# DEVELOPMENT TOOLS FOR PERVASIVE COMPUTING IN EMBEDDED SYSTEMS (PECES) MIDDLEWARE

Ran Zhao, Kirusnapillai Selvarajah and Neil Speirs
*School of Computing Science, Newcastle University, Newcastle, NE1 7RU, U.K.*

Keywords: Smart space, Middleware, Pervasive computing, Wireless networking, Context ontologies, Modelling, testing, Dynamic addressing, Communication gateway, Eclipse.

Abstract: The main objective of the PECES project is the development of system software to enable the communication among heterogeneous devices across multiple smart spaces, breaking the traditional barrier of "smart islands" where only the services offered in a nearby spatial area can be used easily. PECES development tools help the application developer to build and test the PECES middleware based applications. This paper presents a set of tools, namely Peces Project, Peces Device Definition, Peces Ontology Instantiation, Peces Service Definition and Peces Role Specification Definition which enable application developers to build the PECES middleware based application using the novel concepts such as role assignment context ontologies and heterogeneous communication technologies. Furthermore, this paper proposes new tools which enable to model and simulate the smart space applications.

## 1 PECES PROJECT

The objective of the PECES project (PECES project, 2009) is the creation of a comprehensive software layer to enable the seamless cooperation of embedded devices across various smart spaces on a global scale in a context-dependent, secure and trustworthy manner. The increasing number of devices that are invisibly embedded into our surrounding environment as well as the proliferation of wireless communication and sensing technologies are the basis for visions like ambient intelligence, ubiquitous and pervasive computing. The benefits of these visions and their undeniable impact on the economy and society have led to a number of research and development efforts. These include various European projects such as EMMA (EMMA, 2006) that develop specialized middleware abstractions for different application areas such as automotive and traffic control systems or home automation. These efforts have enabled smart spaces that integrate embedded devices in such a way that they interact with a user as a coherent system. However, they fall short of addressing the cooperation of devices across different environments. This results in isolated "islands of integration" with clearly defined boundaries such as the smart home or office. For many future applications, the integration of embedded systems from multiple smart spaces is a primary key to providing a truly seamless user experience. Nomadic users that move through different environments will need to access information provided by systems embedded in their surroundings as well as systems embedded in other smart spaces.

The PECES project is committed to developing the technological basis to enable the global cooperation of embedded devices residing in different smart spaces in a context-dependent, secure, and trustworthy manner. The most innovative features of the PECES middleware are to enable the communication among heterogeneous devices across the different smart spaces using dynamic addressing, security and context ontologies. The PECES project has the several scientific and technical objectives such as development of a flexible ontology, development of a middleware, development of set of application development tools and validation of the abstractions using lab tests and prototype applications.

In this paper, the PECES development tools are discussed. Section 2 describes the PECES project and its novel features such as role specification, dynamic addressing, and gateway concepts. Section 3 provides an introduction to the development

Zhao R., Selvarajah K. and Speirs N..
DEVELOPMENT TOOLS FOR PERVASIVE COMPUTING IN EMBEDDED SYSTEMS (PECES) MIDDLEWARE.
DOI: 10.5220/0003406900210026
In *Proceedings of the International Conference on Wireless Information Networks and Systems* (WINSYS-2011), pages 21-26
ISBN: 978-989-8425-73-7

environments for pervasive computing. Section 4 describes the PECES development tools that have been developed by the PECES consortium for PECES middleware based application development. An additional set of tools that are currently being under development to provide features to test and analyse the smart space based application is given in Section 5. Conclusions are presented in Section 6.

## 2 PECES MIDDLEWARE

The PECES project consortium has built the targeted cooperation layer on top of BASE middleware (Becker, Schiele, Gubbels and Rothermel, 2003). The BASE layered architecture is shown in Figure 1. This enables the project consortium to focus the development efforts on the novel and innovative features of the PECES middleware. BASE is freely available as open source under BSD license which facilitates the necessary modifications and extensions and enables the free reuse – even for commercial exploitation. The BASE middleware enables the communication between devices that are within communication range. Yet, in order to achieve the goal of providing a cooperation layer that enables the seamless interaction within and across the boundaries of a single smart space, it is necessary to extend the BASE middleware concepts. The extension of the BASE middleware focused on communication gateways, dynamic addressing and smart space and internet registry.

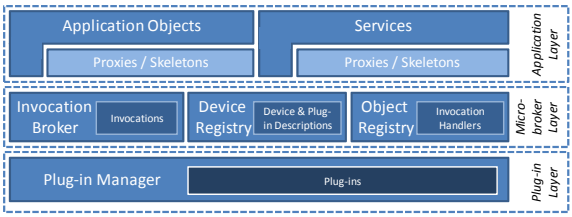


Figure 1: BASE Architecture.

### 2.1 Communication Gateway

Due to the heterogeneity of devices and communication technologies, it is not safe to assume all future devices will be equipped with the same set of communication technologies. As an example consider that a sensor node might only be equipped with ZigBee but not with Bluetooth in order to enable energy efficient communication. Thus, in order to enable a Bluetooth device to communicate with such sensor nodes, it is necessary to use a device that is equipped with both technologies as a local gateway. Similarly, due to the associated costs and other factors, not all devices will have a direct connection to a global interconnection network like the Internet. In order to enable the communication between devices that are not directly connected to the Internet, it is necessary to enable some devices to act as remote gateways for others.

PECES middleware support local gateways as well as remote gateways. The main difference between these two types of gateways is that the local gateway locally shares the required knowledge. In the remote case, the knowledge sharing should be restricted to a minimum in order to avoid the costly distribution of frequently changing information. The remote gateways need to be realized differently in that they require an external entity to distribute the information that is distributed by means of device discovery in the local case. This information is distributed by means of the registry that is specified in the PECES Communication Mechanism and Registry Interface Specification deliverable (PECES project, 2009).

### 2.2 Generic Role Assignment

Due to the continuous changes in context, the mobility of devices and the network topology can be highly dynamic in the pervasive computing environments. So that it is vital to enable pervasive computing applications such as PECES prototype applications to adapt to the continuous changes in context and device availability. The responsibility for adaptation can be shifted between different entities. In cases where changes are infrequent, a user may manually configure and adapt the system. However, if changes are frequent, manual configuration and adaptation are clearly not a viable approach as they conflict with the goal of distraction free support for tasks. In order to mitigate this, the adaptation can be automated through the application. This approach relieves the user from performing manual adaptation but it complicates the development of applications and it may result in inefficiencies in cases where multiple applications implement and use similar adaptation mechanisms. As a result, the PECES middleware is aiming to automate the initial configuration and the continuous adaptation to changes in order to shield the user and the application developer from the accompanied complications.

In order to be suitable for a broad range of different systems and in order to minimize the utilization of resources that are required for automation, PECES provides configuration and

adaptation support by means of a uniform abstraction. To create a uniform abstraction that is suitable for a broad range of different configuration tasks, it is necessary to introduce a clear separation between the result of a configuration, the computations that need to be done to produce it and the utilization of this result. This enables the reuse of the same basic mechanisms for different tasks. Generic role assignment provides such a uniform abstraction. More detailed information about the role assignment concepts can be found in PECES Addressing Scheme Specification deliverable (PECES project, 2009).

### 2.3 Smart Space

A smart space can be defined as a group of networked devices that cooperate to support their users. The boundaries of a smart space are typically defined on the basis of a geographic location, e.g. a room or a building. However, such narrow definitions are not flexible enough to support the application prototypes in the PECES project. Obviously, these smart spaces cannot be defined on the basis of a single location. For example in applications based upon a car, the whole car, i.e. the smart space itself, is mobile.

To support the smart space concept, the PECES middleware introduces three additional components which are coordinator, member and gateway. These components can be easily motivated by looking at the anatomy of the smart spaces that are identified in the PECES Use-Case Specification deliverable (PECES project, 2009). The coordinator is responsible for identifying the members of the smart space defined by the role specification. The Gateway functionality provides connectivity for other devices (for the coordinator and device) in the smart space.

## 3 DEVELOPMENT ENVIRONMENTS

The PECES project provides a set of development tools that can be used by the application developers to develop, test, and analyse their applications. Development tools also provide an environment to simulate/emulate applications. These types of development tools are economical because developers can carry out experiments without the actual hardware and it is a feasible way to test scalability of any proposed applications. For wired/wireless networked based application and protocol development, there are many simulators (NS 2, OPNET) have been proposed.

A Middleware based application framework for Active Space applications was proposed in (Roman and Campbell, 2003). The Active Space consists of the Gaia middleware OS (Roman and Campbell, 2000) managing a distributed system composed of several systems. The framework focuses on providing an application framework that leverages the functionality provided by the Gaia middleware OS to assist developers in the construction of Active Space application.

The Distributed Trust Toolkit (DTT) (Lagesse, Kumar, Paluska and Wright, 2009) proposed a framework for implementing and evaluating trust mechanisms in pervasive computing systems and introduced two new abstractions: trust groups and trust blocks. Trust groups allow associated application devices to share recorded trust data and trust computations. Trust blocks makes policy decisions based on data gathered by the computation component which implements network based trust protocols and allows the DTT to interoperate with legacy trust systems. The DTT facilitates the extension and adaptation of trust mechanisms by abstracting trust mechanisms into interchangeable components.

UbiWise, a simulator for ubiquitous computing system was proposed in (Barton and Vijayaraghavan, 2002). UbiWise concentrates on computation and communication devices situated within their physical environments. Multiple users can attach to the same server to create interactive ubiquitous computing scenarios. The devices are specified through a combination of a device-description file in XML and Java. UbiREAL simulator (Nishikawa, et al., 2006) was proposed for realistic smart space systematic testing. UbiREAL facilitates reliable and inexpensive development of ubiquitous applications where application software controls a lot of information appliances based on the state of external environment, user's contexts information.

A suite of tools which used to construct domain models and knowledge-based application with ontology was provided by Protégé (Protégé, 2004). Protégé implements a rich set of knowledge-modelling structures and actions that support the creation, visualization, and manipulation of ontologies in many different representation formats. Protégé also provide a Java API for other developer to build their own tools and applications.

DiaSuite (Cassou, Bruneau and Consel, 2010) framework includes 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. DiaSim (Jouve, Bruneau and Consel, 2009) is a part of the DiaSuite which provides a parameterized simulator for pervasive computing applications. A modelling and simulation framework is proposed in (O'Neil, Conlan and Lewis, 2010) to assist context aware system design for pervasive computing applications. The simulator provides analysis tool with information relating to the actual state of the environment and not just the sensed state that the prototype system receives. The framework also presented number of validation case studies.

The tools discussed here provide limited support for application developers, as they have been designed for different goals and concepts. Although these tools are available to support pervasive computing environment application development, they only offer little methodological support for role assignment, context-awareness and security, which are the core features of the PECES middleware. For example, Protégé may provide support for context ontology instantiation for the devices but this information may be difficult to integrate with other PECES middleware services without the use of any specific tools.

# 4 PECES DEVELOPMENT TOOLS

PECES development tools focus on configuring devices, modelling smart spaces and context dynamics and testing the novel concepts provided by the PECES middleware. The tools provide support for application developers to build PECES middleware application and simulate and analyse the smart space behaviours with respect to the context dynamics and network changes. The PECES project provides a set of tools which are integrated into the Eclipse development environment. This way, the usual development assistance provided by the Eclipse IDE can also be used for PECES focused development support.

The following sections explain the different tools which are the Peces Project, Peces Device Definition, Peces Ontology Instantiation, Peces Service Definition and Peces Role Specification Definition. The following subsections are arranged according to the development sequence that the developers would typically follow during the middleware application development.

## 4.1 Peces Device Definition Tool

As the first part of the development process, developers should use Peces Project tool to generate a PECES general project in the Eclipse workspace (for example, *PrototypeDemo* project can be created by the Peces Project tool with *ConfigurationTool*, *ModellingTool* and *TestingTool* folders). The Peces Device Definition tool now be used to define communication plugins such as IP, Bluetooth, ZigBee (e.g. *MxIPBroadcastTransceiver, MxIPMuticastTransceiver, MxSpotTransceiver*), and device functionalities (*Coordinator, Gateway, Coordinator&Gateway, Member*) and also device name. All smart space applications should define one device with coordinator functionality which is responsible of the coordination of the smart space. The devices can be placed using drag and drop method. Different colours will be shown according to the selected device functionality (e.g., Coordinator is red).

Figure 2 shows an example in which three devices are defined (*Consumer, PrintTextProvider* and *PrintNumberProvider*). The screenshot also shows three different Java projects namely *Consumer, PrintTextProvider* and *PrintNumberProvider* in the workspace. These Java projects contain PECES middleware libraries (peces-1.0.jar) and necessary Java files under the *src* folder.

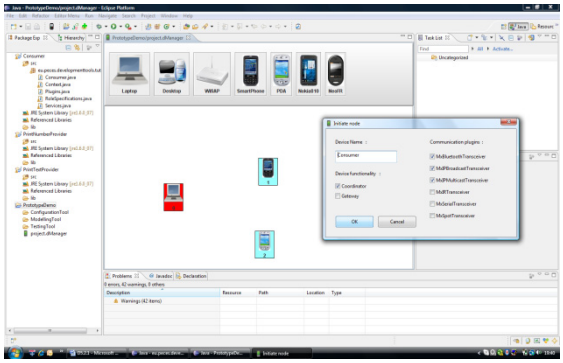

Figure 2: Screenshot of the Peces Device Definition Tool.

## 4.2 Peces Ontology Instantiation Tool

The PECES context ontologies are composed by the *SmartSpace*, *Measurement*, *Device profile*, *User Profile* and *Event* ontologies and these ontologies are freely available to download at the project website (PECES project, 2009). The PECES Context

Ontology and Query Specification deliverable (PECES project, 2009) clearly explains the dependencies among them, as well as the external ontologies which provide a basis for the PECES concepts and properties. The core ontology for representing contextual information of a smart space is the *SmartSpace* ontology. The basic concepts to model the contextual information of a smart space are *Device*, *Context*, *Location* and *Service*.

The Device profile ontology provides vocabularies to model specification of devices inside smart space. There are three categories of devices defined in the PECES prototype application which are *PECESEmbeddedDevice*, *Accessory* and *SensorDevice*. *PECESEmbeddedDevice* represents those embedded devices that deploy the PECES middleware. There are three categories of embedded devices according to their role inside a smart space, namely gateway, coordinator and member. In order to specify which kind of accessories an embedded device has, the property *hasAccessory* can be used to link a *PECESEmbeddedDevice* instance to an *Accessory* instance. Accessory instances are *Keyboard, Touch Screen, Speaker, Screen* and *Microphone*. In addition to this, *SensorDevice* has two sub-concepts: *Detector* and *MeasuringSensor*. A *MeasuringSensor* instance represents a sensor which can measure a measurement such as light, noise, temperature, etc.

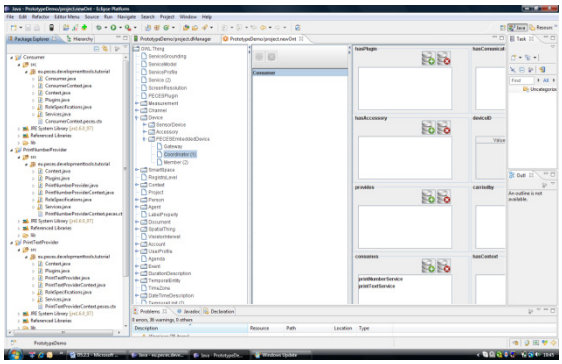

Figure 3: Screenshot of the Peces Ontology Instantiation Tool.

Peces Ontology Instantiation tool enables application developer to instantiate the devices. The tool automatically loads the participating device name and its functionality information from the project.xml file which is generated by the Peces Device Definition tool. The Peces Ontology Instantiation tool provides GUI where application developers can add instances and link properties. When the instantiation process is completed, the tool creates a RDF file (project.owl) in the *PrototypeDemo* project *ConfigurationTool* folder and also creates *.peces.ctx files which contains the device context information for each devices. Those device *.peces.ctx files are placed in the appropriate device Java project and once the file is placed in the Java project, the tool will automatically create necessary Java files for the middleware from the *.peces.ctx files.

Figure 3 shows an example in which three devices defined (*Consumer, PrintTextProvider* and *PrintNumberProvider*) in the Device Definition tool are automatically loaded by the Ontology Instantiation tool. Two new services (*printTextService* and *printNumberService*) are defined here with Ontology Instantiation tool. The *printTextService* linked (via provides) with *PrintTextProvider* device and *printNumberService* linked (via provides) with *PrintNumberProvider* device. Also the Consumer device is linked (via consumes) with both *printTextService* and *printNumberService*.

In addition to the tools that have been described here, the PECES development tools also include other tools such as Peces Service Definition tool and Peces Role Specification Definition tool. The Service Definition tool provides features to generate necessary *Proxy* and *Skeleton* for the application services defined in the Ontology Instantiation tool. The “Scope” option in the Peces Service Definition tool determines at which scope (Internet, Space, and Device) the service will be published. The Peces Role Specification Definition tool provides an interface where developers can define the different rules that the application will use to dynamically form groups (smart spaces) of collaborative devices. The Role Specification Definition tool automatically generates all necessary Java code in the required device Java projects to define and instantiate it using the middleware. Due to space limitations we are unable to describe these tools in more details here but they may be provided upon request.

## 5 FUTURE WORK

Section 4 only presented the tools that have been developed to provide support for application development using PECES middleware. The next phase of the development tools will provide support for modelling (Peces Event Tool), emulating and visualising (Peces Testing Tool) the PECES middleware based smart space application.

The Peces Event tool will be used to generate several events such as device switch ON, device

switch OFF, context changes and network topology changes. This tool will also provide features (Event Diagram Editor) to define the generated events in a sequence. The defined event sequence information will be stored in a XML file (events.xml) and placed in the *ModellingTool* folder. The events.xml file will have different tags for different events with delay time information.

The Peces Testing tool will provide support to execute the applications (each device application considered as separate JVM) built by the other tools. The Testing tool will load necessary device related information from the project.xml file and event related information from the events.xml file and display in the Testing tool Multi-page editor page. Application developers will be able to define necessary time for the test. All middleware and application related information will be appended into a single log file with the specific JVM (with device name) information and the log data will be used to visualise and analyse the smart space features.

## 6 CONCLUSIONS

One of the main objectives of the PECES middleware is to provide a cooperation layer that enables seamless interaction and coordination among devices in and across smart spaces in a secure manner. This paper presented a set of tools which provide support for PECES middleware based application development. The tools provided support for device configuration, instantiation, role specification and service definition. This paper also outlined a new set of tools which are currently under development to provide features to model and simulate the smart space applications. The new set of tools will provide features for dynamics modeling testing visualizing the smart spaces. It is the authors' intention to present at the conference that the tools already developed as well as the new set of tools which are currently under development.

## ACKNOWLEDGEMENTS

The work presented here is sponsored by EC under FP7 programme (FP7-224342-ICT-2007-2) and authors also would like to thank all the project partners for their contributions.

## REFERENCES

Barton, J., and Vijayaraghavan, V., 2002. *UBIWISE, A Ubiquitous Wireless Infrastructure Simulation Environment*, [online] Available at: <http://www.hpl.hp.com/techreports/2002/HPL-2002-303.html> [Accessed December 2010].

Becker, C., Schiele, G., Gubbels, H., and Rothermel, K., 2003. BASE - A Micro-broker based Middleware For Pervasive Computing, In IEEE, *1st International Conference on Pervasive Computing and Communications*, pp. 443-451, Fort Worth, USA, March 2003.

Cassou, N., Bruneau, J., and Consel, C.. A tool suite to prototype pervasive computing applications, In: IEEE. *8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, Mannheim, Germany, March 2010.

EMMA Project, (2006). [online] Available at: <http://www.emmaproject.eu> [Accessed May 2011].

Jouve, W., Bruneau, J., and Consel, C., DiaSim: A parameterized simulator for pervasive computing applications, In: IEEE, *the Seventh Annual IEEE International Conference on Pervasive Computing and Communications*, Galveston, Texas, March 2009.

Lagesse, B., Kumar, M., Paluska, J., and Wright, M., 2009 DTT: A Distributed Trust Toolkit for Pervasive Systems, In: IEEE, *Pervasive Computing and Communications Conference*. Galveston, Texas, USA, March 9-13, 2009.

Nishikawa, H., Yamamoto, S., Tamai, M., Nishigaki, K., Kitani, T., Shibata, N., Yasumoto, K., and Ito, M., UbiREAL: Realistic Smartspace Simulator for Systematic Testing, In: UbiComp. *8th International Conference on Ubiquitous Computing,* LNCS4206, pp. 459-476, Irvine CA, USA, September. 2006.

O'Neil, E., Conlan, O., and Lewis, D., 2010. *Modelling and simulation to assist context aware system design*, [online] Available at: <http://sim.sagepub.com/content/87/1-2/149>, [accessed May 2011]

PECES Project, (2009). [online] Available at: <http://www.ict-peces.eu> [Accessed May 2011].

Protégé. (2010). [online] Available at: <http://protege.stanford.edu/> [Accessed May 2011].

Roman, M., and Campbell, R., 2000. Gaia: Enabling Active Spaces, In: SIGOPS EW'00, 9th ACM SIGOPS European Workshop, pp.229-234, Kolding, Denmark, September 2000. New York: USA.

Roman, M. and Campbell, R., 2003. A Middleware-based Application Framework for Active Space Applications, In: ACM/IFIP/USENIX, *International Conference on Middleware*. Rio de Janeiro, Brazil June 2003.