

An intelligent domotics system to automate user actions

Natalie Kcomt Ché, Niels Pardons, Yves Vanrompay, Davy Preuveneers, Yolande Berbers

Abstract Home automation systems are ambient intelligence systems that are designed to help people proactively, but sensibly. In this paper we propose a system that learns and automates patterns in the interactions of the user with the home automation devices. We will show our approach and architecture. We use an event processing tool to handle the events from the home automation devices, prediction algorithms to predict the next action and reinforcement learning to decide which action are suitable to be automated.

1 Introduction

Today many homes are filled with home automation devices and sensors. In the future these devices will work together to form ambient intelligence (AmI) systems. A possible definition for an AmI system is as follows: a digital environment that proactively, but sensibly, supports people in their daily lives [2].

A person who lives in a house often uses the same devices every day and in the same order. The goal is to learn, recognize and automate these patterns of interactions with the various home automation devices. A classic example of such a pattern is to wake up, turn off the alarm clock and then make coffee. An intelligent learning system could learn this pattern and automatically make coffee in the morning, when the person turns off the alarm clock. The main assumption by our system is that humans follow such patterns. This has been demonstrated already in related work.

Natalie Kcomt Ché, e-mail: natalie.kcomtche@student.kuleuven.be · Niels Pardons, e-mail: nielsroger.pardons@student.kuleuven.be · Yves Vanrompay, e-mail: Yves.Vanrompay@cs.kuleuven.be · Davy Preuveneers, e-mail: Davy.Preuveneers@cs.kuleuven.be · Yolande Berbers, e-mail: Yolande.Berbers@cs.kuleuven.be
Katholieke Universiteit Leuven, Dept of Computer Science Celestijnenlaan 200A 3001 Heverlee, Belgium

When there is small noise, because of the users not following patterns, the system should not automate the action and should also not learn this noise.

The main challenge for such a system is to predict and automate enough actions in a timely fashion, avoiding that the user manually has to start these actions, while preventing to automate actions that the user does not want.

The following is a possible user scenario for our system. "Somewhere in a house in the future Sofie wakes up and turns off the alarm clock. The curtains open automatically. Sofie gets up and goes to the bathroom. The light turns on automatically as soon as she enters. The system has learned that she always showers in the morning and the water is heated to the preferred temperature. After the shower Sofie goes to the kitchen where her black coffee is already made. The system has learned that she prefers black coffee in the morning."

In section 2 we explain the architecture of our system. Related work is discussed in section 3. Conclusions and future work follow in section 4.

2 The intelligent domotics system

We adopt the principles of SOA for the creation of the services provided by the various domotic devices and sensors in the environment. Furthermore, the principles of Event Driven Architecture (EDA) are incorporated as they are used for systems that send events between independent software components and services. Events are generated by domotic devices and sensors in the environment and the learning system must respond. Our architecture uses a combination of EDA and SOA, called event driven service oriented architecture (ED-SOA)[3]. Here EDA is used to expand SOA with the publish-subscribe communication pattern and the ability to collect events over a long period of time, after which they can be processed and correlated. SOA is needed because the domotic devices provide services to each other and to the user. EDA handles the events that the devices generate asynchronously and that need to be collected over a long period of time.

Figure 1 shows the design of the learning domotics system, consisting of four different layers. The physical layer contains the various sensors and home automation devices. The communication layer provides communication between the different devices. The information layer retrieves information and higher level abstraction from the sensors and home automation devices. Finally, the decision layer controls the devices. We will first discuss the information layer and then the decision layer.

The information layer includes the event processing tool, Esper, which receives, processes and forwards the events to the other components. Esper can do filtering, but could also aggregate events to represent more hierarchical or composite events. The prediction algorithms in this layer will predict the next action based on recent history. We will use the Jacobs-Blockeel algorithm[6] and the FxL algorithm[5] for the prediction. Jacobs-Blockeel is based on IPAM, which uses a first order Markov model. Jacobs-Blockeel uses a mixed order instead to calculate the probability distribution for next event. Initially first order Markov models are used. Whenever the

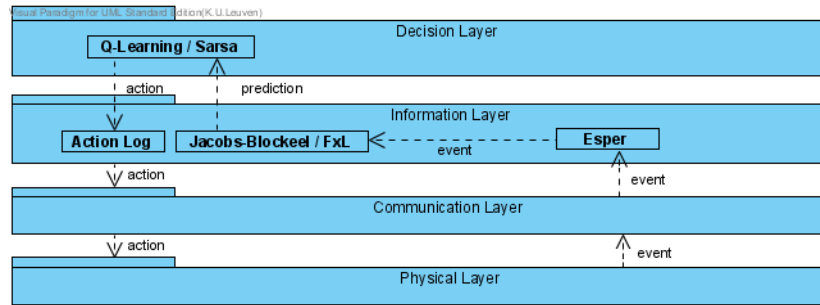


Fig. 1 The design of the learning system

algorithm makes a correct prediction, a higher order is activated, otherwise the order does not change. For example if the system correctly infers the rule that after observing 'a', 'b' will be observed then the system treats 'b after a' as a single observation. The advantage of using a mixed order Markov model is that high orders are only used when necessary, which benefits the storage and processing requirements. Jacobs and Blockeel claim that the highest order for the Markov model is not always the best choice to determine the probability for the next event. We chose for the Jacobs-Blockeel algorithm because in a home automation environment there are lots of possible actions to take. Using a system such as IPAM that only takes the most recent action into account is not sufficient here. Furthermore the Jacobs-Blockeel algorithm but mixes orders intelligently treating frequently occurring sequences as single observations.

FxL is an on-demand approach for combining the results of different order Markov models. It is based on an n-gram tree that contains the frequencies of different input subsequences with a length up to a specified value k . These n-gram models assign a score to each symbol, which represents the probability that a symbol appears next in the input sequence. This relatively new algorithm yields very good results in the context of predicting commands in a Unix terminal. FxL was chosen because it is able to get these good results while keeping the storage costs limited by the specified k and the amount of possible user actions, whereas Active LeZi, with similar results, has storage costs that grow with the dataset size.

The decision layer contains the decision algorithms Q-learning and SARSA to determine when to execute which action. The prediction of the algorithm is recorded in the state of the Q-learning algorithm. The Q-learning algorithm is a reinforcement learning algorithm that is able to compare the different actions to take without explicitly modeling the whole environment with very good results. SARSA is a variant of Q-learning that does not make the assumption that the optimal policy can be followed. The final action to be executed is sent by the Q-learning algorithm to the home automation device in question. This layer works on top of the prediction system as an extra precaution to protect against possible errors the prediction system might make.

3 Related work

MavHome[1] uses the Active Lezi (ALZ) prediction algorithm and its accuracy is improved by using frequent patterns discovered through Episode Discovery (ED). The decision algorithm is TD(0) reinforcement learning. The iDorm[4] system extracts fuzzy membership functions and fuzzy rules from the user data to model the behavior of the user and employs a fuzzy logic controller to determine which action to automate. The Adaptive House[7] uses artificial neural networks to predict the next state of the house. It balances energy cost and discomfort cost using Q-learning to determine which action to automate. The design of our system is inspired by MavHome. Our system differs from MavHome in the use of an event processing tool to process events and the use of other learning techniques: ED is not employed and Jacobs-Blockeel and FxL are used instead of the ALZ algorithm. Furthermore, a variant of Q-learning, SARSA, is implemented as the decision algorithm.

4 Conclusion

We proposed an intelligent domotics system that learns patterns in the interactions of the user with the home automation devices and then automates these interactions. The architecture of the system is a combination of EDA and SOA, also called ED-SOA. For learning and automating user actions, we apply various machine learning techniques: Jacobs-Blockeel, FxL, Q-learning and SARSA. These algorithms decide what action should be performed. Concerning future work, the system will be implemented with these different algorithms, and be tested and evaluated with both synthetic and real data from MavHome. We will compare the performance of the two prediction algorithms and of the two decision algorithms. Finally, we will compare our system with other existing projects.

References

1. Diane J. Cook. MavHome: An Agent-Based Smart Home. In *PERCOM '03*, page 521, 2003.
2. Diane J. Cook. Review: Ambient intelligence: Technologies, applications, and opportunities. *Pervasive Mob. Comput.*, 5(4):277–298, 2009.
3. Sadhana Yogesh Ghalsasi. Critical success factors for event driven service oriented architecture. In *ICIS '09*, pages 1441–1446, 2009.
4. Hani Hagraas. Creating an Ambient-Intelligence Environment Using Embedded Agents. *IEEE Intelligent Systems*, 19(6):12–20, 2004.
5. Melanie Hartmann and Daniel Schreiber. Prediction Algorithms for User Actions. In *LWA*, pages 349–354, 2007.
6. N. Jacobs and H. Blockeel. Sequence Prediction with mixed order Markov chains. In *Proceedings of the Belgian/Dutch Conference on Artificial Intelligence*, 2003.
7. M. Mozer. The neural network house: an environment that adapts to its inhabitants. In *Proceedings of the AAAI Spring Symposium on Intelligent Environments*, pages 110–114, 1998.