Agents in domestic environments

Leo van Moergestel, Wouter Langerak,
Glenn Meerstra, Niels van Nieuwenburg,
Franc Pape, Daniël Telgen and Erik Puik
HU Utrecht University of Applied Sciences
Utrecht, the Netherlands
Email:leo.vanmoergestel@hu.nl

John-Jules Meyer
Utrecht University
Utrecht, the Netherlands
Email: J.J.C.Meyer@uu.nl

***Abstract*—This paper describes an agent-based architecture for domotics. This architecture is based on requirements about expandability and hardware independence. The heart of the system is a multi-agent system. This system is distributed over several platforms to open the possibility to tie the agents directly to the actuators, sensors and devices involved. This way a level of abstraction is created and all intelligence of the system as a whole is related to the agents involved. A proof of concept has been built and functions as expected. By implementing real and simulated devices and an easy to use graphical interface, all kind of compositions can be studied using this platform.**

I. Introduction

Our research group focuses on the applications of agent technology. An interesting application field for agent technology is domotics. Domotics is also called home automation and it is a field within building automation. Though building automation focuses normally on big buildings where people come together for work, education, shopping, recovering, sporting or having a meeting, domotics is specializing in the specific automation requirements of private homes. The application of automation techniques is meant for the comfort and security of its residents. Domotics applies many techniques used in building automation such as light and climate control, control of doors and window shutters, security and surveillance systems, etc. but additional features are used in domotics. These additional functions in home automation include the control of multi-media home entertainment systems, automatic plant watering and pet feeding, and automatic scenes for dinners and parties [1]. Additional features are also security and adaptation of the system to the behaviour of the inhabitants.

An important difference between building automation and home automation is, however, the human interface. In home automation, the control of the system is not done by highly trained technical people as is the case in building automation. Because the control should be done by the home inhabitants the control should be easy, largely image-based and self-explanatory.

Home automation could use wireless techniques, but normally a wired infrastructure is used. A wired infrastructure is a bit more reliable and when home automation is installed during construction of a new home, usually control wires can be added without much extra work. In standard automation systems these control wires run to a controller, which will then control the environment. However, in practice home automation is often added after the home has been built and even then it should be easily adaptable in the future when new opportunities and techniques become available. In automation there is a trend towards more intelligent devices and a distributed approach for the system as a whole.

In the next sections at first we focus on domotics and its characteristics. In this section also the goal of the research project is explained resulting in system requirements. Next, the design of the system is discussed. In that section hardware and software platforms are introduced. The system architecture is explained in a separate section that will be followed by the implementation, the results and a of course a discussion about related work and a comparison of our work with other research in the field of domotics.

II. Characteristics of domotics

In this section we first discuss domotics and its levels and global architectures. Next the formulation for the goals of our system will be introduced as well as the global system requirements.

A. Domotics

Home-automation is sometimes used as a synonym for domotics, but Harper [2] describes five levels of home automation and states that only level four and five apply to domotics [3]. The five levels are:

1) Homes containing stand-alone intelligent objects;
2) Houses containing intelligent communicating objects. In this case a performance gain can be achieved by sharing information between the objects;
3) Homes that communicate by themselves. In this case internal and external data communication networks open the possibilities to remote control and monitoring;
4) Learning homes; activity patterns are recognized and applied to optimize the technology in house;
5) Attentive homes; the activity and location of people and objects within the homes are constantly registered, and this information is used to control technology in anticipation of the occupants needs.

Looking at these levels we observe an increase of the amount of communication, interoperability and artificial intelligence techniques going from the first level to the highest level. Thus to open the road to the highest level, from the starting point technologies should be applied that do not obstruct this path towards higher levels. In [1] three possible architectures are described.

1) Centralized architecture: a centralized controller gets information from all kind of sensors or sensor networks and controls the actuators available;
2) Distributed architecture: the sensors and actuators are intelligent by themselves and communicate to get the desired actions.
3) Mixed architecture, both sensors and actuators are intelligent but there is also a central system to coordinate the actions.

B. Goals for our system

For our domotic system the most important goals are to develop a system that is simple to use, easy to implement, reliable and expandable. To achieve these goals interoperability between components is an important issue. This interoperability could be possible by adhering to open standards that are widely supported like network protocols and software platforms that can be connected by applying these protocols. These platforms should support modern software applications. Nowadays powerful computing platforms are available having a small size and a low price.

C. Global system requirements

Considering the characteristics and the aforementioned goals, we come to the following system requirements.

1) modularity; to be expandable a modular design must be followed;
2) configuration of the system should be easy but the configuration system should also be expandable;
3) maintenance and monitoring should be properties of the system to assure high reliability;
4) adaptivity to new situations, like new devices and new rules for operation should be possible.

As a general definition we consider a device to be a sensor or actuator. The system is rule-based where rules are applied to events, measurements and preferences of the end-users. By applying these concepts the domotic system is not only a set of automation islands, but offers integration of the different parts. Smart integration, where rule-based knowledge adjusts to the needs of the users (humans) should be possible. To make this possible, end-users are represented by agents to communicate their preferences to the system. A high level of integration is achieved by putting agents in devices at the hardware level, thus introducing a kind of abstraction layer. The system as a whole is a multi-agent system (MAS) en its working is based on inter-agent communication. The reasons for this approach will be explained in the next section.

III. Design considerations

To design the domotic system according to the requirements mentioned in the previous section, some considerations have to be made. Why is agent technology apt for the system, what communication model should be used and finally is there a relationship with other work in our research department.

A. Agents

A device in a domotica system is acting in an environment and its actions are influencing that same environment. The devices should have the possibility to communicate and cooperate to achieve systemwide goals. Looking at these requirements and properties of devices, they are also found in the definitions of agents for example the definition given by Wooldridge [4]: *An agent is an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives.* Considering these similarities agent technology seems to be a natural choice.

To mention a few tasks that should be done by the device agents:

- at start-up testing the device and registering the device in the MAS
- interfacing at the lowest level with the hardware for actions to be performed or measurements to be done.
- offering an abstraction layer usable for interoperability
- monitoring the use and health of the device.

B. Communication models

When agents must be coupled with devices several approaches are possible. In figure 1 the situation is shown where agents reside in an agent runtime environment that is coupled with the actual devices by several means of communication links. Interfaces will provide the actual coupling. A drawback of this situation is that most devices offer different interfaces so the communication methods depend largely on the types of devices used. The advantage is that interagent communication is simple, having the agents running in the same environment. Another possibility is shown in figure 2. Here the agents are

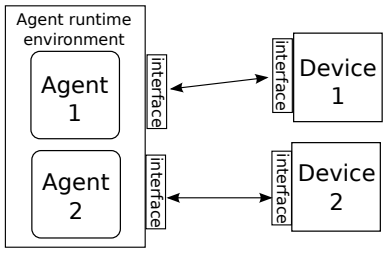

Fig. 1. Communication model 1

tightly coupled with the devices and a network, depicted as a cloud, will be used for the agents to communicate. In this model the communication is only between agents and this can be done on basis of TCP/IP using standard webtechnologies. The problem that agents are running on different platforms can be solved by using an environment that makes interoperability of agents on different platforms possible. As we shall see the Jade platform couples different containers running agents over the network in a transparent way.

C. Connection with related research in our group

In earlier research the roles of agents in the life-cycle of products were investigated [5]. From this research it became

Fig. 2. Communication model 2

clear that adding an agent to a product or device offers all kind of possibilities and advantages. To mention a few advantages of adding this so called product agent:

- if connected to the internet, products can communicate worldwide. The embedded agent is the enabling technology for the concept of the Internet of Things [6];
- the embedded product agent can monitor the use of a product;
- the product agent can perform a power-on self test (POST) where the functionality of the product is tested every time it is switched on and the product agent can also test the subsystems of a device;
- in case of a broken subsystem the embedded agent can search for a replacement.

Considering these advantages embedding the agents in the device itself or make a tight coupling with the device as proposed is figure 2 can introduce the aforementioned advantages. In [7] a production model is described where agents are added during manufacturing of products (or devices). The product agent introduced in that model is the basis of the product agents in the life-cycle of a product. However, a product agent can also be added at a later stage in the life-cycle as has been done in [5] where a product agent is added in the use-phase of a product.

It is important to emphasize that the role and responsibilities of product agents in the use-phase of a product is different from the role and responsibilities of the agents that are part of the domotics MAS. However, a product agent is closely tied to a product and thus needs a hardware platform to run on. It can share this platform and software environment with the agents that play a role in de domotics system. This way, it is a natural step to create a distributed multiagent system as the basis for our domotics system and the advantages as described here of also having product agents in the system are available.

D. Hardware set-up

To implement the distributed approach it is necessary to create an environment for agents near the devices. Devices were equipped with a small computer system for the agents to run. After some research for platforms for our system, the Raspberry-Pi seemed to fit the requirement of offering a stable and cheap hardware platform, capable and powerful enough to run a Java virtual machine to support the Jade environment, offering standard ethernet connection and last but not least having the possibility to attach the hardware device itself to this system. Though this might seem overkill to use such a sophisticated device, it offers opportunities for the embedded agent having a huge system resource for future expansions. These resources might be needed is case of expanding the domotics system to the level of attentive homes.

E. Software platform

For software the Jade platform was used. Jade [[8]] was used as a platform for the MAS. The reasons for choosing Jade are:

- the system is a multi-agent-based system. Jade provides most of the requirements we need for our application like platform independence and inter agent communication;
- Jade is Java-based. Java is a versatile and powerful programming language;
- because Jade is Java-based it also has a low learning curve for Java programmers. This should be considered an advantage for the developers of a product on the Jade platform, not for the end-users of a product developed on Jade;
- the agents should be capable to negotiate to reach their goals. Jade offers possibilities for agents to negotiate. If we need extra capabilities, the Jade platform can easily be upgraded to an environment that is especially designed for BDI agents like 2APL [[9]] or Jadex [[8]]. Both 2APL as well as Jadex are based on Jade but have a more steep learning curve for Java developers;
- agents can migrate, terminate or new agents can appear.

The Jade runtime environment implements message-based communication between agents running on different platforms connected by a network. In figure 3 the Jade platform environment is depicted.

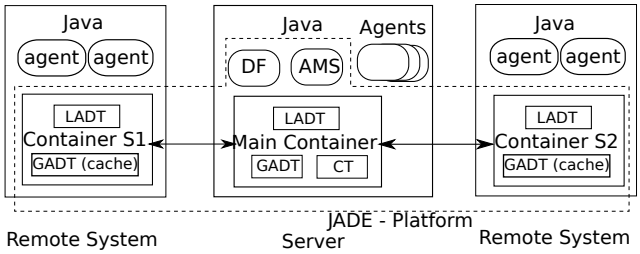

Fig. 3. The Jade platform

The Jade platform itself is in this figure surrounded by a dashed line. It consists of the following components:

- A main container with connections to remote containers;
- A container table (CT) residing in the main container, which is the registry of the object references and transport addresses of all container nodes comprising the platform;
- A global agent descriptor table (GADT), which is the registry of all agents present in the platform, including their status and location. This table resides in the main container and there are cached entries in the other containers;
- All containers have a local agent descriptor table (LADT), describing the local agents in the container;

- The main container also hosts two special agents AMS and DF, that provide the agent management and the yellow page service (Directory Facilitator) where agents can register their services or search for available services.

Agents running on this platform are also visible in figure 3. These agents can be implemented in Java by extending the agent-class offered by Jade. Every container can run its set of agents and these agents can communicate with each other.

IV. SOFTWARE ARCHITECTURE

In this section the architecture of the system is presented. First the global architecture. In the next subsection the roles and responsibilities of the agents involved are discussed as well as the global architecture of two design models of the device agent. Finally the Inter-agent communication and message types are presented.

A. Global system architecture

In figure 4 the global architecture of the domotics system is shown. A GUI subsystem is provided for configuration, control and monitoring. The blackboard system in the middle is the place where all relevant information that could be shared among the participating agents is collected. This blackboard system supports a publish and subscribe system that is used for inter-agent communication. At the bottom we see the actual device agents.

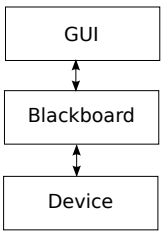

Fig. 4. Global system architecture

B. Agent roles and responsibilities

In agent-oriented software engineering (AOSE) [10], the roles and responsibilities of the agents form the basic of the agent software model. Our MAS contains four types of agents.

1) The device agent, closely coupled to the hardware devices in the domestic environment;
2) The human agent: representing the human inhabitants of a home;
3) the blackboard agent controlling the inter-agent communication and the storage of important data;
4) the GUI-agent serves as a middleman between the GUI and the MAS.

Looking at roles and responsibilities results in the following observations: The device agent will directly control a device. The actual control depends on the device being a sensor or actuator, so actually a device agent can be an actuator agent or sensor agent. It will receive information that it has subscribed to and it will publish information on the blackboard. Depending on the information received and its rule-base it will control the device. In an earlier section III-C, the concept of a product agent was introduced. This product agent could be the representation of a product in the Internet of Things and has the responsibilities mentioned in section III-C. Being tightly coupled to a device (actually a product) and capable of communicating, the device agent could possibly also play the role of a product agent, this means; monitor the device, perform a power-on-self-test to check the health of the device and collect information about usage of the device. However, the product agent can also be implemented as a separate agent running on the same hardware platform in the same software environment as the device agent.

The human agent will present a human in the system. This agent is implemented as a sensor agent and shows the location of a human inhabitant along with its preferences and physical situation. The blackboard agent will control the agent network, storing information and giving information to other agents. It will keep track of subscriptions done by the other agents and will inform these subscribed agents when requested or when an update is done by another agent. The GUI-agent: is a part of the GUI subsystem. It has been implemented to make communication with the blackboard agent and other agents in the MAS at an inter-agent communication level possible.

In figure 5 the inheritance model of a device agent for a lamp device is shown. First a general device agent will be responsible for communication with the outside world. This agent is expanded to a lamp agent at the application layer and finally this results in a real hardware lamp agent or a simulated lamp agent. This way it is easy in our implementation to introduce simulated devices. These simulated devices are visible in a GUI environment and can be used to build a system for testing purposes or in situations were the actual hardware is not yet available. The actual working of these non-existent devices can be observed using the GUI.

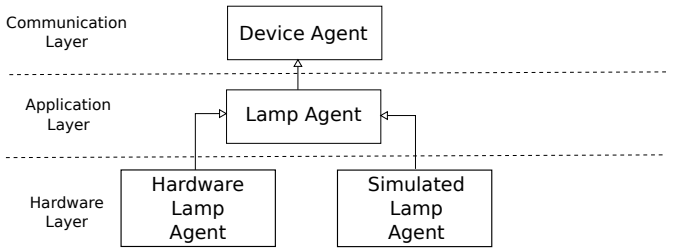

Fig. 5. Inheritance model for a lamp device

In figure 6 another type of device agent is shown: a light sensor. This agent is also derived from the device agent. However at the application layer an extra functionality typical for sensors is added. This extra block performs polling of the sensor to get new data values.

C. Inter-agent communication

All communication for the system as a whole is done at the agent level. The device agent have their own specific and perhaps dedicated communication interface with sensors and actuators. The Jade environment supports the FIPA-standard for inter-agent communication (FIPA stands for Foundation

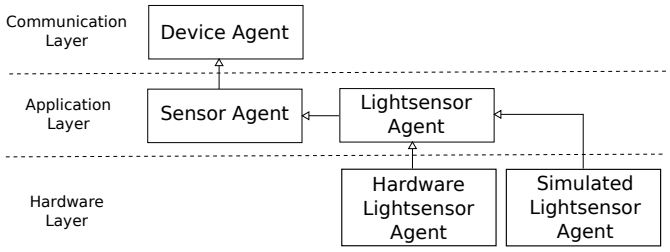

Fig. 6. Inheritance model for a light sensor

for Intelligent Physical Agents). So the standard FIPA possibilities are already implemented and supported. Within FIPA a message format must be chosen. Four possibilities were investigated:

- design of a new specific format;
- CSV (Comma Separated Values);
- JSON (JavaScipt Object Notation);
- XML (eXtensibel Markup Language).

The first two options were rejected. A new format means a lot of extra software tools to be developed. CSV is too primitive, for example nesting is not supported. This leaves JSON and XML as a choice. For both choices validator tools and libraries are available, however XML is more mature and this was the reason we have chosen XML.

A message contains the following items:

- A value;
- A key, if a device can send different kind of values;
- The topic the value is related to (i.e. light etc.)
- An agent/device identifier.

Using this information a device agent can send different values by submitting key-value pairs, where key identifies a specific type for the value. By sending these kind of messages to the blackboard agent, the information is stored in a database and the blackboard agents will direct this information to other agents that have subscribed to this information.

Several types of messages are possible: to mention a few:

- Subscribe: subscribe to information for a certain topic. If the information of a topic changes, automatic information update is send to a subscriber;
- Unsubscribe: used to stop a subscription;
- Request Value: get information from the blackboard about an agent;
- Publish: this can be done by a device agent. This way it will put information on the blackboard.

V. Implementation

In the next three figures some details of the internal structure of the agents and gui system is shown. The communication layer is part of all agents and the GUI system. In figure 7 the internals of a device agent is depicted. These type of agents interact with the actual hardware using the hardware layer though it is also possible to simulate the hardware. The embedded GUI subsystem in this agent is available to monitor the actual devices or in case of a simulation the simulated device. Artificial intelligence software runs in the application layer in combination with a message processor for interpreting the messages received from other agents (using the blackboard) or to construct messages meant for other agents. The blackboard

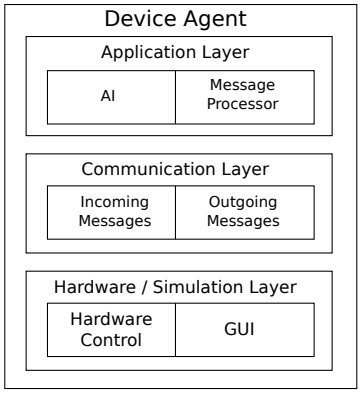

Fig. 7. Internal structure of a device agent

agent takes care of data storage in the application layer and the message processor implements the publish and subscribe mechanism. The communication layer serves the same goal as in the device agent: supporting communication with other agents. The GUI system is not an agent, but has an embedded

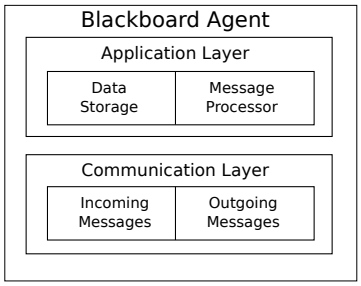

Fig. 8. Internal structure of the blackboard agent

agent to make it part of the multi agent system. This agent has only a communication layer to support the communication with the MAS. The actual implementation was made on a standard

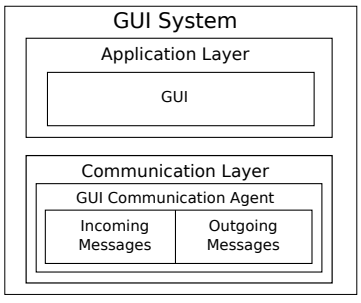

Fig. 9. Internal structure of the GUI system

desktop system connected by ethernet interfaces to several raspberry-Pi systems. Ethernet over powerlines was used to

minimise additional cabling requirements. Because many devices are actually connected to powerlines this approach seems to be the most natural. In the future devices could embed an agent environment system based on embedded technology and by attaching these devices to the power, the communication infrastructure is immediately established. On top of ethernet, TCP/IP is used as the carrier of inter-agent communications. Devices having special interfaces are connected to the agent platforms close to these devices. This way it is not necessary to support all kind of exotic or non-standard cabling systems.

Using the GUI drawing tool, a map of the home was easy to draw and in this map the position of several devices could be drawn. The map is stored as an XML file so other XML aware applications can easily get the actual information data about the map. Figure 10 gives an impression of how this part of the GUI looks like. This tool is based on graphic standards

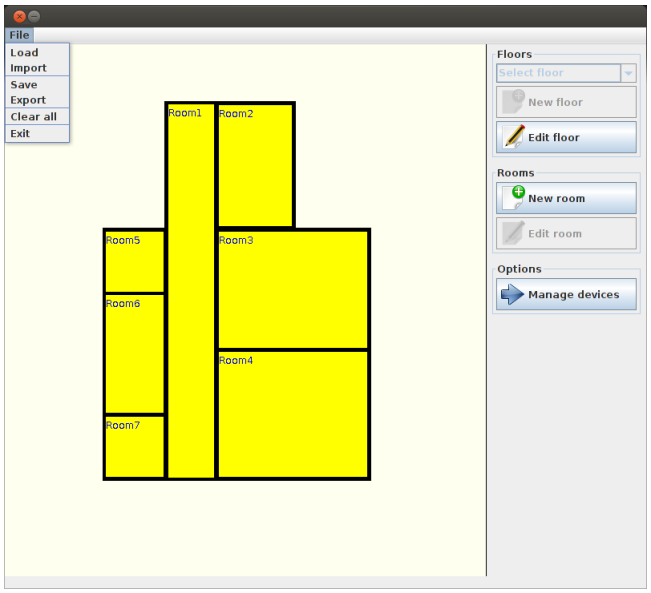

Fig. 10. The design part of the GUI subsystem

that can also be used in other applications. This way a portable and open system is also applied at the GUI level. Using this map, several types of devices can be added to the system by the end-user of the system. A device related agent will be created as well. In figure 11 a dialogwindow for creating an new device is shown. Using this GUI a domotics system for a home could easily be built by the end-user because of the intuitive and simple user interface. As shown in figure 11 a

Fig. 11. Adding a new device to the system

device can also be removed from the system. This removal includes the device related agent.

VI. Results

What has been created is a domotics implementation based on agent technology. By using agent technology the devices involved were closely tied to agent making these devices versatile as well as intelligent. For the system as a whole these devices could be considered as entities capable to operate at a high abstraction level. These combined device-agent system can be seen as nodes operating in the internet of things. By using a blackboard and a publish and subscribe system these agents could interact and exchange information. This fits the basic requirements of the system.

Figure 12 shows the actual implementation of the lamp-device. By just plugging in this device into the power outlet the device is powered, the Raspberry-Pi is activated and the communication over the power-line is set up. The lamp is coupled with the raspberry-Pi by a simple interface on the breadboard shown in the picture. This interface is connected to a relay for switching the lamp on and off. Other devices like

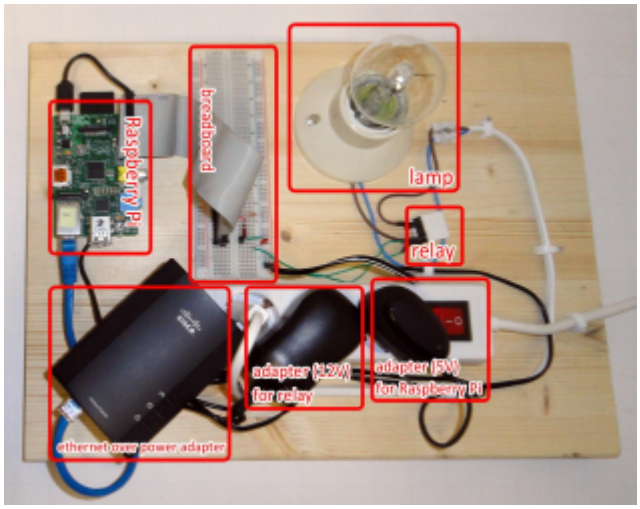

Fig. 12. Prototype of the lampdevice hardware

controllers have been implemented as simulated devices. The same is true for the agents representing the human inhabitants. These simulated agents can be accessed and controlled by the GUI to check the working of the domotics system as a whole. In the domotics system, both communication and the software implementation were based on open en widely available standards. This makes it easy to expand the system and combine it with other open standard based techniques. This was an important goal of this project.

VII. Discussion

First the focus is on the network infrastructure used in our solution. In domotics among others KNX is a widely supported solution [11]. Most KNX-based systems use a centralized control and configuration system. KNX claims to have the following advantages:

- Interoperability. KNX devices from different manufacturers will opperate together;
- International standard. KNX is an international standard, adopted among a wide range of manufacturers;

- High product quality. Products conforming the KNX requirements should have a high quality;
- Manufacturer-independent tooling. Tools for supporting KNX are available from different manufacturers;
- All home and building control. KNX claims to support all control requirements within home and building automation;
- Different kind of building. Simple and complex systems can use the same standard;
- Support for different configuration mode. Simple and complex configuration modes are possible;
- Different communication media, like wireless, twisted pair or using the wiring of the power net;
- Connection to other systems is possible;
- Independent from any hard- or software technology.

Almost all of these advantages except perhaps the high product quality also apply to a TCP/IP based system as described in the current paper. TCP/IP does not guarantee high quality of a product, only adhering to the TCP/IP standard. A further advantage is that in most cases existing network infrastructure is already based on TCP/IP thus introducing new devices is easy to accomplish. In the industry the last decades have shown an trend towards the use of standard networking techniques like ethernet and TCP/IP replacing so called field-buses except for special situations where extreme conditions occurring in the chemical industry require special solutions or when extreme hard real-time requirements play a role. For most situations however standard networking solutions, often with enhanced reliability like industrial ethernet, offer the same reliability as field-bus-based solutions and are also very cost effective. The advent of the next version of the Internet Protocol (IPv6) will also ease the adoption of standard internet in new fields, because the new features like an enormous number of node-addresses, support for different types of network traffic (Flow labeling and priority) and security (IPsec is standard included in IPv6) [12]. Integration with smart-phones becomes easy because these systems can operate in dual mode. By using the cellphone network they are capable to cooperate with the home network from all over the world. If the smartphone is within the reach of the local wireless network it can be a node in the system without introducing extra costs.

As a second aspect we will now focus on the properties of the agent-based solution proposed in the current paper. What has been done was bringing a powerful yet cheap and reliable platform close to the device. This approach also fits with the product agent concept introduced earlier in III-C. This agent is monitoring, performing a power-on self test and other tasks mentioned in section III-C. This results in two levels of error detection. At one level it might be that the whole node is unavailable and therefore this node will be excluded from the domotics system and an error message can be generated on the GUI. At another error level within the node itself a part or subsystem of the device is not operating. In this case the domotics system can pinpoint the problem to be solved.

Using a MAS where humans are represented by agents themselves is an enabling technology to integrate the humans in a MAS-controlled home automation system. A human is represented by an agent in the MAS and is capable to influence the system each using their own preferences.

VIII. Related work

The implemention of domotics systems based on agent technology has been done by several authors. Some publications like [13] and [14] focus on systems for disabled people. This can be considered as a specialisation within domotics where for example multimodal interfaces (gestures, text, voice and haptic devices) are used to interact with the domotic system. In [15] a Jade-based system is presented that has been developed to support elderly people. In that publication the authors also promote the use of open standards and open systems to make interoperability possible. However, their system has been designed for a special target group, while our system is meant for all kind of implementations of domotics. Other papers describe systems where agents are used to save energy like [16] and [17]. In this work agents focus on coordination of devices to minimise peak loads or to shift to moments of low energy prices.

The work of Bolzabi and Netto [18] describe the Home Sapiens, a smart home framework. They developed their own framework where three types of agents cooperate. User agents, representing users, Micro agents, representing devices and system agents to glue these components together. Our approach is different from this, by using Jade and agents that are tied closely to the devices, a clean abstraction layer is presented. All kinds of interaction between these agents is in principle possible without the need for extra system agents.

DomoBuilder presented in [19] is an agent-based system using the Jade platform. The agent model is used to achieve an abstraction level. So far for the similarities with our model. The difference with our model is that DomoBuilder is using a central controlling system called Kernel. This Kernel has some of the functionalities realised by the blackboard-agent in our model, but it has much more power to actually control the system using timers and event handlers. This results in a more centralized model while in our model the control power is delegated to the agents in the devices, resulting in a distributed model. In case of a communication problem, a distributed model has the advantage that the agents tied to the devices can control these devices them selves using rules how to operate in case of missing information, while in a centralized model the control of the devices is lost.

The work of R. Nunes as presented in [20] and [21] has its focus on the intelligence of home automation systems as well as smart energy management. In our work we focus on an architecture that enables the use of artificial intelligence techniques.

In [22] a Labview simulation is presented by Conte e.a. to control resource management in Home automation systems. In this paper the human user is also considered to be an agent. Though this simulation is based on agent technology, it does not take advantage of an agent-aware platform, as Labview is a programming environment based on graphical building blocks to build all kind of experimental and technical systems. Being commercial software Labview is also tied to licence costs. Except for DomoBuilder, most systems are based on

a central system where the MAS is implemented and agents keep contact with sensors and actuators by a specific data communication infrastructure and interfaces.

Other related work that is of interest are systems where agents are tied to products for monitoring, repair and recycling [5] [23]. This is work that is also done in our research group.

IX. Conclusion

In this research we implemented a powerful system. In our proof of concept only a few real devices were implemented. However the concepts that are used makes it easy to expand this approach to a much more complicated system without the need for redesigning the system as a whole. We also implemented simulated devices to show this possibility. An important aspect is the inclusion of a user interface for the end-user for configuring the system for a certain environment. This interface can also be used to adapt the system to a newly created situation. In future research smarter and learning agents will be used. Also the interaction with real human inhabitants should be implemented. This could be done by using smartphone devices but other possibilities should also be investigated. Having this platform available is a good start for this new research.

References

[1] *http://future.wikia.com/wiki/Domotics*, 2008.

[2] R. Harper, *The Connected Home: The future of domestic life*. Springer, 2011.

[3] R. Harper et al., *Inside the smart home*. Springer, 2003.

[4] M. Wooldridge, *An Introduction to MultiAgent Systems, Second Edition*. Sussex, UK: Wiley, 2009.

[5] L. v. Moergestel, E. Puik, D. Telgen, H. Folmer, M. Grünbauer, R. Proost, H. Veringa, and J.J. Meyer, "Monitoring agents in complex products enhancing a discovery robot with an agent for monitoring, maintenance and disaster prevention," *ICAART 2013 proceedings*, 2013.

[6] K. Ashton, "That 'internet of things' thing," *RFID Journal, 22 July*, 2009.

[7] L. v. Moergestel, J.J. Meyer, E. Puik, and D. Telgen, "Decentralized autonomous-agent-based infrastructure for agile multiparallel manufacturing," *ISADS 2011 proceedings*, pp. 281–288, 2011.

[8] R. Bordini, M. Dastani, J. Dix, and A. E. F. Seghrouchni, *Multi-agent Programming*. Springer, 2005.

[9] M. Dastani, "2apl: a practical agent programming language," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 3, pp. 214–248, 2008.

[10] R. Bordini, L. Braubach, M. Dastani, A. Seghrouchni, and J. Gomez-Sanz, *A survey of programming languages and platforms for multi-agent systems*. The Free Library, 2006.

[11] *http://http://www.knx.org/knx-standard/main-advantages/*, 2012.

[12] A. Tanenbaum and D. Wetherall, *Computer Networks*, 5th ed. Prentice Hall, 2010.

[13] C. Muñoz, D. Arellano, F. Perales, and G. Fontaned, "Perceptual and intelligent domotic system for disabled people," *Proceedings of the 6th IASTED International Conference on Visualization, Imaging and Image Processing*, pp. 70–75, 2006.

[14] M. Ruta, F. Scioscia, G. Loseto, and E. Di Sciascio, "Perceptual and intelligent domotica systems for disabled people," 2006.

[15] M. Brink, A. Jessurun, F. Franchimon, and J. v. Bronswijk, "An open agent-based home automation system," *Gerontechnology 2008;7(2)*, 2008.

[16] M. Ruta, F. Scioscia, G. Loseto, and E. Di Sciascio, "An agent framework for knowledge-based homes," *Third International Workshop on Agent Technologies for Energy Systems (ATES 2012). A workshop of the Eleventh International Conference on Autonomous Agents an Multiagent Systems (AAMAS 2012)*, 2012.

[17] S. Abras, S. Ploix, S. Pesty, and M. Jacomino, "A multi-agent home automation system for power management," *Proceedings of the Third International Conference in Control, Automation, and Robotics, ICINCO 2006*, pp. 3–8, 2006.

[18] C. Bolzani and M. Netto, "The engineering of micro agents in smart environments," *International Journal of Knowledge Based Intelligent Engineering Systems 13, no. 1*, pp. 31–38, March 2009.

[19] A. Addis and G. G. Armano, "Domobuilder: A multiagent architecture for home automation," *Proceedings of the 11th Workshop Dagli Oggetti Agli Agenti (WOA 2010)*, vol. 621 of CEUR Workshop Proceedings, September 2010.

[20] R. Nunes and S. da Silva, "Adding intelligence to home automation systems," *IADIS international conference applied computing*, 2004.

[21] R. Nunes, "Home automation - a step towards better energy management," *International conference on renewable energies and power quality*, 2003.

[22] G. Conte, G. Morganti, A. A. Perdon, and D. Scaradozzi, "Multi-agent system theory for resource management in home automation systems," *Journal of physical agents, Special session on practical applications of agents and multiagent systems*, vol. 3, NO 2, 2009.

[23] L. v. Moergestel, E. Puik, D. Telgen, and J.J. Meyer, "Embedded autonomous agents in products supporting repair and recycling," *ISADS 2013 proceedings*, 2013.