Designing Agent-Based Household Appliances

K. Steblovnik[a], D. Zazula[b]

[a]Gorenje d. d., Partizanska 12, 3320 Velenje, Slovenia

[b]University of Maribor, Faculty of EE and CS, Smetanova 17, 2000 Maribor, Slovenia

Abstract

As agents gain acceptance as a technology there is a growing need for practical methods for developing agent applications. Our paper introduces new technologies which are driven by fast new development of embedded systems and becoming very attractive for the design of intelligent household appliances. We assess the tools, design and development environments which have to support the implementation ideas in this field, we briefly focus on necessary agent structures, and reveal the parallels of the selected Prometheus agent design methodology and agent execution environment called Jadex. Finally a practical design example of an agent-based household appliance, namely Multiagent Washing Assistant as a special instance of Rational Home Assistant, is demonstrated. The applied steps have proven effective in assisting all the phases from the development, design, and documentation, to the system implementation and simulation.

Keywords: Multiagent systems, household appliances, rational agent, BDI model, agent design methodology

1. Introduction

Ambient intelligence, ubiquitous computing, intelligent multiagent systems, all combined with the Internet capabilities they bode significant changes of human dwelling environments in a very near future [1]. At the same time, the idea of intelligent home has matured, embedded computer technologies have reached the price/performance ratio acceptable for wide consumer community, while fast communication infrastructure spread out to all levels of today's society. This is bringing forth a new type of consumers that expect the way of usage and operation of technological devices will draw as close to the natural as possible.

Consequently, new families of household appliances will have to follow these trends. In the first place, this applies to home entertainment facilities and the products of white-ware. Some of such high-end appliances can already be found on the market: washing machines, refrigerators, or ovens are controlled by 16-bit microprocessor technologies, touch-sensitive screens, and built-in speech-recognition interfaces. Higher levels of integration and more computing performance are becoming imperatives. Trend-setting companies are developing embedded systems with 32-bit Arm RISC technology, along with more powerful image and speech processing algorithms.

These developments are going to mitigate the most frequent household activities, such as laundry, cooking, baking, refrigeration, etc. New appliances will show a certain degree of intelligence, they will be able to act autonomously, to plan and induce actions in order to achieve the requested goals, and even to learn. Communication with them is going to be user-friendly, mainly based on visual and speech acts in both directions. A variety of new sensors is

assumed to enable this kind of devices to obtain a wide spectrum of crucial pieces of information, such as the quality of washing powder, water turbidity, the level of liquid in the refrigerated bottles, etc. The machine will be able to complement the user's knowledge and wishes with its technical aspects, so that any action will be optimized through an initial user-appliance dialogue and hand-shaking procedure.

On the other hand, agent-based and multiagent theory [2,3,4] and computer-software technologies [5] have also achieved the stage, where different technical and technological solutions successfully implement them [6] and convenient design methods [7] and execution engines [8] became available. The fundamental features of agents are aimed at autonomous, collaborative, and intelligent operation [9]. Thus, combined with powerful computer systems they can serve as a firm basis for the implementation of intelligent household appliances.

In the continuation, we are going to assess the tools which have to support the implementation ideas in the field of intelligent household appliances. In Section 2, a brief overview of necessary agent structures is given. Section 3 reveals the parallels of a selected agent design methodology and agent execution environment, whereas a practical design example is demonstrated in Section 4. Section 5 concludes the paper.

2. Rational agents and BDI architecture

From the generic point of view, an agent is supposed to be able to perform adapted and autonomous actions within a dynamic, unpredictable and open environment [3]. Considering today's computer systems, they certainly meet this definition. By looking in the reverse order, this means that a computer installation may qualify as an agent. What properties do they have to show?

Agents sense the stimuli from their environment and they react accordingly. They also are proactive, persistently pursuing their goals, they adapt to unpredictable situations, and they are robust in the sense they detect and try to correct their own faults and faults of their collaborative agents. Another important property of agents is their ability to interact and build up multiagent communities [9].

Yet, the multiagent systems need more human-like characteristics in order to interact in a rational and deliberative manner. They have to mimic mental states, such as to proceed from believes, have their own wishes and goals, and be intentionally oriented to achieving them. By attributing these kinds of states to the machine, we can describe it as a rational agent with *believes*, *desires*, and *intentions*. This defines it as the so called BDI agent or BDI agent model [10].

2.1. Generic BDI model

Fig. 1 shows typical elements of generic BDI agent model.

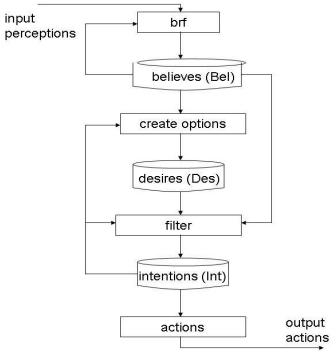

Fig. 1: Generic BDI agent model

Assume a bit more formal description of Fig. 1. If T stands for a set type, then $\gamma(T)$ represents the set's cardinality. If further $\times$ means Cartesian product, the process of updating the agent's believes may be derived as the *believe revision function* (*brf*):

$$\gamma(Believes) \times Perceptions \rightarrow \gamma(Believes).$$

Current believes and perceptions generate a new set of believes. The agent's *deliberation* is divided in two separate functions: *create options* and *filtering*. The former creates a set of alternatives, while the latter chooses among competitive alternatives. Creation of new options and filtering out one of them is formally written as:

$$\gamma(Believes) \times \gamma(Intentions) \rightarrow \gamma(Desires) \text{ and}$$
$$\gamma(Believes) \times \gamma(Desires) \times \gamma(Intentions) \rightarrow \gamma(Intentions).$$

This actually means the process of the agent's deliberation. Agents achieve their goals by applying the appropriate plans:

$$\gamma(Believes) \times \gamma(Intentions) \text{ -> } Plan.$$

When implementing an agent, it is crucial it's believes update promptly to make the agent reconsider its goals, revise them if necessary and create new plans accordingly. The model from Fig. 1 corresponds to the structure which was adopted in the design of our intelligent multiagent household assistant. The design process was based on the methodology supported by the Prometheus environment. The resulting structures are ready to be implemented in the BDI architecture simulators or execution engines. Next two sections are devoted to a case study researching the design possibilities in the filed of agent-based household appliances.

3. Design and simulation environments for multiagent systems

Multiagent systems most often control complex environments and perform a variety of rational acts. It is therefore of great importance to have a powerful design methodology at one's disposal. We want to construct a rational system of a Multiagent Washing Assistant which will be apt of simple induction and learning. The BDI agent architecture turned out to be the most suitable one. This decision was alleviated by finding several parallels and good matching between the Prometheus design methodology and the Jadex simulation environment.

3.1. Agent-oriented design methodologies

The Prometheus methodology covers the main steps in a multiagent system design. However, using it in combination with the Jadex simulation/execution environment, certain precautions must be taken. Jadex adopts classical BDI architecture and terminology, as it deals with perceptions, believes, desires, goals, intentions and plans. Prometheus generates all structures connected to goals and plans, it helps building agent internal structures, verifies the correctness of all communication links and messages, and prints valuable reports as exemplified in Section 4. It naturally suggests commencing an agent-based design by setting up the goals first. Different ways how these goals can be achieved are designated use case *scenarios* in Prometheus. Each scenario might include new goals and/or *roles*, and these roles are played by collaborating agents. So, the definition of scenarios and roles dictates the agent structure and interconnections, the input perceptions, and plans for their acting.

The Prometheus design methodology is based on the following development phases [11]:

1. System specifications:
- Recognize external influences, such as other applications and users who interact with the system
- Define system goals
- Conceive system scenarios
- Identify basic system functionalities, i.e. system roles
- For each scenario, select input perceptions and output acts
- Link each scenario to a specific goal
- Follow iteratively the hierarchy of goals (by asking How and why?) and scenarios (by inserting goals, acts, and partial scenarios)
- Combine goals and roles when the overall system architecture and organization become accurate enough
- Incorporate perceptions and acts in the roles

2. System architecture design
- Define types of agent in the system
- Connect believes (data in the databases) and roles with agents
- Determine agent interlinks and system architecture
- Set communication protocols between co-operating agents

3. Detailed design of the agents
- Define a detailed internal structure of each agent
- Incorporate roles they are supposed to act
- Implement agent plans, which means software algorithms, e.g. in Java; the plans are triggered by the agent believes, goals and internal events, whereas the execution of a plan can generate new events and goals
- Connect each agent to its environment by making it consider, and respond to, the messages according to the implemented protocols
- Finally, determine the overall hierarchy of goals to induce the sequences of plan executions.

This general Prometheus framework is always adapted to the concrete problem under consideration. The abovementioned steps are applied iteratively, so that the obtained hierarchy of goals, interconnections of goals and roles, scenario structures, agents, and eventually the overall system architecture suit our target system most effective.

3.2. *Development and simulation environment for multiagent systems–Jadex*

Jadex is called a BDI-based reasoning machine [12]. Jadex resides on the JADE platform [13]. It integrates BDI agent features by using object-oriented approach and XML-based descriptions. JADE respects the FIPA-introduced standards and takes care of all the system communications. Jadex is actually a virtual machine which interprets a user-defined hierarchy and structure of multiple interconnected BDI agents. It thoroughly supports the internal structures of believes, plans and goals. The agents are intrinsicly assumed rational, as they share the properties described in Section 2. Any Jadex-based system is linked to its environment via a built-in messaging. Input messages trigger internal events, such as updating believes or commencing a plan execution, while the output messages communicate system actions. This mechanism applies equally to all agents whose interaction is based on message exchange as well.

Jadex offers a well structured framework for all steps for a multiagent system construction. Flexible and open XML data structures are available for declaring the agents' believes and goals. At the same time, plans are supposed to be coded in Java and symbolically linked to corresponding data structures (believes and goals). Jadex interprets such user-defined systems, takes care of the agents' deliberation and updates their believes according to the external events or perceptions, register all events and corresponding arguments, and equips the user with a variety of useful statistics and graphical interpretation tools, such as an on-line display of message passing or current believes of selected agents. Jadex is thus not only a multiagent system development tool, but it is also a sophisticated testing tool, model simulation environment, and agent-oriented execution engine.

4. Prometheus methodology and Multiagent Washing Assistant design

This section exemplifies the conceptual design of a Multiagent Washing Assistant (MWA)–a household appliance whose intelligent behavior and environmental perception is meant to make laundry easy and effective. As we are designing an agent-oriented system, we resort to the agent-oriented Prometheus design methodology [11]. Eventually, it is important to consider which parts of the system should be treated as agents. We described the approach of Prometheus in Subsection 2.1.

In our example of the MWA design we made a preliminary decision in favour of a multiagent system structure. We specified one supervisory agent that is helped by the agent assistants, because of two reasons: firstly, suitable physical/hardware structure of the target system, and secondly, the agents in such organization correspond to the basic tasks of an MWA. Our desired structure of MWA is shown in Fig. 2.

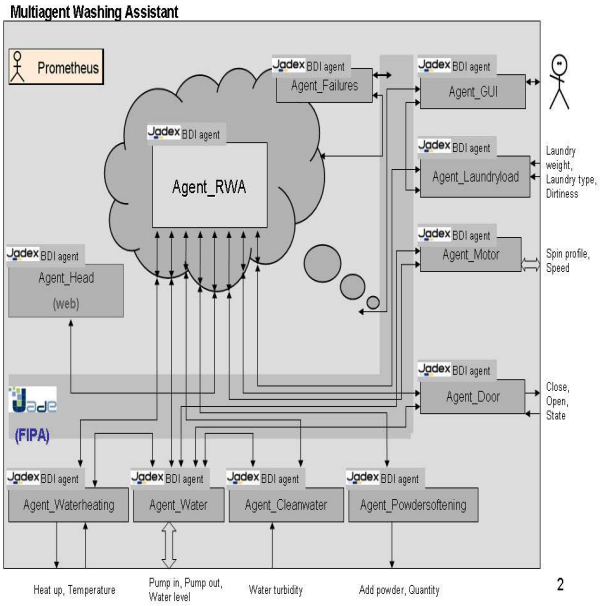

Fig. 2: The MWA organization as induced by today's physical/hardware appliance structure

An important issue linked to our preliminary decisions must be paid a special attention. We are using Prometheus for the system specification, design and consistency test phases, whereas we plan the implemention phase of an agent-oriented system using the Jadex development tool [12]. This has a significant impact to the way Prometheus is used. A general approach in Jadex considers the BDI agent model as ideally suited for describing an agent's mental state and desires (goals) representating its motivational stance as the main source for agent's actions. This strongly emphesizes Jadex as goal-oriented agent development tool. Actually, the goals are playing a central role in both tools, Prometheus and Jadex.

Jadex supports four types of goals: '*achieve*', '*perform*', '*maitain*', and '*query*' [12]. This is why we decided to adopt the same type of goals in our Prometheus design of MWA. Our design is therefore strongly goal-oriented. Each goal can have a number of plans that can be used to achieve it. Each plan can have a number of sub-goals that themselves can have multiple applicable plans. This can then naturally be depicted in a *goal-plan* tree. The children of each goal are *alternative* ways of achieving that goal (an OR structure), whereas the children of each plan are sub-goals that must all be achieved in order for the plan to succeed (an AND structure) [11].

Generally, the basic step suggested by Prometheus is to define system goals and then use them to develop use case scenarios illustrating the system's operation. An example of such scenarios is shown in part in Fig. 3.

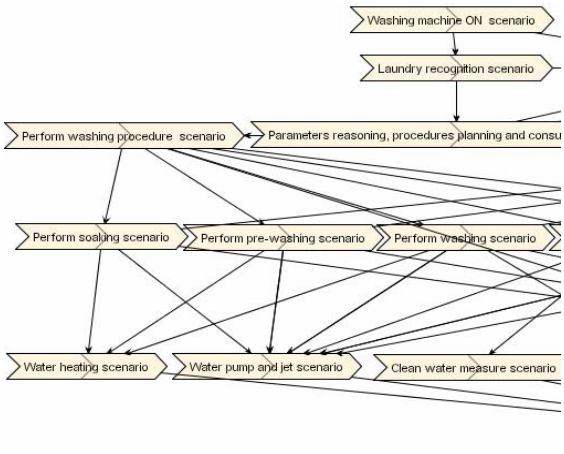

Fig. 3. MWA scenarios

Table 1 concentrates only on a single, very simple scenario of the agent for water heating (*Agent_Waterheating*). This agent originates in the general structure assumed for an MWA depicted in Fig. 2.

The next step must identify the basic functionalities of the system. Prometheus specifies separate roles to perform a scenario. A simple role of '*Water heating in the drum*' is shown in Fig. 4.

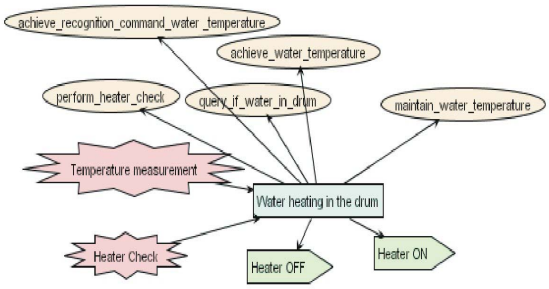

Fig. 4. The water heating role

The system top goal of our MWA is '*achieve_ most_effective_laundry_washed'*. The scenario '*Perform washing procedure scenario*' describes how this top level goal can be achieved. All other goals represent its sub-goals and are its subordinates. Describing this subordinate structure of goals, Prometheus guides the user through an iterative goal design, and provides a harmonized and verified final structure of goals. The example of our MWA is depicted in Fig. 5. Only a part of the whole system goal structure is clipped. The path marked by a thicker solid line through the goals hierarchy exemplifies which sequence of goals and plans reaches the '*maintain_water_temperature'* sub-goal. This sub-goal is one of several that must be materialized within the tree-hierarchical manner to obtain the highest level goal '*achieve_ most_effective_laundry_washed'*.

Table 1: The water heating scenario

Name	**Water heating scenario**						
Description	**Water heating system**						
Priority	**High**						
Stakeholders							
Initiated by	**System**						
Trigger	**RWA**						
Steps	**#**	**Type**	**Name**	**Role**	**Description**	**Data used**	**Data produced**
	1	Goal	achieve_recognition_command_water _temperature	Water heating in the drum	TOP GOAL (system SUB-GOAL)		Waterheatersystembelief
	2	Percept	Temperature measurement	Water heating in the drum			
	3	Goal	maintain_water_temperature	Water heating in the drum	SUB-GOAL	Waterheatersystembelief	Waterheatersystembelief
	4	Goal	query_if_water_in_drum	Water heating in the drum	SUB-GOAL		Waterheatersystembelief
	5	Goal	achieve_water_temperature	Water heating in the drum	SUB-GOAL	Waterheatersystembelief	Waterheatersystembelief

	6	Action	Heater ON	Water heating in the drum			
	7	Action	Heater OFF	Water heating in the drum			
	8	Goal	perform_heater_supervision	Water heating in the drum	SUB-GOAL	Waterheatersystembelief	Waterheatersystembelief
	9	Percept	Heater check				

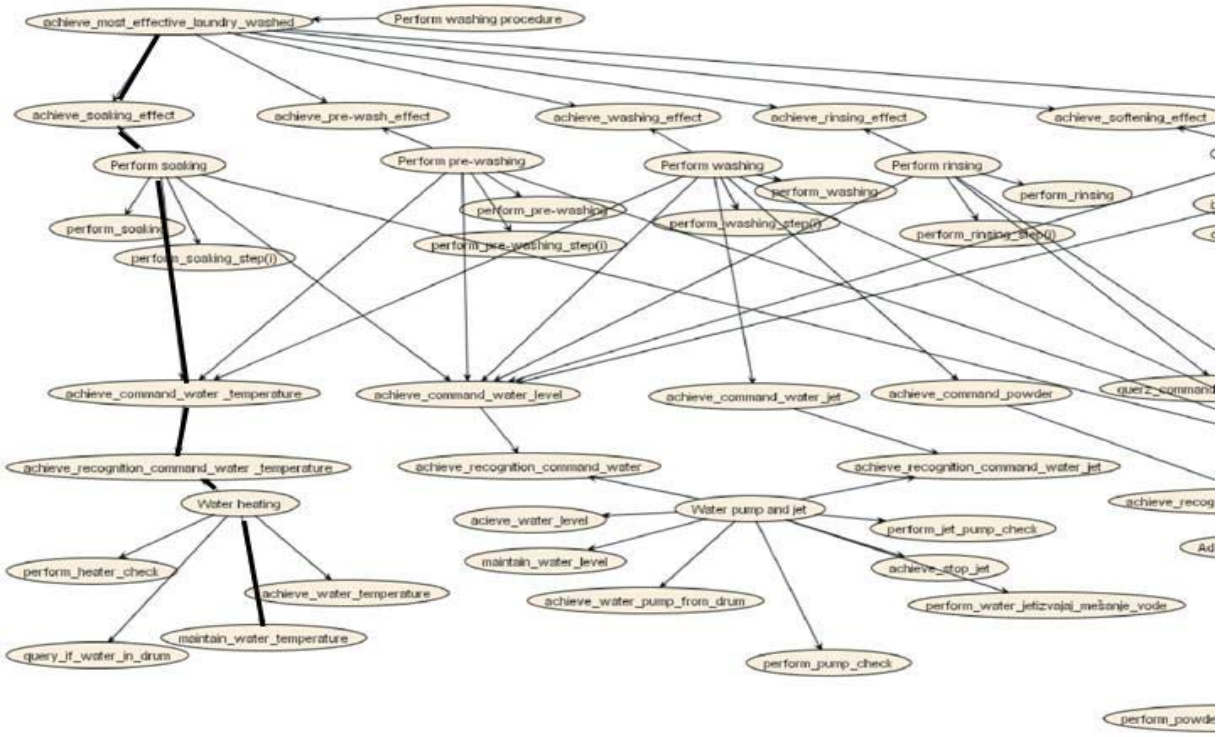

Fig. 5. A segment of the MWA goal structure

The next phase of the system design by Prometheus instructs the user to the *architectural design*, going through *agent types*, *agent descriptors*, *interaction diagrams* and *interaction protocols* to the *system overview diagram* as depicted in Fig. 6. From this figure, we can see the system diagram mimics our desired *organization structure* suggested in Fig. 2 perfectly. There are also some other steps in this basic design, such as *agent acquaintance diagram, inter-agent messages definition, shared data repositories,* but because of limited space we are not going to describe them in detail.

The Prometheus *detailed design* leads to a progressive refinement of each agent with its *capabilities, internal events, detailed data structure,* and *process specification*. Fig. 7 depicts the aforementioned simple agent *Agent_Waterheating*, whose structure exemplifies a simple AND/OR goal-plan tree. It is also obvious the agent's placement in its environment introduces perceptions, actions and messages to other agents. Internal agent messages take care of its goals which basically maintain the water in the drum heated and also perform the system check and report failure conditions.

5. Conclusion

We have briefly described the key aspects of using the Prometheus methodology in designing agent-based household appliances. The methodology has been in use for several years and we found it very appropriate for the specification and design phase of an agent-based system development, in combination with Jadex as an implementation and simulation environment. Because the both tools are strongly goal-oriented, the results of using them in BDI agent-based systems proved to be synergetic

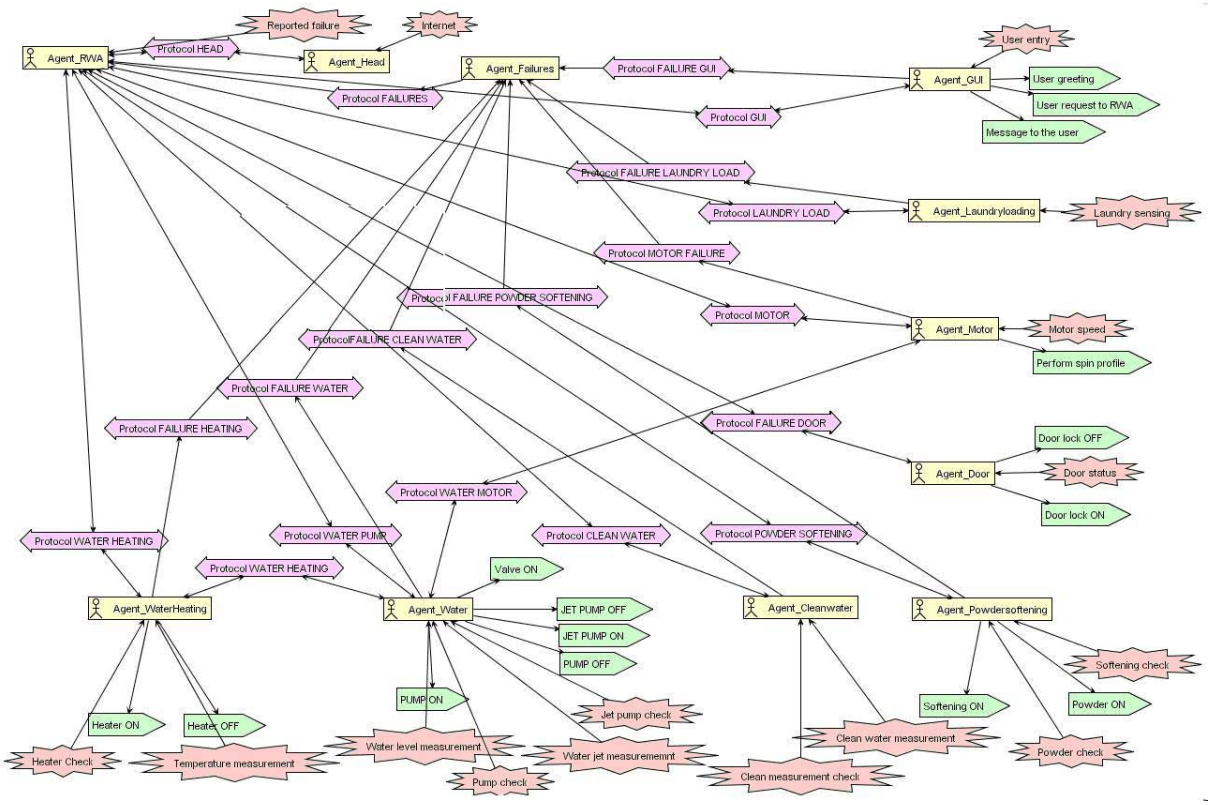

Fig. 6. The MWA system overview diagram as designed by Prometheus

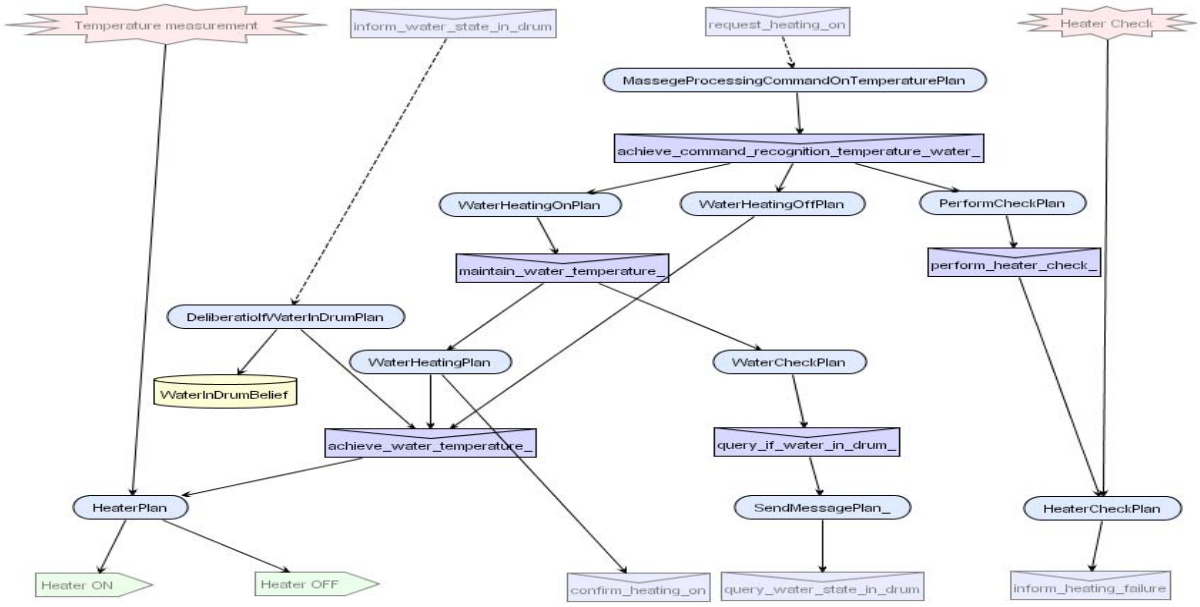

Fig. 7: Detailed design of the Agent_Waterheater agent

and gave promising results. One of the advantages of this approach is also the number of cases where the automated tools can be used for the consistency checking against various artifacts of the design process. For example, the input and output events for an agent must coincide when checked in the system overview diagram and the agent overview diagram. The final detailed design as an AND/OR goal plan tree for a particular agent, which is obtained by Prometheus, is ready to be directly coded in Jadex environment. Although Prometheus is sometimes not as user friendly as it could be, and it does not provide goal items in the detailed agent design phase (instead the inter-plan messages are used), our overall experience is very positive and we seriously believe that Prometheus is a useful and very appropriate design tool for goal-oriented BDI agent-based systems, especially in combination with the Jadex implementation and simulation environment.

Acknowledgement

System Software Laboratory from the Faculty of EE and CS of Maribor, headed by Prof. D. Zazula, is an Associate Partner of the I*PROMS Network of Excellence.

References

[1] J. Ahola, *Ambient Intelligence*, Internet publication, 2001. Accessible: http://www.ercim.org/publication/Ercim_News/enw47/intro.html.

[2] N. R. Jennings, M. Wooldridge, "Application of Intelligent Agents", In N. R. Jennings and M. Wooldridge (Eds.), *Agent Technology: Foundations, Applications, and Markets*, Springer-Verlag, New York, 1998. Accessible: http://agents.umbc.edu/introduction/jennings98.pdf

[3] S. Russel, P. Norvig, *Artificial intelligence: A Modern Approach*, Second Edition, Prentice Hall, 2003.

[4] M. J. Wooldridge, *An Introduction to Multiagent Systems*, John Wiley & Sons, Ltd., 2002.

[5] B. Hermans, *Intelligent Software Agents on the Internet: an inventory of current offered functionality in the information society & a prediction of (near-)future development*, Tilburg University, Internet reference, Tilburg, The Netherlands, 1996. Accessible: http://www.hermans.org/agents/index.html

[6] D. Kazakov, D. Kudenko, *Multi-Agent Systems and Application, chapter Machine Learning and Inductive Logic Programming for Multi-Agent Systems*, Springer-Verlag, New York, pp. 246-270, 2001. Accessible: http://www-users.cs.york.ac.uk/%7Ekazakov/papers/acai01.htm

[7] P. Massonet, Y. Deville, C.Nève, "From AOSE Methodology to Agent Implementation", *Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, Bologna, pp 27-34, 2002. Accessible: http://portal.acm.org/citation.cfm?id=544747

[8] A. Pokahr, L. Braubach, W. Lamersdorf, *Jadex: A BDI Reasoning Engine*, University of Hamburg, Chapter: *Multi-Agent Programming*, Springer Science and Business Media Inc., USA, 2005.

[9] M. J. Wooldridge, *Reasoning about Rational Agents*, The MIT Press Cambridge, Massachusetts, 1998.

[10] A. S. Rao, M. P. Georgeff, *BDI Agents: From Theory to Practice*, Technical Note 56, Australian Artificial Intelligence Institute, 1995. Accessible: http://www.agent.ai/doc/upload/200302/ rao95.pdf

[11] L. Padgham, M. Winikoff, *Developing Intelligent Agent Systems*, John Wiley & Sons, Ltd, 2004.

[12] L. Braubach, A. Pokahr, D. Moldt, W. Lamersdorf, "Goal Representation for BDI Agent Systems", *Second International Workshop on Programming Multiagent Systems: Languages and Tools*, New York, pp. 9-20, 2004. Accessible: http://vsis-www.informatik.uni-hamburg.de/publications/view.php/208

[13] F. ,Bellifemine, G. Rimassa, A. Poggi, "JADE – A FIPA-compliant agent framework", *4th International Conference on the Practical Applications of Agents and Multi-Agent System*, London, pp. 97-108, 1999.