

Agent-based Ubiquitous Systems: 9 Lessons Learnt

N. Hristova, G.M.P. O'Hare & T. Lowen

PRISM Laboratory, Department of Computer Science, University College Dublin (UCD),
Belfield, Dublin 4, Ireland
{nataliya.hristova, terry.lowen, gregory.ohare}@ucd.ie
<http://www.cs.ucd.ie/>

Abstract. The recent emergence of agent prototyping environments for developing Java agents (Aglets, JATLite, JACK, Agent Factory) and the availability of the J2ME (Java 2Micro Edition) on light devices (PDAs and cellular phones) provides the ability for strong agenthood to be delivered in the mobile and ubiquitous sector. This is an experience paper that discusses the lessons learnt in the construction of agent-based ubiquitous systems. Within this paper we consider four such systems namely Ad-me, WAY, Gulliver's Genie, and Easishop. They all have been developed at UCD. The common denominator for all prototypes is the use of Lightweight BDI agents and the intelligent pre-caching of content.

1 Introduction

The Agent Oriented Software Engineering (AOSE) has emerged as a viable design approach for complex, dynamic and distributed systems. Ubiquitous and context sensitive systems are an instance of this broad class of system. The recent emergence of agent prototyping environments for developing Java agents (Aglets, JATLite, JACK, Agent Factory) and the availability of the J2ME (Java 2Micro Edition) on light devices (PDAs and cellular phones) provides the opportunity for the deployment of *strong* agents within the mobile and ubiquitous sector.

Within the context of this paper we wish to impart some of the lessons we have learnt from the design, delivery and evaluations of a variety of agent based ubiquitous systems. We identify a landscape of related work in section 2 before briefly introducing the systems that contribute to our experience portfolio. Within section 4 we formulate 9 lessons learnt from agent-based ubiquitous system design.

2 Related Research

Numerous influential systems in the ubiquitous sector have been widely discussed in the literature. Exemplar prototype systems that already have demonstrated the use of agent techniques are Personal Travel Assistant [10], Agents2Go[11], Agora[12], Im-

pulse[13], Mihailescu and Binder's m-commerce framework (MB) [14]. A comparison of such systems is given in Table 1. For more detailed description see [15].

Table 1. A comparison of Agent-based Ubiquitous Systems

System Name	App./ Framework	Agent System	M-commerce system	Migration	Autonomy	Personalization of Content	Multiple use	Context Sensitive
Agents2Go	F	Agents-2Go	✓	—	—	—	—	✓
Agora	A	Zeus	✓	—	—	—	—	—
PTA	A	Zeus	✓	—	✓	✓	—	—
Impulse	A	Hive	✓	—	✓	—	—	✓
MB	F	Aglets/ KVM	✓	✓	—	✓	✓	—
Ad-me	A	AF	✓	—	✓	✓	✓	✓
WAY	A	AF	—	✓	✓	✓	✓	✓
Genie	A	AF	—	✓	✓	✓	✓	✓
EasiShop	A	AF	✓	✓	✓	✓	✓	✓

3 Our Agent-Based Ubiquitous Systems Experience Portfolio

The common development platform that has been commissioned within all our systems has been that of Agent Factory (AF). Indeed to an extent the initial motivation in the design of each of the systems was little more than to test the viability of this environment for the rapid deployment of multi-agent systems. However each of these systems have taken on a life of their own and have made a significant research contribution in the mobile and ubiquitous sector.



Fig. 1. Graphical User Interfaces of Ad-me, WAY and EasiShop Systems

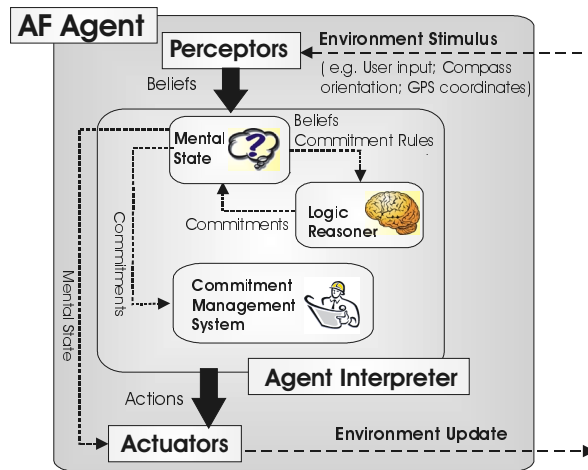


Fig. 2. AF Agent Schematic Architecture

AF [1,2] is a cohesive framework that supports a structured approach to the development and deployment of agent-oriented applications. Specifically, it delivers extensive support for the creation of Belief-Desire-Intention (BDI) agents. Such agents adopt an *Intentional Stance* through the explicit modeling of a mental state based upon an aggregation of various mental attitudes. AF is organized into two core environments: the AF Development Environment, and the AF Run-Time Environment. The former environment delivers a set of Computer-Aided Software Engineering (CASE) tools that support the Agent Fabrication Process, whilst the latter delivers support for the deployment of agent-oriented applications over a wide range of network-enabled Java-compliant devices.

An agent consists of a *mental state*, *commitment rules*, *perceptors* and *actuators* (see Fig.2). The mental state contains model of the current environment stored in the form of beliefs. Commitment rules represent the behaviour of an agent and define situations in which an agent should adopt a given commitment. The commitments are the result of the agent's decision making process and represent the actions that has to be taken. These actions are realized by the actuators. Actuators are the functional units that an agent uses to affect its environment, while perceptors are the functional units that an agent uses to build a model of its environment.

The feasibility of the AF has been proved via a number of applications, namely Adme (Advertising for the Mobile E-commerce user) [3,4], the WAY system [5,6], Gulliver's Genie [7] and EasiShop [8,9]. The common denominator for these systems is the adoption of a Multi Agent System design philosophy made through the utilization of AF. The systems are characterised by two important aspects firstly intelligent pre-caching of content and secondly intelligent adaptability both occurring in a variety of highly dynamic environments. All agents within these systems are implemented using the AF, they all utilize the same PDA namely the HP IPAQ (or versions thereof) and all systems for portability reasons adopt Java as the implementation medium. Some screenshots of these systems are depicted on Fig. 1.

Ad-me is an advertising service for the mobile user. The motivation and the added value service to the user is to develop context-sensitive tourist services accommodated upon a Personal Digital Assistant (PDA) or cellular phone. Ad-me focus is on a number of key research challenges in development of such systems including: system design; adaptation of the Graphical User Interface to the various device infrastructures; methods for utilizing context in advertising on wireless devices; taking into account user's emotions to increase effectiveness of advertising. The system agents matches location, time, previous user behavior in the system, as well as user preferences together with the properties of advertised objects to tailor advertisements for particular users.

The Where Are You (WAY) system is assisting mobile users in the location, tracking and rendezvousing with a variety of moving entities. In particular one of the most common and costly usage of mobile devices is the synchronization and rendezvous of people. WAY seeks to provide an alternate solution to this problem by deploying GPS, wireless and mobile agent based technologies. WAY enables location postings and updates to be passed between users with the minimum of fuss. Additional functionality allows users to rendezvous with other entities (e.g. Public Transport).

Gulliver's Genie (Genie) is a context-aware tourist guide for roaming tourists. The main objective of the system is dissemination of context sensitive information to tourists with particular emphasis on meeting the needs and expectations of cultural users. In particular the system aims to provide personalized, multimedia-enhanced presentations about attractions wirelessly transmitted in a timely manner [7].

EasiShop is a context-aware u-commerce shopping application targeted at mall shoppers, facilitated by the Bluetooth wireless transmission medium. It seeks to partially automate the shopping process. It can inform the shopper of optimum consumer conditions-cheapest price, best availability, best after sales service, geographic proximity-all.

While these systems offer similar challenges to the designer the design decisions taken within each have often been different and some of these systems have emerged building on the experiences of the others. While significant differences exist like the operating system choice, the architecture model adopted, the degree of agent migration supported the nature of the Graphical User Interface, and the very user service offered, nevertheless significant factors were adhered to.

4 The 9 Lessons Learnt

Emerging from our experiences in the development of ubiquitous systems the following nine lessons were identified.

Lesson 1: Client-Server Distribution. In order to cope with the various resource limitations a developer must decide how to distribute the different system components between the client and server. Pushing all computational processes to the server, e.g. using *thin* client and a *fat* server is a sensible approach when dealing with limited

handheld devices. However one must consider issues of scalability ensuring that additional server side agents are spawned in order to meet increases in client numbers, which may run to thousands or even millions. Pushing all computation to the server could overload the system easily. A careful balance is thus necessary in distributing the system. Typically software for a Client-Server architecture includes three distinct subsystems that can be allocated to the client, the server or distributed between both of them. These are *User Interaction/Presentation Subsystem*, *Application Subsystem* and *Database Management Subsystem*. In general we recommend a *Thin Client* design. The client-side should handle lightweight processing and occasion more demanding computational tasks, thus taking cognizance of the limited computational power of its host. In particular the interaction/presentation subsystem should be placed on the client. While the database is typically located on the server as it has to be shared by multiple users. The balance may need to shift dynamically. For this reason we have found agent migration to be crucial. In certain circumstances agents may migrate to the client in order to perform a specific task *en situe*, thereafter returning to the server or other client-side devices. Agents can not only to migrate, but also collectively make decisions as to when it is prudent to migrate. This capability offers powerful performance possibilities (see lesson 9).

Lesson 2: Minimise the Software Footprint in order for the application to function efficiently on the PDA. A form of the AF runtime environment has been developed with this in mind. Called AFLite, it is used by each application and is designed in such a way as to minimize memory requirements.

Lesson 3: Intelligent Pre-caching of Content. Data transfer rates over wireless networks are still slow. To minimize latency and improve performance an intelligent pre-caching mechanism has been implemented. The mechanism is utilized by the Genie and Ad-me systems. In terms of Genie, the user position is monitored and a list of near-by attractions is passed to the Cache agent. The agent then monitors user position and adjusts the list accordingly. Consultation with the User Profile Agent results in the partial loading of several candidate presentations based on these attractions. As soon as the user moves towards one of them, the system disambiguates the set and fully assembles the relevant one disposing of the others. A similar system is used by Ad-me when it loads relevant advertisements.

Lesson 4: Dynamic Agent Tuning. When several agents are running simultaneously on the same device considerations should be made for a sensible task scheduling. It is not imperative to have all the agents running at maximum power all the time. In fact, it may actually degrade the overall application by consuming resources unnecessarily. When initially testing the WAY system on the iPAQ we discovered that the map took an unreasonable length of time to load and sometimes it would not load at all. The iPAQ simply did not have enough resources to carry out this operation. The iPAQ had a map agent and two other system agents resident upon it. While the map agent was busy loading the map from the server the other agents were still active and the processor still poled them in order to identify any required activities. It was discovered that

increasing the sleep time of the idle agents could decrease the map load time. In addition we have begun to incorporate a mechanism whereby agents may autonomously and collaboratively negotiate relative process weightings opportunistically.

Lesson 5: Less is More. On a small device with a small screen, an elegant single screen GUI is preferable to a multi-windowed monstrosity. It is easier for the user to see and use while also proving far less memory intensive, resulting in a faster, more efficient application. All the AF applications employ this approach.

Lesson 6: Judicious Software Selection. The issue of thin client affects not only the distribution of data and processing but also the choice of software packages used. For example Swing has superseded AWT for Java GUI development on the desktop PC. Swing offers all the functionality of AWT and more features besides. It is also aesthetically more pleasing to the eye. It is even possible to get it running on PDAs. However AWT has the advantage over Swing in that it far less memory intensive. This is a vital advantage on memory-restricted devices, and we discovered that it gives AWT a great speed advantage over Swing. At the moment AWT is the API of choice for our PDA applications.

Lesson 7: Agent-based Application versus Web Server Application. From our experiences we have learnt the advantages afforded by agents over the traditional web server application approach. Agents offer proactivity, adaptivity, increased system robustness and the potential for graceful system expansion. Considering the Ad-me system in the initial prototype the browser, available on the user device, provided the medium through which the service was presented [3,4]. The user interface was developed within PHP scripts enhanced by the HAWHAW PHP library. The library ensured the presentation of Ad-me on various existing web browsers available. The browser as a running environment presented the following problems: constant network connection; inefficient speed and security restrictions notorious to applets; finally the processes are by nature asynchronous (regular updating of user position necessitated process synchronization). To fulfill the demand for a highly interactive application we replaced the browser-client with specialized stand-alone client agents. Our experience showed: Greater portability; greater control over the GUI and a reduction of network data transfer. The application performance, however, slowed down due to the increased load on the client side and the limited device memory. We consider system responsiveness to be in a large part due to redundant frequent requests made by the client agent to the server. Performance issues necessitate an optimization of the system. For this reason agent migration and tuning abilities are central.

Lesson 8: Operating System Choice. Ad-me and WAY were designed to be executed on the default operating system supplied with the IPAQ, PocketPC/Windows CE. The Genie project ran on the SavaJe XE operating system. SavaJe OS is unique in providing wireless developers with a totally native Java platform for applications. This greatly increased the speed of Gene. However problems arose when trying to install certain hardware devices due to the paucity of the SavaJe device drivers. As a result

we had to develop our own drivers. *Familiar v0.6.1 linux* build was used on the PDA by Easishop. This offered the benefits of stability, speed and efficiency. However again there were problems with lack of drivers. The trade-off between the various operating systems is that SavaJe and Linux offer greater speed and stability while WindowsCE/PocketPC are more widely supported by hardware manufacturers.

Lesson 9: Dynamic Load Balancing. Dynamic load balancing has proven necessary and is achieved through agent migration. Agents can come and go to devices as and when needed and therefore not use scarce resources unnecessarily. It is essential that migration of agents is both robust and efficient. Tests were conducted to determine transmission speed of agents across a Wireless LAN (see figure 3).

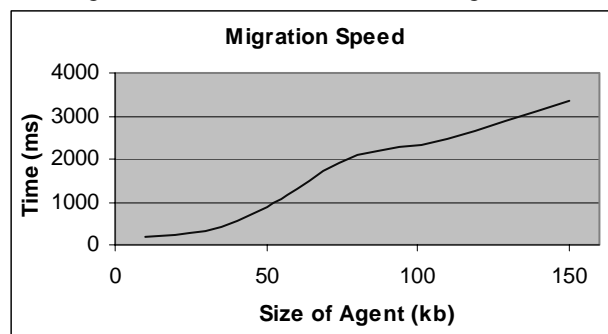


Fig. 3. Graph of Migration Speed vs. Size of Agent

The transmission data size of agents in AF is on average 20KB. We tested the migration speed of agents up to 150KB in size. Times for the largest agents came to be 3500ms. However it would be very rare in practice for AF agents to approach this size. Times of around 150ms for the agents were the norm. Testing within Easishop using Bluetooth indicated agent migration takes on average 320ms [8]. Easishop conducted series of performance and scalability tests that measured the reliability and efficiency of agent migration. Scalability tests have investigated the performance degradation as the agent community increased. At the time of writing, we could not locate any migration speed data for other existing mobile agent architectures.

5 Conclusions

Within this paper we have sought to outline some of our experiences in the design and development of agent-based ubiquitous systems. We have briefly characterized those systems, which have offered a fertile ground from which we have been able to harvest our mistakes and subsequently articulate our experiences. We have proffered these in the form of the 9 commandments for the design of agent-based ubiquitous systems.

These experiences have formed the catalyst for the design of the ACCESS [15] architecture, which seeks to provide a generic framework upon which a wide range of ubiquitous context sensitive services can be developed. ACCESS both envelops and

utilizes the AF system offering a multi-user environment offering personalization of content, by user profiling and context, support for mobile lightweight intentional agents and intelligent prediction of user service needs.

References

1. Collier, R.: Agent Factory: A Framework for the Engineering of Agent-Oriented Applications, Ph.D. Thesis, Department of Computer Science, UCD, Ireland (2001)
2. Collier, R.W., O'Hare G.M.P., Lowen, T., Rooney, C.F.B.: Beyond Prototyping in the Factory of the Agents, Proc. of 3rd Central and Eastern European Conference on Multi-Agent Systems (CEEMAS'03), Prague, Czech Republic. (2003)
3. Hristova, N., O'Hare, G.M.P.: Ad-me: A Context-Sensitive Advertising System, In Proc. of 3rd International Conference on Information Integration and Web-based Applications & Services (IIWAS). (September 2001)
4. Hristova, N. & O'Hare, G.M.P., Ad-me: Intelligent Context-Sensitive Advertising within a Mobile Tourist Guide, Proc. of 12th Irish Conference on Artificial Intelligence and Cognitive Science (AICS), NUI, Maynooth. (2001)
5. Lowen, T.D., O' Hare, P.T., O' Hare G.M.P, Mobile Agents point the WAY: Context Sensitive Service Delivery through Mobile Lightweight Agents. Proc. of Autonomous Agents and Multi-Agent Systems (AAMAS-2002), Bologna. (2002)
6. O'Hare, P., O'Hare G.M.P., and Lowen, T., Far and away: Context sensitive service delivery through mobile lightweight PDA hosted agents. In Proc. FLAIRS 02. (2002)
7. O'Hare, G.M.P. & O'Grady, M .J.: Gulliver's Genie: A Multi-Agent System for Ubiquitous and Intelligent Content Delivery, In Press, Computer Communications, Vol. 26, Issue 11, Elsevier Press. (2003) 1177-1187
8. Keegan, S. & O'Hare, G.: EasiShop: Context sensitive Shopping for the Mobile User through Mobile Agent Technology, Proc. of 13th PIMRC, IEEE Press, Portugal (2002)
9. Keegan, S. and O'Hare, G.M.P., Easishop: Enabling uCommerce through Intelligent Mobile Agent Technologies, Proceedings of 5th International Workshop on Mobile Agents for Telecommunication Applications (MATA'03), Marrakesh, Morocco, Springer-Verlag LNCS, (October 8th-10th, 2003)
10. Ndumu, D. T., Collis, J. C. & Nwana, H. S.: Towards Desktop Personal Travel Agents, BT Technological Journal 16 (3). (1998) 69-78
11. Ratsimore O., Korolev V., Joshi A., Finin T.: Agents2Go: An infrastructure for Location-Dependent Service Discovery in the Mobile Electronic Commerce Environment. First ACM Mobile Commerce Workshop, Rome (2001)
12. Fonseca S., Griss M., Letsinger R.: An Agent-Mediated E-Commerce Environment for the Mobile Shopper. HP Technical Report HPL-2001-157 (2001)
13. Youll J., Morris J., et al.: Impulse: Location-based Agent Assistance, Software Demos, Proc. of 4th Int. Conference on Autonomous Agents, Barcelona, (2000)
14. Mihailescu, P., Binder, W.: A Mobile Agent Framework for M-Commerce. Agents in E-Business (AgEB-2001), Workshop of the Informatik 2001, Vienna, Austria, (2001)
15. Muldoon, C., O'Hare, G.M.P., et al., ACCESS: An Agent Architecture for Ubiquitous Service Delivery, Proc. of Seventh International Workshop on Cooperative Information Agents (CIA), Helsinki. (August 2003)