Computers and Electrical Engineering 000 (2016) 1–20

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

A model-driven approach for quality of context in pervasive systems

José R. Hoyos[a,*], Jesús García-Molina[a], Juan A. Botía[a], Davy Preuveneers[b]

[a] Universidad de Murcia, Facultad de Informatica, Campus de Espinardo, 30100 Murcia, Spain
[b] iMinds-DistriNet-KU Leuven, Belgium

ARTICLE INFO

Article history:
Received 5 August 2015
Revised 1 July 2016
Accepted 1 July 2016
Available online xxx

Keywords:
Quality of context
Model driven engineering
Domain specific language
Context modeling
Context-aware

ABSTRACT

Handling context is a crucial activity in context-aware systems. In building such systems, the creation of models helps developers to understand and reason on the context information. The quality of context information is required to achieve these systems behavior according to the requirements. Therefore, context models should not only represent the context information but also the quality of context (QoC). MLContext is a textual domain-specific language (DSL) designed to model context, which has been used to automatically generate software artifacts related to the context management for some context-aware frameworks. In this article we present an MLContext extension for modeling QoC. The QoC added features include constructs to express context situations (i.e. constraints on context information), quality requirements and quality levels. The new constructs have been mapped to code for two frameworks supporting QoC (COSMOS and SAMURAI). These mappings have been validated by means of a case study.

1. Introduction

Context-aware systems are able to sense their environment and automatically adapt their behavior to the changes occurring. The environment is characterized by context information that includes data such as the current location and time, user activities and profiles, and device parameters. A key element in a context-aware architecture is therefore the component that manages the context information, which is in charge of representing and manipulating such information. In the last decade, a number of approaches for context modeling have been proposed and several context managers have been built. However, the mastering of complexity in managing context information is still a great challenge and there are few tools and models that actually help developers as noted in [1]. The context information is obtained mainly from sensors that gather data on the environment.

The context managers often erroneously assume that the information retrieved from the sensors is accurate and reliable, but limitations of sensors and the situation of a specific measurement can affect the value of the context information. The quality of context information severely influences the adaptiveness of context-aware applications [2], and the lack of information about the quality of context (QoC) can result in degraded performance of these applications in pervasive environments. QoC information can help context-aware applications to resolve uncertain and conflicting situations within the gathered context information.

* Corresponding author. Fax: +34 868 88 4151.
E-mail addresses: jose.hoyos@um.es (J.R. Hoyos), jmolina@um.es (J. García-Molina), juanbot@um.es (J.A. Botía), davy.preuveneers@cs.kuleuven.be (D. Preuveneers).

http://dx.doi.org/10.1016/j.compeleceng.2016.07.002
0045-7906/

The QoC notion was proposed in [3] as an essential aspect to be taken into account in context-aware software. The authors defined QoC as the information that describes the quality of the context information. Since then QoC received a lot of attention from the research community and several works have contributed proposals to represent, measure and evaluate the QoC. However there are few context managers with QoC support as noted in [1].

The Model-Driven Engineering (MDE) paradigm is gaining acceptance mainly due to its ability to improve the software productivity by raising the level of abstraction and automation in developing software [4,5]. Models are created at different abstraction levels to represent aspects of the system, and model transformations are used to generate software artifacts. Some works have shown how MDE techniques can also be useful to build context-aware software [6,7]. These works mainly focused on the definition of context modeling languages and the automation of development tasks. However, how the management of QoC can take advantage of the MDE techniques is a topic that has received little attention in the past, with COSMOS being one of the more recent relevant proposals [1]. Actually, an important research effort is still needed to provide appropriate languages, methods and tools to efficiently support the model-driven development of context-aware systems.

QoC is an important subject in several areas of computing systems. In this article, we present an extension of MLContext whose purpose is modeling the QoC aspects of pervasive systems. MLContext is a specific-domain language for modeling context information, which is the core element of an MDE approach to manage context information in context-aware applications. This language was presented in [7], where we show that MLContext is a simple, elegant and legible language. In that previous work, we also explain how software artifacts related to the context management can be automatically generated from MLContext models. A “separation of concerns” was applied in the design of MLContext: context information is modeled by separating application-dependent aspects (e.g. quality) from application-independent aspects (e.g. entities and sources of context). In that previous work, we also explain how software artifacts related to the context management can be automatically generated from the MLContext model but any application-dependent aspect was not considered.

A preliminary work was presented at [8] where we pointed out the possibilities of modeling QoC by adding to MLContext a basic set of QoC constructs and show how Java code can be generated for a simple example without validation. We will show here in detail the result of the work carried out from that initial proof of concept. A review of the literature allowed us to identify the main guidelines for the DSL quality extension design. The result is a DSL that provides (i) constructs to model context situations and quality parameters, (ii) mechanisms that allow the composition and parameterization of situations when they are modeled, and (iii) support for external function calls and the possibility to define quality requirements taking into account the multidimensional aspect of the context quality information. Our QoC modeling approach has been validated for two different context-aware middlewares that support context quality: COSMOS [1] and SAMURAI [9]. Using the same case study, we show how software artifacts for the QoC management can be automatically generated from MLContext models. In particular, we demonstrate how MLContext models are able of generating the models managed in the COSMOS approach which are expressed at a lower abstraction level. Therefore, the contribution of our work is twofold. On the one hand, we present and evaluate an MDE approach for QoC in context-aware applications. To the best of our knowledge, our proposal is the only model-driven approach able to generate code for different middlewares supporting QoC. On the other hand, we illustrate how the separation of concerns is applied to add QoC to MLContext.

This paper is organized as follows. In Section 2 we define the notion of context quality and identify the main quality parameters used in the literature. In Section 3, we present the design decision we have taken to create a context quality extension for the MLContext DSL. Section 4 shows the context quality modeling process with MLContext. In Section 5 the *Application metamodel* for the abstract syntax of the QoC extension is briefly explained. Section 6 presents a case study based on a flash sale scenario, to be used to illustrate the modeling process and the automatic code generation. We also show how we can use MLContext to model the flash sale scenario with QoC. Section 7 presents a brief description of the COSMOS and SAMURAI context-aware frameworks which provide QoC support, and explain how we can automatically generate specific code for them, following and MDE approach. In Section 8 we evaluate our approach in terms of productivity, maintainability and reusability. Section 9 presents the related work. Finally Section 10 shows the conclusion and future work.

2. Background

As indicated above, context-aware systems should be concerned about the quality of the context information that they manage. Inaccurate or erroneous information could lead to a system malfunction when adapting to changes in the environment. Before describing our QoC modeling approach in detail, we shall discuss in this section what “quality of context” means and introduce some basic concepts such as quality parameter and context situation. Moreover we will present a brief summary of the MLContext language.

2.1. Quality of context

Quality of context refers to the quality of information that is used as context information and not to processes or devices that possibly provide the information [3]. Several works on information quality [10,11] have assumed the idea that quality is simply a “fitness for use” according to the definition of quality proposed by Juran [12]. This definition entails that information quality is application-dependent. A user might consider that the quality of a piece of information is appropriate for

Please cite this article as: J.R. Hoyos et al., A model-driven approach for quality of context in pervasive systems, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.07.002