Virtual Personal Assistants in a Pervasive Computing World

John F. Bradley, Brian R. Duffy, Gregory M.P. O'Hare, Alan N. Martin and Bianca Schön
Department of Computer Science,
University College Dublin,
Belfield, Dublin 4,
Ireland
{john.bradley, brian.duffy, gregory.ohare, alan.martin, bianca.schoen}@ucd.ie

Abstract - *Computing paradigms are constantly evolving – from mainframe to desktop, and on to ubiquitous computing. With a wide variety of devices, each with varying constraints, it would be quite difficult to develop services that would operate in a uniform manner across all platforms. With this in mind we offer an agent based solution. Agent attributes of autonomy, adaptability and mobility make them highly suitable for use in pervasive computing environments. Agent Chameleons illustrates a framework for Virtual Personal Assistants (VPA) for the information age. This paper briefly outlines some of the technologies and concepts that underpin the functionality of a VPA.*

Keywords: agents, migration, mutation, human-agent interaction, virtual personal assistants

1 Introduction

From mainframe to desktop, through to more mobile devices such as laptops and handhelds (mobile phones & personal data assistants), the concept of the computer has advanced from laboratory environments out into the real world. Some predict that this evolution is ultimately leading to a world of pervasive computing [8], where information & communication technologies are ubiquitously integrated into our environment. In this scenario, people will require the ability to perform tasks using any device at hand regardless of whether the device is suitable for the task [4][5][7].

With a wide variety of devices, each with varying constraints, it would be quite difficult to develop services that would operate in a uniform manner across all platforms. With this in mind we offer an agent based solution. Agent attributes of autonomy, adaptability and mobility make them highly suitable for use in pervasive computing. Agent Chameleons[3][1] illustrates a framework for Virtual Personal Assistants (VPA) for the information age.

A VPA would offer their user guidance through the technological quagmire that is cyber space – a realm that is ever changing, ever growing and ever becoming more difficult to navigate. In this task, these agents are facilitated through their ability to opportunistically migrate, through judicious selection (based on their tasks at hand), to different platforms and avail of the individual, and possibly unique, capabilities of the various platforms. This is useful from the point of view that the agents offer a uniform interface to any number of devices or services; also, for example, agents are very useful in load balancing where they can take resource intensive task off a resource constrained platform and perform it on a more suitable device. The following briefly outlines some of the technologies and concepts that underpin the functionality of a VPA.

2 Agent Chameleons

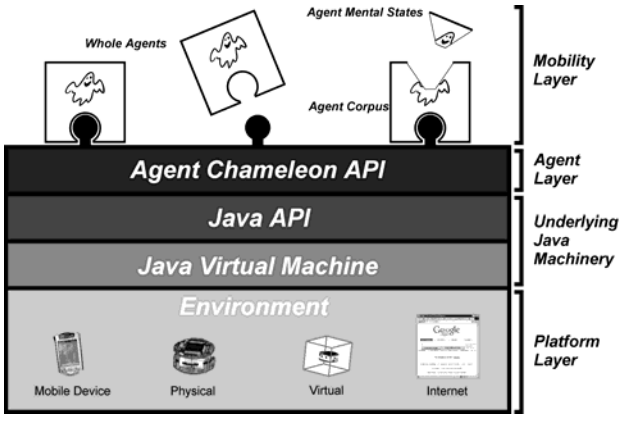

Figure 1. Agent Chameleon Architectural Strata

The Agent Chameleons project extends the traditional notions of an agent environment and its constraints by expanding through mobility/migration and mutation to virtual environments (i.e. avatar), physical environments (i.e. robot), and software environments (i.e.

[1] The Agent Chameleons framework is build upon Agent Factory [1] a FIPA compliant agent platform developed at UCD.

OS desktops, PDA's) (see Figure 1). This capacity to change the context of the agent's actions as it migrates necessitates a new approach to the traditional interpretations of how the environment affects the reasoning mechanisms of the agent.

2.1 Agent Architecture

The architecture of the agents is based upon the Social Robot Architecture (SRA), work previously carried out by one of the researchers [2]. Like the SRA, "a modular structure is used to divide the levels of complexity into incremental functionality … More abstract levels provide increasing complexity and subsume lower level functionality. Reactive or reflex survival behaviours are implemented at the reactive level with more complex behaviours defined within the deliberative level" [2]. This architecture is illustrated in Figure 2.

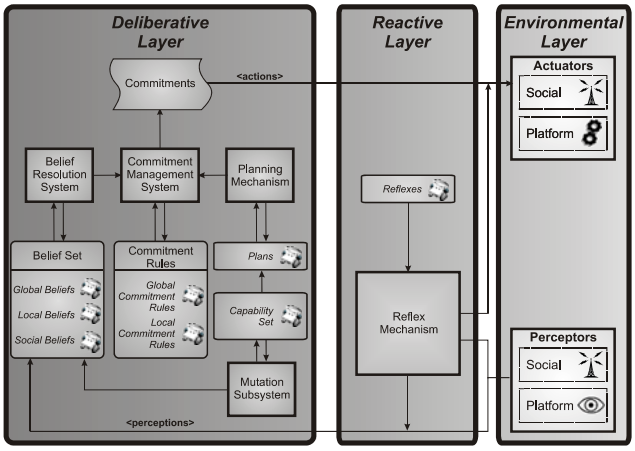

Figure 2. Agent Chameleon Architecture

The agent architecture is comprised of three layers – Environmental, Reactive and Deliberative. The modular design facilitates reflex behaviours for unexpected or dangerous events that require immediate reaction. Any planning would impede any real-time response required for these reactive behaviours. Hence, there is a higher layer to deal with more deliberative requirements.

The Environmental Layer is responsible for the agent's low-level interaction with its environment. It consists of two types of structures – Perceptors and Actuators. Perceptors are responsible for the monitoring of the environment. They pass relevant information about it to the Reactive and Deliberative layers. On the other hand, Actuators are used to affect the environment and are triggered by information from the Reactive and Deliberative Layers. Perceptors and Actuators are each further decomposed into social and platform. Platform Perceptors and Actuators provide an Agent-Software (i.e. interaction with other software) and Agent-Hardware (i.e. interaction with platform hardware) Interface. Social Perceptors and Actuators provide an Agent-Agent interface (i.e. communication with other agents).

The purpose of the Reactive Layer is to eliminate reliance on deliberation mechanisms for all agent actions, a classical failing of purely deliberative architectures. A series of basic reflexes empower the agent with a collection of survival instincts. Generally, these behaviours are dependent on the agents environment, e.g. if the agent was in control of the physical robot it would have collision avoidance behaviours. However, there are some behaviours that would be independent of the platform, e.g. if the agent environment became unstable it would attempt to protect itself through migration. After the Reactive Layer performs an action it informs the Deliberative Layer accordingly. In the event that the Reactive Layer is unable to deal with a situation, the problem is passed on as a priority to the deliberative mechanisms.

The Deliberative level provides the necessary functionality to deal with complex tasks that the agent will be required to perform. In order to achieve deliberative proactive agents we use the Belief-Desire-Intention (BDI) methodology. Agents are equipped with beliefs about their environment; such as what type of environment it is (e.g. robot, virtual environment, PDA, internet) and what the agent can achieve within this environment. In addition agents are equipped with beliefs about other environments, what constraints are in those other environments and whether they are capable of migration to those environments. A series of commitment rules help to drive the agents towards their goals. The mechanisms employed to maintain consistency across platform migration are based on a functionality set with active and inactive components depending on the instantiation. This facilitates the knowledge set of the agent in choosing possible body instantiations for particular problem sets.

3 Facilitating Virtual Personal Assistants

3.1 Agent Platforms

Software agents, on no matter what platform, can be seen as virtual entities based on digital devices. As a result, all input/output and processing is through digital means; hence, it is not required that they be fixed to one platform, as long as the range of platforms they may "inhabit" supports these computational entities.

We can assume that these agent platforms are installed on the devices by a human. To suggest otherwise would mean implying that an agent could seek out devices

on a network and install itself at will regardless of the wishes of the owner of such a device – this in turn alludes to the concept of a super virus. Such agents could also be achieved using a standard level of support across all operating systems – a concept that would create a security nightmare. In assuming the mediation of a human user in agent platform proliferation we can also assume that this person could enter in some basic information about the device they are using. This information, combined with the information the agent can derive from the device, can be used to construct a profile based on a device ontology.

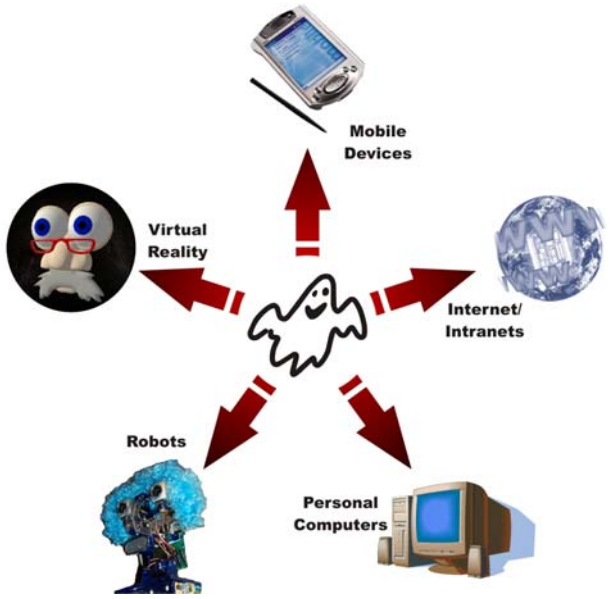

Figure 3. The Agent Chameleon Spirit and its Body Instantiations

Within a pervasive computing realm, there is generally a strong heterogeneity between the devices, and as a result there will be, relatively speaking, resource constrained devices. Information on individual devices will be contained in their respective device profiles. Such information includes: processor type, memory, connection description, user interfaces and any other relevant to the facilitation of agents as they exploit device diversity. The information in the profile will determine the complexity of the agent platform to be installed on the device. The profile will then contain information about the agent runtime environment on the platform. Also, platforms and agents will have access to a central module repository and, based on the device profile, the platform will be able to download components required to avail of the devices specific capabilities.

3.2 Agents, Cloning & Migration

In order to facility movement between diverse platforms, Agent Chameleons uses weak migration. In this form of migration, agents create suspended clones on destination devices, when the original is destroyed on the source device, the agents execution is resumed on the destination. Generally complete migration only occurs either at the request of the user (for example, in order to have the agent on the device in the users current context), or, as an act of self preservation (for example, when the platform where the agent currently resides becomes unstable or when power levels are critical on mobile devices). In most cases the performance of tasks will be through cloning. If a user requests a service from the agent, and the agent cannot perform the task using the current device it will create a clone at the required resource without destroying itself. This enables the agent to stay with its user and react to their needs while its clone performs the task; it also enables seamless task execution regardless of context. Clones are generally useful when there any tasks that need to be, or can be, performed concurrently.

Agents are created with unique IDs. Agent's clones are distinguished based on a timestamp combined with a parent platform ID. When a clone's usefulness has run out the relevant information contained within the clone will be integrated into the original agent, or into the oldest available clone, and the clone will be destroyed.

Figure 4. The Agent Chameleon Spirit and its Body Instantiations: Mobile Devices (PDA), Robot, PC, Web, and Virtual Reality Respectively

Agents operating in networks of extremely heterogeneous devices will have to be able to adapt to different operating conditions and device capabilities. This adaptation is performed through "elastic mutation" (i.e. a lossless adaptation of agents where any mutation state can be reached from any other mutation state). If the device is constrained in a manner that would not support an agent complete, the agent would be able to eliminate elements that are non-essential to its task at hand and scale down its capabilities. On platforms with minimal resources the agent may just send a small part of itself, which is required only to its current task, while, in the

case of complete migration, the bulk of the agent can remain passive on the source device.

Thus Agent Chameleons introduces a new breed of reasoning computational entity that is able to migrate, mutate and evolve on its journey between real and digital information spaces.

3.3 Human-Agent Interaction

Communication is vital with any personal assistant. As a result of the varying capabilities of devices, communication between these agents and their users must be multi-modal (that is, through speech, GUI, text, or, sometimes, not at all). From an agent's perspective, humans can be viewed in the same manner as other agents, however with special rights and privileges. In viewing humans as agents, interaction can occur through an agent communication language (ACL), although not directly. Agent social perceptors are used to translate speech, text, GUI based commands into an ACL act that the agent can understand, and similarly, social actuators convert ACL acts into text or speech that the human user can understand.

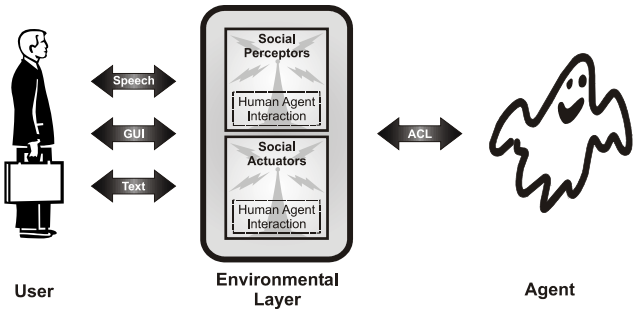

Figure 5. Human-Agent Interaction via the Environmental Layer

Communication with the agent is more that just a means of entering commands or getting results, prolonged interaction with the agent will facilitate familiarisation mechanisms through which the agent will learn about the user and thus develop a personality of its own to suit the needs of their user. These concepts extend the keyboard and mouse paradigm to a notion of *personable computing*.

3.4 Many Agents Many Platforms

Up until now we have dealt with only one agent that is tied to one particular user. In the real world there are many users and many more devices and agents. In order for an agent to be as useful as possible, on top of everything else, it will have to interact with other agents and utilise resources outside those controlled by its user.

When a user installs a platform on a device, by default, it will be indicated as a private resource for the user's agent. However, in business, or even family, environments there will be resources required by multiple people, and their associated agents. Akin users in operating systems, different agents will have different access rights on different platforms, determined by the platforms human owner. There will also be publicly available resources, such as collaborative virtual environments, that will require an agent to leave its 'safe' domain. In these regions of cyberspace agents can be isolated, have their internal states examined and/or altered and this can pose additional problems (issues, such as mobile agent security, that are beyond the scope of this article).

The nomadic nature of the agents is assisted by the use of 'white and yellow pages' for locating people and services. Thus the agents would have access to directory services that allow them to access publicly available resources and other agents. The agents would also have a private directory for accessing resources specific to its owner and not publicly available.

4 Conclusions

The roles of human personal assistants (PA) have changed over the years in conjunction with the integration of computers in the work environment. A PA's boss, with the aid of a computer, would now do a number of tasks for themselves traditionally performed by a PA. At the same time PAs have moved more toward people oriented tasks (such as greeting people, answer phones, etc) areas where current technology would not be suitable use. In the future these person-to-person duties could be subsumed by technology but these tasks would have to be carried out with a very high resolution before they would be acceptable to clients/customers.

The current notion of the personal digital assistant is that of the physical device, through Agent Chameleons we extend this notion beyond that of just the gadget to the concept of a virtual personal assistant that is independent of any one physical device. These entities will effectively give anyone there own personal assistant that will help with the information overload in daily life, assists with personal communications and offer a generic interface to any number of devices. They will have the ability react to the current needs of there user, and beyond this they will grow and learn to anticipate future needs and requirements.

Acknowledgements The work undertaken as part of the Agent Chameleons project (http://chameleon.ucd.ie) a collaborative project undertaken between the Department of Computer Science, University College Dublin (UCD) and Media Lab Europe (MLE), Dublin. We gratefully acknowledge the financial support of the Higher Education Authority (HEA) Ireland and the Irish Research Council for Science, Engineering and Technology: funded by the National Development Plan. Gregory O'Hare gratefully acknowledges the support of Science Foundation Ireland under Grant No. 03/IN.3/1361.

References

[1] R.W. Collier, *Agent Factory: A Framework for the Engineering of Agent-Oriented Applications,* Ph.D. Thesis, Department of Computer Science, University College Dublin (UCD), Ireland, 2001.

[2] B.R. Duffy, The *Social Robot*, PhD Thesis, Department of Computer Science, University College Dublin (UCD), Dublin, Ireland, 2000.

[3] B.R. Duffy, G.M.P. O'Hare, A.N. Martin, J.F. Bradley and B. Schön, "Agent Chameleons: Agent Minds and Bodies", Proc. 16th International Conference on Computer Animation and Social Agents (CASA 2003), Rutgers University, New Jersey, May 7-9, 2003.

[4] P. Horn, "Autonomic Computing: IBM's Perspective on the State of Information Technology", IBM Research, Autonomic Computing Manifesto, International Business Machines Corporation, 15th October 2001 http://www.research.ibm.com/autonomic/manifesto/autonomic_computing.pdf, 2001.

[5] A. Messer, I. Greenberg, P. Bernadat, D. Milojicic, D. Chen, T.J. Giuli and X. Gu, "Towards a Distributed Platform for Resource-Constrained Devices", Proc. IEEE 22nd International Conference on Distributed Computing Systems (ICDCS'2002), pp. 43-5, 2-5, Vienna, Austria, July 2002.

[6] L. Steels, "Engeln mit Internetfluegeln" (German version of Digital Angels), In: Die Gegenwart der Zukunft, p90-98, Verslag Klaus Wagenbach, Berlin, 2000.

[7] D.L. Tennenhouse, "Proactive Computing," *Communications of the ACM*, Vol 43, No. 5, pp. 43-50, May 2000.

[8] M. Weiser, "The Computer for the Twenty-First Century", *Scientific American*, p94-100, September 1991.