Secure Mobile Services Infrastructures for mGovernment: Personalised, Context-aware Composition of Pervasive Mobile Services

Alan Davy
Telecommunications
Software & Sys. Group
Waterford Institute
of Technology,
Waterford, Ireland
E-mail:adavy@tssg.org
http://www.tssg.org

Fiona Mahon
Telecommunications
Software & Sys. Group
Waterford Institute
of Technology,
Waterford, Ireland
E-mail:fmahon@tssg.org
http://www.tssg.org

Kevin Doolin
Telecommunications
Software & Sys. Group
Waterford Institute
of Technology,
Waterford, Ireland
E-mail: kdoolin@tssg.org
http://www.tssg.org

Brendan Jennings
Telecommunications
Software & Sys. Group,
Waterford Institute
of Technology,
Waterford, Ireland
E-mail: bjennings@tssg.org
http://www.tssg.org

Mícheál Ó Foghlú
Telecommunications
Software & Sys. Group
Waterford Institute
of Technology,
Waterford, Ireland
E-mail: mofoghlu@tssg.org
http://www.tssg.org

Abstract: *This paper discusses service discovery, composition and adaptation, illustrating their usefulness in pervasive mobile environments in which a multitude of services are available to users. It addresses how service discovery and composition, incorporating personalization and context awareness, can provide focused sets of services tailored to a user's individual needs, shielding users from the potentially bewildering range of offered services. It is argued that because users' needs will be constantly evolving, these service sets must also continually adapt to changing requirements. The Pervasive Services Platform developed by the EU FP6 integrated project Daidalos is described and a usage scenario for it that demonstrates how service discovery, composition and adaptation can be successfully integrated is outlined.*

Keywords: Pervasive Services, Service Composition, Service Adaptation, Mobile Computing, Daidalos

1. Introduction

The continuing deployment of new communications technologies, like 3G mobile networks, is resulting in a gradual proliferation of services being offered to mobile users. In the future, the number and diversity of available services will in itself be a risk, since users may be dissuaded from searching for services they require because of the difficulty of identifying those services most appropriate to their needs. This is somewhat analogous to the difficulties many users today experience in attempting to find information on the World Wide Web. A potential solution to this problem is the introduction of facilities that would automatically identify and generate appropriate service bundles that are tailored to the needs of individual mobile users, and adapt the operation of these services as users' needs change. Such facilities are generally referred to as service composition facilities, although systems providing service composition often also include facilities for service discovery and service adaptation.

This paper discusses how service composition facilities in general, and service composition incorporating personalisation and context awareness in particular, offer the potential to simplify the life of the mobile user. Specifically they:

- Present users with only services relevant to their current context. For example, a person outside a government office may be reminded that their car tax is out of date next week and needs renewing;
- Automatically make service selection decisions based on user preferences for non-functional aspects, for example, price, or quality;
- Discover relevant services as users go about their daily lives. For example, a visually-impaired person attempting to read a time table at a bus stop may be alerted to the availability of an audio time-table service;
- Compose services to provide complete, user-centred offerings, alleviating the need for the users to themselves find related services. For example, a non-native speaker seeking information on grants available to entrepreneurs in Ireland could be provided with a composed service incorporating a service providing the relevant information in English, a service that translates that information to the appropriate language, and a service that annotates the translated information with additional information relating the information to related schemes in the person's home country;
- Dynamically adapt executing services to the user's current context. For example, if when a user is reading sensitive information on a PDA, another unknown person moves within viewing range of the PDA the screen automatically blanks, thus ensuring privacy concerns are met.

Our contention is that this kind of behaviour is only possible if there is close collaboration between service discovery, composition, personalisation, context awareness and adaptation functional components.

The structure of the paper is as follows. Initially an overview of recent work on service discovery, composition and adaptation is provided. The service composition related components of the Daidalos Pervasive Service Platform (PSP) is then presented. The operation of the PSP is explained by describing its use to provide an education service is illustrated. Finally, a discussion on the platform is given and a note on intended future work to be completed.

2. Background

This section provides a brief summary of the state-of-the-art in relation to service discovery, service modelling, service composition and service adaptation, all of which are capabilities required to deliver user-centred services in a pervasive computing environment.

2.1. Service Discovery

Within a pervasive services environment, services can be introduced and withdrawn to/from the environment in an ad-hoc fashion. To accommodate this, there must be a method of managing the discovery, registration and deregistration of these services. With an appropriate service discovery system in place, a user can introduce a new device such as a phone or a TV, and its associated services will be integrated into the current environment without any need for human interaction. The remainder of this section gives an overview of some available methods of service discovery, and mentions how they may be of relevance to a pervasive computing environment.

UPnP (Universal Plug and Play) is an architecture providing a means of enabling ad-hoc peer-to-peer connectivity between UPnP compliant devices. It provides a multi-cast notification mechanism that advertises the existence of new devices to UPnP control points that act as a local register of the visible devices. This architecture uses open standards based communications protocol allowing heterogeneous services to interact. A drawback of UPnP is that it does not scale well, as it is mostly designed for small networks.

The Service Discovery Protocol (SDP) is used to find Bluetooth enabled devices in very close proximity to the requesting device. Bluetooth typically has a maximum reliable range of 10 metres for communication. However, it is better suited to applications that are more localised than this. Such examples would include in-car phone systems making use of Bluetooth for a wire-free car kit for the phone or mobile phone to laptop communication via Bluetooth to avoid the clutter of wires or the unreliability of infrared port communications.

The Service Location Protocol is an open standard for service discovery and implementations of the protocol exist in the form of OpenSLP. A very appealing aspect of the SLP is its language neutrality; it does not have any specific language functionality in order to work. As such, it is possible to implement the protocol in C++, Java and a host of other languages.

JINI is a Java-based implementation of a universal device registration and deployment platform. As it is solely java based, it cannot interact with other non-java based services. This is a major limitation to a service discovery platform. Within the pervasive computing environment, the ability to discover a wide variety of heterogeneous services is a key feature. The Open Services Gateway Initiative (OSGi™) provides a service platform that allows developing, deploying, and managing distributed services in a coordinated fashion. It is specifically designed for Java. The platform deploys services as bundles, which can be remotely installed and managed by the platform. A major benefit of the OSGi platform is that various service discovery bridges are available. This can allow for the discovery and integration of UPnP services and Web Services.

2.2. Service Modelling

As services can be added and removed from the pervasive services environment at will, service management system(s) can have no prior knowledge of the service, or how to use them. An important feature of pervasive services, therefore, is how they are viewed or modelled within the environment. When the user or service provider introduces a service into the environment, the service itself must provide the system with details of its requirements and behaviour. The use of a service model indigenous to the introduced service allows the pervasive system to integrate the new service into the environment and use its functionality to offer the user new added value services.

One such method of defining services openly is to apply the Web Service approach. WSDL (Web Service Description Language) is a general framework for the describing network services as collections of communication endpoints capable of exchanging messages. It describes where the service is located, what operations are supported, and the format of the message to be exchanged based on how the service is invoked. WSDL does not mandate a specific communication protocol to use; it can support various bindings such as SOAP, HTTP, and MIME. The open service interface approach of web services is an ideal method of introducing new pervasive services into the environment, as the interface to the service is defined by the service itself. However, in order for a service to be useful within the pervasive environment the system must also recognise and understand the service's functionality.

With the use of ontologies, semantic meaning can be reasoned from service models by computer systems, as long as the service adheres to the semantic information or knowledge base represented by the ontology. The DARPA Agent Markup Language (DAML) Program has developed an ontology for representing the semantics of web services. Alternatively, with the Web Ontology Language for Services (OWL-s), web services can be marked up with richer semantic description to enable fuller, more automated service provisioning. An example of one benefit can be seen in the discovery of services where a search can be performed on not just name but on service functionality (Davy 2004).

2.3. Service Composition

Service composition involves combining a number of services together to create a more complete service offering. The relationship between the services determines a specific type of composition: Sequential composition is where input to one service is the output of another and Parallel composition is where the services run concurrently and do not depend on results from the others.

There must be a method defined allowing any indeterminate service interact with another. Orchestration is the manner in which services are combined together in a well-defined pattern and how they are subsequently executed in accordance with this pattern specification. Once services are discovered and registered, the system can interact with the services by using interfaces declared within their service model. As new services are discovered, they too may be integrated into the service composition to add additional functionality to the service.

There exist languages specifically aimed at the orchestration of web services. Examples of these languages are Business Process Execution Language for Web Services (BPEL4WS) and Web Service Choreography Interface (WSCI). Both of these languages are XML-based mark-ups defining the exchange of and flow of messages between web services. With BPEL4WS and WSCI, pervasive services can be chained together to offer the user new, value added services from services available within the environment. These process plans can be executed by a workflow system, which handles all message passing and service execution within the composed service.

2.4. Service Adaptation

The behaviour of a service is considered to be adaptive as long as the service can continue to perform the operations of which it was designed within a changing, and unpredictable environment (Meyer J.A 1991). The pervasive computing environment of the mobile user is constantly changing, for example, where the user is moving from location to location, new services will become available to the user. Executing services must adapt to the new environment of the user, taking advantage of any newly available services. There have been various approaches taken in addressing service adaptation, ranging from adapting network resources, to modifying service behaviour. The following section provides an overview of some approaches taken for service adaptation.

(Huang 2004a) has defined a method for services to be self-configuring. Service-specific knowledge can optimise the process of selecting services and greatly reduce the space of feasible configurations. As user or system requirements change so does the configuration of the service component. This approach solely depends on the service specific knowledge defined by the developer for each service.

(Hirschfeld 2004) talks about methods of how services themselves can be self-adapting, by the method they are designed and implemented. He combines aspect-oriented programming (Filman 2001) with computational reflection and late binding to adapt services and service platforms when changes actually require doing so, as late as possible, preferably without disruption of service. (Araniti 2003) deals with the adaptively controlling the Quality of Service of multimedia wireless applications through user preference based adaptation. This technique uses user preferences to make decisions on how to adaptively reserve network resources for the delivery of a service as network bandwidth fluctuates. A main concern with adapting services within a composite service is the problem of coordinating the adaptations effectively, so as to optimise the behaviour of the overall service. In some cases the adaptation of a service within a composition may adversely affect the operation of the composite service.

3. Pervasive Services Platform

This section presents a summary of the service composition capabilities of the Pervasive Service Platform (PSP) as developed in Daidalos (Daidalos 2005) and how this platform may be used in various service scenarios such as mGovernment. The composition manager of the PSP belongs to the Pervasive Service Manager (PSM). The PSM contains components for Service Discovery, Service Composition, Service Selection and Service Actuation, all of which are involved in the creation of the composed service offerings. Service discovery and composition in Daidalos is not a standalone function. It relies heavily for its innovative qualities within a pervasive environment on the Personalisation and Context Awareness components. Service discovery uses personalisation within service selection, to select the service offerings for a particular service type that are most suited for a specific user e.g. cheapest, best quality. Personalisation uses context parameters when personalising services, so for example a person's location, proximity to other people, buildings and services are all taken into account. Service composition repeatedly calls on service discovery when looking for services to form a composite service, and thus the composed service is completely personalised to the user's taste. Even while a service is being used it will continue to be adapted and personalised according to the user's preferences within his/her context. The following sections detail how the two areas of service discovery and service composition work. Following that is a section presenting how composition is used in the context of one particular everyday scenario.

3.1. Services in Daidalos

In Daidalos, services encompass a broad range of service types that can be classified in a number of ways. From the Daidalos perspective services can be broadly divided into:

- Daidalos Enabling Services (DES): Network functionality abstractions (including composition, personalisation and context awareness) that may be used by other pervasive value adding services;
- Value Adding Services (VAS): Pervasive or non-pervasive services defined at a high-level i.e. that are visible and deliver certain value to the end user.

These VAS may either be 3rd party or operator provided. They can encompass mobile services, hardware services, web services and many more. These services are modelled in Daidalos using WSDL. WSDL files allow the definition of service interfaces from which stubs used for communication with the services can be generated. The communication protocol used in Daidalos is SOAP.

3.2. Service Discovery Architecture

This service discovery architecture represents a part of the overall Pervasive Service Manager (PSM). Service discovery is the mechanism by which services conforming to a certain set of criteria are found. It will return all services that support any of the discovery mechanisms as used by the PSM that conform to the criteria given. The service discovery component is only one step in the overall service composition function. It is usually called several times during the construction of a composite service. Figure 1 gives a representative architecture of the Service Discovery component. This component is responsible for the discovery, registration and deregistration of all services within the Pervasive Services Platform. If a user wishes to use a new device with other services within the environment, it must first be discovered and registered by this component.

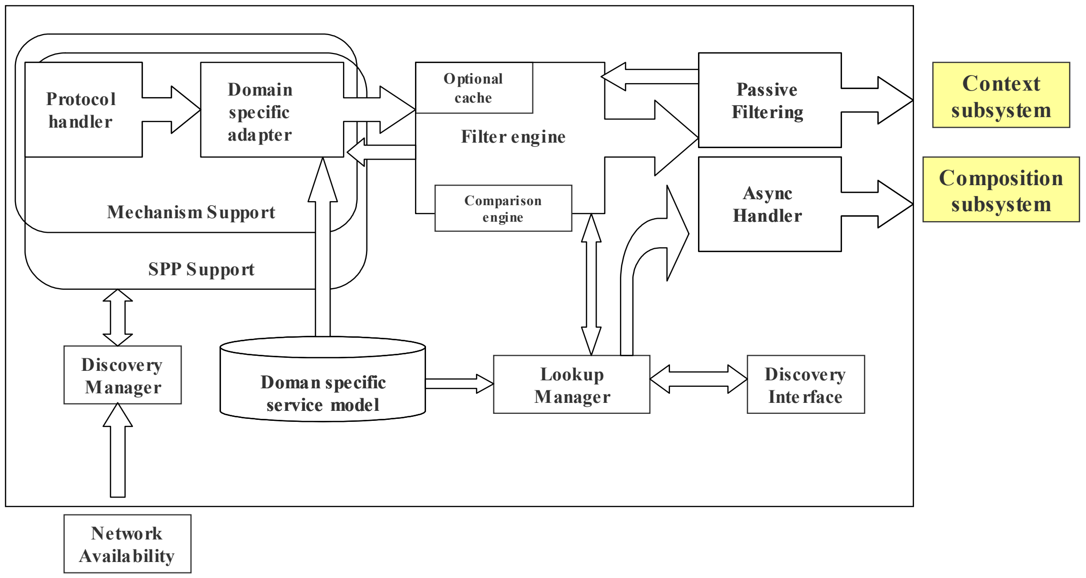

Figure 1: Service Discovery Architecture

The *discovery manager* is responsible for the instantiation of possibly multiple instances of the protocol and adapter components. These paired components act as a discovery mechanism, where one pair exists for each of the supported protocols, e.g. SLP, UPnP. The only permanent discovery mechanisms are the internal-to-the-node OSGi lookup and the one provided by the Service Provisioning Platform, the network layer below the PSP. Other discovery mechanisms are instantiated as and when needed, for example, when an ad-hoc wireless network is detected.

The *protocol handler* component encapsulates the components and/or library, which provide the specific discovery mechanism. Its purpose is to interface with the service components, using the specific protocols, and determine the existence and logical location of service components.

The primary purpose of the *domain adapter* is to hide the operation of the specific discovery mechanism. In addition, the adapter translates the information contained in the discovery messages to a standardised format. Extension is provided by interaction with the domain specific model, where future device and service models can be represented.

The *filter engine* evaluates a query against the cache of discovered information. This query is a series of expressions, ranked in order of relative importance. The filter engine may resort to issuing a request to the underlying discovery mechanism, or just use cache information. In either case, the data is then subjected to filtering. The filter engine may retain filters and combine them in chains to form more complex queries as necessary. In addition, the filter engine will allow pluggable comparison engines. These engines are registered indirectly with the filter engine, via the lookup manager interface. Possible comparison engines could include Boolean expression matchers and fuzzy logic engines.

The *lookup manager*'s primary responsibility is to answer on demand queries for the availability and location of specified services. To this end, it interacts with the filter engine, to evaluate the result set. In addition to providing on-demand results, the query manager also allows the delayed asynchronous delivery of results, through the use of passive filter component.

The *asynchronous handler* component is a handler for returning asynchronous or delayed replies. To make use of this facility the consumer provides a call back interface when specifying the query. Should a definitive reply for the query not be available when the query is made, the lookup manager registers the call back interface with the Asynchronous handler. As and when the results of the query become available the call back interface is notified.

The purpose of the *passive filter* component is to filter updates to the lookup cache. The primary purpose is to contribute to the context information given to service composition. For example, if the operation of passive filter is configured to be "sticky", then the last five filters associated with lookup queries remain active. This means alternate services are passively offered.

3.3. Composition Management

The composition manager, within the Pervasive Services Platform is responsible for managing the operation of a composite service. The composition manager also handles adaptation of the composite service based on personalisation constraints defined by the user.

3.3.1 Service Adaptation

Service adaptation must not be confused with content adaptation. Content adaptation deals specifically with tailoring the content of the service. Service Adaptation encompasses content adaptation and more. Service adaptation can be carried out both during service composition and while the service is running. In Daidalos, adaptation of the service can be triggered by changes in context for example users' location, proximity of other people or discovery of new services. The adaptation occurs as the system constantly attempts to match the users' needs and the available resources and capabilities of the service, by constantly monitoring changes in those changing needs in relation to the users' environment.

One of the scenarios in Daidalos shows how content adaptation and service adaptation might interact. This scenario involves moving a streaming session from a PDA to a wall display. This move cannot take place unless both the content and the service change. The low resolution, low bit rate content is substituted with a high-quality stream. This is adaptation is visible by the user. Other changes however, happen behind the scenes. One example might be in the way the content is delivered. Most likely a (slow) wireless channel is replaced with a high-speed Ethernet connection. As a result of improved quality the user should probably pay more for the content (and perhaps less for the use of the service because a cheaper distribution channel is used).

One powerful requirement of service adaptation is the ability to deliver the same information to multiple users using any available channel. The scenario mentioned above is an example of adaptation of a service to fit personal needs and context. There may be a wall display available to some users allowing high quality information streamed to it; others only have their mobile terminal at hand and should get the essential information delivered by SMS or MMS.

3.3.2 Composition Management Architecture

Service composition involves repeatedly looking for (discovering) services until the all the required services for the composed service have been found. When the list of services for one set of criteria has been found, they are then fed to the Service Selection module. This module then interacts with Personalisation to select the most appropriate services for the user, given their preferences and current context. These services are then fed to the composition manager module. The module will continue to make requests to the Service Discovery module until all services required for the complete composed service offering have been found. Once the service has been composed, it is the Service Actuator that looks after the composed service instantiating it as a complete service and monitoring it during its lifetime.

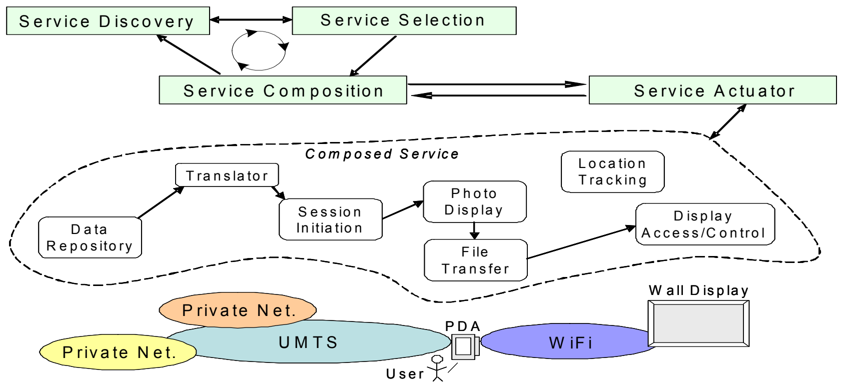

Figure 2 Process of Composition

The architecture of service composition is shown in Figure 3. This architecture shows the components involved and the interactions with outside components.

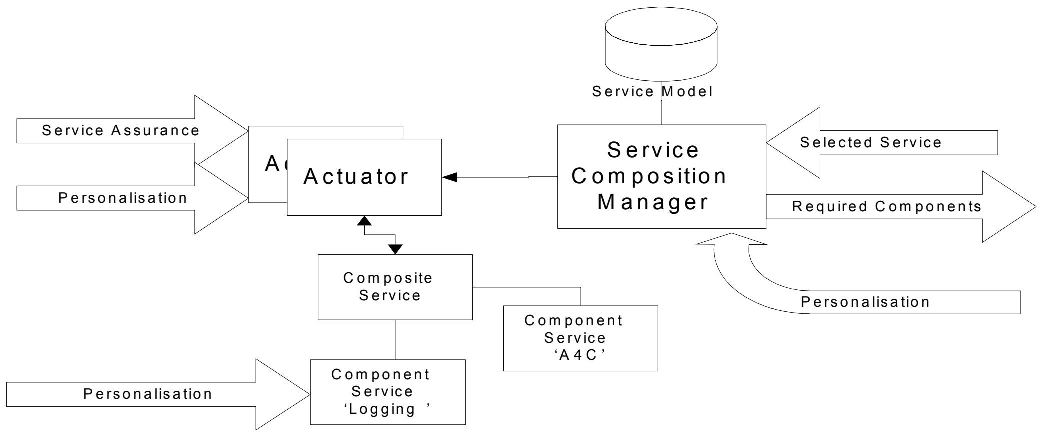

Figure 3 Composition Manager Architecture

The *service model* is the conceptual repository of composite information. It contains the detailed information for all services. This process information includes the services requirements on other services, the services interfaces it exports. It is a distributed service as the composition information may be cached from previous successful service compositions or retrieved from the services or service providers. The information retrieved using the latter methods, may be incomplete, or impose too many restrictions to be useful. Thus the caching of previous working service configurations is essential.

The *service composition manager* is responsible for performing reasoning on the selected composite service. It is given a descriptor of the selected service, and from this obtains the detailed service information. This information includes the specific requirements for component services. This information is parsed, and a most likely composition order is decided upon. The requests for component services are then issued to the service discovery and selection component grouping.

If no component services can be found then the composition manager contacts an inference engine to try to infer a component service collaboration that will yield the desired service. These interactions may require use of other composition managers, with more specific service composition knowledge. When a potential combination is found, the running order and component and composite service access information is passed to the service actuator pool. For example, initially a service giving an Irish newspaper in Japanese may be searched for. Should no results be found an English/Japanese translator service and the English version of the newspaper might be composed in a linear fashion.

The behaviour of the service composition manager may itself be adjusted through interactions with the personalisation components. The principal mechanism for customisation is the application of personalised composition strategies, which subtly adjusts the mechanism for including components.

The *Service Actuator* grouping is responsible for instantiating the instances of the service components. It adopts the behaviour of a pool manager/factory for service component instances. It is responsible for informing the composite service of the component service instances. The service composition manager provides a "running order" for the composition to take place. In addition, the service invocation details are also passed. The actuators attempt to optimise the use of new/existing service instances, by acting as a factory or pool manager depending on the situation. An example would be re-using an existing component instance, as opposed to creating a new one or creating a second application window, instead of an application instance.

The service actuator takes an active role in requesting personalisation parameters for the composite service. These parameters are used to provide initial initialisation values for the composed service and service components. Individual, service components may also be personalised through direct interaction with the personalisation components.

In addition, the service actuators receive information on the performance of the composite process. Should sufficient degradation in the performance of the service be detected by the service assurance components, the actuator asks the Service composition manager to provide alternative component parts. This role also extends to the monitoring via the Rules Engine of context conditions, which are of interest to the composed service. In particular, service specific context change events, which may result in the service actuator again asking the composition manager to re-compose part of the service.

4. Pervasive Services Scenario

This section will present a scenario to outline the process of service composition within the Pervasive Services Platform. This scenario gives an amalgamated view of how the various components work to create the end user experience.

4.1.Service scenario – Classroom usage of Government provided Language Education Service

Miss Anderson is a primary teacher. As part of her job she is equipped with a PDA. An mGovernment service 'WordADay' has been developed for primary schools to encourage the learning of foreign languages at a young age. The WordADay service presents to students a picture with associated word (in the required language), and also provides teacher specific content to aid the teaching process. The WordADay service will always display itself on the largest available display, and the teacher content will remain on the teachers PDA.

As shown in Figure 2, fulfilling this task involves the composition of software services for photo storage, translation (of image files into formats appropriate for the display device in use), UMTS communications session initiation, PDA content display, file transfer, location tracking and wall display access/control. In the use case scenario the service desires that word and photo are displayed on the largest display device in their physical vicinity. The service composer initially identifies the types of services required to fulfil this requirement; it then requests the service discovery component to provide it with a list of candidate services; a subset of the candidate services are then composed based on optimising against personalisation parameters. Execution of the composed service is performed by the service actuator, which uses information from the location tracking service, and service discovery component, to find the new display and trigger the switch between displays.

It is assumed that the service composition manager is aware of the service interfaces for both a 'WordADay' service and also a transformation service that will be used to convert and retrieve the pictures requested by the user. Miss Anderson has been continually using this service over the past few months and as such the system has inferred certain preferences and usage patterns.

- Initially Miss Anderson started the service herself when she entered the room (classroom). The system inferred from usage patterns that at this time of day (morning) in this location (classroom) the service should be automatically started;
- Miss Anderson historically stopped the service at lunchtime, so the system again infers that the service is being stopped at this time of day. The system can then ask her if this service is to be stopped every lunchtime. If she refuses, the system can offer to remind Miss Anderson five minutes before lunchtime;
- The default language for the service initially was French. The first time Miss Anderson used the service she selected to use the German language. This is the automatic language for the service now, unless reconfigured by Miss Anderson.

The scene begins as Miss Anderson enters the classroom.

1. Miss Anderson walks into the classroom. Her current context, in-the-classroom-at-9am, triggers a rule that starts the 'WordADay' service. Note that usually a service such as the 'WordADay' would have been cached after the first use, so there would be no need for retrieval from the network, but for the purposes of illustration let us assume it does need to be retrieved;
2. The trigger causes the search for the 'WordADay' service;

3. There is only one service of this type, so there is no service selection required;
4. The service is passed to service composition where the service model of the ‘WordADay’ service is assessed. The service model of the ‘WordADay’ service infers it should be composed with a Wall Display service;
5. Service discovery again searches for Wall Display services;
6. Two are found, the large display at the front of the class and the small one on Miss Andersons desk that she uses in conjunction with other services from her PDA;
7. These two services are passed to Service Selection. Service selection (using the personalisation component) infers from the personalisation parameters associated with the ‘WordADay’ service, that the larger display should be used;
8. The selected display service is passed to the service composition component. The selected display service has no inferred dependencies in its service model, so the service composition is complete;
9. The service is now passed to the service actuator and automatically started by the service actuator (again owing to the personalisation component);
10. Once started the Word and Picture are automatically displayed on the Large Display and the teacher notes remain on her PDA;
11. When it reaches lunchtime context ‘lunch-time’ triggers the service to be stopped. The service actuator stops the ‘WordADay’ service, and all associated services that have been composed with it including the display service.

This scenario should give the reader a realistic idea of how services are to exist and be used in the pervasive Daidalos world. The system will continue to learn about the usage patterns of the service and, the behaviour of Miss Anderson and will continue to monitor context parameters belong to her and create triggers for the usage of the services in the environment around her.

5. Discussions and Future Work

This document has attempted to demonstrate how service composition in conjunction with personalisation and context awareness can simplify the user experience of a pervasive environment in the face of the availability of an exponentially increasing number of services. The paper begins by giving a background to the area of service composition and the challenges faces when thinking of service composition. It then introduces the Daidalos project and describes the details of the composition component of the Pervasive Service Platform as defined in the Daidalos project. The paper continues on by giving a scenario where service composition is used and describes the steps involved. It should be clear having read the paper that the methods used for service discovery and composition allow:

- A wide coverage of discoverable services by using a range of discovery mechanisms;
- Limitation and often elimination of the work the user must do to retrieve services that are relevant to them;
- A better service offering and more service re-usage by composing services;
- Continuous adaptation of services to the users needs as they move through the service environment.

In the Daidalos envisaged world of services, it is relatively easy for new services to be deployed, but even more importantly easy for those services to be used. In a world that is becoming increasingly technically driven, it is important that this focus does not leave sections of society behind. Technology is only as powerful as the people who use it. In a Daidalos world there is no segregation since services become invisible only ‘poppingup' when needed. By making use of a close coupling between discovery and composition, personalisation and context awareness, Daidalos has established a mechanism to create a truly user-centric environment.

The functionality described in this paper has been partially implemented and tested on a laboratory scale. Most of the components described in Service Discovery Manager, and Composition Manager are implemented and work on a small scale. This prototype was built on an OSGi Service Platform. SOAP was used for remote communication, while the discovery of services and components was based on the ontology-enhanced version of the SLP protocol. In December 2004, the first prototype implementation of the Daidalos Pervasive Services Platform was finalized and successfully demonstrated and evaluated via a set of illustrative real life scenarios similar as the one in this paper.

Future work on this platform includes, validation of the platform against similar systems, from the perspective of personalised service composition and service adaptation. It is also hoped to extend the adaptation process within the platform to cater optimise service performance in response to changing context conditions by integrating adaptation methods such as the ones defined in section 2.4.

6. References

Araniti (2003). Adaptively Controlling the QoS of Multimedia Wireless Applications Through User Profiling Techniques, IEEE Journal on Selected Areas in Communications. 21.

Daidalos. (2005). " Daidalos EU Framework Programme 6 Integrated Project." Retrieved 24/02/2005, from www.ist-daidalos.org.

Davy, A. (2004). "Task Driven Service Composition for Pervasive Computing Environments." M-Zones White Paper Retrieved June 2004, from www.m-zones.org.

Filman (2001). Aspect-Oriented Programming is Quantification and Obliviousness, In Proceedings of the ECOOP 2001 Workshop on Advanced Separation of Concerns, Budapest.

Hirschfeld, R., Kawamura, K. (2004). Dynamic Service Adaptation. 4th International Workshop on Distributed Auto-adaptive and Reconfigurable Systems, Tokyo, Japan, IEEE Computer Society.

Huang, A.-C., Steenkiste, P. (2004a). Building Self-configuring Services Using Service-specific Knowledge. Honolulu, Hawaii USA, The Thirteenth IEEE International Symposium on High-Performance Distributed Computing.

Meyer J.A, A. G. (1991). Simulation of adaptive behaviour in animals: Review and prospect. From animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behaviour, Cambridge, MA, MIT Press / Bradford Books.

The work described in this paper is partially based on results of IST FP6 Integrated Project DAIDALOS. DAIDALOS receives research funding from the European Community's Sixth Framework Programme. Apart from this, the European Commission has no responsibility for the content of this paper. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

Alan Davy was awarded a BSc (Hons) in Applied Computing from Waterford Institute of technology (WIT) in 2003. Since September 2003 he has been a PhD student with the Telecommunications Software & Systems Group (TSSG) in WIT. He is currently funded by the PTRLI strand 3 M-Zones programme. This programme addresses issues relating to the design, configuration and management of "smart space" ubiquitous computing environments. His research interests are in the area of dynamic service composition, real-time service adaptation, and performance analysis. He has published two peer-reviewed publication in national and international conference proceedings, and two non peer-reviewed white papers.

Fiona Mahon was awarded a first class BSc (Hons) in Computer Science from University College Dublin in 2000. She spent one year in Ericsson's R&D centre in Athlone working on aspects of Ericsson's 3G network. She spent a further 2 years involved in investigating and developing cutting edge customer experience management for GPRS and GSM networks in Aran Technologies. In 2004 Fiona joined the Telecommunications Software & Systems Group (TSSG) in WIT, where she primarily works on the IST FP6 project Daidalos. This project aims to create a fully deployable pervasive network integrating personalisation, context awareness, service discovery and composition, security and privacy and rules management. Her main area of research is in service discovery and composition.

Kevin Doolin has the position of Competence Centre Head of Pervasive Communications Systems in the TSSG; this includes responsibility for the TSSG's contributions to large EU FP6 projects such as Daidalos. He has a First Class Honours BTech in Electronic Engineering. He spent the last 18 months as a project executive with Ireland's Investment and Development Agency (http://www.idaireland.com) based in Waterford. His focus has been on R&D development initiatives with foreign multinational companies that have established (and/or are planning to establish) a presence in Ireland. The role included building a detailed awareness of Europe's strategic agenda in Information and Communications Technology, and supporting companies interested in harnessing potential funding opportunities in this space. Before this, he has spent 8 years with Ericsson Systems Expertise ending as a Strategic Product Manager. He was responsible for the specification and development of the operation and maintenance solution for Ericsson's 3G Telecoms and IP Multimedia offerings (e.g.http://www1.ericsson.com/products/hp/Multimedia_pa.shtml). This role included running an international project across multiple Ericsson sites, with a view towards specifying the high-level requirements and systems specification for the O&M solution to be provided. He has also acted as International Project Manager, Software Architect, Systems Designer, and International Team Leader.

Brendan Jennings was awarded a PhD in Telecommunications from Dublin City University in 2001 and a BEng in Electronic Engineering from Dublin City University in 1993. Since June 2003 he has been a Senior Investigator with the Telecommunications Software & Systems Group (TSSG) in Waterford Institute of Technology. He is a lead researcher in the HEA-funded M-Zones programme, addressing issues relating to the design, configuration and management of “smart space” ubiquitous computing environments. His research interests are in the areas of accounting systems for dynamically composed services; user-centric, distributed policy-based management systems; performance management; and agent technology. He has published fifteen peer-reviewed publications in international journals and conference proceedings, two book chapters and has authored contributions to standards bodies ETSI and FIPA.

Mícheál Ó Foghlú is the Research Director of the Telecommunications Software & Systems Group in Waterford Institute of Technology. This group, founded in 1996, has engaged in over 40 basic and applied research projects, winning nearly 20 Million EUR of funding. The group now has 56 staff and students and acts as a regional catalyst for ICT innovation in the South East. Mícheál’s research interests centre on the use of next generation networks to deliver flexible services. This includes an interest in IPv6, IPv6 QoS and Security, mobile IPv6 and web-based technologies to build flexible services running over IP networks.