A Tool Suite to Prototype Pervasive Computing Applications (Demo)

Damien Cassou, Julien Bruneau, Charles Consel

A Tool Suite to Prototype Pervasive Computing Applications

Damien Cassou
LaBRI/INRIA
Talence, France
damien.cassou@labri.fr

Julien Bruneau
Thales Airborne Systems
Pessac, France
julien.bruneau@inria.fr

Charles Consel
ENSEIRB/INRIA
Talence, France
charles.consel@inria.fr

***Abstract*—Despite much progress, developing a pervasive computing application remains a challenge because of a lack of conceptual frameworks and supporting tools. This challenge involves coping with heterogeneous entities, overcoming the intricacies of distributed systems technologies, working out an architecture for the application, encoding it in a program, writing specific code to test the application, and finally deploying it.**

We present DiaSuite, a tool suite covering the development life-cycle of a pervasive computing system. This tool suite comprises a domain-specific design language, a compiler for this language, which produces a Java programming framework, an editor to define simulation scenarios, and a 2D-renderer to simulate pervasive computing applications.

We have validated our tool suite on a variety of comprehensive applications in areas including telecommunications, building automation, and health-care.

***Keywords*-Toolkit, Programming support, Simulation, Pervasive computing architectures, Programming paradigms for pervasive systems.**

I. Introduction

Pervasive computing systems are being deployed in a growing number of areas, including building automation, assisted living, and supply chain management. These systems involve a wide range of devices and software components, communicate using a variety of protocols, and rely on intricate distributed systems technologies. Developing pervasive computing applications is a difficult task because it requires to deal with a wide range of issues: heterogeneous devices, entity distribution, entity coordination, low-level hardware knowledge… Besides requiring various areas of expertise, programming such applications involves writing a lot of administrative code to glue technologies together and to interface with both hardware and software components. A pervasive computing system thus requires tool support during all stages of the development, from design to deployment and test.

Our approach

We propose an approach that covers the development life-cycle of a pervasive computing application in the form of a suite of tools named DiaSuite. Specifically, we have developed a design language, named DiaSpec [1], dedicated to describing pervasive computing systems. From a given DiaSpec description, a compiler provides customized support for each development stage of a pervasive computing system, namely, implementation, testing, and deployment, as depicted in Figure 1. A simulation editor and 2D-renderer are also part of DiaSuite to simulate the resulting pervasive computing application. Let us now give an overview of our tool-based methodology and its main stages.

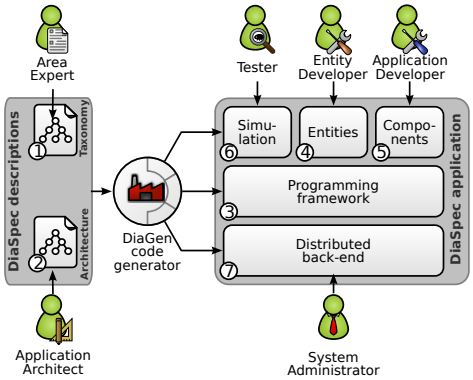

Figure 1. Development life-cycle

A. A design language for pervasive computing systems

DiaSuite provides a language, DiaSpec, dedicated to architecturing pervasive computing systems. DiaSpec allows an area expert to define a taxonomy by declaring the required entities of the system. DiaSpec also provides constructs to declare the architecture of the system in the form of context and controller components.

Taxonomy. Because of their heterogeneity, entities of pervasive computing environments need to be specified in a high-level manner to abstract over their variations. The number of existing entities also require to characterizing the ones that are relevant to a given area. To address these issues, we provide an area expert with a declarative language to define a hierarchy of entities (stage ①). Each class of entity is characterized in terms of the types of data that are gathered from the environment and the actions that are supported. Attributes are also used to

characterize properties of device instances (*e.g.,* location and ownership).

Architecture. A taxonomy definition is used as a basis to declare the architecture of pervasive computing applications. DiaSpec offers constructs to declare the architecture of an application following an architectural pattern commonly used in the pervasive computing domain [2]. Data gathered from the environment by the entities are refined by *context* components to match the application needs. Context data are then passed to *controller* components to make decisions by triggering entity actions, declared in the taxonomy (stage ②).

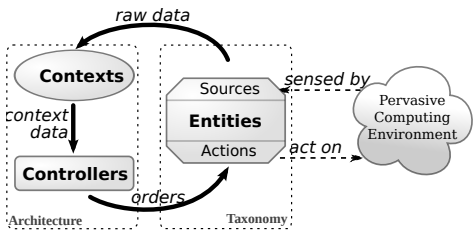

Figure 2. Architecture of a pervasive computing system

B. A generated Java programming framework

The programmer is then required to implement the entities and components (contexts and controllers). This development is supported by a Java programming framework generated by the DiaSpec compiler from the taxonomy and architecture declarations (stage ③). The compiler produces an *abstract class* for each entity or component declaration, providing *abstract methods* to allow the developer to program the application logic (*e.g.,* triggering entity actions) and *concrete methods* to support the development (discovery and interactions).

The generated programming framework has been devised to closely guide the development of the application. Implementing a DiaSpec-declared entity or component is done by *subclassing* the corresponding generated abstract class and by *implementing* each abstract method of the super class (stages ④ and ⑤).

C. A graphical editor to define simulation scenarios

Deploying a pervasive computing system for testing purposes can be expensive and time-consuming because it requires to have acquired, tested and configured all equipments and software components. Furthermore, some scenarios are difficult to test because they involve exceptional situations such as fire. To cope with these issues, DiaSuite includes a simulator for pervasive computing systems, named DiaSim [3], [4]. This tool leverages declarations provided at earlier development stages. It provides support to simulate the physical environment and execute pervasive computing applications developed in DiaSuite (stage ⑥). This is achieved without requiring any changes to the application code

1) Modeling the environment: The first step to simulate a pervasive computing application is to model the physical environment. This model can be used to test multiple pervasive computing applications. The modeling of the physical environment is realized in an editor illustrated in Figure 3. The layout of a physical environment is defined by an editor, including structural characteristics (*i.e.,* walls and areas). Then, the tester places entity instances in the environment model, using a DiaSpec taxonomy.

Figure 3. DiaSim editor

2) Defining the simulation scenarios: DiaSim provides support to define simulation scenarios to test pervasive computing applications. A simulation scenario consists of a series of evolutions of a physical environment and simulated entity instances. In a pervasive computing application, data sources sense stimuli from the physical environment; the collected data are used for context processing. Simulating the environment stimuli allows to test an application in a simulated environment. Defining the evolution of the physical environment consists of defining these simulated environment stimuli. For each stimulus needed in a simulation scenario, the tester defines how its values evolve. To ease stimulus configuration, a library of commonly used stimuli is provided. For instance, this library enables simulated persons to move around the simulated environment. Another library is provided to the developer with commonly used behaviors for entities. Yet, new stimuli and behaviors can be introduced; this development is facilitated by generated programming support.

D. A 2D renderer to simulate pervasive computing systems

Simulation scenarios are executed in the DiaSim renderer. This platform includes a 2D-graphical renderer, based on Siafu[1], to simulate pervasive computing applications. The

[1]http://siafusimulator.sourceforge.net

simulation renderer is illustrated in Figure 4. The simulated entities are displayed in the environment representation and messages appear above the entities when sensing or actuating is performed.

Fine-grained simulation can be achieved by manually injecting stimuli during the simulation and plotting trajectories to move simulated persons.

Finally, a tested application can be executed in *hybrid* environments, combining simulated and real entities. Hybrid simulation is a key feature to successfully transition to a real environment: it allows real entities to be added incrementally in the simulation, as the implementation and deployment progress.

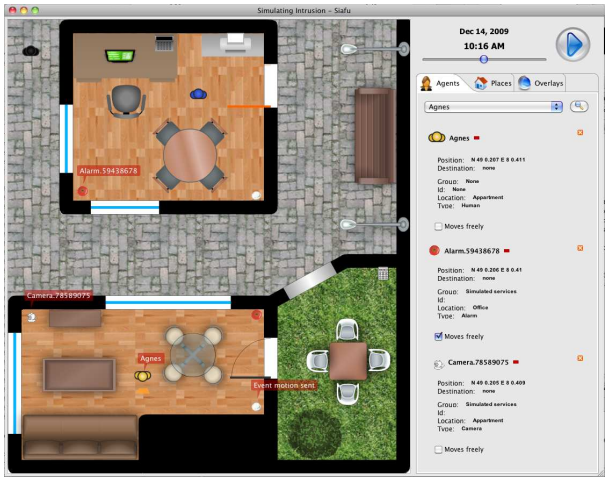

Figure 4. DiaSim renderer

II. Demonstration

The goal of this demonstration is to illustrate each step of the development cycle of a pervasive computing application using the DiaSuite tool suite. During this demonstration, we will design, implement, simulate and partially deploy a security system in a home environment.

A. Demonstrated Application

The demonstrated application is a security system responsible for securing an apartment. The security system can be activated or deactivated using a locker protected by a password. When the system is secured, it detects intrusion using motion detectors installed in the apartment. If an intrusion is detected (*i.e.,* the system is secured and a motion has been detected), the alarms of the apartment are turned on. Moreover, a supervisor is alerted by the security system when an intrusion occurs. To alert the supervisor, the security system displays a warning message on his supervision screen and turns on the alarm in his office. Finally, the system sends him an email with a picture taken by a camera covering the intrusion area.

B. Demonstration Steps

Specifying the application architecture: The first step of our demonstration shows how to specify a pervasive computing application using our dedicated language DiaSpec. We describe all required entities along with the context and controller components of the architecture. We also demonstrate the features of the Eclipse plugin we have developed to ease this step.

Implementing the application: From the previously specified architecture, we use our compiler to generate a dedicated programming framework. On top of this dedicated programming framework we develop Java implementations of the entities and components of the application.

Defining simulation scenarios: The next step of our demonstration is the definition of simulation scenarios. Scenarios will be defined using our scenario editor (Figure 3). In this step, we first define the simulated physical environment. Then, from a DiaSpec specification, simulated entities are either graphically defined using a wizard, or developed using the generated simulation programming framework. Both strategies are illustrated by the demonstration. Simulation scenarios are composed of stimulus producers. These stimulus producers are defined using the generated simulation programming framework, as well as libraries of generic stimulus producers. Our demonstration shows examples of such stimulus producers used by our security system (*e.g.,* production of motion stimuli to simulate an intrusion).

Testing the application: In the final step of our demonstration, our security system is tested against the previously defined simulation scenarios. To follow the evolution of the simulated environments, DiaSim extends an existing visualization tool: the Siafu open source context simulator. Siafu provides a 2D rendering and time-control functionalities. To illustrate the testing of applications in hybrid environments, we use both real entities (*e.g.,* a locker on an iPodtouch and real motion detectors) and simulated entities to test our security system.

References

[1] D. Cassou, B. Bertran, N. Loriant, and C. Consel, "A generative programming approach to developing pervasive computing systems," in *Proceedings of GPCE '09*, 2009.

[2] A. K. Dey, G. D. Abowd, and D. Salber, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications," *Human-Computer Interaction*, vol. 16, no. 2, pp. 97–166, 2001.

[3] J. Bruneau, W. Jouve, and C. Consel, "Diasim: A parameterized simulator for pervasive computing applications," in *Proceedings of Mobiquitous'09*, 2009.

[4] W. Jouve, J. Bruneau, and C. Consel, "Diasim: A parameterized simulator for pervasive computing applications," in *Proceedings of PERCOM '09 (Demo Session)*, 2009.