The Journal of Systems and Software 118 (2016) 251–276

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

A systematic review on the engineering of software for ubiquitous systems

Alejandro Sánchez Guinea*, Grégory Nain, Yves Le Traon

Interdisciplinary Center for Security, Reliability and Trust, University of Luxembourg, 4 rue Alphonse Weicker, L-2721 Luxembourg, Luxembourg

ARTICLE INFO

Article history:
Received 16 October 2015
Revised 11 April 2016
Accepted 13 May 2016
Available online 16 May 2016

Keywords:
Empirical software engineering
Evidence-based software engineering
Systematic review
Research synthesis
Software development cycle
Ubiquitous systems
Development methods
Pervasive systems

ABSTRACT

Context: Software engineering for ubiquitous systems has experienced an important and rapid growth, however the vast research corpus makes it difficult to obtain valuable information from it.

Objective: To identify, evaluate, and synthesize research about the most relevant approaches addressing the different phases of the software development life cycle for ubiquitous systems.

Method: We conducted a systematic literature review of papers presenting and evaluating approaches for the different phases of the software development life cycle for ubiquitous systems. Approaches were classified according to the phase of the development cycle they addressed, identifying their main concerns and limitations.

Results: We identified 128 papers reporting 132 approaches addressing issues related to different phases of the software development cycle for ubiquitous systems. Most approaches have been aimed at addressing the implementation, evolution/maintenance, and feedback phases, while others phases such as testing need more attention from researchers.

Conclusion: We recommend to follow existing guidelines when conducting case studies to make the studies more reproducible and closer to real life cases. While some phases of the development cycle have been extensively explored, there is still room for research in other phases, toward a more agile and integrated cycle, from requirements to testing and feedback.

1. Introduction

Ubiquitous Computing is aimed at integrating computation with the physical world in a way that blends in so completely that it becomes unnoticeable (Want and Pering, 2005). The last 20 years or so have brought a considerable progress towards this aim with the proliferation of connected devices of different scales and ownership models (Abowd, 2012). This progress has inspired a great deal of research and development on areas such as pervasive computing, Ambient Intelligence (AmI), Internet of Things (IoT), cyber-physical systems, etc., which while under the umbrella of ubiquitous computing is focused on different specific aspects.

Back in 1999, Gregory D. Abowd brought to the attention of software engineering researchers the area of ubiquitous computing at the International Conference on Software Engineering (ICSE) (Abowd, 1999), arguing about the importance of software engineering for the development of ubiquitous systems. From that year to date there has been a rapid growth of the field and its diversification, which has derived into a very large and generally unorganized research corpus containing approaches to address a variety of issues for different phases of the software engineering cycle for ubiquitous systems. Given the size and heterogeneity of the literature (e.g., a quick search on google scholar may provide not less than ten thousand results), and in the light that to the best of our knowledge only informal surveys restricted to specific issues have been conducted, we have seen the need to organize and synthesize the existent research corpus to obtain a clear idea on the main approaches that have been proposed for the different phases of the development of software for ubiquitous systems, what issues they addressed, what their limitations are, and what open issues still need to be tackled.

A systematic literature review is a systematic approach to building a body of knowledge about a particular topic or research question(s), and to identify problems for future research and support decision making and technological selection (Zhang and Babar, 2013). The systematic review herein presented seeks to find, evaluate, and synthesize the most relevant approaches that have been

* Corresponding author.
E-mail addresses: alejandro.sanchezguinea@uni.lu (A. Sánchez Guinea), gregory.nain@uni.lu (G. Nain), yves.letraon@uni.lu (Y. Le Traon).

http://dx.doi.org/10.1016/j.jss.2016.05.024
0164-1212/© 2016 Elsevier Inc. All rights reserved.

proposed to date in the literature for any of the issues related to the software development life cycle for ubiquitous systems. We classify the approaches according to the phase they relate to in the development life cycle and the general concerns they address. It is our belief that this review will be useful to practitioners, willing to stay up-to-date on approaches, their limitations, and unresolved general issues, as well as for researchers trying to identify issues that have been researched or where research is lacking. The results provided by the review should be relevant for the software intensive industry.

The article is organized as described next. In Section 2, we provide an overview of software development for ubiquitous system, talk about the phases that have been recognized in the literature as part of the software development life cycle of this type of systems, and identify any existing related review. Section 3 describes the method that has been applied for undertaking this review. Section 4 presents the results of the review, including information about the main phases considered for the development of software for ubiquitous systems, the main approaches for each phase, issues that they address and their limitations, and the issues that still remain open in general. Section 5 discusses the benefits and limitations of the review, and analyses the strength of evidence and implications for both research and practice. Sections 6 concludes the paper.

2. Background - engineering of software for ubiquitous systems

Ubiquitous computing, as envisioned by Weiser (1993) more than 20 years ago, enhances computer use by making many computers available throughout the physical environment, while making them effectively invisible to the user.

In the years that have passed since Weiser vision numerous successful prototypes have been built and evaluated, demonstrating the utility of many different aspects of ubicomp systems, however many challenges are still to be solved before we can realize many of the core ubicomp scenarios on any large scale (Caceres and Friday, 2011). Due to the key role that software plays in ubiquitous systems, an important number of the challenges of ubicomp systems are related to the need for software engineering advances to support the research endeavors in ubiquitous computing (Abowd, 1999). This need has led software engineering researchers to propose, over the last 15 years or so, a large number of approaches for different phases of the engineering process specifically tailored for the characteristics of ubiquitous systems (Abowd, 2012).

2.1. Summary of previous reviews

Reviews, surveys, and state of the art overviews have been prevalent in the literature on software development for ubiquitous systems (e.g., Shi et al., 2011; Raychoudhury et al., 2013; Krupitzer et al., 2015; Martin et al., 2011). Lots of articles have been published about the challenges of this type of development, most of the time based on rather informal literature reviews and the authors' experience (e.g., Abowd, 2012; Conti et al., 2012; Tang et al., 2011; Cook et al., 2009; Want and Pering, 2005; Edwards and Grinter, 2001; Abowd, 1999).

The reviews that can be found in the literature related to the development of software for ubiquitous systems have typically focused on general concepts such as autonomic computing (Huebscher and McCann, 2008) and self-adaptive systems (De Lemos et al., 2013), as well as on particular implementations of ubiquitous systems such as smart home (Solaimani et al., 2015; Alam et al., 2012; Chan et al., 2008) and ambient intelligence (Cook and Das, 2007; Cook et al., 2009), specific phases of the development life cycle (e.g., feedback Brun et al., 2009, design Tang et al., 2011), specific tasks such as situation identification (Ye et al., 2012), specific concerns such as application mobility (Yu et al., 2013), or particular techniques or tools such as middleware (Raychoudhury et al., 2013; Martin et al., 2011) and models at runtime (Szvetits and Zdun, 2013).

Among the reviews we found, only one deals with software engineering approaches but targets a more general type of systems than ubiquitous systems, that is, self-adaptive systems (Krupitzer et al., 2015). All reviews we found followed an informal process except for two, one about smart home (Solaimani et al., 2015) and one about models at runtime (Szvetits and Zdun, 2013).

2.2. Objectives of this review

The area of software development for ubiquitous systems has considerably grown in the last decade, with an important amount of approaches being proposed for addressing different concerns on each phase of the development life cycle. However, as seen in Section 2.1, to the best of our knowledge no systematic review covers the most relevant phases of the ubiquitous systems' software development cycle. Therefore, this review aims at filling this gap by providing a systematic review of the literature on that subject.

To this end, this review will use the following research questions as a guide:

RQ1 What are the stages of the development life cycle of software for ubiquitous systems that have been mainly considered?
RQ2 What are the main approaches that have been proposed for each of these stages?
RQ3 What are the current limitations of such approaches?
RQ4 What are the open issues to be further investigated in regard to the development of software for ubiquitous systems?

3. Review method

In order to conduct this review, we broadly followed the guidelines by Kitchenham and Charters (2007) and their updated version by Kitchenham and Brereton (2013). We first performed a search and selection process to obtain the primary studies that constituted the basis for obtaining our results. The primary studies were passed through a quality assessment step to provide validation of the quality of each study. Next, data related to approaches' characteristics and limitations, future work, and general open issues was extracted from the primary studies. Finally, we synthesized the data to obtain the results of the review.

The workload of the review was distributed among the authors of this paper according to their experience. The two less experienced researchers (first and second authors in order of appearance on the title page, a PhD student and a post-doctoral researcher, respectively) were responsible for directly performing the review, while the most experienced researcher (third author, professor for more than nine years) was responsible for checking and verifying each of the intermediate and overall outcomes of the review, seeking to ensure consistency and completeness.

3.1. Search and selection process

The process followed to search and select primary studies is depicted in Fig. 1. We have devised this process inspired by the multi-stage process used by Kitchenham and Brereton, in what can be considered the updated guidelines for conducting SLR's in software engineering (Kitchenham and Brereton, 2013).

First, an informal search step (`prel1` in Fig. 1) was performed to ensure having enough papers for the review and enough