A Unified Platform for the Dynamic Evolution of Context-Aware Highly Agile Services

Xiaodong Liu
Edinburgh Napier University, UK

Zakwan Jaroucheh
Edinburgh Napier University, UK

Sally Smith
Edinburgh Napier University, UK

Huiqun Zhao
North China University of Technology, China

1. INTRODUCTION

In pervasive systems, e.g., smart homes and smart vehicles, users will be surrounded by many mobile devices and sensors embedded everywhere in the systems which will cooperate together to provide a context-aware pervasive environment that supports humans in everyday activities, e.g., in business, health care, or education. In this respect, the users will enjoy a new experience in a non-obtrusive way as the existing infrastructures will be more proactive and dynamically adaptable to the current situations, user preferences and environmental context in a less intrusive way (Weiser, 1991). Context-awareness refers to the capability of an application or a service being aware of its physical environment or situation (e.g. context) and to respond proactively and intelligently based on this awareness (Baldauf et al., 2007). We define context-aware service adaptation as *the action that modifies the service in a way that causes its behaviour to evolve according to the evolution of business and user requirements, and the context considered relevant to that service.*

Currently, despite the active research into and the development of services over the last several years, context-aware services are still difficult to build for the following reasons. Firstly, current service development standards such as Business Service Execution Language (BPEL), are not sufficient for describing and handling context information (Sheng et al., 2010). Secondly, service engineers usually implement everything related to context management, including acquiring, dissemination, and usage of context information, in an ad hoc manner. Moreover, the mobility of users and devices across different context domains (areas) induces changes to the application operating context and makes the development and evolution of services more difficult. In particular, we argue here that context information relevant to the current task at hand may come from a number of spatial domains. For instance, a dynamic recalculation of the quickest routes for a trip involves acquiring the latest contextual information such as traffic congestion from remote sources. We can imagine a domain-oriented context management system where the context information available in each domain is managed by a separate context manager. While moving, the user roams across the domains. In addition, each domain may maintain its own sensors and mechanisms for inferring context related to this user. To tackle the above challenge, consequently, collaborative context management across domains is needed.

Finally, services should be dependable in the sense that it should meet the user and business requirements and needs, which are highly changeable all the time in pervasive systems. Therefore, service modelling must be flexible enough to deal with these constant changes – both at the business level (e.g. evolving business rules) and at the technical level (e.g. contextual information and platform upgrades). As services get deployed on

DOI: 10.4018/978-1-4666-5888-2.ch274

an increasingly larger diversity of computing platforms and operate in different execution environments, developers and maintenance engineers (namely, service engineers) should make considerable effort to manually porting services code to different platforms or new versions of the same platform. Many different solutions have been proposed by researchers to the problem of context-aware adaptation during service development and provision. However, existing solutions do not allow the service engineers to logically view and model the service variant in terms of the features that determine the difference between service variants in each usage context. As a consequence, developing and maintaining services is still a difficult, cumbersome, and time consuming activity.

Supporting the development and provision of such service applications raises several challenges that involve models, middleware, methods and tools. Thus, the main challenges in this direction are related to the support that can be offered to the service engineers to satisfy the user expectations in a dynamic and highly variable environment. Here we proposed a unified development and evolution approach that relies on model-based solutions to build context-aware highly adaptive (agile) services. The proposed approach encompasses a set of methods to generate service models and adaptable code from UML-based specifications in a dynamic and constantly evolving way based on the available context information in different domains. All these methods are supported by an integrated framework which is based on a conceptual model.

The remainder of the article is structured as follows: Section 2 critically analyses the related work. Section 3 describes the proposed service development and evolution approach and outlines the developed technologies supporting it. To realize the approach, Section 4 describes the implemented *Apto* platform prototype that facilitates the development and evolution tasks. Section 5 and 6 present the conclusion and future work.

2. BACKGROUND

The future success of the Internet of Things depends not only on technical innovations in the underlying hardware, but also on appropriate software methodologies, technologies and tools in fields such as operating systems, middleware, ubiquitous and pervasive computing technology (Welbourne et al., 2009).

Context management has focused on pervasive computing environments and although it has been widely researched in recent years, serious problems remain (Castelli et al., 2008; Serrano et al., 2007; Timo, 2010). Context awareness at physical, social and cultural levels is central to pervasive services that deliver services to end-users in a dynamically optimal way with the best quality possible. The main problems with understanding context include i) weak support for dynamic context capture at the deep semantic level; ii) lack of effective automatic matching between context and contents and service specification; iii) lack of commonality, such as a generic context model, which is needed for context remembrance across applications.

To ease and support the development of context aware applications, the context models and their context management systems should meet a set of requirements (Bettini et al., 2007). In addition, a generic (i.e. application and domain-agnostic) and dynamic context model is needed that could to be reused by different applications and shared among different systems. This generic context model should dynamically capture various types of context information and the dependencies between them.

Generally speaking, the existing solutions for context-aware services engineering can be roughly divided into six categories: middleware solutions and dedicated service platforms, use of ontologies, rule-based reasoning, source code level programming/ language extensions, model-driven approaches, and message interception (Kapitsaki et al., 2009). Obviously, each solution has its pros and cons. For instance, apart from its advantages, model-driven approaches add some extra complexity to the development process to keep the synchronization between high-level models and low-level executable code all the time. However, as the service engineering process passes through the stages of analysis and design prior to the actual code development, model-driven solutions allow service engineer to consider the context and adaptation also in these stages. Moreover a more powerful solution could be achieved by leveraging ideas from several solutions into a unified approach at the same time.

In the context of model-driven solutions, several approaches have been proposed. Sheng (Sheng et al., 2010) presented ContextServ, a platform for simplifying the development of context-aware Web services.