J Ambient Intell Human Comput (2010) 1:15–29
DOI 10.1007/s12652-009-0003-5

ORIGINAL RESEARCH

Towards evolutionary ambient assisted living systems

Michael J. O'Grady · Conor Muldoon · Mauro Dragone · Richard Tynan · Gregory M. P. O'Hare

Received: 4 July 2009 / Accepted: 21 October 2009 / Published online: 2 December 2009

Abstract Ambient assisted living (AAL) is advocated as technological solutions that will enable the elderly population maintain their independence for a longer time than would otherwise be the case. Though the facts motivating the need for AAL are indisputable, the inherently heterogeneous nature and requirements of the elderly population raise significant difficulties. One particular challenge is that of designing AAL systems that can evolve to meet the requirements of individuals as their needs and circumstances change. This demands the availability of an adaptive, open, scalable software platform that incorporates a select combination of autonomic and intelligent techniques. Given that the first generation of AAL systems will be deployed in the near future, it is incumbent on designers to factor this need for evolution and adaptivity in their designs and implementations. Thus this paper explores AAL from a number of prospective and considers an agent-based middleware approach to realising an architecture for evolutionary AAL.

M. J. O'Grady (✉) · C. Muldoon · M. Dragone · R. Tynan · G. M. P. O'Hare
Centre for Sensor Web Technologies, School of Computer Science and Informatics, University College Dublin, Belfield, Dublin 4, Ireland
e-mail: michael.j.ogrady@ucd.ie
URL: http://www.csi.ucd.ie/

C. Muldoon
e-mail: conor.muldoon@ucd.ie

M. Dragone
e-mail: mauro.dragone@ucd.ie

R. Tynan
e-mail: richard.tynan@ucd.ie

G. M. P. O'Hare
e-mail: gregory.ohare@ucd.ie

Keywords Ambient assisted living (AAL) · Pervasive health · Middleware · Ambient intelligence · Multi-agent systems

1 Introduction

A major demographic shift is ongoing in most developed countries resulting in an increased proportion of their respective populations being middle aged and over. Over time, this will result in a larger older population being supported by a smaller active working population. This emerging scenario is well documented, and it raises profound questions about how society over the next 50 years will evolve. In practice, this demographic shift will have serious ramifications for all elements of governmental policy but particularly health and pension planning, and it can be safely assumed that others will emerge over time. In the case of the European Union (EU), it is envisaged that the population will be the same in 2060 as it is today, 500 million, even when birth rates, life expectancy and migration factors are considered (EC 2009). However, 30% of the population will be over 65 as distinct from 17% at present. In essence the ratio would move from 4 people of working age for every person over 65 to a ratio of 2 to 1.

Ambient assisted living (AAL) (Steg et al. 2006) was conceived as one strategy for addressing the difficulties that this forthcoming demographic shift will give rise to. It proposes the select harnessing of Information & Communications Technologies (ICTs) to deliver innovative applications and services that would enable the elderly to live independently for longer, and reduce the need for long-term care. Independence is closely associated with health; the World Health Organisation (WHO) regards health as a combination of physical, mental and social wellbeing, as distinct from the mere absence of disease (Callahan 1973). Thus it behold AAL to contribute to the maintenance of each. Moreover, it has been demonstrated that greater independence, implying fewer care needs, correlates with better housing conditions (Schaie et al. 2003). It is here that a key justification for proceeding with AAL, rather than the mere financial, may be found.

2 Ambient assisted living

Ambient assisted living is multifaceted (Fig. 1) and is envisaged as supporting the person in their home, communities and work places (AALIANCE 2009) Promoting social interaction is key objective of AmI as, unfortunately, isolation, social exclusion and depression are frequent results of the aging process in practice. A second key objective concerns the maintenance of the person's well-being in their own home, ensuring that the person is safe and capable of functioning independently. This later scenario has attracted significant research effort, possibly due a perception that maintaining independence and well-being in the home is essential and that all other benefits will accrue from this.

2.1 AAL: technological approaches

AAL is multi-disciplinary and harnesses a diverse range of technologies from various domains. In the case of AAL in the home, smart environments (Cook and Das 2007) and Ambient Intelligence (AmI) (Cook et al. 2009; Vasilakos and Pedrycz 2006) are perceived as two key enabling technologies.

A range of issues relating to AAL have been described in the literature. These range from stand-alone application-specific systems to fully instrumented environments in which real people live and work. In the former case, a number of efforts at detecting falls have been described (Bourke et al. 2007; Sixsmith and Johnson 2004) as the consequences may be fatal. The issue of monitoring various biometric parameters, such as ECG and blood pressure for example, has been investigated (Kang et al. 2006; O'Flynn et al., 2006). In the case of diabetes, Preuveneers and Berbers (2008) have developed a mobile phone based application that assists with the identification of the correct daily dosage levels for the maintenance of stable blood glucose levels. Though these systems are essentially stand-alone, it can be envisaged that they may be integrated into AAL enabled environments, a number of which have been described in the literature.

EasyLiving (Brumitt et al. 2000) represents an initial effort at harnessing a diverse range of technologies to deliver a coherent user experience. Carelab (de Ruyter and Pelgrim 2007), developed by Philips in the Netherlands, is structured as a one bedroom apartment that incorporates a sensor infrastructure that is used to construct behaviour patterns and identify activities. An AAL laboratory has been constructed by Fraunhofer that is used primarily for usability and accessibility tests with elderly people (Kleinberger et al. 2007). AwareHome (Kientz et al. 2008), I-Living (Wang et al. 2006) and BelAmI (Anastasopoulos et al. 2005) are likewise concerned with various aspects of smart environments and assisted living. The Gator Tech House project involves the complete instrumentation of an apartment with a range of smart objects, all coordinated by an OSGi server (Helal et al. 2005). Though many prototype environments have been developed, detailed descriptions of those that in which real people are living are rare. One such example, however, is the AAL project at the University of Kaiserslautern in which AAL systems have been deployed in a block of 20 flats with the inhabitants aged between 45 and 85 (Floeck and Litz 2008).

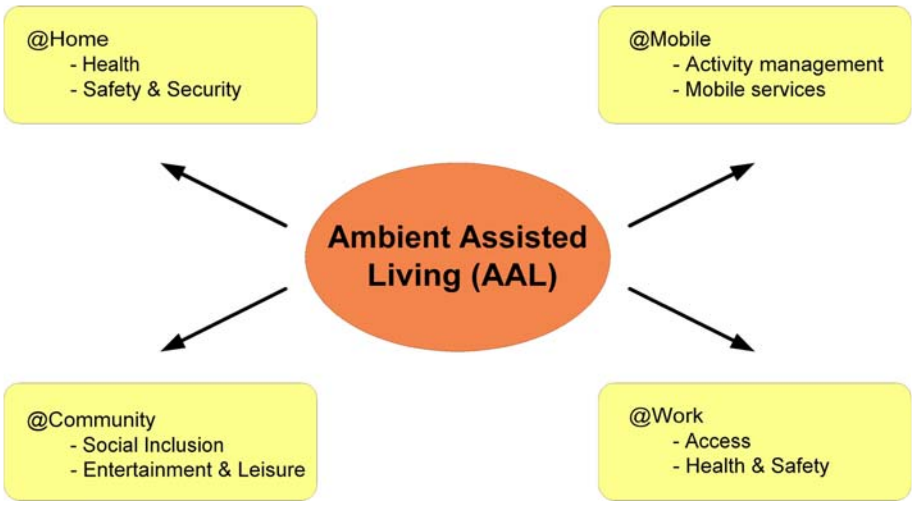

Fig. 1 Key facets of the ambient assisted living concept

Though deployments of AAL systems can be expected to increase in the coming years, a number of questions remain about how best to construct such systems from a software engineering perspective, what tools should be used and what characteristics the core software infrastructure itself should exhibit, irrespective of what services are build on top of it. A number of approaches have been described. For example, Segarra and André (2009) consider context-awareness as being a key feature. Adaptivity is also considered important, a view shared by Giner et al. (2009). The use of artificial intelligence (AI) techniques is one that has attracted significant attention as they enabling inferencing and reasoning with noisy and incomplete data. The realisation that dealing with uncertainty is an essential characteristic of smart environments motivated Dimitrov et al. (2007) to propose a probabilistic reasoning framework for use in smart homes. Likewise, Pollack (2005) envisages AI as being a key enabler of technologies for those with cognitive impairments.

2.2 AAL: who are the stakeholders?

Ultimately, society has most to gain by ensuring the welfare of those in its midst, and particularly the most vulnerable. According to Ghandi and Truman, the way the elderly (amongst others) are treated indicates the greatness and degree of civilisation of a society. Within society, AAL assumes a greater importance for some rather than for others. The key stakeholders are as follows.

- *The elderly* Both individually, and as a collective, the elderly have the greatest stake in the success of the AAL vision.
- *The family* The responsibility for caring for the elderly frequently falls upon their immediate family; thus the family has a singular interest in AAL and how those charged to their care adapt to it. It is important to note that while the independence of the elderly is usually cited as the key driver of AAL, an implicit result of its successful adoption is the independence of the family to continue their own lives. Thus, AAL has a dual role in contributing to the independence of both elderly individuals and their families.
- *External carers and heath professionals* For many of the elderly, the use of carers is inevitable at some stage. Many of the benefits that accrue to families apply here with the possible exception of carer independence. The use of AAL will enable carers plan the use of their often scarce resources in a more effective manner.
- *Governments* Full time health care either in the home or an institution comes with an expensive price tag. Thus governments, from a budgetary perspective, have a vested interest in ensuring the successful adoption of AAL technologies.

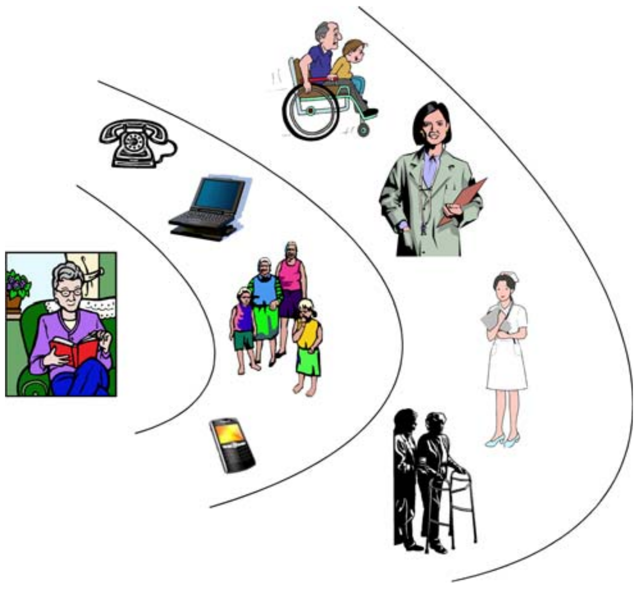

Fig. 2 Over time, an increasing number people and technologies will be essential for the effective care of the elderly

As people grow older, the range of people and technologies involved in their care will increase (Fig. 2), thus increasing the diversity of those organisations with an interest in AAL.

2.3 AAL: the nature of the elderly population

The elderly represent a microcosm of the general population. Their chronological age may differ by decades, their health will vary significantly as will their essential abilities. Indeed, the variations in abilities and needs for a given age group is much wider for the elderly than at any other stage in the human life cycle (Fugger et al. 2007). This variation poses significant problems engineering AAL hardware artefacts and software services to support them. Such solutions may well be inherently complex; however, it is essential that this complexity be transparent to the elderly as their perception of complexity is a key barrier to their acceptance of technologies (Vastenburg et al. 2008). Interestingly, the elderly are not hostile to technology per se, but they can be inclined to think its use is beyond their capabilities (Giuliani et al. 2005). However, a realisation of the need for assistance as well as a perception of "product quality" contributes to the acceptance of technology (McCreadie and Tinker 2005).

For prospective designers of AAL, there are two key categories of the elderly who may have similar needs yet who may require radically different solutions. These are those who are either elderly now or are approaching retirement, and those will be retiring in the next decade. In the former case, exposure to technologies may be negligible, and a reluctance to adapt more pronounced. In the later case, it can be expected that the population will be more at ease with technology. However, their needs may be radically different. Those who are elderly need systems that work out-of-the-box. Those who will be retiring in the next decade must be supported with systems that learn and adapt over time. It is for such a population that the use of evolutionary AAL systems is envisaged, and it is at the current juncture that key decisions will be made that may ultimately determine how successful or otherwise this endeavour will be.

2.4 AAL: potential environments for deployments

AAL is predominantly aimed at home environments. Though the nature of homes may vary considerably, there are really only two categories that are of importance: those with an integrated AAL infrastructure and those without such an infrastructure. As AAL systems are currently restricted to laboratories and some other prototype deployments, homes with an integrated AAL infrastructure are non-existent. However, it is a realistic expectation that as AAL matures, guidelines on how to incorporate AAL infrastructure into new buildings will be developed, and may be even incorporated in national legislation. This will really reflect a pattern that has been ongoing from the last century. Once electricity reached a critical mass, it was incorporated into house designs. Later, infrastructures for communications were incorporated into new house builds. More recently, home networking for infotainment has become standard in many new housing developments. Thus, it is a realistic expectation that a dedicated infrastructure for AAL with become an integral part of the next generation of housing. However, for many people, the option of acquiring a new AAL equipped home will not be feasible for a variety of reasons.

In practice, AAL systems will frequently need to be deployed in pre-existing houses. This may represent a constraint on the architecture adopted and the services that can be deployed as the nature of the environment may be militate against an optimal solution. A more subtle problem may involve the elder person's attitude, particularly if they perceive the technology as being intrusive or disrupting their home environment or routine. In the case of homes that demand the deployment of a dedicated AAL infrastructure, the potential for such problems increases.

2.5 AAL: where in the home?

A question that arises is where in the home should AAL technologies be deployed so as to be harnessed to their maximum benefit. In 2002, it was reported that one death occurs in the home every 16 min (outside of natural causes) and one disabling injury every 4 s (Home Safety Council 2002). Thus, the home is a potentially dangerous place, and the danger is accentuated in the case of the elderly. Though accidents can and do occur anywhere within the home, there is a consensus that the kitchen and the bathroom represent those places where accidents are most likely to happen. A study conducted by Leonardi et al. (2009) indicates that the elderly are conscious of the dangers in each of these locations and would be tolerant of AAL technologies that contribute to their safety there. The bedroom is more problematic and represents the room where reluctance to accept change may be greatest. This may reflect a tendency in older people to be more concerned with privacy of space rather than information (Kwasny et al. 2008). Given that a significant amount of time is spend in the bedroom, frequently an unhealthy environment (Hasselaar and van Ginkel 2004), this is problematic and needs to be approached with tact and consideration.

2.6 The need for evolutionary AAL systems

The scenarios that AAL is proposed to address are complex. A key source of this complexity is the inherent heterogeneity of the end-user population, their housing arrangements and their individual situations. Thus, complete off-the-shelf solutions for AAL that successfully addresses such disparate requirements are unlikely to materialise. Rather, customised solutions for individual circumstances must be constructed if maximum gain is to be attained. This immediately suggests AAL systems may come at a premium price; should this occur, the adoption of AAL will be poor as a low cost is essential for their widespread take-up. The following costs may be incurred:

- *Installation* It is probable that, for insurance purposes, trained personnel will be required for installation, testing and certification. Undoubtedly, DIY AAL systems can and will be deployed also.
- *Hardware* For the average installation, very little hardware may need to be acquired. Indeed, the cost of many common PIR motion sensors is only a few euro at present, so hardware will not contribute significantly to the total cost.
- *Software* The software component for AAL may be quite sophisticated and complex, resulting in it contributing significantly to the cost. However, it may be that a successful open source initiative is launched, in which case the cost would be minimal.
- *Maintenance* The status of AAL systems can be monitored remotely using conventional technologies so this will not be a significant cost. Indeed, simple tasks like battery replacement may be undertaken by the elderly themselves.

For the remainder of this discussion, focus will be on the software component as this is key to the success of AAL. As people age, their need for AAL technologies will increase, resulting in AAL systems evolving and growing more complex over time. Some of the inherent characteristics of such a platform that supports evolution are as follows:

- *Open* Invariably, propriety AAL systems will be marketed. However, these can be expected to address only a subsection of those issues that AAL is called to address—probably those that are most common. Given the heterogeneity of the elderly population, their diverse needs and housing situations, it is difficult to envisage how a propriety system can be guaranteed to address all circumstances that may arise.
- *Scalable* AAL systems must support the integration of additional hardware sensors in a seamless and transparent manner. Implicit here is a support for any international standards that arise in the AAL domain.
- *Adaptive* AAL must be robust and be capable of adapting in real-time to situations as they arise. Such situations may involve the elderly, or it may involve dealing with component failure.
- *Intuitive* Support for a sophisticated range of interaction modalities is essential. How this can be realised in practice will be context dependent. In some cases, a passive AAL system may be adequate. In others, it may be need to be more proactive and need access to range of actuators and effectors.

For the remainder of this paper, the discussion will focus on those issues necessary to realise a robust, evolutionary platform upon which a range of services of the elderly can be built upon.

3 A middleware perspective

Middleware (Geihs 2001) may be regarded as a software construct that mediates between two or more disparate software components. More colloquially, it is frequency seen as the "glue" that binds different components together to realise a new application or service. Though frequently associated with enterprise information systems, middleware can be developed for almost any application domain. If designed appropriately, it represents a useful vehicle for delivering the characteristics of openness, scalability and adaptivity described earlier. From an AAL perspective, middleware may be envisaged as representing a common interface to the range of sensors that are essential for monitoring behaviour, as well as aiding in both the construction of behaviour models and identification of times when interventions might be appropriate. A more challenging question is how and to what degree a level of intelligence might be imbued in the middleware so that, for example, it can associate collective sensor behaviour with individual contexts, and identify appropriate courses of action.

3.1 Middleware in conventional WSNs

In order to bring together the distinct components of a distributed sensor based application, a mechanism is required to mediate access to data. Traditionally, this has been provided by distributed databases resident on high powered nodes connected to the network through a gateway. Recently, there has been a shift towards in situ processing models provided by a middleware distributed on all the nodes of the network including the sensors themselves. Both the respective advantages and disadvantages of each are now considered. In addition to providing a dissemination function, the in situ processing capability of a middleware can also be leveraged to provide adaptivity within the network both from a user's perspective and a system perspective. Such an evolution of the system can deliver a consistent quality of service (QoS) and help maintain network performance over a prolonged period of time.

3.1.1 Distributed database approach

Traditionally, the model adopted for distributed sensor network applications revolves around distributed databases. Sensor information sent from the nodes is funnelled into these databases, either based on some schedule, or in response to messages from one or more controllers. Applications that require the data then harvest it from the databases and are generally unaware of the physical sensors that generated the data (Fig. 3).

3.1.1.1 Advantages One advantage of the database approach is the simplicity of the system. Many distinct vendors can provide integration across multiple networks by depositing the data in tables in the databases. Applications can then access this data in a homogenous manner to deliver the required service to the user. Secondly, the databases can be located as close as possible to the data production, delivering some degree of energy efficiency. However, there will always be some degree of communication as the database software requires a node with considerable resources to operate efficiently. By altering the frequency of the harvesting schedules, a rudimentary degree of adaptivity can be achieved by the networks; however, complex decisions based on established AI techniques cannot be accomplished within this model.

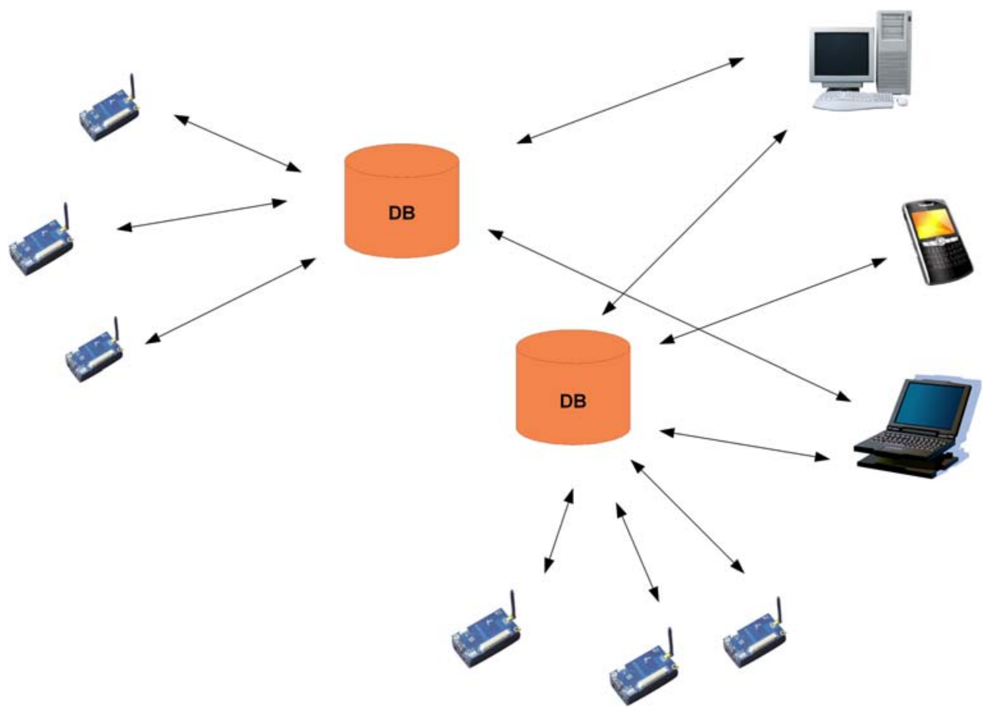

Fig. 3 Traditional DB model for WSN application in which WSN node report sensed readings to a database which are in turn accessed in a conventional manner

3.1.1.2 Disadvantages However, the traditional approach suffers from a number of significant pitfalls. Firstly, some of the data delivered to the databases may not be required by any user or automated service leading to inefficient storage and consumption of valuable bandwidth and energy of the nodes. Secondly, the delivery of all information over many hops may only be required for a simple calculation. Performing the calculation closer to the produced data can be significantly more energy efficient. Typically, such an approach has a limited capacity to conduct any adaptivity to the user and/or system as the only real decisions available are the data harvesting schedules.

3.1.2 Distributed processing approach

The distributed processing approach on the other hand does not centralise the processing and storage of the information but may include databases as the application requires them. The data handling is delegated to the nodes themselves and, in addition, so is the decision-making about the behaviour of the individual node. This user and system centric decision making will be considered in Sect. 3.2.

3.1.2.1 Advantages The advantages of this approach stem from the in situ reasoning and processing capabilities. While this issue will be examined in more detail subsequently, the primary benefit of this is the network adaptation and decision making which optimise the network's performance. Conducting this in a centralised fashion can either lack responsiveness or be too power consuming to be of practical benefit to the system as a whole. One of the inherent disadvantages of the database model is the potential for transmission of large data volumes and this can be mitigated using the distributed approach. By processing the data where it is generated, vast reductions in network traffic may be achieved. However, the savings will be application-specific as some deliberations may require data from multiple sources in order to complete their operation. In addition, when the middleware is realised using a multi-agent system (MAS), there may be the added benefit of agent migration to mitigate transmission cost further. Agent migration in WSNs has been considered previously by Muldoon et al. (2009). Again, its effectiveness will depend on the communication patterns of the distributed entities.

3.1.2.2 Disadvantages The main drawback with such an approach, as with any other distributed system, is the complexity of the solution as well as the testing and debugging of the components. In the database model, the tasks of the individual components are well defined. For example, the sensor network vendor's job is complete once the data is successfully stored in the data base. The application developer need only be concerned with accessing the database. Such a clean interface adds simplicity to the solution; however, it also introduces numerous degradations in system performance as previously discussed.

3.2 System and user adaptivity

While one of the primary functions of the middleware is concerned with the instantaneous operation of the system, another goal is the evolution of the system over time. This is split into two distinct areas:

1. the adaptivity to changing user needs over time;
2. the adaptivity of the system to both meet those new requirements as well as runtime constraints on the system such as power consumption.

3.2.1 User centric evolution

As an individual ages, AAL systems should evolve to meet their changing needs and preferences. These can either be explicitly stated by the user through explicit interaction with the system or can be learned as a result of observation and implicit interaction (O'Grady et al. 2008).

3.2.2 System centric evolution

AAL systems must also evolve in response to changes in the contexts in which they operate. For example, the current generation of sensor nodes are battery powered and therefore, there must be judicious use of power if system longevity is to be attained. In opposition to this is the requirement for as much data as possible for the application, as such data is essential for the construction and maintenance of sophisticated behaviour models. Balancing the longevity-QoS trade-off is only something that can be conducted at runtime in response to unforeseen events such as the loss of a node and/or the individual patterns of a user requiring more activity from a particular subset of nodes, for example. Thus, runtime decisions may need to be taken that will adversely affect one or both of these performance parameters. In practice, this may result in a trade-off between the longevity of the AAL system, and the QoS it provides, depending on the prevailing context. Given the nature of the AAL domain, it is probable that maintaining QoS will be a key imperative.

3.3 An intelligent programming abstraction

In light of the previous discussion, it can be seen that significant trade-offs may need to be made if the middleware is to function in a reliable yet adaptive manner. This may demand a sophisticated decision-making capability. Such a capability is also essential for behaviour model construction and instantaneous behaviour recognition. Thus, the availability of contextual data, both system and user originated, needs to be tightly coupled with the sensor data fusion and decision making processes. While a number of approaches may be harnessed, the case of lightweight embedded intelligent agents (O'Hare et al. 2006) is now considered. Such agents may operate on computationally limited devices such as sensor platforms, are inherently distributed and possess an innate ability to collaborate in the pursuit of application goals and objectives.

4 Intelligent agents

An agent is an autonomous entity, that is, one that is capable of deciding at run time what is required to achieve its goals or objectives without human intervention. A MAS is a form of computationally reflective system that is concerned with the interaction, collaboration, and competition of multiple autonomous agents. Though endowed with particular responsibilities, each individual agent interacts with other agents in the framework to fulfil the objectives of the system. In AI, the notion of an agent merges the concepts of behavioural systems with logical approaches. Agents are situated in the environment, but they are autonomous and do not simply react to it. Computational reflection is one of the most important areas of AI; it is a technique that enables a system to maintain meta-information about itself and use this information to change its behaviour or, in other words, to adapt. In agent parlance, this meta-information is commonly referred to as an agent's belief set or an agent's model of the world.

Intelligent agents symbolically model their environment and manipulate these symbols in order to act. To model intentional agency, an abstraction level is chosen for the symbols such that they represent mental attitudes. Agent oriented programming is a paradigm for directly programming the behaviour of agents using languages whose semantics capture a theory of rational agency (Shoham 1993). Agents often have a set of beliefs, desires, and intentions and an interpreter that determines how the agents should achieve their goals within the environment (Rao and Georgeff 1995).

Central to the coordination and collaboration of multiple agents is the existence of an agent communication language (ACL) that is shared and understood by all agents.[1] The necessity to support inter-agent communication has led to the development of an international ACL standard, which has been ratified by the foundation for intelligent physical agents (FIPA). More recently, FIPA has been subsumed into the IEEE computer society and forms an

[1] Game Theory approaches can be used to enable agents to coordinate actions without communication through the assumption of mutual rationality, but in the majority of frameworks agents do communicate.

autonomous standards committee with the objective of facilitating interoperability between agents and other non-agent technologies.

4.1 An embedded agent framework

For the purposes of this discussion, agent factory micro edition (AFME) (Muldoon et al. 2009; Muldoon 2007), which is a minimised footprint intelligent agent platform for embedded devices, is considered. AFME is based on agent factory (Collier et al. 2003; O'Hare 1996), a pre-existing agent platform for desktop environments. In AFME, as with many other intelligent agent platforms, system functionality is delivered through a combination of imperative and declarative code.

In AFME, the imperative functionality is in the form of a set of perceptors and actuators.[2] The declarative functionality is in the form of commitment rules, which define the conditions under which agents should adopt commitments to perform primitive actions or plans. In short, perceptors generate meta-information (beliefs) about the system state along with information related to the environment, potentially coming from hardware infrared sensors, power monitors, or video cameras for example. Using this information, the agent will decide, using its internal declarative rule set, on the actions that need to be performed. Actuators provide the functionality for primitive or atomic actions. In AFME, the functionality for the perceptors and actuators is implemented within Java (the CLDC J2ME configuration). It should be noted that when an agent adopts a plan or a primitive action fails, a commitment management process is invoked. A discussion of commitments and the commitment management process is beyond the scope of this discussion.[3] An example of a simple AFME rule is as follows:

$a(?var), b(?var) > doSomething(?var);$

In the above example, if the agent adopts the beliefs a(?*var*) and b(?var), the action to *doSomething(?var)* will be adopted. That is, the belief sentence represents the predicates or conditions under which actions are performed.[4] In AFME, the **?** symbol represents a variable. In AFME, the truth of the belief sentence is evaluated using resolution based reasoning, which is the goal-based querying mechanism used within Prolog.

4.2 Harnessed agents in AAL

In this section, the use of agents for the tracking and monitoring of an elderly person suffering from Alzheimer's disease is considered. In our laboratory, various areas, corridors, and rooms represent different parts of an instrumented house. In this environment, a range of Crossbow TelosB motes and infrared sensors are deployed. The infrared sensors provide room level tracking of user movement. In order to debug and tweak various parameters of the sensor deployment, we are using Octopus, which is a tool for the management, and user directed control, of sensor networks (Ruzzelli et al. 2009).

In the current configuration, the agents are not residing on the motes; rather they are deployed on an intermediate workstation. The agents periodically query a database through the use of a JDBC bridge. The code for invoking the JDBC calls is embedded within AFME preceptors. As noted previously, a preceptor is a Java class that enables agents to adopt or generate meta-information, referred to as beliefs. Using these beliefs and rules defined within a predicate logic, agents decide how to act. When an agent receives a result set from the database, it loops through the entries within a preceptor and adds various beliefs to its belief set.

In addition to receiving information from the sensors via the database, agents send messages to control the sensors by passing messages to the gateway of the network. This message passing is performed through the use of a message sending actuator. If the agent is residing on the same platform as the gateway mote, the serial port is used directly for sending messages to the gateway. If the agent is operating on a separate platform, messages are sent to the serial forwarder. The serial forwarder is a component of TinyOS that facilitates remote access to the serial port. In this architecture, from a logical perspective, these two approaches are equivalent; however, with the latter approach, the latency of message passing will be increased. It is therefore desirable to have the agents as close to the gateway as possible, if not residing on the gateway itself. Prior work has demonstrated the viability of deploying agents on the Stargate WSN gateway (Tynan et al. 2008).

In order to illustrate the use of the system, consider the situation in which an elderly person does not leave an area containing a staircase for an extended period of time (Fig. 4). A Patient Monitoring Agent periodically connects to the database and obtains information related to the patient's location, that is, the area of the building in which the patient is located. The Patient Monitoring Agent will not do anything if the patient is moving. If 20 min elapse

[2] It should be noted that the word perceptor is used rather than sensor to distinguish the software component from the hardware. Additionally, there are perceptors, such as perceptors that monitor system state, that do not have a hardware analogue.

[3] The interested reader is directed to (Collier 2001).

[4] For ease of discussion, we have simplified description of the process here. A more accurate description would be the belief sentence specifies the conditions under which a commitment management process commences.

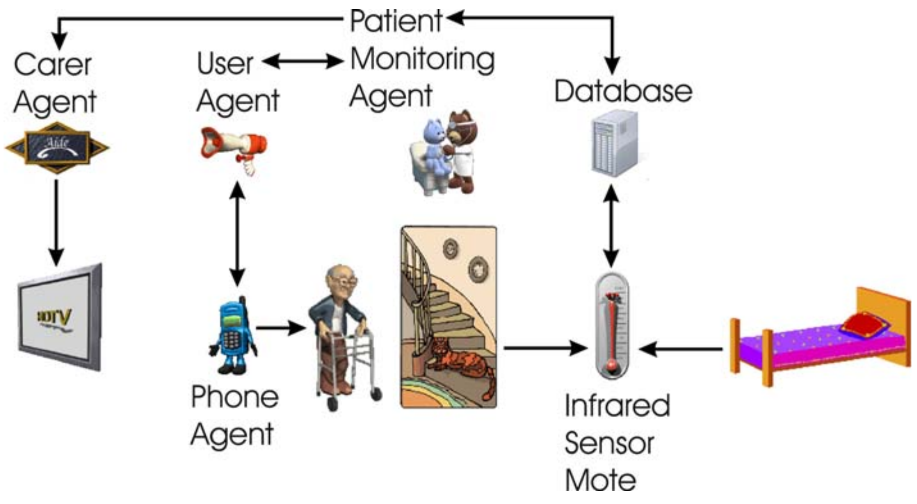

Fig. 4 Monitoring the patient in the home using a suite of collaborating agents

and the patient has not moved, however, it will obtain descriptive information related to the room in which the patient is located. In this system, descriptive and declarative information related to room content and functionally, along with geo encoding data, is stored within an ontology. At present, this ontology is stored on a server that agents can query. Using this information, agents can infer certain facts about perceived information. For instance, in this example, if the patient were in their bedroom, the alarm would not be raised. As discussed earlier, the decisions that agents make are determined through the use of a predicate logic. That is, the conditions that the agent believes to be true, based on perceived information and inferred facts through the use of an ontology, will determine the commitments that it adopts and the actions that it takes with respect to those commitments. The conditions and actions that an agent takes are encoded within the agent design by the developer when writing the code for the agent. In this example, the agent will raise the alarm, as it will determine (from the geo coded information and data coming from the server) that the patient is in a corridor and there is a staircase in the corridor. Of course, the agent must be aware of the consequences of having beliefs related to the staircase and corridor. This is achieved through the use of an inference engine that interprets information encoded within the ontology.

When the alarm is to be raised, the Patient Monitoring Agent must decide what action to take. Initially, it contacts the User Agent. The User Agent is responsible for communicating with the patient. To do this, the User Agent first determines if there is a visual television display in the patient's vicinity that it can send a message to. In this case, there is no visual display in the patient's area. The User Agent subsequently contacts the patient's Phone Agent, which resides on the patient's mobile phone. The Phone Agent determines that it is in the same area as the user and thus displays a message on the screen and makes a beeping noise. It then waits for a response from the user. After 1 min has elapsed, the User Agent informs the Patient Monitoring Agent that the patient has not responded to the message. On receiving this information, the Patient Monitoring Agent contacts the Carer Agent, which is responsible for communicating with the carer for the patient. The Carer Agent initially checks the database to see if the carer is in the house. On locating the carer, it obtains information related to the communication devices within the carer's area. It determines that there is a visual display unit close to the carer. It interrupts the carer's television program and sends the message: "Check up on Pat at the staircase". The carer proceeds to the staircase. Had the carer not been in the house, their Phone Agent would have been contacted.

In this architecture, the database is useful in that it enables the sensed values to be stored. This enables analysis to be performed, and data mining and post processing algorithms to be used, on the information. It facilitates the reruns of experiments and applications, which can be visualised within Octopus. The information can also be used to drive a simulator with real data. Algorithms may be developed using the stored information whilst other applications are operating on the motes. Developing algorithms with the same fixed set of information rather than unpredictable information coming from a real deployment is useful in certain situations. The reason for this is similar as to why when developing algorithms with a random component, developers often use the same seed for the random number generator in the testing stage; it enables results of different algorithms to be compared accurately. If a degenerate case occurs in one test, it will also occur in subsequent tests, as the same set of pseudo random numbers will be generated. In a similar vein, using information from the database facilitates the accurate comparison of algorithms developed for the network. It enables experiments to be accurately repeated with the same data set.

There are some limitations to using offline data for the development of agent algorithms, however. With agent systems, data is not just simply harvested from the network and stored. Typically, with agents, there is a decision process that takes place and based upon this decision process actuation occurs. If stored data is used and the algorithm or decision process is changed, this change will not be reflected in the stored data as it is static. Therefore, this approach is only useful for rerunning the same agent algorithm. To test a new agent algorithm, it must affect a real sensor deployment. Subsequently, post processing, simulation, etc. can be performed on the stored information.

4.3 Ongoing developments

It is intended to deploy agents directly within the WSN in the next iteration. In this case, much of the decision making and intelligence will occur at the leaves rather than the trunk of the network. Traditionally, with sensor networks, the minimal amount of processing would be performed on the motes, due to the severely limited computational resources available. In a wireless sensor network the most expensive operation to be performed is communication. For instance, on a typical node, the sending of 1 bit of information is equivalent of the execution of 1000 instructions locally (Hill 2003). It is clear that, in certain situations, it will be better to perform processing locally, rather than sending information back to the base station and waiting for a response.

In order to deploy agents within the network with AFME, more powerful motes are required, for example, the Sun SPOT or the Imote2. AFME has been ported to these devices. It is important to make a distinction between these types of devices and other Java sensor motes, such as the Sentilla. Devices, such as the Sentilla, run on computers that are considerably more resource-constrained than other high specification devices. For instance, if the Sentilla is compared with the Sun SPOT in terms of memory, speed, and power consumption, the difference is in orders of magnitude. This effectively means that more adaptive and intelligent algorithms can be developed for high specification devices, but at a cost in terms of resource usage. Although it may not be possible to deploy reflective agent platforms, such as AFME, on the lowest specification type sensors, there is still a rich body of research, such as collaboration and coordination algorithms within multi-agent systems that can be harnessed.

Most sensor networks, in their current state, are not resource aware and do facilitate truly collaborative behaviour when it comes to resource management. One notable exception to this is Peloton (Waterman et al. 2009), which is an agent-based Sensor OS that has been developed at Harvard University. Peloton targets low specification devices. It has been developed in nesC, but it differs from TinyOS in that it facilitates coordinated resource management. Since it has been developed in nesC, it can used on devices developed for TinyOS without have to develop new drivers. This work also seeks to develop coordination and collaboration algorithms for low specification devices, but it differs from Peloton in that it targets the Java language.

Although, nesC has certain technological advantages over Java and facilitates certain compile time optimisations that could not be done with Java, in the future, Java is likely to dominate WSN application development. The reason for this is simple. Companies will be able to leverage their current Java development staff to develop sensor network applications with little or no additional training costs required. Java is easier to program and has a lower learning curve than nesC. The advantages of object oriented approaches over procedural approaches have been well documented throughout the 1980s and 1990s. Although the Sun SPOT has a considerably higher overhead than Berkeley type mote, it is quite possible to use Java on lower specification devices. For instance, Java has been deployed on Mica2 motes with the TakaTuka JVM (Aslam et al. 2008). This is lowest specification type of device that the J2ME CLDC configuration has been deployed to. TakaTuka is expected to be become publicly available in late 2009.

5 Integration and evolution

While the technologies described in the previous sections offer one approach to realising viable and scalable AAL systems, our experience to date has concretely highlighted the repercussions of the design choices upon the maintainability, the interoperability, the scalability, and the performances of both the architecture and the developed services. These issues are crucial in the realisation of evolutionary AAL systems and merit further discussion.

5.1 Challenges

From a system engineering perspective, the requirement to deliver long-term, personalised and adaptive services, demands that the system should offer a sub-stratum over which the capabilities and the knowledge of intelligent agents may be openly applied, not only across different parts of the same smart environment, but also across heterogeneous and evolving deployments. One of the problems is that software modules and services are affected during the life cycle of an AAL system, either directly or indirectly. For example, in addition to direct modifications, software modules will be indirectly affected by replacements or updates of both hardware (sensors, actuators) and other software elements (operating systems, third party libraries) involved in their operations. Such occurrences are normal in any AAL research project, which is not only subjected to software and hardware ageing issues but which also requires experimentation with possible configurations of the system. In addition, replacement of obsolete/inadequate hardware and changes to other parts of the system, such as the introduction of new processes and functionalities, will likely have an indirect impact upon the previously deployed solutions. The problem may be exacerbated if the same functional elements are re-used across multiple services. Considerations such as code-reuse and design efficiency may mean that different modules will probably share parts of their code even if, at run-time, there is no connection (either in the form of control or data) between their instantiated versions (for example through standard object oriented mechanisms such as polymorphism and inheritance).

Supporting heterogeneous and integrated solutions also requires flexible management of the data communication mechanisms employed in each development and applicative context. Crucially, the differences between the possible data exchanged within multiple AAL applications and services (in terms of content, rates and bandwidth requirements, and also considering the diversity of stakeholders and their requirements) cannot be handled by a single communication mechanism. Instead, there is the need to leverage upon diverse dedicated solutions and legacy systems, for example by combining ZigBee wireless communication with Bluetooth, JDBC interfaces, HTTP-based web services and really simple syndication (RSS) feeds, and even with CORBA to better support rapid deployments and re-configuration of more heavyweight software modules thanks to network transparency and naming services. In addition, one system may need to combine these communication mechanisms with middleware that specifically addresses the distribution of media data (video/sound) such as the Java Media Framework (JMF), or combine data discovery mechanisms, e.g., based upon data dissemination techniques such as multicast, with point-to-point agent communication systems. All these communication mechanisms provide dedicated and optimum solutions to specific problems or provide access to a large base of associated tools and libraries; hence they should be integrated for opportunistically satisfying all the needs and constraints of one system. Additionally, an integrated approach may be the only solution if architectures need to be integrated with other legacy systems that are incompatible, for example, in terms of communication protocols.

However, in addition to the complexity of managing distinct data-exchange formats and transport mechanisms, combining multiple solutions also adds the overhead of managing their distinct configuration and evolution, as required for example to support new versions of the hardware and/or the underlying software components. Crucially, it then becomes difficult to manage their relationships, for example, to guarantee some global policies, such as policies for security, privacy or resource management.

5.2 Research directions

In order to face these challenges, the re-use of select elements of solutions already developed is advocated, in concert with the combination of different software engineering paradigms, tools, and design methodologies described elsewhere (Dragone et al. 2009; Dobson et al. 2007).

5.2.1 Agent-based autonomic systems

Though Agent Oriented Software Engineering (AOSE) has been leveraged thus far in terms of multiple autonomous software agents with symbolic, context-aware reasoning capabilities, a greater involvement of such agents in the management of the system's computational environment is advocated. One important example of such an approach is the two-tiered communication used in the RETSINA MAS (Berna-Koes et al. 2004) to integrate low-level (non-ACL) communication mechanisms (backchannels). RETSINA's backchannels are specifically designed to cater for the flow of the type of low-level data (e.g. video, sensor data, and control instructions) that could not be carried through ACL or would result in inefficiencies in terms of greater processing overheads being invoked, without a tangible benefit being observed. The management of these backchannels is, however, carried out via ACL, with agents communicating in this way in order to establish and close backchannels as necessary in order to accommodate their differences while safeguarding and respecting global policies in a scalable manner.

More opportunities for system's modularity may be enabled through the adoption of role- and goal-based AOSE methodologies. By employing an organisational/social view through which analysis and modelling of inter-agent interaction can be performed, the former help structuring one system as a set of more manageable sub-systems by acting as abstract specifications of behavioural patterns. The latter is the key to enabling the modular refinement of each agent design, as the application domain can be examined in terms of what needs to be achieved, rather than the types of behaviour that will lead to achieving it. The fundamental observation is that goals, as compared to behaviours or plans, are more stable in any application domain, and changes to existing plans can be accommodated as new ways of achieving the same goal. Within AAL applications, both principles can be harnessed to account for the diversity and the evolution of AAL systems by enabling agents to operate at the knowledge level while abstracting from the specific implementation details of the system's functional layer. While, in principle, autonomic functionalities can thus be implemented by using agents to create high-level behaviours coordinating the system's functional layer, it is important to provide the latter with proper interface and reflective functionalities, as will now be explained.

5.2.2 Component and service based systems

Component orientation is one of the most promising methodologies for supporting the development of modular AAL architectures. Component-Based Software Engineering (CBSE) (Szyperski 1999) operates by posing clear boundaries between architectural modules (the components) and guiding the developers in assembling these components into system architectures. Compared to Object Oriented Programming (OOP), Component Oriented Programming (COP) represent a more formal and systematic approach to modular design which enhances both system and code reuse through the introduction of strict rules regulating both the analysis/design of the architecture and its development process. Contrary to object's call interfaces, components' interfaces are separate explicit architectural elements that can be programmatically manipulated. Such a level of flexibility is essential for implementing framework-level mechanisms for ensuring that inter-component dependencies evolve along well-defined and anticipated directions so as to guarantee the preservation of system wide quality attributes. Within CBSE, domain analysis captures the principle quality attributes of a class of application and expresses them in the form of a component model. This typically provides an unambiguous description of the different component types—their features and behavioural properties, and the set of their legitimate collaborations. The same directions are pursued in service-based systems, especially in Internet-based systems with service interfaces based on SOAP and/or X-RPC protocols. For AAL applications, components and services organised in component contexts based on standard CBSE technologies, such as OSGi, should be used to:

- wrap resources (e.g. databases) and sensors,
- reify particular services and functions offered by the smart environment (for example, activity recognition modules);
- provide a consistent interface to interact with the system's functional layer.

As has been described, a middleware designed around the agent paradigm offers one interpretation of how such an architecture might be constructed.

5.2.3 XML-based encoding, SensorML and ontologies

Essential to achieving the characteristics outlined in Sect. 2.6 is the adoption of well accepted standard encoding formats to describe sensor's data, as well as events and agent's beliefs inferred through the use of agent perception modules. Standard encodings and associated meta-data annotations are important to increase system's interoperability and also provide a consistent view of the underlying information, for instance, for testing newly developed services against past activity logs, but also to relate new and past observations and thus leverage on previously acquired skills to serve the needs of the user despite changes in the smart home environment. To these ends, the great advantage of XML-based data encoding is that XML provides a structured data representation built for the purpose of separating content from presentation but that, at the same time, can be used to carry both the data and the metadata describing it. A huge array of software already support the adoption of XML, such as DOM, SAX, and JAXB, for parsing, automatic marshalling/de-marshalling and data binding; and XML-based communication middleware, such as XML-RPC and SOAP. Most importantly, XML is nowadays the de facto standard for electronic data interchange on the Web and it is used as the encoding format of various ontology languages used within the Semantic Web initiative.

System interchanging XML-encoded data can query the metadata component and learn the characteristics of its associated data, for example, in terms of array sizes, data formats, and so on. These characteristics enable the manipulation and transformation of content independently of its presentation, for example, to processing or display modules, and also to dramatically reduce development and maintenance costs by easing the integration of heterogeneous systems. XML ensures the easy addition or removal of attributes and entities without invalidating old data, or being forced to update all the code that processes old versions of the data. In addition, if necessary, it is also possible to more formally define the structure of XML documents and their content through Document Type Definition (DTD) and/or XML Schema. Compared to OMG IDL, the difference is that these validation steps can be optional and validation may happen at authoring time but it can also be done at serving, loading, or runtime if needed, e.g. to adjust to different computational constraints.

Although XML is self-describing, human readable and simple to amend, its verbosity can be a serious drawback when it comes to its use of communication bandwidth as part of online analysis tools. One alternative that is receiving increasing attention is to preserve the structure of XML but use non-text representations for greater efficiency, base64 binary encoding being a case in point. Another approach is to tailor encodings to the application by introducing application-specific data types, defined with XML Schema as done in Fast Web Services (Sandoz et al. 2003), for example. These can be equivalent to a pure binary representation of the data and offer the opportunity to both reduce the size of the data and eliminate much of the processing overhead compared to text-only XML.

For AAL applications, OGC SensorML (OGC 2007) provides a specialisation of XML that is specifically oriented in describing sensors and sensors' data, and which also includes the instructions for deriving higher-level information from observations. Processes described in SensorML are discoverable and executable. All processes define their inputs, outputs, parameters, and methods, as well as providing relevant metadata. In addition to the focused modelling effort brought about in more than 10 years of the OGC initiative, SensorML is valuable for the way it builds on common data definitions that are used throughout the OGC Sensor Web Enablement (SWE) framework, and in the associated standardization of related Web Services, such as the OGC Sensor Observation Services (SOS), Sensor Planning Services (SPS), and Sensor Alert Services (SAS).

The self-describing characteristic of SensorML-enabled sensors and processes could already support the development of auto-configuring sensor networks, as well as the development of autonomous sensor networks in which sensors can publish alerts and tasks to which other sensors can subscribe and react. However, it is very important to enrich the low-level description of devices and services with high-level, application-specific concepts. For instance, two devices/services with the same input/output signatures could be used for very different objectives in the same system. In the case of AAL, the system should not only be aware of the existence of a passive infrared sensor in a room at a given coordinate but also that the sensor is actually detecting the user entering into the kitchen, in order to enable a chain of inferences about the user's likely future activities.

To this end, the proprietary logic languages used in multiagent toolkits should be provided with standard ontological reasoning capabilities. Based on Description Logic, ontology languages such as the Ontology Web Language (OWL) used within the SemanticWeb initiative can be used to define complex concepts and relationships among them, as well as specific factual information. Both

```
<swe:DataRecord
definition="urn:ogc:def:property:OGC:atmosphericConditions">
<swe:field swe-om:Quantity rdf:about="#AirTemperature"
name="AirTemperature">
<swe:Quantity definition =
"urn:ugc:def:property:OGC:AirTemperature">
<swe:uom code="Cel" swe-om:hasUomIdentifier
rdf:about="http://sweet.jpl.nasa.gov/ontology/units.owl#degreeC"/>
<swe:value swe-om:hasDoubleValue rdf:datatype="&xsd;double">
35.1</swe:value>
</swe:Quantity
</swe:field>
</swe:DataRecord>
```

Fig. 5 Sample sensor data in XML + RDFa form

the ability of SensorML XML encoding to support semantic extensions through references to external ontologies or through RDF annotations (see Fig. 5) and the partial modelling of SensorML concepts in standard ontologies may be harnessed to create a correspondence between low-level and high-level concepts (Niles and Pease 2001; Russomanno et al. 2005) in order to improve the modularity of future agent-based solutions and thus ease their portability across heterogeneous deployments by creating a common repository of services and resources.

Ongoing research in this area involves how best to leverage on SensorML across the whole AAL system, for example, using its modelling of computational constrained devices without deploying web services on each device. A second area of research considers how computationally expensive ontologies can be most effectively utilised within computational efficient agent toolkits.

6 Conclusion

Developments in AAL over the next few years will have a profound affect on society and the elderly in particular. Decisions made at this time juncture in the design of AAL systems may well have ramifications for years to come. It beholds those charged with the engineering of AAL systems to be acutely conscious of this and ensure their designs provide for flexibility, scalability and evolution during the aging process.

AAL must cater for two constituencies: those who are currently elderly and are more likely to be technology averse, and those who will be elderly in the next decade but who may be technology friendly. This paper focuses on the needs of the later constituency, and argues that AAL systems must evolve to meet the needs of individuals as they grow older. In particular, the issue of evolution must be considered both from the elderly or user perspectives, as well as from a systems perspective. In the later case, planning for and enabling the evolution of AAL systems poses significant engineering problems. One approach, outlined in this paper, proposed the harnessing of embedded agents as a potential mechanism for enabling open, scalable and adaptive infrastructures upon which evolutionary AAL services can be delivered. Ongoing research is concerned with the practical system engineering of such systems, and the identification of a real-world setting for its deployment such that the ethical and legal issues that govern such deployments are satisfactory addressed.

Acknowledgments This work is supported by Science Foundation Ireland (SFI) under Grant 07/CE/I1147.

References

AALIANCE (2009) Ambient assisted living roadmap. http://www.aaliance.eu/public/documents/aaliance-roadmap/aaliance-aal-roadmap.pdf. Accessed 15 October 2009

Anastasopoulos M, Niebuhr D, Bartelt C, Koch J, Rausch A (2005) Towards a reference middleware architecture for ambient intelligence systems. In: ACM conference on object-oriented programming, systems, languages, and applications

Aslam F, Schindelhauer C, Ernst G, Spyra D, Meyer J, Zalloom M (2008) Introducing TakaTuka: a Java virtualmachine for motes. In: SenSys 2008: proceedings of the 6th ACM conference on Embedded network sensor systems. ACM, New York, pp 399–400

Berna-Koes M, Nourbakhsh IR, Sycara KP (2004) Communication efficiency in multi-agent systems. ICRA 2004, pp 2129–2134

Bourke AK, O'Brien JV, Lyons GM (2007) Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm. Gait Posture 26(2):194–199

Brumitt B, Meyers B, Krumm J, Kern A, Shafer S (2000) EasyLiving: technologies for intelligent environments. In: Proceedings of HUC 2000, Bristol, UK

Callahan D (1973) The WHO definition of "health". Hastings Centre Stud 1(3):77–87

Collier RW (2001) Agent factory: a framework for the engineering of agent-oriented applications. Ph.D. dissertation, University College Dublin, Dublin

Collier RW, O Hare GMP, Lowen T, Rooney C (2003) Beyond prototyping in the factory of the agents. In: 3rd Central and Eastern European conference on multi-agent systems (CEEMAS 2003), Lecture Notes in Computer Science (LNCS), p 2691

Cook DJ, Augusto JC, Jakkula VR (2009) Ambient intelligence: technologies, applications, and opportunities. Pervasive Mobile Comput 5(4):277–298

Cook DJ, Das SK (2007) How smart are our environments? An updated look at the state of the art. J Pervasive Mobile Comput 3(2):53–73

Dimitrov T, Pauli J, Naroska E (2007) A probabilistic reasoning framework for smart homes. In: Proceedings of the 5th international workshop on middleware for pervasive and Ad-Hoc computing (MPAC '07), California, USA, pp 1–6

Dobson S, Nixon P, Coyle L, Neely S, Stevenson G, Williamson G (2007) Construct: an open source pervasive systems platform. In: Proceedings of the IEEE consumer communications and networking conference, Las Vegas, NV, USA

Dragone M, Lillis D, Collier RW, O'Hare GMP (2009) SoSAA: a framework for integrating components & agents. In: Proceedings of the 24th annual symposium on applied computing (ACM SAC 2009), special track on agent-oriented programming, systems, languages, and applications, March, Honolulu, USA, pp 8–12

European Commission (2009) Aging communication: a renewed strategy for tackling Europe's demographic challenge

Floeck M, Litz L (2008) Activity- and inactivity-based approaches to analyze an assisted living environment, emerging security information, systems and technologies. In: Proceedings of the second international conference on SECUREWARE 2008, 25–31 August, pp 311–316

Fugger E, Prazak B, Hanke S, Wassertheurer S (2007) Requirements and ethical issues for sensor-augmented environments in elderly care. In: Stephanidis C (ed) Universal Access in HCI, LNCS, vol 4554, pp 887–893

Geihs K (2001) Middleware challenges ahead. IEEE Comput 34(6):24–31

Giner P, Cetina C, Fons J, Pelechano V (2009) Building self-adaptive services for ambient assisted living. IWANN, LNCS, vol 5518, pp 740–747

Giuliani M.V, Scopelliti M, Fornara F (2005) Elderly people at home: technological help in everyday activities. In: ROMAN 2005. IEEE international workshop on robot and human interactive communication. IEEE Computer Society Press, Los Alamitos, pp 365–370

Hasselaar E, van Ginkel JT (2004) The healthy bedroom. In: Bonnefoy X (ed) Proceedings of the 2nd WHO international housing and health symposium, Bonn, Germany, pp 336–344

Helal S, Mann W, Zabadani H, King J, Kaddoura Y, Jensen E (2005) The gator tech smart house: a programmable pervasive space. IEEE Comput 38(3):50–60

Hill J.L (2003) System architecture for wireless sensor networks. Ph.D. dissertation, University of California, Berkeley

Home Safety Council (2002) The state of home safety in America

Kang, D-O, Lee H-J, Ko E-J, Kang K, Lee J (2006) A wearable context aware system for ubiquitous healthcare. In: Proceedings of the 28th annual international conference of the IEEE engineering in medicine and biology (EMBS 2006), New York, NY, USA, pp 5192–5195

Kientz JA, Patel SN, Jones B, Price E, Mynatt ED, Abowd GD (2008) The Georgia Tech aware home. In: Extended abstract of CHI, ACM, pp 3675–3680

Kleinberger T, Becker M, Ras E, Holzinger A, Müller P (2007) Ambient intelligence in assisted living: enable elderly people to handle future interfaces. In: Stephanidis C (ed) UAHCI 2007 (Part II). LNCS, vol. 4555, Springer, Heidelberg, pp 103–112

Kwasny M, Caine K, Rogers WA, Fisk AD (2008) Privacy and technology: folk definitions and perspectives. In: CHI 2008 extended abstracts on human factors in computing systems (Florence, Italy, April 05–10, 2008). CHI 2008. ACM, New York, NY, USA, pp 3291–3296

Leonardi C, Mennecozzi C, Not E, Pianesi F, Zancanaro M, Gennai F, and Cristoforetti A (2009) Knocking on elders' door: investigating the functional and emotional geography of their domestic space. In: Proceedings of the 27th international conference on human factors in computing systems—CHI '09 (Boston, MA, USA, 2009). ACM, New York, NY, pp 1703–1712

McCreadie C, Tinker A (2005) The acceptability of assistive technology to older people. Ageing Soc 25:91–110

Muldoon C (2007) An agent framework for ubiquitous services. Ph.D. dissertation, School of Computer Science and Informatics, Dublin, Ireland

Muldoon C, O'Hare G.M.P, Collier R.W, O'Grady M.J (2009) Towards pervasive intelligence: reflections on the evolution of the agent factory framework. In: Multi-agent programming: languages, platforms and applications. Springer, Berlin, pp 187–210

Niles I, Pease A (2001) Origins of the standard upper merged ontology: a proposal for the IEEE standard upper ontology. In: Working notes of the IJCAI-2001 workshop on the IEEE standard upper ontology, Seattle, WA, USA

O'Flynn B, Angove P, Barton J, Gonzalez A, O'Donoghue J, Herbert J (2006) Wireless biomonitor for ambient assisted living. In: Proceedings international conference on signals and electronic systems, ICSES 2006, Lodz, Poland, September 17–20, pp 257–260

O'Grady MJ, O'Hare GMP, Keegan S (2008) Intelligent modalities in mobile contexts. In: Virvou M, Jain L (eds) Studies in computational intelligence (SCI) 104, pp 89–106

O'Hare GMP (1996) Agent factory: an environment for the fabrication of distributed artificial systems. In: O'Hare GMP, Jennings NR (eds) Foundations of distributed artificial intelligence, sixth generation computer series, Wiley Interscience, London, pp 449–484

O'Hare GMP, O'Grady MJ, Muldoon C, Bradley JF (2006) Embedded agents: a paradigm for mobile services. Int J Web Grid Serv 2(4):379–405

OGC (2007) OpenGIS sensor model language (SensorML) implementation specification. Available at http://www.opengeospatial.org/standards/sensorml. Accessed 15 October 2009

Pollack ME (2005) Intelligent technology for an ageing population: the use of AI to assist elders with cognitive impairment. AI Magazine 26(2):9–24

Preuveneers D, Berbers Y (2008). Mobile phones assisting with health self-care: a diabetes case study. In: Proceedings of the 10th international conference on human computer interaction with mobile devices and services (Amsterdam, The Netherlands, September 02–05, 2008). MobileHCI 2008. ACM, New York, NY, pp 177–186

Rao AS, Georgeff MP (1995) BDI agents: from theory to practice. In: Proceedings of the first international conference on multi-agent systems (ICMAS 1995) (June 1995), pp 312–319

Russomanno DJ, Kothari C, Thomas O (2005) Sensor ontologies: from shallow to deep models. In: Proceedings of the 37th southeastern symposium on systems theory, IEEE Press, Tuskegee, Alabama, pp 107–112

Ruyter B, Pelgrim E (2007) Ambient assisted-living research in carelab. Interactions 14(4):30–33

Ruzzelli AG, Dragone M, Jurdak R, Muldoon C, Barbirato A, O'Hare GMP (2009) Poster abstract: an extensible dashboard for sensor networks control and visualisation. In: (EWSN) The 6th European conference on wireless sensor networks

Sandoz P, Pericas-Geertsen S, Kawaguchi K, Hadley M, Pelegri-Llopart E (2003) Fast Web services. Available at http://java.sun.com/developer/technicalArticles/WebServices/fastWS/. Accessed 15 October 2009

Schaie KW, Wahl HW, Mollenkopf H (2003) Aging independently, living arrangements and mobility. Springer, New York, pp 55–56

Segarra MT, André F (2009) Building a context-aware ambient assisted living application using a self-adaptive distributed model. In: Proceedings of the fifth international conference on autonomic and autonomous systems, pp 40–44

Shoham Y (1993) Agent-oriented Programming. Artif Intell 60(1):51–92

Sixsmith A, Johnson N (2004) A smart sensor to detect the falls of the elderly. IEEE Pervasive Comput 3:42–47

Steg H, Strese H, Loroff C, Hull J, Schmidt S (2006) Europe is facing a demographic challenge: ambient assisted living offers solutions. IST project report on ambient assisted living

Szyperski C (1999) Component software: beyond object oriented programming. Addison-Wesley, Reading

Tynan R, Muldoon C, O'Grady MJ, O'Hare GMP (2008) A mobile agent approach to opportunistic harvesting in wireless sensor networks (Demo Paper). In: AAMAS 2008: Proceedings of the seventh international joint conference on autonomous agents and multiagent systems, Estoril, Portugal, ACM (May 12–16 2008), pp 1691–1692

Vasilakos A, Pedrycz W (2006) (eds), Ambient intelligence, wireless networking, ubiquitous computing, Artec House

Vastenburg MH, Visser T, Vermaas M, Keyson DV (2008) Designing acceptable assisted living services for elderly users. In: Aarts et al (eds) Ambient intelligence, LNCS, vol 5355, pp 1–12

Wang Q, Shin W, Liu X, Zeng Z, Oh C, Alshebli BK, Caccamo M, Gunter CA, Gunter E, Hou J, Karahalios K, Sha L (2006) I-Living: an open system architecture for assisted living. In: Proceedings of SMC 2006, Taipei, Taiwan

Waterman J, Challen GW, Welsh M (2009) Peloton: Coordinated resource management for sensor networks. In: 12th workshop on hot topics in operating systems (HotOS-XII)