Communicating Process Architectures 2012
P.H. Welch et al. (Eds.)
Open Channel Publishing Ltd., 2012

A Distributed Multi-agent Control System for Power Consumption in Buildings

Anna Magdalena KOSEK [1], and Oliver GEHRKE

Intelligent Energy Systems, Dept. of Electrical Engineering,
Technical University of Denmark

Abstract. This paper presents a distributed controller for adjusting the electrical consumption of a residential building in response to an external power setpoint in Watts. The controller is based on a multi-agent system and has been implemented in JCSP. It is modularly built, capable of self-configuration and adopting to a dynamic environment. The paper describes the overall architecture and the design of the individual agents. Preliminary results from proof-of-concept tests on a real building are included.

Keywords. agent system, JCSP, distributed control, home automation, smart building.

Introduction

According to the Danish Energy Agency [1], electricity generation from renewable energy has increased by 76.7% between 2000 and 2010. In 2010 solar and wind energy accounted for 21% of all electricity production in Denmark. The Danish government's target is to reach a wind energy penetration of 50% by the year 2020. With an increasing share of electricity generation from fluctuating sources, other players on the electricity grid need to participate in balancing generation and demand in the system. Dynamic control of the electricity demand, following the needs of the power system, has become a main focus of research interest in the area of smart grids.

In 2010, the electricity consumption of Danish households was 10.2 TWh [1] which accounts for 31.8% of the overall national consumption. At least part of the residential consumption is assumed to be flexible, that is the demand can be controlled or shifted in time, for example because energy can be stored in thermal masses. In order to enable this potential, control infrastructures are needed to coordinate the actions of many of these loads.

Today, the electrical loads in a building are independent of each other and directly controlled by its inhabitants. Heating is activated when people feel cold, lights are being switched on when there is not enough light in the space, other devices are being operated depending on users preferences and needs. In many cases, these preferences do not actually include the need to have the device operate in a very specific way, as long as the outcome meets certain criteria. For example, inhabitants of a building are not usually interested in the thermostatic cycling of a space heater, as long as the temperature stays within an acceptable range.

Using a building automation system, devices can be controlled and coordinated at the building level. The building automation system itself provides a point of access to controlling demand from the grid side. Not all power consumption in a building can be controlled in this way though; many devices do not offer any exploitable amount of flexibility. Ideally, a residential demand controller should adapt to the behaviour of uncontrollable devices, take

[1]Corresponding Author: *Anna Magdalena Kosek, DTU, Intelligent Energy Systems, Frederiksborgvej 399, 4000 Roskilde, Denmark*. Tel.: +45 40191792; E-mail: amko@elektro.dtu.dk.

advantage of the flexibility of those devices which are controllable, and all the time maintain user comfort.

Most home automation systems currently on the market are centralized: a single controller operates all devices available in a household. A controller of this type usually requires detailed information about all devices and a direct communication channel to each device. If the system setup changes after the time of installation - devices are added or removed, new types of devices appear or the layout of the building changes - manual reconfiguration is needed. For such dynamic systems where forward-compatibility is important, decentralized controllers can offer additional flexibility.

In a decentralized control system, intelligence and decision making mechanisms are embedded in many devices. Devices can be provided with the ability to reorganize and to cooperate towards a common goal. An overall system behaviour emerges from the combined behavior of the individual decision making mechanisms. Examples of distributed home control systems based on self-organization can be found in [2,3].

On the minus side, a distributed architecture requires more effort in the design process, and an optimal system-level control response may be more difficult to achieve than in the centralized case.

In this paper a distributed approach to control was chosen, exploring self-organization, reconfigurability and adaptation of a network of power-consuming devices. A building is required to follow external power setpoint and organize devices operation so as comfort maintained as best as possible with available power. Setpoint, in the context of the Control Theory, is a reference value that a controller attempts to reach. In the implementation of the architecture presented in this paper a house is treated as a controllable load, and it is expected to follow the setpoint as close as it is possible. If the power consumption setpoint is to high, the house is till obliged to follow it and distribute the excess power over controllable devices. The flexibility of power use is considered to be a service to the power system, in this paper the presented flexibility in consumption is extreme, more conservative approach can be chosen within the same architecture.

Distributed multi-agent approach used for energy distribution was presented first in [4], since then many home automation systems aimed at energy conservation while maintaining user comfort with use of various multi-agent distributed system architectures [5,6,7,8,9]. Home automation system presented in [10,11] highlights usability of agent negotiation to match local production from renewable power resources with house power consumption needs. In this work the power management system performance highly relies on weather,and user behavior prediction, while adaptable schedule assumes well described power consumption models for particular devices. In extreme situations hard constrains of user comfort, does not allow match production and consumption. In case of lack of predictions and accurate device models, system is unable to find optimal control solution. A more flexible approach to control an unknown set of devices, with inaccurate power consumption prediction, taking under consideration users spontaneous decisions to operate devices at any time, is presented in this work, exploring scenarios close to limits of a house power consumption flexibility.

Many existing approaches, for example these presented in [8,9,12], highlight usability of multi-agent system to simulate a home environment, with fixed number and type of devices, not taking under consideration equipment faults, speed of reaction to signals and variability of the baseline load in the building. In this work experiments with a real building were performed, with set of actual devices changing their state from controllable to uncontrollable, adapting to unknown base consumption, performing a service to the power system while maintaining user comfort.

In the next section of this paper we provide a classification of devices based on their controllability and use. In section 2 system considerations and design choices are explained. Section 3 describes the design of the device controllers and introduces the building blocks

of the systems and their interactions. In section 4 we describe a concrete implementation of the proposed architecture, using JCSP. A proof-of-concept experiment on actual building hardware, and results obtained from this experiment are presented and analyzed in section 5.

1. Classification of household devices

The primary purpose of household appliances and HVAC systems in residential buildings is defined by the local needs of their users; providing balancing services to the power grid has to count as an afterthought. Therefore, controlling these devices should not conflict with their primary tasks. The following classification, developed as part of the iPower project [13], lists devices based on their use, and extends work presented in [14]. At the highest level, devices can be classified into two categories, controllable and uncontrollable.

1.1. Controllable devices

A device is called (remotely) controllable if its internal controller can be affected by an external signal. If the communication is bidirectional, a device can provide an acknowledgement or feedback, or negotiate the request. In case there is no feedback, a device may be obliged to act as requested, or it can either react or ignore the request depending on internal logic. The controllable devices can be further divided into the following subcategories:

Flexible - devices that can be controlled at any time, consuming a variable amount of power. These devices are usually used every day, or their use is strongly influenced by external factors such as weather conditions, and their usage patterns can therefore be predicted. Examples of devices in this category include electrical space heaters or air-conditioners.

Programmed - devices that, once switched on, will operate for some period of time according to an internal program. The switch-on time may be shifted depending on an external request. Examples here would be washing machines or dishwashers.

1.2. Uncontrollable devices

A device is called uncontrollable if it cannot be controlled by an external signal; this category can be further divided into two subcategories:

Spontaneous use - devices that are available to the user at any time. These devices usually have a high priority because their use is based on a consumer desire. It is usually difficult to predict when these devices will be used. An example of a spontaneous use device is a TV set.

Base load - devices which are usually active at all times. Examples in this category could be home alarm systems or ventilation fans.

A classification based on device use is very important when considering user comfort and intent. Because attention is turned towards the primary purpose of a device and user intention, classifications like these can help to ensure user comfort and satisfaction.

All controllable devices can offer their flexibility to the power grid, as long as user comfort is not significantly reduced. Flexibility in this context relates to a controlled increase or decrease of power consumption. A single building may contain a variety of different device types.

Power consumed by base load can be estimated, operation of spontaneous use devices can be compensated with reconfiguration of controllable devices and adaptation. Programmed devices can, once turned on, maintain their state, and flexible devices can compen-

sate for the rest of the devices, while following a power setpoint request. This work presents a controller that is able to manage different types of devices, respect users preferences and adapt to external requirements.

2. System Architecture

In the following, we describe a design for a distributed architecture which is capable of reconfiguration, adaptation and self-organization while executing a set of two control tasks: following an externally requested power setpoint and the control of indoor temperature in the rooms of a building. The system needs to perform several independent tasks, and should handle interactions between actors with different goals, needs and priorities. These conflicting interests and interactions between different actors need to be represented in the architecture. Since these tasks can be identified, they can be deployed in separate actors responsible for performing some actions.

Multi-agent systems are an attractive option for implementing distributed systems. Several independent tasks can be performed by different agents, deployed across different devices. In a dynamic environment where devices may appear and disappear from the control system, robustness can be achieved by redundant agents. If the responsibilities of failed or disappearing agents can be taken over by other agents, the system can tolerate partial failures and reconfigure. Multi-agent systems can also be built to handle a heterogeneous set of components which may not need to be known at design time. These features are desirable for a modular end extensible distributed home automation system.

In the chosen design, the building indoor temperature controller is distributed and based on agents which encapsulate different functionalities and cooperate in order to meet a system-wide behavior goal. The proposed architecture consist of many managers, negotiators, operatives, cores and users, and one smart meter present in the space, as shown in figure 1.

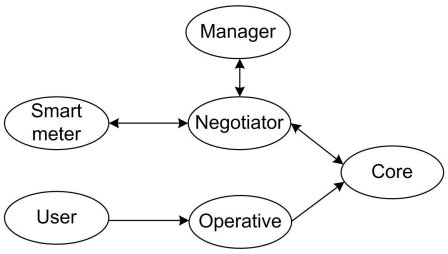

Figure 1. Agents and their cooperation.

Manager is an agent that is responsible for managing negotiations between different devices in the building.

Negotiator is an agent responsible for starting and maintaining negotiation from the point of view of a single device.

Operative is an agent responsible for receiving and passing direct commands and signals to a device, that triggers no negotiation.

Core is an agent responsible for controlling a device directly and its behavior depends on a device type and implemented control algorithms.

Smart meter is an agent responsible for passing power setpoint to all the devices in the house and matching it with actual power consumption.

User can control all devices in the space, overriding automatic settings.

The controller described in this paper is a system regulating indoor temperature and following energy consumption setpoint in Watts that is requested by a smart meter. The power

setpoint can be sent to the house by a power utility, performing regulation of power consumption of an aggregation of many buildings. The system is driven by signals from a smart meter; if the meter is not present the control system cannot provide services to the power system.

Several assumptions about devices availability were made in this architecture. Presence of a smart meter is needed, with total power consumption readings, temperature sensors distributed in the space and presence sensors in rooms are also needed. This input data is used to adapt energy consumption, adjust temperature and prioritize keeping setpoint temperature in occupied rooms.

The smart meter agent receives the power setpoint for all building and its task is to match the requested setpoint to the actual power consumption. The setpoint is passed to all devices and received by Negotiator agents. The smart meter have no control over what devices will be chosen in the negotiation to match the requested setpoint, but it can observe the impact of its request to the house and react to the aggregated power consumption of all devices. Since smart meter can observe the overall house consumption, including all uncontrollable devices, both spontaneous use and base load types, it can adjust its power setpoint request in order to trigger the desired reaction. For example, if the Smart meter agent requested 5 kW from the household, and after some time it observed that consumption is 4 kW it issues a new request for 6 kW, therefore 1 kW more than previously requested, to find more devices willing to operate. A reason that devices consume less or more than they promised could be device failure, wrong estimation of power use or appearance of uncontrollable devices in the space. In any case the network of operating devices need to be reconfigured to match the new setpoint with the addition of an error of 1 kW. Smart meter adaptation techniques allow the system to match the power setpoint better while taking into consideration disturbances that can appear during operation.

All devices available in the space calculate their own urgency to operate, this factor varies between 0 and 100. The calculation of this factor is presented in section 3.4. Based on this factor and the power setpoint provided by the smart meter devices are allowed or refused to operate for a single negotiation period. The negotiation process is described in section 3.1. In a default state devices are obliged to respect the outcome of a negotiation they participated in. If a device chooses to operate otherwise or a User agent takes over its operation, the control system is able to react and reconfigure devices available for negotiation. In this case a device changes its type from flexible to spontaneous use. A User agent can control a device using an Operative agent by influencing the chosen device directly.

3. Device design

As the cost of micro-controllers decreases and their capabilities grow, an increasing number of household devices are equipped with embedded controllers. With present technology, such devices are able to serve multiple concurrent processes. A multi-agent system was chosen as a paradigm for modeling the behavior of the device controllers. The multi-agent system consists of five types of agents, one system-wide smart meter agent and four agent types which exist on the distributed devices (see figure 2). Responsible for device functionality are three agents called Negotiator, Operative and Core. A Manager agent is also associated with each physical device, but it does not have any direct interaction with the device state. The reason for having one Manager agent per physical devices is robustness: to ensure that, regardless of the number of devices present or active, there is always at least one Manager agent available.

In the following section we will give a detailed description of the above agents, their functionality, election, decision making and information exchange mechanisms.

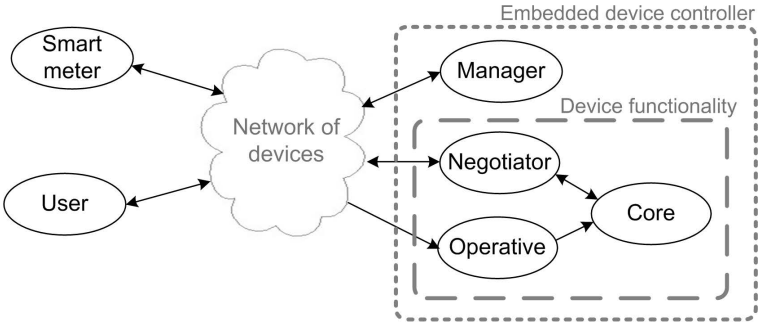

Figure 2. Device agents: Manager, Negotiator, Operative and Core.

3.1. Negotiator

The interaction between agents can be described as a sequence of transactions, each of which contains a negotiation process. A new negotiation is triggered when the smart meter receives a new power setpoint request from an external source. The Smart meter agent broadcasts the setpoint to the house network on which all negotiator agents listen. Upon receiving a setpoint, the Negotiator agent changes its state from idle to negotiation (figure 3) and sends a request for starting a new negotiation round to all available Manager agents. These proceed to elect a responsible manager for the transaction (see section 3.2).

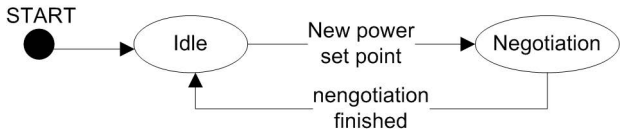

Figure 3. Negotiator agent state diagram.

The Negotiator agent then waits until it receives a notification from the elected transaction manager after a decision about a power consumption schedule for the devices has been made. The negotiator then returns to idle state. Since the negotiation is only two-stage process, the is no timeout needed, device waits for the decision of a particular negotiation. If the decision does not arrive or the connection is lost, the Negotiation agent waits for another trigger for negotiation from Smart meter or user input.

3.2. Manager

Manager agents are responsible for executing the negotiation process within a transaction. They can be in one of three states: idle, election or supervision (figure 4).

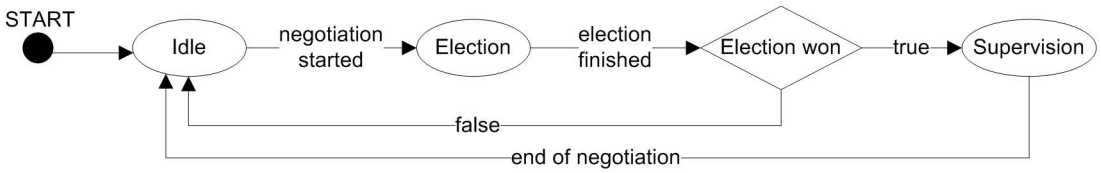

Figure 4. Manager agent state diagram.

The Manager agent maintains a list of all ongoing negotiations and tracks their state to be one of the following:

- *started* - a negotiation has been requested but not initiated yet.

- *active* - a negotiations has been *started* and is supervised by the local manager.
- *dropped* - a negotiation has been *started* and is supervised by another manager.
- *finished* - a negotiation has previously been *active* and has now finished executing.

The list of negotiations is used to manage the state of negotiation and to calculate statistics. When a Manager agent receives a negotiation request, it triggers an election mechanism and changes the state from idle to election. The Manager then calculates a personal quality factor. The quality factor, q_{m_i} of a manger m_i is composed of three parts:

- $\phi(m_i)$ - fairness function, proportional to the number of negotiations (both active and finished) managed by the local manager, relative to the total number of negotiations,
- $\varphi(m_i)$ - occupancy function, discrete function determining if the manager m_i is already busy supervising another negotiation,
- α_{m_i} - random number between 0 and 100.

The quality factor is calculated as follows:

$$q_{m_i} = \lambda_1 \times \phi(m_i) + \lambda_2 \times \varphi(m_i) + \lambda_3 \times \alpha_{m_1} \tag{1}$$

Values for λ_1, λ_2 and λ_3 were empirically determined to be 0.5, 0.25 and 0.25. The quality factor calculated using equation (1) is sent to all managers competing for supervision over a particular negotiation. The Manager agents then wait for quality factors from all other managers. After a fixed election time interval, all managers independently compare quality factors and choose a manager for the negotiation. For the experimental work presented in this paper, an election time interval of two seconds has shown useful. The Manager agent with the highest quality factor sends a notification to the other participating devices, indicating the start of the negotiation and claiming to be the responsible manager. If two or more managers claim equally large quality factors in a single election, a reelection mechanism is triggered between the two competing managers. Due to the random factor α_{m_i}, the renegotiation will eventually be finished. The Manager agent winning the election changes its state to supervision.
In the supervision state the Manager agent polls the core agents for information about the power ratings of the devices they manage, and about their urgency to be activated. The urgency is determined by every device (see section 3.4). Another fixed time interval is allocated for this process - for the work presented in this paper, five seconds were chosen. Based on the information received, the Manager sorts the device list in order of urgency. Devices are then taken from the top of the list until their cumulative power rating matches the desired power setpoint as closely as possible. These devices then receive a mandatory activation message.

At the end of the negotiation process, the Manager returns to the idle state and notifies the negotiator agent.

3.3. Operative

The sole purpose of the Operative agent is to receive and forward direct commands and signals which require no response from a device, to the core agent. These signals override decisions made by the device during negotiation, and can be triggered by an external device or user, adjusting the device behavior. Once a command from an Operative agent is issued to the core agent, the associated device is obliged to keep the requested state. The device will then not participate in negotiations, and changes its demand type from controllable to spontaneous use.

3.4. Core

Core agents are responsible for controlling the physical resources of a device. It maintains information about the device type and controls its behavior accordingly. The Core agent of a

device d_i calculates a factor of urgency to operate u_{d_i}, composed of four components:

- $\phi(d_i)$ - fairness function. This discrete function determines if the device d_i is in operation. If the device is being used, its urgency to operate decreases, allowing other devices to be activated.
- $\varphi(d_i)$ - difference between actual and desired comfort in the environment influenced by a device. Depending on the device, comfort may be expressed in terms of temperature, lighting conditions, humidity or other measurable comfort indicators.
- $\chi(d_i)$ - room occupancy. This discrete function reflects if presence sensors indicate that space is used. If no sensor is present, it is assumed that the space is not occupied.
- α_{d_i} - random number between 0 and 100.

The urgency factor is calculated as follows:

$$u_{d_i} = \lambda_1 \times \phi(d_i) + \lambda_2 \times \varphi(d_i) + \lambda_3 \times \chi_{d_1} + \lambda_4 \times \alpha_{d_1} \tag{2}$$

Useful values for λ_1, λ_2, λ_3 and λ_4 were determined to be 0.1, 0.7, 0.1 and 0.1. If a device cannot be switched off, because it is already performing an ongoing action, the urgency factor u_{d_i} is set to maximum. In this way, while the future use of a device is not known yet, the probability that the device will be chosen is high. If there are many devices which need to be used continuously, and the power setpoint is too low to operate all of these devices, some equipment has to be switched off.

All devices are requested to work towards controlling the power consumption of the building close to the setpoint while maintaining comfort in the space. The agents in the proposed architecture enable negotiation, dynamic reconfiguration, fair distribution of available power and adaptation to an external schedule. The architecture itself is scalable and does not depend on the number of devices. Communication bandwidth might pose scalability problems, since all messages are being broadcasted, but the size of the messages is small and they are exchanged seldomly. An assumption was made that the number of devices sharing a single space may not exceed 100. Since no bandwidth problems are expected with this number of devices, broadcast messages were chosen for communication.

Another important feature of the proposed system is its expandability. Since most decisions are made inside the devices, new devices types can appear in the space, as long as they adhere to the common interface for communicating with the existing devices. The network of devices can be extended and reduced during operation without manual configuration effort. Spontaneous use devices can be turned on at any time, as the control system reacts by decreasing and increasing consumption. Section 4 presents details of the presented software architecture.

4. Software architecture

For the implementation of the multi-agent system, a software architecture based on autonomous parallel processes on top of a message passing system was chosen. The indoor temperature control system was developed using JCSP (Communicating Sequential Processes [15] for Java [16]). JCSP uses Java threads to provide a concurrent programming environment in which processes execute and communicate simultaneously based on the CSP model. The usefulness of JCSP as a software architecture for pervasive environments has been shown in [17]. Its process oriented approach is appropriate for implementing autonomous agents in a message based system; a similar implementation of a distributed messaging system based on JCSP Agents can be found in [18].

CSP was chosen to model device architecture consisting of interconnected blocks encapsulating different functionalities, as shown in figure 5. The process-oriented software ar-

chitecture was chosen to ease the implementation and enable possible extensions to device functionality in the proposed architecture. Since the presented home automation architecture is broadcast based and CSP does not offer a broadcast channel, a repeater was implemented to propagate messages between devices. The JCSP implementation of the proposed architecture, as described in section 5.1, consists of 94 CSP processes, running 142 Java Threads, the code consists of 2305 lines distributed over 18 Java classes.

It is possible to add and remove devices in the house and they can dynamically join and leave the negotiation in the proposed architecture, but the specific implementation of the repeater need to be changed. In this implementation repeater use static CSP channels to communicate with devices. If devices can be addressable with use of IP, the repeater can use simple Java sockets and broadcast messages to the entire sub-network. In this case the number of devices is only limited by the size of the sub-network, and leaving an joining the network can be executes seamlessly.

Instances of the indoor temperature control system are installed on all participating devices in a building. In addition, there are single instances of two special processes: smart meter and a user agent. The single smart meter process is associated with a power meter. Its implementation is simple, since it only reacts on one of two triggers: arrival of a new external setpoint, and a time trigger to perform an adaptation at regular intervals. In the adaptation, the smart meter process calculates the deviation between the external setpoint and the actual consumption. If the difference is too large, corrected setpoints are calculated and distributed to the negotiator agents. The errors between the desired and the actual consumption are logged and used in the adaptation algorithm.

The user agent is split into a user process and a GUI (Graphical User Interface). The user process can receive requests from the GUI and distributes them to the operative agents if the user wants to override the control system's actions by directly activating or deactivating devices in a selected room, and The user intervention in operation of a device excludes it from the negotiation process. The number and types of devices depends on the room (see figure 7).

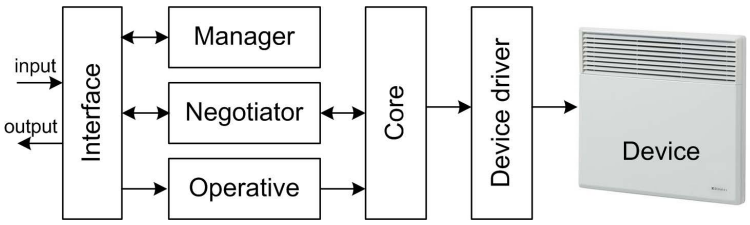

Figure 5. Detailed device architecture, including manager, negotiator, operative and core agents as well as interface and device driver modules.

All power consuming devices contain identical instances of the manager, negotiator and operative agents, as well as the interface component. Only the core agent may be device specific, depending on the needs of the device driver.

The interface process is responsible for passing messages between devices, smart meter and users. Once a message arrives, the interface forwards it to the specified instance stated in the message destination: manager, negotiator or operative agent. The manager agent sends and receives messages for the election and negotiation mechanism described in sections 3.2 and 3.1. The negotiator agent receives messages from the smart meter and manager agents as part of the negotiation sequence. The operative agent only receives messages from the user process and sends messages with direct commands to the core process.

The core process receives messages from the negotiator and operative processes; if requested it can return information about its state, urgency factor or availability for negotiation.

Figure 6. The PowerFlexHouse building.

If requested, the core process changes the state of its associated device through the device driver component.

Device driver is a feature of the PowerFlexHouse platform implemented at DTU Electrical Engineering, enabling sending commands and receiving signals from devices present in the building.

5. Proof of concept

The proposed algorithm has been implemented and tested in the PowerFlexHouse facility at Technical University of Denmark (DTU).

5.1. Experimental setup

The PowerFlexHouse is a small, free-standing office building which has been converted into a platform for research on demand side management. The building contains seven offices, a meeting room and a kitchen. The electrical load of the building has a peak consumption of about 20kW and consists of heating, lighting, air-conditioning, a hot-water supply system as well as various household appliances, such as a refrigerator and a coffee machine.

All loads in the building have been equipped with wireless actuators and can be switched on and off by the central building controller, an embedded PC. The same controller is able to receive input from a multitude of sensors inside and outside of the building (figure 7). Each room is equipped with a temperature sensor, motion detector, wireless light switch, window and door contacts. Additional data is supplied by a meteorology mast with instruments for wind speed, wind direction, solar irradiation, humidity and temperature. The building is an integral part of the SYSLAB [19] smart grid research facility and can receive its electrical energy either from the public grid or from various renewable and non-renewable sources within the facility.

The experiment uses a setup consisting of 15 actuated devices (10 electrical heaters and 5 air-conditioning units), each associated with an instance of the agent platform. Eight temperature sensors - one in each room - provide control feedback to the core agents. Some devices are interdependent, because they influence the temperature in the same room. They also share the same temperature sensor. To represent these relations in the software, virtual rooms were introduced.

The smart meter agent is associated with a power metering device which registers the electricity consumption of the whole building. Devices communicate using broadcast messages. For the purpose of the experiment, all core agents are programmed to maintain a fixed target temperature of $20°C$ in each room, with a permitted hysteresis of $\pm 1°C$ (comfort range).

To simplify the experiment, the power setpoint is generated in the smart meter agent itself as a random walk. The generated setpoints fall in an interval between 3 and 11 kW, with a maximum step size of 3 kW. Setpoints change every 10 minutes, and the system

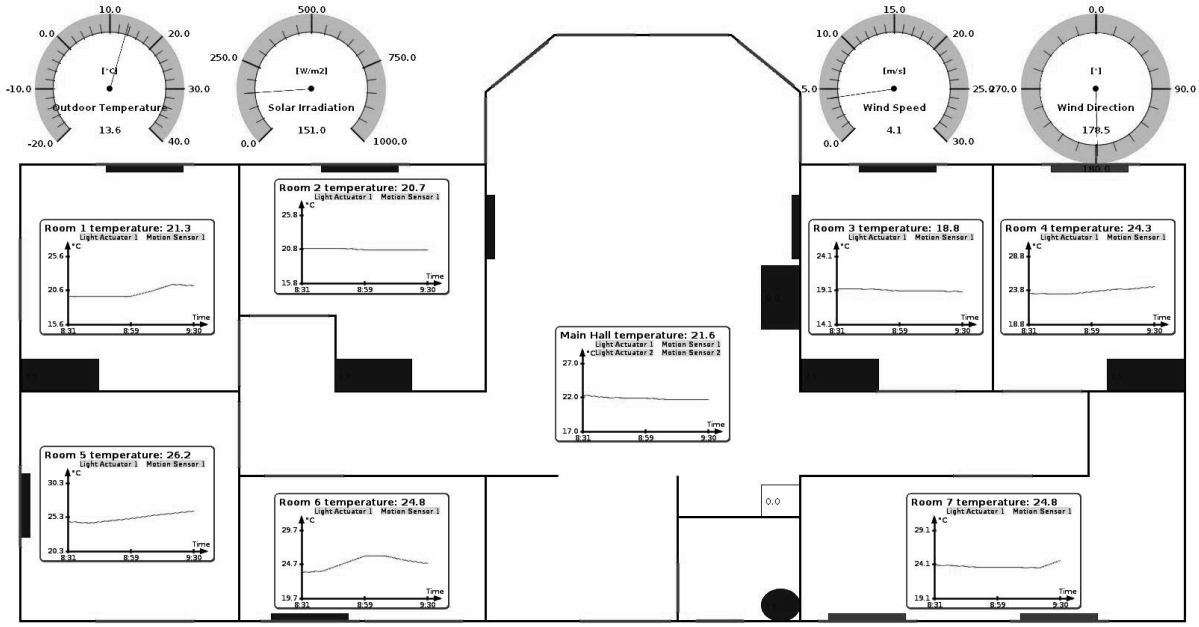

Figure 7. The PowerFlexHouse building layout, with electric heaters in every room and air-conditioning in rooms 1,2,3,4 and in the main hall.

adapts every minute. The PowerFlexHouse platform is centralized by design, for example it does not have decentralized computing capabilities located at the controlled devices. Instead, heaters and air conditioning devices are remotely controlled from a single building controller. Due to these limitations of the experimental platform, all agents are running on the same physical computer. However, the implementation is network transparent and does not contain any dependencies which would prevent agents from be implemented on separate distributed devices and communicate via a wireless connection. At present, loads in PowerFlexHouse are not equipped with sensors to measure their own consumption in real time. For the purpose of the experiment, an average power consumption value was estimated for each device, based on previous measurements (Table 5.1).

Table 1. Estimated device power consumption in PowerFlexHouse.

Device name	Room	Estimated power [kW]	Revised estimate [kW]
heater 1	main hall	1	1
heater 2	main hall	1	1
air-con 1	main hall	1	0.35
heater 3	room 1	1	1.25
air-con 2	room 1	1	0.35
heater 4	room 2	1	1.25
air-con 3	room 2	1	0.35
heater 5	room 3	1	0.75
air-con 4	room 3	1	0.35
heater 6	room 4	1	0.75
air-con 5	room 4	1	0.35
heater 7	room 5	1	0.75
heater 8	room 6	1	1
heater 9	room 7	1	0.75
heater 10	room 7	1	1.25

5.2. Results

Three experiments were run, an initial one for calibration and two main experiments for data collection. All three were run with the same set of devices. This section discusses the setup of the experiments, observations and analysis of the collected data.

5.2.1. The initial experiment

In the initial experiment no adaptation is performed by the smart meter agent; the devices receive purely random setpoints. It can be seen in figure 8 that the system reconfigures upon every request, but the actual consumption does not match the requested consumption very well.

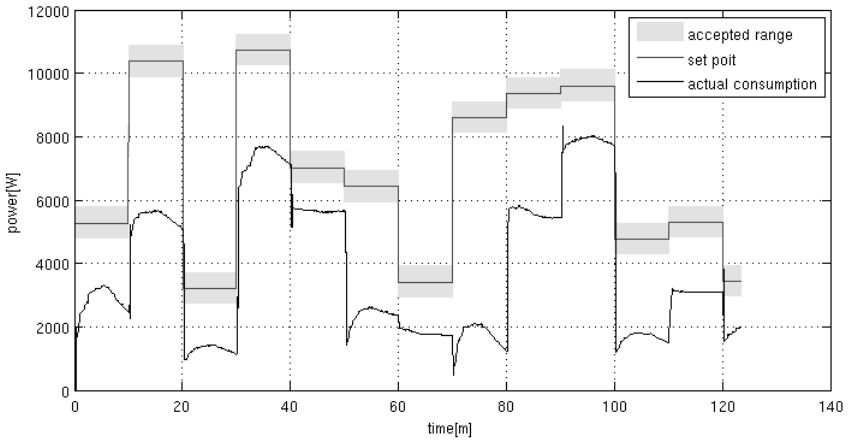

Figure 8. Power consumption over time, calibration experiment.

Figure 9 contains a plot of the deviation between the expected and the actual power consumption. It is clear that the devices do not consume as much power as they requested in the negotiation state. In order to keep the requested setpoint, the overall consumption of the building must be used as a control input.

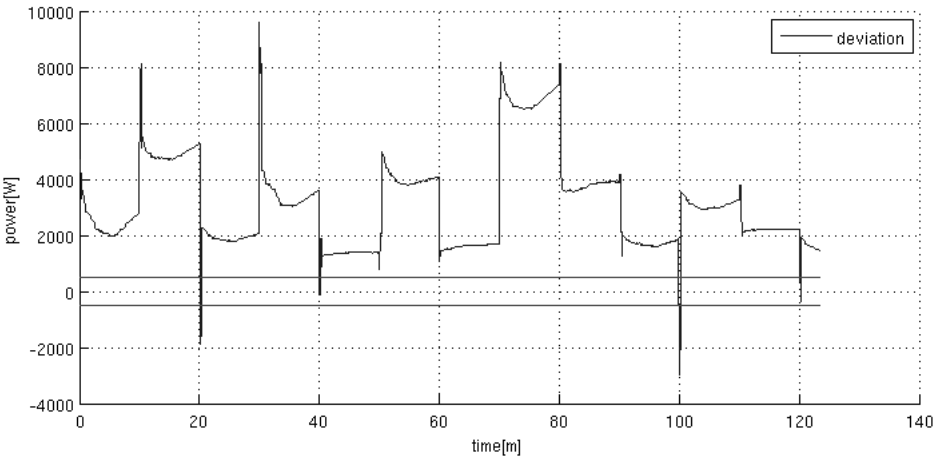

Figure 9. Setpoint deviation, initial experiment.

5.2.2. Consumption adaptation

The reason for under- or over-consumption can be expected to be one of the following:

- inaccurate estimation of a device's own power use, as reported in the negotiation bid,
- uncontrollable devices, either of type base load or spontaneous use,
- devices changing their type from controllable to uncontrollable after the negotiation,
- non-constant power consumption of particular types of equipment.

In the experimental setup, the Smart meter does not have any knowledge of the detailed behavior of individual devices, and only their aggregated consumption can be measured. Therefore the Smart meter is unable to determine the reason for the deviation.

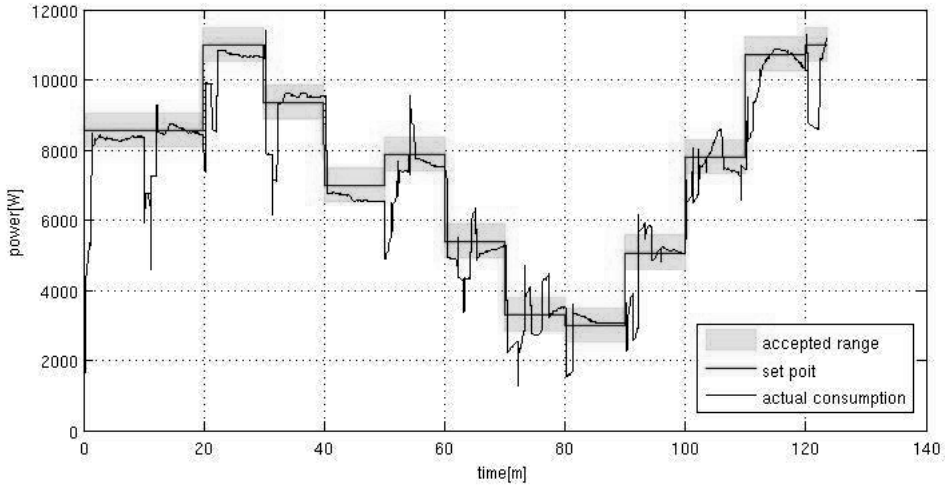

Figure 10. Power consumption over time, first experiment.

Every minute, the Smart meter agent determines the difference between the expected and measured overall power consumption. Based on the sign of the error, it then overlays a corrective value on top of the external setpoint and broadcasts the combined setpoint in order to trigger a new negotiation. As shown in figure 10 the tracking error is reduced from the calibration experiment. The performance of the control system is shown in terms of setpoint deviation (figure 11).

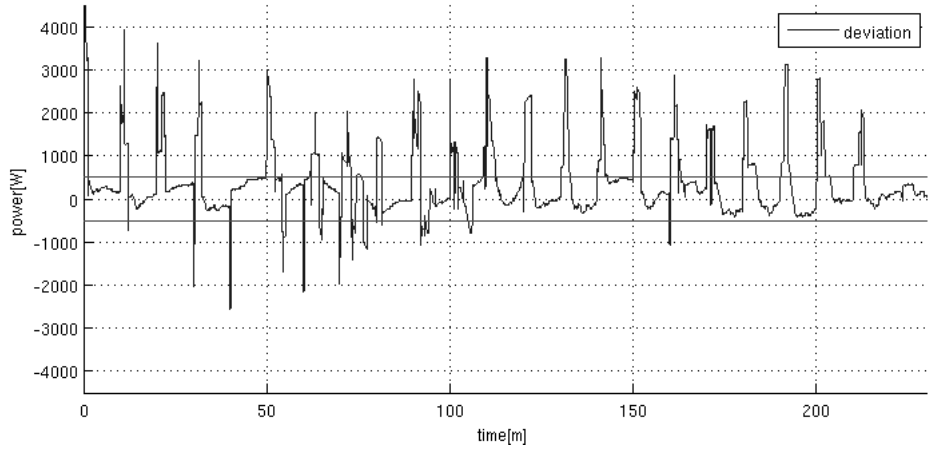

Figure 11. Setpoint deviation, first experiment.

As observed in figure 11, the setpoint deviation is usually positive. This might indicate that some devices report too high power consumption in their negotiation bids. When under-consumption appears, the system needs to adapt to match the requested setpoint as closely as possible. To minimize this problem, the reported power consumption of the individual devices was revised based on the observed setpoint errors.

5.2.3. Reducing spikes in power consumption

In figure 10, spikes can be observed at regular intervals. They may occur for two reasons: Firstly, air conditioning units based on inverter fancoils exhibit a much more complex power consumption pattern than the simple heating resistors in the space heaters. At startup, the units have a low power consumption which then gradually increases. If the control algorithm causes load to be switched from a heater to an air-conditioning unit, the downramping of the heater unit is near-instantaneous while the upramping AC unit will not respond equally quick. The second reason for the appearance of spikes - observable in the plot between minutes 130 and 150 - may be caused by the different cycles of random setpoint generation and adaptive correction in the smart meter agent.

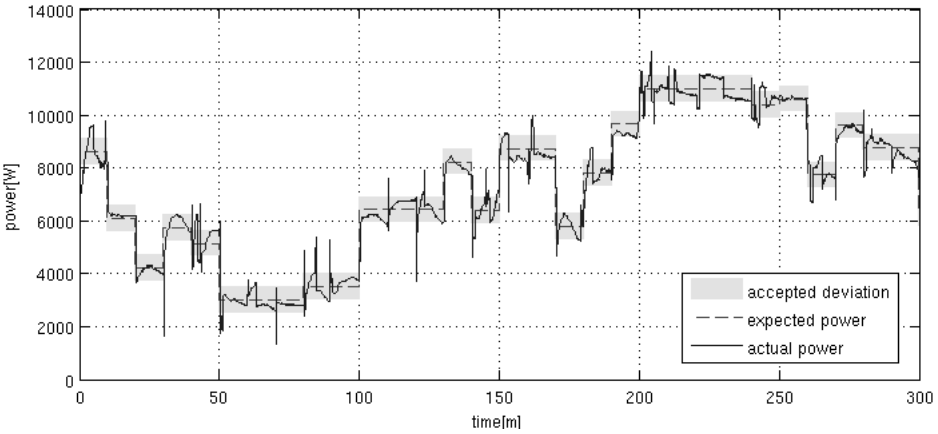

Figure 12. Power consumption over time, second experiment.

For the second experiment, the control algorithm was modified to reduce the occurrence of spikes, and the power consumption values reported by the core agents were tweaked. The results are shown in figures 12 and 13. Spikes are less prominent, and the residual setpoint error has much improved symmetry, bringing its integral closer to zero, therefore the setpoint is followed closely. Spikes in power consumption caused by device switching are still present

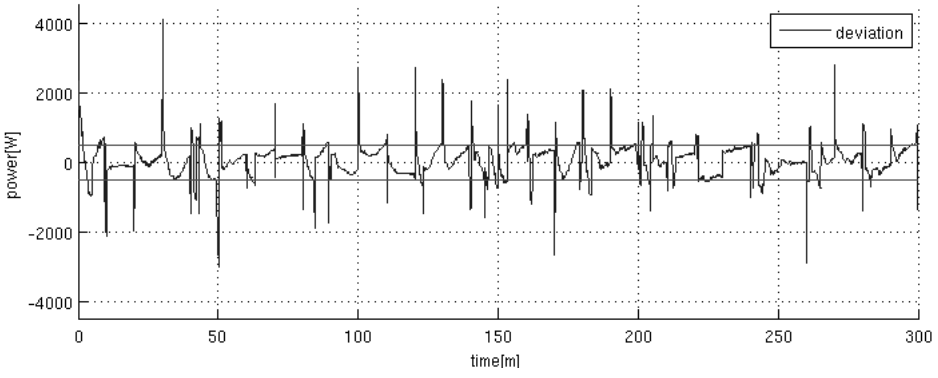

Figure 13. Setpoint deviation, second experiment.

in the overall building consumption, but appear less regularly. Because they are dependent on individual device characteristics and the switching sequence emerging from the negotiations, it is unlikely that they can be entirely avoided. Figure 14 shows the temperatures in different rooms of the house during the experiment. It can be observed that, for some period of time, the temperature in individual rooms drifts outside the desired range. This behavior is expected, since the control system has been designed for the primary goal of tracking the

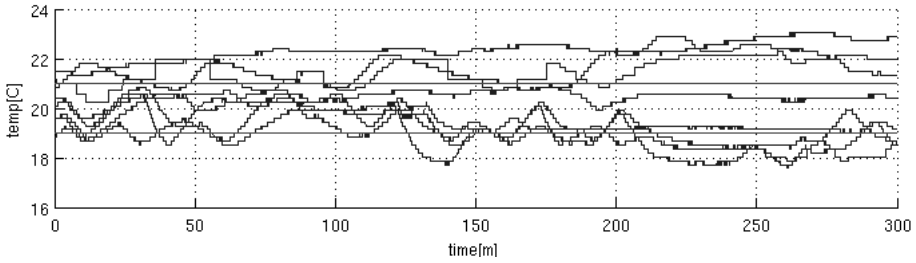

Figure 14. Room temperatures, second experiment.

power consumption setpoint. As a secondary goal, the core agents try to keep the indoor temperature within agreed bounds, but this goal may be overridden by the manager agent's decisions. Between minutes 200 and 300 the temperature can be seen to increase in individual rooms, while it decreases in other rooms beyond the comfort range. In this time period, the house was requested to consume the maximum amount of 11 kW. Therefore, even though many core agents reported very low urgency factors, the manager agent requested them to operate. This is expected behavior and a logical consequence of allowing external setpoints near the maximum combined rated power of all units.

6. Future work and conclusions

In this paper, we have presented the design and architecture of a multi-agent-based, distributed control system for controlling loads in residential buildings. System design considerations were explained and justified, and the distributed multi-agent software architecture was described in detail. Behavior and roles of the individual agents in the system were discussed, and experimental data from a laboratory implementation was presented. The results provide proof of concept of system functionality.

The three experiments presented in section 5 helped to observe the aggregated response of the system, to improve its behavior and to provide measurable performance criteria. There are more improvements which can be designed and implemented in the proposed distributed multi-agent architecture. One potential area of improvement is the suppression or reduction of spikes in the power consumption, caused by imperfect device scheduling (see section 5.2.2 for a discussion of the problem). Another issue is the remaining single point of failure introduced by closing the overall control loop through the smart meter agent. If it fails, it removes the system's ability to adapt to changes in spontaneous load. In a future implementation, the task of matching actual and desired power consumption could be moved to the currently active manager agent. To do this, measured values of the instantaneous power consumption would have to be distributed to the manager agents. Alternatively, power consumption measurements built into individual devices could be used to estimate the overall consumption in case the central measurement fails.

The negotiation process can also be improved. A mechanism to renegotiate the manger's decision might be included in the next version of the negotiation protocol, for example ensuring that all devices that cannot be immediately turned off, for example a programmed devices like washing machine, are allowed to continue to operate for some period of time.

In the current architecture, a user override is achieved by completely locking the device and removing it from the negotiation process until the user releases the lock. A more elegant way of handling this process could be through a more complex mechanism for changing the device type from spontaneous use to controllable. A time-based automatic release of a user lock could be a useful part of this.

References

[1] Danish Energy Agency. Energy Statistics 2010, 2011.

[2] Aly A. Syed, Johan Lukkien, and Roxana Frunza. An ad hoc networking architecture for pervasive systems based on distributed knowledge. In *Proceedings of Date2010, Dresden*, 2010.

[3] A. Kosek, A. Syed, and J. Kerridge. Evaluating an Emergent Behaviour Algorithm in JCSP for Energy Conservation in Lighting Systems. In *Communicating Process Architectures 2011*, 2011.

[4] S. Hagg and F Ygge. Agent-oriented programming in power distribution automation. phd thesis, 1995.

[5] S. Sharples, V. Callaghan, and G. Clarke. A multi-agent architecture for intelligent building sensing and control. *Sensor Review*, 19(2):135–140, 1999.

[6] G. Conte and D. Scaradozzi. Viewing home automation systems as multiple agents systems. *Multi-agent system for industrial and service robotics applications, RoboCUP2003, Padova, Italy*, 2003.

[7] N. Roy, A. Roy, and S.K. Das. Context-aware resource management in multi-inhabitant smart homes a nash h-learning based approach. In *Pervasive Computing and Communications, 2006. PerCom 2006. Fourth Annual IEEE International Conference on*, pages 11–pp. IEEE, 2006.

[8] F. Allerding, M. Premm, P. Shukla, and H. Schmeck. Electrical load management in smart homes using evolutionary algorithms. *Evolutionary Computation in Combinatorial Optimization*, pages 99–110, 2012.

[9] Z. Lin, Z. Guiqing, S. Bin, X. Xiuying, and Y. Qiao. Building energy saving design based on multi-agent system. In *Industrial Electronics and Applications (ICIEA), 2010 the 5th IEEE Conference on*, pages 840–844. IEEE, 2010.

[10] S. Abras, S. Ploix, S. Pesty, and M. Jacomino. A multi-agent design for a home automation system dedicated to power management. *Artificial Intelligence and Innovations 2007: from Theory to Applications*, pages 233–241, 2007.

[11] H. Joumaa, S. Ploix, S. Abras, G. De Oliveira, et al. A mas integrated into home automation system, for the resolution of power management problem in smart homes. *Energy Procedia*, 6:786–794, 2011.

[12] L. Klein, G. Kavulya, F. Jazizadeh, J. Kwak, B. Becerik-Gerber, P. Varakantham, and M. Tambe. Towards optimization of building energy and occupant comfort using multi-agent simulation. In *The 28th International Symposium on Automation and Robotics in Construction (ISARC)(June 2011)*, 2011.

[13] The iPower project. www.ipower.dk.

[14] E.H Mathews, D.C Arndt, C.B Piani, and E van Heerden. Developing cost efficient control strategies to ensure optimal energy use and sufficient indoor comfort. *Applied Energy*, 66(2):135 – 159, 2000.

[15] C. A. R. Hoare. Communicating sequential processes. *Commun. ACM*, 21(8):666–677, 1978.

[16] P. H. Welch and P. D. Austin. The JCSP Home Page. http://www.cs.ukc.ac.uk/projects/ofa/jcsp/ , 1999.

[17] Anna Kosek, Aly Syed, Jon Kerridge, and Alistair Armitage. A dynamic connection capability for pervasive adaptive environments using jcsp. In *The Thirty Fifth Annual Convention of the Society for the Study of Artificial Intelligence and Simulation of Behaviour (AISB'09)*, 2009.

[18] A. Kosek, J. Kerridge, and A. Syed. JCSP Agents-Based Service Discovery for Pervasive Computing. In *Communicating Process Architectures 2009*, sep 2009.

[19] O. Gehrke and H. Bindner. Building a test platform for agents in power system control: Experience from SYSLAB. In *Intelligent Systems Applications to Power (ISAP) 2007*, Sep 2007.