Integrating and Demonstrating Pervasiveness in a Scenario Driven Approach

Michael ANGERMANN[1], Sarah McBURNEY[2], Christoph KUHMUENCH[3], Fiona MAHON[4], Jelena MITIC[3], Patrick ROBERTSON[1], John WHITMORE[5]

[1]*Institute for Communications and Navigation, DLR, PO Box 1116, 82230 Wessling, Germany, Email: <firstname.lastname>@dlr.de :* ***correspondence authors***
[2]*School of Math & Comp Sc, Heriot-Watt University, Riccarton, Edinburgh, UK, Email: ceesmm1@macs.hw.ac.uk*
[3]*Siemens AG, CT SE 2, Munich, Germany Email: <firstname.lastname>@siemens.com*
[4]*Telecommunications Software & Systems Group, Waterford Institute of Technology, Waterford, Ireland, Email: fmahon@tssg.org*
[5]*Lake Communications, Ireland, Email: john.whitmore@lakecommunications.com*

Abstract: This paper describes the effort that was taken to demonstrate a pervasive computing platform that was integrated with advanced networking functionalities within the Daidalos project. The Demonstrations were aligned to the two key scenarios which were the basis of the project's development process and served as an exposition of the key innovations and the overall vision. The use of a new visualisation tool is described that is used to accompany the demonstration in real-time in order to illustrate the platform's major features. We conclude with a critical review of the effectiveness of the demonstrations in order to convey the innovations and important processes.

1. Introduction

This contribution will outline the process of demonstrating the pervasive Systems platform being developed in the Daidalos project [1, 2]. An important aspect of all technical projects that produce running prototypes a well planned demonstration process that is able to convey the work achieved in the technical development. The scope of this paper is to address the following:

1. How effectively demonstrate key innovations: This entails the identification of these innovations and an evaluation on how to demonstrate them effectively. This task has to resolve conflicts that result from this evaluation process and thus to prioritize the choices appropriately. Stand-alone demonstrations are a means in order to focus some demonstrations on specific innovations.
2. How to visualize and demonstrate difficult concepts and complex processes. A common problem facing developers of ambient computing technology is the fact that a smooth user experience appears simple and the demonstration itself does not convey the solution to the specific problem. Common examples are the handover of a session or a seamless communications handover. To address this, a tool was developed within the project that serves as a real-time exposition of the messages and state of important system components.
3. A critical demonstration review: The goal is to assess how well the demonstrations were chosen and executed to convey the above key innovations and integration aspects.

The vision of Daidalos is of a world in which: 1) Mobile users can enjoy a diverse range of personalised services – seamlessly supported by the underlying technology and transparently provided through a pervasive interface. 2) Mobility has been fully established through open, scalable and seamless integration of a complementary range of heterogeneous network technologies. 3) Network and service operators are able to develop new business activities and provide profitable services in such an integrated mobile world. It was decided that two demonstration phases should take place in order to convey this vision.

After the first year, a number of stand-alone demonstrations were prepared that convey a number of the key concepts. This tight schedule allowed for only a 2 month integration period after the initial architecture was developed and the first implementations of subsystems had been performed. In the following year, the architecture was refined and two further pervasive systems demonstrations were conducted according to the overall project scenario which was chosen as the demonstration baseline. The flagship demonstration (called the fully Integrated "Nidaros"[1] demonstrator) served to integrate all major components of all 3 technical workpackages of the Daidalos project. A second "lite" version of this demonstrator integrated all functions of the pervasive systems layer as well as the reference third party services, as well as a subset of lower layer functions.

2. Scenarios as a starting point

The Daidalos project is based on two basic scenarios that serve to develop the overall system requirements, the architecture and to align all demonstration activities to a common story [3]. These two scenarios "Mobile University" and "Automotive Mobility" target end-users who are portrayed in their use of future telecommunications and pervasive computing services in different environments.

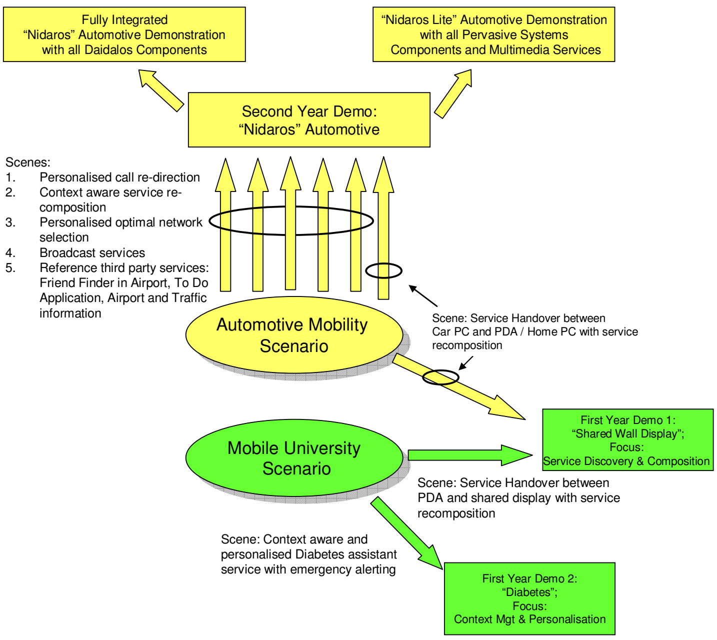

Figure 1: Mapping of the scenes to the four demonstrations with pervasive systems. The two main scenarios serve as the starting point for the first and second year demonstrators. In the second year the focus was on a large subset of the "Automotive Mobility" scenario called "Nidaros".

The Mobile University scenario addresses students and staff of a campus who use the Daidalos system to interact with each other, communicate, be entertained, enhance their use of university infrastructure, and to become better organised and supported in their special daily activities. The Automotive Mobility scenario includes such functions as transferring

[1] The scope of this demonstration was decided at a Daidalos meeting in Trondheim (Norway), a town known formerly as "Nidaros"

active user-sessions from devices outside of a car (e.g. a PDA or Home PC) to a Car PC, and adapting these services as the network environment changed during travel. Both scenarios share enough similarities to yield a common platform but each introduces a new set of innovations. In Figure 1 we illustrate the relationships between the two main scenarios and the four demonstrations; two in the first year and two in the second.

For the first set of demonstrations we address some of the common scenes and features of the Mobile University and Automotive scenario. In particular, we choose a scene where a user-session is transferred from one device to another newly discovered device (a shared wall-mounted display, in our example), and re-composed and personalised in the process. The other scene shows how a student suffering from Diabetes uses a personalised, sensor-based third-party blood sugar level (BSL) monitoring service to guide her medication and food intake, and to follow a multi-level alarm plan should her BSL go out of range. In final stages of such an escalation a personalised messaging service would alert nearby friends and inform a rescue service.

The second set of demonstrations is more fully integrated and requires real-time visualisation to illustrate the platform functions. They centre on an employee who moves between his home, his car and an airport environment using different devices and services. As he changes his surrounds his services automatically migrate to the appropriate devices and take into account the users' current situation. The integration with the other system layers include authentication and sign-on, broadcast service integration, personalised network selection, SIP messaging, and support for the transfer and re-composition of multimedia services.

3. Key Innovations Conveyed by the Demonstrations

The Daidalos consortium developed a deployable service platform with specific support for mobile and pervasive systems [1]. This pervasive service platform (PSP) enables the straightforward development of advanced personalised services, available to the mobile user working in an environment rich in computing and wireless communication components. Supporting the deployment of pervasive services and integrating the PSP with an underlying mobile network is the key innovation of the project. Based on the user-centred approach, the following key aspects of innovation can be identified:

Context-Awareness – the adaptability of services needs to be based on awareness of both user and service contexts. Context awareness is the ability to know the environment of the user and the services as they constantly change. These factors can be as simple as *time* and as complex as emotional state. One of the primary innovations in the area of context awareness is the integration of a multi-entity context module, so that a fully functional, integrated solution for context-aware, personalized, rule-based and event driven service discovery and composition can be provided to large a number of users. Another innovation of the context module is the pervasive context model. Our context modelling approach is generic, extensible, addressing temporal aspects of context and the fact that context information is highly interrelated, distributed and scalable. Finally the context module is capable of federating context information between an arbitrary number of domains.

Proactiveness – in addition to being aware of their environment and its changes, services must be able to adapt to those changes, where the user would require. Ultimately, this is the reaction of the service to context changes, and is what defines a service as being context aware.

Personalisation – users tend to put their own stamp on the technologies they use, from things as simple as colours and backgrounds to fundamental attributes such as language. This usually requires human interaction with technology, which if avoided and automated can increase the pervasiveness of the service. Two key innovations developed in this area are personalised redirection of multimedia sessions including messages, voice calls and video streams and personalised service selection and composition [5].

Service Composition – In order to solve a specific task, services often need to interact with each other to form composite services. The process of composing services depends on user context and preferences and hence, may also imply a need for dynamic adaptation (re-composition) in the case of changes. The innovations here are: The architecture of a pervasive service management component (PSM) provides for a pluggable discovery mechanism, i.e. arbitrary low-level discovery mechanisms like UPnP, SLP etc, can be added. Furthermore, the platform offers a service model that allows for (re-)composition of services. Finally, the platform supports session mobility, i.e. a service instance can be moved between platforms running on different devices.

Security and Privacy – come into play when access to sensitive user-related data is required. The most obvious example is the user's context information. Daidalos follows the approach of multiple identities for protecting privacy. This means, that a user can use several identities and disclose in the context of each identity only a small amount of personal data. Thus, no entity knows all the sensitive information about a user but only a small portion – the data, which is explicitly necessary in the respective context. The platform incorporates a flexible privacy policy negotiation process to negotiate agreements from various different fields using a common formal ontology and reasoning. Moreover, multiple identities are used within one composed service and dynamic identity selection is based on the negotiated privacy policy [4].

4. Demonstrations

4.1 First generation Demonstrations

The first public workshop and Audit were held in December 2004, and thus served as the first chance to demonstrate the high level of intra-WP4-integration which had been achieved at this time.

Many of the innovations are not just a result of a single subsystem, but can be seen as a kind of emergent quality from interactions between subsystems. As a result, for this first demonstration the choice fell upon two demonstrated scenarios: 1) "Service Discovery & Composition" and 2) "Context Management & Personalisation":

In the first scenario, the user combines the Enabling Services with three 3rd party sub-services to provide an application called the "Slideshow":

(1) A photo service, which allows for transmission of arbitrary photos,
(2) A PDA Display service that displays these arbitrary photos on the user's PDA
(3) A Wall Display service that displays these arbitrary photos on a wall display.

Based on rules and personalization settings, stated by the user, the service is recomposed when new services come into reach of the user. This scenario primarily serves as an overall test of the architecture since conceptual problems and errors become clear.

The second scenario, informally known as the "diabetes scene" involves two actors "Maria" and "Dani". Initially, all nodes start up and upon the initialisation of the Context Management Systems (CMs), the blood sugar level (BSL) and location sensors (monitoring the users' BSL and location respectively) are registered to the sensor manager of the CM residing on the users' devices (BSL and location sensors are registered to Maria's PDA, and a location sensor to Dani's PDA). Then the CM begins to federate Maria's and Dani's context information at every context update. Maria's context is federated between her PDA, a central server and the Hospital Monitoring Service (HMS), while Dani's context is federated between his PDA and the central pervasive systems server. A rule is in place checking constantly if Maria's BSL sensor value is within the acceptable range (BSL-rule).

The two users log in their PDAs. At some point in time, Maria's BSL context data is changed so that it lies out of norm and the new value is federated from her PDA to the HMS and the central pervasive systems server. Then, an alarm is raised in the HMS (triggered by the PSM), and Maria is directly informed by the personalised redirection service (RS) (which itself is triggered by the BSL-rule that receives a BSL context update event fired by the CM) that her blood sugar level requires her to eat, or take an insulin injection. If she responds to the message, no further action is taken by the system. However, if at some point, the BSL sensor indicates that Maria's condition is reaching a critical level and that Maria is possibly entering a diabetic coma, the BSL-rule again fires (due to the BSL context update event sent by the CM to the REM), triggering an alarm in the HMS (via the PSM), while also trying to solicit a response from Maria by triggering an auditable alarm at her PDA (via the RM). If Maria does not respond, an emergency signal is triggered at the HMS that requires the system to notify the nearest ambulance and emergency medical team of Maria's location and situation (her location is retrieved from the CM). Then, Maria's nearest friend location is retrieved by the RS from the CM. Dani is the friend located near by. The RS sends an emergency alarm to Dani (informing him about Maria's critical condition), along with Maria's location, so that he can help her.

5.2 Second generation Demonstration

The second generation demonstration (Nidaros "Lite") is a stand alone demonstration with its purpose being, to give an overview of the full Nidaros scenario while highlighting the functionality of the Pervasive Service Platform (PSP). The demonstration focuses on the user experience and application level – not on details in the lower network layers, therefore several aspects from these layers were simplified and partially simulated as described below.

In order to run through this scenario, all pervasive subsystems, a number of special 3rd party applications and the Multi-Media Service-Proxy User-Agent (MMSP-UA), Streaming Server, DVB Multicast Server and MMSP modules were integrated together in a separated simplified testbed. Seven notebooks representing Bart-PDA, Boss-PDA, Rossalyn-PDA, Car-PC, Home-PC, central pervasive systems server and a visualization server were connected via a single switch and static IPv6 connections were established between all the machines. This setup was less complicated and more robust compared to the fully integrated Nidaros test-bed.

The main differences between Nidaros "Lite" and the fully integrated test-bed were:

- No integration of Mobile IP into mobile terminals – no terminal mobility (all such changes were simulated by a manual context emulator)
- No integration of different network technologies (TD-CDMA, WLAN), no network handovers (simulated via context emulator)
- No real Quality of Service control (QoS) – MMSP-UA worked against dummy QoS
- No integration into real car
- No integration of Intelligent Interface Selection (IIS)
- DVB-T multicast was replaced by LAN multicast

All scenes from the Nidaros scenario were then executed on the Nidaros Lite testbed which, along with the visualization tool which had been scaled to highlight activity within the PSP, provided a demonstration of PSP functionality and user experience at application level.

4. Visualisation

Due to the fact that the involved platform components by definition "act behind the scenes" and are often not visible on the actual user interface, it is a challenge to illustrate the functionality of these components. Therefore, a slide-show presentation was used for the first demonstrations in order to illustrate the functionality of the platform components; it was forwarded synchronously with the flow of the demonstration. However, the preparation of a presentation that can handle the increased scope, complexity and flexibility of an extensive, fully integrated demonstration is not feasible. As a result of this shortcoming, a visualisation tool was developed that allows a real-time display of the workings of main system components during a demonstration, and indeed also during development, deployment, and integration testing. The requirements on the visualization tool are:

- The visualization system should display block diagrams, where the blocks (called knots) are placed on a background image - these blocks represent major system components. The layout is to be defined by configuration files.
- A Multi-client / server based approach: Clients send messages to the server, which then does the visualization by displaying them on the appropriate blocks. Clients are responsible for creating their respective messages.
- The protocol to be used between client/server should be simple to implement and supported efficiently over different networks (HTTP over TCP/IP was chosen).
- Messages and blocks should support color coding (e.g. error, standard, warning, ...)
- Arrows should be supported that connect knots. These arrows can be static or dynamic (in terms of their color). Dynamic arrows should be addressable by special messages whereby they can act as simple visualisation of the amount of data flow between two blocks (we refer to this as arrow "heating" where the temperature relates to traffic).
- There should be simple APIs that take clients' logging messages via method call and generate the relevant visualisation commands.

Upon start-up, the implemented visualisation server reads a configuration file that specifies the topology of the layout and the knots and arrows, and a file that defines how incoming messages are translated, and the background image file. The translation is necessary for a greater convenience for the developer by mapping program-code-names of subsystems to demonstration-target-friendly names.

For the second set of demonstrations the following basic visualization guidelines for creating visualization messages from within major pervasive subsystems are used:

- Context Management (CM): Every time Context is updated and also when updating federated Context across different deployment nodes.
- Rules and Event Management (REM): Every time a Rule is triggered to transfer a service or change a service status or a trigger that results in a change of Context.
- Pervasive Service Management (PSM): Every time applications are started by PSM or when applications are stopped by PSM as well as during service discovery.
- Interface to Multimedia and Call Control Interface (MCCS): Every time when multimedia sessions are changed or when lower level interfaces are invoked.

- VoIP and Newscast Application: Every time when their sessions are changed or the respective service is started or stopped.
- Redirection Service (RS): Every time when RS redirects an incoming call or when RS accesses REM or MCCS.

Figure 2 depicts the top-level layout for the Car-PC view of "Nidaros" after some messages have been passed. It shows the simulated status of the network components (Network-SPP, Terminal-SPP, bearer services) and the messages from the major pervasive subsystems (N, PSP, T-PSP) and six third party services. The colour of the arrows indicates the presence of recent message traffic over these connections.

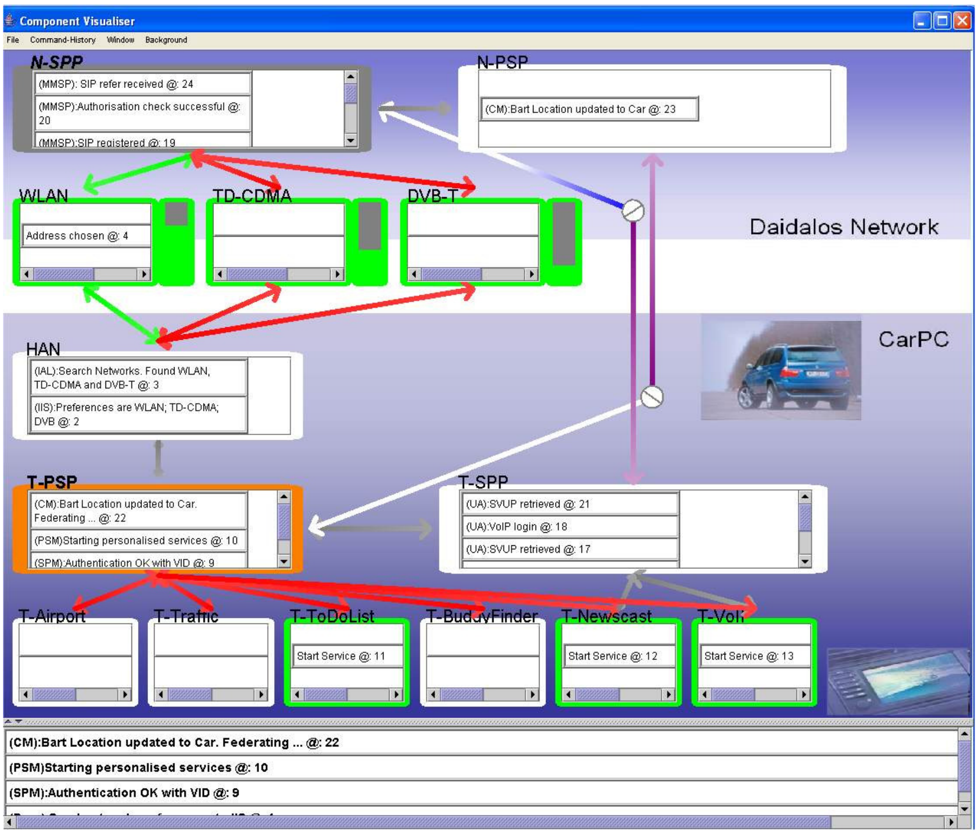

Figure 2: Visualisation Layout for the Car-PC showing a top-level view of functionality

6. Critical Demonstration Review

The purpose of the first demonstrator was to convey the capabilities of the Pervasive Service Platform (PSP), the top-level layer of the Daidalos system. The demonstrator did successfully show most of the capabilities of the PSP within a simple scenario, such as context awareness, personalisation of services, dynamic composition and recomposition of services; all based on the ultimate aim to create a *platform* that facilitates the creation of an environment of pervasive services. It was less successful in showing the Security and Privacy Management (SPM) subsystem. Although this is an important component of the platform, it is not easy to visualise or recognise in a conventional demonstrator. Additionally, the impression of the demonstrator was that of a set of services, rather than a service container that was controlling these services. Therefore, this aspect of the demonstration was lost in the midst of the services. The lesson from this was that the SPM system should be visualised in addition to the normal demonstrator to allow the specific capabilities of this system to be made clear. In addition, focus should be drawn to the functionalities of the PSP and how it acts on the services that it manages.

The purpose of the second ("Nidaros") demonstrator was to convey the entire Daidalos system as an integrated entity, and to show how it could facilitate pervasiveness on all levels of the network within a complex scenario. Given this overall goal, some specific innovations of the PSP were not focussed on. Most were demonstrated successfully, directly through the GUIs of the demonstrator or indirectly with the Visualisation system that linked directly into the running prototype. From SPM it showed the complex process of negotiating privacy policies between services and users, the selection of an appropriate VID as a consequence of the agreed policy as well as the overall flow of control inside the SPM subsystem. All key concepts of the Rules and Event Management (REM) subsystem, such as cascading of rules or inference of context attributes were successfully utilised. In particular, the system's context awareness was successfully demonstrated by autonomous reactions on various context attribute changes. However, context changes were entered into the system by an operator (through a sensor simulator GUI) – in future, real sensors should replace this. Regarding the Personalisation subsystem, we showed the personalisation of services, the selection of the most appropriate services to the user and personalised redirection of SIP calls.

Some aspects of the platform, however, were not represented since the same services were not used by different users. Therefore, the differences between how services would be presented to different users based on the actions of the Personalisation subsystem was not always clear. The Pervasive Service Management (PSM) subsystem plays an important role throughout the Nidaros demonstration, and its capabilities were made very clear. It controls the service in starting and stopping, discovery and recomposing and service transfer. All these capabilities are apparent in the demonstrator, for example, when the Traffic Information Service is recomposed to become the Airport Information Service, by simply switching one of the three constituent components that creates this overall service. Again, the management of service transfer is shown when the Newscast Service transfers from the Home PC to the Car-PC. The Context subsystem was also well represented in the Nidaros demonstrator, for the storing of the context attributes, the context federation where context values federate across the system and the interaction with the REM to result in action occurring in the system.

To conclude, while the demonstrations were able to convey the overall vision, care has to be taken to make clear that not the services, but the *service management underneath* is being demonstrated.

Acknowledgements

The work described here is based on results of the project DAIDALOS, funded under the European Community's FP6 Programme. Apart from this, the European Commission has no responsibility for the content of this paper. The information in this paper is provided as is and no guarantee or warranty is given that it is fit for any particular purpose. The user thereof uses the information at his sole risk and liability. The authors wish to thank all colleagues in the Daidalos project developing the pervasive system.

References

[1] "Developing Pervasive Services for Future Telecommunication Networks," B. Farshchian, et al., in *Proc. WWW/Internet 2004*, Madrid, Spain, October 2004.

[2] The Daidalos Integrated Project, www.ist-daidalos.org

[3] On the use of a Scenario-based Approach in identifying Ubicomp Application Sets, F. Mahon, T. Pfeifer, B. Jennings, UbiApp Workshop, Pervasive 2005, Munich, May 2005

[4] "Pervasive systems: Enhancing trust negotiation with privacy support", K. Dolinar, et al., Int. WS on Research Challenges in Security and Privacy for Mobile and Wireless Networks, Miami, November 2005

[5] "Personalized Dynamic Composition of Services and Resources in a Wireless Pervasive Computing Environment", H. Williams et al., 1st Int. Symposium on Wireless Pervasive Computing, Phuket, Thailand, August 2005