

Software Infrastructure

Though M-Zones will be conducting research into adaptive and policy-based techniques for performing adaptive smart space management, this can only achieve results if supported by a clear view of common software services and capabilities that will be available for application developers. We call this set of common software services, technologies and capabilities the software infrastructure. In any particular domain of computing, the software infrastructure is constantly developing, with new functionality being added, often at higher levels of abstraction than before. A good indication that some piece of technology is accepted widely enough to become part of the ‘infrastructure’ is when associated interoperability standard emerges. Sometimes the emergence of such standard is the catalyst for widespread uptake and ‘infrastructuralisation’ of a technology. This section reviews aspects of the state of the art in the software infrastructure that appears to be in place for ubiquitous computing. M-Zones moves forward with the assumption that such an infrastructure will form the basis of our research into adaptive smart space management, but also with the expectation that technologies that are at the research and development stage now will become part of the software infrastructure as the project progresses. Equally, improvement of the infrastructure will be an active research area in M-Zones.

This section addresses software infrastructure through three state of the art whitepapers:

- Context Management by Keara Barrett (WIT) and Ruaidhri Power (TCD)
- Service Composition by Steffen Higns and David Lewis (TCD)
- Middleware by Ray Carroll (WIT), Sinead Cummins (CIT), Fergus O’Reilly (CIT) and Jason Finnegan (WIT)

Context information is important both to enable ubiquitous computing systems to adapt to a changing physical and computing environment and to support them in understanding changing user needs and the best corresponding course of action to take in any particular situation. Context information includes computing context, the user context and the physical context as well as historical information about these. As any form of adaptive smart space management system will need access to context information, inclusion of context management capabilities into the infrastructure is a sound architectural decision. Context management is already in place in a rudimentary sense, for example in the way desktop office application use information on past and current activities to provide context sensitive help or in the way web browsers and server collaborate to detect if someone has visited a web page before. Much work to date in relation to ubiquitous computing has focussed on location detection as a primary source of context information, however the Context Management white paper points out that a much broader view of context information is required if context management is to be offered as an infrastructural service. This will require standardisation of context information through adoption of common models and service for manipulating context information, rather than the wide range of highly application dependent models and service currently available. This will require some commonality in how context data is collected from sensors, applications and other parts of the infrastructure, but also strong privacy control mechanisms to provide people with the confidence to allow personal context information to be shared with encountered applications and ubicomp environments.

A core assumption that is essential to developing adaptive system that can dynamically integrate components from multiple developers and operated in multiple organisational domains, is that components only interact through well defined interfaces. This is known as a service-oriented approach, where such well defined interfaces, or services, are reflective, i.e. their capabilities can be examined at run-time. The level of expressiveness and semantics used in a reflective service

description determines the level of automated decision making that can be performed by an adaptive system. The assembly of services into a more highly functional aggregate service is called service composition, and a principle aim of adaptive smart space management of to perform service composition is as dynamic and seamless a manner as possible. The Service Composition white paper discusses the various activities that are involved in service composition, including the description of services at various levels of abstraction, the advertisement of services and the use of brokers to locate services that best fit requirements and trust requirements for using foreign services. The automated composition of services is still a very open research area and centres on the combination of various reasoning techniques with common mechanisms for expressing semantics via ontologies. Standardisation of ontology language is underway in ISO (Topic Maps) and W2C (based on DAML+OIL). Building on these DAML-S utilises concepts from workflow and web service flow languages to provide a semantic service description and composition language.

Underlying application development in any distributed computing environment is the concept of middleware, which aims to provide the application programmer with a portable API and a set of common services. Middleware systems and standards are very mature, with CORBA, Sun's Java RMI and Microsoft's .NET being the most widely used. A wide range of general purpose and application specific services are available for these platforms and interoperation solutions between platforms are widely available. In addition, most of these platforms now take a component-oriented approach to application to ensure ease of portability and deployment on different platforms. Middleware system also typically support exposing interfaces as web services so that these services can be easily accessible over the Internet (most middleware platforms cannot negotiate unmodified firewalls). Despite this maturity the middleware area is developing fast along a number of axes described in the Middleware whitepaper, which may be useful for adaptive smart space management. One specific example is Jini, which is Java based middleware aimed at ad hoc communication between a wide range of personal, office and domestic devices. Another broader movement is that of intelligent mobile agents. These primarily take advantage of the code mobility offered by Java to allow active software components, called agents, to move around a network in order to accomplish some task. As agents will have no a priori knowledge of other agents they will encounter and collaborate with in accomplishing tasks, there is quite an amount of research on agent communication mechanisms where capabilities are exchanged and mutual activities negotiated. This makes agent very applicable to the highly dynamic computing environments of smart spaces. Finally there are some standardised application specific APIs that may be relevant to the integration of smart spaces with other system. Parlay is a good example, being an industry API intended to make management and control functionality of telephony systems available to other systems.