Leveraging the Web Platform for Ambient Computing: An Experience

Fabio Mancinelli, XWiki SAS, France

ABSTRACT

This paper explores the idea of what can be achieved by using the principles and the technologies of the web platform when they are applied to ambient computing. In this paper, the author presents an experience that realizes some of the goals of an Ambient Computing system by making use of the technologies and the common practices of today’s Web Platform. This paper provides an architecture that lowers the deployment costs by maximizing the reuse of pre-existing components and protocols, while guaranteeing accessibility, interoperability, and extendibility.

Keywords: Ambient Computing, Deployment Costs, Distributed Systems, Interoperability, Web Architecture

1. INTRODUCTION

Pervasive and Ambient computing propose a paradigm where devices present in a physical environment collaborate in order to support people in carrying out their daily tasks (Satyanarayanan, 2001). Though many technologies have been introduced in order to implement this paradigm (Román, Hess, Cerqueira, Ranganathan, Campbell, & Nahrstedt, 2002), there are still a lot of open issues that must be addressed and solved. In particular many of these issues concern the “ubiquitousness” of these solutions and their interoperability with the multitude of heterogeneous devices that might interact with such a kind of systems. In this paper we explore a very simple idea: what can be achieved by using the principles and the technologies of the Web Platform when they are applied to Ambient Computing? This idea is motivated by the fact that by leveraging the Web Platform we can address interoperability issues by relying on a uniform platform that has a well defined set of protocols and semantics. This is very important because different solutions must inter-operate in order to provide a better experience to the end-users. By using the Web Platform as a common ground, and by building on top of it, many of the interoperability issues are solved forefront. In this paper, basically, we took the Web Platform as the reference platform, and built on top of it a simple system that follows the Ambient Computing principles. Our aim was to describe the experience we did in building an actual system and to present the advantages that, in our opinion, derive from following such a kind of approach.

DOI: 10.4018/jaci.2010100104
In the remainder of the paper we will introduce our reference platform (i.e., the Web Platform) and its characteristics; then we will present our architecture and we detail all the aspects that are addressed and how they are addressed by our system. Finally we present conclusions with final remarks.

2. THE WEB PLATFORM

In this section we will briefly introduce the Web Platform and what it consists of. With the term “Web Platform” we refer to the ensemble of the protocols and standards the World Wide Web is built upon. The term “Web Platform” has been circulating for a while in the Web community, and has gained more and more importance after that Google organized its first “Google I/O” conference (Google Inc., 2008a) whose aim was that of “Advancing the Web as a Platform”. The term platform is crucial because nowadays developers are using the Web, its architecture and the technologies it is based on, as an actual platform for engineering, developing and deploying their applications. This is basically something similar to what happened with the Java Platform (Sun Microsystems, 2008), with the introduction of the Java language, its standard libraries and all their extensions like the “Enterprise Edition” or the “Micro Edition”. The idea of the Web as a platform is also corroborated by all the companies that are producing advanced tools and solutions for easily building complex application without taking care of all the low-level details (Ruby, Thomas, & Hanson, 2008) (Google Inc., 2008b).

The Web Platform has an architecture that is based on a well defined set of principles and constraints (Fielding, 2002) that are implemented by a set of standard and widely deployed communication protocols (Fielding, Gettys, Mogul, Frystyk, Masinter, Leach, & Berners-Lee, 1999), and makes use of commonly used data formats for exchanging information (IANA). A key element in this architecture is the concept of resource and its associated representations (i.e., how a resource, which can be also a physical resource, is represented for being used and manipulated in a digital context). The term “resource” is highly generic but, as stated in Richardson and Ruby (2007), we can say that a resource is “anything that is important enough to be referenced as a thing in itself”. Representations are not the resources themselves, but are the means for retrieving and manipulating actual resources. For example, an image or an XML file might represent a physical person. By manipulating those representations we can change the way the physical person is “used” in the digital system, though without “modifying” the actual person.

The most important principles and constraints described in Fielding (2002) that are relevant for the purpose of this paper are:

- **Addressability**: Every resource must be addressable by a well-defined resource identifier.
- **Uniform Interface**: A single interface, with a well-defined semantics, must be used for accessing and manipulating all the resources.
- **Hypermedia as the engine of application state**: By leveraging addressability, the execution of the application is obtained by following hypermedia links to resources that represent the “next state” of the application.

These principles are implemented in the Web Platform in the following way: addressability is given by using Uniform Resource Identifiers (URIs) that provide a “simple and extensible way for identifying an abstract or physical resource” (Berners-Lee, Fielding, & Masinter, 1998). URIs are used extensively in order to address any resource, and to operate on their representations. URIs also enables hypermedia features that are used as the primary mechanism to make applications change their state. By providing links embedded in resource representations, applications may evolve by “visiting” those links and discovering new or related application states.
Related Content

Filtering Structures for Microblogging Content
www.igi-global.com/article/filtering-structures-for-microblogging-content/128838?camid=4v1a

Vehicular Networks Security: Attacks, Requirements, Challenges and Current Contributions
www.igi-global.com/article/vehicular-networks-security/1371?camid=4v1a

Differential Evolution with Self-Adaptation
www.igi-global.com/chapter/differential-evolution-self-adaptation/10291?camid=4v1a
Relational Representations and Traces for Efficient Reinforcement Learning
www.igi-global.com/chapter/relational-representations-traces-efficient-reinforcement/60929?camid=4v1a