

Task Driven Service Composition for Pervasive computing environments

Alan Davy
June 2004
adavy@tssg.org

Introduction

Pervasive computing environments, such as smart spaces, offer a wide variety of heterogeneous services to users. These environments are becoming more and more service oriented and modular. With the availability of so many services to perform diverse individual tasks within the smart space, users find themselves having to design custom-built applications to perform certain tasks, or invoke each service individually to achieve required tasks. This is a time consuming activity and usually means that setting up and becoming familiar with the environment takes longer than performing the intended tasks. This also means the user must have a reasonably in-depth knowledge of how to configure services based on his/her requirements.

This situation will not be acceptable in the future. The fact that the environment is pervasive should also mean that the services it provides are pervasive. Services are being designed to interact with each other, not just with the end users. The user must be able to take advantage of this, without being aware of how these services interact. Task-driven computing (Zhenyu Wang & David Garlan 2000) aims to translate the users computing intention i.e. task, into a collection of dynamically organised services interacting to satisfy the requirements of the users task.

In order for a system to be able to translate user intent, or a task definition into a composition of services, the services must first have well defined interfaces and more importantly they must be accurately described not only by functional but non-functional information such as the semantic of the service (Justin O'Sullivan, David Edmond, & Arthur Ter Hofstede). In the Web Service arena, semantic web languages, such as the Web Ontology Language (OWL)(Dean et al. 2002) or DAML+OIL (Horrocks et al. 2001), provide the foundations for such sufficiently rich descriptions.

The aim of this research work is to develop a process for the composition and management of services within a smart space environment. The following sections outline current technologies that are being employed in this area, an overview of what tasks are and how they may be represented to a system, an overview of what service semantics are and how they are required for service composition, and a section on the process and requirements of service composition for pervasive computing environments. Finally the paper outlines a Policy management system for service, in use at WIT for the Mzones program, and how service composition will be integrated into this platform.

Current Technologies

At present most attention in the area of service composition is in Web Service composition. In May 2001, the DARPA Agent Mark-up Language (DAML) Program released the first version of DAML-S (Ankolekar et al. 2002), a set of ontologies for describing the properties and capabilities of Web Services. The purpose of DAML-S mark-up of Web Services is to support effective automation of various Web Services related activities including service discovery, composition, execution, and monitoring. (Dan Wu 2003).

It is a safe assumption that pervasive services within the smart space environment will be based on the Web Services paradigm, and that these services will be described with an appropriate set of ontologies similar to, DAML-S. What is required is an appropriate lightweight task definition and service composition model for pervasive computing environments.

Task Representation

If the smart space is to compose service based on the tasks a user wishes to complete, then there must be a way of letting the user define his/her task in a way understandable to the smart space, such as a task specification language. A task specification language must provide the ability to express flows and primitives. Task flows decompose a complex task into a sequence of steps (sub tasks or primitives). A task primitive is a unit of action that has to be carried out by the actual service. (Zhenyu Wang & David Garlan 2000) talks about task management and the need for lightweight task creation. This is very important for pervasive computing environments, because of possible lack in computational power. One way of representing tasks is to use the DAML-S process ontology to encode a description of how to compose Web Services for tasks. In the DAML-S process ontology, services are modelled as processes. There are three kinds of processes: atomic processes, composite processes and simple processes. Atomic processes are single web services, which are executed by passing them particular input values (if required). Composite processes are executed compound Web Services that can be de-composed into atomic Web Services, or further composite Web Services.

Service Semantics

Each service definition has several properties, including (optional) inputs, preconditions, (conditional) outputs and (conditional) effects. These are further described by (Dan Wu 2003). A major part of service composition is describing the non-functional attributes of services. The non-functional properties of services include temporal and spatial availability, channels, charging styles, settlement models, settlement contracts, payment, service quality, security, trust and ownership. To give an example if a user want to perform a task, which involved printing a document, s/he would rather it be printed to the most local printer, hence the printer service would need to include non functional information relating to its spatial availability. (Justin O'Sullivan, David Edmond, & Arthur Ter Hofstede 2002) provide an in depth discussion on the description of non-functional service properties.

Service Composition

For a composed process to be managed within a system, the corresponding composed service request must be represented in an appropriate way to the system. The system must be able to represent the current state of the environment, compose a service automatically, verify the execution of the composed service within the environment and monitor its process through out its execution path. One way of representing this is by using the situation calculus. Situation¹ calculus (Hector Levesque 1998) is a first order logical language for representing dynamically changing worlds in which all of the changes are in direct result of an action performed, in this case the execution of the composed service. Since DAML-S is actually a process modelling language, (Srin Narayanan & Sheila A.McIlraith 2002) have introduced a method of mapping DAML-S to the situation calculus. With this highly expressive language they go on to explain how services, (termed actions) can effect the situation i.e. the smart space. With this technique they can model the effects of invoking an action on the situation, perform simulations, automatically compose a service, and validate a composed service.

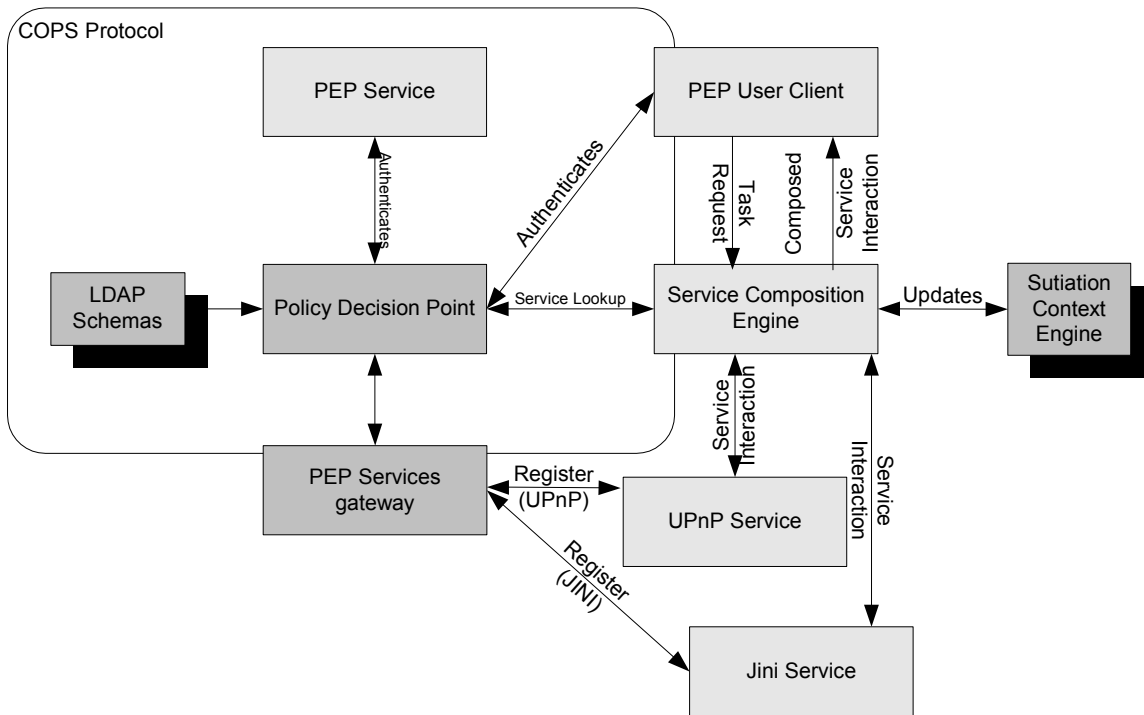
Policy Management System

The following section is an overview of the planned test bed to be developed for Mzones by TSSG. This will support policy management of services, and users. It will provide service discovery, and service invocation based on user and service policies. With the support of this testbed, service composition and service composition management can be employed and tested.

The Policy Management System (PMS) is used to control access to services within a smart space environment. Groups can be set up with policies of access to services. User accounts can be created. Policies can be applied to individual users, with regards access to services, or users can be added to groups, thus inheriting its policies. The PMS uses the COPS protocol (2004) to negotiate policies between the Policy Enforcement Point (PEP) and the Policy Decision Point (PDP). The system uses COPS-Service Discovery Protocol (COPS-SD) to discovery services within the smart space. The system can also use a gateway to other service discovery protocols such as UPnP and JINI. A User must be logged in and authenticated before s/he can use a service. Services must also be authenticated before they can be offered to users. The system follows the Common Information Model (CIM) to represent users, services and profiles within the system.

¹ In Situation Calculus a situation is a sequence of actions, evolving from an initial distinguished situation. For further information see (Hector Levesque 1998).

This platform will be extended to integrate a service composition engine, where services available in the environment will be discovered through UPnP / JINI. Service requests / user task definitions will be passed to the service composition engine, composed service will be validated against the user's profile and access rights. Automated execution of composed service will be performed as service is being executed.



References

The COPS (Common Open Policy Service) Protocol. 2004.

Ankolekar, A., Burstein, M., Hobbs, J. R., Lassila, O., Martin, D., McDermott, D., McIlraith, S. A., Naryanan, S., Paolucci, M., Payne, T., & Sycara, K. The DAML Services Coalition. DAML-S: Web Services Description for the semantic Web. Proceedings of the First International Semantic Web Conference. 2002.

Dan Wu. Automating DAML-S Web services Composition using SHOP2. Bijan Parsia, Evern Sirin, James Hendler, and Dana Nau. 2003. In Proceedings of 2nd International Semantic Web Conference (ISWC2003).

Dean, M., Connolly, D., van Hermeln, F., Hendler, J., Horrocks, I., McGuinness, D. L., & Patel-Schneider, P. F. S. L. A. Web Ontology Language (OWL) Reference Version 1.0. Recent Trends and Developments. 2002.

Hector Levesque. Foundations for the Situation calculus. Fiora Pirri and Ray Reiter. 3. 1998. Linkoping electronic Articles in Computer and Information Science, Linkoping University Electronic Press.

Horrocks, I., van Hermeln, F., Patel-Schneider, P., Berners-Lee, T., Brickley, D., Connolly, D., Dean, M., Decker, S., Fensel, D., Hayes, P., Heflin, J., Hendler, J., Lassila, O., McGuinness, D. L., & Stein, L. DAML+OIL. 2001.

Justin O'Sullivan, David Edmond, & Arthur Ter Hofstede. What's In a Service? Towards Accurate Description of Non-Functional Service Properties. Distributed and Parallel Databases, 117-133. Kluwer Academic Publishers.

Justin O'Sullivan, David Edmond, & Arthur Ter Hofstede. What's In a Service? Towards Accurate Description of Non-Functional Service Properties. Distributed and Parallel Databases, 117-133. 2002. Kluwer Academic Publishers.

Srini Narayanan & Sheila A. McIlraith. Simulation, verification and Automated Composition of Web Services. 7-11-2002. ACM Honolulu, Hawaii, USA.

Zhenyu Wang & David Garlan. Task-Driven Computing. 2000.