

# Daidalos: A Platform for Facilitating Pervasive Services

Robert Mullins<sup>1</sup>, Fiona Mahon<sup>1</sup>, Christoph Kuhmuench<sup>2</sup>,  
Micheál Crotty<sup>1</sup>, Jelena Mitic<sup>2</sup>, Tom Pfeifer<sup>1</sup>

<sup>1</sup> Telecommunications Software & Systems Group, Waterford Institute of Technology,  
Waterford, Ireland  
{fmahon, rmullins, mcrotty, tpfeifer}@tssg.org

<sup>2</sup> Siemens, Munich  
{christoph.kuhmuench, jelena.mitic}@siemens.com

**Abstract:** The multi-partner research project Daidalos is focused on the integration of various fixed and mobile communication technologies such as Ethernet, WLAN, TD-CDMA and DVB-T, and the creation of a pervasive services framework. That framework leverages the underlying communications infrastructure for the deployment of innovative multimedia services for consumption by both businesses and personal users.

The framework embodies technologies and research work drawn from several different areas to address the requirements of pervasive services: security and privacy, accounting and charging, location and context awareness, personalisation, rules and event management, dynamic service discovery and (re)composition.

This allows the creation of services which are aware of and capable of learning personal preferences, which are aware of where the user is and what he or she is doing, and are capable of changing their form and make-up to reflect new situations. The accompanying video illustrates an example scenario where several such pervasive services interact with the user and his environment to help him with his daily tasks in a helpful and unintrusive manner.

## 1 Introduction and Background

This paper gives an overview of the Service Discovery and Composition architecture in the DAIDALOS project (Designing Advanced network Interfaces for the Delivery and Administration of Location independent Optimised personal Services, [1]), and how it is applied to the automotive scenario as shown in the accompanying video.

Daidalos is an EU 6th framework programme Integrated research Project with 46 partners coming from the academic, research and industrial sectors. Its overall aim is to radically improve Telecommunication technologies by integrating mobile and broad-cast communications, and following a user-centred, scenario-based approach to deliver ubiquitous end-to-end services across heterogeneous technologies.

In this paper, the application of a scenario based methodology is introduced in the following. Section 3 discusses the pervasive network of the project. Section 4 presents the pervasive characteristics of the components in Daidalos, and Section 5 gives details of the automotive scenario before Section 6 concludes.

## 2 The Daidalos Scenario Based Methodology

The Daidalos project has taken a scenario-based approach to the design and development of its architecture [3]. This approach involves taking a simple life scene that has

the potential to use ubiquitous and pervasive systems, and breaking it down on many levels and axis. This brings focus and a common thread to what can potentially be an expansive and complicated area of research.

Daidalos has defined two key scenarios, from which all scenes are extracted, and on which all technical discussion, application selection, development and analysis is based. These scenarios are the *University Scenario* and the *Automotive Mobility Scenario*, both of which are given loose descriptions, giving scope for definition of many scenes, all which must fit into one of these overall scenarios.

### 3 The DAIDALOS pervasive network

In DAIDALOS a complete network is being built, with all the required basic network layers. In addition to these traditional layers, a layer to facilitate the provision of pervasive services has been added on top. This layer is called the Pervasive Service Platform. It enables a service to fulfil the requirements of being pervasive. It is similar to application containers that take a certain amount of work away from the application developers, allowing them to focus on what they really want to develop.

In an EJB container for example, application developers do not need to worry about such things as transaction handling and distribution. Developers simply develop the service functionality they require and leave the service container to take care of these standardised areas of functionality. With the DAIDALOS Pervasive Service Platform, application developers do not need to worry about such things as accessing user preferences, policy negotiation for privacy and activating and deactivating services. Service providers building services on the DAIDALOS platform have the option of using the facilities to endow their services with the characteristics listed in Table 1.

**Table 1** DAIDALOS service characteristics

Characteristic	Short description
Discoverable	A pervasive service has to expose its functionality, the supported protocols and its attributes in a standardised way (i.e. in terms of ontology), independent from the particular Service Discovery Protocol. Nevertheless, it has to support at least one (e.g. SLP).
Composable	A pervasive service has to be able to cooperate with other services by implementing the relevant Daidalos PSP interfaces.
Context-aware	A service may be context aware. This can be achieved by implementing the relevant Daidalos PSP interfaces.
Personalisable	A pervasive service may be aware of the user's personal preferences, i.e. it may have parameters that can be personalised. This can be achieved by implementing the relevant Daidalos PSP interfaces.
Private and Secure	A pervasive service may specify privacy and security requirements when accessing sensitive user-related data. When implementing the relevant Daidalos PSP interfaces the service supports a privacy and security-policy negotiation process.

The DAIDALOS *Pervasive Service Platform* (PSP) is made of five components that co-operate together to provide capabilities to the service layer above them. Those components are *Pervasive Service Management* (PSM), *Rules & Events Management* (REM), *Personalisation* (P), *Security & Privacy Management* (SPM) and *Context Management* (CM). The internals of these components and how they are able to provide the capabilities detailed in the above table are described in [2].

## 4 Pervasiveness provided by DAIDALOS

The components outlined in the previous section provide pervasive capabilities to the services built on top of them. This section analyses what elements of pervasiveness are provided.

*Context-Aware* – the Context Management (CM) subsystem takes care of monitoring context attributes of entities within the network such as services and users. It defines a model that can be used to add various sensors to the system so a wide array of attributes can be monitored. It does not make any inferences, but makes this information available to the other subsystems on the Pervasive Service Platform. Third party services may also avail of the information, however access is limited based on the privacy policies of the entities for which information is requested. Access to this information is guarded by the Security & Privacy Management subsystem.

*Context-Sensitive* – the Rules & Events Management (REM) subsystem makes the services context sensitive. The REM is aware of the values in the values held in the CM. Rules can be set with various guards such as context attribute values or ranges, which will cause a rule to fire. Rules belong to services and triggering them causing various actions to be carried out on the services, depending on how the rule is defined. Currently, rules are static and are defined as part of the service, however it is envisaged that some rules will be generated as part of a learning process within the system, and managed by the Personalisation (P) subsystem.

*Auto-Individualising* – the Personalisation (P) subsystem also takes care of auto-individualising services by setting various attributes of the service based on a general set of user preferences that the P subsystem monitors. Personalisation is also involved where a service discovery request is instantiated (perhaps by human intervention or perhaps by a rule triggered on a context value). A service discovery may in fact require a service composition, where multiple services need to be combined together before the service is returned for use to the user. An example is a word processor that needs to be combined with an appropriate printer. Personalisation can select a printer based on the user's context and preferences, for example the closest reasonably priced printer. So again this is auto-individualising of the service for its target user.

*Auto-Configuration/Installation* – the Pervasive Service Management (PSM) component handles this aspect of pervasiveness. In the example described in 'Auto-Individualising', it is the PSM that in fact manages the service discovery and composition, with some interaction with Personalisation during the process to ensure a personalised result. It can be seen in the example that the setting up of the printer is handled automatically, requiring no user intervention. Additionally, where services need to be downloaded to a machine, PSM manages this. The user does not need to know anything is being downloaded. All services in the environment appear available to the user, and will be downloaded for the user where required. The overall framework in DAIDALOS means that no user installation of services is required. Everything is automatic and in the background. PSM can even facilitate services following the user from node

to node in a consistent state, for example a video call from a car monitor to a PDA when the user leaves the car.

*Technology independent* – again, it is the PSM component that manages technology dependent services. All services in the DAIDALOS environment have a set of attributes associated with them. PSM can use these to assess if the service is appropriate for use by a particular user and their machine.

## 5 The Automotive scenario

A set of services were built on the DAIDALOS PSP as a demonstrator of the capabilities of the platform, and the pervasive behaviour it could bestow on services. The development of this demonstration took a particular scenario, and built some applications around this scenario on the PSP platform. The basic flow of the scenario is as follows:

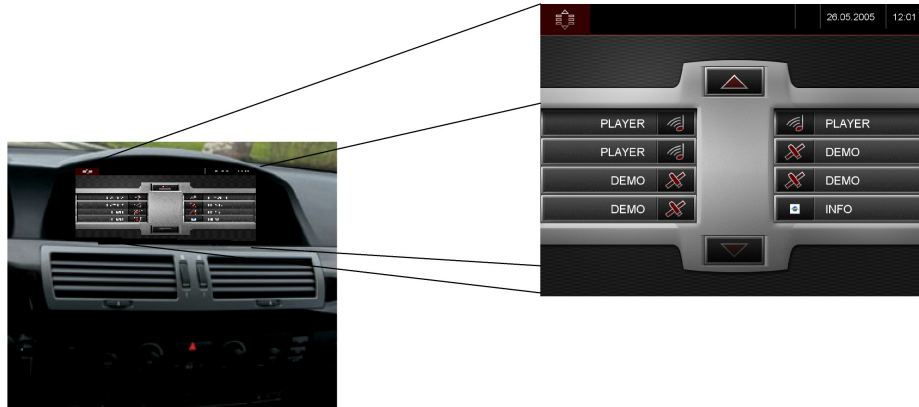
*Bart is at home watching his newscast on his home monitor. A VoIP call comes in for him on his PDA from his boss. The newscast pauses while he is talking to his boss. His boss asks him to go to the airport to pick up a customer. When Bart hangs up the call, the newscast continues where it left off. Bart walks to his car. The newscast is automatically transferred to his car. When Bart starts driving, the newscast goes into sound only mode. It runs to the end and finishes. Additionally another service on the car is the Traffic Information Service that takes traffic information off the DVB-T network that services the area. As he reaches the airport, the Information Service is re-composed to become an Airport Information service, and now takes information off the Airport WLAN. When Bart sees the customer's plane has landed, he gets out of the car causing the Airport Information Service to transfer to his PDA. Bart goes to meet the customer.*

The set of applications and their component services required for this scenario were a Newscast Service, a VoIP service and an Information Service – including the Traffic and Airport specific services. The implementation of these services followed the requirements of the PSP, so that they could be managed in a pervasive manner by the PSP.

In the scenario, there are examples of context triggers that cause certain behaviour within the system. This was implemented using rules that were triggered when certain contexts occurred, for example *Newscast* will pause when Bart's context is 'BUSY', this context value can be set by any other application, in this case the *VoIP* application. Similar behaviour is visible throughout the scenario.

An example of Personalisation is evident with the *Information Service*, which will display information based on the known native language of the user. In the demonstrated scenario, this was limited to English. Of course, in reality this could be extended to many other languages. This ability is made possible by the PSP, and requires very little effort on the part of the application developers themselves. They simply need to provide a method, which allows Personalisation to ask what parameters need to be personalised, 'language' in this case. They also need to provide a method to allow Personalisation to tell the application what the value of the parameter should be, in this case either German or English.

PSM's control of Service Discovery and Composition is also evident in the scenario. The most obvious example of composition is when the Information Service re-composes from a Traffic Information Service to an Airport Service. The service itself is composed of an 'InformationGUI' and a 'TransportInformationService'. On the car, PSM composes the BMWInformationGUI with the TrafficInterpreterService. On



**Fig. 1.** Information GUI in the automotive scenario prototype

reaching the airport, a re-composition occurs based on a context trigger. PSM listens for events from the rules triggered and will cause the re-compose accordingly, swapping the TrafficInterpreterService, that was listening to data coming off the DVB-T, for the AirportInterpreterService that listens to data coming off the airport WLAN. PSM also controls the service transfer, which is seen in two places in the scenario; when Bart leaves the house for the car, the newscast follows him to the car, and when the Airport Information Service follows him from the car to the PDA. Again, all this is possible by having the applications provide some simple interface methods that the PSP can then use to make the service pervasive.

This prototype (cf. Figure 1) as described in the scenario above is demonstrated in the accompanying video. The prototype was demonstrated on at a test site in December 2005. The demonstration used SIP based components for the voice calls and the newscast. Additionally the newscast service used a media streaming server for the data content. A redirection component was integrated with the personalisation component to allow for personalised redirection of calls.

The context system as developed by DAIDALOS was used to trigger the actions based on particular contexts, but the context system was not integrated with real sensors. Instead a dummy component was plugged into the context system to simulate certain context events.

A DVB-T network was used to broadcast the traffic information, with the airport information being broadcast over a dedicated WLAN. The test site also included a BMW with a display monitor as used in the video, to which during the demonstration the paused newscast was transferred triggered by Barts location. This monitor also displayed the traffic information from the DVB-T network and the airport information for the airport network.

The test site included several lower level components responsible for issues as QoS, billing, authentication and network handover. All the services in the demonstration were managed by the PSP.

This prototype has shown how services can be made pervasive in a simple way by the PSP platform. Additionally, this has been shown on a real network with all the issues associated with a network accounted for. These were real services giving real value in a pervasive manner.

## 6 Conclusion

The Daidalos project has shown the potential of an integrated telecommunications infrastructure and how a Pervasive Services Platform can be built upon it. This allows the delivery of valuable pervasive services to users in a real world context as demonstrated by the prototype in the accompanying video.

This project has another three years to develop a platform that addresses increasing amounts of the complex issues of pervasiveness and that will truly allow Service Providers create pervasive services. The next phase of the Daidalos project will attempt to further extend the development of Integrated Telecommunications and Pervasive Services Platforms, addressing areas such as service ontologies, network intelligence and overall robustness.

## References

- [1] Daidalos EU Framework Programme 6 Integrated Project, <http://www.ist-daidalos.org> (2005)
- [2] Daidalos Deliverable D412 – Revised architecture for Daidalos pervasive systems. (2005)
- [3] Mahon, F., Pfeifer, T., Crotty, M.: "Scenario Based Methodologies in Identifying UbiComp Application Sets". – IT&T 2005, Information Technology & Telecommunications annual conference, Cork, Ireland, 26-27 Oct (2005)
- [4] Chakraborty, D., Yesha, Y., Joshi, A., "A Distributed Service Composition Protocol for Pervasive Environments". WCNC 2004 - IEEE Wireless Communications and Networking Conference, vol. 5, no. 1, March (2004) pp. 2579-2584
- [5] Brar, A. and Kay, J.: "Privacy and Security in Ubiquitous Personalized Applications". User Modelling Workshop on Privacy-Enhanced Personalization, Edinburgh, UK, 25 Jul, (2005)
- [6] Wagner, M., Balke, W.-T., Hirschfeld, R., Kellerer, W.: "A Roadmap to Advanced Personalization of Mobile Services". - In Proceedings of the 10th Int. Conf. on Cooperative Information Systems (CoopIS) Industry Program 2002, Irvine, CA, USA, 30 Oct - 1 Nov (2002)
- [7] Lewis, D., O'Donnell, T., Feeney, K., Brady, A., Wade, V.: "Managing User-Centric Adaptive Services for Pervasive Computing". Int. Conf. on Automatic Computing (ICAC'04), New York, May 17-18 (2004) 248-255
- [8] Pfeifer, T.: "Redundant Positioning Architecture". - Computer Communications, Vol. 28 (2005) 13, pp. 1575-1585 (2 August 2005)
- [9] Amsterdam (NL): Elsevier Science
- [10] Huang, C., Garlan, D., Schmerl, B., Steenkiste, P.: "An Architecture for Coordinating Multiple Self-Management Systems". Proceedings of the 4th Working IEEE/IFIP Conference on Software Architecture (WICSA-4), Oslo, Norway, 12-15 Jun (2004)
- [11] Hirschfeld, R., Kawamura, K.: "Dynamic Service Adaptation". The 4th International Workshop on Distributed Auto-adaptive and Reconfigurable Systems, Tokyo, Japan, IEEE Computer Society, 23-26 Mar (2004)