SCENARIO

In this section, we consider the case of a smart campus as shown in section II (case study). We focus in the interaction of the actors of the Department1 and SCS1 through SCG1. The Department1 is used by the unit researchers, the teachers, the students to achieve their work and to enhance collaboration between the different actors.

- For instance, at the beginning of each course session in R11, R12, R13 and R14, the presence sensor located in each room captures and localizes the mobile actors. Their positions is then forwarded to SCS1. SCS1 holding a context analyzer module as illustrated in section

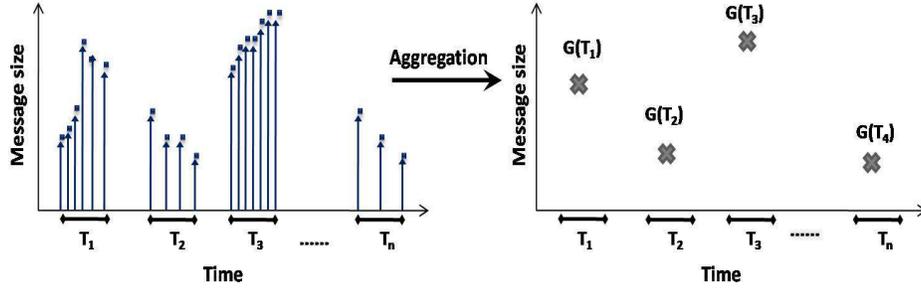


Figure 8. Aggregate application

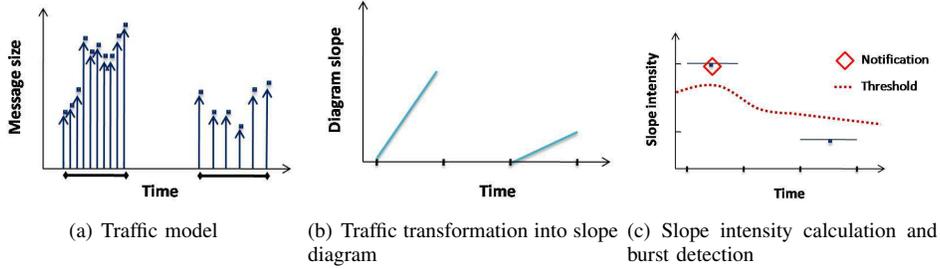


Figure 9. Threshold calculation for the resource whose evolution behavior is characterized by a burst

III, runs the analysis algorithm based on thresholds on the context data (mobile actors ' position) and takes the appropriate desicion.

- In case of a large amount of data received by the gateway SCG1 at a time t_i , the load of SCG1 increases rapidly reaching a high value (i.e a peak) and exceeding the adaptive threshold. The analysis algorithm integrated in SCG1 raises a notification and the gateway decides to split its load with its neighboring gateway namely SCG2.
- During the holidays, the SCS1 receives empty positions from the presence sensor (position=0) due to the non attendance of the teachers and the students. SCS1, by using the analysis algorithm based on thresholds finds that the threshold is always exceeding the context parameter (position). Consequently, the lights are switched off in R11, R12, R13 and R14 in order to reduce the power consumption.
- Each student participating to a course uses a tablet device to which the course will be dispatched through bluetooth. The tablet device holds the analysis algorithm described in section III in order to detect context changes. During the course, the students' tablet display appropriate slides and they follow their courses. Furthermore, the students can write annotations on their tablets and publish their comments to share knowledge between all the group members to enrich the course and enhance the collaboration. A student holding a tablet is

participating to the course by exchanging information and slides. For each amount of data received, the tablet device retrieves the memory consumption from the operating system using probes. Then, it compares each value with the threshold which can be either predefined, adaptive or step function since memory consumption belongs to the trend context parameter class. The tablet detects that at a time t_i , the memory consumption exceeds the threshold. So, the device displays a message preventing from receiving data. This scenario highlights the interaction of the campus application with context in such a way that it can detect context changes using the analysis approach in order to adapt its behavior accordingly.

V. CONCLUSION AND FUTURE WORK

The challenges for context aware applications design and implementation are to handle both context capture, context classification, context analysis and to react dynamically to every context changes. In this paper, we have proposed a novel context classification based on context parameter evolution behavior. Three classes have been identified. Context parameters whose behavior is modeled by a trend, context parameters whose behavior is modeled by peaks and context parameters whose behavior is modeled by bursts. Then, we have presented an analysis approach which aims at analyzing context and identifying context changes. The analysis approach is composed of a context provider and a context manager. The context provider is out of the scope

of this paper. We focus on the context manager. It is divided into four modules namely a context collector module, a context interpreter module, a context database module and a context analyzer module. The context analyzer module is the main component of our analysis approach. It is responsible for retrieving context information from the context database module, analyzing stored context information, identifying context changes and notifying the context aware application. Towards detecting the changes, the context analyzer module relies on thresholds which can be either fixed, adaptive according to the context parameter class. As future work, we plan to stretch the context parameter classification. For further development, we intend to implement our analysis approach as a first step and to integrate it into the framework FACUS [17]. The future work includes also ontology to support the context modeling because it makes context description more easier and rich of semantics.

ACKNOWLEDGMENT

This research is supported by the ITEA2's A2NETS (Autonomic Services in M2M Networks) project ¹.

REFERENCES

- [1] M. Weiser, "The computer for the 21st century," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 3, no. 3, pp. 3–11, Jul. 1999. [Online]. Available: <http://doi.acm.org/10.1145/329124.329126>
- [2] R. Ben Halima, K. Guennoun, K. Drira, and M. Jmaiel, "Non-intrusive qos monitoring and analysis for self-healing web services," in *Applications of Digital Information and Web Technologies, 2008. ICADIWT 2008. First International Conference on the*, aug. 2008, pp. 549–554.
- [3] K. Mitchell, "Supporting the development of mobile context-aware computing," Ph.D. dissertation, Lancaster University, 2002.
- [4] A. Schmidt, "Ubiquitous computing - computing in context," PhD thesis, Doctor of Philosophy, Computing Department, Lancaster University, England, U.K., November, 2002.
- [5] P. Prekop, "Activities, context and ubiquitous computing," *Computer Communications*, Mar. 2003. [Online]. Available: [http://dx.doi.org/10.1016/S0140-3664\(02\)00251-7](http://dx.doi.org/10.1016/S0140-3664(02)00251-7)
- [6] T. Rodden, K. Chervest, N. Davies, and A. Dix, "Exploiting context in hci design for mobile systems," in *in Workshop on Human Computer Interaction with Mobile Devices*, 1998.
- [7] B. Schilit, N. Adams, and R. Want, "Context-aware computing applications," in *In Proceedings of the Workshop on Mobile Computing Systems and Applications*, pp. 85–90.
- [8] P. J. Brown, J. D. Bovey, and X. Chen, "Context-aware applications: from the laboratory to the marketplace," *Personal Communications, IEEE [see also IEEE Wireless Communications]*, no. 5, pp. 58–64.
- [9] T. Cioara, I. Anghel, I. Salomie, M. Dinsoreanu, G. Copil, and D. Moldovan, "A self-adapting algorithm for context aware systems," in *Roedunet International Conference (RoEduNet), 2010 9th*, june 2010, pp. 374–379.
- [10] D. Zheng, J. Wang, W. Han, Y. Jia, and P. Zou, "Towards a context-aware middleware for deploying component-based applications in pervasive computing," in *Proceedings of the Fifth International Conference on Grid and Cooperative Computing*, ser. GCC '06, 2006, pp. 454–457.
- [11] C. Taconet, Z. Kazi-Aoul, M. Zaier, and D. Conan, "Ca3m: A runtime model and a middleware for dynamic context management," in *Proceedings of the Confederated International Conferences, CoopIS, DOA, IS, and ODBASE 2009 on On the Move to Meaningful Internet Systems: Part I*, ser. OTM '09, 2009, pp. 513–530.
- [12] I. Bouassida Rodriguez, G. Sancho, T. Villemur, S. Tazi, and K. Drira, "A model-driven adaptive approach for collaborative ubiquitous systems," in *Proceedings of the 3rd workshop on Agent-oriented software engineering challenges for ubiquitous and pervasive computing*, ser. AUPC 09, 2009, pp. 15–20.
- [13] G. K. Palshikar, "Simple Algorithms for Peak Detection in Time-Series," in *Proc. 1st Int. Conf. Advanced Data Analysis, Business Analytics and Intelligence*, 2009. [Online]. Available: <https://sites.google.com/site/girishpalshikar/Home>
- [14] D. Klan, M. Karnstedt, C. Politz, and K. Sattler, "Towards burst detection for non-stationary stream data," in *KDML*, 2008, pp. 57–60. [Online]. Available: <http://www.dbis.prakinf.tu-ilmnau.de/publications/files/DBIS:KlaKarPoe08.pdf>
- [15] X. Yang, "Designing traffic profiles for bursty internet traffic," in *In Proceedings of IEEE Global Internet*, 2002.
- [16] I. Lahyani, N. Khabou, and M. Jmaiel, "Qos monitoring and analysis approach for publish/subscribe systems deployed on manet," *Parallel, Distributed, and Network-Based Processing, Euromicro Conference on*, vol. 0, pp. 120–124, 2012.
- [17] G. Sancho, "Adaptation d'architectures logicielles collaboratives dans les environnements ubiquitaires. contribution l'interoprabilit par la smantique," LAAS, 138p., LAAS Reports 10779, 2010-12-14.

¹<https://a2nets.erve.vtt.fi/>