Chapter 2

Mixed Reality Agent (MiRA) Chameleons

Mauro Dragone, Thomas Holz, G.M.P. O'Hare and Michael J. O'Grady

CLARITY Centre for Sensor Web Technologies, School of Computer Science & Informatics, University College Dublin, Belfield, Dublin 4, Ireland

{*mauro.dragone, thomas.holz, gregory.ohare, michael.j.ogrady*}*@ucd.ie*

Abstract

Human-Robot Interaction poses significant research challenges. Recent research suggests that personalisation and individualisation are key factors for establishing lifelong human-robot relationships. This raises difficulties as roboticists seek to incorporate robots into the digital society where an increasing amount of human activities relies on digital technologies and ubiquitous infrastructures. In essence, a robot may be perceived as either an embedded or mobile artefact in an arbitrary environment that must be interacted with in a seamless and intuitive fashion. This chapter explores some of the alternative ways and the design issues to achieve these objectives. Specifically, it describes a new system, which we call Mixed Reality Agent (MiRA) Chameleon, that combines the latest advancements on agent-based ubiquitous architectures with mixed reality technology to deliver personalised and ubiquitous robot agents.

2.1 Introduction

The diffusion of robotic platforms into our daily lives involves many new design challenges. The fact that we are all individuals influences our expectations and design requirements for those tools and systems with which we interact. Robots are no different in this respect. They need to operate in our daily life environments, such as hospitals, exhibitions and museums, welfare facilities and households. Not only will these robots have to deal with various complicated tasks, but they are also expected to behave in a socially intelligent and individualised manner to meet the diverse requirements of each user [Dautenhahn (1998)]. For these reasons, personalised, custom-made robotic design is one of the technology strategies advocated by the Japan Robot Association (JARA) for creating a "Robot Society" in the 21st Century [JARA (2008)].

The other challenge faced by today's roboticists is the integration of robots into digital society, as an ever-growing amount of human activities relies on digital technology. Trends such as inexpensive internet access and the diffusion of wireless computing devices have made ubiquitous or

E. Mangina et al., *Agent-Based Ubiquitous Computing*, Atlantis Ambient and Pervasive Intelligence 1,
DOI 10.2991/978-94-91216-31-2_2, © 2009 Atlantis Press/World Scientific

pervasive computing a practical reality that augments the normal physical environment and supports the delivery of services to human users anytime and anywhere. Endowing these ubiquitous devices with intelligent behaviour, and thus creating intelligent environments, is termed Ambient Intelligence (AmI) [Aarts (2004)].

However, while many robotic initiatives now share the thesis that robots are a compelling instance of those artefacts which comprise and deliver the ambient space, reconciling the personalization/social aspect with pervasiveness and ubiquity - e.g. through the integration with existing ubiquitous infrastructures - still remains a largely unexplored area of research. On both fronts, user interface agents, e.g. acting as virtual assistants to their user, have been already widely adopted as intelligent, adaptive social interfaces to the digital world, e.g. in form of virtual characters interacting with the user via PCs, PDAs, and the Internet. As such, the experience accumulated it these applicative domains may be purposefully used to inform robotics' research. While in the past, software agents and robots have usually been attributed to distinct domains, software and hardware respectively, the modern conception is, in fact, to consider them as particular instances of the same notion of agent - an autonomous entity capable of reactive and pro-active behaviour in the environment it inhabits.

In addition, one of the foremost aids to the design of both modern robot and ubiquitous/pervasive systems comes from the developments in agent-oriented software engineering (AOSE). AOSE promotes the designing and the developing of applications in terms of multiple autonomous software agents for their characteristic ability of delivering context-sensitive, adaptive solutions. As such, multiagent techniques, related software engineering methodologies, and development tools are natural candidates for the implementation of these systems.

The work described in this paper is representative of the of the kind of multi-disciplinary effort characterising today's agent research. This paper describes the MiRA (Mixed Reality Agent) Chameleons project (Figure 2.1), which leverage on past experiences and agent-related technologies to build socially competent robot agents by combining physical robotic bodies with virtual characters displayed via Mixed Reality (MR) visualization. Through such an innovative integration approach, a MiRA Chameleon exhibits tangible physical presence while offering rich expressional capabilities and personalisation features that are complex and expensive to realise with pure hardware-based solutions. Crucially, thanks to its AOSE methodology, it also promises to maintain all the expressivity of the MR medium without loosing the tremendous opportunities for ubiquity and adaptation associated with its virtual component.

Fig. 2.1 Live demonstration of MiRA Chameleons at the UCD Open Day, December 2007 (actual images displayed in the user's head mounted display during live experiments and user trials with our application).

Before proceeding with the rest of the paper, though, it is necessary to reflect further on the intelligent agent paradigm.

2.2 Social interface agents

AmI was conceived in response to the observation that many embedded artefacts all competing for the user's intention would result in environments that were uninhabitable. Agent-based intelligent user interfaces are seen as key enabling technology for AmI, and a solution to the interaction issue.

Much of the effectiveness of these interfaces in general stem from their deliberate exploitation of the fact that end-users consistently assume, if only on a subconscious level, an intentional stance toward computer systems [Friedman (1995)].

Affective Computing is a branch of HCI that aims to endow computer systems with emotional capabilities (for example, to express, recognize, or "have" emotions) in order to make them more life-like and better adapted to interact with humans. The natural and more recent evolution of intelligent agents has resulted in the social agent paradigm, that is, the exploitation of social and psychological insights on human social behaviour to create agents that are capable of acting as more equal social partners, while also developing and maintaining long-term social relationships with humans. The investigation of this type of human-agent social interaction is becoming increasingly important, as both software agents and robots are gradually populating the human social space.

Fong et al. [Fong *et al.* (2003)] use the term "socially interactive robots" to describe robots for which social interaction plays a key role. For Fong et al. these robots are important in domains where robots must exhibit peer-to-peer interaction skills, either because such skills are required for solving specific tasks, or because the primary function of the robot is to interact socially with people, e.g. as in robot companions.

Among the requirements for some these applications is that of being persuasive [Fogg (1999)], that is, changing behaviour and attitudes of the humans they interact with, and also being capable of building and maintaining long-term relationships with them. Acceptance and success of such a system, and its utility in real-world applications, depends upon the robot's capacity to maintain relationships based upon believability, credibility and trust. In addition to improving their effectiveness in working with humans, robots can also harness social interaction for learning practical skills, for example, through imitation [Billard and Dautenhahn (1999)].

While all these applicative scenarios suggest different degrees of autonomy and intelligence (skills like mobility, grasping, vacuuming and object recognition), they also open up the common question of what specific capabilities are necessary for a robot to socially interact with humans and what the general issues imposed by such a new social context are. Crucially, the same objectives are addressed in software-only domains, for example, in supporting the development of virtual characters that assist HCI, for example, as game opponents or personal assistants (see [Bartneck (2002)] for a survey).

Human-like virtual characters (virtual humans) are being used with success as virtual representatives of human users in virtual conference applications [Slater *et al.* (1999)], or as fully autonomous agents inhabiting virtual worlds to enhance the user's experience and ease his interaction with the virtual world. Such characters can make the interaction more engaging and make the user pay more attention to the system, e.g. in educational and training applications [Chittaro *et al.* (2005)]. A much appreciated feature in the latter type of applications is that virtual humans can provide pedagogical assistance that can be tailored to the needs and preferences of the learner [Baylor *et al.* (2006)].

Studies focusing on how the appearance of virtual characters can affect cooperation, changing attitudes, and motivating users [Rickenberg and Reeves (2000); Zanbaka *et al.* (2007)] indicate that humans treat them as social partners and, in particular, that many of the rules that apply to human-human interaction carry over to human-agent interaction.

The result is that, despite technical and methodological differences between dealing with robotic and software/virtual domains, today a large number of issues behind the construction of successful social agents cross the boundaries of agent species.

What distinguishes all the research in socially intelligent agents is the emphasis given to the role of the human as a social interaction partner of artificial agents and, subsequently, to the relevance attributed to aspects of human-style social intelligence in informing and shaping such interactions. The consensus in social agent research is that effective human-agent interaction greatly leverages the instauration of a human-style social relationship between human and agent.

Effective social interaction ultimately depends upon the recognition of other points of view and the understanding of their intentions and emotions [Dautenhahn (1997)]. Humans convey this information through explicit communication (natural language) as well as an array of non-verbal cues [Kidd and Breazeal (2005)], such as tactile interaction, postures, gestures, and facial expressions. The latter are used both to convey emotions and also as social signals of intent, e.g. for regulating the flow of dialogue, while gestures are commonly used to clarify speech. Deictic spatial gestures (e.g. gaze, pointing), in particular, can compactly convey geometric information (e.g. location, direction) while also being a major component of joint visual attention between humans [Kidd and Breazeal (2005)].

2.3 Ubiquitous robots

Having reflected on how agents offer a suitable paradigm for realising intelligent user interfaces in general, the problem of effectuating an intelligent social interface paradigm in ubiquitous spaces can now be considered. In particular, the question of integrating robots into ubiquitous spaces and enabling effective Human-Robot Interaction (HRI) are explored.

Ubiquitous robots, that is, robots embedded in ubiquitous and intelligent environments, are the most recent class of networked robot applications motivated by the increasing interest raised by ubiquitous computing scenarios. The role of robots in this context is double faceted: first, toward the users, the robot is seen as one of the many arms of the environment, providing them with the services they need, anytime and anywhere [Ha *et al.* (2005)]. Second, from the robot perspective, this also implies the augmentation of the robot's capabilities, which are extended with the services provided by the ubiquitous environment but also by virtue of sharing an information network with the user.

The Ubiquitous Robotic Companion (URC) project [Ha *et al.* (2005)] provided one of the first examples of mass deployment with roughly one thousand URCs distributed to households in Seoul and the vicinity where they guarded the home, cleaned rooms and read to the kids. Other notable projects include the UbiBot [Kim (2005)] system, developed at the Robot Intelligence Laboratory KAIST, Republic of Korea, the Agent Chamaleon project [Duffy *et al.* (2003)], developed at University College Dublin (UCD), Ireland, and the PlantCare [LaMarca *et al.* (2002)] system developed by the IBM's autonomic computing group. Both NEC's robot PaPeRo^{TM}(Partner-Type Personal Robot) [Fujita (2002)] and the iCat robot [van Breemen *et al.* (2005)], developed by Philips Research, are good examples of robots designed to act as an intelligent interface between users and network appliances deployed in ubiquitous home environments. They can, for instance, check the user's e-mail, tune the TV onto the user's favourite channel, and access the Internet for retrieving stories to narrate to the children.

In its child care version, PaPeRo^{TM}also presents some social attributes thanks to the advanced interaction abilities enabled by an array of sensors, including touch sensors, sonar, directional microphones, and cameras. With these sensors, PaPeRo^{TM}can act as a teacher by locating, identifying, taking attendance, imparting lessons, and quizzing its child students.

The iCat robot, a research prototype of an emotionally intelligent robot that can provide an easy to use and enjoyable interaction style in ambient intelligence environments, is also used to investigate

the social interaction aspect. In particular, iCat can communicate information encoded in coloured light throgh multi-color LEDs in its feet and ears, speak natural language, and also use facial expression to give emotional feedback to the user. One of the key research questions investigated within this project is to find out whether these expressions and capabilities can be organised to give the robot a personality, and which type of personality would be most appropriate during interaction with users.

These studies have shown that mechanically-rendered emotions and behaviours can have a significant effect on the way users perceive – and interact with – robots. Moreover, they have also shown that users prefer to interact with a socially intelligent robot for a range of applications, compared to more conventional interaction means.

2.3.1 *Augmented HRI and immersive interfaces*

Much of the work in ubiquitous robotics focuses on control issues, e.g. by explicitly addressing the interoperability between heterogeneous hardware and software components, and on augmenting traditional HRI capabilities by leveraging on wireless networking and wearable interfaces. For instance, wearable RFID tags [Ha *et al.* (2005); Herianto *et al.* (2007)] have been proposed in conjunction with ubiquitous robotic systems to aid the detection and the location of users in the environment, as these devices are sensed by the ubiquitous infrastructure and the relevant information stored in it (user identity, location, and other information) is communicated to the robots. Notably, such an approach can substitute conventional (and more difficult to realize) robot-centric perception as even simple robots (e.g. with no camera) can effectively recognise and track the humans in their environment.

Direct wireless communication between robots and wearable setups can also be used in this sense, for instance, for overcoming the limitations of today's speech recognition systems. Users of PaPeRo [Fujita (2002)], for example, employ wireless microphones for cancelling the impact of changing background noise, while in [J. and A. (2002)] speech recognition is carried out on the wearable computer thus overcoming the robot's difficulty to recognize the voice of different users without training.

A particular approach is to use these techniques in conjunction with immersive interfaces. Immersive interfaces are the natural complement to ubiquitous spaces as they enable the visualisation of virtual elements in the same context in the real world.

The first immersive interfaces to virtual environments were invented by Sutherland [Sutherland (1998)] who built a Head-Mounted Display (HMD), a goggles-like displaying device, that, when worn by the user, replaces the user's normal vision of the world with computer-generated (stereoscopic) 3D imagery. By also availing of positioning devices (e.g. magnetic) for tracking the 3D pose of the HMD, it is then possible to modify the rendering perspective of the computer-generated imagery in order to match the movement of the user's gaze and, subsequently, to give the user the impression of being immersed in a 3D graphical environment.

More generally, by using see-through HMDs and availing of AR technology, human users can be immersed in a shared space obtained by superimposing virtual elements that are rendered in the same perspective as the real scene. This form of supplementing the physical space with digital information and augmenting interaction capabilities is what Young & Sharlin call a MR Integrated Environment (MRIE) [Young and Sharlin (2006)].

Traditionally, these techniques have been used in robotics in the context of tele-presence systems to enhance the user's understanding of the remote environment in tele-robotic systems (e.g. [Milgram *et al.* (1995)]). More recently, however, there have been a few applications involving humans working side by side with mobile robots.

For [Collett and MacDonald (2006)], who implemented an AR visualization toolkit for the robot device server Player [Gerkey *et al.* (2003)], AR provides an ideal presentation of the robot's world-view, which can be very effective in supporting the development of robot systems. Specifically, by

viewing the data (e.g. sensor data such as sonar and laser scans) in context with the real world, the developer is able to compare the robot's world view against the ground truth of the real world image thus promoting a better understanding – and consequent evaluation – of the inner workings of the robot control system. As such, AR/MR visualisation can also be used as natural complement of hardware in the loop simulations.

Stilman et al. [Stilman *et al.* (2005)], for example, instrumented a laboratory space with a motion capture system that tracks retro-reflective markers on robot bodies and heads and other objects such as chairs and tables. By combining this ground truth with virtual artefacts, such as simulated robots or simulated objects, a hybrid real/virtual space is constructed that can be used to support simulated sensing. Through HMDs, researchers can visualise real or simulated robots during walking tests among both real and virtual obstacles. In addition, they can also avail of virtual artefacts, such as the visualisation of the robot's intended footsteps on the floor, to aid their judgment of the robot's sensing and path planning capabilities.

These applications are often realised with mobile AR systems [Feiner *et al.* (1997)], which enable the combination of AR immersive visualisation and user mobility. Compared to stationary computer systems, portable solutions enable human operators to gain an understanding and evaluate the state and the intentions of a robot system while moving around in the same environment that the robot is operating in.

Giesler [Giesler *et al.* (2004)] developed a mobile AR system for control of an autonomous mobile platform. The system combines robot self-localisation with the localisation of the HMD-wearing user in the same frame of reference. The position and orientation of the user's HMD is found by means of multiple fiducials distributed in the working area, which are then recognised and tracked through the ARToolkit software library [Kato and Billinghurst (1999)]. The user is also equipped with a cursor, in the form of a pen terminating in a cube formed by ARToolkit fiducials. Since the orientation of the cube is visible from every angle, the user can point to imaginary spots on the floor, just like moving a mouse on a desktop computer. Giesler's application enables the rapid prototyping and manipulation of topological maps that are interactively defined in the working environment. Consequently, the user can instruct the robot with commands such as "go to this node".

Finally, Daily et al. [Daily *et al.* (2003)]. describe an AR system for enhancing HRI with robotic swarms, e.g. in search and rescue scenarios. Their system is highly specialised toward a very interesting but narrow class of applications (search and rescue) as the particular solution adopted for tracking and data communication (optic-based with no networking) suffers of obvious bandwidth limitations. However, their application is very interesting as it shows how AR can enhance the robot's interaction capabilities – in this case by allowing each robot in the swarm to point to the direction of the victim without the need of a physical component (e.g. an arm).

2.4 Ubiquitous agents

Having reviewed the major issues confronted by projects involved in the integration of robots within ubiquitous environments, we can now have a look at what has been in done in the HCI front. In particular, this section will focus on works that have tried to deploy social interface agents in ubiquitous settings.

The availability of agent platforms for light and embedded devices such as PDAs and cellular phones has already provided an opportunity for the deployment of a large number of interface agent applications in the mobile and ubiquitous sector [Hristova *et al.* (2003)]. Often, the effectiveness of these systems depends on the ability of software agents to take into account the state of the physical environment where their components are deployed, as well as the state of their human users. Furthermore, their ability to migrate between devices in the network and adapt to different circumstances, e.g. different computational power or different interface capabilities, makes software agents the nat-

ural candidates for maintaining the connection with the user through his daily activities [Barakonyi *et al.* (2004)].

However, to date, there are not many examples of virtual characters being used in these applications. The few projects that do [Gutierrez *et al.* (2003)] suffer of limitations that are inherent in their screen-based representation. It is not possible for the agents to leave the screen and wander freely or to interact with physical objects in the real world. In such a context, and with the divide between their 2D screen embodiment and the 3D physical world, it is also more difficult for them to attract and engage the user. For example, they cannot effectively point (or look) in a 3D physical surrounding. Hence, a different, and more popular, approach is to enable virtual agents to share the same 3D space as humans by employing immersive user interfaces.

The easiest implementations of such systems employ table-top AR platforms, such as Magicbook [Billinghurst *et al.* (2001)] and TARBoard [Lee *et al.* (2005)], in which the user's HMD pose is tracked only within a limited range. Other applications are instead confronted with mobile AR systems. Notable implementations include pervasive game environments, such as ARQuake [Thomas *et al.* (2000)] or AquaGuantlet [Tamura *et al.* (2001)], both of which enable multiple players to fight against virtual enemies added to the real environment through AR visualisation.

In both table-top and mobile AR settings, users can interact with virtual conversational agents, through speech or tangible user interfaces, e.g. by manipulating objects whose 3D pose is also tracked. These systems constitute suitable frameworks for investigating face-to-face human-agent social interaction. The conversational agent Welbo [Anabuki *et al.* (2000)], for example, is an interface agent that guides, helps, and serves the users in an "MR Living Room" where the users can visually simulate the location of virtual furniture. Pedagogical Embodied Conversational Agents (PECA) [Doswell (2005)] are similar virtual agents that apply proven pedagogical techniques to interact with human learners in various learning scenarios, including outdoor student tours of historical buildings.

For this purpose, an increasing number of these systems are employing sophisticated agent control architectures. In particular, the Nexus [O'Hare *et al.* (2004)] and the UbiAgent [Barakonyi *et al.* (2004)] frameworks demonstrate how virtual agents equipped with BDI (Belief, Desire, Intention) control systems can provide the reasoning apparatus for creating believable characters that are responsive to modifications and stimuli in their environment, but are also proactive and goal-oriented. Both systems demonstrate the capabilities of creating applications (e.g. supporting collaborative scenarios) with multiple virtual agents and multiple users, where virtual agents posses a high degree of autonomy and can watch the MR space, in particular by sensing each other's relative positions as well as the movements of other physical objects. In addition to position-tracking devices, these systems may also employ other sensors, such as light and temperature sensors for gathering information about the physical environment in which they are visualized.

2.4.1 *Discussion*

The review reported in the last two sections helps to highlight a convergence between some of the agent applications produced in both the robotic and virtual domain. From the virtual domain, ubiquitous agents such as in UbiAgent [Barakonyi *et al.* (2004)] and Nexus [O'Hare *et al.* (2004)] (see Section 2.4) help virtual characters escape purely virtual environments and socially engage humans in their physical environments, especially when they employ MR technology. From the robotic domain, ubiquitous robots and augmented HRI systems (with or without AR visualisation) are motivated by the difficulty of creating autonomous robot agents that sense and act in the physical world.

Independently from the genre, either robotic or virtual, the construction of agents operating in the ubiquitous space offers some characteristic engineering challenges. For example, their effectiveness requires a geometric correspondence between virtual and real elements of the ubiquitous space,

which can usually be obtained by employing dedicated localisation mechanisms. Notably, however, geometric correspondence represents only a basic requirement in these types of applications. The hybrid model constructed within the hybrid simulation system [Stilman *et al.* (2005)] reviewed in the previous section is a good example of ubiquitous space that can be thought as the result of the superimposition of two sub-spaces, namely: the real and the virtual sub-space. A more sophisticated system that allows for scenarios involving tactile interaction would necessarily pose the additional constraint of physical realism (e.g. impenetrability, gravity, etc.), in order to recreate the same robot-environment dynamic of real experiments.

On the other hand, in order to be employable as effective social characters in real environments, virtual characters, in systems such as Nexus and UbiAgent, need to perceive not only the position (for gaze behaviour [Wagner *et al.* (2006)] [Prendingera *et al.* (2004)] and the placement of the virtual agent [Anabuki *et al.* (2000)]) but also the state of the humans they interact with [Klein *et al.* (1999)], e.g. their moods, given, for example, by the state of the conversation and their gestures. Since these agents, in contrast to robots, lack a physical body they need to cooperate with a ubiquitous infrastructure, e.g. wireless sensor networks including cameras and localisation sensors deployed in the environment where human-agent interaction takes place.

The source of most of the difficulties in engineering agent systems present in the ubiquitous environment, compared, for example, to purely real or purely virtual scenarios, can be explained in terms of agent embodiment, intended as the structural coupling between the agent and its environment (see [Ziemke (2003)]). If we want to embody an agent in such an environment, its virtual and its real sub-space need to be structurally coupled. If this consistency between the two sub-spaces is not properly engineered, the agent's interaction with one sub-space would not be affected by the actions of the agent with the other sub-space, which will violate the requirement for structural coupling between the agent and its environment.

The other issue to be resolved for embodying social interface agents into ubiquitous settings is to decide the type of their embodiment. While a lot of studies that compare physical and virtual agents credit the robot with better results in a lot of areas (e.g. credibility, social presence, social awareness, enjoyment, engagement, persuasiveness, likeability), there are others that indicate that some of these results could originate from the nature of the task investigated. For example, Yamamoto et al. [abd K. Shinozawa *et al.* (2003)] have compared the influence of a robot and a virtual character on user decisions in a recommendation task. They carried out identical tasks, once with virtual objects on a computer screen and once with physical objects in the real world, and have found that the virtual agent scores better in the virtual world and the physical agent scores better in the real world. These experiments suggest that the nature of the task has a big influence on which type of embodiment is appropriate for the agent.

While there are other works showing that an AR virtual character can be the medium to make the user interact with the AR space, e.g. to move virtual furniture around the real room in Welbo, or architectural 3D models in PECA, an AR character still lacks the capability to physically affect the real world.

Consequently, an agent that wants to be truly ubiquitous and assist the user in both real and virtual tasks needs to be able to assume both physical and virtual embodiment forms. But to maintain an individualised and personal relationship, the agent has to also display the same persona/identity to the user. A possible solution to this conundrum is the concept of dynamic embodiment which will be explored in the next section.

2.5 Dynamic embodiment

As discussed in the previous section, an agent can only be truly ubiquitous if it can assume a physical and a virtual body, depending on the task. The identity problem, on the other hand, demands that

the agent projects the same (or, at least, a recognisably similar) persona to the user. So what is really needed is a form of dynamic embodiment, whereby the agent can change its embodiment form to fit the task at hand.

Our idea of dynamic embodiment is closely related with the concepts expressed in the work of Kaplan [Kaplan (2005)], in which he observes how it is often possible to separate a robot software system from the robot hardware. Modern robot architectures (e.g. see [Dragone (2007)] for a review) already adhere to the common software engineering practice of placing a middleware layer between application and operating system. This middleware layer is mainly concerned with enabling access to the system's hardware (e.g. through interfaces/drivers to sensors and actuators) and computational resources, resolving heterogeneity and distribution and also automating much of the development of new systems by providing common, re-usable functionalities.

While traditional robot middleware enable the rapid configuration of the software to support new robot platforms or hardware layouts, these are essentially off-line integration features. Kaplan's work, instead, refers to the concept of agent migration, i.e. the ability to move the execution of software agents across the network. Under this perspective, a robot can be considered as a software agent in charge of a robotic platform.

Kaplan proposes to use the term teleportation to refer to migration between two identical bodies (e.g. two different robots of the same model/composed of the same hardware components), and metamorphosis to refer to the transfer of a software agent migrating between two non-identical bodies (e.g. a personal robot and a PDA). Using teleportation and metamorphosis, software agents controlling robots can change their body in order to find the most appropriate form for any given situation/task. Physical robots then become another observation point, from which software agents can follow the user in his/her activities and movements.

This also increases the number of possible agent-user interaction scenarios, which can include a variety of real world situated interaction, supporting, for example, applications in which the agent needs to learn to adapt to the user. A notable example of mutation and metamorphosis in agent-based ubiquitous robots is investigated within the Agent Chameleons project [Duffy *et al.* (2003); Martin *et al.* (2005)], which will be considered shortly.

2.5.1 *Agent chameleons*

The Agent Chameleons project investigates agent teleportation and metamorphosis in the context of personal assistant agents helping the user across different interface devices and even crossing the software/hardware divide.

Agent Chameleons is modelled on the idea of realising a digital friend that will evolve a unique individual personality through prolonged interaction with the user. Such a trait tries to ensure agent-person familiarisation over time and across platforms (dimensions/information spaces).

In addition to agent teleportation, aimed at acquiring different operational functionalities, the other concept explored within the Agent Chameleons project is the use of agent mutation (Kaplan's metamorphosis) to alter the agent's external appearance in order to engender a sense of identity. This is motivated by the fact that embodiment is more than just the material body, but it also carries a gendered, social, political and economic identity. One of the most common psychological effects of embodiment is the rendering of a sense of presence [Biocca (1997); Sas and OHare (2001)]. Biocca and Nowak [Biocca and Nowak (1999b,a)] stressed the importance of body in achieving a sense of place, space and of another entitys presence.

One of the applications considered within the project is the development of personal travel-assistant agents. These can take the form of a virtual character operating on a personal computer that helps the user to find and book a flight to a certain conference, for example, in the style of Microsoft's Office Assistant. The agent may then migrate to the user's PDA so that it can effectively

travel with the user, helping with flight connections and attending tasks such as filtering e-mails or negotiating the transit of the user through the airport. Once at the hotel room, the agent may then leave its computational environment by migrating from the user's PDA to a computer in control of a service robot that is gracefully provided by the (futuristic) hotel. With its new physical body, the agent will then be able to continue assisting the user, this time by also using its physical capabilities.

Notably, this scenario is substantially different from the one in which the user is simply assigned a service robot on his arrival at the hotel. In the scenario foreseen by the Agent Chameleons project, the robot would effectively act as the user's personal assistant. By having access to his personal data, the robot would, for example, be able to serve the perfect coffee without having to query the user about his preferences. Such results may be achieved with the personal assistant agent communicating the necessary instructions to the robot. However, a solution based upon migration may be more efficient and would also offer the advantage of not having to release private user details. In addition, an Agent Chameleon goes a step further by making sure that the user knows that all the agents are a mutated form of his personal assistant. For instance, the Agent Chameleon will use the same synthetic voice or will display other characteristic personality traits that are independent from his specific bodily form, thus preserving the advantages of the familiar relationships between the user and its assistant.

Experiments have been undertaken that demonstrate the concepts of teleportation, and mutation of Agent Chameleons. Figure 2.2 illustrates the layers of the system architecture behind those experiments. At the base exists a middleware layer, interfacing with the specific environment inhabited by the agent. This may consist either of an API toward the functionalities of a specific application, in the case of the agent inhabiting a purely software environment, or of a hardware-control API, in the case where the agent possesses a robotic body. Above this is the native Java Virtual Machine (JVM). Beyond this resides the Java API and built on top of that is Agent Factory (AF) [O'Hare *et al.* (1998); Collier (2001)], a multiagent programming toolkit developed in UCD.

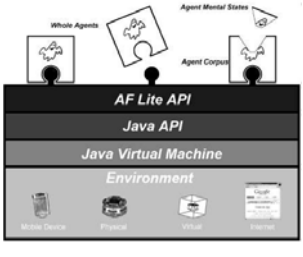

Fig. 2.2 Agent Chameleons system layers.

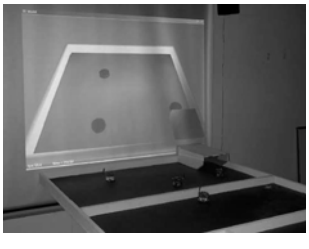

Fig. 2.3 Proof of concept migration between real and virtual world in Agent Chameleons.

Agent Chameleons realises a proof of concept demonstration of migration of a physical robot from the real world to the virtual and vice-versa (see Figure 2.3). In the specific demonstrative system the physical world is extended by a virtual world depicted on a computer screen adjoining the physical world. A small Kephera robot can navigate and explore the desk-mounted world and dock in a robot garage at the edge of the physical world thus removing the physical robot from vision. Thereafter the robot seamlessly crosses into the virtual world.

2.5.2 *Discussion*

The implementation of agent mutation in the demonstrative system realised within the Agent Chameleons project is limited in several aspects. First of all, it relies upon an existing isomorphism between the entities involved in the mutation, namely the software agents, the robots and their

respective environments (simulated vs. real). Second, both systems are kept very simple by using miniature Kephera robots for table-top experiments. As a result, mutation can be simply achieved by migrating the agent in charge of the simulated robot from the simulator to the real robotic platform. Even within such constraints, the main problem of this type of mutation is that the agent is oblivious to the type of environment it is operating in, e.g. ignoring if its perceptions are originated by the simulator or by physical sensing or if its actions are carried out by simulated or hardware actuators. This entails the assumption that the same agent is equally able to control both the simulated and the real robot, provided that the connections between its sensory-motor apparatus and the environment are appropriately rewired (to the simulator or to real robot hardware). Although such an assumption is admissible in the specific conditions posed by the demonstrative systems, it is an exception rather than the norm. A more effective implementation of the Agent Chameleon should be mindful of the differences between simulated and real world in order to better adapt to the different scenarios and different robots, but also benefit from the different possibilities awarded to the agent by each of its distinct forms.

In addition, such basic implementations of agent mutation do not address the more general case in which a generic agent – not a simulated robot – wishes to take control of a robotic platform in order to execute a task in the physical world. In particular, the truly interesting potential of the Agent Chameleon scenario is the ability to support individualised and possibly social interaction with the user in both real and virtual settings. However, the Agent Chameleon project lacks a test-bed to investigate these ideas.

Although it is possible to investigate the possibilities of agents mutating their form when embodied in a virtual environment [Martin *et al.* (2005)], they loose this capability in the physical world. A possible solution to these limitations would be to use a screen-based interface, e.g. as in the robot GRACE [Simmons *et al.* (2003)]. This would give a user's personal assistant agent, which assumes the form of a virtual character on the user's PC and PDAs, the possibility to appear on the robot's screen to signal that the agent has taken the control of it. The service robot in the hotel example discussed in Section 2.5.1 will then have the same appearance and behaviour of the service robot used by the user in his own home.

A second option, explored in the remainder of this chapter, is to use mixed reality as an interface between the Agent Chameleon and the end-user.

2.6 MiRA chameleons

Mixed Reality Agents (MiRAs) are an innovative class of applications which combine physical robots with virtual characters displayed through MR technology [Dragone *et al.* (2006); Young *et al.* (2007); Shoji *et al.* (2006)]. By giving a virtual interface to a physical robot and a physical body to a virtual character, the agents in these systems exhibit tangible physical presence while offering rich expressional capabilities and personalisation features that are complex and expensive to realise with pure hardware-based solutions. These systems constitute a characteristic example of dynamic embodiment in which the capabilities of both mediums are merged into a single agent.

An obvious disadvantage of employing such an augmentation approach is the cumbersome and expensive hardware imposed on each user, which at the moment clearly encumbers the deployment of MiRAs in applications with high user-to-robot ratio. However, this situation is on the verge of change as both head-mounted displays and wearable computers are becoming cheaper and less invasive.

Collectively, these systems showcase the advantages of employing MR visualisation to combine physical robot platforms with virtual characters. Among the possibilities, the virtual character can be overlaid as a form of virtual clothing that envelops the physical robot and acts as a visualisation membrane, de-facto hiding the robot's hardware [Holz *et al.* (2006); Shoji *et al.* (2006)]. Alternatively, the virtual character can be visualised on top of the robot, as a bust protruding from the robot's

body, or even figuring as the robot's driver [Holz *et al.* (2006)]. In every case, in contrast to robots with virtual characters visualised on a screen placed on top of them, such as GRACE [Simmons *et al.* (2003)], the mixed reality characters are visible from all angles and are not subjected to diminishing visibility at greater distances.

When they employ simple robots, such as in Jeeves [Young *et al.* (2007)] or Dragone et al.'s MiRA project [Dragone *et al.* (2006)], these systems are advantageous in applications with a high robot-to-user ratio, as a single wearable interface can augment the interaction capabilities of multiple simple robots (e.g. with no screen, head or arms).

Dragone et al.'s MiRA and Jeeves also take greater advantage of their mixed reality components, as they are free from the engineering effort of realising sophisticated mechanical interfaces. For example, a MiRA can have the ability to point and gaze in 3D by means of virtual limbs without having to construct any physical counterpart. In this manner, it can overcome the inherent limitations of screen-based solutions, as well as provide a rich repertoire of gestures and facial expressions, which can be used to advertise its capabilities and communicate its state (see Fig. 2.4).

Fig. 2.4 Examples of gestures and facial expressions in MiRA Chameleons (actual images displayed in the user's head-mounted display during live experiments and user trials with our application).

Notably, being based on virtual artefacts, behavioural capabilities of Mixed Reality Agents are not limited to "natural" human-like forms, but can also include more complex effects involving other virtual objects and cartoon-like animations. Jeeves [Young *et al.* (2007)], for example, "tags" real objects with virtual stickers and uses cartoon animation as an intuitive form of social interaction.

The remaining of this paper will focus on Dragone et al.'s MiRA project, which is renamed MiRA Chameleons here in order to more clearly distinguish it from Jeeves and U-Tsu-Shi-O-Mi [Shoji *et al.* (2006)]. The other reason for this name is that the MiRA Chameleons project carries over experience accumulated within the Agent Chameleons project by adding an agent-based coordination dimension between the wearable MR user interface and the robot forming a MiRA.

The vision realised within the MiRA Chameleons project is to build an ubiquitous agent that can truly cross the divide between the digital and physical world, by taking advantage of all aspects of dynamic embodiment, and which can also simply mutate its external appearance by acting on its virtual part.

2.6.1 *Requirements*

In order to create a proper test-bed for such an integrated approach, it is important to support its implementation in a flexible manner that would overcome the limitations of the early implementations of Agent Chameleons and also ease multiple instantiation of the system. For example, rather than ad-hoc coupling of the robot and the user's wearable MR interface, the system should work with heterogeneous robot hardware and computational platforms. Rather than using pre-configured connections between a fixed set of devices, the user should be free to move and chose among the devices

that are available.

In order to enable long-term experiments into the personalisation aspect of MiRA Chameleons, any practical implementation of the system should offer a sub-stratum over which the capabilities and the knowledge of a personal user agent may be openly re-configured and transferred across the different devices in the network. In particular, the MiRA Chameleon system can be facilitated by supporting:

- Dynamic discovery and management of spontaneous networks between users' MR wearable interfaces, portable devices and robots;
- Run-time, adaptive configuration of system components;
- Portability across different robot platforms and computational devices.

One software framework that incorporates these features is the Socially Situated Agent Architecture (SoSAA)

2.6.2 *The socially situated agent architecture (SoSAA)*

SoSAA [Dragone (2007)] is a software framework primarily intended to serve as a foundation on which to build different intelligent ubiquitous applications while fostering the use of AOSE by promoting code-reuse and integration with legacy systems and third party software. In particular, SoSAA leverages existing well-established research in the Component-Based Software Engineering (CBSE) domain [Clements (2001)]. Here, components are not merely passive, but play an essential role in managing the reactive part of the behaviour of an agent.

Figure 2.5 helps illustrating the SoSAA integration strategy. SoSAA combines a low-level component-based infrastructure framework with a MAS-based high-level infrastructure framework that augments it with its multi-agent organisation and goal-reasoning capabilities.

The low-level component framework in SoSAA can be used to instantiate different component-based systems by posing clear boundaries between different functional modules (the components), and by stressing the concepts of information hiding and composition rules that guide the developers in assembling these components into the system architecture. In particular, components interact via inter-component communication channels, which, contrary to object calls in object-oriented programming, are explicit architectural elements that are applied across the whole architecture and can be programmatically manipulated.

The low-level component framework in SoSAA provides:

(1) Support for the most important component composition styles, namely: connection-driven/procedural interfaces and data-driven interfaces (based on messaging and/or events).

(2) Brokering functionalities to be used by components to find suitable collaboration partners for each of the composition styles supported. This enables indirect collaboration patterns among participating components that are not statically bound at design/compilation time but that can be bound either at composition-time or at run-time.

(3) Container-type functionalities, used to load, unload, configure, activate, de-activate, and query the set of functional components loaded in the system, together with their interface requirements (i.e. in terms of provided and required collaborations).

(4) Binding operations, with which client-side interfaces (e.g. event listener, data consumer, service client) of one component can be programmatically bound to server-side interfaces (e.g. event source, data producers, service providers) of other components.

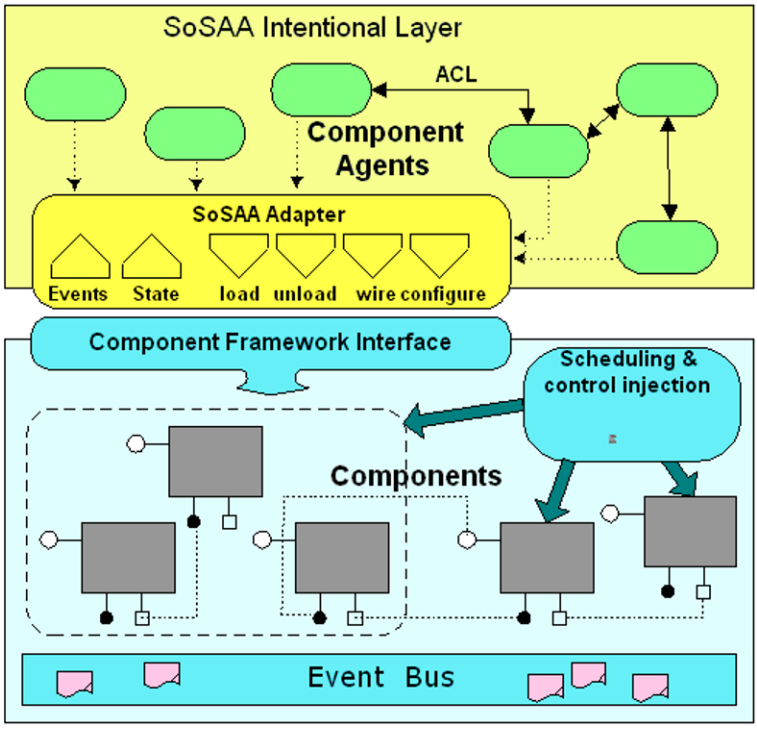

Fig. 2.5 SoSAA integration strategy.

(5) A run-time engine in charge of the execution and scheduling of activity-type components (e.g. sensor drivers, sensory-motor behaviours, data-processing routines).

In addition, an adapter layer in SoSAA provides a set of meta-level operators, which collectively define an interface to the intentional layer. In particular, the adapter layer allows access by multiple intentional agents, called component agents. The adapter layer provides meta-level operators and perceptors that collectively define the computational environment shared by the component agents in the intentional layer. Perceptors query the component framework and provide component agents with knowledge about events, the set of installed components, their interfaces, their actual binding, and their run-time performances, while actuators control the loading, unloading, configuration, activation, de-activation, and binding of components.

In summary: SoSAA requires the wrapping of functional skills within low-level components before they can be administered by component agents in the intentional layer. This enables the adoption of a low-level component model, for example, in terms of different component types with specific behavioural and collaboration patterns, which can be oriented toward supporting specific application domains. Low-level functional components in SoSAA will react to events according to a particular behaviour until they are instructed to do otherwise by the agents in the SoSAA intentional layer. Additionally, individual components may communicate amongst one another at the sub-symbolic level using inter-component communication channels. This results in the intentional layer being free to concentrate on higher-level reasoning. Furthermore, at the intentional level, SoSAA can leverage on ACL communication and dedicated AOSE methodologies, employing an organisational/social view

through which analysis and modelling of inter-agent interaction can be performed. In particular, roles in role-based architectures help structuring one system as a set of more manageable sub-systems by acting as abstract specifications of behavioural patterns Finally, since the adapter layer is defined in terms of both standard agent capabilities and common features of component models, SoSAA's design facilitates the replacement of different agent platforms and different component-based frameworks.

The separation between component agents and functional components in SoSAA is essential to decouple – for engineering purposes – the agent's mind from their working environment. Under this perspective, functional components provide the agent's body, that is, the medium through which the agent's mind can sense and affect a working environment. While the agent's mind can then be programmed according to different cognitive models, for example BDI, domain and environment-specific issues can be taken into account in the engineering of the underlining functional components.

There are currently two instantiations of SoSAA, both based on two open-source frameworks, namely: the Agent Factory (AF) multi-agent toolkit and the Java Modular Component Framework (JMCF)[1]. In addition to the standard versions, both frameworks come in versions that address computationally constrained devices, namely with AFME (AF Micro Edition), and JMCFME (JMCF Micro Edition). In particular, these latter versions loose some of their flexibility in favour of their smaller footprint, for instance, by renouncing Java's reflection functionalities. However, the result is that by combining the respective versions, a SoSAA system can be distributed on different computational nodes on a computer network (SoSAA nodes), each characterized by the presence of the SoSAA infrastructure.

2.6.3 *Implementation*

By supporting the integration of intentional agents and functional components in open and ubiquitous environments, the SoSAA framework enables the characteristic polymorphic combination of robotic and virtual components that characterises MiRA Chameleons.

Figure 2.6 shows how a MiRA Chameleon system is implemented as the collaboration between different types of SoSAA nodes, namely the standard SoSAA nodes, which are deployed on the robots and on back-end server-type nodes, and SoSAA-ME nodes, which are deployed on users portable devices such as PDAs or mobile phones as well as on MiRA MR wearable interfaces.

In essence, a MiRA Chameleon is the result of a collaboration between a robot and a user node (see Figure 2.7), which communicate over an ad-hoc wireless network link in order to exhibit cohesion and behavioural consistency to the observer. Tracking is achieved by simply placing a cubic marker on top of the robot and tracking its position from the camera associated with the user's HMD. The tracking information is then used to align the image of the virtual character with that of the real robot.

In order to be a believable component of the mixed reality agent, the behaviour of the virtual character needs to exhibit a degree of awareness of its surroundings, comparable to a robot being physically embodied through an array of physical sensors and actuators. In MiRA Chamaleon, the instrument for such situatedness is the update input stream, which notifies the MR interface about the nature and relative position of the obstacles and other objects of interest perceived by the robot, and also about the intentions of the robot. This allows direct control of deictic and anticipatory animations of the associated virtual character, which can visually represent the robot's intentions (similar to Jeeves's cartoon-like expressions of the robot's state [Young *et al.* (2007)]).

On the other hand, since the MR overlay of virtual images onto the user's field of vision requires exact knowledge about the position and gaze of the user, this information can also be relayed to the robot. In doing so, the wearable interface helps the robot to know the position (and the identity) of

[1] SoSAA, AgentFactory and the JMCF are all available for download at `http://www.agentfactory.com`

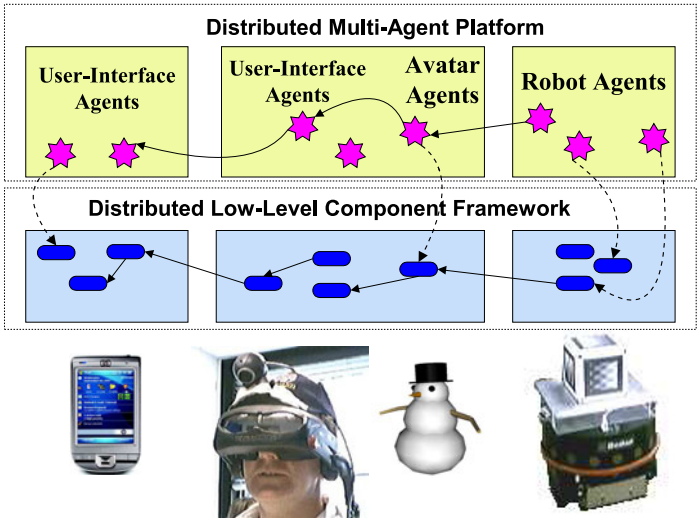

Fig. 2.6 SoSAA nodes for MiRA Chameleons.

the user, while the user can use his gaze direction to spatially reference objects and way-points in order to influence the robot's behaviour. The communication between wearable interface and robot is therefore essential in reinforcing the embodiment of each part of a MiRA (see the discussion in Section 2.4.1) and augmenting the system's HRI capabilities by merging these parts into a single agent.

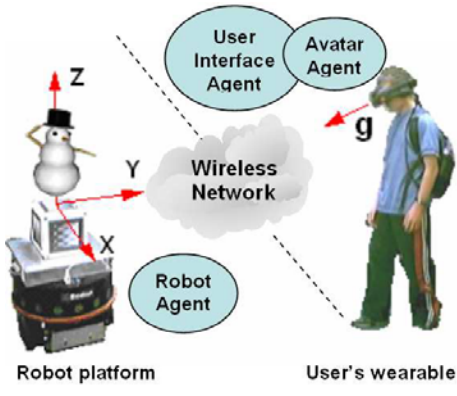

Fig. 2.7 Collaboration between user and robot node.

The nature of the functional components deployed on each node depends on the node type. Functional components in portable devices wrap application-specific and user interface (e.g. text-based) functionalities. On a typical robot node, components can range from active components encapsu-

lating behaviour production or data-processing functionalities, to passive data components granting access to either bodies of data (e.g. task- and environment-related information, knowledge bases, configuration) or to a robot's hardware. Finally, functional components in MiRA's MR wearable interfaces are similar to those deployed on robot nodes, although the hardware in this case is constituted by the virtual models of the MiRA's virtual characters, while data-processing components encapsulate AR-tracking functionalities or proxy-type components consuming sensors data transmitted by the robots.

Other than disjointed, passive functionalities that are exported through a low-level API, as in the original Agent Chameleon, functional components in a SoSAA node are organized in component contexts that are already capable of primitive functionalities. For instance, in isolation and without its intentional layer, the low-level component framework of a robot node will still be capable of reactive behaviours, such as avoiding colliding with approaching obstacles, or responding to simple instructions, such as wander and turn. Similarly, a virtual character in a MiRA MR wearable interface will be able to track the position of the user, exhibit simple base-type autonomous behaviour, such as breathing and eye blinking, and respond to simple instructions, such as gaze-toward-point, and move or rotate joints.

On top of their respective component-based frameworks, each SoSAA node is supervised by a number of basic component agents attending a number of high-level behaviours and facilitating the integration between the node and the rest of the system. These component agents become aware of the low-level functional capabilities available in the node by querying their component-based framework with the meta-operators defined in the SoSAA adapter. They can thus abstract from the specific implementation details of the functional components by operating at the knowledge level of the component types and component signatures (in terms of required and exported interfaces). Subsequently, they can employ domain specific knowledge, which is expressed at the same level, to harness the functionalities of the node to create high-level behaviours, for instance, by sequencing and conditioning the activation or primitive behaviours within the low-level component framework. At the same time, by publicising their high-level capabilities in a common ontology, these component agents insulate the specific details of each node and enable other component agents that are unaware of these details to operate within the same node. For example, both robots equipped with laser range sensors and robots equipped with sonar will publicise that they are capable of moving to a certain room, although they will ultimately employ different plans to achieve the same objective.

Collectively, these mechanisms give each SoSAA node reflective capabilities both at the functional and at the ACL level, which respond to the requirements dictated by the MiRA Chameleon by enabling a flexible migration of functionalities across heterogeneous SoSAA nodes. In the migration by cloning, as performed in the old prototype of the Agent Chameleon system, an agent's deliberative layer had to find an exact copy of its ties with the environment in each platform it migrated to. In contrast, functionalities in MiRA Chameleon migrates in terms of component agents, typically covering user- and task-specific roles, which then adapt to their new hosting platform by finding, possibly organising, and finally initiating the necessary collaborations with the functionalities available in situ.

The adaptation stage in the migration of component agents in SoSAA is what enables the dynamic integration of MiRA Chameleon functionalities over heterogeneous nodes. In addition, SoSAA facilitates the distribution of these functionalities thanks to a hybrid communication model similar to the one employed within the RETSINA MAS [Sycara *et al.* (2003)]. Specifically, the low-level frameworks of each node are connected thanks to the distribution mechanisms associated to the different component composition styles supported by SoSAA, that is, RMI (Remote Method Invocation) for component services, and JMS (Java Messaging Services) for data and event distribution. In addition, as in RETSINA, components can open inter-component back-channels employing different communication mechanisms and network protocols that are specifically designed to cater for specific flows of data, such as video streaming through RTP (Real Time Protocol) or UDP multicasting for peer discovery. On top of that, dedicated component agents (communicators) in the SoSAA intentional layer

employ a set of coordination protocols, based on FIPA-ACL, to supervise the federation of their respective low-level component frameworks, and also to facilitate opening, closing, and controlling the flow of information, and bearing higher-level conversation between inter-component backchannels.

Once aware of each other's presence, e.g. through the UDP peer-discovery service, communicators in the different nodes can manage the migration of component agents from the user's PAA to the his portable device to the MiRA MR wearable interface. Similarly, as soon as a robot's communicator starts to collaborate with the user's wearable node, the two can exchange information, e.g. about each other's identity, and also agree on migrating some functionalities from the user node to the robot node before the robot enters the visual field of the user's HMD. After this first connection, and until robot and human depart from each other, the two nodes will then collaborate to deliver a MiRA composed of the real robot and a virtual character visualised through the user's HMD.

2.6.4 *Testbed*

An example will better illustrate how the functionalities of MiRA Chameleon are implemented availing of the SoSAA framework. In particular, in order to drive the implementation of the new MiRA Chameleon system, we implemented a simple application scenario to demonstrate the joint exploitation of gaze tracking and positional awareness[Dragone *et al.* (2006)], and the expressivities capabilities of MiRAs [Dragone *et al.* (2007)] by asking users to observe and collaborate with a soccer playing robot. Specifically, the user can ask the robot to find and fetch an orange colored ball, and also direct some of the robot's movements to facilitate the successful and speedy execution of the task. A number of application-specific component agents and functional components in each node complement the generic infrastructure, described in the previous section.

2.6.4.1 *Robot node*

Additional component agents on the robot supervise functional context (sub-systems), such as navigation, localisation, object tracking etc. In order to enable the soccer game scenario, low-level components implement object recognition functionalities (based on blob colour-tracking), which are used to recognise the ball, and behaviours for approaching, grabbing and dribbling the ball.

The robot control system also includes an object-tracking manager component agent, used for the definition and maintenance of way-points which can be associated with objects tracked by the on-board camera or initialised to arbitrary positions registered with the robot positioning system (i.e. its odometry). Successively, these way-points can be used as inputs for other behaviours. The mechanism enables behaviour persistence, which allows the robot to keep approaching the ball even when it gets momentarily occluded, and also supports multiple foci of interest, so that the robot can turn toward the user and then return to pursuing the ball.

Notably, within the SoSAA-based implementation, setting such a demonstrative scenario allows fixing the application-specific ontology describing the high-level capabilities of the robot, e.g. in terms of actions that can be performed (e.g. `turn right/move forward`), and objectives that can be achieved by the robot (`canAchieve(close(?object,?way_point))`).

2.6.4.2 *User node*

Figure 2.8 roughly illustrates the organisation of part of the wearable SoSAA node in the MiRA Chameleon system. Within the node, component agents supervise two functional areas, namely the user interface and the control of the virtual character (avatar) associated with the robot. The user interface components control the display of text and other 2D graphic overlays in the user's HMD, and process user input. Through them, the user can be informed of details of the task and the state of the robot. These components also process user utterances availing of the IBM ViaVoice^{TM} speech

recogniser and the Java Speech API (`http://java.sun.com/products/java-media/speech/`). The vocal input may be used to trigger a predefined set of commands in order to issue requests to the robots. To do this, the recognised text is matched and unified with a set of templates and the result transformed into the corresponding ACL directive, e.g. `request (?robot, activate_behaviour (turnRight))`, `request (?robot, achieve-goal (close (ball, user)))`, `request (?robot, achieve-goal (close (robot, user)))`.

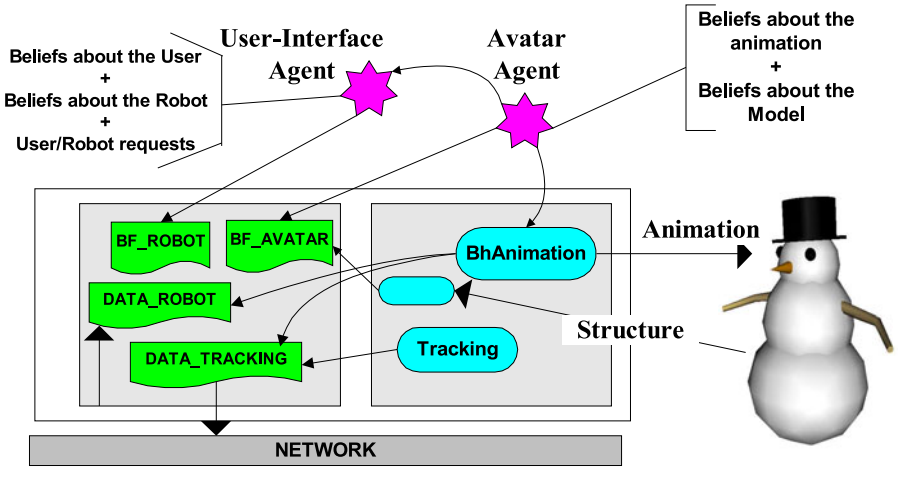

Fig. 2.8 Organisation of the user's MR node.

The behavioural animations of the virtual character are implemented via OpenVRML (`http://www.openvrml.org`), an open source C++ library enabling a programmatic access to the VRML model of the avatar. The avatars employed in MiRA Chameleons are also part of the distribution of the Virtual Character Animator software by ParallelGraphic. Some of these avatars are simple caricature characters (as the one depicted on the left in Figure 2.9) while others are cartoon-type humanoid characters compliant with the H-Anim 1.1 standard (see the right avatar in Figure 2.9). By using the access provided by OpenVRML to the virtual models loaded in the application, the system can automatically form a picture of the structure of each virtual character, e.g. reporting the joints in their models, if they are H-Anim models, and also the presence of pre-defined animations scripts. This information is then exported through the SoSAA adapter and finally represented in the SoSAA intentional layer in form of belief sets, which are then used by the component agents supervising the avatar.

2.6.4.3 *Collaboration between robot and virtual character*

Most of the coordination between robot and virtual character is achieved through high-level requests from the robots agents to the user interfaces agents (e.g. `<greet the user>`) or communication of intentions (e.g. `<committed turn-right>`) rather than low-level directions (e.g. `<move arms up and wave 3 times>`) which require more data and use up much of the robot's attention. However, in order to execute animations that are coherent with the robot's perception of its environment, once activated, the animation components will also access low-level data such as sensor readings or other variables used within the robot agent, and the tracking information reporting the position of the observer in the robot's coordinate system.

Fig. 2.9 Different examples of avatars (actual images displayed in the user's head mounted display during live experiments and user trials with our application).

Let's say, for instance, that the robot wants to greet the user. Since the robot does not have hardware capabilities to do that, it will request the help of the user node through an ACL request `<greet the user>`. The user interface agent will forward the request to the avatar agent in charge of the virtual character associated with the robot. The avatar character will then carry out this high-level request by using the specific capabilities of the virtual character. As a result, the snowman character will greet the user by turning its whole body toward the user and by waving its hat - an animation built into the character's VRML model. In contrast, H-Anim characters will be able to more naturally gaze at the user and wave their hands. In both cases, facing the user is possible because the animation activities know the position and the orientation of the user in relation to the robot-centric frame of reference.

2.6.5 *Discussion*

In general, through its SoSAA-based implementation, MiRA Chameleons can easily adapt to different users, different robots, and different applications, as there is no pre-defined coupling between the robot and the appearance and behaviour of its associated virtual character. Instead, thanks to the agent-based coordination described in the previous section, the system as a whole can take context-sensitive decisions in order to deliver a personalised, adaptive, augmented HRI interface.

In particular, the MiRA Chameleons system is a suitable testbed to experiment with external mutation as envisaged within the Agent Chameleon project. By way of example, the system may utilise the role description for the robot as well as a profile of the observer (e.g. his identity and preferences) in order to personalise the form of the avatar, e.g. to project a snowman avatar while engaging in playful activity and a more serious avatar when teaching.

Such a personalisation of form may be augmented by way of personalisation of behaviour. As the user interface agent resides on the user's wearable computer, it may easily access (or indeed learn) useful personal data about the user that can be consequently used to negotiate the robot's behaviour to better suit the perceived needs of the user. Notably, an important advantage of having the user interface agent acts as an intermediary between the user and the robot is that the robot's behaviour

can be influenced without disclosing personal data of the user. For instance, if the user needs to follow the robot, the user interface agent proactively may ask the robot to slow down in order to prevent possible collision with the user.

2.7 Conclusion

This paper has advocated that robots play a significant role in the digital society. However, their heterogeneous nature, as well as their characteristically rigid and inflexible forms, pose significant barriers to the development of effective interaction modalities. Thus the objective of attaining seamless and intuitive interaction, as per the ubiquitous computing and ambient intelligence visions, remains somewhat distant. One cost-effective and flexible solution to this is to engage an eclectic mixture of technologies that harmonises human-robot interaction with other interaction modalities.

Dynamic embodiment offers one avenue for realising agents that can traverse technology boundaries and offer a consistent interface to the end-user while taking advantage of the facilities that individual platforms, of which robots are one instance, offer.

Mixed reality, incorporating virtual characters, offers an intriguing approach for modelling robotic forms in a flexible and adaptable manner. A combination of dynamic embodiment and MR enables the robotic entities to be endowed with interfaces that both harmonise with and augment conventional user interfaces.

Making the leap from proof of concept to practical system is an essential step to investigate the usability of such systems in everyday applications. This paper describes an important step in such a direction by presenting our system architecture based on the SoSAA software framework.

Future work will be dedicated to improve the interoperability of our solution, by employing standard representations of the system's functional ontologies, and to create a stable environment for testing and developing suitable methodologies supporting the adaptability of the system (e.g. through learning).