ReWiRe: Designing Reactive Systems for Pervasive Environments

Geert Vanderhulst, Kris Luyten, and Karin Coninx

Hasselt University – transnationale Universiteit Limburg – IBBT
Expertise Centre for Digital Media Wetenschapspark 2,
3590 Diepenbeek, Belgium
{geert.vanderhulst,kris.luyten,karin.coninx}@uhasselt.be

Abstract. The design of interactive software that populates an ambient space is a complex and ad-hoc process with traditional software development approaches. In an ambient space, important building blocks can be both physical objects within the user’s reach and software objects accessible from within that space. However, putting many heterogeneous resources together to create a single system mostly requires writing a large amount of glue code before such a system is operational. Besides, users all have their own needs and preferences to interact with various kinds of environments which often means that the system behavior should adapt to a specific context of use while the system is being used. In this paper we present a methodology to orchestrate resources on an abstract level and hence configure a pervasive computing environment. We use a semantic layer to model behavior and illustrate its use in an application.

1 Introduction

Although pervasive computing environments have gained much importance over the last years, they remain among the most complex environments to develop interactive software for. Generic development environments that explicitly target ambient spaces are scarce because of several reasons:

- **Lack of engineering approaches**: most pervasive applications are ad-hoc coded and hence are only applicable in just one situation [5].
- **New middleware requirements**: middleware is required to abstract hardware, deal with distributed computing resources, steer the migration of user interfaces, etc [7].
- **Support for situation-aware human-computer interaction**: the context in which tasks are executed affects the user’s interaction with the system [2].

In this paper we report on the ReWiRe framework [8] which supports the dynamic composition and adaptation of behavior rules in a pervasive environment. With services and devices that enter and leave the user’s environment, the ability to support the dynamic composition of the interactive system is a strong requirement. Our approach relies on a semantic layer that captures the context of the entire environment (its users, devices, services, etc) and uses this information to configure the behavior of resources (section 3). Since orchestration is performed at an abstract level, we can mask the underlying service technologies (section 4). To accomplish this, we have underpinned our framework with semantic Web frameworks such as RDF, OWL and...
OWL-S [8]. We demonstrate our approach by means of a test-bed that illustrates how services can be orchestrated and (re)wired at runtime to take advantage of changes in the environment configuration (section 5).

2 Related Work

The emergence of Web services has lead to different solutions to coordinate distributed business processes, e.g. BPEL [1]. Pervasive services demand for similar orchestration tools that take into account the full environment context. This goes beyond dealing with preconfigured service compositions, but also involves runtime adaptation of the environment configuration whilst users are interacting with it. Muñoz et al. [5] propose a model-driven approach for the development of pervasive systems. A domain specific language (PervML) is used to specify the system using conceptual primitives suitable for the target domain.

Mokhtar et al. [4] also study highly dynamic pervasive computing environments where users need to perform tasks anytime anywhere, using the available functionality of the pervasive environment. Grimm [3] identified three requirements that should be fulfilled by systems that support these dynamic interactive pervasive environments: support for a continuously changing context of execution and make this explicit in the system design, support for ad-hoc composition of devices and services and collaboration among users should be supported out-of-the-box. With ReWiRe we tackle exactly these requirements.

3 Environment and Behavior Model

We use a semantic layer to describe the context of use of an interactive software system during its lifetime. This layer includes both an environment and a behavior model which are described by an ontology. Several (domain-specific) ontologies can be merged at runtime and offer a dynamic schema that evolves when new software components become available. The system’s configuration is linked with an instance of these ontologies. Figure 1 presents the environment and behavior ontology together with the OWL-S ontologies. The OWL-S ontology describes a service in terms of what it does (profile), how it is used (model) and how to interact with it (grounding). Although OWL-S services are usually considered to be semantically enriched Web services, a service can be any arbitrary piece of functionality that can be used in the environment. With OWL-S one can describe a service (e.g. its inputs and outputs) in a uniform way and define a custom grounding that provides details on how to invoke that service. We use OWL-S service descriptions to attach functionality to ‘resources’ in the environment model. A resource represents everything that can be included in this model, e.g. users who interact with the surroundings, devices that offer computing power, storage and input modalities, etc. Domain-specific ontologies that introduce new concepts such as light resources are merged with an upper environment ontology at runtime. The environment ontology defines ‘sensors’ and ‘actions’ to interact with these resources: