# Using the GQM Method to Evaluate Calmness in Ubiquitous Applications

Rainara M. Carvalho[1(✉)], Rossana M.C. Andrade[1], and Káthia M. Oliveira[2]

[1] Group of Computer Networks, Software Engineering and Systems (GREat), Federal University of Ceará (UFC), Fortaleza, Brazil
{rainaracarvalho, rossana}@great.ufc.br
[2] Laboratory of Automatic Control, Mechanics and Computer Science for Industrial and Human-Machine Systems (LAMIH), CNRS UMR 8201, University of Valenciennes and Hainaut-Cambrésis (UVHC), Valenciennes, France
kathia.oliveira@univ-valenciennes.fr

**Abstract.** Ubiquitous systems change the way users interact with computers, because their services must be available everywhere at any time, supporting users in various everyday activities. An essential element for these systems is their calm interaction with users, which means the system should not disturb them unnecessarily. Literature currently lacks work focusing on how to evaluate calmness and case studies made in a real usage situation. The aim of this work is to propose a model, defined using the Goal-Question-Metric (GQM) method, for calmness evaluation in ubiquitous systems and to show our results from a case study with three ubiquitous applications.

**Keywords:** Ubiquitous applications · Calmness · Software measures · GQM

## 1 Introduction

With the improvement in miniaturization of computational devices and also in wireless communications, the last years have seen an increase in ubiquitous applications development [1]. Such applications completely change the way users interact with technology, once their services must be available everywhere at any time. They must support users in everyday activities and remain transparent with little or no need of attention. Thus, an essential characteristic for these applications acceptance is *calmness*, which was firstly cited by Weiser and Brown (1997) as a new approach to properly fit computing into people's lives [2].

As ubiquitous applications are embedded within everyday objects (*e.g.,* cell phones, watches) and environments (*e.g.,* home, office), there is a high risk of users

This work is a result of Maximum project supported by FUNCAP and CNRS under grant number INC-0064-00012.01.00/12.
R.M. Carvalho—PhD Scholarship (MDCC/DC/UFC) sponsored by CAPES.
R.M.C. Andrade—Researcher scholarship - DT Level 2, sponsored by CNPq.


N. Streitz and P. Markopoulos (Eds.): DAPI 2015, LNCS 9189, pp. 13–24, 2015.
DOI: 10.1007/978-3-319-20804-6_2

feeling annoyed and overwhelmed by them. A calm application must support user's activities at the right time and place, delivering the best service possible [3, 4]. We believe that calmness has a great impact on the user satisfaction, therefore, on the usability and acceptance of the ubiquitous application.

We propose in this paper to evaluate calmness by using software measurements. To perform these measurement, firstly, it is necessary to define what software measures should be collected. In this case, there is a well-known method in the Software Quality area called Goal-Question-Metric (GQM) [5]. GQM is a goal-oriented approach that follows a hierarchical structure model starting with the definition of a measurement goal, that is refined in several questions and, finally, into metrics[1], which will provide information to answer the questions. By answering these questions, it is possible to analyze if the goal is achieved.

This paper presents a model composed of a goal, questions and software measures, defined by the GQM method, for evaluating the calmness characteristic in ubiquitous applications. By applying this model, it is possible to verify if the application presents a good level of calmness and what could be improved within the application to improve calmness level. We also present results from a case study involving three mobile ubiquitous applications.

## 2 Calmness Evaluation in Ubiquitous Applications

Ubiquitous applications are those capable of monitoring environment and users in order to provide services as natural as possible. To achieve this goal, they have to comply with challenging requirements such as autonomy, heterogeneity, coordination of activities, mobility and context-awareness [6]. In the scope of Human-Computer Interaction (HCI), there is another indispensable characteristic for ubiquitous applications being used and accepted by users: *calmness*.

According to [2], a calm technology should move easily from the periphery of attention to the center, and back. Periphery is used to describe what we are attuned to without focusing on it explicitly. For example, when we are driving, we do not focus on sounds but rather on the road. However, when a specific sound related to an event occurs, this information comes to the center of attention. That means we can keep information in periphery and, only when necessary, we can attend to it.

Through a literature search, we found one study [3] that carefully defines *calmness* and how it can be evaluated. It proposes a conceptual framework for evaluating calmness in ubiquitous applications. The authors classify calmness from two statements: calm timing and calm interaction. Calm timing means that the ubiquitous application should interact with the user in the right situation. Calm interaction means that the application should remain out of the user's attention whenever possible. They propose a subject evaluation using values such as High, Medium, Low and Very Low. However, they do not apply the proposed framework to evaluate ubiquitous

[1] Metric was the term used previously, but currently, the standard ISO 25000 SQuaRE recommends the use of the term measure.

applications in a real usage situation. Also it does not define a measurement function or a collect method for the proposed values.

Other previous work that was found related to calmness were [4, 7]. They sate that calm technology should allow users to access new information peripherally, enabling them to decide whether to divert their attention and change their focus. These works aimed to create a model for evaluating if a technology is calm or not. Anthropology-Based Computing (ABC) and Peripheral Interaction (PI) are the objects of their studies. However, they still are testing fade-ups and calm ringtones.

We argue that an evaluation using software measurement is also needed for calmness in ubiquitous applications. According to [10], measurement is the process of defining, gathering and analyzing data about products, in order to provide meaningful information for the purpose of improving it. There are several papers in literature that use measures to evaluate ubiquitous applications [8–11], including a paper with measures to evaluate context-awareness [12]. However, the literature study did not encounter any papers with software measures to evaluate calmness.

## 3 A GQM Model to Evaluate Calmness in Ubiquitous Applications

Using the GQM method, we defined our goal as follows: "**Analyze** the ubiquitous application, **for the purpose** of evaluating, **with respect to** calmness, **from the viewpoints** of the user". Then, we derived three questions and eleven measures based on both literature review [3, 8–14] and interviews with five experts on ubiquity. The resulted GQM model is presented in Fig. 1.

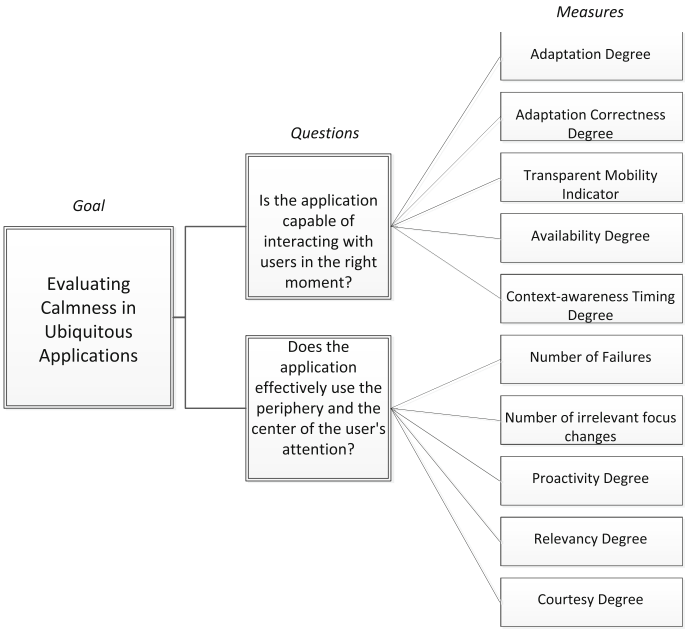


**Fig. 1.** The GQM model to evaluate calmness in ubiquitous applications

### 3.1 Question 1: Is the Application Capable of Interacting with Users in the Right Moment?

Users need to feel the ubiquitous application is available anytime and anywhere. This can be achieved by the following statements: (*i*) the context-awareness adaptation should happen when requested by a contextual change; (*ii*) this adaptation should be correct; and (*iii*) the application should have the characteristic mobility, which refers to a continuous or uninterrupted use of the systems as the user moves through several devices. Based on all these statements, we defined five measures showed in Table 1.

**Table 1.** Software measures for question 1

| Name | Measurement function | Interpretation | Collect method |
|---|---|---|---|
| Adaptation degree | $X = \frac{\left(\sum_{j=1}^{N} \frac{Aj}{Bj}\right) * 100}{N}$<br>N = Number of the different adaptations<br>Aj = Number of times the system adapts<br>Bj = Number of times an adaptation j was requested (the context changed) | Closer to 100 % is better | Automatic and observation |
| Adaptation correctness degree | $X = \frac{\left(\sum_{i=1}^{N} \frac{Ai}{Bi}\right) * 100}{N}$<br>Ai = Number of correctly performed adaptations i<br>Bi = Number of performed adaptations i<br>N = Number of the different adaptations | The closer to 100 % is better | Automatic and observation |
| Indicator of transparent mobility | X = A, where A is<br>(0) Nonexistent<br>(1) Low<br>(2) Medium<br>(3) High | (0) Nonexistent<br>(1) Low<br>(2) Medium<br>(3) High | Observation |
| Availability degree | X = B, where B is the mode of<br>(0) Nonexistent<br>(1) Low<br>(2) Medium<br>(3) High | (1) High<br>(2) Medium<br>(3) Low<br>(4) Very low | User form |
| Context-awareness timing degree | X = C, where C is the mode of<br>(1) High<br>(2) Medium<br>(3) Low<br>(4) Very low | (1) High<br>(2) Medium<br>(3) Low<br>(4) Very low | User form |

The two first software measures are related to context-awareness adaptation. When the context changes (*e.g.*, location and activity) the application has to adapt to this new context, if this does not happen, the user probably will feel the application has not worked well. So, the measure *Adaptation Degree* counts how many times the system adapts when requested by a context change.

*Adaptation Correctness Degree* aims to identify if that adaptation happened in an expected way for the user, it means checking if the services and/or information were delivered in the right way with respect to what is expected for the user at that specific moment. To calculate this, it is necessary to identify which adaptations the system has that can be collected (N). Thus, it is possible to count how many times a particular identified adaptation occurred during a certain usage period of the application (Bi) and how many of times those adaptions have occurred correctly (Ai). As a system can have several adaptations happening, the measure shows a summation aiming to add up all the adaptations occurred, and then calculating an average by dividing by N.

*Indicator of Transparent Mobility* was based on the information about mobility from the work of [13]:

- **High** is when the application can move from one device to another, keeping the past interactions and adapting resources to the new device (*e.g.,* screen size), so the user can continue their tasks seamlessly.
- **Medium** is when the application can move from one device to another, keeping the past interactions and adapt to the new features. However, the user is likely to have to wait a long time to start interacting with the application on the new device.
- **Low** is when the application can move to another device. However, the application does not adapt to it. For example, the new screen size is not taken into account.
- **Nonexistent** is when the application cannot move from one device to another.

The two last measures (*Availability Degree and Context-awareness Timing Degree*) are qualitative and collected from the user after the use of the application.

### 3.2 Question 2: Does the Application Effectively Use the Periphery and the Center of the User's Attention?

This question is related to the first consideration Weiser wrote in his paper that a calm technology should move easily from the periphery of our attention to the center, and back. This leads to measure if the application is being proactive to reduce the decision-making time of the user, the number of times the user is unnecessarily harassed and the number of failures that occurred during use.

It is necessary to know if the application interacts with the user only when needed, delivering relevant information and requests. To answer this question, five measures were defined and presented in Table 2.

The first measure (*Number of irrelevant Focus Changes*) aims to identify the amount of time the user had to change focus due to the technology. This shift in focus happens when some user action has to be performed for the application to work

**Table 2.** Measures for question 2

| Name | Measurement function | Interpretation | Collect method |
|---|---|---|---|
| Number of irrelevant focus changes | X = A, where<br>A = Number of actions that changes user's focus during use of the application | The further away from 0 is better | Automatic and observation |
| Proactivity of the application | X = N - A where<br>N = Number of total actions developed that can be supported by sensors<br>A = Number of actions the application replaces | The closer to 0 is better | Developer form |
| Number of failures | X = N, where<br>N = Total number of failures have occurred | The closer to 0 is better | Automatic and observation |
| Relevancy degree | X = B, where B is the mode of<br>(1) High<br>(2) Medium<br>(3) Low<br>(4) Very low | (1) High<br>(2) Medium<br>(3) Low<br>(4) Very low | User form |
| Courtesy Degree | X = D, where D is the mode of<br>(1) High<br>(2) Medium<br>(3) Low<br>(4) Very low | (1) High<br>(2) Medium<br>(3) Low<br>(4) Very low | User Form |

correctly during use. For example, restart a particular sensor during use and/or restart the application are actions that change the user attention unnecessarily.

The second measure (*Proactivity of the application*) aims to identify what the degree of proactivity is by counting how many user actions the application is able to replace. This measure is calculated by counting how many actions were developed in the application that can be supported by sensors and which among these actions are replaced by the sensors. The closer to zero the better, as this means every action capable of being developed by the sensors was actually developed by them.

The third measure (*Number of failures*) aims to identify how many failures happened while the user was using the application. The more failures that occur, the more distractions the user will face.

Finally, the two last measures (*Relevancy Degree and Courtesy Timing Degree*) are qualitative and collected from the form the user has to answer after using the application.

### 3.3 Collection Methods

Each measure, as already presented above, is collected by one or more of the following collected methods:

- **User interaction Logs:** Generation of user interaction logs with the application through code instrumentation. These logs should be collected during the use of the application. Each application that will be evaluated need to be instrumented.
- **Questionnaires:** Two questionnaires were developed to collect some measures. One questionnaire[2] is composed of questions to the application developer and the other[3] to the user who answers after using the application, to collect the qualitative measures.
- **Observation by an Evaluator:** A form for manual observation by an evaluator is filled when following the user during the application use.

## 4 Case Studies

The collection of the proposed measures was performed through case studies with three ubiquitous applications developed for mobile devices on the Android platform as follows: GREatPrint, GREatMute and GREatTour.

### 4.1 GREatPrint

This application aims to print documents at the nearest printer from the user. The application works as follows: after choosing a file, the user clicks on the print button for a given document, then the application searches for the Wi-Fi network with the highest signal intensity, which signifies that it is probably the closest to the mobile device. With this information, the application checks which printer is in the range of that network. Thus, the application sends the document to be printed on that printer, and informs the user which printer was selected.

Twelve users participated on this application evaluation. The developer also participated by answering a form specific to the applications developers. The twelve users were divided in four groups of three people to execute the application in different floors and rooms. The task defined to be performed by the users was to print a pre-established document on the application. Users were asked to use the application in the GREat research lab, because GREatPrint are targeted for this environment.

An evaluator was present during the usage, noting, for example, if the captured context was correct, if the application failed, and other valid information. After usage, users were asked to answer the user questionnaire. Table 3 presents the results of the measures collected.

We can conclude that the application **must** be improved to better address context-awareness and mobility. An improvement suggestion is to add more context

[2] Details in http://www.great.ufc.br/maximum/images/arquivos/developerquestionnaire.pdf.

[3] Details in http://www.great.ufc.br/maximum/images/arquivos/userquestionnaire.pdf.

**Table 3.** Results from the GREatPrint evaluation

| Question | Measure | Result | Interpretation |
|---|---|---|---|
| Is the application capable of interacting with users in the right moment? | Adaptation degree | 79 % | The application is not able to recognize reliably how to interact with the user, because all the achieved results were under the appropriate. For example, the *Adaptation Correctness Degree* was very low; the application sends the document to the wrong printer almost half times |
| | Adaptation correctness degree | 52 % | |
| | Indicator of transparent mobility | Nonexistent | |
| | Availability degree | Medium | |
| | Context-awareness timing degree | Medium | |
| Does the application effectively use the periphery and the center of the user's attention? | Number of focus changes | 15 actions | Despite presenting a large number of failures and focus changing, the application implements all functions replaceable by sensors and also the users feel all information and service were relevant and the interaction was friendly |
| | Number of failures | 33 failures | |
| | Proactivity degree | 0 | |
| | Relevancy degree | High | |
| | Courtesy degree | High | |

information to infer more precisely the actual location of the user by utilizing more sensors such as the accelerometer or the magnetometer. Also an improvement would be to allow the use of the application on other device types, for example: desktops and notebooks.

### 4.2 GREatMute

GREatMute is a service that runs in the background of the user's mobile phone. It monitors the mobile user's Google Calendar for events during which the user cannot receive calls, *e.g.*, "meeting" or "class". By discovering such events, the application places the user's phone on silent mode during the event time, so user does not get disturbed.

Also, this application allows the user to specify which events they would like to the device to be on silent mode, this is performed by the registry of keywords to be monitored. When the application finds an event in which the title has one of the keywords registered, GREatMute schedules based on the information extracted from start time and end time of the event. During this time the phone uses the silent profile.

To execute the tests with GREatMute, users were asked to install the application on their own device and use it during a week. Thus, users experienced the use of the

application in their actual day to day environments. The task for the user to perform was registering at least one event that would actually happen throughout the week on the calendar, as well as a corresponding keyword to the name of this event in the GREatMute application. A period of one week was selected because it was considered sufficient time for real events to happen that could be monitored.

For the testing of GREatMute, it was important to recruit people who knew how to use Google Calendar. If the users had previously used Google Calendar to schedule events, they would not have to also learn how to use it.

Only eight of the twelve invited users participated in the GREatMute evaluation. Table 4 presents the results.

**Table 4.** Results from the GREatMute evaluation

| Question | Measure | Result | Interpretation |
|---|---|---|---|
| Is the application capable of interacting with users in the right moment? | Adaptation degree | 92 % | The application has a very low degree of adaptation correctness. Almost always the application does not adapt properly (27 %). Also, the application is not able to move between several different devices when necessary and the users did not feel the application interact when was necessary |
| | Adaptation correctness degree | 27 % | |
| | Indicator of transparent mobility | Nonexistent | |
| | Availability degree | High | |
| | Context-awareness timing degree | Very low | |
| Does the application effectively use the periphery and the center of the user's attention? | Number of focus changing | 0 actions | All measures presented good results. The application can replace actions of users so they are not bothered to take many decisions |
| | Number Of failures | 0 failures | |
| | Proactivity degree | 0 | |
| | Relevancy degree | High | |
| | Courtesy degree | High | |

The application needs to improve in relation to the capability of interacting at the right moment. GREatMute could take into account two additional pieces of context information: the user's current location and the location of the event he/she registered on the calendar. Thus, it is possible to know if the user is at the event that they registered and thus change the mobile profile more reliably. If the user is not in an approximate radius of the registered venue, the application does not need to put the phone on silent mode.

### 4.3 GREatTour

GREatTour is a mobile guide for visiting the GREat Lab. It provides information about the environments of the laboratory that the user is visiting. The application works as follows: the user must scan a QR Code that is found on the door of the environment to

update user location. Then, a map of the lab is displayed, highlighting the environment where the user is. So, the user can view media options related to this environment (texts, photos and videos). However, the rendering of this media depends on the battery level of the device. When battery is low (0–9 %), only text is displayed, when it is medium (10–20 %), text and images are displayed, and finally, when it is high (21–100 %), texts, images and videos are displayed.

Like with GREatPrint, the users were asked to use the applications in the GREat research lab, since this application is targeted for this environment. Also, an evaluator was present during the usage. In this case study, only six of the twelve users actually used the application, due to unavailability of the other users. Each user made three visits (tours) to the laboratory GREat, with different battery levels in order to test all application states. Each visit consisted of seeing three environments. In each environment, users updated the map and accessed all available media, according to battery of the phone. Table 5 presents the results.

**Table 5.** Results from GREatTour evaluation

| Question | Measure | Result | Interpretation |
|---|---|---|---|
| Is the application capable of interacting with users in the right moment? | Adaptation degree | 72 % | Although the degree of context correctness is high (100 %), the adaptation not always happens (72 %). Besides, users think the availability and timing need improvement |
| | Adaptation correctness degree | 100 % | |
| | Indicator of transparent mobility | Nonexistent | |
| | Availability degree | Medium | |
| | Context-awareness timing degree | Medium | |
| Does the application effectively use the periphery and the center of the user's attention? | Number of focus changes | 14 actions | Despite users feel the application shows only relevant requests and use effectively the periphery and the center of the attention, the degree of user attention requisition is very high and is completely far from acceptable value. GREatTour requests a lot of actions, including unnecessarily actions (14 actions), like restart the application and restart the connection with internet |
| | Number of failures | 6 failures | |
| | Proactivity degree | 14 | |
| | Relevancy degree | High/Medium | |
| | Courtesy degree | High | |

GREatTour provides a good degree of context-awareness correctness. However it **must** be improved to better address adaptation degree, mobility, availability, context-awareness timing, focus changing and proactivity. The following changes could improve GREatTour: (*i*) allow mobility between different devices (different phones and tablets) without loss of information in the new device screen and maintain the previous tour information; and (*ii*) use mobile sensors (*e.g.,* magnetometer, accelerometer and wi-fi) to detect the user location more transparently without the need for user intervention (QR code scans).

## 5 Conclusion and Future Work

By using the GQM method, it was possible to develop a model to evaluate calmness in ubiquitous applications. This model is composed of software measures that were applied in three ubiquitous applications developed for mobile devices: GREatPrint, GREatMute and GREatTour. As a result of these case studies, the measures indicate that these applications still need to improve to have good levels of calmness. This reveals an indication that the proposed model was able to assess Calmness.

However, due to the variety of application fields within ubiquitous computing, each application may require different test procedures. For instance, GREatMute required that the user use the application during the week on their mobile phones and after that the subjective measures were collected. On the other hand, GREatPrint and GREatTour were used inside the GREat Lab and an evaluator could observe the use. However, each one of them required different planning of scenarios, because the context is not the same between these applications.

Therefore, this proposed model does not exclude a plan for how an evaluation must be in different ubiquitous applications. The perspective for future work and improvement of this work is to create a methodology capable of systematically guiding an evaluator, so that they are able to assess calmness and investigate the influence of calmness on other characteristics of HCI, for example, Usability.

**Acknowledgement.** We would like to thank the Software Quality and Testing Cell (CTQS) of GREat for the technical support that they provided.

## References

1. Weiser, M.: The computer for the 21st century. Sci. Am. **265**, 94–104 (1991)
2. Weiser, M., Brown, J.S.: The coming age of calm technology. In: Denning, P.J., Metcalfe, R.M. (eds.) Beyond Calculation. Copernicus, New York (1997)
3. Riekki, J., Isomursu, P., Isomursu, M.: Evaluating the calmness of ubiquitous applications. In: Bomarius, F., Iida, H. (eds.) PROFES 2004. LNCS, vol. 3009, pp. 105–119. Springer, Heidelberg (2004)
4. Brown, J.N.A., Leitner, G., Hitz, M., Mallofré, A.C.: A model of calm HCI. In: Peripheral Interaction: Shaping the Research and Design Space, Workshop at CHI (2014)

5. Basili, V., Rombach, H.: Goal question metric paradigm. In: Marciniak, J. (ed.) Encyclopedia of Software Engineering – 2, vol. 1, pp. 528–532. Wiley, Chichester (1994)
6. Rocha, L.S., Ferreira, J., Lima, F.F.P., Maia, M.E.F., Viana, W., Castro, M.F., Andrade, R. M.C.: Ubiquitous software engineering: achievements, challenges and beyond. In: Brazilian Symposium on Software Engineering (in portuguese) (2011)
7. Leitner, G.: A measure of calm. In: Conference on Human Factors in Computing Systems - Proceedings (2014)
8. Song, J., Park, K.R., Kwon, S., Lee, J.H., Yun, M.H.: The development of human-system interactivity metrics for ubiquitous service applying user-centered design methodology. In: World Congress on Services (2009)
9. Lee, J., Song, J., Kim, H., Choi, J., Yun, M.H.: A user-centered approach for ubiquitous service evaluation: an evaluation metrics focused on human-system interaction capability. In: Asia-Pacific Conference, APCHI (2008)
10. Ranganathan, A., Al-Muhtadi, J., Biehl, J., Ziebart, B., Campbell, R.H.H., Bailey, B.: Towards a pervasive computing benchmark. In: International Conference on Pervasive Computing and Communications Workshops (2005)
11. Scholtz, J., Consolvo, S.: Toward a framework for evaluating ubiquitous computing applications. IEEE Pervasive Comput. **3**, 82–88 (2004)
12. Santos, R.M., Oliveira, K.M., Andrade, R.M.C., Santos, I.S., Lima, E.R.R.: A quality model for human-computer interaction evaluation in ubiquitous systems. In: Latin American Conference on Human Computer Interaction, CLIHC (2013)
13. Yu, P., Ma, X., Cao, J., Lu, J.: Application mobility in pervasive computing: a survey. Pervasive Mob. Comput. **9**(1), 2–17 (2013)
14. Kourouthanassis, P.E., Giaglis, G.M., Karaiskos, D.C.: Delineating the degree of 'pervasiveness' in pervasive information systems: an assessment framework and design implications. In: Pan-Hellenic Conference on Informatics, PCI (2008)